**Interpoliavimas ir aproksimavimas**

# Interpoliavimas daugianariu

Duota interpoliuojamos funkcijos analitinė išraiška (1 lentelė):

1 lentelė. Interpoliavimo daugianariu užduotis

|  |  |
| --- | --- |
| Nr. | Funkcijos išraiška |
| 16 |  |
| Bazinė funkcija: Vienanarių | |

## Interpoliacinės funkcijos radimas, kai taškai pasiskirstę tolygiai

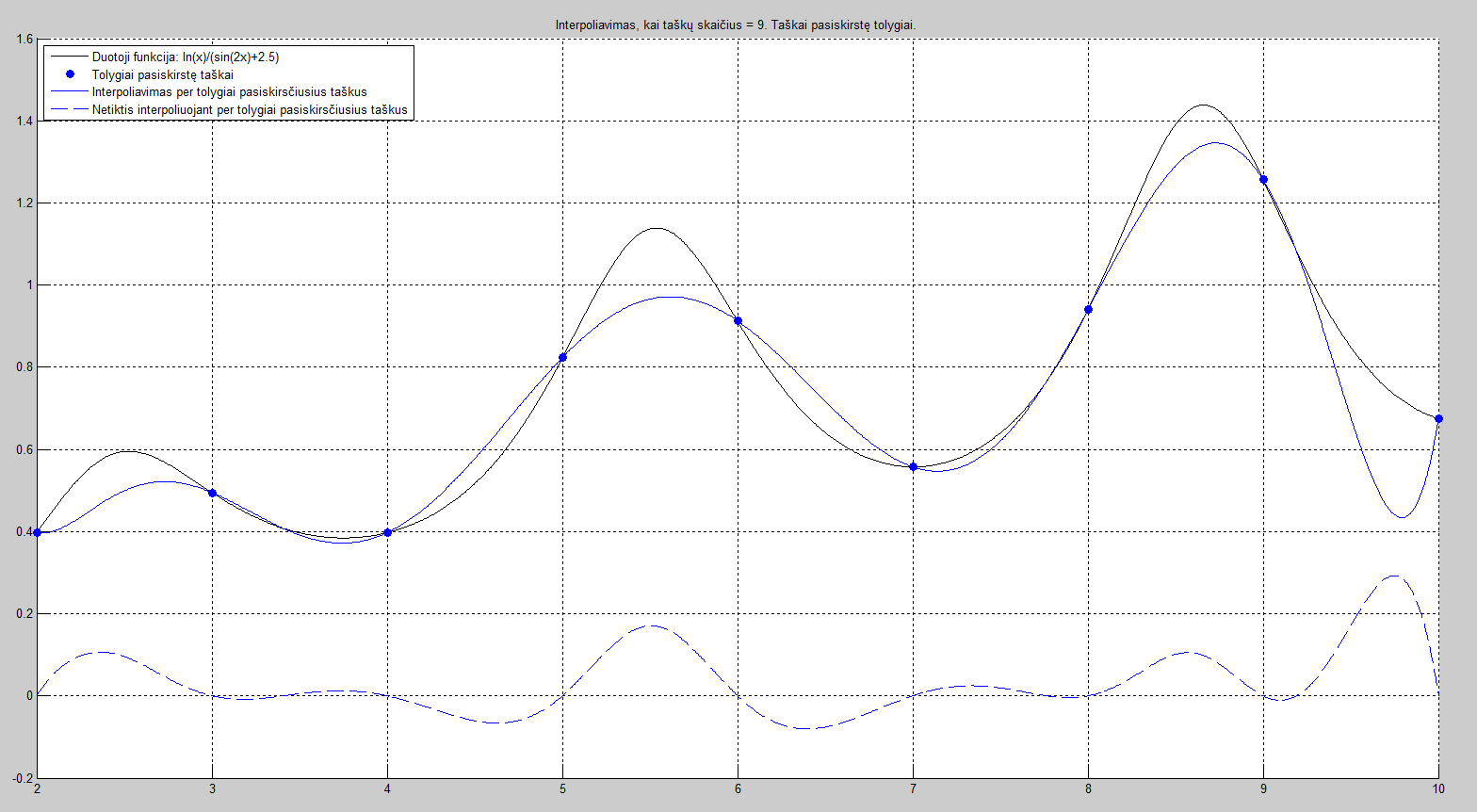
## Interpoliacinės funkcijos parametrai

2 lentelėje pateikiami interpoliacinės funkcijos parametrai, apibrėžiantys šią funkciją.

2 lentelė. Interpoliacinės funkcijos parametrai

|  |  |
| --- | --- |
| Interpoliavimo taškų skaičius | 9 |
| Interpoliavimo taškų koordinatės |  |
| Interpoliacinės funkcijos išraiška |  |

## Taškų pasiskirstymas ir interpoliacinės funkcijos bei jos netikties vaizdavimas

1 paveikslėlyje vaizduojamas interpoliacinės funkcijos grafikas, bei jo funkcijos reikšmių netiktis, lyginant su interpoliuojama funkcija.

