KAUNO TECHNOLOGIJOS UNIVERSITETAS
INFORMATIKOS FAKULTETAS

Skaitiniai metodai ir algoritmai (P170B115)

Laboratorinių darbų ataskaita

Atliko:
IFF-1/4 gr. studentas
Mildaras Karvelis
2023 m. spalio 26 d.
Priėmė:
Prof. Barauskas Rimantas

KAUNAS 2023

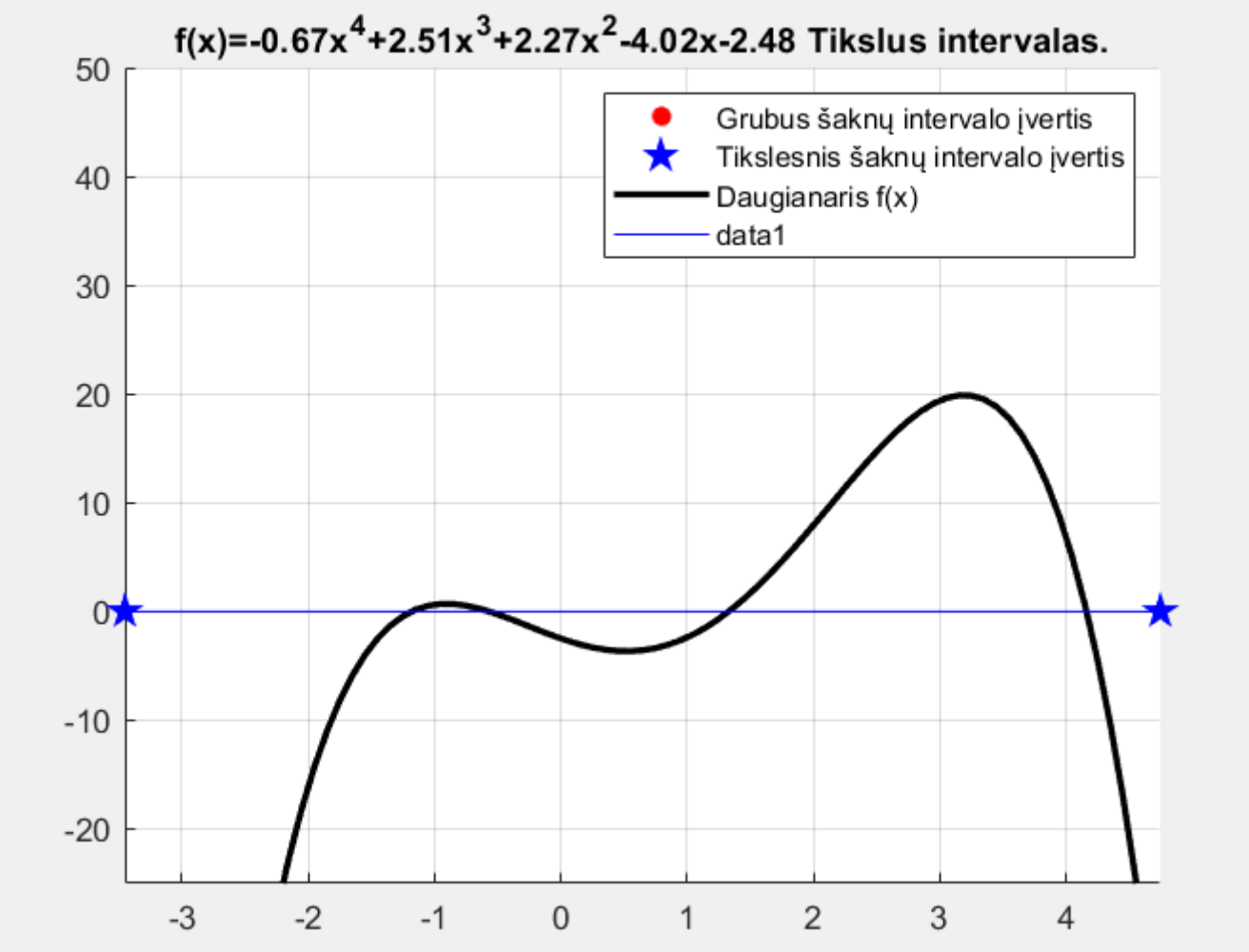
TURINYS

1. Netiesinių lygčių sprendimas

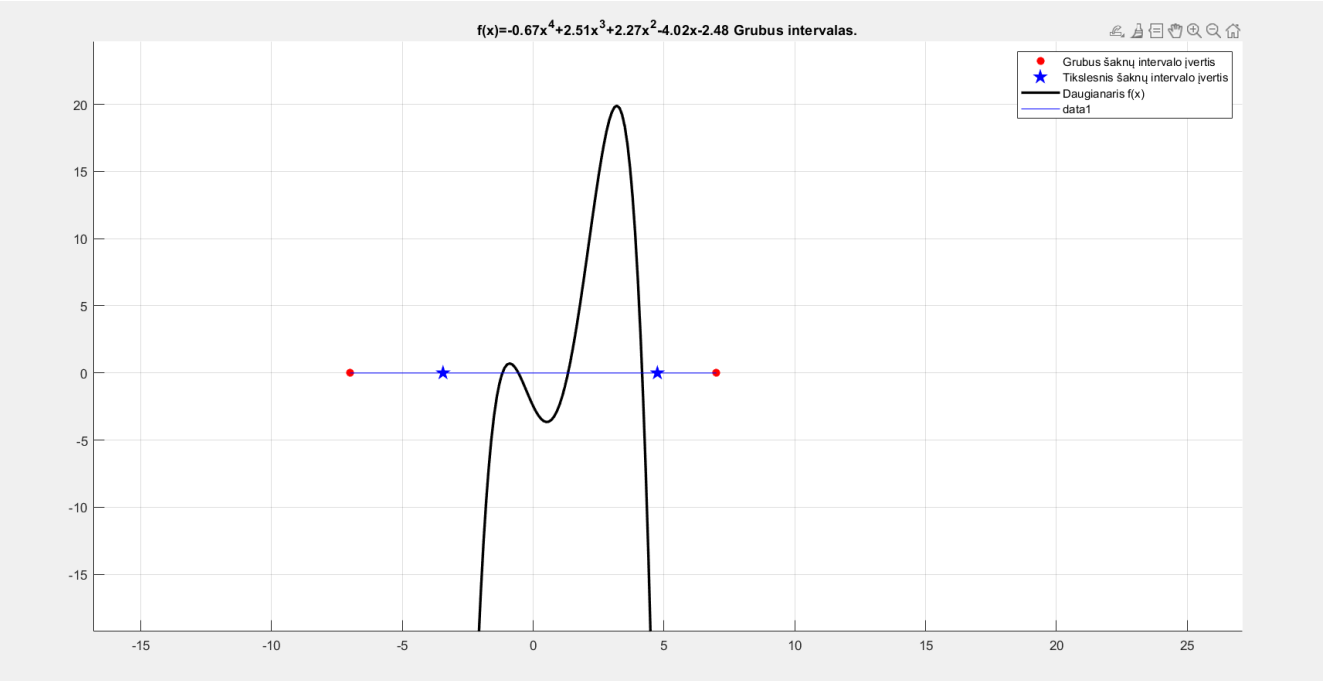
Varianto Nr.	Daugianariai $f(x)$	Funkcijos $g(x)$	Metodai[1]
8	$-0.67x^4 + 2.51x^3 + 2.27x^2 - 4.02x - 2.48$	$e^{-x^2}\sin(x^2)(x+2);\ -3 \le x \le 3$	2, 4

Sprendimo metodai: Pusiaukirtos ir Kvazi-Niutono (kirstinių)

