# Interpoliavimas ir aproksimavimas

**Interpoliavimas daugianariu**

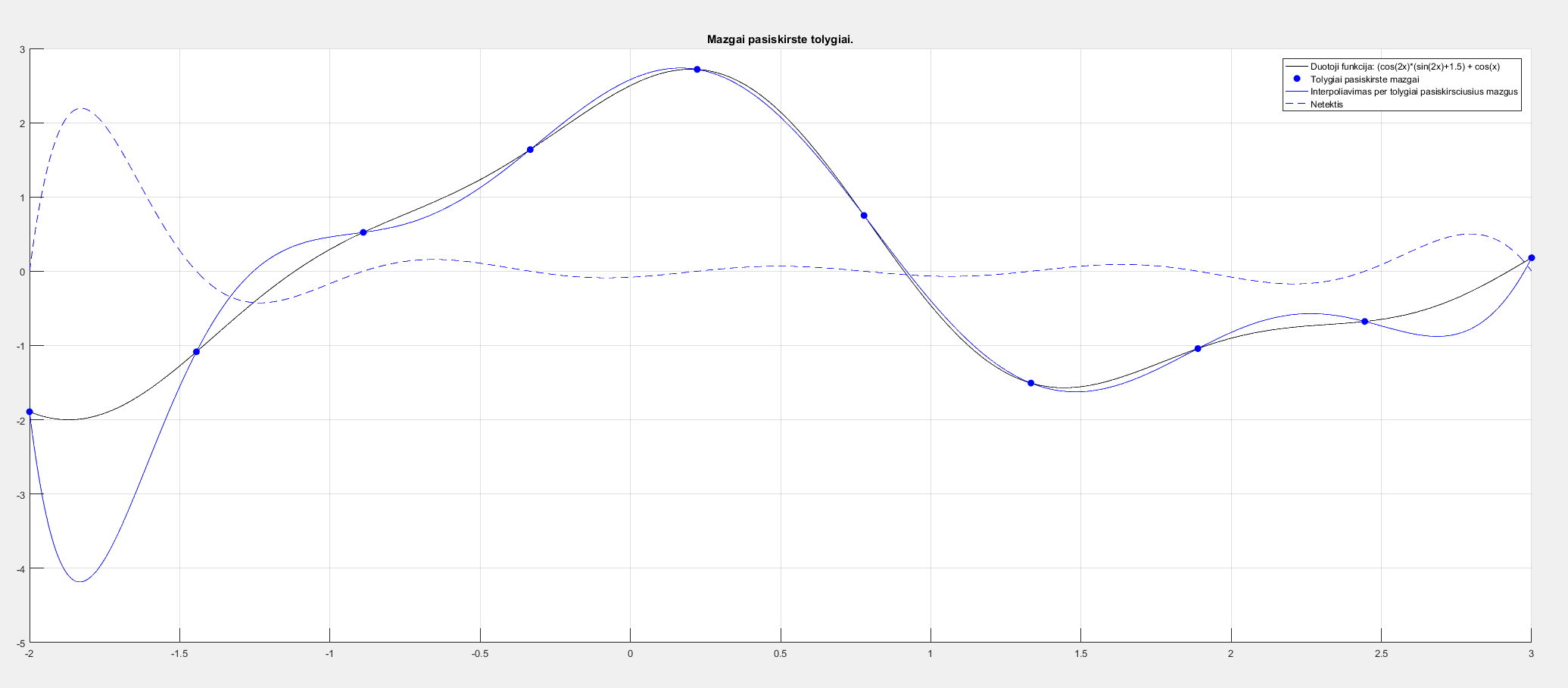
Užduotis:

Duota interpoliuojamos funkcijos analitinė išraiška. Pateikite interpoliacinės funkcijos išraišką, kai:

* Taškai pasiskirstę tolygiai.
* Taškai apskaičiuojami naudojant Čiobyševo abscises.

|  |  |  |
| --- | --- | --- |
| Var. Nr. | Funkcijos išraiška | Bazinė funkcija |
| 11 |  | Vienanarių |

