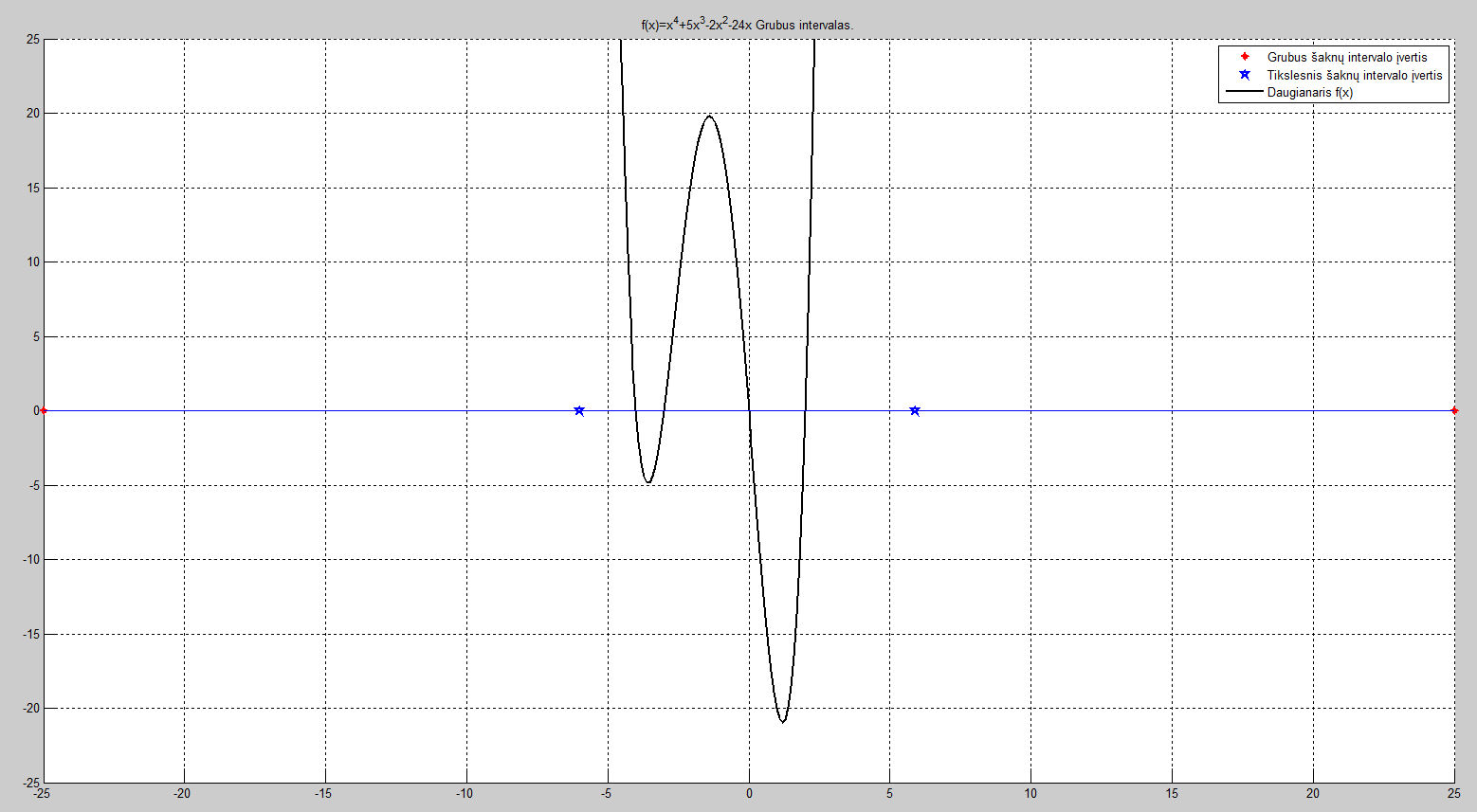
# Netiesinių lygčių sprendimas

Duotos dvi netiesinės lygtys: daugianaris f(x) = 0 ir transcendentinė funkcija g(x) = 0.