1.1. f(x) daugianario sprendimas



1 pav. Daugianario tikslus įvertis

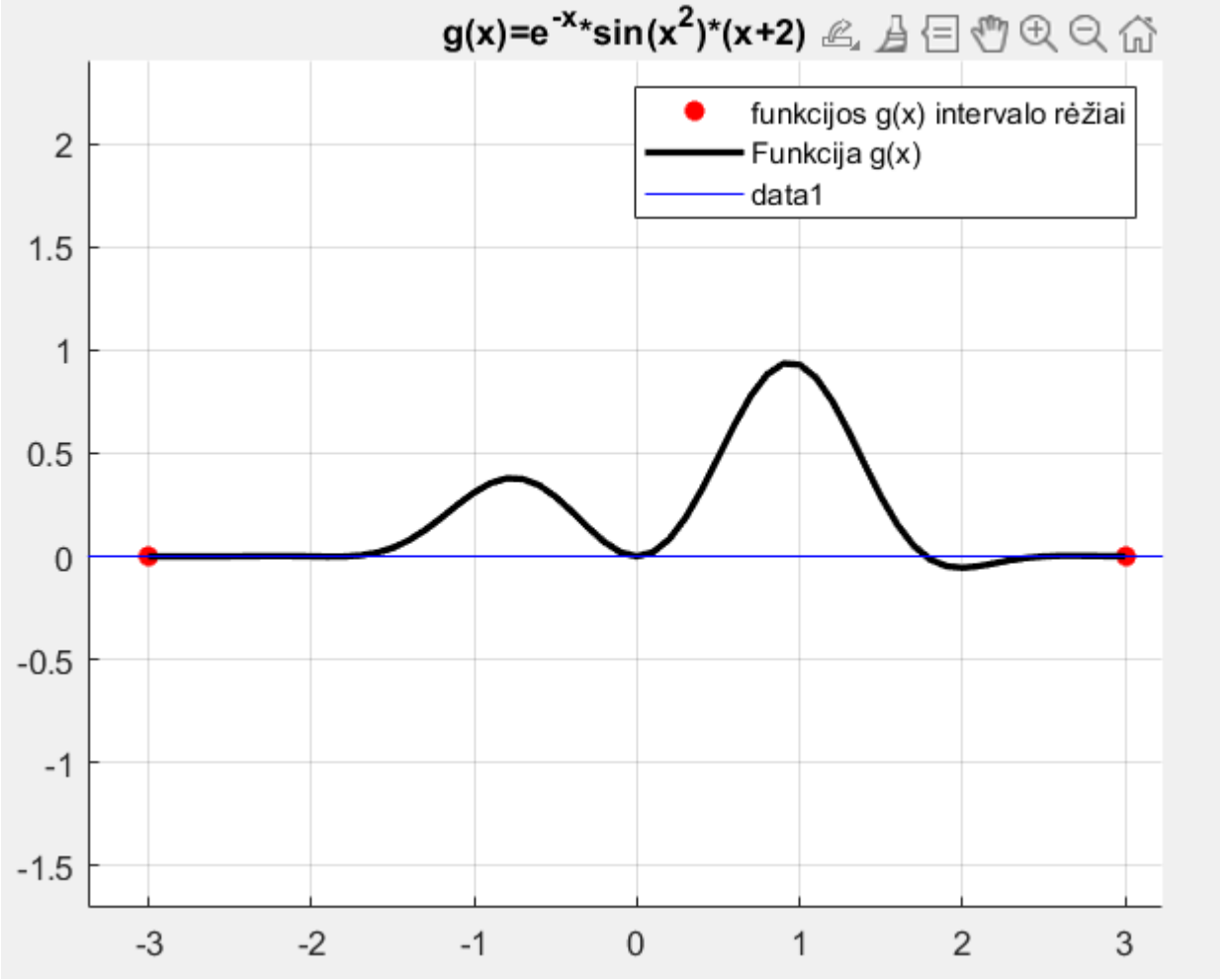


2 pav. Daugianario grubaus įverčio grafikas

Grubus lygties f(x) = 0 šaknų intervalo įvertis	[-6.999999999999999 ; 6.999999999999999]
Tikslesnis lygties f(x) = 0 šaknų intervalo įvertis	[-3.449489742783178 ; 4.746268656716417]

1 lentelė Tikslūs ir grubūs įverčiai

1.2. g(x) daugianario grafiškas vaizdavimas



3 pav. g(x) daugianario grafinis vaizdas

1.3. Programos kodo fragmentai

```
function iverciai_ir_grafikai
clc, close all, clear all;
format long;
% ------------------------------------
% Daugianaris f(x)
f = @(x)(-0.67*x.^4) + 2.51*x.^3+2.27*x.^2-4.02*x-2.48;
%f = @(x)(0.67*x.^4) - 2.51*x.^3-2.27*x.^2+4.02*x+2.48;
f_name = '-0.67x^4+2.51x^3+2.27x^2-4.02x-2.48';
% ------------------------------------
% Funkcija g(x)
g = @(x)exp(-1*x.^2).*sin(x.^2).*(x+2);
g_name = 'e^-^x*sin(x^2)*(x+2)';
% ------------------------------------
a = [0.67 -2.51 -2.27 4.02 2.48];
n = numel(a);
[R_grub, R_neig, R_teig]=Reziai(n, a);
Grubus = [-R_grub R_grub]
Tikslesnis = [R_neig R_teig]
% ------------------------------------
% grafikų braižymas
grubus_intervalas = -R_grub:0.1:R_grub;
tikslus_intervalas = R_neig:0.1:R_teig;
% ------------------------------------
% f(x) grubus
figure(1); hold on; grid on;
% plot(-min(R_grub, R_neig), 0, 'bp', 'LineWidth', 2);
% plot(min(R_grub,R_teig),0,'bp', 'LineWidth', 2);
plot([-R_grub,R_grub],[0 0],'r*', 'LineWidth', 2);
```

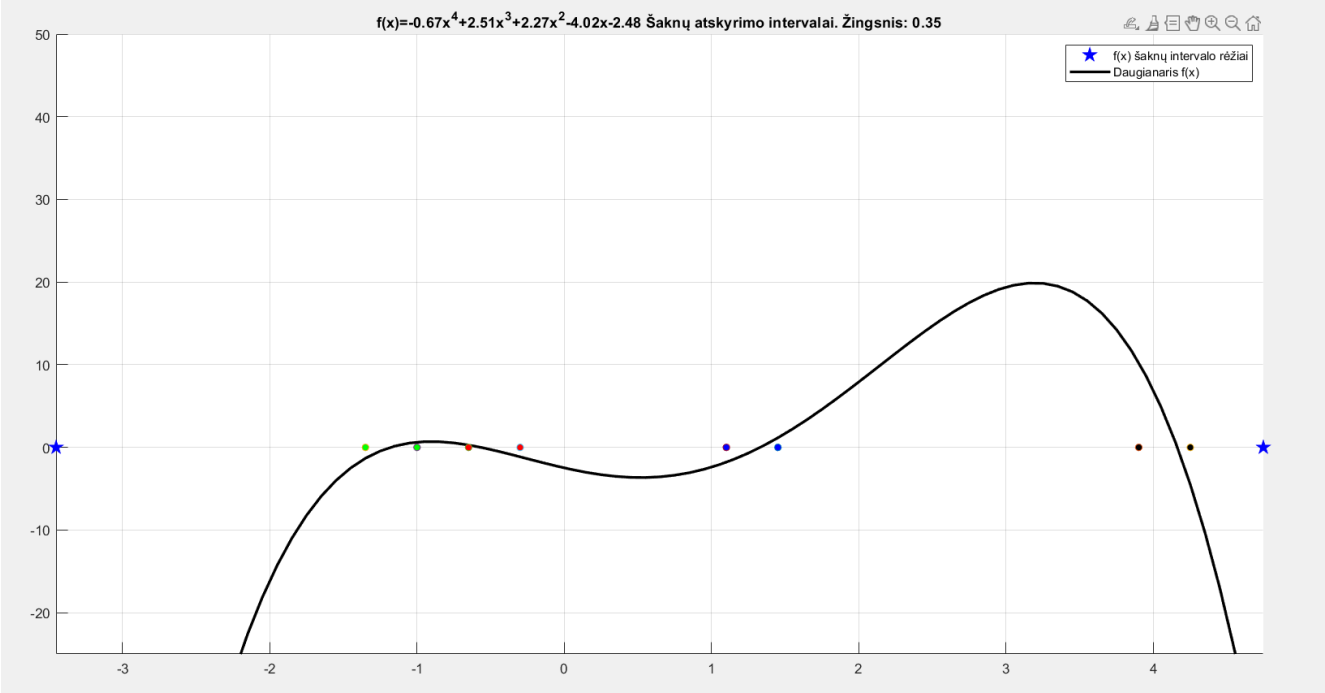
```
plot([R_neig R_teig], [0 0], 'bp', 'LineWidth', 2);
plot(grubus_intervalas, f(grubus_intervalas), 'k-', 'LineWidth', 2);
title(['f(x)=', f_name, ' Grubus intervalas.']);
legend('Grubus šaknų intervalo įvertis', 'Tikslesnis šaknų intervalo įvertis', 'Daugianaris f(x)');
axis([-R_grub R_grub -R_grub R_grub]);
plot([-R_grub, R_grub], [0, 0], 'b'); % X ašies linija
% ------------------------------------
% f(x) tikslus
figure(2); hold on; grid on;
plot([-R_grub,R_grub],[0 0],'r*', 'LineWidth', 2);
plot([R_neig R_teig], [0 0], 'bp', 'LineWidth', 2);
plot(tikslus_intervalas, f(tikslus_intervalas), 'k-', 'LineWidth', 2);
title(['f(x)=', f_name, ' Tikslus intervalas.']);
legend('Grubus šaknų intervalo įvertis', 'Tikslesnis šaknų intervalo įvertis', 'Daugianaris f(x)');
axis([R_neig R_teig -25 50]);
plot([-R_grub, R_grub], [0, 0], 'b'); % X ašies linija
% ------------------------------------
% g(x)
figure(3); hold on; grid on;
g_min = -3;
g_max = 3;
g_intervalas = g_min:0.1:g_max;
plot([g_min g_max], [0 0], 'r*', 'LineWidth', 2);
plot(g_intervalas, g(g_intervalas), 'k-', 'LineWidth', 2);
title(['g(x)=', g_name]);
legend('funkcijos g(x) intervalo rėžiai', 'Funkcija g(x)');
axis([g_min g_max -6 2]);
plot([-R_grub, R_grub], [0, 0], 'b'); % X ašies linija
end
```

```
function [R_grub, R_neig, R_teig] = Reziai(n, a)
    %Rgrub
    R_grub = 1 + max(abs(a(2:end)))/a(1);
    % Rteig skaiciavimas
    b = a(2:end);
    B = max(abs(b(b<0)));
    k = n - (n - (find(b<0, 1)));
    R_teig = 1 + (B/a(1))^(1/k);
    % Rneig skaiciavimas
    if mod(n, 2) == 0
        a(end:-2:1) = -a(end:-2:1);
        b = a(2:end);
        B = max(abs(b(b<0)));
        k = n - (n - (find(b<0, 1)));
        R_neig = 1 + (B/a(1))^(1/k);
        R_neig = -R_neig;
    else
        a(end:-2:1) = -a(end:-2:1);
        a = a.*-1;
        b = a(2:end);
        B = max(abs(b(b<0)));
        k = n - (n - (find(b<0, 1)));
        R_neig = 1 + (B/a(1))^(1/k);
        R_neig = -R_neig;
    end;
end
```