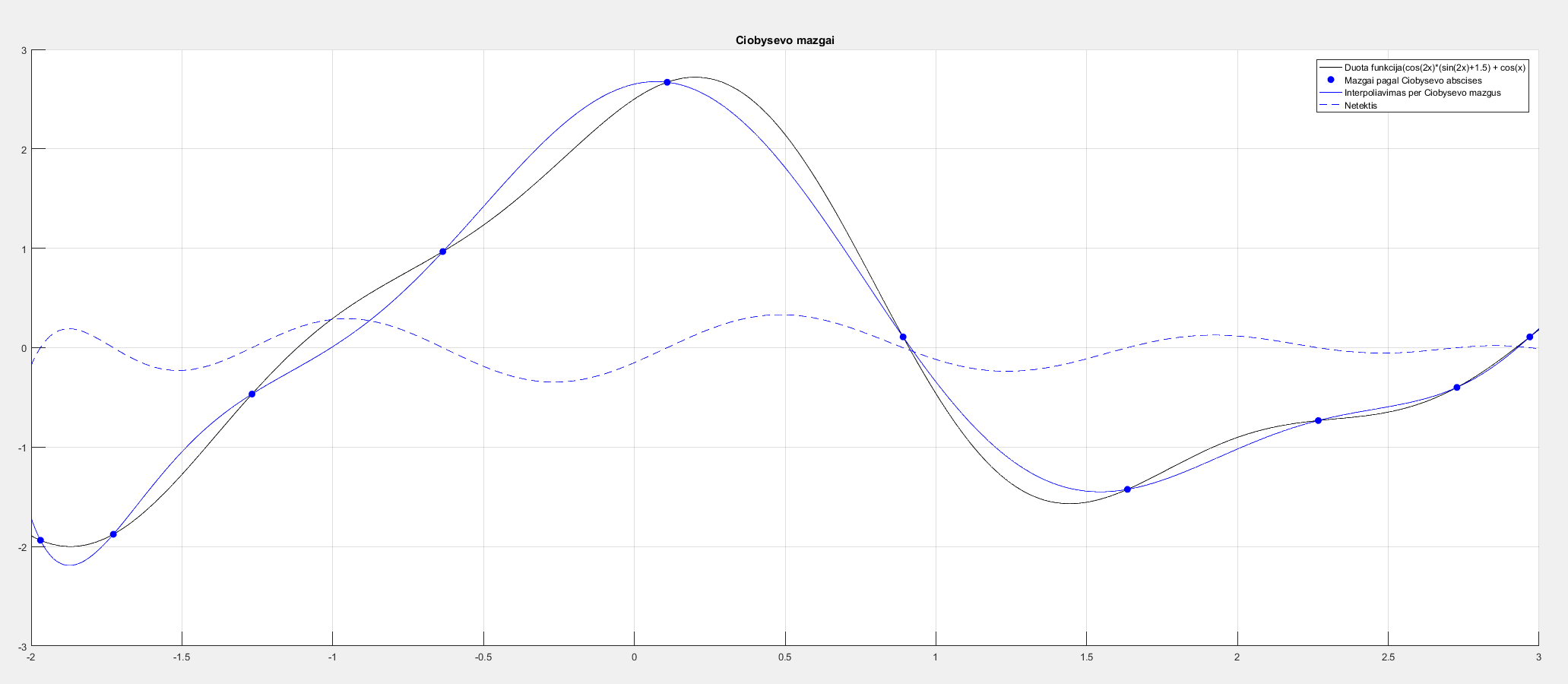
Rezultatas, kai interpoliavimo mazgai pasiskirtę tolygiai:



1 pav. Interpoliavimas per tolygiai pasiskirsčiusius mazgus

Rezultatui gauti buvo naudota 10 interpoliavimo mazgų. Didžiausias nuokrypis nuo tikrosio funkcijos buvo pastebėtas interval galuose.

Rezultatas, kai interpolivimo mazgai parinkti pagal Čiobyševo abscises:



2 pav. Interpoliavimas per Čiobyševo mazgus

Rezultatui gauti buvo naudota 10 interpoliavimo mazgų apskaičiuotų pagal Čiobyševo abscises. Visame reikšmių interavale nuokrypio amplitude išlieka pastovi.

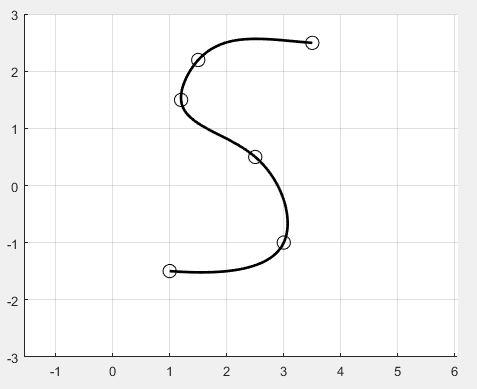
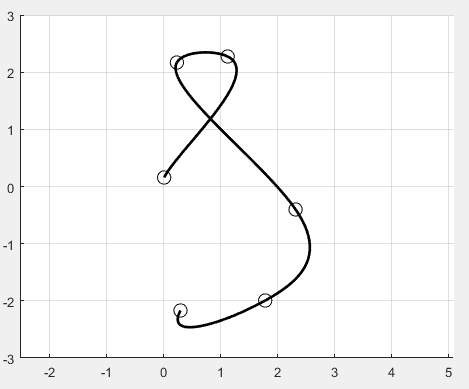
Pagal gautą rezultatą galima teigti, kad tiksleni rezultatai gaunami naudojant Čiobyševo abscises, nes buvo gautas mažesnis nuokrypis, nuo tikrosios funkcijos (tai ypač pastebima intervalų galuose).

**Parametrinis interpoliavimas**

Užduotis:

Naudodami nurodytą interpoliavimo metodą parinkti interpoliavimo taškus taip, kad interpoliuojančios kreivės suformuotų Jūsų vardo ir pavardės pirmąsias raides.

|  |  |
| --- | --- |
| Var. Nr. | Interpoliavimo metodas |
| 11 | Pirmos eilės defekto splainas |

*3 pav. Pirmos eilės defekto splainai*

Abu splainai turi po 6 taškus. Kairiame paveiksliuke pavaizuota spausdintinė S raidė, o dešinėje pusėje rašytinė.