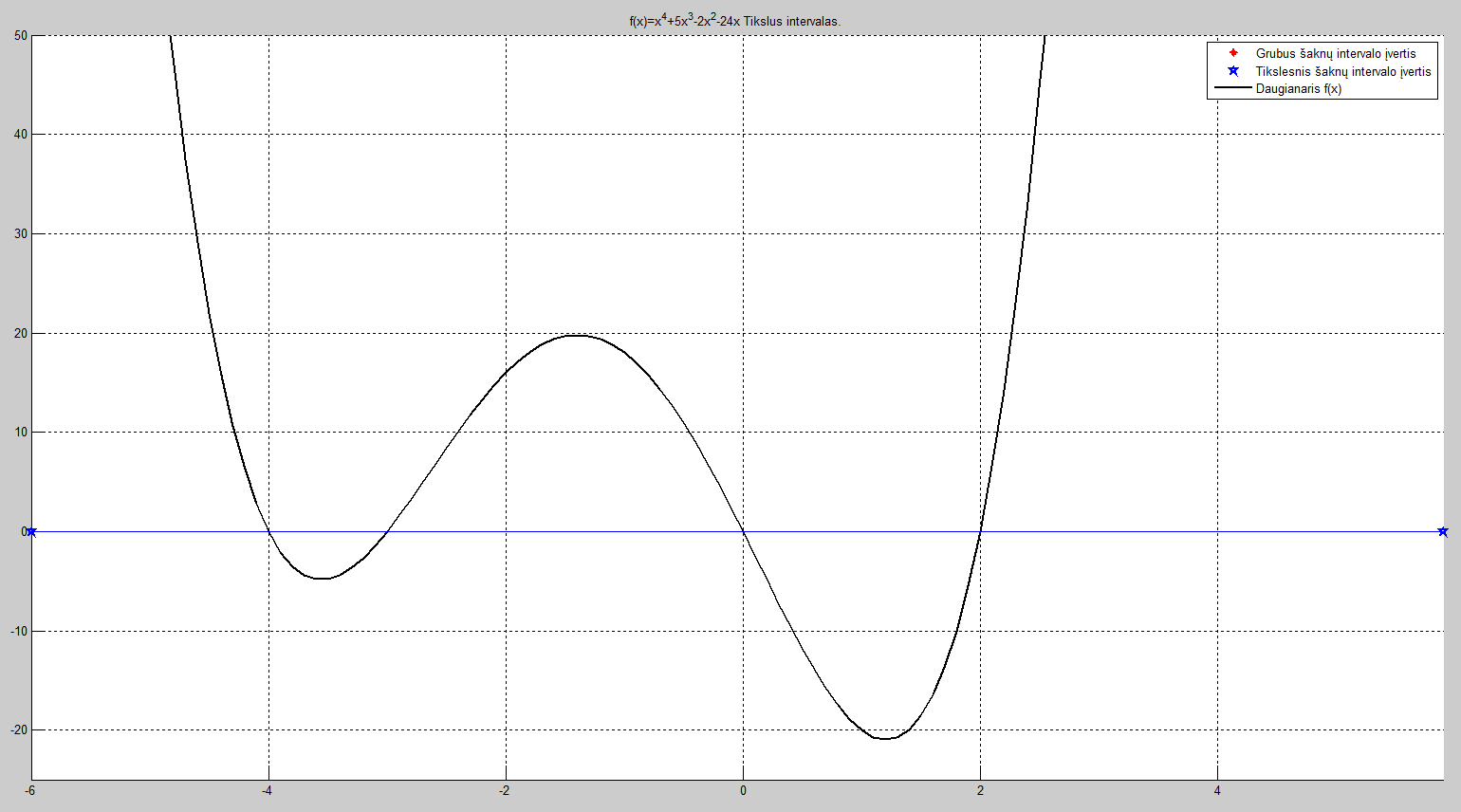
|  |  |  |
| --- | --- | --- |
| Nr. | Daugianaris f(x) | Funkcija g(x) |
| 16 | 𝑥4+5𝑥3−2𝑥2−24𝑥 | sin(𝑥)ln(𝑥)−𝑥/6; 1≤𝑥≤20 |
| Sprendimo metodai: skenavimo, stygų, Kvazi-Niutono (kirstinių) | | |

## Lygties f(x) = 0 (f(x) – daugianaris) sprendimas

## Daugianario šaknų intervalo įverčiai.



1 pav. Daugianario šaknų intervalo įverčiai.



2 pav. Grafinis funkcijos vaizdas tikslesniame šaknų intervale.