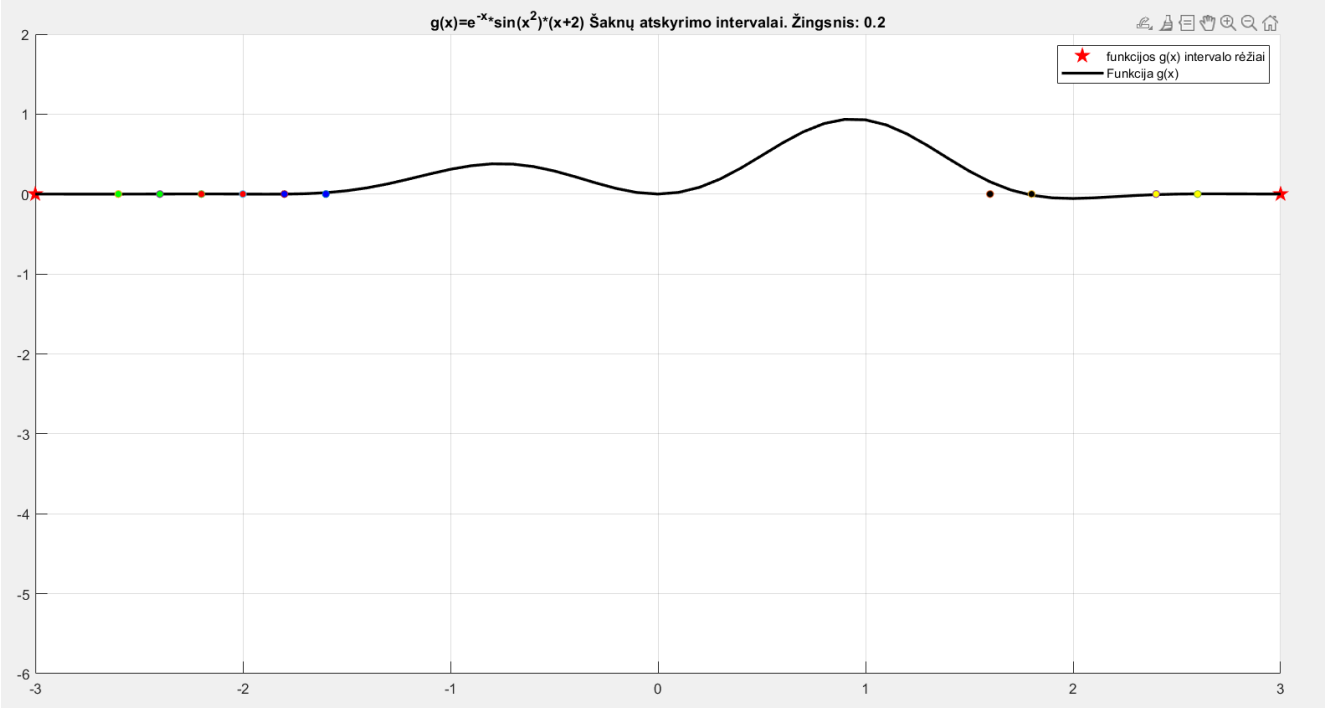
2. Šaknų atskyrimas skenavimo metodu.

2.1. f(x) ir g(x) grafikai ir intervalai

Skenavimas atliekamas intervale [-3.449 ; 4.746], skenavimo žingsnis lygus 0,35. Funkcijos [-3; 3]



4 pav. Daugianario šaknų atskyrio intervalai.



5 pav. Funkcijos šaknų atskyrimo intervalai

Intervalas Nr.	Daugianario intervalas
1	[-1.349489742783177 ; -0.999489742783177]
2	[-0.649489742783177 ; -0.299489742783177]
3	[1.100510257216822 ; 1.450510257216822]
4	[3.900510257216823 ; 4.250510257216823]

Intervalo Nr.	Funkcijos intervalas
1	[-2.600000000000000 ; -2.399999999999999]
2	[-2.199999999999999 ; -1.999999999999999]
3	[-1.799999999999999 ; -1.599999999999999]
4	[1.600000000000000 ; 1.800000000000000]
5	[2.400000000000001 ; 2.600000000000001]

2.2. Programos kodo fragmentai

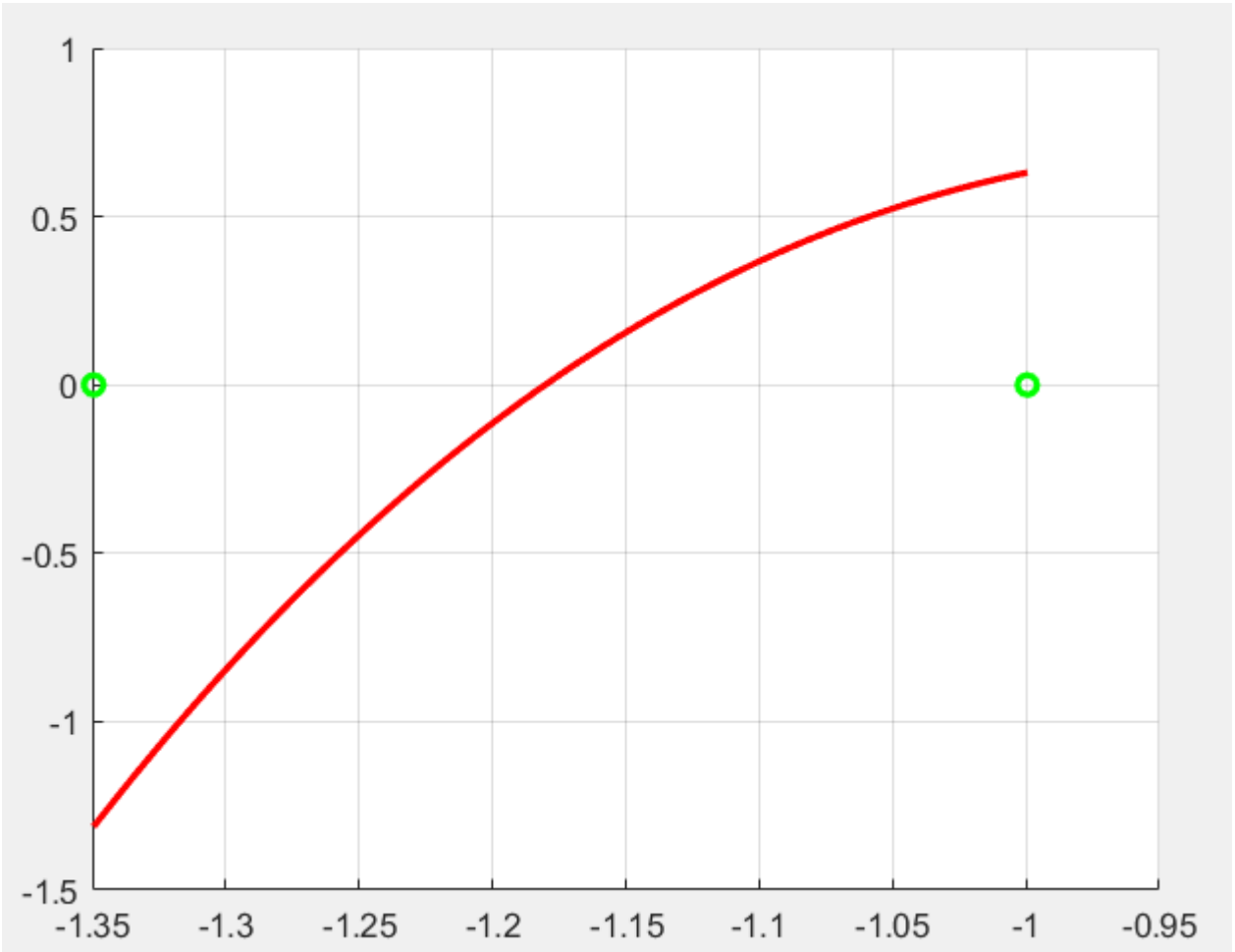
```
function saknu_intervalai
    clc, close all, clear all;
    format long;
    % ------------------------------------
    % Daugianaris f(x)
    f = @(x)(-0.67*x.^4) + 2.51*x.^3+2.27*x.^2-4.02*x-2.48;
    %f = @(x)(0.67*x.^4) - 2.51*x.^3-2.27*x.^2+4.02*x+2.48;
    f_name = '-0.67x^4+2.51x^3+2.27x^2-4.02x-2.48';
    % ------------------------------------
    % Funkcija g(x)
    g = @(x)exp(-1*x.^2).*sin(x.^2).*(x+2);
    g_name = 'e^-^x*sin(x^2)*(x+2)';
    % ------------------------------------
    a = [0.67 -2.51 -2.27 4.02 2.48];
    n = numel(a);
    [R_grub, R_neig, R_teig]=Reziai(n, a);
    colors = ['g', 'r', 'b', 'k', 'y', 'c'];
    % ------------------------------------
    % šaknų intervalų atskyrimas daugianariui f(x)
    zingsnis = 0.35; % zingsnio nustatymas
    [SaknuIntervalai_fx]=SkenavimasPastoviu(R_neig, R_teig, zingsnis, f);
    SaknuIntervalai_fx
    % daugianario f(x) ir jo šaknų intervalų atvaizdavimas
```