1 pav. Interpoliacinės funkcijos bei jos netikties grafinis vaizdas

## Interpoliacinės funkcijos radimas, kai taškai apskaičiuojami naudojant Čiobyševo abscises

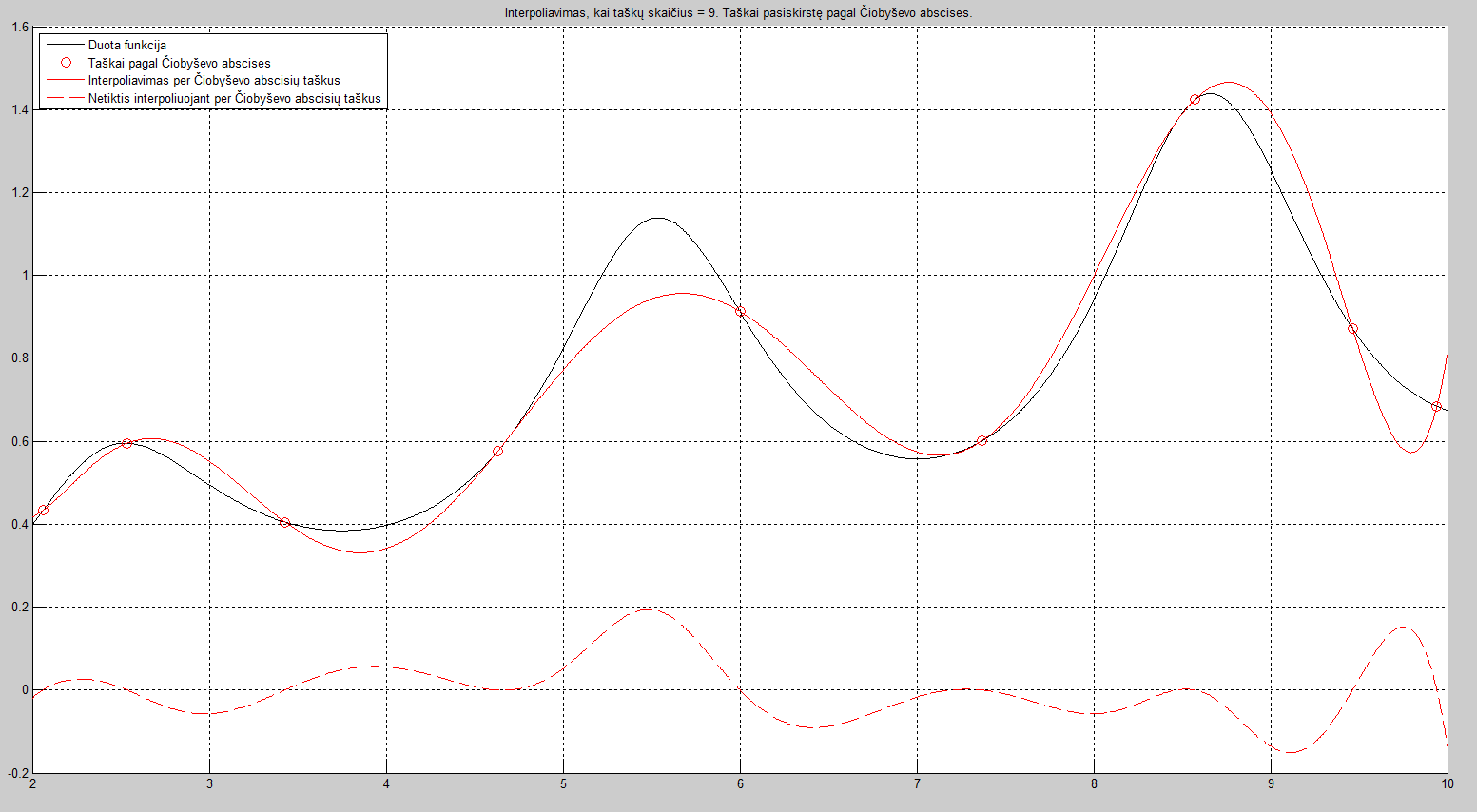
## Interpoliacinės funkcijos parametrai

3 lentelėje pateikiami interpoliacinės funkcijos parametrai, apibrėžiantys šią funkciją.

3 lentelė. Interpoliacinės funkcijos parametrai

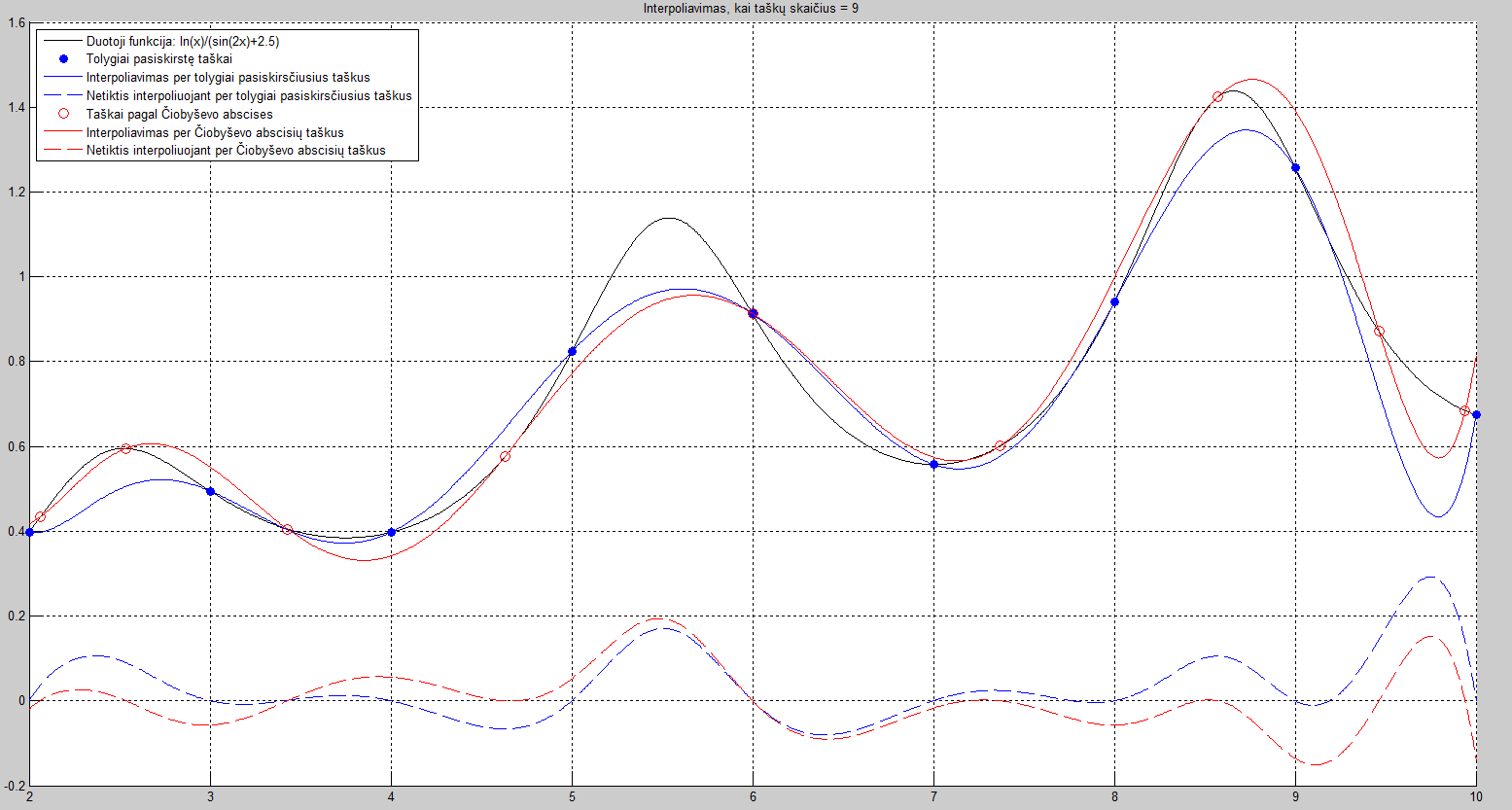
|  |  |
| --- | --- |
| Interpoliavimo taškų skaičius | 9 |
| Interpoliavimo taškų koordinatės |  |
| Interpoliacinės funkcijos išraiška |  |

