Darbą atliko: Kastytis Kaškonas

Grupė: IFK-1

Varianto nr. 14

# Interpoliavimas daugianariu

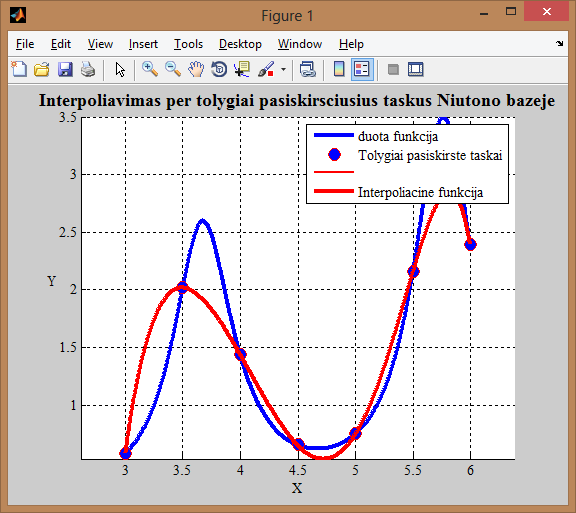
ln(x)/(sin(3\*x)+1,5); 3 x 7

Interpoliavimo taškų skaičius: 7

Interpoliavimo mazgai:

X= 3.00 3.50 4.00 4.50 5.00 5.50 6.00

Y= 0.57 2.02 1.44 0.65 0.75 2.16 2.39



Programos kodas

function NiutonasA

clc

clear all

close all

xmin= 3; %Intervalo pradzia

xmax= 6; %Intervalo pabaiga

n=7; %Interpoliavimo taskus skaicius

fprintf(1, 'Interpoliavimo tasku skaicius: %g', n);

%============================================================

X=[xmin:(xmax-xmin)/(n-1):xmax]; %Tolygiai paskirstyti taskai (x asis)

fprintf(1, '\nTolygiai paskirstyti taskai (x asis): \n');

fprintf(1, ' %g ', X);

Y=fnk(X); %Tolygiai paskirstyti taskai (y asis)

fprintf(1, '\nTolygiai paskirstyti taskai (y asis): \n');

fprintf(1, ' %g ', Y);

%===============================================================

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

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

plot(x,fnk(x), 'b-', 'LineWidth', 3) %Pradine funkcija

%============================================================

n=length(X);set(gca,'Fontname','Times New Roman Baltic');

fprintf('\n');

fprintf('\*\*\*Interpoliavimas per tolygiai pasiskirsciusius ta?kus Niutono baz?je\*\*\*\n\n')

fprintf('Interpoliavimo mazgai:\n')

fprintf('\nX= ')

for i=1:n

fprintf('\t%4.2f',X(i))

end

fprintf('\nY= ')

for i=1:n

fprintf('\t%4.2f',Y(i))

end

fprintf('\n')

xx=zeros(n,n);

xx(:,1)=1;

for j=2:n

for i=j:n

san=1;

for k=1:j-1

san=san\*(X(i)-X(k));

xx(i,j)=san;

end

end

end

xx;

fprintf('\nBaziniø funkcijø reik?m?s interpoliavimo mazguose:\n\n')

for i=1:n

for j=1:n

fprintf('\t%9.4f',xx(i,j));

end

fprintf('\n')

end

A=inv(xx)\*Y';

fprintf('\nNiutono interpoliacin?s i?rai?kos koeficientai:\n\n');

for i=1:n

fprintf('\t%9.4f',A(i))

end

fprintf('\n')

%=================================================================

title('Interpoliuota pagal tolygiai pasiskirsciusius taskus')

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

f=A(1);

for i=2:n

sand=1;

for k=1:i-1

sand=sand.\*(x-X(k));

end

f=f+A(i).\*sand;

end

%===============================================================

plot(X,Y,'o','MarkerEdgeColor','r','MarkerFaceColor','b','MarkerSize',10);

hold on; grid on; plot(x,f,'r','LineWidth',2);

set(gca,'Fontname','Times New Roman Baltic');xlabel('X');ylabel('Y');

set(get(gca,'YLabel'),'Rotation',0.0);

title('Interpoliavimas per tolygiai pasiskirsciusius taskus Niutono bazeje','FontWeight','Bold','Fontsize',14);

legend('Pradine funkcija','Interpoliavimo mazgai','Interpoliacine funkcija','Location','NorthWest')