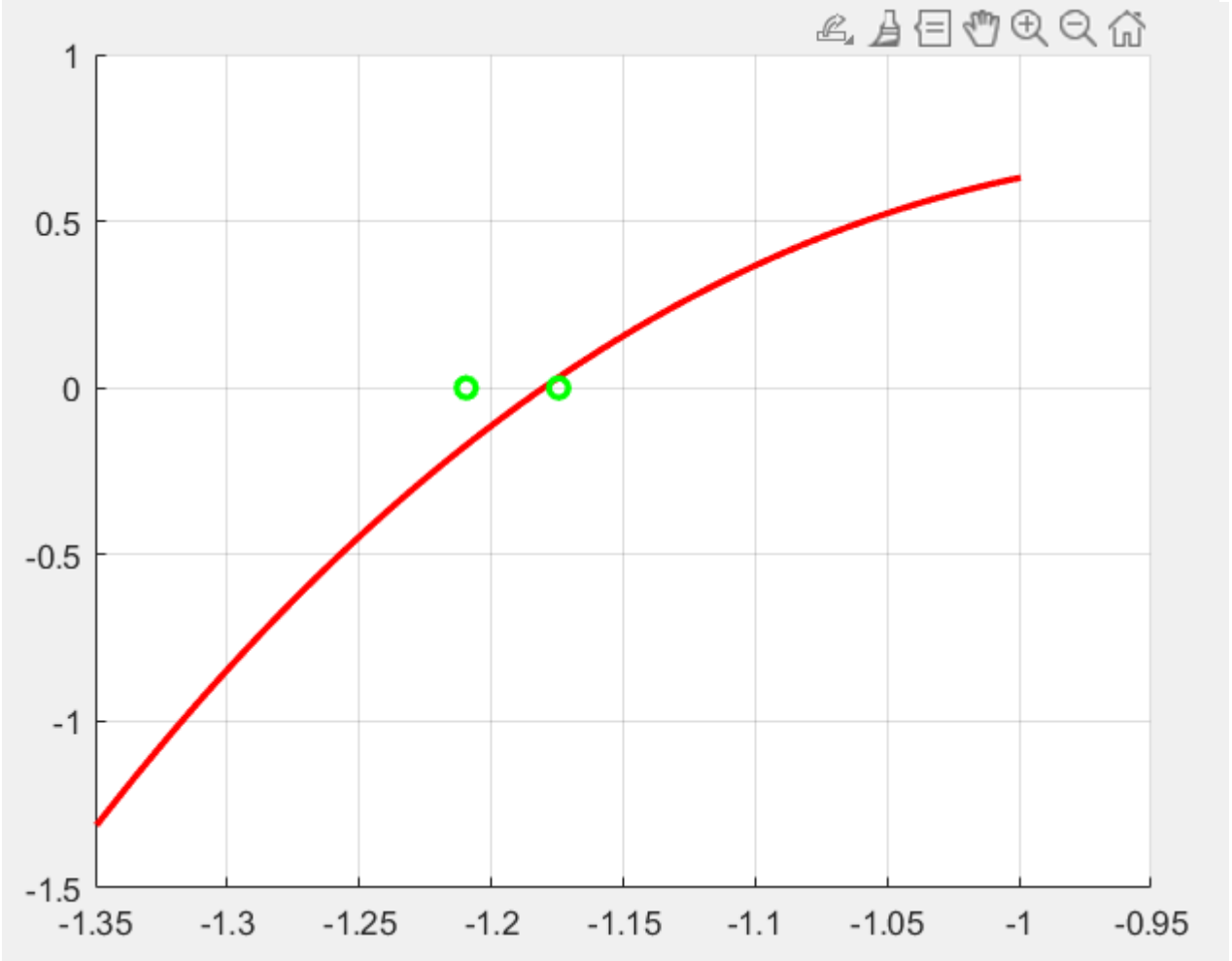
```
    figure(1); hold on; grid on;
    tikslus_intervalas = R_neig:0.1:R_teig;
    plot([R_neig R_teig], [0 0], 'bp', 'LineWidth', 2);
    plot(tikslus_intervalas, f(tikslus_intervalas), 'k-', 'LineWidth', 2);
    for i = 1:length(SaknuIntervalai_fx)
        plot(SaknuIntervalai_fx(i, 1), 0*SaknuIntervalai_fx(i, 1), 'o', 'MarkerFaceColor', colors(i),
'MarkerSize', 5);
        plot(SaknuIntervalai_fx(i, 2), 0*SaknuIntervalai_fx(i, 2), 'o', 'MarkerFaceColor', colors(i),
'MarkerSize', 5);
    end
    title(['f(x)=', f_name, ' Šaknų atskyrimo intervalai. Žingsnis: ', num2str(zingsnis)]);
    legend('f(x) šaknų intervalo rėžiai', 'Daugianaris f(x)');
    axis([R_neig R_teig -25 50]);
    % ---------------------------------
    % šaknų intervalų atskyrimas funkcijai g(x)
    zingsnis = 0.2; % zingsnio nustatymas
    g_min = -3;
    g_max = 3;
    g_intervalas = g_min:0.1:g_max;
    [SaknuIntervalai_gx]=SkenavimasPastoviu(g_min, g_max, zingsnis, g);
    SaknuIntervalai_gx
    figure(2); hold on; grid on;
    plot([g_min g_max], [0 0], 'rp', 'LineWidth', 2);
    plot(g_intervalas, g(g_intervalas), 'k-', 'LineWidth', 2);
    for i = 1:length(SaknuIntervalai_gx)
        plot(SaknuIntervalai_gx(i, 1), 0*SaknuIntervalai_gx(i, 1), 'o', 'MarkerFaceColor', colors(i),
'MarkerSize', 5);
        plot(SaknuIntervalai_gx(i, 2), 0*SaknuIntervalai_gx(i, 2), 'o', 'MarkerFaceColor', colors(i),
'MarkerSize', 5);
    end
    title(['g(x)=', g_name, ' Šaknų atskyrimo intervalai. Žingsnis: ', num2str(zingsnis)]);
    legend('funkcijos g(x) intervalo rėžiai', 'Funkcija g(x)');
    axis([g_min g_max -6 2]);
end
```

