UNIVERZITA PALACKÉHO V OLOMOUCI
PŘÍRODOVĚDECKÁ FAKULTA

BAKALÁŘSKÁ PRÁCE

Metoda nejmenších čtverců v matematickém softwaru R.

Katedra matematické analýzy a aplikací matematiky
Vedoucí bakalářské práce: **RNDr. Jitka Machalová, Ph.D.**
Vypracovala: **Markéta Coufalová**
Studijní program: B1103 Aplikovaná matematika
Studijní obor: Matematika–ekonomie se zaměřením na bankovnictví/pojišťovnictví
Forma studia: prezenční
Rok odevzdání: 2017

BIBLIOGRAFICKÁ IDENTIFIKACE

Autor: Markéta Coufalová

Název práce: Metoda nejmenších čtverců v matematickém softwaru R.

Typ práce: Bakalářská práce

Pracoviště: Katedra matematické analýzy a aplikací matematiky

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

Rok obhajoby práce: 2017

Abstrakt: Bakalářská práce Metoda nejmenších čtverců v matematickém softwaru R se zaměřuje na možnosti aproximace funkcí ve smyslu metody nejmenších čtverců. V teoretické části jsou tyto možnosti aproximace popsány a doplněny o vlastní příklady. Dále jsou zde sestaveny vlastní programy v softwaru R, které jsou následně ukázány na konkrétních úlohách.

Klíčová slova: Metoda nejmenších čtverců, aproximace, polynomy, Gramovy polynomy, trigonometrické polynomy, aproximace nelineárního typu, matematický software R

Počet stran: 47

Počet příloh: 1

Jazyk: český

BIBLIOGRAPHICAL IDENTIFICATION

Author: Markéta Coufalová

Title: The least square approximation in software R.

Type of thesis: Bachelor's

Department: Department of Mathematical Analysis and Application of Mathematics

Supervisor: RNDr. Jitka Machalová, Ph.D.

The year of presentation: 2017

Abstract: Bachelor's The least square approximation in software R focuses on possibilities of approximation of functions in sense of least squares method. These possibilities of approximation are described and supplemented by own examples in theoretical section. Further own programs in software R are compiled here which are shown then on specific assignment.

Key words: Least squares method, approximation, polynomials, Gram polynomials, trigonometric polynomials, nonlinear approximation, mathematical software R

Number of pages: 47

Number of appendices: 1

Language: Czech

Prohlášení

Prohlašuji, že jsem bakalářskou práci zpracovala samostatně pod vedením paní RNDr. Jitky Machalové, Ph.D. a všechny použité zdroje jsem uvedla v seznamu literatury.

V Olomouci dne ....

podpis

Poděkování

Na tomto místě bych ráda poděkovala vedoucí mé bakalářské práce RNDr. Jitce Machalové, Ph.D. za odborné vedení, trpělivost a cenné rady při zpracovávání této práce. Děkuji také Mgr. Ondřeji Vencálkovi, Ph.D. za pomoc při práci v softwaru R.

Obsah

Použité symboly

$\mathbb{R}$	množina reálných čísel
$\mathbb{N}$	množina přirozených čísel
$\mathbb{N}_0$	$\mathbb{N} \cup \{0\}$
x_i, x_k	body aproximace
$f(x_i), f(x_k)$	funkční hodnoty v bodech aproximace
w_i, w_k	váhy příslušné k bodům x_i, x_k
$\tilde{f}(x)$	funkce aproximace
$\tilde{g}(x)$	transformace funkce $\tilde{f}(x)$
$G_j(x)$	Gramův polynom stupně j

Úvod

Aproximační metody jsou významnou součástí numerické matematiky. Můžeme se setkat s různými metodami. Tato práce se zaměřuje na jednu z nich, a to na metodu nejmenších čtverců na diskrétní množině bodů.

Cílem této bakalářské práce je nastudovat a představit možnosti aproximace funkcí ve smyslu metody nejmenších čtverců pro diskrétní data a sestavit vlastní programy v matematickém softwaru R pro konkrétní typy aproximace.

Celá práce je rozdělena do čtyř kapitol. První kapitola krátce popisuje rozdíl mezi metodou nejmenších čtverců a interpolací, a zdůvodňuje, proč se přikláníme právě k aproximaci pomocí metody nejmenších čtverců. Dále se věnuje principu metody nejmenších čtverců. Jsou zde také definovány pojmy, jako je nejlepší aproximace ve smyslu metody nejmenších čtverců, normální soustava a Gramův determinant. Druhá kapitola pojednává o ortogonálních systémech funkcí a speciálně pak o aproximaci Gramovými polynomy. Další, třetí kapitola, rozebírá aproximaci dat s periodickým charakterem s využitím trigonometrickým polynomů. Ve čtvrté, poslední kapitole, je popsána nelineární aproximace ve smyslu metody nejmenších čtverců a tzv. linearizace.

Jednotlivé kapitoly jsou doplněny řešenými, ručně počítanými příklady na daný tvar aproximace. Po nich následují mnou vytvořené programy v matematickém softwaru R. Předtím než jsou programy vloženy, je popsána jejich práce s daty. Uvedené R Scripty jsou poté předvedeny na již v práci řešených příkladech, aby bylo možné porovnat ručně vypočítané koeficienty s koeficienty vygenerovanými sestavenými kódy. Každá kapitola obsahuje také grafické znázornění nejlepší aproximace ve smyslu metody nejmenších čtverců příslušné k jednotlivým řešeným příkladům.

1. Metoda nejmenších čtverců

Metoda nejmenších čtverců patří mezi tzv. aproximační metody. Mezi tyto metody se řadí také interpolace, ale od ní se v situacích, kdy máme vysoký počet dat nebo data zatížená chybami upouští a využívá se právě aproximace pomocí metody nejmenších čtverců.

Jednou z výhod metody nejmenších čtverců je, že nepožadujeme, aby aproximační funkce procházela všemi body aproximace. Zatímco interpolační funkce musí přesně procházet všemi známými body.

V aproximačních metodách vycházíme buď z diskrétně zadaných dat, nebo z explicitního vyjádření funkce. V této práci se budeme zabývat jen aproximací diskrétního typu.

1.1. Princip metody nejmenších čtverců

V této kapitole se podíváme, jak se s metodou nejmenších čtverců obecně pracuje a uvedeme několik definic a vět, které je potřeba znát. Využili jsme především zdrojů [2], [4] a [6].

Předpokládejme, že máme zadané funkční hodnoty $f(x_i)$ na diskrétní množině bodů x_i, $i = 0, 1, \ldots, n, n \in \mathbb{N}_0$, kde $x_i \neq x_k$ pro $i \neq k$. Dále nechť jsou dány váhy $w_i > 0, i = 0, 1, \ldots, n$. Zvolíme si funkce $\varphi_0(x), \varphi_1(x), \ldots, \varphi_m(x)$, přičemž $n > m$ (v případě, že $n \leq m$ se jedná o úlohu interpolace). Naším úkolem je potom najít funkci

$$\tilde{f}(x) = \sum_{j=0}^{m} c_j \varphi_j(x). \tag{1}$$

tak, aby funkce $\tilde{f}$ minimalizovala „souhrnnou odchylku“ definovanou vztahem

$$\varrho^2(f, \tilde{f}) = \sum_{i=0}^{n} w_i [f(x_i) - \tilde{f}(x_i)]^2 = \sum_{i=0}^{n} w_i [f(x_i) - \sum_{j=0}^{m} c_j \varphi_j(x)]^2. \tag{2}$$

Funkci $\tilde{f}(x)$, pro kterou tato veličina nabývá své minimální hodnoty, nazýváme **nejlepší aproximací funkce $f(x)$ ve smyslu metody nejmenších čtverců** na množině dat x_i, $i = 0, 1, \ldots, n$. Tzv. vahová funkce $w_i, w_i > 0, i = 0, 1, \ldots, n$, nám umožňuje zdůraznit vliv resp. přesnost jednotlivých naměřených hodnot. V případě, kdy nemáme tyto informace k dispozici, nebo jsou data stejně významná, volíme $w_i \equiv 1, \forall i = 0, 1, \ldots, n$.