## Taškų pasiskirstymas ir interpoliacinės funkcijos bei jos netikties vaizdavimas

 1 paveikslėlyje vaizduojamas interpoliacinės funkcijos grafikas, bei jo funkcijos reikšmių netiktis, lyginant su interpoliuojama funkcija.

2 pav. Interpoliacinės funkcijos bei jos netikties grafinis vaizdas

## Abiejų interpoliacinių funkcijų palyginimas

Viename grafike vaizduojamos tiek per tolygiai pasiskirsčiusius taškus, tiek per Čiobyševo abscisėmis apskaičiuotus taškus nubrėžtos interpoliacinės funkcijos bei duotoji interpoliuojama funkcija (3 pav.):

3 pav. Abiejų interpoliacinių funkcijų bei jų netikčių ir interpoliuojamos funkcijos grafikai

**Išvados:** Iš interpoliacinių funkcijų grafikų (1-2 pav.), bei bendro tų pačių funkcijų grafiko (3 pav.) matome, kad funkcijos duotąją funkciją interpoliuoja gana tiksliai (didžiausia netiktis – apie 0,3). Taip pat galima teigti, kad labai nežymiai, bet tiksliau interpoliuoja ta funkcija, kurios interpoliuojamų taškų koordinatės buvo apskaičiuojamos naudojant Čiobyševo abscises, todėl pasitvirtina tai, jog per Čiobyševo interpoliavimo mazgus pravesta interpoliuojančio daugianario kreivė yra mažiausiai „banguota“.

# Parametrinis interpoliavimas

Duotas parametrinio interpoliavimo metodas (4 lentelė):

4 lentelė. Parametrinio interpoliavimo užduotis

|  |  |
| --- | --- |
| Nr. | Interpoliavimo metodas |
| 16 | Antros eilės defekto splainas |

## Pirmosios vardo raidės „M“ formavimas

## Interpoliavimo parametrai

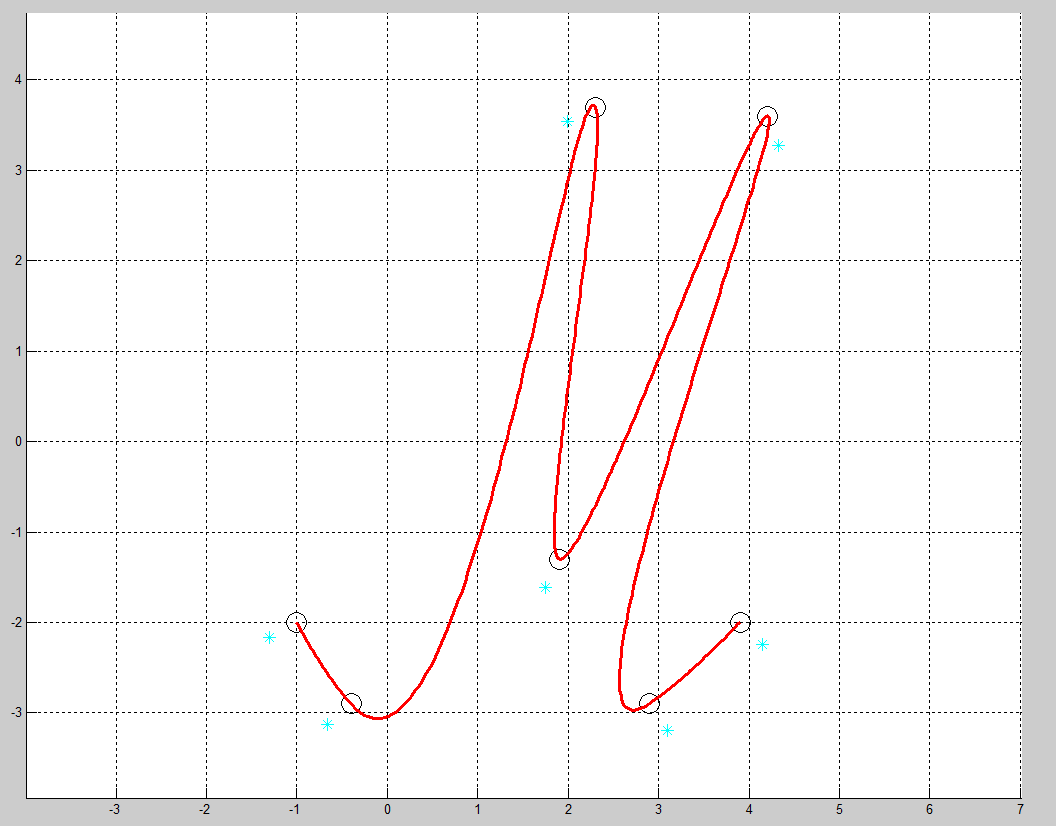
Raidės „M“ formavimui reikalingi parametrai nurodyti 5 lentelėje:

