Vilniaus Universitetas

Perkelties metodas,

kubinis splainas

3 laboratorinis darbas

Skaitiniai metodai

Darbą atliko:

Dovydas Martinkus

Duomenų Mokslas 4 kursas 1 gr.

Vilnius, 2022

**Turinys**

[1. Užduoties ataskaita 3](#_Toc118034663)

[1.1 Perkelties metodas 3](#_Toc118034664)

[1.2 Kubinis splainas 4](#_Toc118034665)

[Priedas 7](#_Toc118034666)

# Užduoties ataskaita

## Perkelties metodas

Reikalinga sudaryti programą, tiesines lygčių sistemas sprendžiančią perkelties metodu. Programos veikimo derinimui naudojama tiesinių lygčių sistema:

Tarkime turime tiesinių lygčių sistemos matricą:

.

Žinoma, kad jei tiesinės lygčių sistemos įstrižainė yra vyraujanti, t. y. |bi| > |ai| + |ci|, i = 1, 2, . . . , n ir |b1| > |c1|, tai sprendžiant perkelties metodu, dalyba iš nulio yra negalima.

Nesunku pamatyti, kad ši sąlyga galioja ir anksčiau pateiktai tiesinių lygčių sistemai.

## Kubinis splainas

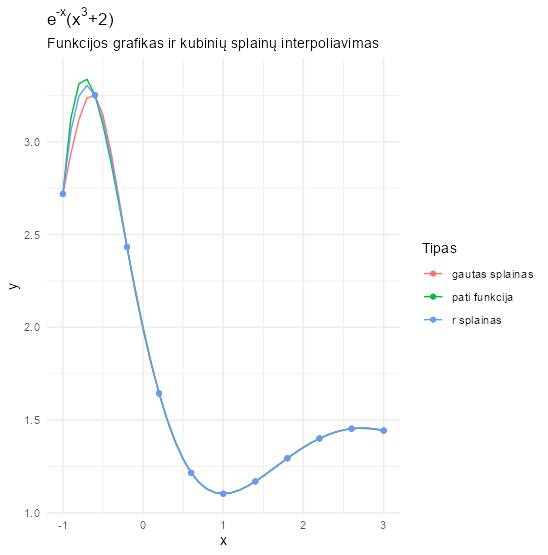
Reikalinga sudaryti kubinį splainą funkcijai intervale [-1,3]. Intervalas dalijimas į 10 vienodo ilgio intervalų taip gaunant 11 vienodai nutolusių interpoliavimo mazgų.

1 lentelėje pateiktos funkcijos reikšmės interpoliavimo mazguose (3 skaičių po kablelio tikslumu):

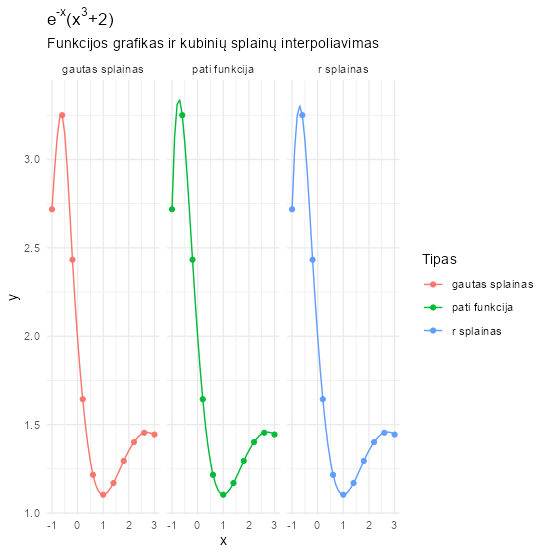
*1 lentelė Funkcijos reikšmių lentelė interpoliavimo mazguose*

|  |  |
| --- | --- |
| x | e-x(x3 + 2) |
| -1 | 2.718 |
| -0.6 | 3.251 |
| -0.2 | 2.433 |
| 0.2 | 1.644 |
| 0.6 | 1.216 |
| 1 | 1.104 |
| 1.4 | 1.17 |
| 1.8 | 1.295 |
| 2.2 | 1.401 |
| 2.6 | 1.454 |
| 3 | 1.444 |

Gauti rezultatai pateikti grafiškai (1 ir 2 pav.). Lygintas pats funkcijos grafikas, apskaičiuotas kubinis splainas ir naudojant R funkciją *splinefun* gautas kubinis splainas.



*1 pav. Duotosios funkcijos ir abiejų splainų kreives su pažymėtais interpoliavimo taškais*



*2 pav. Duotosios funkcijos ir abiejų splainų kreives su pažymėtais interpoliavimo taškais*

Kubinių splainų reikšmės interpoliavimo mazguose pateiktos 2 lentelėje(3 skaičių po kablelio tikslumu):

*2 lentelė Splainų reikšmių lentelė interpoliavimo mazguose*

|  |  |  |  |
| --- | --- | --- | --- |
| x | e-x(x3 + 2) | Apskaičiuotas splainas | *splinefun* splainas |
| -1 | 2.718 | 2.718 | 2.718 |
| -0.6 | 3.251 | 3.251 | 3.251 |
| -0.2 | 2.433 | 2.433 | 2.433 |
| 0.2 | 1.644 | 1.644 | 1.644 |
| 0.6 | 1.216 | 1.216 | 1.216 |
| 1 | 1.104 | 1.104 | 1.104 |
| 1.4 | 1.17 | 1.17 | 1.17 |
| 1.8 | 1.295 | 1.295 | 1.295 |
| 2.2 | 1.401 | 1.401 | 1.401 |
| 2.6 | 1.454 | 1.454 | 1.454 |
| 3 | 1.444 | 1.444 | 1.444 |

# Priedas

Žemiau pateiktas naudotas programinis kodas:

# Dovydas Martinkus

# Duomenų Mokslas 4k. 1gr.

# 3 uzduotis

###

# Perkelties metodas

## Funkciju aprasymas

vyraujanti <- function(A) {

result <- TRUE

A <- abs(A)

if (A[1,1] <= A[1,2]) {

result <- FALSE

}

for ( i in 2:(nrow(A)-1) ) {

if(A[i,i] < A[i,i-1] + A[i,i+1]) {

result <- FALSE

}

}

n <- nrow(A)

if (A[n,n] < A[n,n-1]) {

result <- FALSE

}

return(result)

}

perkelties <- function(A,B) {

if (!vyraujanti(A)) {

print("TLS isstrizaine nera vyraujanti")

}

p <- -1 \* A[1,2] / A[1,1]

q <- B[1] / A[1,1]

print(A)

print(B)

for ( i in 2:(nrow(A)-1) ) {

p\_i <- -1\* A[i,1+i] / (A[i,i] + A[i,i-1]\*p[i-1])

q\_i <- (B[i]-A[i,i-1]\*q[i-1]) / (A[i,i] + A[i,i-1]\*p[i-1])

p <- c(p,p\_i)

q <- c(q,q\_i)

}

n <- nrow(A)

q\_n <- (B[n] - A[n,n-1]\*q[n-1]) / (A[n,n] + A[n,n-1]\*p[n-1])

x <- numeric(n)

x[n] <- q\_n

for (i in seq(n-1,1)) {

x[i] <- p[i]\*x[i+1] + q[i]

}

return(x)

}

A <- matrix(c(3, 1, 0, 0,

-1, 4, 3, 0,

0, 2, 4, -1,

0, 0, 2, -3),

ncol=4,nrow=4,byrow=TRUE)

B <- matrix(c(2,-2,1,-1),ncol=1)

x <- perkelties(A,B)

## gauto sprendinio patikrinimas

palyginimas <- cbind(A %\*% matrix(x,ncol=1),B)

colnames(palyginimas) <- c("Gautas B","Norimas B")

t(palyginimas)

# Kubinis splainas

funkcija <- function(x) {

exp(-x)\*(x^3+2)

}

interpoliavimo\_taskai <- function(func,n,a,b) {

step <-(b-a)/n

x <- a + step\*(0:10)

y <- funkcija(x)

return(data.frame(x=x,y=y))

}

kubinis\_splainas <- function(x,y) {

n <- length(x)-1

h <- diff(x)

y\_diff <- diff(y)

B <- numeric(n-1)

A <- matrix(nrow=n-1,ncol=n-1)

print(h)

print(y\_diff)

for ( i in 1:(n-1) ) {

if (i == 1) {

row <- c(2\*(h[i]+h[i+1]),h[i+1],rep(0,n-1-i-1))

}

else if (i == n-1) {

row <- c(rep(0,n-1-2),h[i],2\*(h[i]+h[i+1]))

}

else {

row <- c(c(rep(0,i-2),h[i],2\*(h[i]+h[i+1]),h[i+1],rep(0,n-1-i-1)))

}

b\_row <- 6\*((y[i+2]-y[i+1])/h[i+1] - (y[i+1]-y[i])/h[i])

print(i)

A[i,] <- row

B[i] <- b\_row

}

g <- c(0,perkelties(A,B),0)

G <- g[1:length(g)-1] / 2

e <- y\_diff / h - 1/6\*g[2:length(g)]\*h - 1/3\*g[1:length(g)-1]\*h

H <- diff(g) / (6 \* h)

func <- function(z) {

results <- c()

for (zz in z) {

for ( i in 0:(n-1) ) {

if (x[i+1] <= zz & zz <= x[i+2]) {

results <- c(results,y[i+1] + e[i+1]\*(zz - x[i+1]) + G[i+1]\*(zz - x[i+1])^2 + H[i+1]\*(zz - x[i+1])^3)

break

}

if (i == n-1) {

print("Funkcijos argumentas ne is tinkamo intervalo")

results <- c(results,NA)

}

}

}

return(results)

}

}

a <- -1

b <- 3

n <- 10

lentele <- interpoliavimo\_taskai(funkcija,n,a,b)

gautas\_splainas <- kubinis\_splainas(lentele$x,lentele$y)

r\_splainas <- splinefun(lentele$x,lentele$y)

library(tidyverse)

library(latex2exp)

rezultatai <- tibble(x = seq(-1, 3, 0.1),

`pati funkcija` = funkcija(x),

`gautas splainas` = gautas\_splainas(x),

`r splainas` = r\_splainas(x))

rezultatai2 <- tibble(x = lentele$x,

`pati funkcija` = funkcija(x),

`gautas splainas` = gautas\_splainas(x),

`r splainas` = r\_splainas(x))

rezultatai <- rezultatai %>% pivot\_longer(2:4,names\_to = "Tipas",values\_to="y")

rezultatai2 <- rezultatai2 %>% pivot\_longer(2:4,names\_to = "Tipas",values\_to="y")

ggplot(rezultatai, aes(x,y,color=Tipas)) +

geom\_line() + geom\_point(data = rezultatai2, aes(x,y,color=Tipas)) +

labs(title=TeX("e^{-x}(x^3+2)"),

subtitle = "Funkcijos grafikas ir kubinių splainų interpoliavimas") +

theme\_minimal()

ggplot(rezultatai, aes(x,y,color=Tipas)) +

geom\_line() + geom\_point(data = rezultatai2, aes(x,y,color=Tipas)) +

labs(title=TeX("e^{-x}(x^3+2)"),

subtitle = "Funkcijos grafikas ir kubinių splainų interpoliavimas") +

theme\_minimal() + facet\_wrap(vars(Tipas))