Funkce ϱ^2 je funkce proměnných $c_0, c_1, \ldots, c_m$, značíme $\varrho^2(c_0, c_1, \ldots, c_m)$. Nutnou podmínkou, aby vektor $(c_0, c_1, \ldots, c_m)$ byl stacionárním bodem funkce ϱ^2, je splnění rovnic

$$\frac{\partial \varrho^2(c_0, c_1, \ldots, c_m)}{\partial c_k} = 0, \quad \forall k = 0, 1, \ldots, m.$$

Stacionární bod hledáme proto, protože je nutné najít body, ve kterých funkce nabývá svých minimálních hodnot. Tedy spočítáme parciální derivace podle proměnných c_k pro $\forall k = 0, 1, \ldots, m$, položíme je rovny nule a dostaneme rovnici tvaru

$$2 \sum_{i=0}^{n} w_i [f(x_i) - \sum_{j=0}^{m} c_j \varphi_j(x_i)] \varphi_k(x_i) = 0.$$

Po jednoduchých úpravách rovnici zapíšeme

$$\sum_{i=0}^{n} w_i \sum_{j=0}^{m} c_j \varphi_j(x_i) \varphi_k(x_i) = \sum_{i=0}^{n} w_i f(x_i) \varphi_k(x_i), \quad \forall k = 0, 1, \ldots, m.$$

Odtud

$$\sum_{j=0}^{m} c_j \sum_{i=0}^{n} w_i \varphi_j(x_i) \varphi_k(x_i) = \sum_{i=0}^{n} w_i f(x_i) \varphi_k(x_i), \quad \forall k = 0, 1, \ldots, m. \tag{3}$$

Výhodnější tvar vztahu (3) získáme zadefinováním diskrétního skalárního součinu funkcí $f(x)$ a $g(x)$ společně s vahovou funkcí w_i

$$(f, g) = \sum_{i=0}^{n} w_i f(x_i) g(x_i), \tag{4}$$

potom můžeme rovnici (3) zapsat následovně

$$\sum_{j=0}^{m} c_j (\varphi_j, \varphi_k) = (f, \varphi_k), \quad \forall k = 0, 1, \ldots, m. \tag{5}$$

Tedy obdržíme soustavu $m+1$ lineárních rovnic o $m+1$ hledaných neznámých c_j

$$\begin{aligned} c_0(\varphi_0,\varphi_0)+c_1(\varphi_1,\varphi_0)+\ldots+c_m(\varphi_m,\varphi_0) &= (f,\varphi_0) \\ c_0(\varphi_0,\varphi_1)+c_1(\varphi_1,\varphi_1)+\ldots+c_m(\varphi_m,\varphi_1) &= (f,\varphi_1) \\ &\vdots \\ c_0(\varphi_0,\varphi_m)+c_1(\varphi_1,\varphi_m)+\ldots+c_m(\varphi_m,\varphi_m) &= (f,\varphi_m) \end{aligned} \tag{6}$$

Definice 1.1 *Soustavu lineárních rovnic (6) nazveme normální soustavou a její determinant budeme nazývat Gramův determinant.*

Věta 1.1 *Gramův determinant je různý od nuly právě tehdy, když funkce $\varphi_j(x), j = 0,1,\ldots,m$ jsou lineárně nezávislé na množině bodů $x_i, i = 0,1,\ldots,n$.*

Důkaz: Viz. literatura [1], str. 390, věta 28.4. (ii).

Řešením normální soustavy získáme koeficienty $c_0, c_1, \ldots, c_m$ nejlepší aproximace $\tilde{f}(x)$ funkce $f(x)$ ve smyslu metody nejmenších čtverců. Jejich dosazením do vztahu (1) obdržíme funkci $\tilde{f}(x)$. Na tomto místě je nutné se zabývat existencí a jednoznačností řešení normální soustavy.

Věta 1.2 *Je-li systém funkcí $\varphi_j(x), j = 0,1,\ldots,m$ lineárně nezávislý na množině bodů $x_i, i = 0,1,\ldots,n$, pak ke každé funkci f existuje jediná nejlepší aproximace $\tilde{f}(x) = \sum_{j=0}^{m} c_j\varphi_j(x)$ ve smyslu metody nejmenších čtverců. Její koeficienty $c_0, c_1, \ldots, c_m$ jsou jediným řešením normální soustavy.*

Důkaz: Důkaz můžeme najít v [1], str. 156, důsledek 13.5.

Pokud za funkce $\varphi_0(x), \varphi_1(x), \ldots, \varphi_m(x)$ vezmeme mocniny x, budeme daná data aproximovat pomocí polynomů $\varphi_0(x) = 1, \varphi_1(x) = x, \ldots, \varphi_m(x) = x^m$. Tedy hledáme funkci $\tilde{f}(x)$ ve tvaru polynomu. To si ukážeme v následujících příkladech.

Příklad 1.1 Data zadaná v tabulce aproximujte polynomem prvního stupně. Uvažujte $w_i = 1, i = 0, \dots, 7$.

x_i	-3	-2	-1	0	1	3	4	5
$f(x_i)$	-27	-20	-10	-2	1	3	8	12

Položíme-li za $\varphi_0 = 1, \varphi_1 = x$, hledaný polynom bude mít tvar

$$\tilde{f}(x) = c_0 + c_1 x.$$

Nejprve vypočítáme potřebné skalární součiny

$$(\varphi_0, \varphi_0) = \sum_{i=0}^{7} 1 = 8$$

$$(\varphi_0, \varphi_1) = (\varphi_1, \varphi_0) = \sum_{i=0}^{7} x_i = 7$$

$$(\varphi_1, \varphi_1) = \sum_{i=0}^{7} x_i^2 = 65$$

$$(\varphi_0, f) = \sum_{i=0}^{7} f(x_i) = -35$$

$$(\varphi_1, f) = \sum_{i=0}^{7} x_i f(x_i) = 233.$$

Po dosazení těchto skalárních součinů získáme normální soustavu rovnic ve tvaru

$$\begin{aligned} 8c_0 + 7c_1 &= -35 \\ 7c_0 + 65c_1 &= 233. \end{aligned}$$

Řešením této soustavy jsou hledané koeficienty

$$c_0 = -8.293, \quad c_1 = 4.478.$$

Nejlepší aproximací funkce dané tabulkou ve smyslu metody nejmenších čtverců ve třídě všech polynomů nejvýše prvního stupně je polynom

$$\tilde{f}(x) = -8.293 + 4.478x.$$

Grafické znázornění této aproximace najdeme na Obrázku 1.

Příklad 1.2 Mějte stejná data jako v Příkladu 1.1, ty aproximujte opět polynomem prvního stupně, avšak $w_0 = 2, w_1 = 3, w_2 = 7, w_3 = 4, w_4 = 6, w_5 = 1,$ $w_6 = 2, w_7 = 1.$
Skalární součiny budou rovny

$$(\varphi_0, \varphi_0) = \sum_{i=0}^{7} 1 = 26$$

$$(\varphi_0, \varphi_1) = (\varphi_1, \varphi_0) = \sum_{i=0}^{7} x_i = 3$$

$$(\varphi_1, \varphi_1) = \sum_{i=0}^{7} x_i^2 = 109$$

$$(\varphi_0, f) = \sum_{i=0}^{7} f(x_i) = -155$$

$$(\varphi_1, f) = \sum_{i=0}^{7} x_i f(x_i) = 491.$$

Po dosazení

$$\begin{aligned} 26c_0 + 3c_1 &= -155 \\ 3c_0 + 109c_1 &= 491 \end{aligned}$$

a vyřešením soustavy získáme koeficienty

$$c_0 = -6.502, \quad c_1 = 4.684.$$

Nejlepší aproximací funkce ve smyslu metody nejmenších čtverců ve třídě všech polynomů nejvýše prvního stupně je potom polynom

$$\tilde{f}(x) = -6.502 + 4.684x.$$

Pro porovnání graf viz. Obrázek 2, na kterém vidíme, jak hodnoty w_i ovlivňují průběh aproximace.

1.2. R Script

Pro výpočet nejlepší aproximace ve smyslu metody nejmenších čtverců, kdy uvažujeme aproximaci ve tvaru lineární kombinace funkcí $\varphi_0(x) = 1, \varphi_1(x) = x, \ldots, \varphi_m(x) = x^m$, jsem v programu R vytvořila R Script s názvem MNC-funkce.R. Jako vstupní data jsem si navolila vektor bodů x, vektor funkčních hodnot f v bodech x, vektor vah w příslušný k bodům x a stupeň polynomu N, kterým chceme zadaná data aproximovat. Přičemž pokud nastane situace, že jeden z vektorů má jinou délku než další dva, body vektoru x nejsou různé, body vektoru w nejsou kladné či stupeň polynomu N není celé kladné číslo, nás program upozorní a neprovede zavedené výpočty. Následným výstupem jsou potom koeficienty $c_0, c_1, \ldots, c_m$ hledaného polynomu, který je nejlepší aproximací ve smyslu metody nejmenších čtverců a graf, který nám vykreslí aproximované hodnoty proložené polynomem zvoleného stupně.

```
##Data##
x = c()
f = c()
w = c()
N =

MNC = function(x,f,w,N) {
  n = length(x)
  k = length(f)
  l = length(w)

  ##Ověření stupně polynomu##
  if(N < 0){
    stop("Stupeň polynomu není kladné číslo!")
  }
  if(N != floor(N)){
```

```
  stop("Stupeň polynomu není celé číslo!")
}
##Ověření různosti bodů vektoru x##
for(i in 1:n){
  for(j in 1:n){
    if(i != j & x[i] == x[j]){
      stop("Body vektoru x nejsou různé!")
    }
  }
}
##Ověření kladnosti vektoru w##
for(i in 1:l){
  if(w[i] < 0){
    stop("Váhy nejsou kladné!")
  }
}
##Ověření délky vektorů##
if(n != k){
  stop("Délka vektoru x a vektoru f není stejná!")
}
if(k != l){
  stop("Délka vektoru f a vektoru w není stejná!")
}
if(n != l){
  stop("Délka vektoru x a vektoru w není stejná!")
}

##Funkce##
A = matrix(nrow = (N+1), ncol = n)
for (i  in 0:(N+1)) {
```

```
    A[i,] = x^(i-1)
  }
  ##Skalární součiny##
  B = matrix(nrow = (N+1), ncol = (N+1))
  for(i in 0:(N+1)) {
    for(j in 0:(N+1)) {
      B[i,j] = sum(A[i,] * A[j,] * w)
      B[j,i] = sum(A[i,] * A[j,] * w)
    }
  }
  b = c()
  for(i in 0:(N+1)) {
    b[i] = sum(A[i,] * f * w)
  }

  ##Koeficienty##
  reseni = solve(B) %*% b

  ##Vykreslení##
  t = seq(from = min(x)-1, to = max(x)+1, by = 0.1)
  vyr.hod = rep(0,length(t))
  for (i in 1:(N+1)) {
    vyr.hod = vyr.hod + reseni[i] * (t^(i-1))
  }
  plot(f~x)
  lines(x = t , y = vyr.hod, col = "pink", lwd = 3)

  return(reseni)
}
```