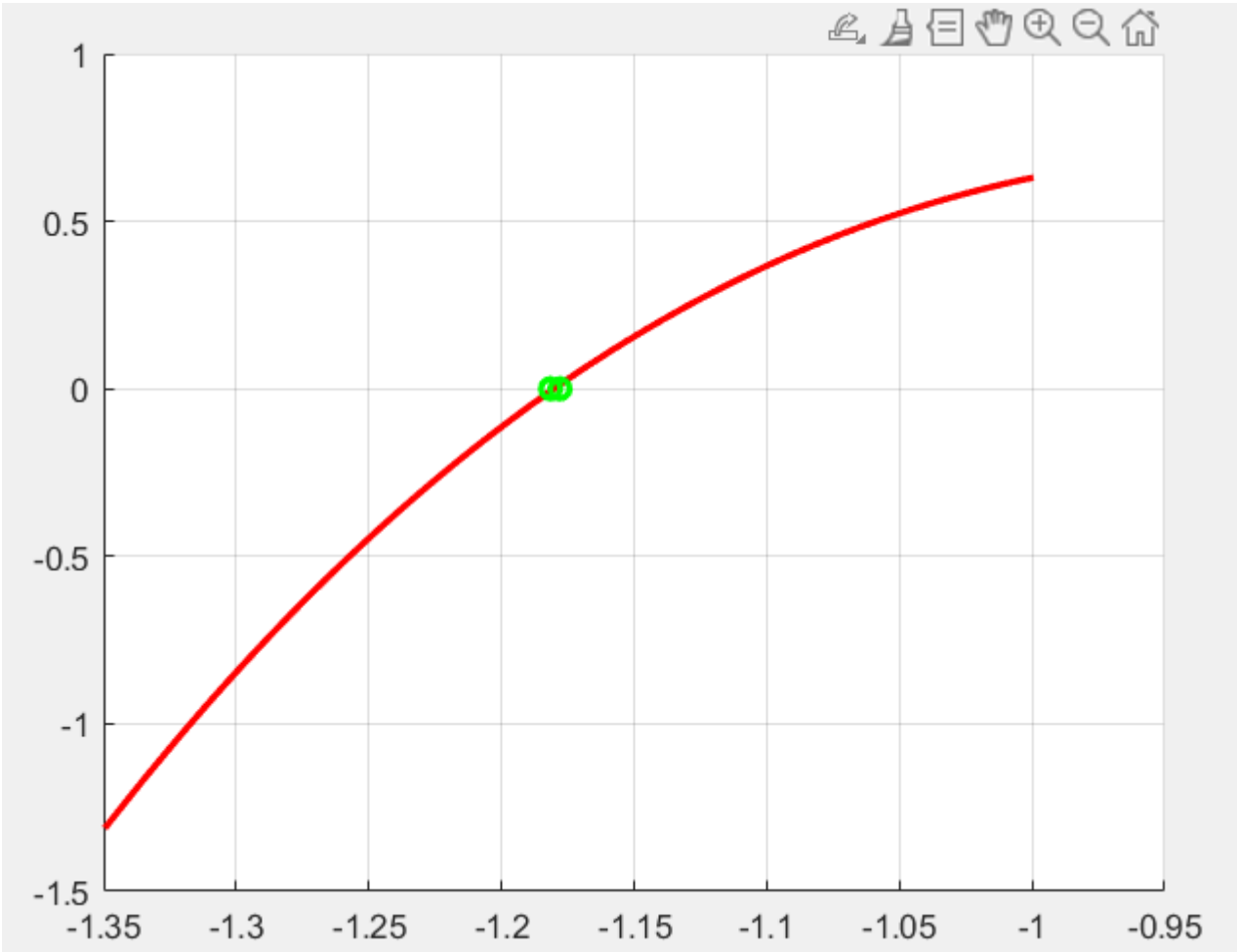
3. Šaknų tikslinimas skenavimo, ~~pusiaukirtos~~, Kvazi-Niutono (kirstinių) metodais.

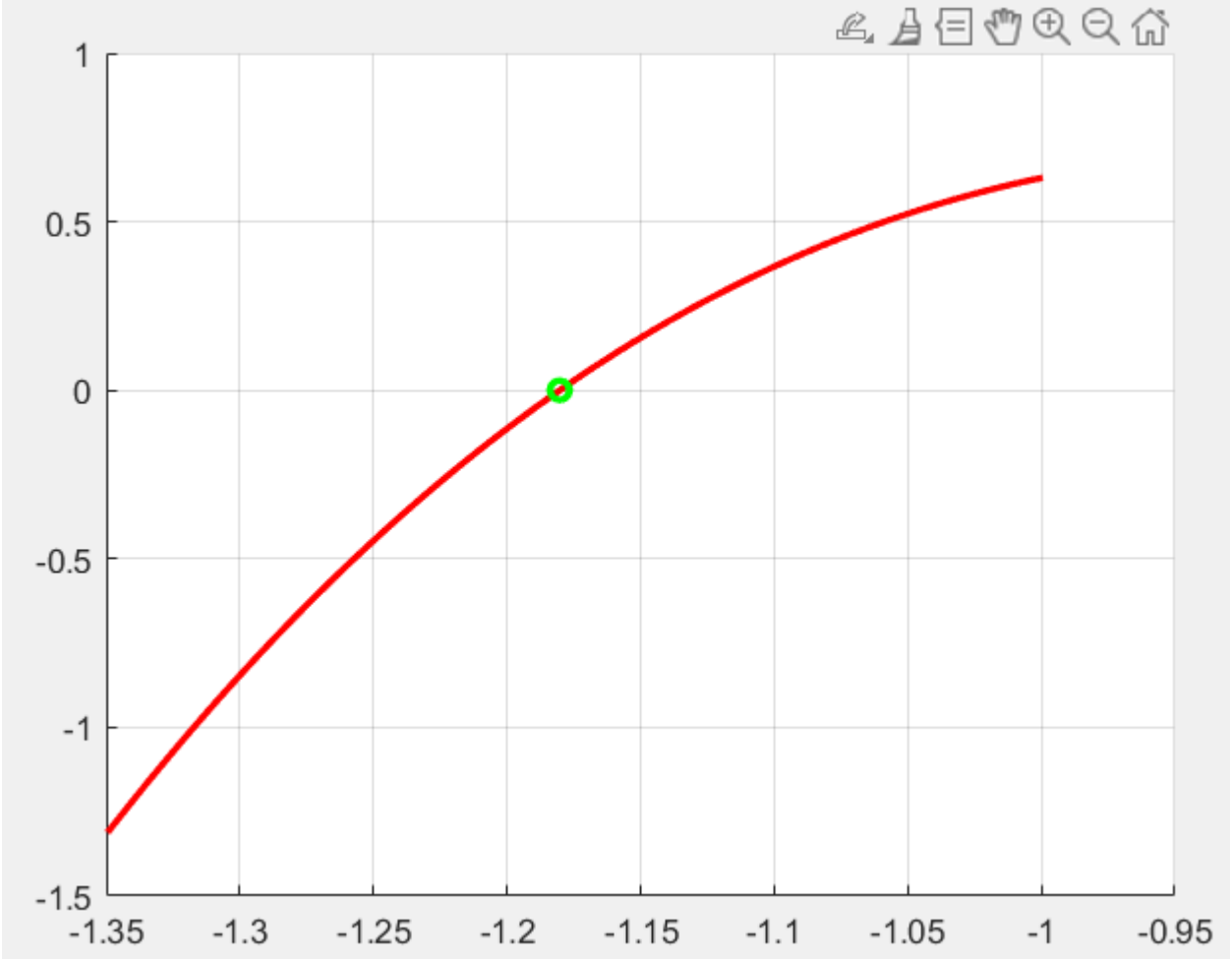
3.1. f(x) daugianaris

	Pradinis intervalas	Šaknis	Tikslumas	Iteracijų skaičius
Kvazi-Niutono metodas	[-1.349489742783177; - 0.999489742783177]	-1.180246029145651	0.000000000000012	7
	[-0.649489742783177; -0.299489742783177]	-0.566365716276059	0.000000000001030	5
	[1.100510257216822; 1.450510257216822]	1.330278155028645	0.000000000006279	5
	[3.900510257216823; 4.250510257216823]	4.162602247119908	0.000000000453074	5

