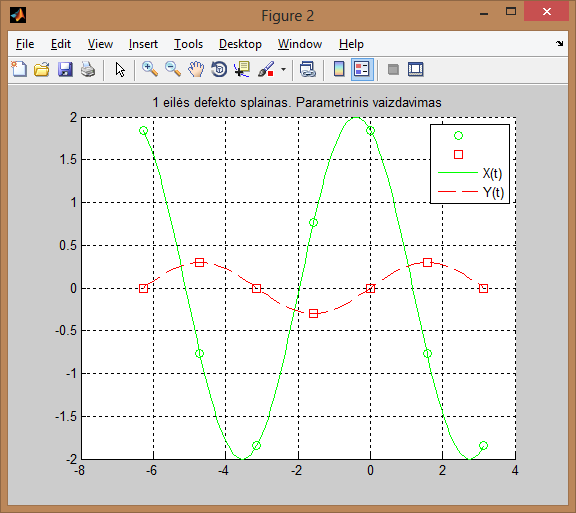
Paulius Ambroza IFE-1

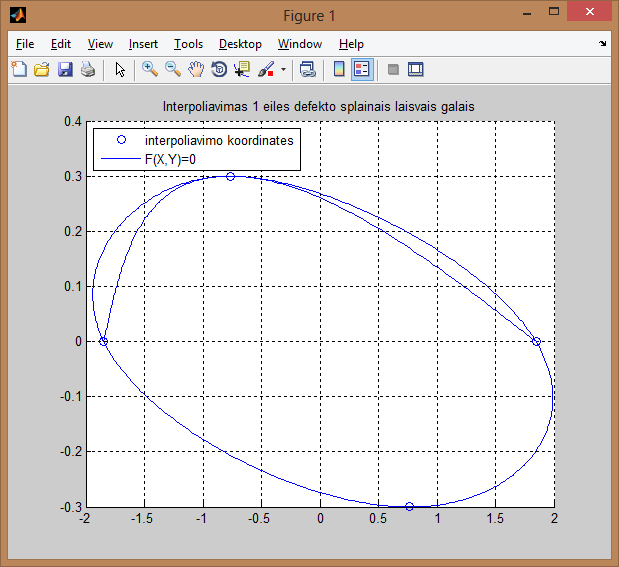
Pirmos eilės defekto splainai laisvais galais

, intervalas , n=7;

t =

-6.2832 -4.7124 -3.1416 -1.5708 0 1.5708 3.1416





Programos kodas:

function InterpoliavimasSplainai

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)

figure(1), hold on, grid on

plot(X,Y,'o');

figure(2), hold on, grid on

plot(t,X,'go');

plot(t,Y,'rs');

tt=[xmin:(xmax-xmin)/((n-1)\*20):xmax];

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

title(sprintf('1 eilės defekto splainas. Parametrinis vaizdavimas'));

plot(tt,funkcijaY(tt),'r-','LineStyle','--');

legend('','','X(t)','Y(t)');

figure(1)

DDFX=splaino\_koeficientai(t,X);

DDFY=splaino\_koeficientai(t,Y);

for iii=1:n-1

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

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

plot(SplainoX,SplainoY)

end

legend('interpoliavimo koordinates', 'F(X,Y)=0','Location','NorthWest');

title(sprintf('Interpoliavimas 1 eiles defekto splainais laisvais galais'));

return, end

function S=splainas(X,Y,DDF)

nnn=100;

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

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

sss=xxx-X(1);

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

DDF=A\b;

return, end

function funkX=funkcijaX(x)

funkX=2.\*cos(x+pi/8)

return, end

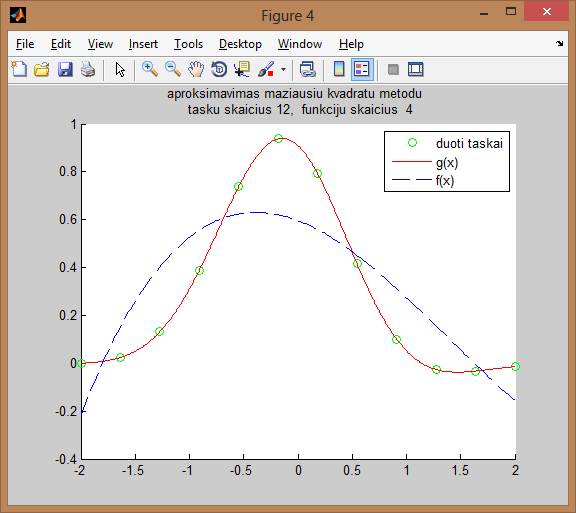
function funkY=funkcijaY(x)

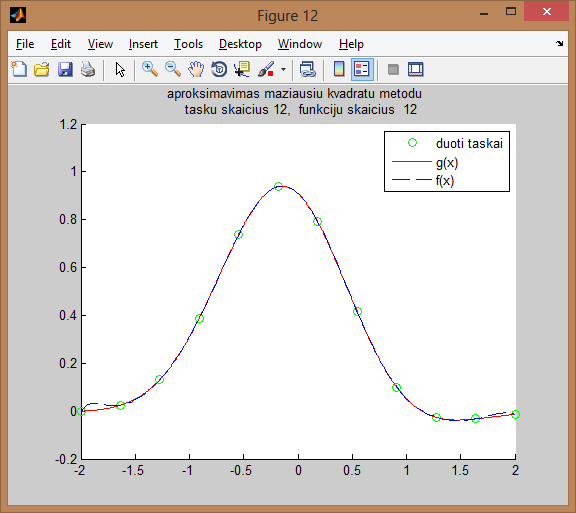
funkY=0.3.\*sin(x)

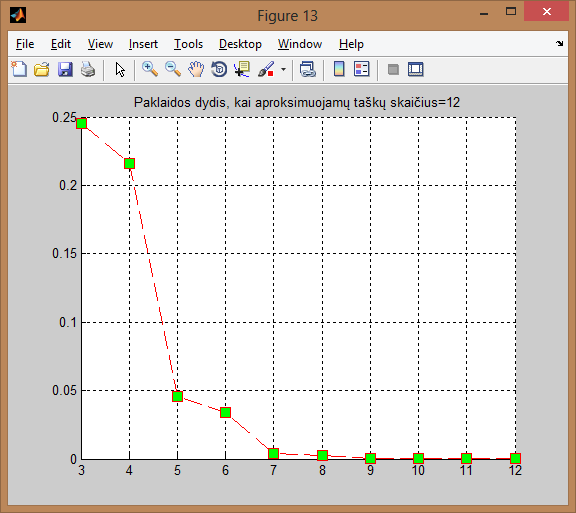
return, end

Aproksimavimas daugianariais vienanarių bazėje

,







Programos kodas:

function daugianariai

clc,close all,clear all

xmin = -2;

xmax = 2;

npower=12;

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();

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,'+-','-');

nnn=200; %vaizdavimo taškų skaičius

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

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

Gv=base(m,ttt);

fff1=Gv\*c;

figure(m); hold on,grid off

plot(SX,SY, 'go');

plot(t,fff,'r');

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

legend({'duoti taskai', 'g(x)', sprintf('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);

title('Paklaidos dydis, kai aproksimuojamų taškų skaičius=12');

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(-x.^2).\*sin(x+2)

return, end