**Antras namų darbas. Pirma dalis.**

Variantas Nr. 3

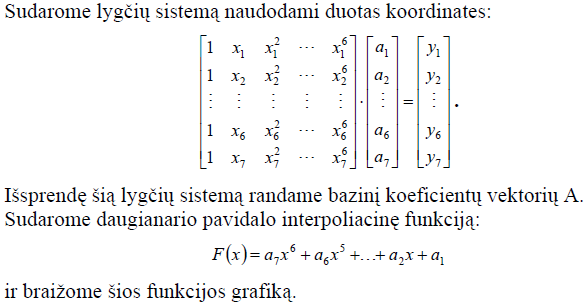
1. Reikia apskaičiuoti interpoliacines funkcijas vienanarių ir Čiobyševo bazėse, kai

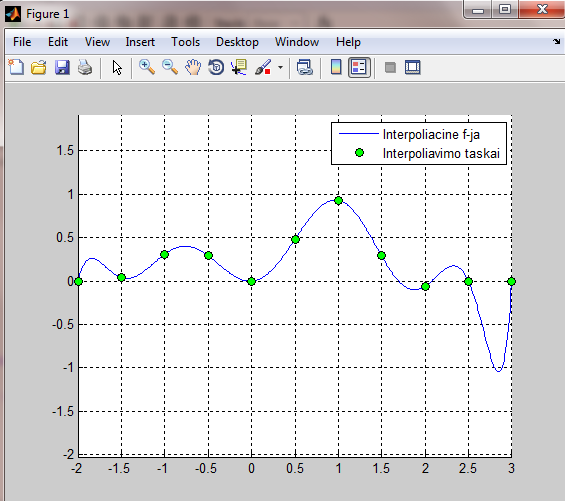
duotos interpoliavimo mazgų koordinatės:

X =[-2 -1.5 -1 -0.5 0 0.5 1 1.5 2 2.5 3]

Y =[0 0.04 0.31 0.29 0 0.48 0.93 0.29 -0.06 0 0]

**Interpoliavimas vienanarių** **bazėje**





1pav. interpoliavimo vienanarių bazėje funkcijos grafikas

Toliau pateikti *Matlab* programos rezultatai:

Interpoliavimo mazgai:

X =

Columns 1 through 9

-2.0000 -1.5000 -1.0000 -0.5000 0 0.5000 1.0000 1.5000 2.0000

Columns 10 through 11

2.5000 3.0000

Y =

Columns 1 through 9

0 0.0400 0.3100 0.2900 0 0.4800 0.9300 0.2900 -0.0600

Columns 10 through 11

0 0

Baziniu funkciju reiksmes interpoliavimo mazguose:

X1 =

1.0e+004 \*

Columns 1 through 9

0.0001 -0.0002 0.0004 -0.0008 0.0016 -0.0032 0.0064 -0.0128 0.0256

0.0001 -0.0001 0.0002 -0.0003 0.0005 -0.0008 0.0011 -0.0017 0.0026

0.0001 -0.0001 0.0001 -0.0001 0.0001 -0.0001 0.0001 -0.0001 0.0001

0.0001 -0.0001 0.0000 -0.0000 0.0000 -0.0000 0.0000 -0.0000 0.0000

0.0001 0 0 0 0 0 0 0 0

0.0001 0.0001 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000

0.0001 0.0001 0.0001 0.0001 0.0001 0.0001 0.0001 0.0001 0.0001

0.0001 0.0001 0.0002 0.0003 0.0005 0.0008 0.0011 0.0017 0.0026

0.0001 0.0002 0.0004 0.0008 0.0016 0.0032 0.0064 0.0128 0.0256

0.0001 0.0003 0.0006 0.0016 0.0039 0.0098 0.0244 0.0610 0.1526

0.0001 0.0003 0.0009 0.0027 0.0081 0.0243 0.0729 0.2187 0.6561

Columns 10 through 11

-0.0512 0.1024

-0.0038 0.0058

-0.0001 0.0001

-0.0000 0.0000

0 0

0.0000 0.0000

0.0001 0.0001

0.0038 0.0058

0.0512 0.1024

0.3815 0.9537

1.9683 5.9049

Vienanariu interpoliacines israiskos koeficientai:

A =

Columns 1 through 9

-0.0000 0.0726 1.9949 0.5630 -1.9922 -0.3904 0.7164 0.0661 -0.1035

Columns 10 through 11

-0.0013 0.0044

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

% programos kodas

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

function Vienanariu

clc, clear all, close all

format

fprintf(1, 'Interpoliavimo mazgai:');