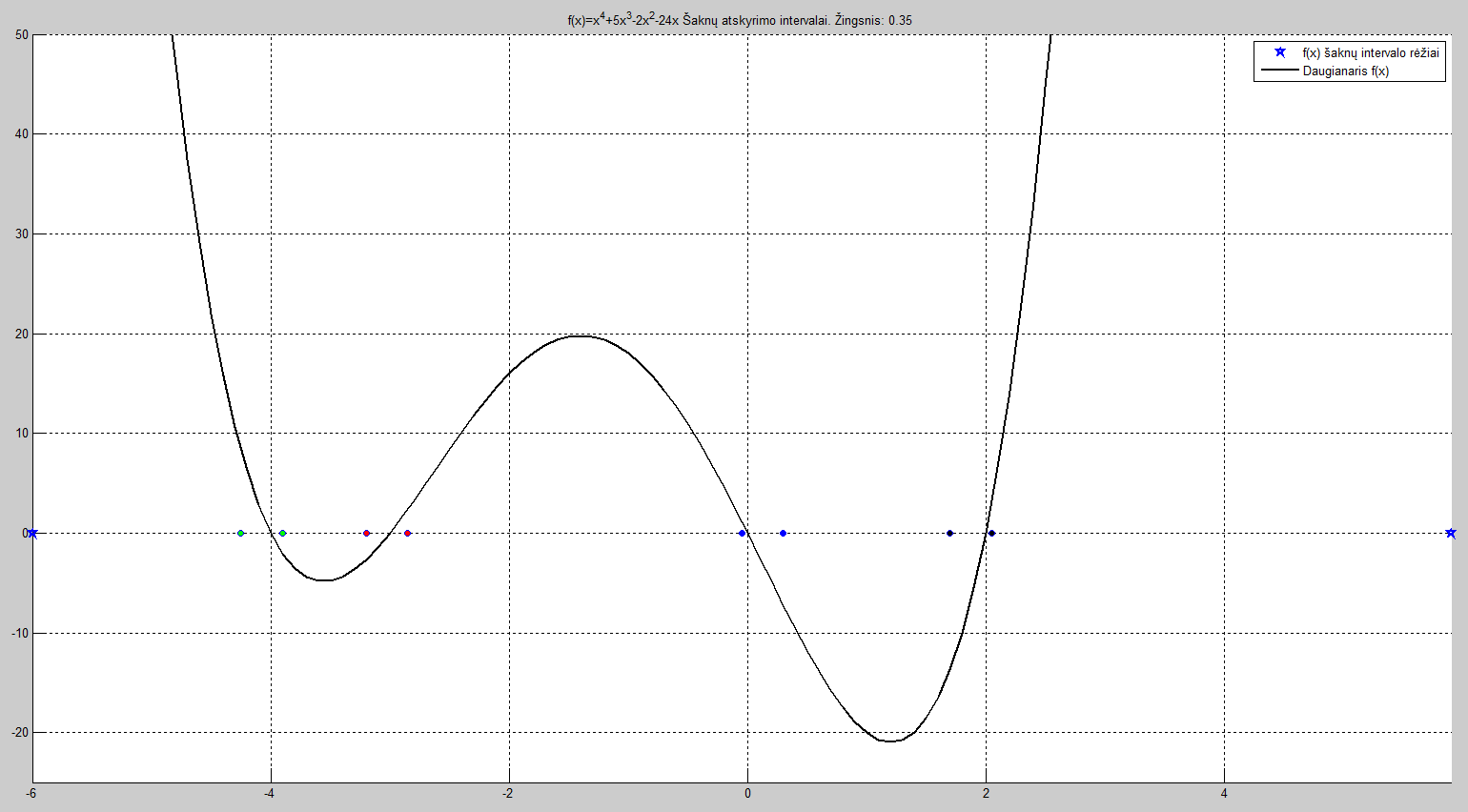
1 lentelė. Šaknų intervalo įverčiai.

|  |  |
| --- | --- |
| Grubus lygties f(x) = 0 šaknų intervalo įvertis | [-25; 25] |
| Tikslesnis lygties f(x) = 0 šaknų intervalo įvertis | [-6; 5,899] |

**Išvados:** Iš grafikų (1 pav., 2 pav.) bei apskaičiuotų šaknų intervalo įverčių (1 lentelė) matome, kad daugianaris turi 4 šaknis, garantuotai esančias intervale [-25; 25] arba tikslesniame - [-6; 5,899].