Příklad 1.3 Nyní předvedeme RScript MNCfunkce.R na Příkladu 1.1. Poté co vypíšeme hodnoty vektorů a stupeň polynomu

```
x = c(-3,-2,-1,0,1,3,4,5)
f = c(-27,-20,-10,-2,1,3,8,12)
w = c(1,1,1,1,1,1,1,1)
N = 1
```

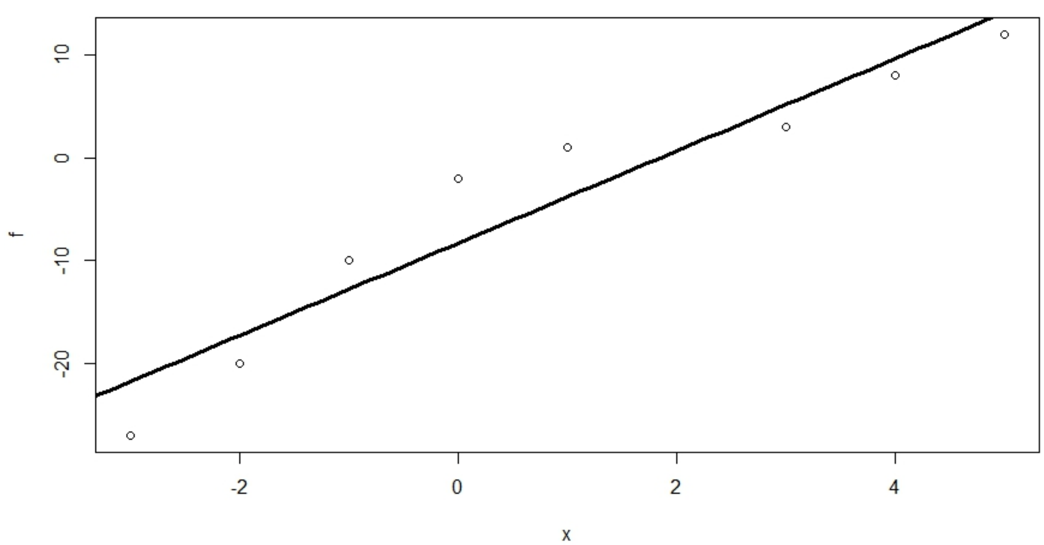
zavoláme funkci

```
MNC(x,f,w,N)
```

která nám vygeneruje vektor koeficientů nejlepší aproximace

```
-8.293 4.478
```

Tyto koeficienty jsou shodné s koeficienty v ruce počítaném Příkladu 1.1. Zároveň funkce vykreslí graf viz. Obrázek 1 této nejlepší aproximace.



Obrázek 1: Lineární aproximace dat z Příkladu 1.1 polynomem prvního stupně.

Příklad 1.4 RScript MNCfunkce.R použijeme opět na data z Příkladu 1.1, ale navíc vezmeme hodnoty vah uvedené v Příkladu 1.2. Vypíšeme vstupní data

```
x = c(-3,-2,-1,0,1,3,4,5)
f = c(-27,-20,-10,-2,1,3,8,12)
w = c(2,3,7,4,6,1,2,1)
N = 1
```

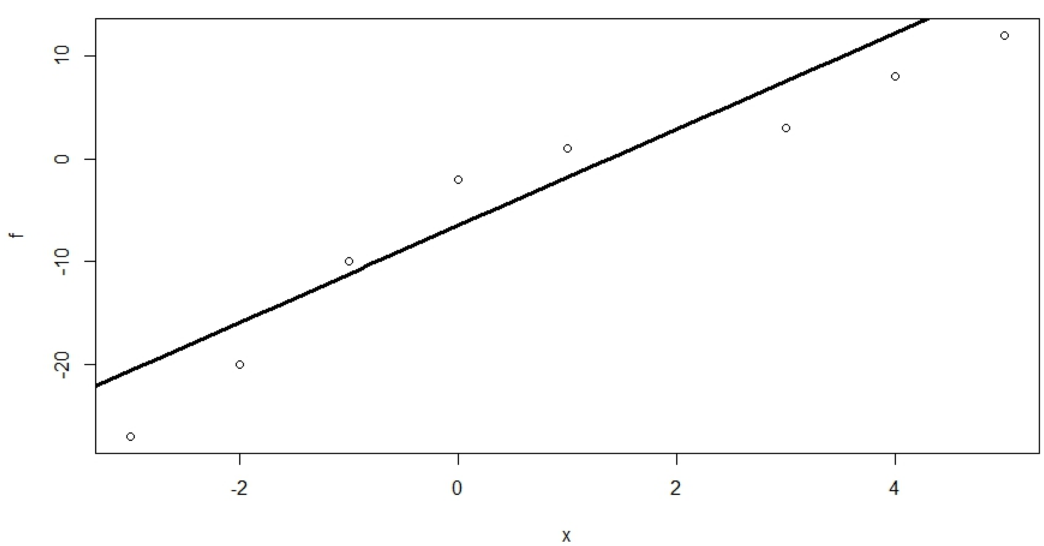
a následně zavoláme funkci

```
MNC(x,f,w,N)
```

Ta v pozadí provede potřebné výpočty, výstupem je pak vektor koeficientů nejlepší aproximace

```
-6.502 4.684
```

který se shoduje s koeficienty vypočítanými ručně v Příkladu 1.2. Vykreslení této aproximace vidíme na Obrázku 2.



Obrázek 2: Lineární aproximace dat z Příkladu 1.1 s váhami z Příkladu 1.2.

2. Ortogonální systémy funkcí

V této kapitole se seznámíme s ortogonálními systémy funkcí, které jsou v aproximačních metodách velmi užitečné, protože umožňují zjednodušit výpočty a zpravidla zajišťují numerickou stabilitu. Čerpali jsme z [2] a [6].

Speciálním případem lineárně nezávislých systémů funkcí jsou systémy funkcí, které jsou navzájem ortogonální. Pojem ortogonálnosti je přitom možné definovat pomocí skalárního součinu uvedeném v (4).

Definice 2.1 *Řekneme, že systém funkcí* $\varphi_0, \varphi_1, \ldots, \varphi_m$ *tvoří ortogonální systém funkcí na množině bodů* $x_i, i = 0, 1, \ldots, n$, *jestliže* $(\varphi_j, \varphi_k) = 0$ *pro* $j \neq k$.

Poznámka 2.1 *Ortogonální systém funkcí je vždy lineárně nezávislý a pojem ortogonality závisí na volbě vah* $w_i, i = 0, 1, \ldots, n$ *a na bodech* $x_i, i = 0, 1, \ldots, n$.

Použitím ortogonálního systému funkcí při hledání nejlepší aproximace ve smyslu metody nejmenších čtverců se soustava normálních rovnic výrazně zjednoduší a bude ve tvaru

$$\begin{aligned} c_0(\varphi_0, \varphi_0) &= (f, \varphi_0) \\ c_1(\varphi_1, \varphi_1) &= (f, \varphi_1) \\ &\vdots \\ c_m(\varphi_m, \varphi_m) &= (f, \varphi_m). \end{aligned}$$

Potom pro řešení této soustavy platí

$$c_j = \frac{(f, \varphi_j)}{(\varphi_j, \varphi_j)}, \quad \forall j = 0, 1, \ldots, m. \tag{7}$$

Za $\varphi_0(x), \varphi_1(x), \ldots, \varphi_m(x)$ lze vzít ortogonální polynomy. Těch je celá řada. Avšak z pohledu diskrétní aproximace uvedeme pouze užití Gramových polynomů (kapitola 2.1.) a speciálním případem, kdy taktéž užíváme ortogonální systémy funkcí, jsou Trigonometrické polynomy (kapitola 3.).

2.1. Aproximace pomocí Gramových polynomů

Gramovy polynomy se řadí do teorie ortogonálních polynomů viz. kapitola 2. Jestliže chceme použít pro aproximaci dat Gramovy polynomy, je nutné aby byly splněny určité podmínky. Takové podmínky, které v případě aproximace funkcemi $\varphi_j, j = 0, 1, \ldots, m$ ve tvaru klasických polynomů nejsou požadovány. Při aproximaci Gramovými polynomy musí být dané body $x_i, i = 0, 1, \ldots, n$ ekvidistantní a váhová funkce $w_i, i = 0, 1, \ldots, n$ příslušná daným bodům nesmí nabývat jiných hodnot než 1.

Zásadní rozdíl mezi aproximací klasickými polynomy ve tvaru $x^j, j = 0, 1, \ldots,$ m a Gramovými polynomy $G_m(x)$ pro $m = 2, 3, \ldots, n$ je vidět při řešení normální soustavy rovnic, která se v případě Gramových polynomů značně zjednoduší. $G_m(x)$ přitom značí Gramův polynom stupně m, který je uvedený v Definici 2.2.