X =[-2 -1.5 -1 -0.5 0 0.5 1 1.5 2 2.5 3]

Y =[0 0.04 0.31 0.29 0 0.48 0.93 0.29 -0.06 0 0]

Y1 = Y';

n = length(X);

X1 = zeros(n);

for i=1:n

for j=1:n

X1(i, j) = X(i).^(j-1);

end

end

fprintf(1, 'Baziniu funkciju reiksmes interpoliavimo mazguose:');

X1

fprintf(1, 'Vienanariu interpoliacines israiskos koeficientai:');

A = X1\Y1;

A = A'

x=X(1):0.01:X(numel(X));

f = fnk(x, A);

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

plot(x, f)

plot(X, Y, 'ko', 'MarkerFaceColor','g');

legend('Interpoliacine f-ja','Interpoliavimo taskai');

end

function f = fnk(x,A)

f = 0;

m = length(A);

for i=m:-1:1

f = f + A(i)\*x.^(i-1);

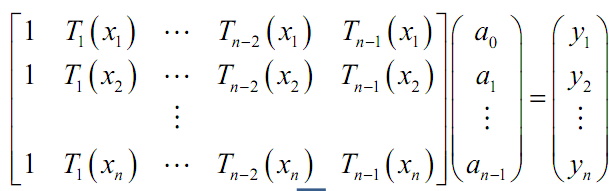
end

return

end

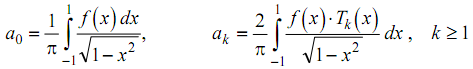
**Interpoliavimas daugianariu Čiobyševo bazėje**

Sudarome lygčių sistemą naudodami duotas koordinates:



Išsprendę šią lygčių sistemą randame bazinį koeficientų vektorių A.

Sudarome tokio pavidalo interpoliacinę funkciją:



ir braižome šios funkcijos grafiką. Žemiau pateikti Matlab programos rezultatai.