**Diskrečioji Furje aproksimacija**

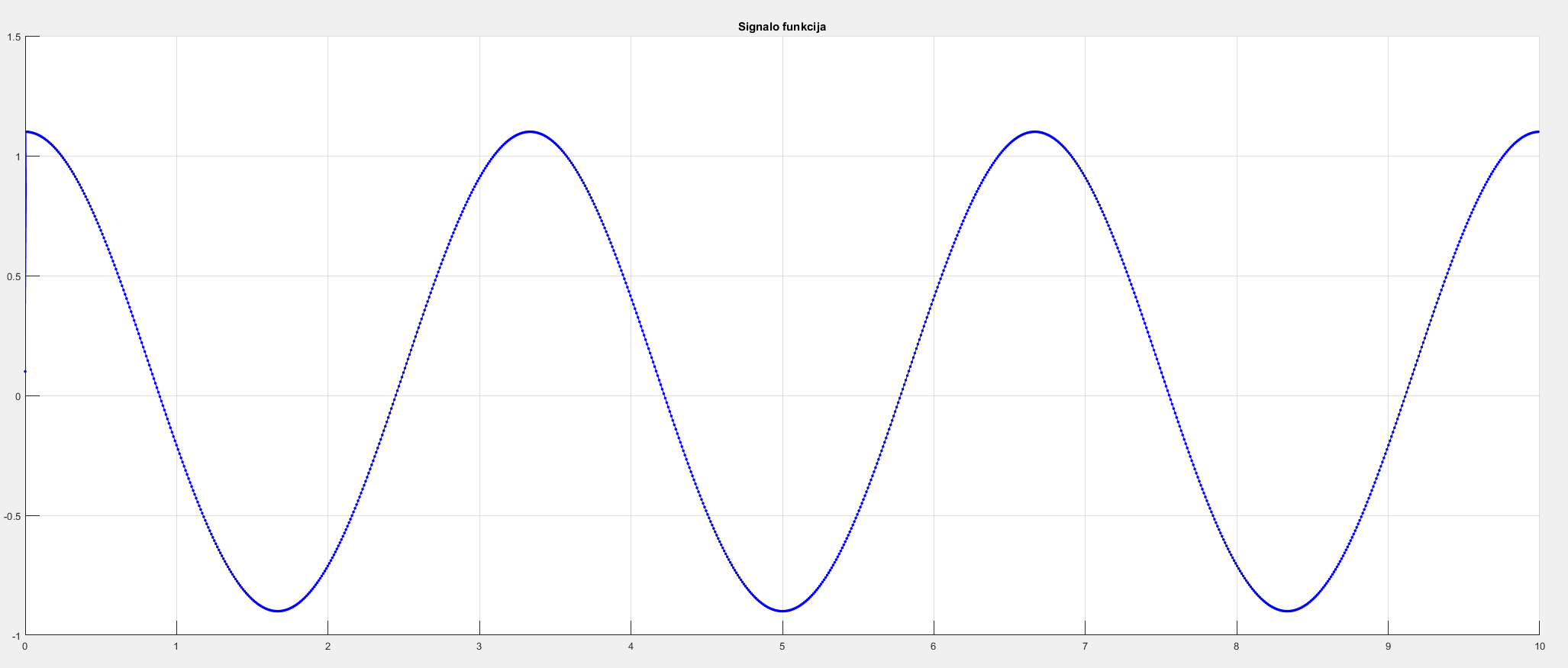
Užduotis:

Duota analitinė periodinės funkcijos F(t) = G(t) + R(t) išraiška, kur G(t) –signalas, R(t) – triukšmas. Reikia atlikti funkcijos F(t) aproksimavimą diskrečiąja Furje transformacija ir išskirti G(t) dviem būdais:

* atmetant harmonines dedamąsias pagal amplitudės slenkstį;
* atmetant harmonines dedamąsias pagal dažnio slenkstį.