Definice 2.2 *Gramovým polynomem stupně* $m, m \in \mathbb{N}_0$ *na ekvidistantní síti bodů* $x_i = i - \frac{1}{2}n, i = 0, 1, \ldots, n$ *s váhovou funkcí* $w_i = 1 \; \forall i = 0, 1, \ldots, n, n \in \mathbb{N}_0, n > m$ *je polynom definovaný rekurentním vztahem*

$$G_m(x) = xG_{m-1}(x) - \frac{(m-1)^2\left[(n+1)^2 - (m-1)^2\right]}{4(4(m-1)^2 - 1)}G_{m-2}(x), \tag{8}$$

m = 2, 3, ..., n, přičemž

$$G_0(x) \equiv 1, \quad G_1(x) = x.$$

Poznámka 2.2 *Gramovy polynomy, splňující rekurentní vztah (8) uvedený v Definici 2.2, mají koeficient u nejvyšší mocniny roven 1. Nám tato definice postačí, avšak lze najít obecnější definici Gramových polynomů, kdy koeficient u nejvyšší mocniny roven 1 není viz. např. [7].*

Věta 2.1 *Systém Gramových polynomů* $G_j(x), j = 0, 1, \ldots, m$ *je ortogonální na ekvidistantní síti bodů* $x_i = i - \frac{1}{2}n, i = 0, 1, \ldots, n$.

Důkaz: V literatuře [7], na straně 266.

Nutnou podmínkou pro sestrojení Gramových polynomů je již zmíněná ekvidistantnost bodů x_i, tj. $x_{i+1} - x_i = h$ pro všechna $i = 0, 1, \ldots, n-1$, kde h je kladné reálné číslo. Ve většině případech ale pracujeme s body x_i, které nesplňují vztah

$$x_i = i - \frac{1}{2}n, \quad \forall i = 0, 1, \ldots, n. \tag{9}$$

I v této situaci můžeme použít aproximaci pomocí Gramových polynomů, ale nejdříve musíme vytvořit novou síť bodů z_i, která již vztah (9) splňovat bude. To provedeme transformací

$$z_i = \frac{x_i - \bar{x}}{h}, \quad \forall i = 0, 1, \ldots, n,$$

kde $\bar{x}$ je prostřední bod původní sítě. Pro lichý počet bodů je $\bar{x} = x_{n/2}$ a pro sudý počet bodů $\bar{x} = \frac{1}{2}(x_{n+1/2} + x_{n-1/2})$. Na takto vytvořené síti bodů $z_i, i = 0, 1, \ldots, n$ aproximujeme hodnoty $f(x_i)$ funkce f. S využitím Gramových polynomů hledáme nejlepší aproximaci pro novou síť bodů ve tvaru

$$\tilde{f}(z) = \sum_{j=0}^{m} c_j G_j(z).$$

Neznámé koeficienty $c_j, j = 0, 1, \ldots, m$, najdeme již uvedeným postupem v kapitole 1. K původní proměnné x se dostaneme dosazením

$$z = \frac{x - \bar{x}}{h}$$

a získáme funkci

$$\tilde{f}(x) = \sum_{j=0}^{m} c_j G_j \left(\frac{x - \bar{x}}{h} \right).$$

Příklad 2.1 Data zadaná v tabulce aproximujte pomocí Gramových polynomů nejvýše třetího stupně.

x_i	-3	-2	-1	0	1	2	3
$f(x_i)$	8	5	3	1	2	4	7

Tabulkové body x_i splňují podmínku ekvidistance s krokem $h = 1$ i vztah $x_i = i - \frac{1}{2}n$, $\forall i = 0, 1, \ldots, n$. Protože máme $n+1$ bodů, do vztahu (8) dosazujeme $n = 6$. Dostaneme Gramovy polynomy ve tvaru

$$G_0(x) = 1, \quad G_1(x) = x, \quad G_2(x) = x^2 - 4, \quad G_3(x) = x^3 - 7x$$

Potom funkce aproximace vypadá následovně

$$\tilde{f}(x) = c_0 + c_1 x + c_2(x^2 - 4) + c_3(x^3 - 7x).$$

Vypočítáme skalární součiny

$$(G_0, G_0) = \sum_{i=0}^{6} 1 = 7$$

$$(G_1, G_1) = \sum_{i=0}^{6} x_i^2 = 28$$

$$(G_2, G_2) = \sum_{i=0}^{6} (x_i^2 - 4)^2 = 84$$

$$(G_3, G_3) = \sum_{i=0}^{6} (x_i^3 - 7x_i)^2 = 216$$

$$(G_0, f) = \sum_{i=0}^{6} f(x_i) = 30$$

$$(G_1, f) = \sum_{i=0}^{6} x_i f(x_i) = -6$$

$$(G_2, f) = \sum_{i=0}^{6} (x_i^2 - 4) f(x_i) = 56$$

$$(G_3, f) = \sum_{i=0}^{6} (x_i^3 - 7x_i) f(x_i) = 6.$$

Pro zbývající skalární součiny platí $(G_i, G_j) = 0, i \neq j$, což plyne z Věty 2.1. Použitím vztahu (7) získáme koeficienty

$$c_0 = 4.286, \quad c_1 = -0.214, \quad c_2 = 0.667, \quad c_3 = 0.028,$$

pomocí nichž sestrojíme funkci

$$\tilde{f}(x) = 4.286 - 0.214x + 0.667(x^2 - 4) + 0.028(x^3 - 7x).$$

Nejlepší aproximací funkce dané tabulkou ve smyslu metody nejmenších čtverců s využitím Gramových polynomů je ve tvaru

$$\tilde{f}(x) = 0.028x^3 + 0.667x^2 - 0.409x + 1.619$$

Na Obrázku 3 můžeme vidět graf k takto vypočítané funkci $\tilde{f}(x)$.

Příklad 2.2 Data zadaná v tabulce aproximujte pomocí Gramových polynomů nejvýše druhého stupně.

x_i	0.0	0.1	0.2	0.3	0.4	0.5	0.6
$f(x_i)$	4.2	3.7	2.5	2.3	2.7	1.8	1.1

Tabulkové body x_i splňují podmínku ekvidistance s krokem $h = 0.1$, avšak nesplňují vztah $x_i = i - \frac{1}{2}n$, $\forall i = 0, 1, \ldots, 6$. Musíme tedy transformací původních bodů x_i vytvořit novou síť bodů

$$z_i = \frac{x_i - 0.3}{0.1}, \quad \forall i = 0, 1, \ldots, 6.$$

Tedy platí

$$z_0 = -3, \quad z_1 = -2, \quad z_2 = -1, \quad z_3 = 0, \quad z_4 = 1, \quad z_5 = 2, \quad z_6 = 3.$$

Dosazením do rekurentního vztahu (8) pro $n = 6$ dostaneme Gramovy polynomy ve tvaru

$$G_0(z) = 1, \quad G_1(z) = z, \quad G_2(z) = z^2 - 4.$$

Potom funkce aproximace transformovaných hodnot vypadá následovně

$$\tilde{f}(z) = c_0 + c_1 z + c_2(z^2 - 4).$$

Vypočítáme skalární součiny

$$(G_0, G_0) = \sum_{i=0}^{6} 1 = 7$$

$$(G_1, G_1) = \sum_{i=0}^{6} z_i^2 = 28$$

$$(G_2, G_2) = \sum_{i=0}^{6} (z_i^2 - 4)^2 = 84$$

$$(G_0, f) = \sum_{i=0}^{6} f(x_i) = 18.3$$

$$(G_1, f) = \sum_{i=0}^{6} z_i f(x_i) = -12.9$$

$$(G_2, f) = \sum_{i=0}^{6} (z_i^2 - 4) f(x_i) = 1.7.$$

Pro zbývající skalární součiny platí $(G_i, G_j) = 0, i \neq j$, což plyne z Věty 2.1. Použitím vztahu (7) získáme koeficienty

$$c_0 = 2.614, \quad c_1 = -0.461, \quad c_2 = 0.02,$$

pomocí nichž sestrojíme funkci

$$\tilde{f}(z) = 2.614 - 0.461z + 0.02(z^2 - 4).$$

Za z dosadíme původní hodnoty x a získáme aproximaci funkce $f(x)$

$$z = \frac{x - 0.3}{0.1}.$$

Nejlepší aproximací funkce dané tabulkou ve smyslu metody nejmenších čtverců s využitím Gramových polynomů je ve tvaru

$$\begin{aligned}\tilde{f}(x) &= 2.614 - 0.461\left(\frac{x-0.3}{0.1}\right) + 0.02\left[\left(\frac{x-0.3}{0.1}\right)^2 - 4\right] = \\ &= 2.024x^2 - 5.821x + 4.179.\end{aligned}$$