X =

Columns 1 through 9

-2.0000 -1.5000 -1.0000 -0.5000 0 0.5000 1.0000 1.5000 2.0000

Columns 10 through 11

2.5000 3.0000

Y =

Columns 1 through 9

0 0.0400 0.3100 0.2900 0 0.4800 0.9300 0.2900 -0.0600

Columns 10 through 11

0 0

T =

1.0e+007 \*

Columns 1 through 9

0.0000 -0.0000 0.0000 -0.0000 0.0000 -0.0000 0.0001 -0.0005 0.0019

0.0000 -0.0000 0.0000 -0.0000 0.0000 -0.0000 0.0000 -0.0000 0.0001

0.0000 -0.0000 0.0000 -0.0000 0.0000 -0.0000 0.0000 -0.0000 0.0000

0.0000 -0.0000 -0.0000 0.0000 -0.0000 -0.0000 0.0000 -0.0000 -0.0000

0.0000 0 -0.0000 0 0.0000 0 -0.0000 0 0.0000

0.0000 0.0000 -0.0000 -0.0000 -0.0000 0.0000 0.0000 0.0000 -0.0000

0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000

0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0001

0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0001 0.0005 0.0019

0.0000 0.0000 0.0000 0.0000 0.0000 0.0001 0.0006 0.0029 0.0139

0.0000 0.0000 0.0000 0.0000 0.0001 0.0003 0.0020 0.0114 0.0666

Columns 10 through 11

-0.0070 0.0262

-0.0003 0.0008

-0.0000 0.0000

0.0000 -0.0000

0 -0.0000

-0.0000 -0.0000

0.0000 0.0000

0.0003 0.0008

0.0070 0.0262

0.0665 0.3188

0.3881 2.2620

a =

0.0000

-0.0000

-0.0007

0.0010

0.0163

-0.0173

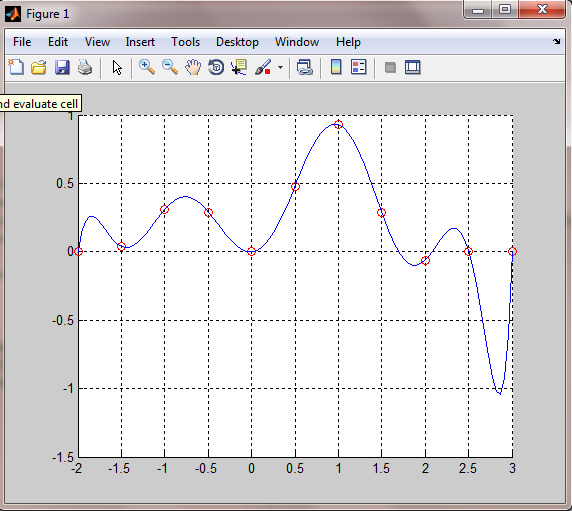
-0.1363

0.0400

0.2937

0.2864

0.4470



2pav. interpoliavimo Čiobyševo bazėje funkcijos grafikas

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

% programos kodas

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

function Ciobysevas

clc, clear all, close all

X =[-2 -1.5 -1 -0.5 0 0.5 1 1.5 2 2.5 3]

Y =[0 0.04 0.31 0.29 0 0.48 0.93 0.29 -0.06 0 0]

n=length(X); %X ilgis

T = zeros(n, n); %0 matrica iš n ilgio n plocio

T(:,1) = 1; % T matricos pirmas stulpelis bus 1

%Ciobyševo daugianariu baze

for i=1:n

x=X(i);

T(i,2)=x;

for j=3:n

T(i,j)=2\*x\*T(i,j-1)-T(i,j-2);

end

end

T % isveda T matrica

a=T\Y'; %gaunami interpoliuojancios funkcijos koeficientai

a=flipdim(a,1) % elementu sukeitimas vietomis (reverse)

figure(1), hold on, grid on

plot(X,Y,'ro') % braizomi duoti taskai

xmin=min(X);xmax=max(X); % min ir max reiksmes duotu x - intervalas

N=n\*10; % interpoliuojancios funkcijos tasku skaicius, kad gautume nekampuota grafika

XC=[xmin:(xmax-xmin)/(N-1):xmax]; % apskaiciuojam tolygiai paskirstytas x reiksmes braizyti interpoliuojanciai funkcijai

YC = klensou(a,XC); % apskaiciuoti interpoliuojancios f-jos Y reiksmes pagal xus ir koeficientus a

plot(XC,YC,'b-'); %braizoma interpoliuojanti f-ja

return

end

function px=klensou(a,x); % suranda interpoliuojancios f-jos ordinates

% KLENSOU apskaiciuoja interpoliacinio polinomo,

% uzrasyto Ciobysovo polinomo bazeje, reiksme.

% ??jimo parametrai

% a - polinomo koeficientai,

% x - argumento reik?mi? masyvas.

% I??jimo parametrai

% px - polinomo reik?mi? masyvas.

n=numel(a);

bk2=0; bk1=0;

for k=1:n

bk=a(k)+2\*x.\*bk1-bk2;

bk2=bk1; bk1=bk;

end;

px=bk-x.\*bk2;

return

end

**Antras namų darbas. Antra dalis. *Funkcijų aproksimavimas***

Duotos funkcijų išraiškos arba grafinis jų vaizdas. Reikia atlikti aproksimavimą per duotus taškus naudojant:

1. **a) Furje transformaciją** ir **b) Haro bangeles**

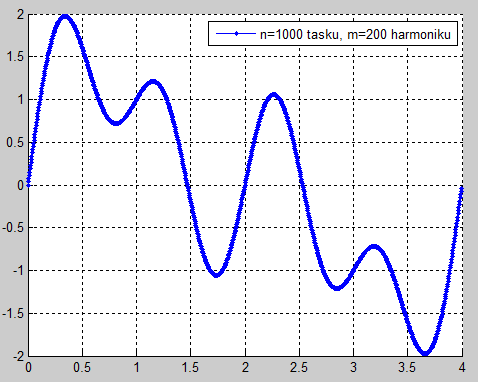
Furje transformacija

Duota funkcija 

ir triukšmas .

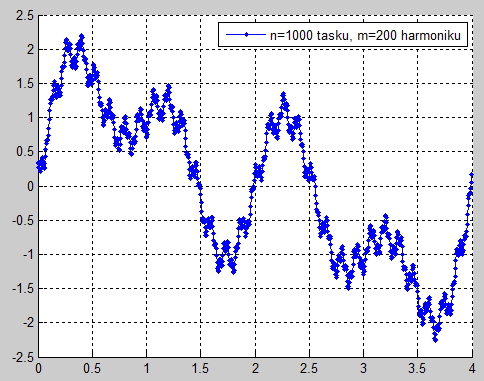
Reikia atlikti funkcijos filtravimą ir išskirti funkciją atmetant harmonines dedamasias pagal amplitudės slenksčio reikšmę ir atmetant harmonines dedamąsias pagal jų dažnį.