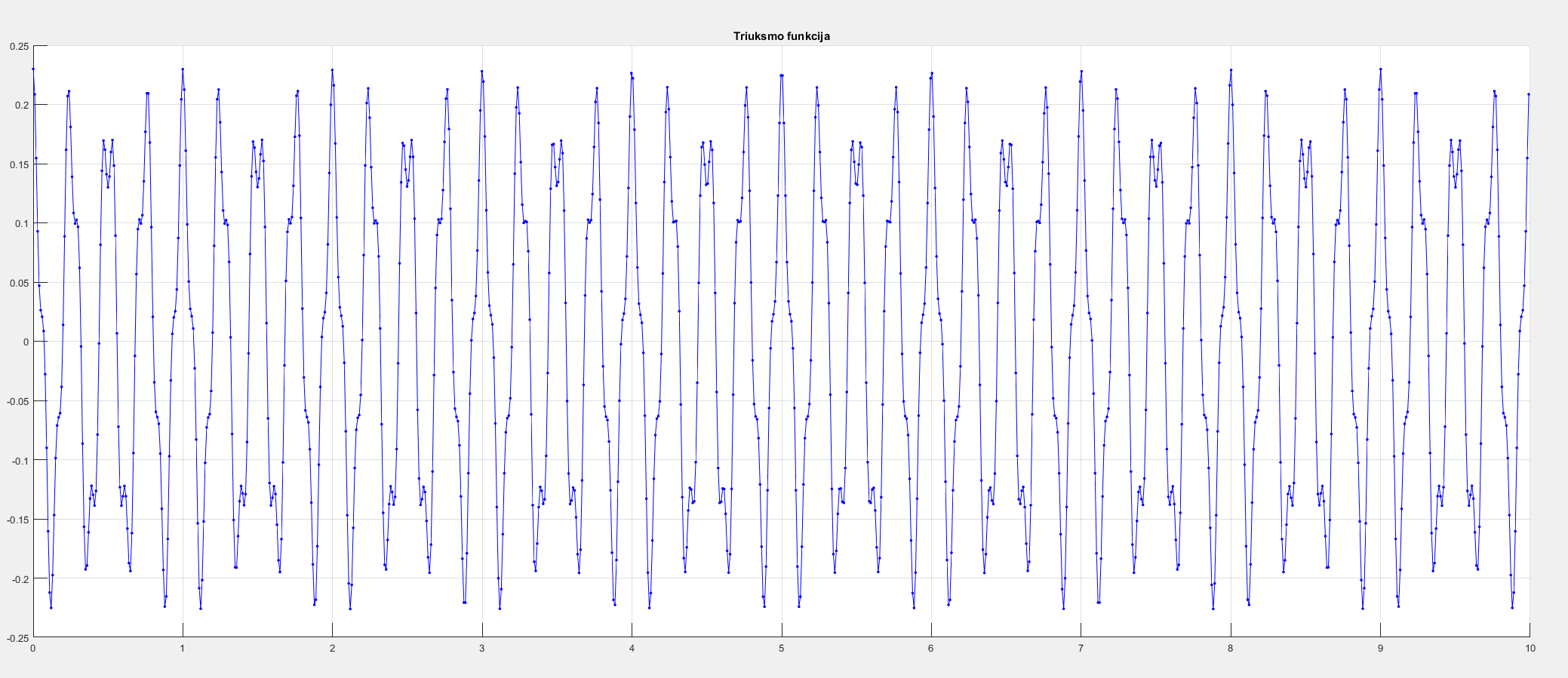
|  |  |
| --- | --- |
| Var. Nr. | Diskrečajai Furje transformacijai |
| 11 |  |
|  |

Signalo funkcija:



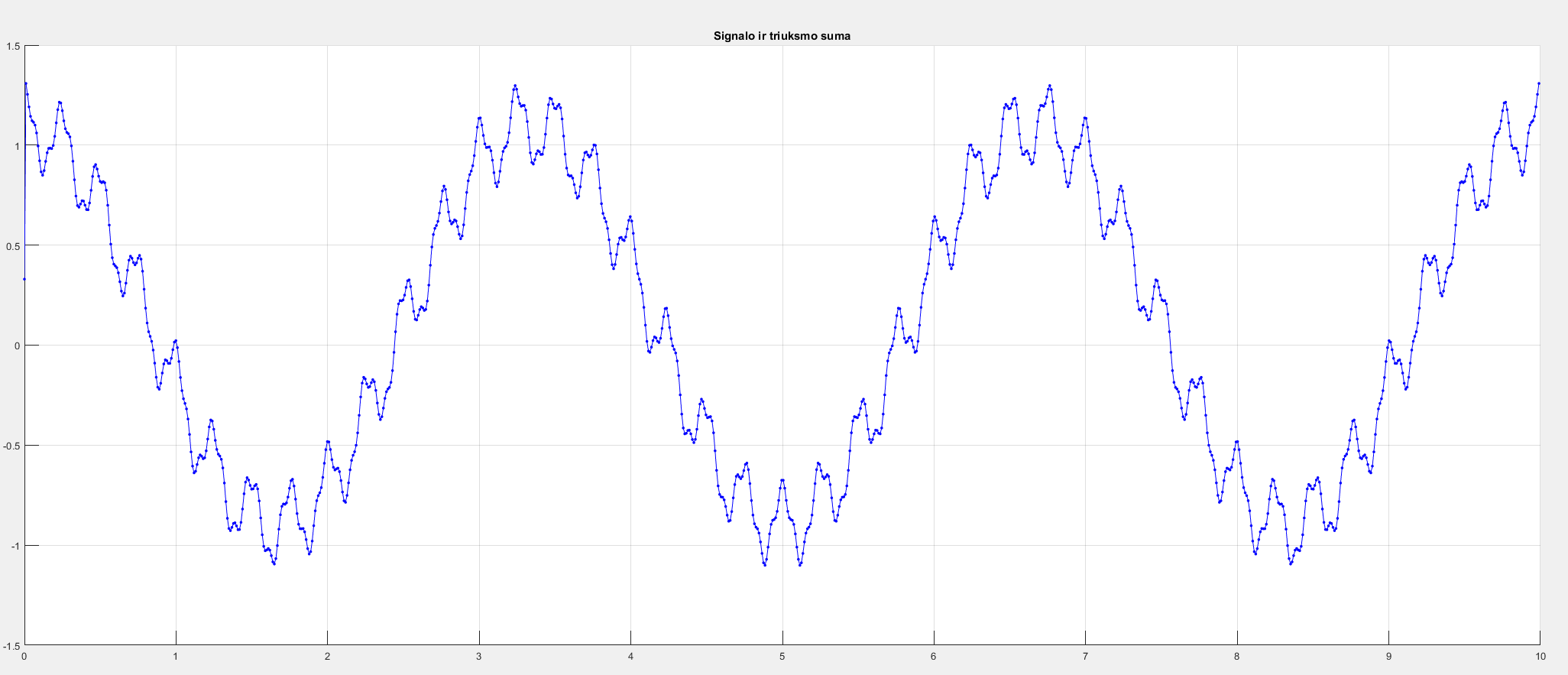
4 pav. Signalo funkcija G(t)