5 lentelė. Raidės „M“ parametrinio interpoliavimo parametrai

|  |  |
| --- | --- |
| Interpoliavimo taškų skaičius | 7 |
| Interpoliavimo taškų koordinatės |  |

## Raidės „M“ vaizdavimas

4 paveikslėlyje vaizduojama suformuota raidė „M“, gauta naudojant 5 lentelėje nurodytus parametrinio interpoliavimo parametrus, naudojant Ermito (2 eilės defekto) splainą.



4 pav. Ermito splainu suformuota raidė „M“

## Pirmosios pavardės raidės „K“ formavimas

## Interpoliavimo parametrai

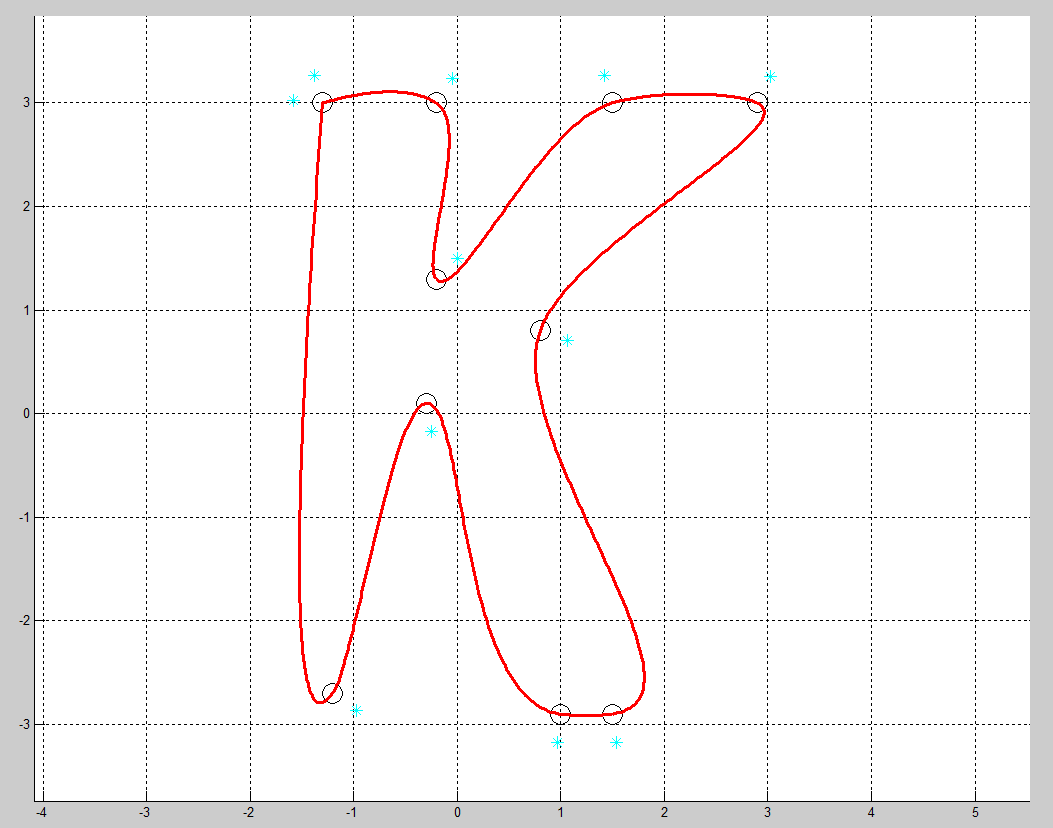
Raidės „K“ formavimui reikalingi parametrai nurodyti 6 lentelėje:

6 lentelė. Raidės „K“ parametrinio interpoliavimo parametrai

|  |  |
| --- | --- |
| Interpoliavimo taškų skaičius | 11 |
| Interpoliavimo taškų koordinatės |  |

## Raidės „K“ vaizdavimas

5 paveikslėlyje vaizduojama suformuota raidė „K“, gauta naudojant 6 lentelėje nurodytus parametrinio interpoliavimo parametrus, naudojant Ermito (2 eilės defekto) splainą.



5 pav. Ermito splainu suformuota raidė „K“

**Išvados:** iš 4 bei 5 paveikslėlių matome, kad iš Ermito splainų galima suformuoti įvairias formas, šiuo atveju, raides „M“ ir „K“. Kadangi kiekviename interpoliavimo taške galime valdyti funkcijos reikšmę bei jos išvestinės reikšmę, interpoliuojančiai kreivei nesunku suteikti pageidaujamą formą. Taip pat keičiant interpoliavimo taško funkcijos ar jos išvestinės reikšmę, keičiasi pakinta tik dviejų su keičiamu tašku susijusių splainų forma, todėl galima daryti išvadą, jog Ermito splainas yra lokalusis.