Žemiau pateikti funkcijos grafikai be triukšmo:



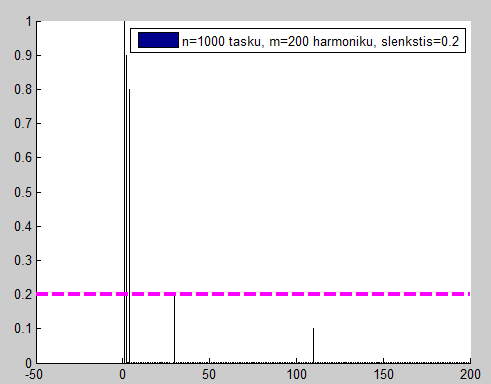
1 pav. Funkcijos grafikas be triukšmo

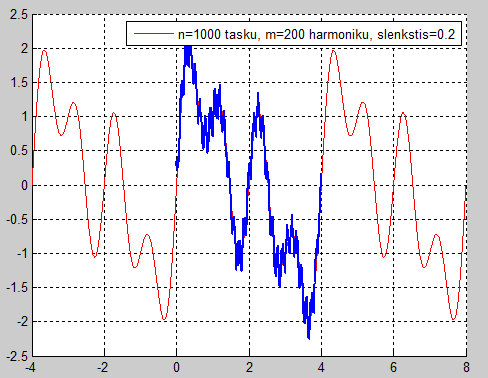
ir su triukšmu:



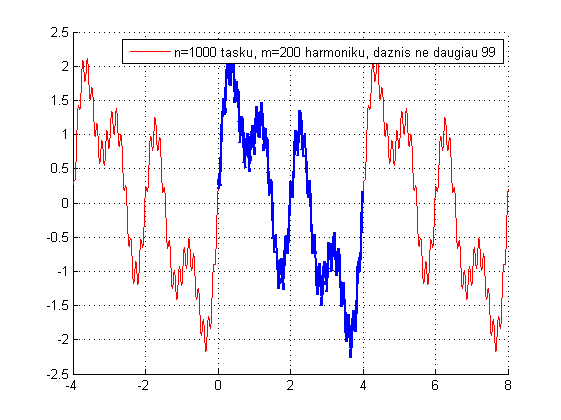
2 pav. Funkcijos grafikas su triukšmu

Atliktas filtravimas atmetant harmonikas, kurių amplitudė mažesnė už nustatytą slenksčio reikšmę 0,2. Pateikti harmonikų ir aproksimuotos funkcijos grafikai.





Atliktas filtravimas atmetant harmonikas, kurių dažnis didesnis už 99. Pateikti harmonikų ir aproksimuotos funkcijos grafikai.



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

% programos kodas

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

function Furje

clc,close all,clear all

n=1000; % n - tasku skaicius

m=200; % m - harmoniku skaicius

M=2\*m-1; % M -koeficientu skaicius

if M > n

'per didelis harmoniku skaicius!';

end

T=4; % T - duotas periodas

slenkstis=0.2 ; % harmoniku amplitudziu slenkstis triuksmu filtravimui

dt=T/n;

N=1000; % N - vaizdavimo tasku skaicius

dttt=T/N;

t=[0:dt:T-dt];

ttt=[-T:dttt:2\*T];

fff=fnk(T,t); % apskaiciuojame ir pavaizduojame duota tasku seka

figure(1),hold on,grid on,plot(t,fff,'b.-','MarkerSize',8);

legend(sprintf('n=%d tasku, m=%d harmoniku',n,m))

ac0=dot(fff,fC(0,T,t))/n;

for i=1:m-1

ac(i)=dot(fff,fC(i,T,t))\*2/n;

as(i)=dot(fff,fS(i,T,t))\*2/n;

end

figure(2),hold on

bar(0:m-1,[ac0,sqrt(ac.^2+as.^2)],0.01)

xx=axis; plot([xx(1),xx(2)],slenkstis\*[1 1],'m--','LineWidth',3); % braizo slenkscio linija

legend(sprintf('n=%d tasku, m=%d harmoniku, slenkstis=%g ',n,m,slenkstis))

fffz=ac0\*fC(0,T,ttt);

frequencies=[1:m-1];