Triukšmo funkcija:



5 pav. Triukšmo funkcija R(t)

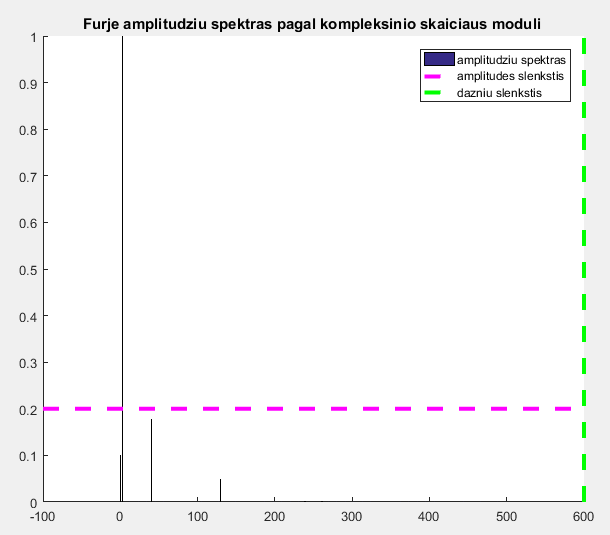
Signalo ir triukšmo funkcija:



6 pav. Signalo ir triukšmo funkcijos suma F(t)=G(t)+R(t)

**Funkcijos aproksimavimas pagal amplitudės slenkstį:**

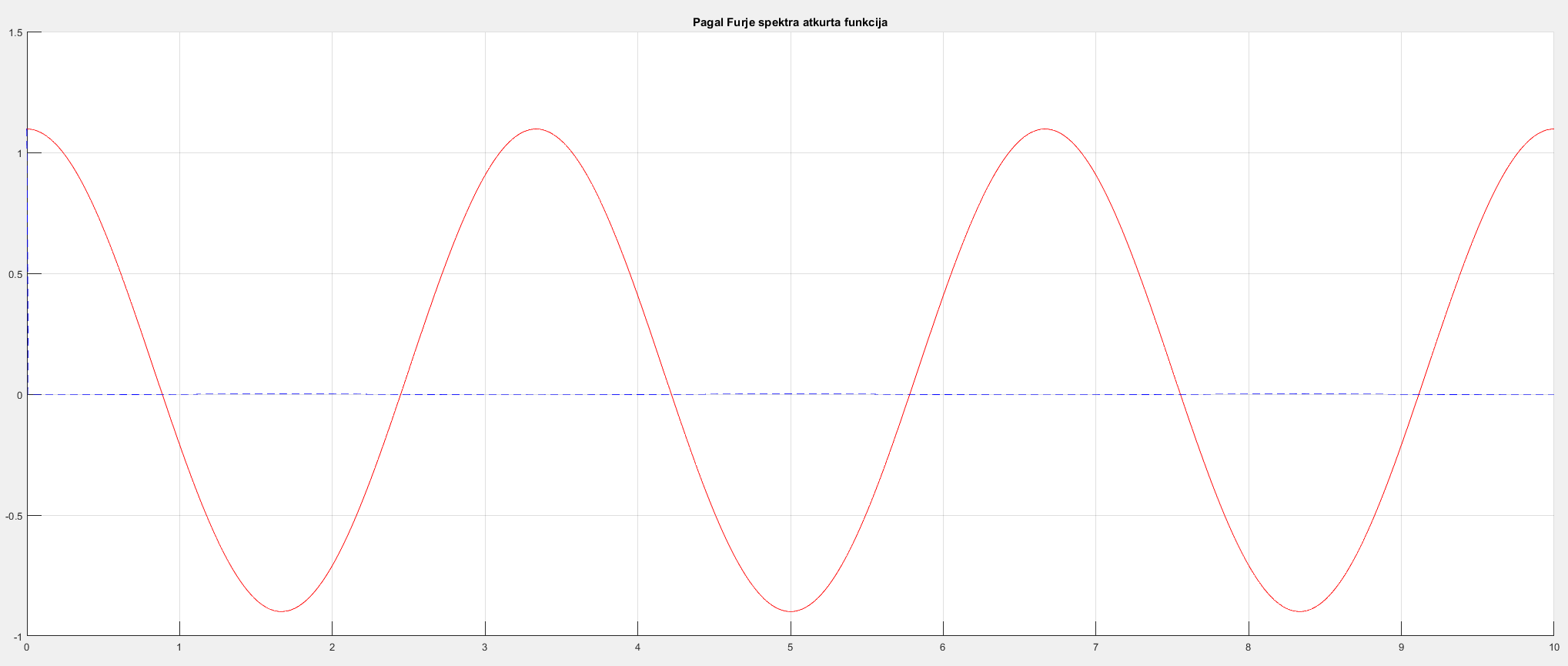
Amplitudės slenkstis 0.2.