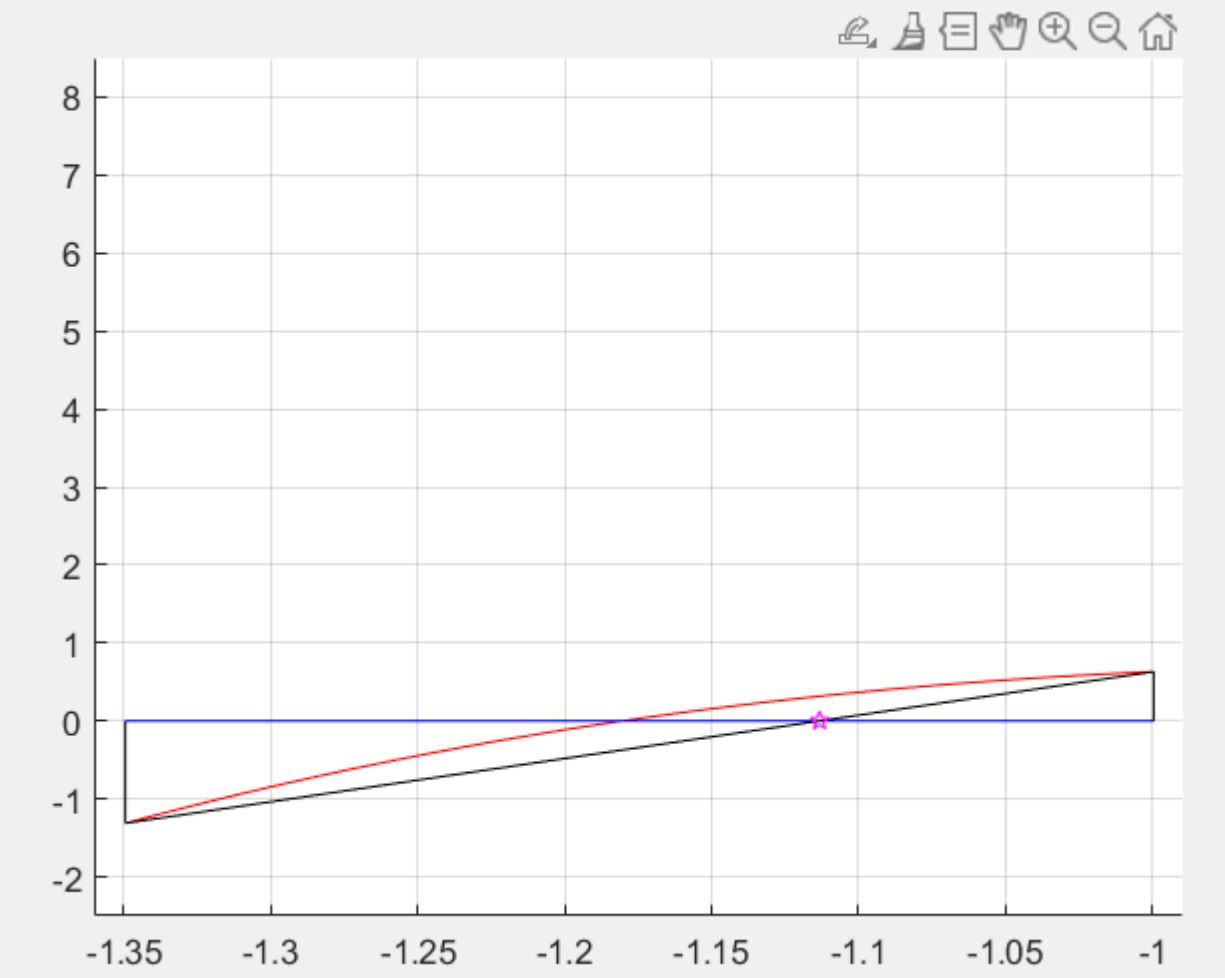

1
0.5
0
-0.5
-1
-1.5
-1.35
-1.3
-1.25
-1.2
-1.15
-1.1
-1.05
-1
-0.95

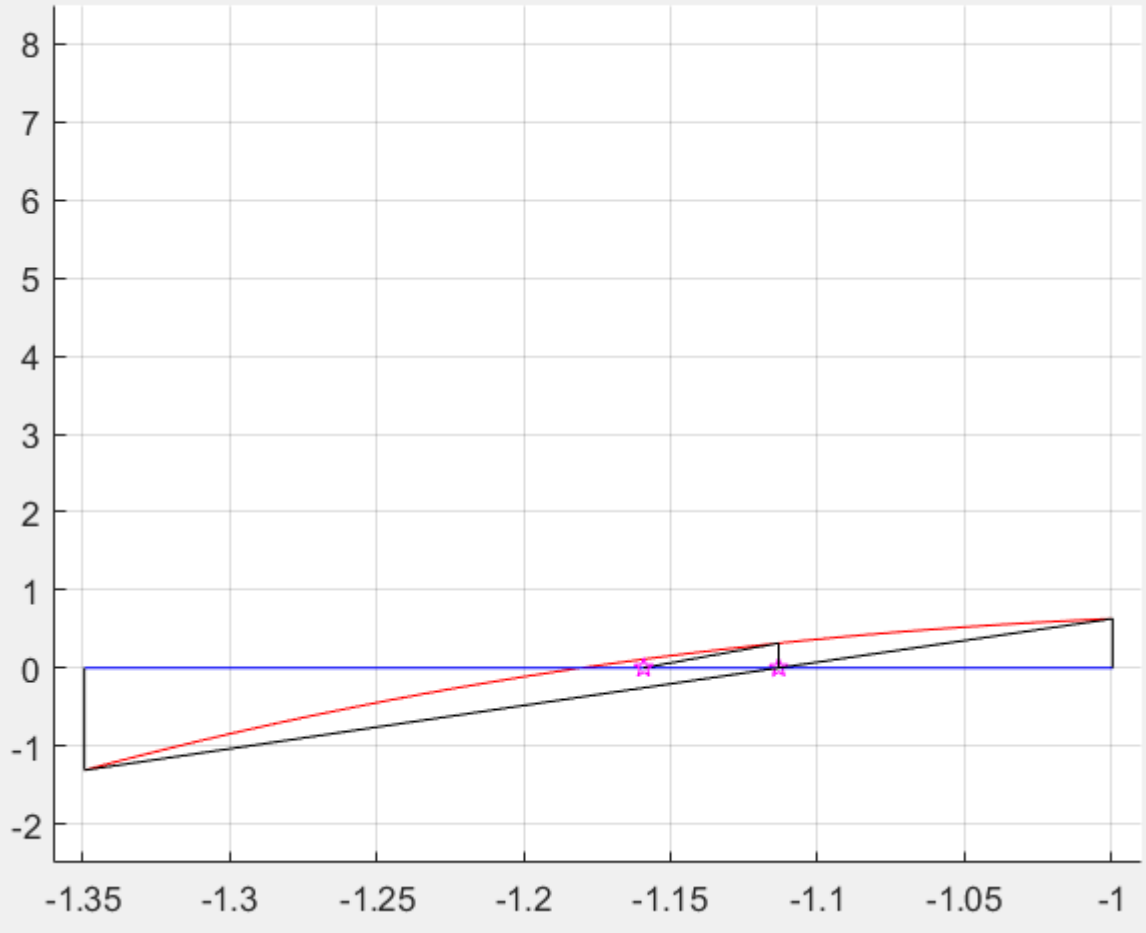

1
0.5
0
-0.5
-1
-1.5
-1.35
-1.3
-1.25
-1.2
-1.15
-1.1
-1.05
-1
-0.95

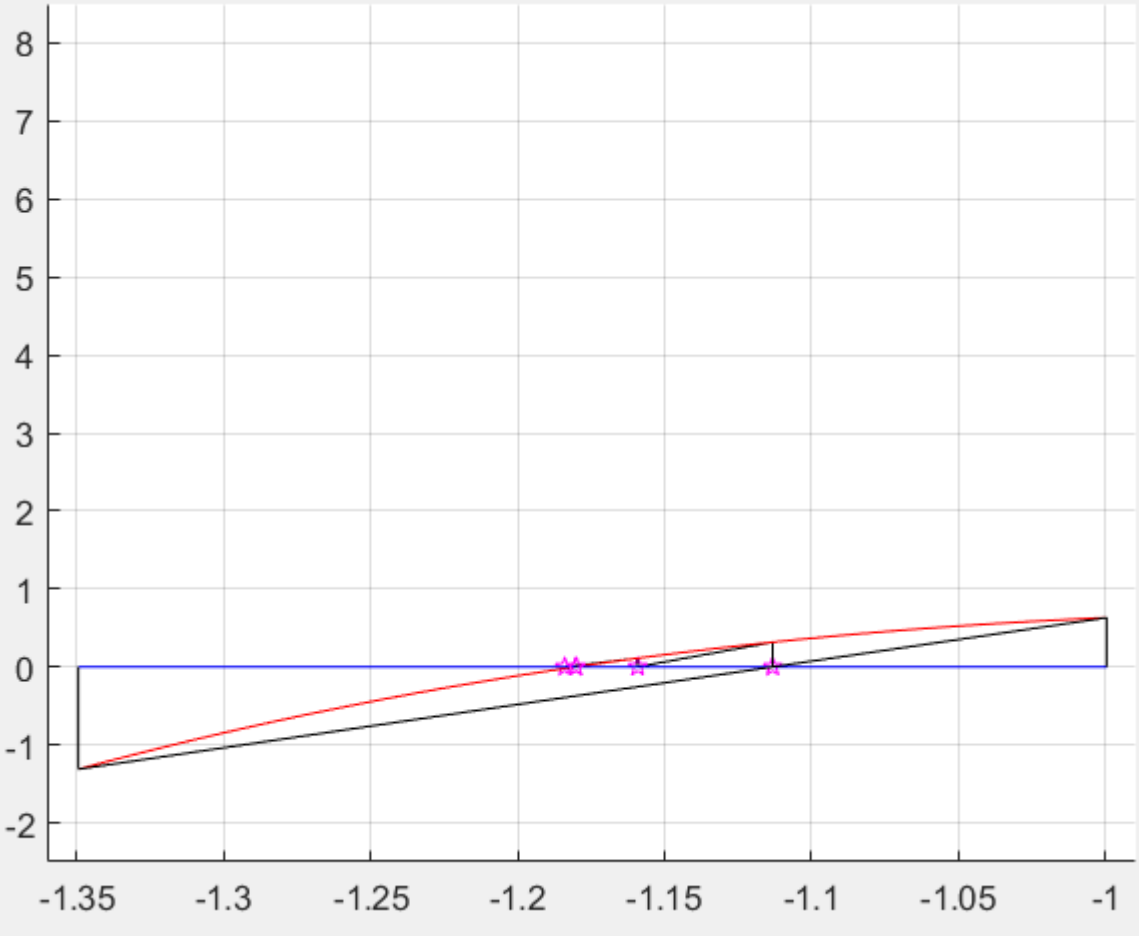

1
0.5
0
-0.5
-1
-1.5
-1.35
-1.3
-1.25
-1.2
-1.15
-1.1
-1.05
-1
-0.95

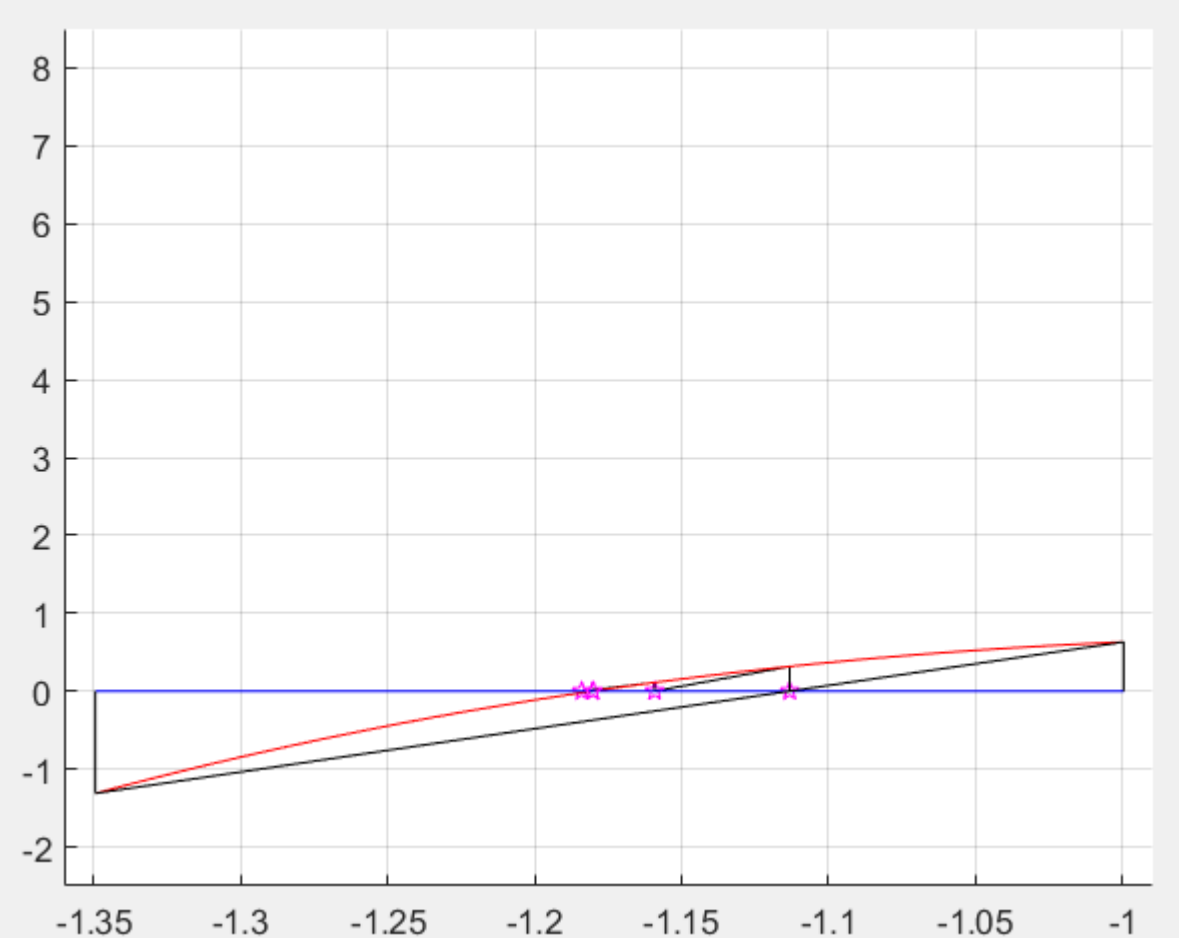

1
0.5
0
-0.5
-1
-1.5
-1.35
-1.3
-1.25
-1.2
-1.15
-1.1
-1.05
-1
-0.95

	Pradinis intervalas	Šaknis	Tikslumas	Iteracijų skaičius
Skenavimo metodas	[-1.349489742783177; - 0.999489742783177]	-1.180246029145651	0.000000004758209	10
	[-0.649489742783177; -0.299489742783177]	-0.566365716276059	0.000000001819280	10
	[1.100510257216822; 1.450510257216822]	1.330278155028645	0.000000006224549	10
	[3.900510257216823; 4.250510257216823]	4.162602247119908	0.000000053046186	10




8
7
6
5
4
3
2
1
0
-1
-2
-1.35
-1.3
-1.25
-1.2
-1.15
-1.1
-1.05
-1


8
7
6
5
4
3
2
1
0
-1
-2
-1.35
-1.3
-1.25
-1.2
-1.15
-1.1
-1.05
-1



	Pradinis intervalas	Šaknis (fzero)	Šaknis (roots)	
MATLAB funkcijos	-1.349489742783177	-1.180246029145653	-1.180246029145654	
	-0.649489742783177	-0.566365716276339	-0.566365716276338	
	1.100510257216822	1.330278155027950	1.330278155027949	
