**Užduoties nr: 19.**

**Autorius: Dovydas Loda IFF-1**

**Užduotis**

1. Atlikti interpoliavimą pirmos eilės defekto periodiniais splainiais

Kai duotos X ir Y funkcijos. Interpoliavimo taškų skaičius 7.

X(t) = 2\*cos(t+PI/8), Y=0.3\*sin(t)

1. Atlikti aproksimavimą daugianariu.

Kai duota funkcija:

e^(-1\*x^2)\*sin(x^2)\*(x+2). X kinta nuo [-2 ;2]

1. **Pirmos eilės defekto periodiniai splainiai**

Programos darbo rezultai:

Taškai:

funkX =

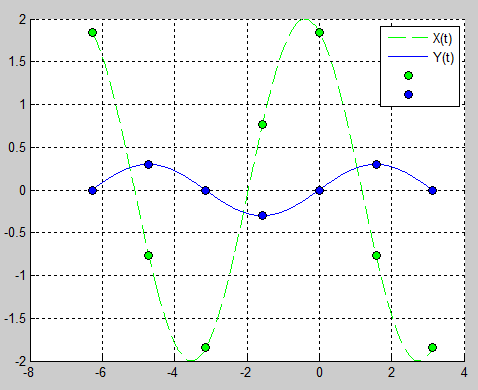
1.8478 -0.7654 -1.8478 0.7654 1.8478 -0.7654 -1.8478

funkY =

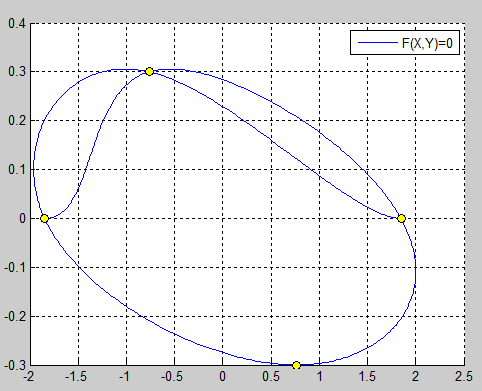
0.0000 0.3000 -0.0000 -0.3000 0 0.3000 0.0000

„Laiko momentas“ t:

T = -6.3 -4.7 -3.1 -1.6 0 1.6 3.1



X=X(t) ir Y=Y(t) grafikas



F(X, Y)=0 grafikas

Programos kodas:

function Periodinis\_pirmos\_eiles\_defekto

clc, close all, clear all

xmin=-2\*pi; xmax=pi; n=7;

t=[xmin:(xmax-xmin)/(n-1):xmax];

X=funkcijaX(t);

Y=funkcijaY(t);

fprintf('T = ');

fprintf('%2.2g ',t);

fprintf('\n');

figure(2), hold on, grid on

tt=[xmin:(xmax-xmin)/1000:xmax];

plot(tt,funkcijaX(tt),'g--');

plot(tt,funkcijaY(tt),'b-');

plot(t,X,'ko','MarkerFaceColor','g');

plot(t,Y,'ko','MarkerFaceColor','b');

legend({'X(t)','Y(t)','',''})

figure(1), hold on, grid on

DDFX=splaino\_koeficientai(t,X);

DDFY=splaino\_koeficientai(t,Y);

for j=1:n-1

SplainoX=splainas(t(j:j+1),X(j:j+1),DDFX(j:j+1));

SplainoY=splainas(t(j:j+1),Y(j:j+1),DDFY(j:j+1));

plot(SplainoX,SplainoY)

end

plot(X,Y,'ko','MarkerFaceColor','y')

legend({'F(X,Y)=0'})

return

end

function S=splainas(X,Y,DDF)

nnn=100;

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

xxx=X(1):d/(nnn-1):X(2);

s=xxx-X(1);

S=DDF(1)\*(s.^2/2-s.^3/(6\*d)) + DDF(2)\*s.^3/(6\*d)+...

((Y(2)-Y(1))/d-DDF(1)\*d/3-DDF(2)\*d/6)\*s+...