2.2. R Script

Pro hledání aproximační funkce ve tvaru lineární kombinace Gramových polynomů $G_0(x), G_1(x)$ až $G_m(x)$ uvedených v Definici 2.2 jsem vytvořila R Script MNCgramfunkce.R. Vstupem je vektor bodů x, vektor funkčních hodnot f v těchto bodech a nejvyšší stupeň Gramova polynomu N. Vektor vah zde neuvažujeme, protože všechny body, pro které Gramovy polynomy počítáme mají vždy váhy rovné 1. Program nás opět upozorní a přeruší všechny výpočty v případě, že vektory x a f nejsou stejně dlouhé a stupeň polynomu N není celé kladné číslo. Jestliže nastane situace, kdy body vektoru x nejsou ekvidistantní, nastavená transformace v kódu tento problém vyřeší a vypočítá nový vektor z, jehož body již ekvidistantní budou. Výstupem z programu je potom vektor koeficientů $c_0, c_1, \ldots, c_m$ nejlepší aproximace ve smyslu metody nejmenších čtverců pomocí Gramových polynomů a grafické znázornění této nejlepší aproximace v závislosti na aproximovaných hodnotách.

```
##Data##
x = c()
f = c()
N =

Gramp = function(x,f,N) {
  n = length(x)
  m = length(x)-1
```

```
  i = 0:(length(x)-1)
  k = length(f)

##Ověření stupně polynomu##
  if(N < 0){
    stop("Stupeň polynomu není kladné číslo!")
  }
  if(N != floor(N)){
    stop("Stupeň polynomu není celé číslo!")
  }

##Ověření délky vektorů##
  if(n != k){
    stop("Délka vektoru x a vektoru f není stejná!")
  }

##Ověření ekvidistance##
  for (j in 3:n) {
    e = 10^(-6)
    p = abs(x[j-1]-x[j-2])
    r = abs(x[j]-x[j-1])
    if(abs(p - r) > e){
      stop("Body vektoru x nejsou ekvidistantní!")
    }
  }

##Lineární transformace##
  xprum = mean(x)
  h = x[2]-x[1]
  z = (x-xprum) / h
```

```
##Podmínka ortogonálnosti##
  xi = i-((1/2) * m)
  l = length(xi)

  if(xi[l] == x[n]){
    X = x
  } else {
    X = z
  }

##Gramovy polynomy##
  G = matrix(nrow = (N+1), ncol = n)
  for (i  in 3:(N+1)) {
    G[1,] = rep(1,length(X))
    G[2,] = X
    if(N>1){G[i,] = X * G[i-1,] - (((i-2)^2 * ((m+1)^2 - (i-2)^2)) /
(4*(4*(i-2)^2 - 1))) * G[i-2,]}
  }

##Skalární součiny##
  a = c()
  for(i in 0:(N+1)) {
    a[i] = sum(G[i,] * f)
  }

  A = matrix(nrow = (N+1), ncol = (N+1))
  for(i in 0:(N+1)) {
    for(j in 0:(N+1)) {
      A[i,j] = sum(G[i,] * G[j,])
```

```
    }
  }

##Koeficienty##
  c = a / diag(A)

##Vykreslení##
  t1 = seq(from = min(x)-1, to = max(x)+1, by = 0.1)

  t2 = seq(from = min(x), to = max(x), by = 0.01)
  tt = seq(from = min(z), to = max(z), by = 0.1)

  if(xi[l] == x[n]){
    s = t1
  } else {
    s = tt
  }

  T = matrix(nrow = (N+1), ncol = length(s))
  for (i  in 3:(N+1)){
    T[1,]=rep(1, length(s))
    T[2,]=s
    if(N>1){T[i,] = s * T[i-1,] - (((i-2)^2 * ((m+1)^2 - (i-2)^2)) /
(4*(4*(i-2)^2 - 1))) * T[i-2,]}
  }

  vyr.hod=rep(0,length(s))
  for (i in 1:(N+1)) {
    vyr.hod = vyr.hod + c[i] * T[i,]
  }
```

```
  plot(f~x)

  if(xi[l] == x[n]){
    u = t1
  } else {
    u = t2
  }

  lines(x = u, y = vyr.hod, col = "plum", lwd = 3)

##Koeficienty Gramových polynomů##
  K = matrix(nrow = (N+1), ncol = (N+1))
  diag(K) = 1
  K[lower.tri(K)] = 0
  K[upper.tri(K)] = 0
  k[1] = 0
  for(i in 2:N){
    k[i] = (i-1)^2 * ((m+1)^2 - (i-1)^2) / (4*(4*(i-1)^2 - 1))
    if(N>1){K[i+1,i-1] = -k[i-1]-k[i]}
  }

##Zpětná transformace##
  if(h != 1 & xprum != 0){
    a = 1/h
    b = -xprum/h
    A = matrix(data = 0, nrow = N+1, ncol = N+1)
    B = matrix(data = 0, nrow = N+1, ncol = N+1)
    C = matrix(data = 0, nrow = N+1, ncol = N+1)
    for(i in 1:(N+1)){
```

```
      for(j in 1:i){
        A[N+2-i,N+2-j] = a^(j-1) * b^(i-j)
      }
    }
    B[N+1,N+1] = 1
    for(i in N:0){
      B[i,N+1] = 1
      for(j in N:i){
        B[i,j] = B[i+1,j] + B[i+1,j+1]
      }
    }
    C = A * B
    d = c[c((N+1):1)]
    e = c()
    e = d %*% C
  }

##Výsledné koeficienty polynomu##
  P = rep(0, length(N+1))
  for(i in 1:(N+1)){
    P = P + c[i] * K[i,]
  }

  if(h == 1 & xprum == 0){
    print(P)
  } else {
    print(e)
  }
}
```

Příklad 2.3 Použití R Scriptu MNCgramfunkce.R ukážeme na Příkladě 2.1. Z příslušné tabulky opíšeme hodnoty do nadefinovaných vektorů a doplníme požadovaný stupeň polynomu, tedy

```
x = c(-3,-2,-1,0,1,2,3)
f = c(8,5,3,1,2,4,7)
N = 3
```

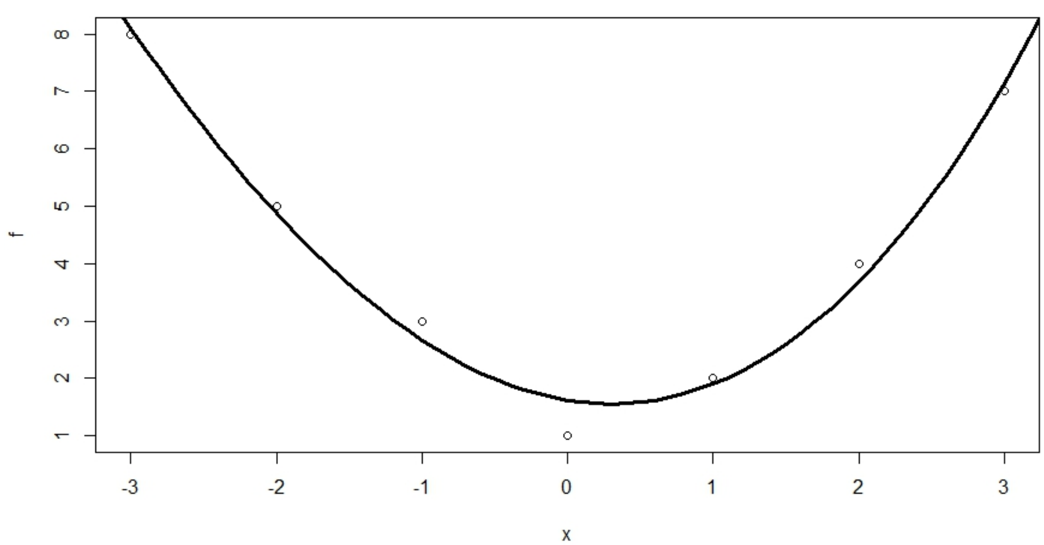
Příkazem

```
Gramp(x,f,N)
```

zavoláme funkci, která nám vypočte tyto koeficienty

```
1.619 -0.409 0.667 0.028
```

shodující se s koeficienty v ruce počítaném Příkladu 2.1. Nejlepší aproximaci s těmito koeficienty vidíme na Obrázku 3.



Obrázek 3: Aproximace dat z Příkladu 2.1 Gramovým polynomem třetího stupně

Příklad 2.4 R Script MNCgramfunkce.R ukážeme také na Příkladě 2.2, kde dané body nesplňují podmínku uvedenou vztahem (9). Doplníme data do příslušných vektorů a stupeň polynomu

```
x = c(0.0,0.1,0.2,0.3,0.4,0.5,0.6)
f = c(4.2,3.7,2.5,2.3,2.7,1.8,1.1)
N = 2
```

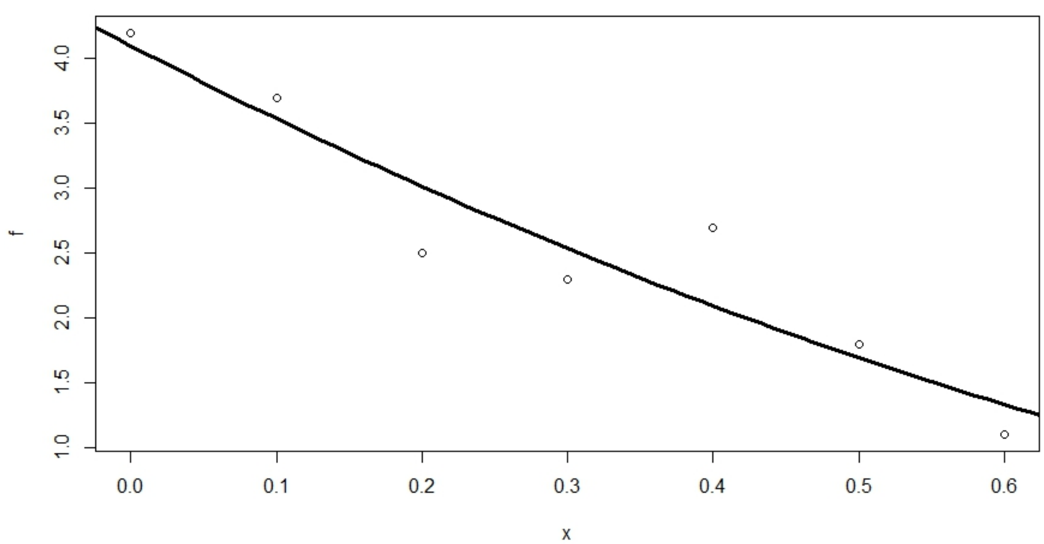
zavoláme funkci příkazem

```
Gramp(x,f,N)
```

a vygenerují se koeficienty aproximace

```
2.024 -5.821 4.179
```

Ty jsou stejné jako v Příkladě 2.2. Vykreslení aproximace viz. Obrázek 4.