	1. 3.900510257216823	4.162602247110461	4.162602247110462	

3.2. g(x) funkcija

	Pradinis intervalas	Šaknis	Tikslumas	Iteracijų skaičius
Kvazi-Niutono metodas	[-2.800000000000000; -2.700000000000000]	-3.069980129481733	0.000000000002991	8
	[-2.399999999999999; -2.299999999999999]	-2.506628140055660	0.000000000638296	4
	[-1.999999999999999; -1.899999999999999]	-2.000000000000000	0	1
	[1.400000000000000; 1.500000000000000]	1.772453850902849	0.000000000001541	7
	[2.200000000000001; 2.300000000000001]	2.506628274240725	0.000000000016466	7

k	Pradinis intervalas	Šaknis	Tikslumas	Iteracijų

				skaičius
	[-2.600000000000000; -2.399999999999999]	-2.506628275000000	0.000000000002993	10
	[-2.199999999999999; -1.999999999999999]	-1.999999999000001	0.000000000027723	10
	[-1.799999999999999; -1.599999999999999]	-1.772453850999999	0.000000000031564	10
	[1.600000000000000; 1.800000000000000]	1.772453850999999	0.000000000632502	10
	[2.400000000000001; 2.500000000000001]	2.506628275000001	0.000000000057759	10

MATLAB funkcijos	Pradinis intervalas	Šaknis (fzero)
	-2.600000000000000	-2.506628274631001
	-2.199999999999999	-2
	1.600000000000000	1.772453850905516
	2.400000000000001	2.506628274631001