Y(1);

return

end

function DDF=splaino\_koeficientai(X,Y)

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 funkX=funkcijaX(x)

funkX= 2\*cos(x+pi/8) % (x/5).\*sin(x);

return, end

function funkY=funkcijaY(x)

funkY= 0.3\*sin(x) %(x/5).\*cos(x);

return, end

1. **Aproksimavimas daugianariais**

Programos darbo rezultatai:

SX =

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

SY =

0 0.0410 0.3096 0.2890 0 0.4817 0.9287 0.2870 -0.0554

ans =

-3.0000 3.0000 -0.5554 1.4287

ans =

-3.0000 3.0000 -0.5554 1.4287

ans =

-3.0000 3.0000 -0.5554 1.4287

ans =

-3.0000 3.0000 -0.5554 1.4287

ans =

-3.0000 3.0000 -0.5554 1.4287

ans =

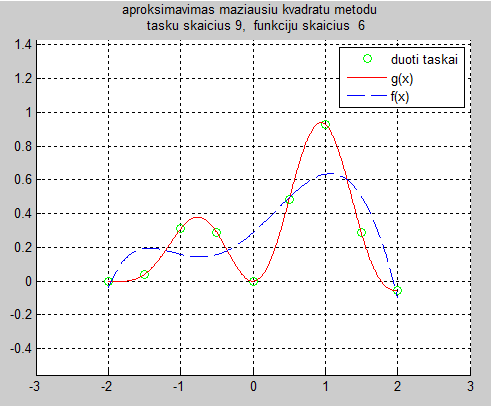
-3.0000 3.0000 -0.5554 1.4287

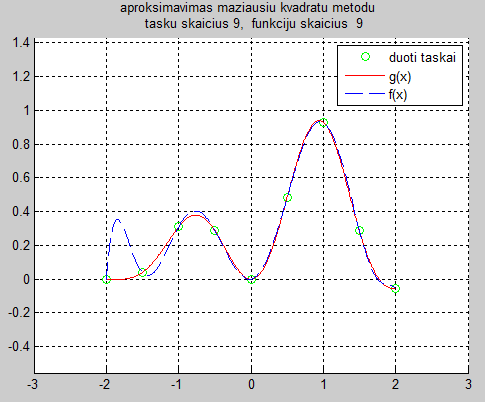
ans =

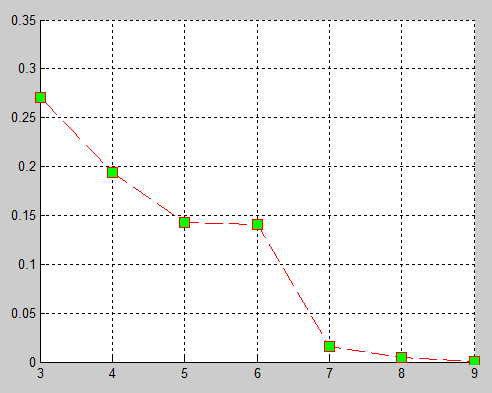
-3.0000 3.0000 -0.5554 1.4287

d =

0 0 0.2703 0.1941 0.1427 0.1409 0.0163 0.0048 0.0000







Programos kodas:

function daugianariai

clc,close all,clear all

xmin = -2;

xmax = 2;

npower=9;

m = 3;

n=2^9-1;

SX = [xmin:(xmax-xmin)/(npower-1):xmax]

SY= fnk(SX)

a=min(SX);b=max(SX);t=[a:(b-a)/n:b];

fff=fnk(t);

d=zeros();

% Maziausiu kvadratu metodo lygciu sistema:

for m=3:npower

G=base(m,SX);

c=(G'\*G)\(G'\*SY');

sss=sprintf('%5.2g',c(1));

for i=1:m-1

sss=[sss,sprintf('+%5.2gx^%1d',c(i+1),i)];

end

sss=strrep(sss,'+-','-');

% Aproksimuojanti funkcija:

nnn=200; %vaizdavimo tasku skaicius

tmin=min(SX);tmax=max(SX);

ttt=[tmin:(tmax-tmin)/(nnn-1):tmax]; %vaizdavimo taskai

Gv=base(m,ttt);

fff1=Gv\*c;

figure(m); axis([min(SX)-1,max(SX)+1,min(SY)-0.5,max(SY)+0.5]),hold on,grid on

axis

plot(SX,SY, 'go');

plot(t,fff,'r');

plot(ttt,fff1,'--');

legend({'duoti taskai', 'g(x)', 'f(x)'})

title(sprintf('aproksimavimas maziausiu kvadratu metodu \n tasku skaicius %d, funkciju skaicius %d',npower,m));

d(m)=paklaida(fff1, SX, SY, ttt);

end

d

figure(m+1);hold on,grid on

plot([3:1:npower], d(3:end), '--rs', 'MarkerFaceColor', 'g', 'MarkerSize', 10);

end

function G=base(m,x)

for i=1:m, G(:,i)=x.^(i-1); end

return

end

function d=paklaida(fff, SX, SY, ttt)

d = 0;

y = interp1(ttt, fff, SX);

for j=1:length(y)

d = d + (y(j) - SY(j)).^2;

end

d = d/2;

return

end

function rez=fnk(x)

rez = exp((-1).\*(x.^2)).\*sin(x.^2).\*(x+2); %1.9\*x.\*sin(x)-(x./1.5-3).^2

return, end