Obrázek 4: Aproximace dat z Příkladu 2.2 Gramovým polynomem druhého stupně

3. Aproximace trigonometrickými polynomy

V této části se seznámíme s další možností, jak aproximovat diskrétní data. Budeme tedy hledat funkci $\tilde{f}(x)$ ve tvaru trigonometrických polynomů. Uvedenou teorii jsme získali převážně z literatury [4], [6] a [7].

Aproximaci trigonometrickými polynomy používáme v případě, kdy máme dána data s periodickým charakterem. Předpokládáme, že x_k, $k = 0, 1, \ldots, n$, $x_k \neq x_l$ pro $k \neq l$ jsou ekvidistantní na intervalu $\langle 0, 2\pi \rangle$ a známe funkční hodnoty $f(x_k)$ v daných bodech. Za funkce $\varphi_j(x)$, $j = 0, 1, \ldots, m$ volíme

$$\varphi_0(x) = 1, \varphi_1(x) = \cos x, \varphi_2(x) = \sin x, \varphi_3(x) = \cos 2x, \tag{10}$$
$$\varphi_4(x) = \sin 2x, \ldots, \varphi_{2M-1}(x) = \cos Mx, \varphi_{2M}(x) = \sin Mx, M \in \mathbb{N},$$

přičemž uvažujeme $2M + 1 \leq n + 1$. Dále nechť jsou dány váhy $w_k = 1$, $k = 0, 1, \ldots, n$, protože právě s takto zvolenou váhovou funkcí jsou funkce (10) ortogonální, což je zformulováno v následující větě.

Věta 3.1 *Funkce* $1, cos\ x, sin\ x, cos\ 2x, sin\ 2x, \ldots, cos\ Mx, sin\ Mx, M \in \mathbb{N}$ *tvoří na ekvidistantní síti* $n + 1$ *bodů* $x_k = \frac{2\pi k}{n+1}, k = 0, 1, \ldots, n$ *s váhovou funkcí* $w_k = 1 \quad \forall k = 0, 1, \ldots, n$ *ortogonální systém funkcí pro* $2M + 1 \leq n + 1$*. Tedy platí*

$$\sum_{k=0}^{n} sin\ ix_k sin\ jx_k = \begin{cases} 0 & i \neq j, \\ \frac{n+1}{2} & i = j \neq 0, \\ 0 & i = j = 0, \end{cases} \tag{11}$$

$$\sum_{k=0}^{n} cos\ ix_k cos\ jx_k = \begin{cases} 0 & i \neq j, \\ \frac{n+1}{2} & i = j \neq 0, \\ n+1 & i = j = 0, \end{cases} \tag{12}$$

$$\sum_{k=0}^{n} cos\ ix_k sin\ jx_k = 0, \quad pro\ \forall i, j = 0, 1, \ldots, M. \tag{13}$$

Důkaz: Větu najdeme na str. 284 v literatuře [7]. Návod na provedení důkazu je ve cvičení 28. str. 298.

Nejlepší aproximaci $\tilde{f}(x)$ ve třídě všech polynomů ve smyslu metody nejmenších čtverců hledáme ve tvaru

$$\tilde{f}(x) = \frac{a_0}{2} + \sum_{j=1}^{M} (a_j cos\ jx + b_j sin\ jx) \tag{14}$$

za předpokladu, že máme $n + 1$ bodů, a kde $2M + 1$ udává počet funkcí, kterým chceme daná data aproximovat. Neznámé $a_0, a_j, b_j \in \mathbb{R}$ jsou koeficienty hledaného polynomu.

Poznámka 3.1 *Funkci danou vztahem (14) nazýváme trigonometrickým polynomem.*

Koeficienty $a_0, a_1, \ldots, a_M, b_1, \ldots, b_M$ hledáme tak, aby minimalizovaly „souhrnnou odchylku“ ϱ^2, přičemž ϱ^2 je funkce proměnných $a_0, a_1, \ldots, a_M, b_1, \ldots, b_M$, tedy

$$\varrho^2(a_0, a_1, \ldots, a_M, b_1, b_2, \ldots, b_M) = \sum_{k=0}^{n} [f(x_k) - \widetilde{f}(x_k)]^2.$$

Dále postupujeme jako v případě aproximace pomocí klasických polynomů. Hledáme stacionární bod, kde by funkce ϱ^2 mohla nabývat svého minima. To znamená splnění rovnic

$$\frac{\partial \varrho^2(a_0, a_1, \ldots, a_M)}{\partial a_j} = 0, \quad \forall j = 0, 1, \ldots, M$$

$$\frac{\partial \varrho^2(b_1, \ldots, b_M)}{\partial b_j} = 0, \quad \forall j = 1, \ldots, M.$$

Tvar koeficientů pro lichý počet bodů $x_k, k = 0, 1, \ldots, n$ odvodíme pomocí podmínek ortogonality (11), (12) a (13). Po dosazení funkcí (10) do normální soustavy rovnic (6), kterou získáme analogicky jako v kapitole 1.1., přičemž zavedeme označení $c_0 = \frac{a_0}{2}, c_{2j-1} = a_j, c_{2j} = b_j$, dostaneme koeficienty ve tvaru

$$a_j = \frac{2}{n+1} \sum_{k=0}^{n} f(x_k) cos\ jx_k, \quad j = 0, 1, \ldots, M, \tag{15}$$

$$b_j = \frac{2}{n+1} \sum_{k=0}^{n} f(x_k) sin\ jx_k, \quad j = 1, 2, \ldots, M. \tag{16}$$

Věta 3.2 *Trigonometrický polynom (14) je při daných tabulkových bodech x_k, $k = 0, 1, \ldots, n$, nejlepší diskrétní aproximací funkce $f(x)$ ve třídě všech polynomů o stejném počtu koeficientů.*

Důkaz: Naznačení důkazu lze nalézt v [6], str. 74.

Situaci, kdy počet bodů $x_k, k = 0, 1, \ldots, n$ je sudý, neuvádíme, protože metoda nejmenších čtverců se užívá právě pro lichý počet bodů x_k.

Příklad 3.1 Hodnoty v tabulce aproximujte pomocí trigonometrických polynomů, přičemž $M = 2$, kde $2M + 1$ je počet funkcí. Předpokládáme periodický charakter funkce s periodou 2π.

x_k	0	$2\pi/5$	$4\pi/5$	$6\pi/5$	$8\pi/5$	$10\pi/5$	$12\pi/5$	$14\pi/5$
$f(x_k)$	0.5	1.7	3.9	2.1	1.4	2.2	2.9	4.5

Hledaná aproximace je ve tvaru

$$\tilde{f}(x) = \frac{a_0}{2} + a_1 cos\ x + b_1 sin\ x + a_2 cos\ 2x + b_2 sin\ 2x$$

Koeficienty a_0, a_1, a_2, b_1, b_2 vypočítáme pomocí vztahů (16) a (17), potom

$$a_0 = 4.800, a_1 = -0.985, a_2 = 1.687, b_1 = 0.273, b_2 = -1.028$$

Výsledkem je nejlepší aproximace ve smyslu metody nejmenších čtverců ve tvaru lineární kombinace funkcí $1, sin\ x, cos\ x, sin\ 2x, cos\ 2x$

$$\tilde{f}(x) = \frac{4.800}{2} - 0.985 cos\ x + 0.273 sin\ x + 1.687 cos\ 2x - 1.028 sin\ 2x.$$

Vykreslení této aproximace lze vidět na Obrázku 5.