# Haro bangelės

## Aproksimavimo parametrai

Duotas automobilio paveikslėlis, kurio viršutiniame kontūre parenkami aproksimavimo taškai (7 lentelė):

7 lentelė. Darbo užduoties paveikslėlis bei aproksimavimo taškų parametrai

|  |  |
| --- | --- |
| Automobilio paveikslėlis |  |
| Aproksimavimo taškų skaičius | 10 |
| Aproksimavimo taškų koordinatės |  |

## Taškų aproksimavimas Haro bangelėmis

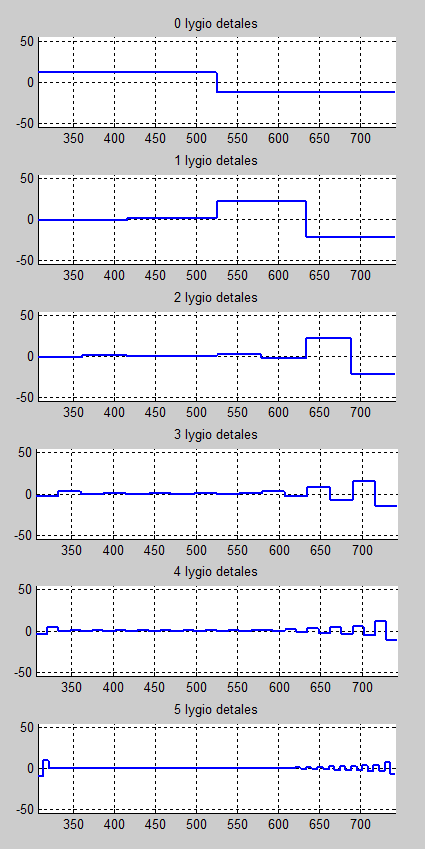
## Duotoji funkcija (sudaryta pagal aproksimavimo taškus)

6 paveiksėlyje pateikta taškų funkcija, suformuota pagal duotus aproksimavimo taškus, naudojant linijinę interpoliaciją.

## Haro bangelės

6 pav. Duotosios funkcijos grafikas, suformuotas pagal interpoliavimo taškus

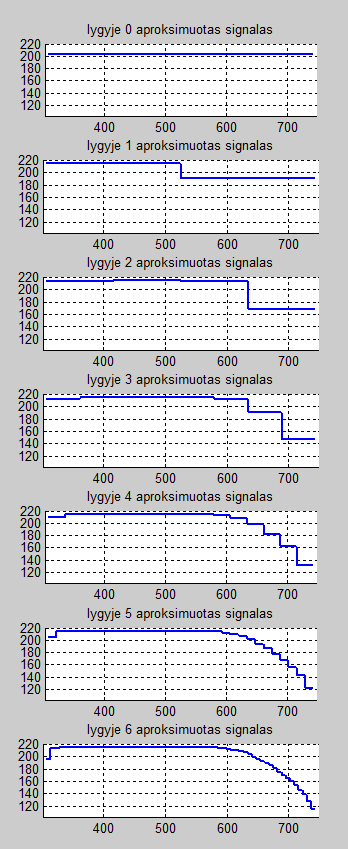
7 paveikslėlyje pateiktos Haro bangelės įvairiuose detalumo lygiuose:



7 pav. Haro bangelių kitimas, priklausantis nuo detalumo lygio

## Aproksimavimo rezultatai

8 paveiksėlyje matome aproksimavimo rezultatus iki 6 detalumo lygio, naudojant Haro bangeles:



8 pav. Atkuriamo vaizdo tikslumo kitimas naudojant aproksimavimą Haro bangelėmis

**Išvados:** kaip matome iš aproksimavimo rezultatų, didėjant detalumo lygiui, atkuriamas automobilio kontūras vaizduojamas vis tikslesnis, o Haro bangelių vis daugėja, jos spaudžiamos ir paslenkamos Ox ašimi.

# Išvados

Užduoties atlikimo metu buvo išmėginti bei nagrinėti įvairūs interpoliavimo bei aproksimavimo metodai. Pirmoje užduoties dalyje buvo galima pastebėti, kad interpoliuojant funkciją, kai interpoliavimo taškų reikšmės paskaičiuotos naudojant Čiobyševo abscises, rezultatas gaunamas tikslesnis nei interpoliuojant tada, kai interpoliavimo taškai pasiskirstę tolygiai. Taip pastebėta ir įsitikinta, kad Ermito splainai yra lokalieji bei keičiant interpoliuojamų taškų bei jų išvestinių reikšmes, pakinta tik dviejų splainų, susijusių su tuo tašku, forma, kas taip pat leidžia nesunkiai koreguoti bei visai interpoliuojančiai kreivei suteikti norimą formą. Iš trečios (aproksimavimo Haro bangelėmis) užduoties matome, kad naudojant Haro bangeles, galima aproksimuoti norimus taškus ir atkurti reikiamą kontūro formą.

# Programos kodas

function L2\_1\_1

clc, close all, clear all;

format long;

atvaizdavimas = 0; % funkcijų atvaizdavimo formatas 0 - viskas viename lange, 1 - dviejuose languose

f=@(x)log(x)./(sin(2\*x)+2.5);

f\_sym = 'ln(x)/(sin(2x)+2.5)';

a = 2;

b = 10;

n = 9; % taškų skaičius

X = a:(b-a)/(n-1):b; % tolygiai pasiskirsčiųsių taškų abscisės

i = 0:n-1;

XC = (b-a)/2\*cos(pi\*(2\*i+1)/(2\*n)) + (b+a)/2; % Čiobyševo abscisės

x = min(X):(max(X)-min(X))/1000:max(X); % vaizdavimui skirtos x reikšmės

M = ones(n);

MC = ones(n);

% M matricos skaičiavimas (eilutėje - vieno taško x^0, x^1... x^(n-1).)

for j=2:n

for k=1:n

M(k,j) = X(k).^(j-1);

MC(k,j) = XC(k).^(j-1);

end

end

A = M\f(X).' % vienanarių bazės koeficientų reikšmės, kai taškai pasiskirstę tolygiai

AC = MC\f(XC).'; % vienanarių bazės koeficientų reikšmės, kai taškų abscisės - Čiobyševo

p = poly2sym(fliplr(A.'))

pc = poly2sym(fliplr(AC.'))