7 pav. Pasirinktas amplitudės slenkstis

Amplitudės slenkstis parenkamas toks, kad būtų atmetami svyravimai turintys mažiausią amplitudę, nes triukšmo funkcijos amplitudė yra maža. Šiuo atveju buvo parinktas slenkstis su reikšme 0.2

Rezultatas:

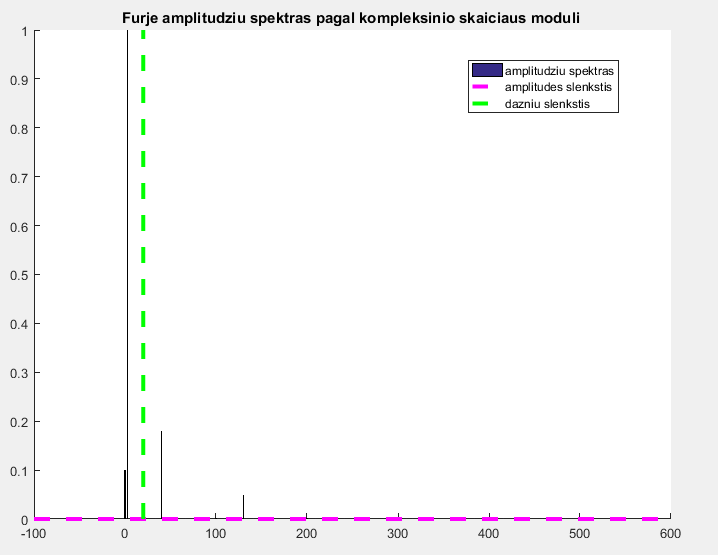


8 pav. Pagal amplitudės slenkstį aproksimuotas signalas

Gauta aproksimuota funkcija turi minimalų nuokrypį nuo tikrojo signalo.

**Funkcijos aproksimavimas pagal dažnio slenkstį:**

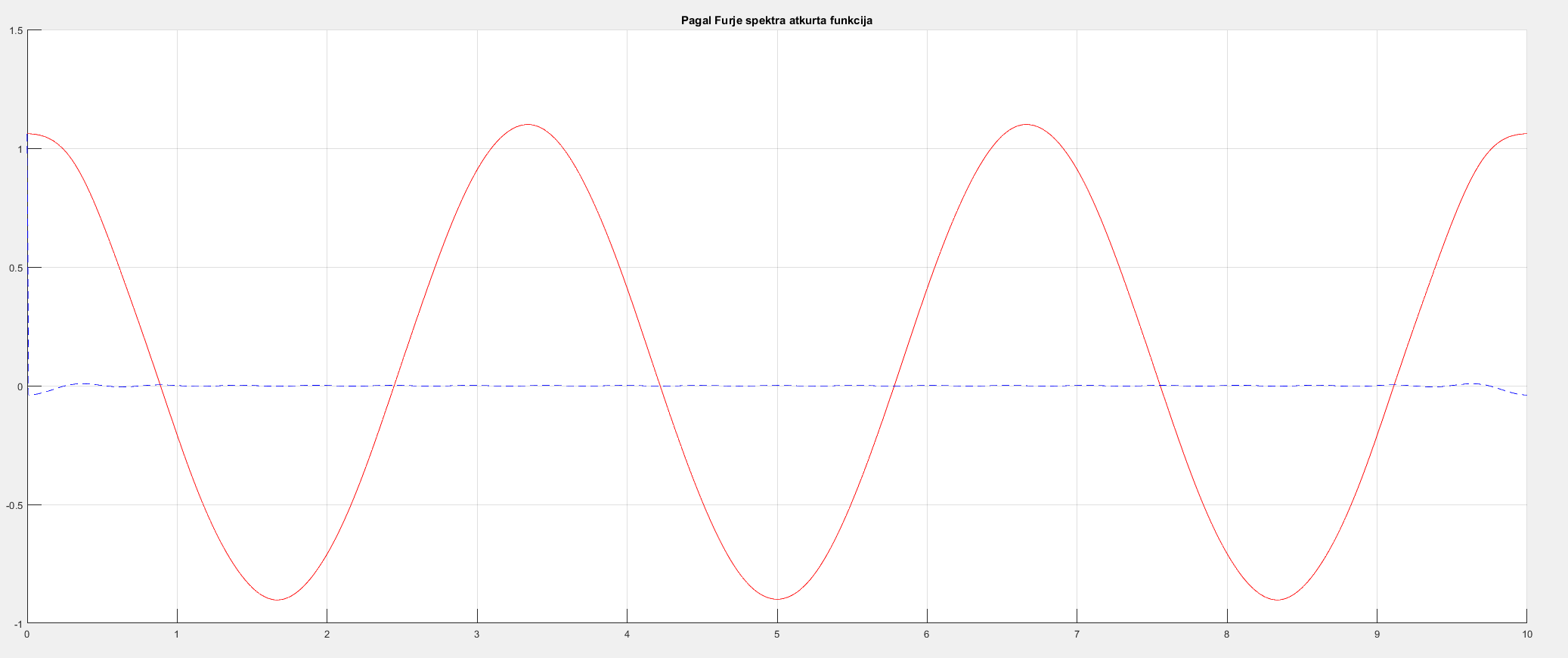
Dažnių slenkstis yra kygus 20.