3.1. R Script

Pro nalezení nejlepší aproximace ve smyslu metody nejmenších čtverců ve tvaru lineární kombinace trigonometrických polynomů (10) jsem vytvořila R Script MNCtrigfunkce.R. Jako vstupní data jsem si zvolila vektor bodů x, vektor funkčních hodnot f v daných bodech a parametr M, kde $2M+1$ je počet funkcí, kterým chceme data aproximovat. Vytvořený kód nám napíše chybu, pokud délka vektoru x a f není shodná, body vektoru x nejsou ekvidistantní, taktéž pokud parametr M není kladné celé číslo. Následným výstupem jsou koeficienty $a_0, a_1, \ldots, a_M, b_1, \ldots, b_M$ v tomto pořadí a vykreslení nejlepší aproximace ve smyslu metody nejmenších čtverců ve tvaru trigonometrických polynomů.

```
##Data##
x=c()
f=c()
M=

MNCtg = function(x,f,M) {
  n=length(x)
  k=length(f)
  h=x[2]-x[1]

  ##Ověření počtu funkcí##
  if(M < 0){
    stop("Počet funkcí není kladné číslo!")
  }
  if(M != floor(M)){
    stop("Počet funkcí není celé číslo!")
  }
  ##Ověření délky vektorů##
  if(n != k){
```

```
    stop("Délka vektoru x a vektoru f není stejná!")
  }
  ##Ověření ekvidistance##
  for (j in 3:n) {
    e=10^(-6)
    p=abs(x[j-1]-x[j-2])
    r=abs(x[j]-x[j-1])
    if(abs(p - r) > e){
      stop("Body vektoru x nejsou ekvidistantní!")
    }
  }

  ##Koeficienty##
  c=c()
  c[1]=2/n*(sum(f))
  for(j in 1:M){
    c[2*j+1]=2/n*(sum(f*sin(j*x)))
    c[2*j]=2/n*(sum(f*cos(j*x)))
  }

  ##Vykreslení##
  t=seq(from = min(x)-(2*pi), to = max(x)+2*pi, by = 0.1)
vyr.hod=rep(0,length(t)) + c[1]/2
  for(j in 1:M){
  vyr.hod = vyr.hod + c[2*j]*cos(j*t) + c[2*j+1]*sin(j*t)
  }
plot(f~x)
  lines(x = t, y = vyr.hod, col = "mediumvioletred", lwd = 3)

  return(c)}
```

Příklad 3.2 R Script MNCtrigfunkce.R předvedeme na datech z Příkladu 3.1. Vypíšeme potřebná data

```
x = c(0,2*pi/5,4*pi/5,6*pi/5,8*pi/5,10*pi/5,12*pi/5,14*pi/5)
f = c(0.5,1.7,3.9,2.1,1.4,2.2,2.9,4.5)
M = 2
```

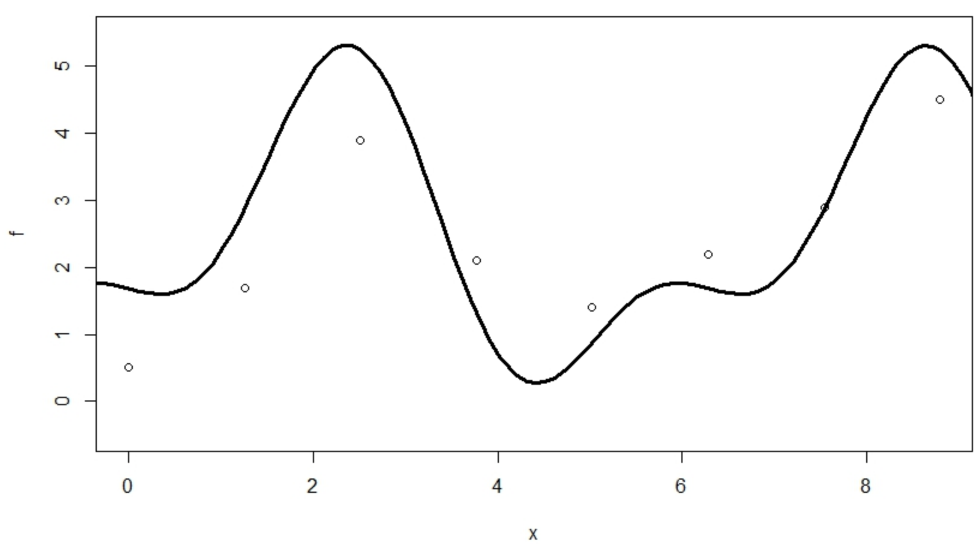
a zavoláme naprogramovanou funkci

```
MNCtg(x,f,M)
```

Na výstupu se zobrazí koeficienty

```
4.800 -0.985 1.687 0.273 -1.028
```

stejné jako v Příkladu 3.1. Nejlepší aproximace s vypočítanými koeficienty je vidět na Obrázku 5.



Obrázek 5: Aproximace dat z Příkladu 3.1 trigonometrickými polynomy

4. Aproximace nelineárního typu

Poslední část práce se zabývá aproximací nelineárního typu. Používáme ji tehdy, kdy předpokládáme exponenciální průběh funkce, proto hledaná aproximace bude ve tvaru $\tilde{f}(x) = ae^{bx}, a, b \in \mathbb{R}$. Optimální variantou je také aproximace tvaru $\tilde{f}(x) = ax^b, a, b \in \mathbb{R}$. Zde jsme nejvíce čerpali z [9].

Nechť jsou dány body $x_i, i = 0, 1, \ldots, n$. Naším úkolem je najít funkci v tomto tvaru

$$\tilde{f}(x) = ae^{bx}, \tag{17}$$

$a, b \in \mathbb{R}$, $a \neq 0$ tak, aby minimalizovala „souhrnou odchylku“ aproximace $\tilde{f}(x)$ od původní funkce f, tj.

$$\varrho^2(f, \tilde{f}) = \sum_{i=0}^{n} [f(x_i) - \tilde{f}(x_i)]^2 = \sum_{i=0}^{n} [f(x_i) - ae^{bx_i}]^2.$$

Funkci ϱ^2 zapisujeme jako funkci proměnných a, b, přičemž vektor (a, b) musí splňovat nutné podmínky pro existenci extrému, tj. musí platit

$$\frac{\partial \varrho^2(a, b)}{\partial a} = 0,$$
$$\frac{\partial \varrho^2(a, b)}{\partial b} = 0, \quad a, b \in \mathbb{R}.$$

Parciální derivace funkce $\varrho^2(a, b)$ podle proměnné a je rovna

$$2\sum_{i=0}^{n} [f(x_i) - ae^{bx_i}]e^{bx_i} = 2\sum_{i=0}^{n} f(x_i)e^{bx_i} - 2a\sum_{i=0}^{n} e^{2bx_i} \tag{18}$$

a podle proměnné b

$$2\sum_{i=0}^{n} [f(x_i) - ae^{bx_i}]ae^{bx_i}x_i = 2\sum_{i=0}^{n} f(x_i)x_ie^{bx_i} - 2a\sum_{i=0}^{n} x_ie^{2bx_i}. \tag{19}$$

Zbývá vypočítané derivace (19), (20) položit rovny nule a získáme soustavu nelineárních rovnic. Takovou soustavu rovnic je možné vyřešit např. Newtonovou metodou (metodou tečen). Avšak řešení nelineárních soustav není předmětem této práce, proto odkazujeme např. na literaturu [8].

Podívejme se nyní na druhou variantu funkce aproximace, konkrétně ve tvaru

$$\tilde{f}(x) = ax^b, \tag{20}$$

$a, b \in \mathbb{R}$, která minimalizuje odchylku $\varrho^2(a, b)$. Postupujeme stejně jako u funkce aproximace (18), tzn. hledáme stacionární bod, ve kterém funkce $\tilde{f}(x)$ nabývá svého minima. Vypočítáme si parciální derivace, položíme je rovny nule a dostaneme opět soustavu nelineárních rovnic

$$\sum_{i=0}^{n} f(x_i)x_i^b - a\sum_{i=0}^{n} x_i^{2b} = 0$$

$$\sum_{i=0}^{n} f(x_i)x_i^b \ln x_i - a\sum_{i=0}^{n} x_i^{2b} \ln x_i = 0,$$

jejíž možné řešení je podrobněji studováno v [8].

My se budeme chtít nelineárním soustavám rovnic vyhnout, proto využijeme tzv. linearizace. Opět máme dány body $x_i, i = 0, 1, \ldots, n$ a funkční hodnoty $f(x_i)$ v těchto bodech. Linearizaci můžeme použít pouze za podmínky, že $f(x_i) > 0$, $\forall i = 0, 1, \ldots, n$, což plyne z definice logaritmu. Jestliže je podmínka splněna, vhodnou transformací funkce $\tilde{f}(x)$ převedeme úlohu o nalezení nejlepší aproximace na tvar

$$\tilde{g}(x) = \ln \tilde{f}(x) = \ln a + bx$$

a zavedením substituce $\ln a = c$ můžeme psát

$$\tilde{g}(x) = c + bx, c \in \mathbb{R}.$$

Tímto způsobem jsme dostali funkci, která již je lineární kombinací lineárně nezávislých funkcí. Nové funkční hodnoty $g(x_i)$ vypočítáme vztahem

$$g(x_i) = \ln f(x_i).$$

Dále je postup nalezení nejlepší aproximace ve smyslu metody nejmenších čtverců totožný s postupem uvedeným v kapitole 1. Dojdeme tak k soustavě lineárních rovnic. Hodnotu parametru a vypočteme zpětnou transformací $a = e^c$.

Obdobně postupujeme i v druhém případě, kdy $\tilde{f}(x) = ax^b$, přičemž funkční hodnoty $f(x_i)$ původní funkce f musí taktéž nabývat pouze kladných hodnot. Funkci $\tilde{f}(x)$ zlogaritmujeme a zapíšeme

$$\tilde{g}(x) = \ln \tilde{f}(x) = \ln a + b \ln x.$$

Substitucí $c = \ln a$ a $z = \ln x$ dostaneme aproximaci ve tvaru přímky

$$\tilde{g}(z) = c + bz$$

a tu již vypočteme známým způsobem. Hodnotu parametru a a parametru x získáme zpětnou substitucí, tj. $a = e^c, x = e^z$.

Jelikož početní operace aproximace tvaru (21) jsou analogické jako s (18), v dalším textu uvádíme příklady pouze na aproximaci v podobě $\tilde{f}(x) = ae^{bx}$.