Y = [];

YC = [];

Y\_NET = [];

YC\_NET = [];

% Ciklas, pagal gautus koeficientus, skaičiuoja taškų ordinates

% kiekviename funkcijos vaizdavimo taške bei skaičiuojama kiekvieno

% taško netiktis

for j=1:numel(x)

y = 0;

yc = 0;

for k=1:n

y = y + x(j)^(k-1)\*A(k);

yc = yc + x(j)^(k-1)\*AC(k);

end

Y = [Y; y];

YC = [YC; yc];

Y\_NET = [Y\_NET; f(x(j))-y];

YC\_NET = [YC\_NET; f(x(j))-yc];

end

% Funkcijų grafikų atvaizdavimas, kur 0 - viskas viename lange, 1 -

% dviejuose languose (viename tolygiai pasiskirstę taškai, kitame -

% taškai, pasiskirstę pagal Čiobyševo abscises)

if atvaizdavimas == 0

% Atvaizdavimas viename lange (vienas langas)

leg={['Duotoji funkcija: ', f\_sym],...

'Tolygiai pasiskirstę taškai',...

'Interpoliavimas per tolygiai pasiskirsčiusius taškus',...

'Netiktis interpoliuojant per tolygiai pasiskirsčiusius taškus',...

'Taškai pagal Čiobyševo abscises',...

'Interpoliavimas per Čiobyševo abscisių taškus',...

'Netiktis interpoliuojant per Čiobyševo abscisių taškus'};

figure(1), hold on, grid on;

title(['Interpoliavimas, kai taškų skaičius = ', num2str(n)]);

plot(x,f(x),'k-'); % duotoji funkcija

plot(X,f(X),'bo','MarkerFaceColor','b','MarkerSize',6); % tolygiai pasiskirstę taškai

plot(x, Y, 'b-'); % interpoliavimas per tolygiai pasiskirsčiusius taškus

plot(x,Y\_NET,'b--'), % netiktis interpoliuojant per tolygiai pasiskirsčiusius taškus

plot(XC, f(XC),'ro','MarkerSize',8) % taškai pagal Čiobyševo abscises

plot(x, YC, 'r-'); % interpoliavimas per Čiobyševo abscisių taškus

plot(x,YC\_NET,'r--'), % netiktis interpoliuojant per Čiobyševo abscisių taškus

legend(leg, 'Location', 'northwest');

elseif atvaizdavimas == 1

% Atvaizdavias atskiruose grafikuose (2 langai)

leg1={['Duotoji funkcija: ', f\_sym],...

'Tolygiai pasiskirstę taškai',...

'Interpoliavimas per tolygiai pasiskirsčiusius taškus',...

'Netiktis interpoliuojant per tolygiai pasiskirsčiusius taškus'};

leg2={'Duota funkcija',...,

'Taškai pagal Čiobyševo abscises',...

'Interpoliavimas per Čiobyševo abscisių taškus',...

'Netiktis interpoliuojant per Čiobyševo abscisių taškus'};

figure(1), hold on, grid on;

title(['Interpoliavimas, kai taškų skaičius = ', num2str(n), '. Taškai pasiskirstę tolygiai.']);

plot(x,f(x),'k-'); % duotoji funkcija

plot(X,f(X),'bo','MarkerFaceColor','b','MarkerSize',6); % tolygiai pasiskirstę taškai

plot(x, Y, 'b-'); % interpoliavimas per tolygiai pasiskirsčiusius taškus

plot(x,Y\_NET,'b--'), % netiktis interpoliuojant per tolygiai pasiskirsčiusius taškus

legend(leg1, 'Location', 'northwest');

figure(2), hold on, grid on;

title(['Interpoliavimas, kai taškų skaičius = ', num2str(n), '. Taškai pasiskirstę pagal Čiobyševo abscises.']);

plot(x,f(x),'k-'); % duotoji funkcija

plot(XC, f(XC),'ro','MarkerSize',8) % taškai pagal Čiobyševo abscises

plot(x, YC, 'r-'); % interpoliavimas per Čiobyševo abscisių taškus

plot(x,YC\_NET,'r--'), % netiktis interpoliuojant per Čiobyševo abscisių taškus

legend(leg2, 'Location', 'northwest');

else

disp('Funkcijų atvaizdavimo pasirinkimas netinkamas');

end

end

% Parametrinis interpoliavimas\_Ermito\_splainais

% kiekvienas intervalas tarp tasku interpoliuojamas

% 2 eiles Ermito daugianariais, nustatant isvestiniu reiksmes pagal

% skaitinio diferencijavimo formules

% valdomos tasku padetys ir isvestiniu reiksmes

function L2\_1\_2

clc,close all

hL=[];

raide = 1; % 0 - raidė M, 1 - raidė K

% X=[-1 0.5 1 1 -1 -1 ]

% Y=[ 1 0.5 1 -1 -1 1 ]

if raide == 0

X=[-1 -0.4 2.3 1.9 4.2 2.9 3.9]

Y=[-2 -2.9 3.7 -1.3 3.6 -2.9 -2]

else

X=[-1.3 -1.2 -0.3 1 1.5 0.8 2.9 1.5 -0.2 -0.2 -1.3]

Y=[ 3 -2.7 0.1 -2.9 -2.9 0.8 3 3 1.3 3 3]

end

nP=length(X); % interpoliavimo mazgu skaicius

t(1)=0; for i=2:nP, t(i)=t(i-1)+norm([X(i) Y(i)]-[X(i-1) Y(i-1)]); end % parametro reiksmes

t

DX=Akima(t,X),DY=Akima(t,Y)

dist=(max(X)-min(X))/15 % posukiu rankeneles ilgis

f=figure(1);axis([min(X)-3\*dist,max(X)+3\*dist,min(Y)-3\*dist,max(Y)+3\*dist]);axis equal;hold on; grid on;

% vaizduojame duotus taskus

for i=1:nP,

h(i)=plot(X(i), Y(i),'ko','ButtonDownFcn',@startDragFcn,'MarkerSize',15);