%frequencies=[1:4];

for i=frequencies

if sqrt(ac(i)^2+as(i)^2) > slenkstis

fffz=fffz+ac(i)\*fC(i,T,ttt)+as(i)\*fS(i,T,ttt);

end

end

figure(3),hold on,grid on, plot(ttt,fffz,'r');plot(t,fff,'b-','LineWidth',2);

legend(sprintf('n=%d tasku, m=%d harmoniku, slenkstis=%g ',n,m,slenkstis))

%pagal dazni

fffz=ac0\*fC(0,T,ttt);

frequencies=[1:m-1];

%frequencies=[1:4];

for i=frequencies

if i < 99

fffz=fffz+ac(i)\*fC(i,T,ttt)+as(i)\*fS(i,T,ttt);

end

end

figure(4),hold on,grid on, plot(ttt,fffz,'r');plot(t,fff,'b-','LineWidth',2);

legend(sprintf('n=%d tasku, m=%d harmoniku, daznis ne daugiau 99 ',n,m))

return

end

function c=fC(i,T,t)

if i==0

c=1\*cos(0\*t);

else

c=cos(2\*pi\*i/T\*t);

end

return

end

function s=fS(i,T,t)

s=sin(2\*pi\*i/T\*t);

return

end

function rez=fnk(T,t)

% su triuksmais

rez=sin(2\*pi\*t/T) + 0.9\*sin(2\*pi\*2\*t/T) + 0.8\*sin(2\*pi\*4\*t/T) + 0.1\*sin(2\*pi\*110\*t/T +pi/4)+ 0.2 \* cos(2\*pi\*30\*t/T);

% be triuksmu

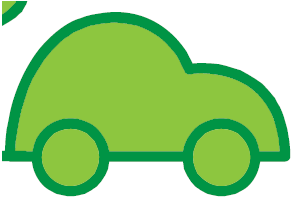
%rez=sin(2\*pi\*t/T) + 0.9\*sin(2\*pi\*2\*t/T) + 0.8\*sin(2\*pi\*4\*t/T);

return

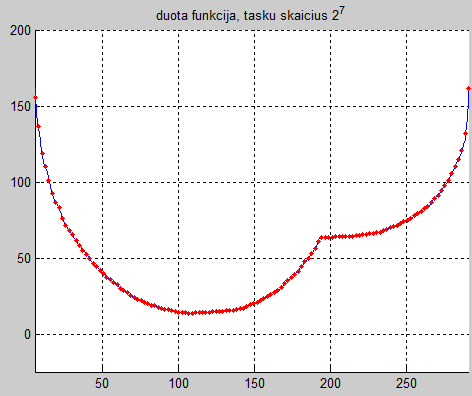
end

Haro bangelių aproksimacija

Duotas paveikslėlis :



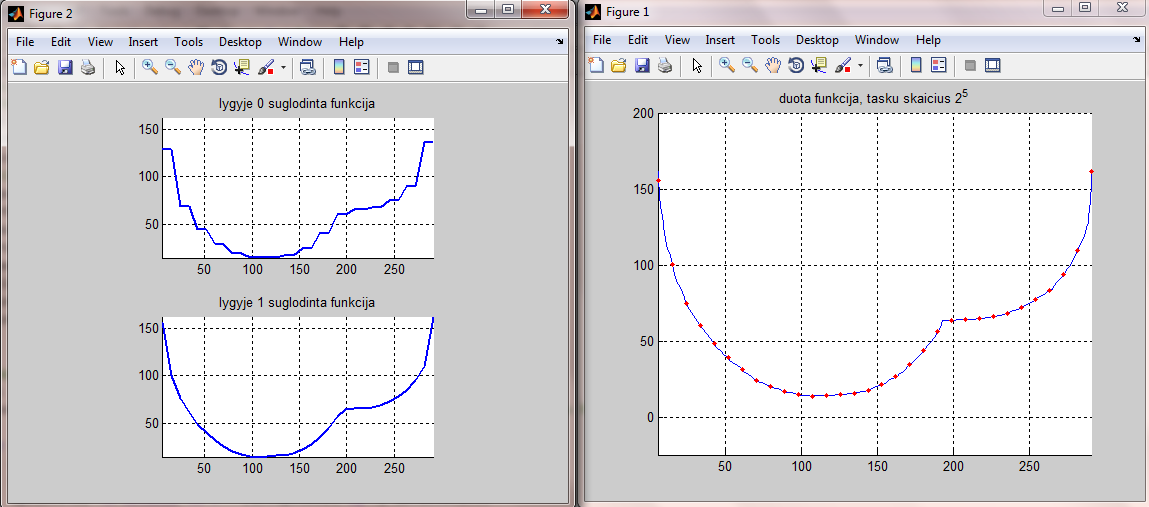
Reikia atlikti šio paveikslėlio viršutinio kontūro aproksimaciją naudojant Haro bangeles.