%====================================================

plot(x, f, 'r-', 'LineWidth', 3) %Braizoma funkcija interpoliuota pagal tolygiai paskirstytus taskus

legend({'duota funkcija','Tolygiai pasiskirste taskai','','Interpoliacine funkcija'})

end

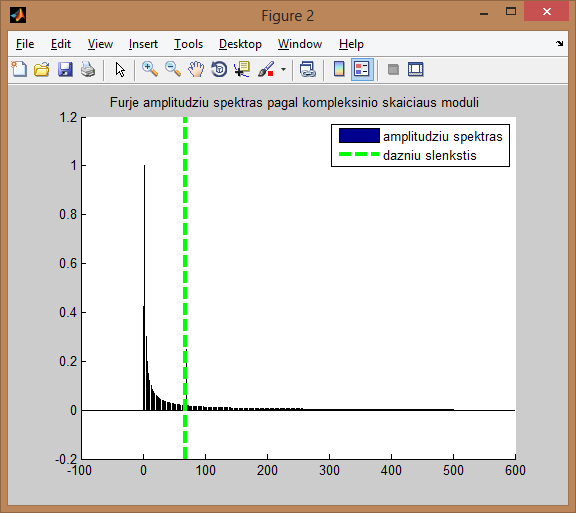
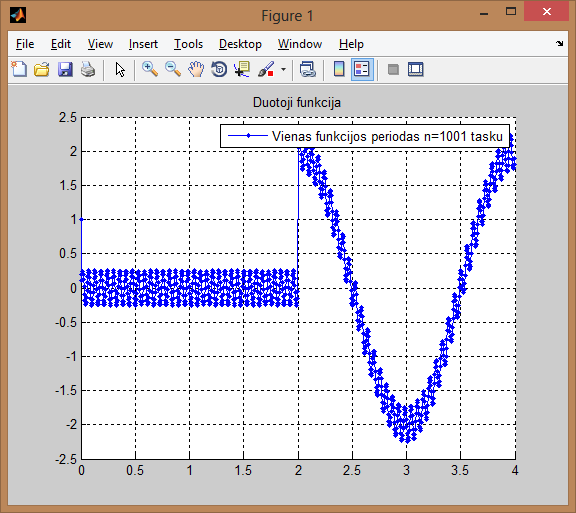
%======================================================

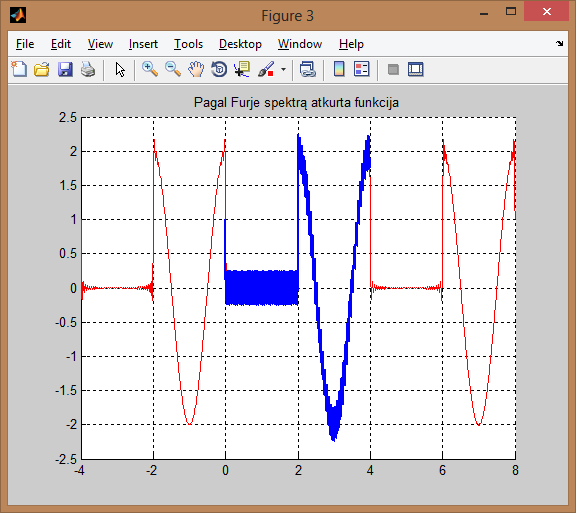
function f=fnk(x)

f=log(x)./(sin(3\*x)+1.5);

return, end

# Furjė aproksimacija





Programos kodas

%

% Viename periode aprasytai funkcijai atliekama diskrecioji Furje

% transformacija

%

function furje

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=4;

dazniu\_slenkstis=69 %m/2 % harmoniku numeriu slenkstis triuksmu filtravimui

ampl\_slenkstis=0 %0.6 % harmoniku amplitudziu slenkstis triuksmu filtravimui

dt=T/n

N=1000 % vaizdavimo tasku skaicius

dttt=T/N

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

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

% disp('kontrole:'),disp(sum(fC(3,T,t).\*fC(0,T,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('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';'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

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

%legend(sprintf('n=%d tasku, m=%d harmoniku, a-slenkstis=%g d-slenkstis=%g',n,m,ampl\_slenkstis,dazniu\_slenkstis))

title(sprintf('Pagal Furje spektrą atkurta 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=(1-sign(sin(2\*pi\*t/T))).\*cos(2\*pi\*2\*t/T)+0.25\*sin(2\*pi\*70\*t/T); return, end

function rez=fnk(T,t), rez=(1-sign(sin(2\*pi\*t/T))).\*cos(2\*pi\*2\*t/T); return, end %be triuksmo