3.3. Programos kodo fragmentai

```
function saknu_tikslinimas
    clc, close all, clear all;
    format long;
    % ------------------------------------
    % Daugianaris f(x)
    f = @(x)(-0.67*x.^4) + 2.51*x.^3+2.27*x.^2-4.02*x-2.48;
    %f = @(x)(0.67*x.^4) - 2.51*x.^3-2.27*x.^2+4.02*x+2.48;
    f_name = '-0.67x^4+2.51x^3+2.27x^2-4.02x-2.48';
    % ------------------------------------
    % Funkcija g(x)
    g = @(x)exp(-1*x.^2).*sin(x.^2).*(x+2);
    g_name = 'e^-^x*sin(x^2)*(x+2)';
    % ------------------------------------
    a = [0.67 -2.51 -2.27 4.02 2.48];
    n = numel(a);
    [R_grub, R_neig, R_teig]=Reziai(n, a);
    % ------------------------------------
    % šaknų intervalų atskyrimas daugianariui f(x)
    % ------------------------------------
    zingsnis = 0.35; % zingsnio nustatymas
    [SaknuIntervalai_fx]=SkenavimasPastoviu(R_neig, R_teig, zingsnis, f);
    % ------------------------------------
    % REKURSINIS SKENAVIMAS (MAŽINANT ŽINGSNĮ)
    % ------------------------------------
    % šaknų tikslinimas daugianariui f(x)
    % ------------------------------------
     fprintf( '-----------------------------------------------------\n');
     fprintf( 'Šaknų tikslinimas skenavimo metodu, mažinant žingsnį\n');
     fprintf( 'Daugianaris f(x)=-0.67x^4+2.51x^3+2.27x^2-4.02x-2.48\n');
     fprintf( '-----------------------------------------------------\n');
     fprintf( 'Stulpelių reikšmės:\n');
     fprintf( '1:2 - pradiniai šaknų tikslinimo intervalai\n');
     fprintf( '3 - šaknis\n');
     fprintf( '4 - tikslumas\n');
     fprintf( '5 - atliktų iteracijų kiekis\n');
     fprintf( '-----------------------------------------------------\n');
    Saknys_intervalai_fx = [];
    Tikslumai = [];
    Iteracijos = [];
    tikslumas = 1e-9;
    for i=1:length(SaknuIntervalai_fx)
```

```
        x_min = SaknuIntervalai_fx(i,1);
        x_max = SaknuIntervalai_fx(i, 2);
        if i == 1
            draw = 1;
            figure(1); grid on; hold on;
            npoints= 1000;
            x = x_min:(x_max-x_min)/(npoints - 1):x_max;
            plot(x, f(x), 'r-', 'LineWidth', 2);
        else
            draw = 0;
        end;
        if (sign(f(x_min)) ~= sign(f(x_max)))
            iteracijos_sk = 0;
            [a, b, it, t]=SkenavimasRekursija(x_min, x_max, zingsnis, tikslumas, f, iteracijos_sk,
draw);
            Saknys_intervalai_fx = [Saknys_intervalai_fx; a b];
            Iteracijos = [Iteracijos; it];
            Tikslumai = [Tikslumai; t];
        end
    end
    close all;
    Saknys_fx = (Saknys_intervalai_fx(:,1) + Saknys_intervalai_fx(:,2))/2;
    Rez_fx = [];
    for i=1:length(Saknys_fx)
        Rez_fx = [Rez_fx; SaknuIntervalai_fx(i,:) Saknys_fx(i) Tikslumai(i) Iteracijos(i)];
    end
     Rez_fx
    % ------------------------------------
    % šaknų tikslinimas funkcijai g(x)
    % ------------------------------------
    % šaknų intervalų atskyrimas funkcijai g(x)
    % ------------------------------------
    zingsnis = 0.2; % zingsnio nustatymas
    g_min = -3;
    g_max = 3;
    [SaknuIntervalai_gx]=SkenavimasPastoviu(g_min, g_max, zingsnis, g);
     fprintf( '-----------------------------------------------------\n');
     fprintf( 'Šaknų tikslinimas skenavimo metodu, mažinant žingsnį\n');
     fprintf( 'Funkcija g(x)=e^-^x*sin(x^2)*(x+2)\n');
     fprintf( '-----------------------------------------------------\n');
     fprintf( 'Stulpelių reikšmės:\n');
     fprintf( '1:2 - pradiniai šaknų tikslinimo intervalai\n');
     fprintf( '3 - šaknis\n');
     fprintf( '4 - tikslumas\n');
     fprintf( '5 - atliktų iteracijų kiekis\n');
     fprintf( '-----------------------------------------------------\n');
    Saknys_intervalai_gx = [];
    Tikslumai = [];
    Iteracijos = [];
    tikslumas = 1e-9;
    for i=1:length(SaknuIntervalai_gx)
        x_min = SaknuIntervalai_gx(i, 1);
        x_max = SaknuIntervalai_gx(i, 2);
        draw = 0;
        if (sign(g(x_min)) ~= sign(g(x_max)))
            iteracijos_sk = 0;
            [a, b, it, t]=SkenavimasRekursija(x_min, x_max, zingsnis, tikslumas, g, iteracijos_sk,
draw);
            Saknys_intervalai_gx = [Saknys_intervalai_gx; a b];
            Iteracijos = [Iteracijos; it];
            Tikslumai = [Tikslumai; t];
        end
    end
    Saknys_gx = (Saknys_intervalai_gx(:,1) + Saknys_intervalai_gx(:,2))/2;
```

```
Rez_gx = [];
for i=1:length(Saknys_gx)
    Rez_gx = [Rez_gx; SaknuIntervalai_gx(i,:) Saknys_gx(i) Tikslumai(i) Iteracijos(i)];
end
Rez_gx
% ------------------------------------
% Kvazi-Niutono (kirstinių) metodas
eps = 1e-9;
% ------------------------------------
% šaknų tikslinimas daugianariui f(x)
% ------------------------------------
 fprintf( '-----------------------------------------------------\n');
 fprintf( 'Šaknų tikslinimas Kvazi-Niutono (kirstinių) metodu\n');
 fprintf( 'Daugianaris f(x)=-0.67x^4+2.51x^3+2.27x^2-4.02x-2.48\n');
 fprintf( '-----------------------------------------------------\n');
 fprintf( 'Stulpelių reikšmės:\n');
 fprintf( '1 - pirmasis pradinis artinys\n');
 fprintf( '2 - antrasis pradinis artinys\n');
 fprintf( '3 - šaknis\n');
 fprintf( '4 - tikslumas\n');
 fprintf( '5 - atliktų iteracijų kiekis\n');
 fprintf( '-----------------------------------------------------\n');
Tikslumai = [];
Iteracijos = [];
Saknys_fx = [];
Artiniai = [];
iteracijos_sk_max = 200;
figure(1); grid on; hold on;
for i=1:length(SaknuIntervalai_fx)
    x0 = SaknuIntervalai_fx(i, 1);
    x01 = SaknuIntervalai_fx(i, 2);
    npoints=1000;
    x=x0:(x01-x0)/(npoints-1):x01;
    axis([(x0-0.01) (x01+0.01) -2.5 8.5]);
    Artiniai = [Artiniai; x0 x01];
    fxn = f(x0);
    fxn1 = f(x01);
    xn = x0;
    xn_plot = x0;
    xn1_plot = x01;
    fxn_plot = f(x0);
    fxn1_plot = f(x01);
    dfxn = (fxn1 - fxn)/(x01-x0);
    tikslumas = 1;
    iteracijos_sk = 0;
    while tikslumas > eps
        iteracijos_sk = iteracijos_sk + 1;
        if (iteracijos_sk > iteracijos_sk_max)
            fprintf('Virsytas leistinas iteraciju skaicius');
            break;
        end
        xn1 = xn - fxn/dfxn;
        if(i == 1 && iteracijos_sk < 7)
            plot(x,f(x),'r-');
            plot([x0 x01],[0 0],'b-');
            plot(x0,0,'mp');
            h = findobj(gca,'Type','line');h1=h(1);
            plot([xn_plot,xn_plot,xn1_plot,xn1_plot],[0,fxn_plot,fxn1_plot,0],'k-');
            plot([xn,xn,xn1],[0,fxn,0],'k-');
            delete(h1);plot(xn1,0,'mp');h = findobj(gca,'Type','line');h1=h(1);
            input('Press Enter'), figure(1);
        end
        fxn1 = f(xn1);
        dfxn = (fxn1 - fxn)/(xn1 - xn);
```

```
            xn = xn1;
            fxn = f(xn);
            tikslumas = abs(fxn);
        end
        Iteracijos = [Iteracijos; iteracijos_sk];
        Tikslumai = [Tikslumai; tikslumas];
        Saknys_fx = [Saknys_fx; xn];
    end;
    close all;
    Rez_fx = [];
    for i=1:length(Saknys_fx)
        Rez_fx = [Rez_fx; Artiniai(i, :) Saknys_fx(i) Tikslumai(i) Iteracijos(i)];
    end;
    Rez_fx
    % ------------------------------------
    % šaknų tikslinimas funkcijai g(x)
    % ------------------------------------
     fprintf( '-------------------------------------------------------\n');
     fprintf( 'Šaknų tikslinimas Kvazi-Niutono (kirstinių) metodu\n');
     fprintf( 'Funkcija g(x)=sin(x)ln(x)-(x/6)\n');
     fprintf( '-------------------------------------------------------\n');
     fprintf( 'Stulpelių reikšmės:\n');
     fprintf( '1 - pirmasis pradinis artinys\n');
     fprintf( '2 - antrasis pradinis artinys\n');
     fprintf( '3 - šaknis\n');
     fprintf( '4 - tikslumas\n');
     fprintf( '5 - atliktų iteracijų kiekis\n');
     fprintf( '-------------------------------------------------------\n');
    Tikslumai = [];
    Iteracijos = [];
    Saknys_gx = [];
    Artiniai = [];
    iteracijos_sk_max = 200;
    for i=1:length(SaknuIntervalai_gx)
        x0 = SaknuIntervalai_gx(i, 1) - 0.2;
        x01 = SaknuIntervalai_gx(i, 1) - 0.1;
        Artiniai = [Artiniai; x0 x01];
        fxn = g(x0);
        fxn1 = g(x01);
        dfxn = (fxn1 - fxn)/(x01-x0);
        xn = x0;
        tikslumas = 1;
        iteracijos_sk = 0;
        while tikslumas > eps
            iteracijos_sk = iteracijos_sk + 1;
            if (iteracijos_sk > iteracijos_sk_max)
                fprintf('Virsytas leistinas iteraciju skaicius');
                break;
            end
            xn1 = xn - fxn/dfxn;
            fxn1 = g(xn1);
            dfxn = (fxn1 - fxn)/(xn1 - xn);
            xn = xn1;
            fxn = g(xn);
            tikslumas = abs(fxn);
        end
        Iteracijos = [Iteracijos; iteracijos_sk];
        Tikslumai = [Tikslumai; tikslumas];
        Saknys_gx = [Saknys_gx; xn];
    end;
    Rez_gx = [];
    for i=1:length(Saknys_gx)
        Rez_gx = [Rez_gx; Artiniai(i, :) Saknys_gx(i) Tikslumai(i) Iteracijos(i)];
    end;
```

```
    Rez_gx
    % Matlab funkcijos
    % Daugianaris f(x)
     fprintf( '-----------------------------------------------------\n');
     fprintf( 'MATLAB funkcijos\n');
     fprintf( '-----------------------------------------------------\n');
     fprintf( 'Daugianaris f(x)\n');
     fprintf( '-----------------------------------------------------\n');
    a = [0.67 -2.51 -2.27 4.02 2.48];
    saknys_roots = roots(a)
    for i=1:length(SaknuIntervalai_fx)
        fzero(f, SaknuIntervalai_fx(i, 1))
    end
    % Matlab funkcijos
    % Funkcija g(x)
     fprintf( '-----------------------------------------------------\n');
     fprintf( 'Funkcija g(x)\n');
     fprintf( '-----------------------------------------------------\n');
    for i=1:length(SaknuIntervalai_gx)
        fzero(g, SaknuIntervalai_gx(i, 1))
    end
end
```