Příklad 4.1 Na data zadaná tabulkou použijte aproximaci nelineárního typu.

x_i	-1	1	2	4	5
$f(x_i)$	2.1	3.4	6.2	9.7	13.1

Aproximační funkci budeme hledat ve tvaru $\tilde{f}(x) = ae^{bx}$ tak, že

$$\tilde{g}(x) = ln\tilde{f}(x) = c + bx,$$

kde c = ln a. Pomocí vztahu $g(x_i) = lnf(x_i), i = 0, 1, \ldots, 4$ vypočteme nové transformované hodnoty

$$g(x_0) = 0.742, \quad g(x_1) = 1.224, \quad g(x_2) = 1.825,$$
$$g(x_3) = 2.272, \quad g(x_4) = 2.573.$$

Dále si vypočteme potřebné součty, tedy

$$\sum_{i=0}^{4} 1 = 5, \quad \sum_{i=0}^{4} x_i = 11, \quad \sum_{i=0}^{4} x_i^2 = 47,$$

$$\sum_{i=0}^{4} g(x_i) = 8.635, \quad \sum_{i=0}^{4} x_i g(x_i) = 26.083.$$

Získáme normální soustavu ve tvaru

$$5c + 11b = 8.636$$
$$11c + 47b = 26.085.$$

Řešením této soustavy jsou koeficienty

$$c = 1.043, \quad b = 0.311.$$

Koeficient a získáme zpětnou transformací

$$a = e^c = e^{1.043} = 2.839.$$

Nejlepší aproximace funkce dané tabulkou ve smyslu metody nejmenších čtverců ve tvaru exponenciální křivky je ve tvaru

$$\tilde{f}(x) = 2.839e^{0.311x}.$$

Graf aproximace viz. Obrázek 6.

4.1. R Script

Pro výpočet nejlepší nelineární aproximace ve smyslu metody nejmenších čtverců ve tvaru ae^{bx} jsem v Rku vytvořila Script s názvem MNCnelinfunkce.R. Vstupními daty je vektor bodů x a vektor funkčních hodnot f v bodech x. Program ověří délky vektorů x a f, různost bodů vektoru x a pokud jsou požadavky splněny, provede transformaci vektoru f a následné výpočty. Výstupními daty jsou potom dva koeficienty a, b a graf, který promítne aproximované hodnoty a nejlepší aproximaci ve tvaru exponenciály ve smyslu metody nejmenších čtverců.

```
##Data##
x = c()
f = c()
```

```
MNCn = function(x,f) {
  n = length(x)
  k = length(f)

  ##Ověření různosti bodů vektoru x##
  for(i in 1:n){
    for(j in 1:n){
      if(i != j & x[i] == x[j]){
        stop("Body vektoru x nejsou různé!")
      }
    }
  }

  ##Ověření délky vektorů##
  if(n != k){
    stop("Délka vektoru x a vektoru f není stejná!")
  }

  ##Ověření kladnosti hodnot vektoru f##
  for(j in 1:length(f)){
    if(f[j] < 0){
      stop("Všechny funkční hodnoty nejsou kladné!")
    }
  }

  ##Transformace funkčních hodnot##
  g = log(f)

  ##Součty##
  a1 = n
```

```
  a2 = sum(x)
  a3 = sum(x^2)
  a4 = sum(g)
  a5 = sum(x * g)

  ##Soustava rovnic##
  A = matrix(data = c(a1,a2,a2,a3), nrow = 2, ncol = 2, byrow = TRUE)
  c = c(a4,a5)
  reseni = solve(A) %*% c

  ##Koeficienty##
  a = exp(1)^reseni[1]
  b = reseni[2]

  ##Vykreslení##
  t = seq(from = min(x)-1, to = max(x)+1, by = 0.1)

  vyr.hod = a * (exp(1)^(b * t))

  plot(f~x)
  lines(x = t , y = vyr.hod, col = "slategray", lwd = 3)

  print(a)
print(b)
}
```

Příklad 4.2 Nakonec ukážeme, jak R Script MNCnelinfunkce.R pracuje. Vezmeme si data z Příkladu 4.1 a zapíšeme

```
x = c(-1,1,2,4,5)
f = c(2.1,3.4,6.2,9.7,13.1)
```

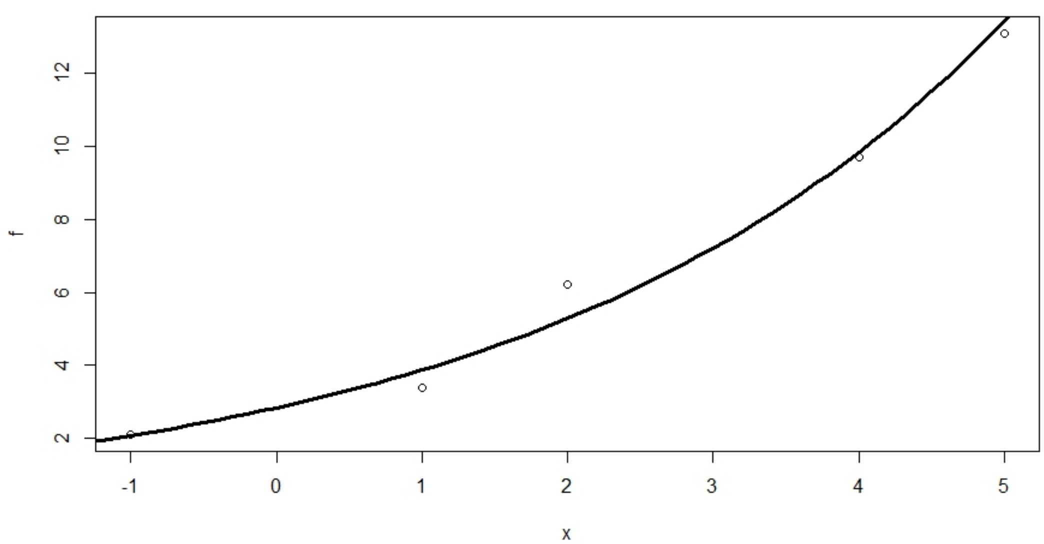
Použijeme příkaz

```
MNCn(x,f)
```

a ve výsledku dostaneme hodnoty koeficientů a a b

```
2.839 0.311
```

Koeficienty se shodují s koeficienty v ruce počítaném Příkladu 4.1. Grafické znázornění aproximace vidíme na Obrázku 6.



Obrázek 6: Nelineární aproximace dat z Příkladu 4.1

Závěr

Cílem mé bakalářské práce bylo nastudovat a ukázat možnosti aproximace funkcí ve smyslu metody nejmenších čtverců. První možnost aproximace spolu s principem metody nejmenších čtverců byla obsažena v kapitole jedna, konkrétně se jednalo o aproximaci pomocí klasických polynomů. Druhá kapitola se zabývala ortogonálními polynomy, přičemž jsme se zaměřili především na Gramovy polynomy. V třetí kapitole jsme se seznámili s trigonometrickými polynomy a nakonec jsme se v poslední kapitole podívali na aproximaci nelineárního typu. Součástí práce bylo také doplnit jednotlivé možnosti aproximace vlastními ručně počítanými příklady a především vytvořit v matematickém softwaru R vlastní programy pro řešení konkrétních úloh.

Výsledky ručně vypočítaných příkladů jsme zaokrouhlovali na tři desetinná místa. Koeficienty vygenerované R Scripty, aplikovaných na již ručně vypočítaných příkladech, jsme v práci taktéž zaokrouhlovali na tři desetinná místa, aby jsme ukázali, že koeficienty jsou totožné. Sestavené programy ale jinak generují koeficienty na více desetinných míst. Všechny uvedené programy byly vytvořeny v *RStudiu* a nahrány na CD, které bylo k této bakalářské práci přiloženo.

Vytváření práce pro mě bylo velkým přínosem. V první řadě jsem si prohloubila znalosti této problematiky, ale také jsem se naučila pracovat s matematickým softwarem R. Pro práci s metodou nejmenších čtverců jsme software použili pouze pro aproximaci diskrétního typu. Aproximace spojitých dat by mohla být námětem pro další tvorbu, avšak pro práci s touto aproximací bych spíše doporučila matematický software *Matlab*, který je uživatelsky vhodnější.

Literatura

[1] Bečvář J.: *Lineární algebra*. Matfyzpress, Praha, 2005.

[2] Hildebrand, F.B.: *Introduction to numerical analysis, second edition*. Dover Publications, Inc., New York, 1987.

[3] Kobza J.: *Numerické metody*. Univerzita Palackého v Olomouci, 1993.

[4] Mošová V.: *Numerické metody*. Univerzita Palackého v Olomouci, 2003.

[5] Navara M., Němeček A.: *Numerické metody*. ČVUT, Praha, 2005.

[6] Přikryl, P., Brandner, M.: *Numerické metody II*. Západočeská univerzita v Plzni, 2000.

[7] Ralston A.: *Základy numerické matematiky*. Academia Praha, 1978.

[8] Suchánková V.: *Řešení soustav nelineárních rovnic*. Matematicko-fyzikální fakulta UK, 1980.

[9] Vampolová J.: *Metoda nejmenších čtverců - bakalářská práce*. Přírodovědecká fakulta UP, 2009.

[10] Gažová N.: *Aproximace trigonometrickými polynomy - bakalářská práce (online)*. Přírodovědecká fakulta MU, 2016. Dostupné z WWW:https://is.muni.cz/th/423227/prif_b/BP_Natalia_Gazova.pdf [22.3.2017].

[11] Konečná K., Koláček J.: *Jak pracovat s jazykem R (online)*. Dostupné z WWW:http://www.ms.mff.cuni.cz/~obdrzalp/NMAI059/Navod_R_cesky.pdf [2.11.2016].