9 pav. Pasirinktas dažnio slenkstis

Dažnio slenkstį reikia parinkti tokį, kad būtų atmetamos reikšmės turinčios aukštą dažnį, nes triukšmo funkcija įprastai turi aukštą dažnį.

Rezultatas:



10 pav. Pagal dažnio slenkstį aproksimuotas signalas

Aproksimuojant pagal dažnio slenkstį gautas rezultatas turi minimalų nuokrypį nuo tikrojo signalo, didžiausias nuokrypis pastebimas signalo intervalo pradžioje.

Aproksimuojant pagal amplitudės ir dažnio slekščius buvo gauti panašūs rezultatai.

**Programų tekstai**

Interpoliavimas daugianariu:

% Saulius Stankevicius IFF-4/2

function Interpoliavimas

clc, close all;

f=@(x)cos(2\*x).\*(sin(2\*x) + 1.5) + cos(x);

f\_sym = '(cos(2x)\*(sin(2x)+1.5) + cos(x)';

minX = -2;

maxX = 3;

N = 10;

X = [minX:(maxX-minX)/(N-1):maxX]; % tolygiai paskirstytu interpoliavimo tasku abscises

k=[0:N-1];

XC=(maxX+minX)/2+(maxX-minX)/2\*cos((2\*k+1)\*pi/(2\*N)); % "Ciobysevo abscises"

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

M = ones(N);

MC = ones(N);

for j=1: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).'; % vienanariu bazes koeficientai

AC = MC\f(XC).'; % vienanariu bazes koeficientai

Y = [];

YC = [];

Yn = [];

YCn = [];

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];

Yn = [Yn; f(x(j))-y];

YCn = [YCn; f(x(j))-yc];

end

legend1={['Duotoji funkcija: ', f\_sym],...

'Tolygiai pasiskirste mazgai',...

'Interpoliavimas per tolygiai pasiskirsciusius mazgus',...

'Netektis'};

legend2={['Duota funkcija', f\_sym],...

'Mazgai pagal Ciobysevo abscises',...

'Interpoliavimas per Ciobysevo mazgus',...

'Netektis'};

figure(1), hold on, grid on;

title('Mazgai pasiskirste tolygiai.');

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

plot(X,f(X),'bo','MarkerFaceColor','b','MarkerSize',6); % mazgai

plot(x, Y, 'b-'); % interpoliavimas

plot(x,Yn,'b--'), % netektis

legend(legend1);

figure(2), hold on, grid on;

title('Ciobysevo mazgai');

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

plot(XC, f(XC),'bo','MarkerFaceColor','b','MarkerSize',6) % mazgai

plot(x, YC, 'b-'); % interpoliavimas

plot(x,YCn,'b--'), % netektis

legend(legend2);

end

Parametrinis interpoliavimas:

function main

clc,close all

hL=[]; % busimu objektu valdikliu masyvas

f=figure; hold on; grid on

%X=[3.5 1.5 1.2 2.5 3 1 ]

%Y=[2.5 2.2 1.5 0.5 -1 -1.5 ]

X=[0 1.2 0.2 2.3 1.8 0.3]

Y=[0.1 2.3 2.1 -0.4 -2 -2.2]

nP=length(X);

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

t

figure(1);axis([-3,3,-3,3]);axis equal;hold on;

% vaizduojame duotus taskus

for i=1:nP,

h(i)=plot(X(i), Y(i),'ko','ButtonDownFcn',@startDragFcn,'MarkerSize',10);

% kas atliekama paspaudus peles klavisa, nurodoma funkcijoje startDragFcn

% tasku objektu valdikliai issaugomi masyve h

end

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

splainu\_parametrinis\_interpoliavimas(X,Y,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 draggingFcn(varargin)

% apraso, kas atliekama, kai pakinta pele valdomo objekto padetis

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

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

X(find(gco == h))=pt(1,1);

Y(find(gco == h))=pt(1,2);