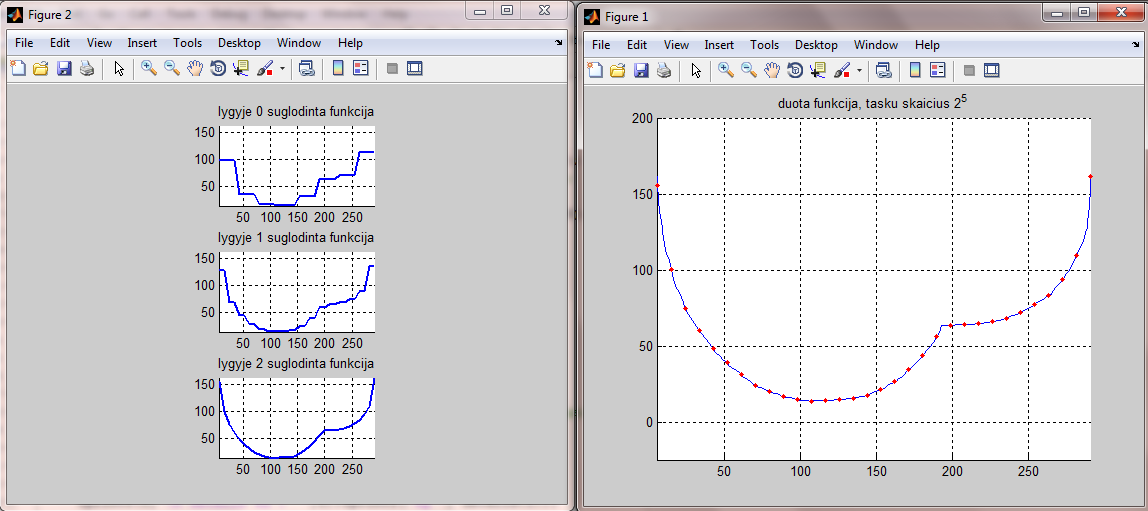
Suformuotas viršutinio kontūro grafikas

Atlikta kontūro aproksimacija Haro bangelėmis:

1 detalumo lygis:



2 detalumo lygis:



Toliau lentelėje pateikti bangelių koeficientai, kai detalumo lygių skaičius *m*=5

|  |
| --- |
| 1982.37 514.796 326.641 235.031 111.872 34.25 -26.5532 -61.3597 -183.108 -327.975 -272.479 -13.2738 -69.0554 -182.303 -379.043 -1853.38 |
| 3032.11 808.088 210.33 -99.7347 -764.774 -239.621 -366.748 -2346.53 |
| 4423.01 62.1749 -2174.4 -2929.52 |
| 5130.6 -4486.4 |
| -3973.43 |

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

% programos kodas

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

%

% Haro bangeliu aproksimacija

function main

clc;close all;clear all;

spalvos={'r-','g-','m-','c-','k-','y-','r.','g.','m.','c.','k.','y.'};

%img = imread('car.png');

%imagesc(img);

%[x,y] = ginput(200)

% Is failu ivedami duomenys:

n=5

nnn=2^n;

fclose all; fhx=fopen('zCarX.txt','r'); fhy=fopen('zCarY.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];

ts=interp1(SX,SY,t);

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

title(sprintf('duota funkcija, tasku skaicius 2^%d',n));

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

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

% Aproksimavimas Haro bangelemis:

m=5 % 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('suglodinta funkcija, detalumo lygmuo %d',n-m)};

figure(2);subplot(m+1,1,1),axis equal,axis([xmin xmax ymin ymax]); hold on,grid on, plot(SX,h,'Linewidth',2);title(sprintf('lygyje %d suglodinta funkcija',0));

for i=0:m-1 %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(m,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(m+1,1,i+2),axis equal,axis([xmin xmax ymin ymax]), hold on,grid on, plot(SX,h,'Linewidth',2);title(sprintf('lygyje %d suglodinta funkcija' ,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