% kas atliekama paspaudus peles klavisa, nurodoma funkcijoje startDragFcn

% tasku objektu valdikliai issaugomi masyve h

end

% vaizduojame liestines

for i=1:nP,

ang=atan2(DY(i),DX(i));

ht(i)=plot(X(i)+dist\*sin(ang), Y(i)-dist\*cos(ang),'c\*','ButtonDownFcn',@startDragFcn\_t,'MarkerSize',10);

% kas atliekama paspaudus peles klavisa, nurodoma funkcijoje startDragFcn\_t

% liestiniu objektu valdikliu taskai issaugomi masyve ht

end

set(f,'WindowButtonUpFcn',@stopDragFcn); % kas atliekama atleidus peles klavisa, nurodoma funkcijoje stopDragFcn

Ermito\_splainu\_parametrinis\_interpoliavimas(X,DX,Y,DY,t); % interpoliuojame pagal ivestus taskus ir

% nubraizome pradine kreive

%\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

% Toliau programa laukia pertraukimo nuo peles klaviso, kuris inicijuja

% startDragFcn arba stopDragFcn vykdyma. Jos savo ruoztu peles judesi susieja arba atsieja

% su draggingFcn

%\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

%----------- vidines funkcijos ------------------

% jos aprasomos anksciau, nei sutinkamas pagrindines funkcijos "end",

% todel visi pagrindineje funkcijoje naudojami kintamieji matomi taip pat

% ir vidinese funkcijose

function startDragFcn(varargin) % apraso, kas atliekama, kai paspaudziamas kairys peles klavisas

set(gcf, 'WindowButtonMotionFcn',@draggingFcn); % nurodo funkcija, kuria reikia nuolat kviesti pelei judant

end

function startDragFcn\_t(varargin) % apraso, kas atliekama, kai paspaudziamas kairys peles klavisas

set(gcf, 'WindowButtonMotionFcn',@draggingFcn\_t); % nurodo funkcija, kuria reikia nuolat kviesti pelei judant

end

function draggingFcn(varargin) % apraso, kas atliekama, kai pakinta pele valdomo objekto padetis

iob=find(gco == h); % kelintas taskas judinamas

x=get(h(iob),'xData');y=get(h(iob),'yData'); % tasko koordinates

xt=get(ht(iob),'xData');yt=get(ht(iob),'yData'); % liestines valdiklio koordinates

pt=get(gca,'Currentpoint'); % perskaitoma nauja padetis

set(gco,'xData',pt(1,1),'yData',pt(1,2)); % pakeiciamos objekto koordinates

X(iob)=pt(1,1); % naujos tasko koordinates i masyva

Y(iob)=pt(1,2);

set(ht(iob),'xData',xt+pt(1,1)-x,'yData',yt+pt(1,2)-y); % pakeiciamos objekto koordinates

% kvieciame savo sukurta funkcija interpoliuojanciai kreivei apskaiciuoti:

Ermito\_splainu\_parametrinis\_interpoliavimas(X,DX,Y,DY,t);

end

function draggingFcn\_t(varargin)

% apraso, kas atliekama, kai pakinta pele valdomo liestines valdiklio padetis

iob=find(gco == ht); % kelinto tasko liestines valdiklis judinamas

x=get(h(iob),'xData');y=get(h(iob),'yData'); % tasko koordinates

xt=get(ht(iob),'xData');yt=get(ht(iob),'yData'); % liestines valdiklio koordinates

ang0=atan2(yt-y,xt-x); % anksciau buves valdiklio rankeneles kampas

pt=get(gca,'Currentpoint'); % perskaitoma nauja padetis

dx=pt(1,1)-x; dy=pt(1,2)-y; % valdiklio vektoriaus projekcijos

normd=norm([dx dy]); dx=dx/normd\*dist;dy=dy/normd\*dist;

set(gco,'xData',x+dx,'yData',y+dy); % pakeiciamos objekto koordinates

ang=atan2(dy,dx); % naujas valdiklio rankeneles kampas

DY(iob)=DY(iob)+sign(DX(iob))\*tan(ang-ang0);

% kvieciame savo sukurta funkcija interpoliuojanciai kreivei apskaiciuoti:

Ermito\_splainu\_parametrinis\_interpoliavimas(X,DX,Y,DY,t);

end

function stopDragFcn(varargin) % apraso, kas atliekama, kai atleidziamas kairys peles klavisas

set(gcf, 'WindowButtonMotionFcn','');% nurodo, kad atleidus peles klavisa peles judejimas nebeturi kviesti funkcijos

end

function Ermito\_splainu\_parametrinis\_interpoliavimas(X,DX,Y,DY,t);

nP=length(X); % interpoliavimo tasku skaicius

if ~isempty(hL), delete(hL); end

for iii=1:nP-1 %------ ciklas per intervalus tarp gretimu tasku

nnn=100;

ttt=[t(iii):(t(iii+1)-t(iii))/nnn:t(iii+1)];

fffX=0;fffY=0;

for j=1:2

[U,V]=Hermite(t(iii:iii+1),j,ttt);

fffY=fffY+U\*Y(iii+j-1)+V\*DY(iii+j-1);

fffX=fffX+U\*X(iii+j-1)+V\*DX(iii+j-1);

end

hL(iii)=plot(fffX,fffY,'r-','LineWidth',2.5);

end %-----------------ciklas per intervalus pabaiga

return

end

function [U,V]=Hermite(X,j,x) % Ermito daugianariai

L=Lagrange(X,j,x); DL=D\_Lagrange(X,j,X(j));

U=(1-2\*DL.\*(x-X(j))).\*L.^2;

V=(x-X(j)).\*L.^2;

return

end

function L=Lagrange(X,j,x) % Lagranzo daugianaris

n=length(X);

L=1;

for k=1:n, if k ~= j, L=L.\*(x-X(k))/(X(j)-X(k)); end, end

return

end

function DL=D\_Lagrange(X,j,x) % Lagranzo daugianario isvestine pagal x

n=length(X);