% kvieciame savo sukurta funkcija interpoliuojanciai kreivei apskaiciuoti:

splainu\_parametrinis\_interpoliavimas(X,Y,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 splainu\_parametrinis\_interpoliavimas(X,Y,t)

nP=length(X) % interpoliavimo tasku skaicius

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

DDFX=splaino\_koeficientai(t,X);

DDFY=splaino\_koeficientai(t,Y);

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

nnn=100;

[SX,sss]=splainas(t(iii:iii+1),X(iii:iii+1),DDFX(iii:iii+1),nnn);

[SY,sss]=splainas(t(iii:iii+1),Y(iii:iii+1),DDFY(iii:iii+1),nnn);

hL(iii)=plot(SX,SY,'k-','LineWidth',2,'MarkerSize',8)

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

% splaino intervalu objektu valdikliai issaugomi masyve hL

return

end

function DDF=splaino\_koeficientai(X,Y)

% apskaiciuojamos antros isvestines splaino mazguose

% iopt=1 - periodinis splainas

n=length(X);

A=zeros(n);b=zeros(n,1);

d=X(2:n)-X(1:(n-1));

for i=1:n-2

A(i,i:i+2)=[d(i)/6, (d(i)+d(i+1))/3,d(i+1)/6];

b(i)=(Y(i+2)-Y(i+1))/d(i+1)-(Y(i+1)-Y(i))/d(i);

end

A(n-1,[1,2,n-1,n])=[d(1)/3, d(1)/6, d(n-1)/6,d(n-1)/3];

A(n,[1,n])=[1,-1];

b(n-1)=(Y(2)-Y(1))/d(1)-(Y(n)-Y(n-1))/d(n-1);

DDF=A\b;

return

end

function [S,sss]=splainas(X,Y,DDF,nnn)

% splaino intervale tarp dvieju tasku apskaiciavimas

% nnn - vaizdavimo tzku skaicius

% S - splaino reiksmes

% sss - vaizdavimo abscises

d=X(2)-X(1);

sss=X(1):(X(2)-X(1))/(nnn-1):X(2);

S=DDF(1)/2\*(sss-X(1)).^2+(DDF(2)-DDF(1))/(6\*d)\*(sss-X(1)).^3+(sss-X(1))\*((Y(2)-Y(1))/d-DDF(1)\*d/3-DDF(2)\*d/6) +Y(1);

return

end

end

Diskrečioji Furje aproksimacija:

function main

clc,close all,clear all

n=1000; n=round(n/2)\*2+1 % tasku skaicius, nelyginis

m=(n+1)/2 % m - harmoniku skaicius

% m=100

T=10;

dazniu\_slenkstis=600;

ampl\_slenkstis=0.2;

dt=T/n

N=1000 % vaizdavimo tasku skaicius

dttt=T/N

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

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

% Mano funkcijos

G=@(t)sign(2\*pi\*t/T).\*cos(2\*pi\*3\*t/T)+0.1; % Signalas

R=@(t)0.05\*cos(2\*pi\*130\*t/T)+0.18\*cos(2\*pi\*40\*t/T); % Triuksmas

F=@(t)G(t)+R(t); % Gautas signalas

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

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

legend(sprintf('Vienas funkcijos periodas n=%d tasku',n))

title('Duotoji funkcija')

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

ac,as

figure(2),hold on

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

xx=axis;

plot(xx(1:2),ampl\_slenkstis\*[1 1],'m--','LineWidth',3); % braizo ampl slenkscio linija

plot(dazniu\_slenkstis\*[1 1],xx(3:4),'g--','LineWidth',3); % braizo dazniu slenkscio linija

title('Furje amplitudziu spektras pagal kompleksinio skaiciaus moduli ')

legend({'amplitudziu spektras';'amplitudes slenkstis';'dazniu slenkstis'})

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

frequencies=[1:m-1];

frequencies=frequencies(find(frequencies < dazniu\_slenkstis))

for i=frequencies

if sqrt(ac(i)^2+as(i)^2) > ampl\_slenkstis

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

end

end

fSignalas = G(ttt);

figure(3),hold on,grid on, plot(ttt,fffz,'r'); % Aproksimuota funkcija

%plot(t,fff,'b-','LineWidth',1); % Signalas su triuksmu

plot(ttt,fffz-fSignalas,'b--'); % Netiktis

xlim([0; 10]);

title(sprintf('Pagal Furje spektra atkurta funkcija'))

% Mano funkciju atvaizdavimas

fF=F(t);

figure(4),hold on,grid on,plot(t,fF,'b.-','MarkerSize',8);

title(sprintf('Signalo ir triuksmo suma'))

fG=G(t);

figure(5),hold on,grid on,plot(t,fG,'b.-','MarkerSize',8);

title(sprintf('Signalo funkcija'))

fR=R(t);

figure(6),hold on,grid on,plot(t,fR,'b.-','MarkerSize',8);

title(sprintf('Triuksmo funkcija'))

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), rez=sign(2\*pi\*t/T).\*cos(2\*pi\*3\*t/T)+0.1 + 0.05\*cos(2\*pi\*130\*t./T)+0.18\*cos(2\*pi\*40\*t./T); return, end