DL=0; %DL israskos skaitiklis

for i=1:n % ciklas per atmetamus narius

if i==j, continue, end

Lds=1;

for k=1:n, if k ~= j && k ~= i , Lds=Lds.\*(x-X(k)); end, end

DL=DL+Lds;

end

Ldv=1; %DL israskos vardiklis

for k=1:n, if k ~= j, Ldv=Ldv.\*(X(j)-X(k)); end , end

DL=DL/Ldv;

return

end

function DY=Akima(X,Y)

% Isvestiniu reiksmiu interpoliavimo taskuose nustatymas pagal skaitinio integravimo formules

n=length(X);

fnk=inline('(2\*x-xi-xip1)/((xim1-xi)\*(xim1-xip1))\*yim1+(2\*x-xim1-xip1)/((xi-xim1)\*(xi-xip1))\*yi+(2\*x-xim1-xi)/((xip1-xim1)\*(xip1-xi))\*yip1')

for i=1:n

if i == 1,xim1=X(1);xi=X(2);xip1=X(3); yim1=Y(1);yi=Y(2);yip1=Y(3);DY(i)=fnk(xim1,xi,xim1,xip1,yi,yim1,yip1);

elseif i == n, xim1=X(n-2);xi=X(n-1);xip1=X(n); yim1=Y(n-2);yi=Y(n-1);yip1=Y(n); DY(n)=fnk(xip1,xi,xim1,xip1,yi,yim1,yip1);

else, xim1=X(i-1);xi=X(i);xip1=X(i+1); yim1=Y(i-1);yi=Y(i);yip1=Y(i+1); DY(i)=fnk(xi,xi,xim1,xip1,yi,yim1,yip1);

end

end

return

end

end % Sis end uzbaigia pagrindine funkcija

%

% Haro bangeliu aproksimacija

%

function L2\_1\_3

clc;close all;clear all;

% Is failu ivedami duomenys:

n=10

nnn=2^n;

fclose all; fhx=fopen('carx.txt','r'); fhy=fopen('cary.txt','r');

figure(1); axis equal; hold on,grid on

SX=fscanf(fhx,'%g '); SY=fscanf(fhy,'%g '); fclose all; plot(SX,SY);

a=min(SX),b=max(SX),t=[a:(b-a)/(nnn-1):b];

axis([min(SX)-5 max(SX)+5 min(SY)-5 max(SY)+5]);

ts=interp1(SX,SY,t);

clear SX SY, SX=t;SY=ts;plot(SX,SY,'r.');

title(sprintf('Duota funkcija, taškų skaičius 2^n, kur n = %d',n));

xmin=min(SX);xmax=max(SX);

ymin=min(SY);ymax=max(SY);

% Aproksimavimas Haro bangelemis:

m=10 % detalumo lygiu skaicius

smooth=(b-a)\*SY\*2^(-n/2); % auksciausio detalumo suglodinimas (pagal duota funkcija)

for i=1:m

smooth1=(smooth(1:2:end)+smooth(2:2:end))/sqrt(2);

details{i}=(smooth(1:2:end)-smooth(2:2:end))/sqrt(2);

fprintf(1,'\n details %d : ',i);fprintf('%g ', details{i});

smooth=smooth1;

end

fprintf(1,'\n smooth %d : ',i);fprintf('%g ', smooth);fprintf('\n');

% Funkcijos rekonstrukcija:

h=zeros(1,nnn); for k=0:2^(n-m)-1, h=h+smooth(k+1)\*Haar\_scaling(SX,n-m,k,a,b); end % suglodinta funkcija

leg={sprintf('aproksimuotas signalas, detalumo lygmuo %d',n-m)};

figure(2);subplot(7,1,1),axis equal,axis([xmin-5 xmax+5 ymin-5 ymax+5]); hold on,grid on, plot(SX,h,'Linewidth',2);title(sprintf('lygyje %d aproksimuotas signalas',0));

for i=0:5 %detalumo didinimo ciklas

% apskaiciuojamos funkcijos detales:

h1=zeros(1,nnn); for k=0:2^(n-m+i)-1, h1=h1+details{m-i}(k+1)\*Haar\_wavelet(SX,n-m+i,k,a,b); end

figure(3),subplot(6,1,i+1), axis equal,hold on,grid on

yshift=(ymin+ymax)/2;axis([xmin xmax ymin-yshift ymax-yshift]), plot(SX,h1,'b-','Linewidth',2);title(sprintf('%d lygio detales',i));

leg={leg{1:end},sprintf('lygmens %d detales',n-m+i)};

h=h+h1; % detales pridedamos prie ankstesnio suglodinto vaizdo

figure(2);subplot(7,1,i+2),axis equal,axis([xmin-5 xmax+5 ymin-5 ymax+5]), hold on,grid on, plot(SX,h,'Linewidth',2);title(sprintf('lygyje %d aproksimuotas signalas' ,i+1));

end

return

end

function h=Haar\_scaling(x,j,k,a,b) % \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

eps=1e-9;

xtld=(x-a)/(b-a); % (a,b) intervale duota kintamojo reiksme perskaiciuojama i "standartini"

% intervala (0,1), kuriame uzrasyta bangeles formule

xx=2^j\*xtld-k; h=2^(j/2)\*(sign(xx+eps)-sign(xx-1-eps))/(2\*(b-a));

return

end

function h=Haar\_wavelet(x,j,k,a,b) % \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

eps=1e-9;

xtld=(x-a)/(b-a); % (a,b) intervale duota kintamojo reiksme perskaiciuojama i "standartini"

% intervala (0,1), kuriame uzrasyta bangeles formule

xx=2^j\*xtld-k; h=2^(j/2)\*(sign(xx+eps)-2\*sign(xx-0.5)+sign(xx-1-eps))/(2\*(b-a));

return

end