## Šaknų atskyrimas skenavimo metodu.

Skenavimas atliekamas intervale [-6; 5,899], skenavimo žingsnis lygus 0,35.



3 pav. Daugianario šaknų atskyrio intervalai.

2 lentelė. Šaknies atskyrimo intervalai.

|  |  |
| --- | --- |
| Intervalo Nr. | Intervalas |
| 1 | [-4.250000000000002; -3.900000000000002] |
| 2 | [-3.200000000000002; -2.850000000000001] |
| 3 | [-0.050000000000001; 0.299999999999999] |
| 4 | [1.699999999999999; 2.049999999999999] |

**Išvados:** Iš grafiko (3 pav.) bei šaknies atskyrimo intervalų matome, kad visos daugianario f(x) šaknys patenka tarp nurodytų šaknų atskyrimo intervalų bei taip pat patenka tarp tikslesnio šaknų intervalų įverčio rėžių. Gauti 4 šaknų intervalai, o kadangi iš grafiko matome, kad daugianaris turi 4 šaknis, todėl galime spręsti, jog žingsnis buvo parinktas teisingai – rastas reikiamas skaičius šaknų intervalų.

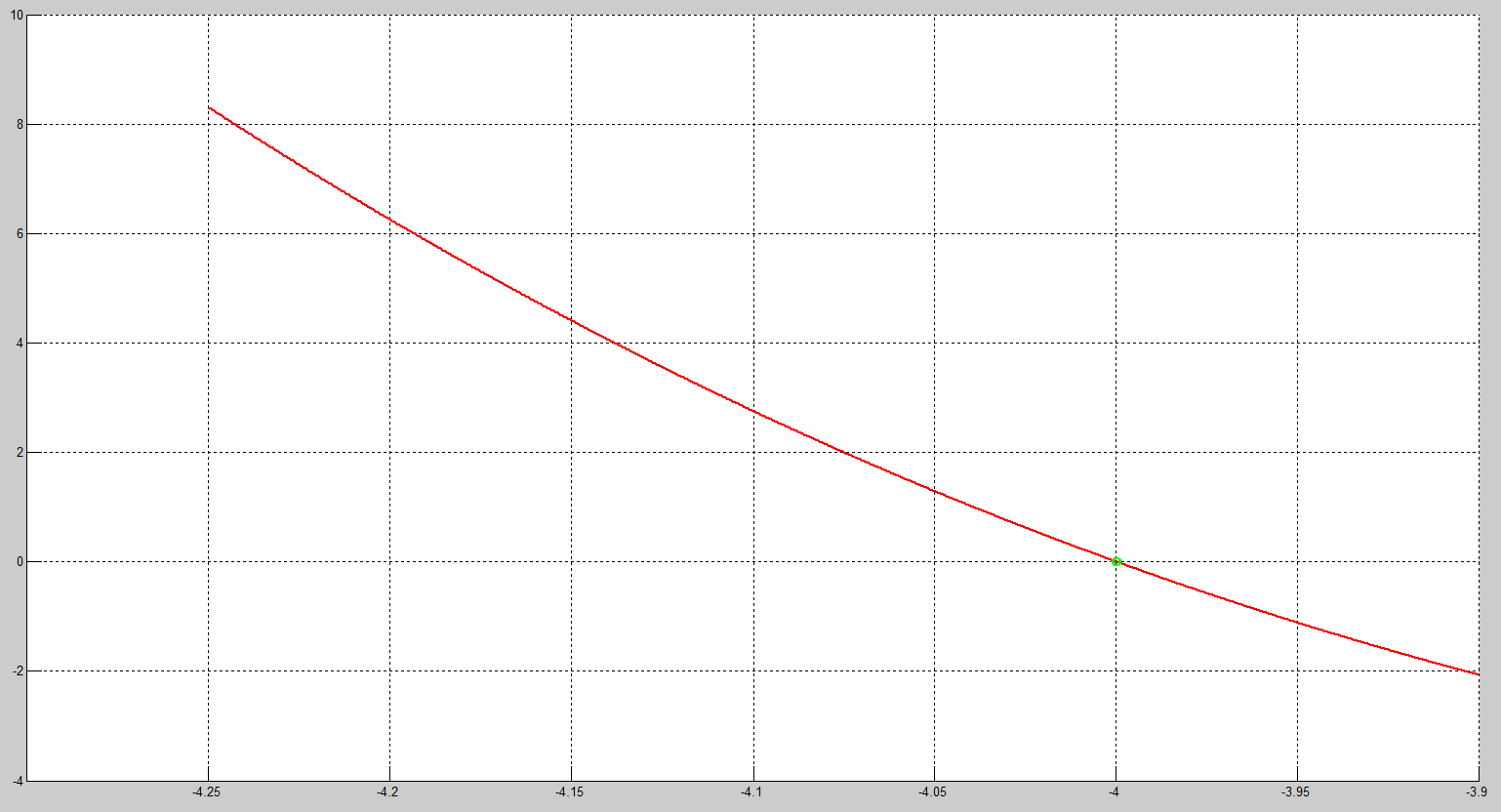
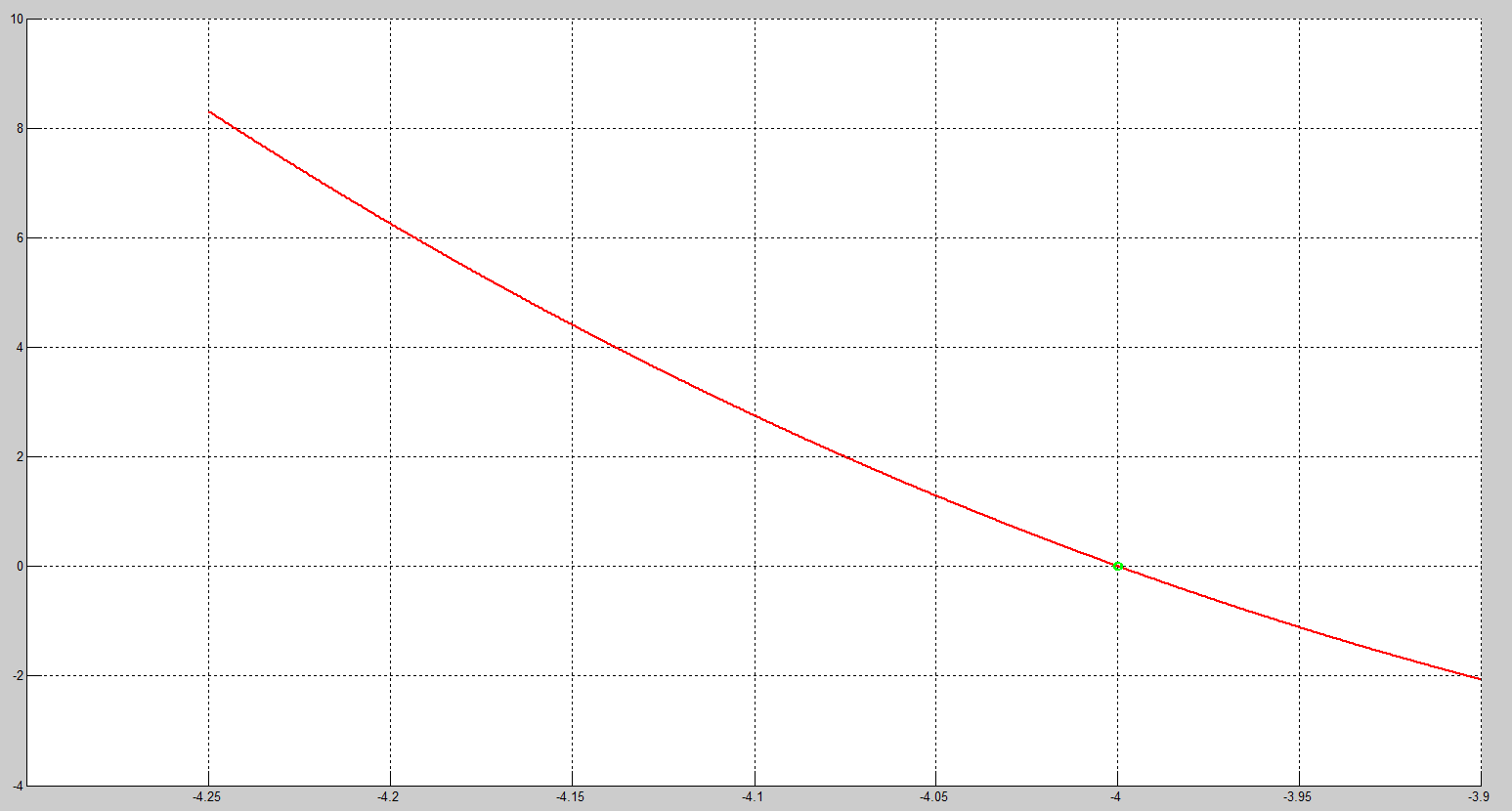
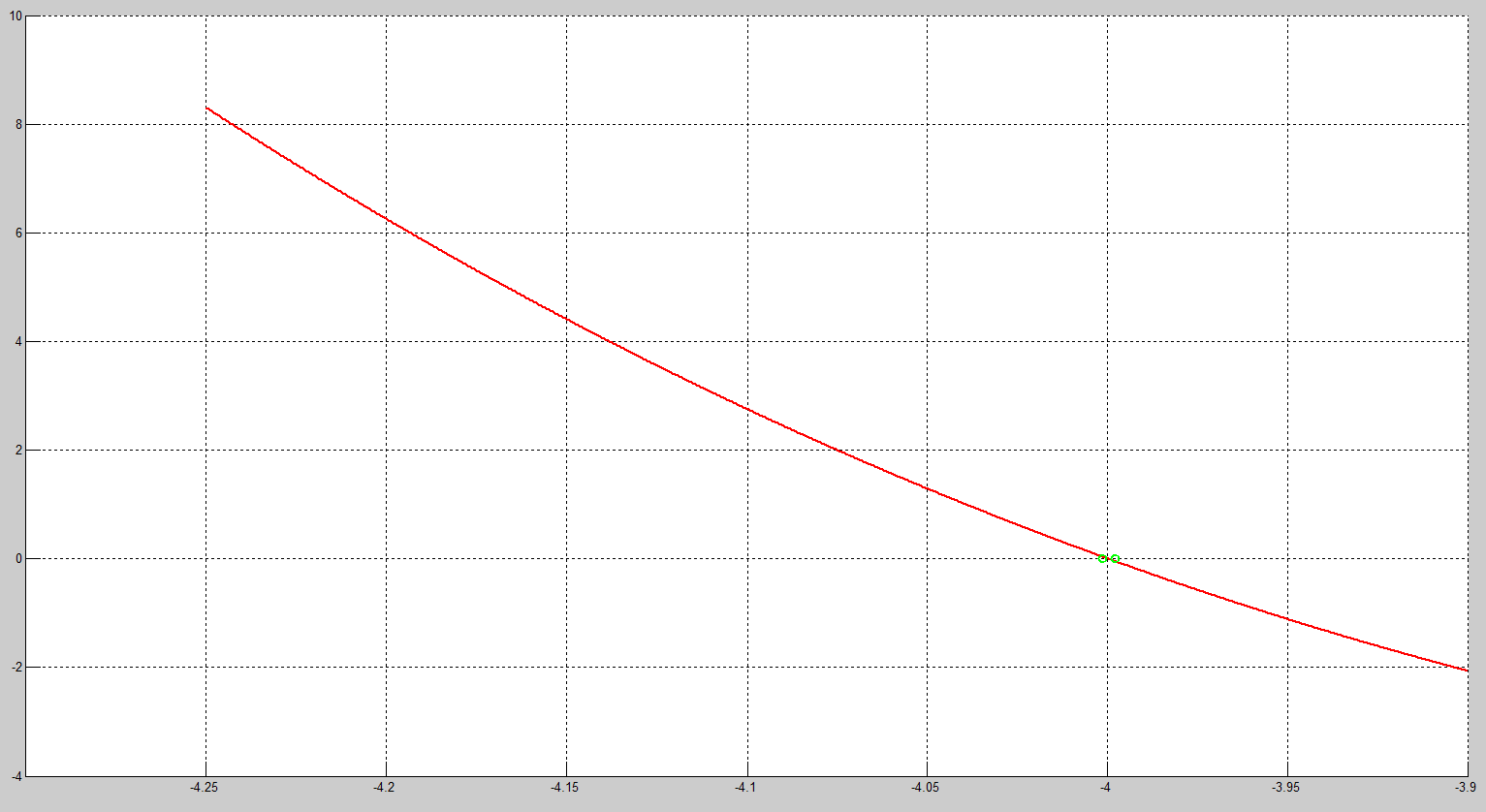
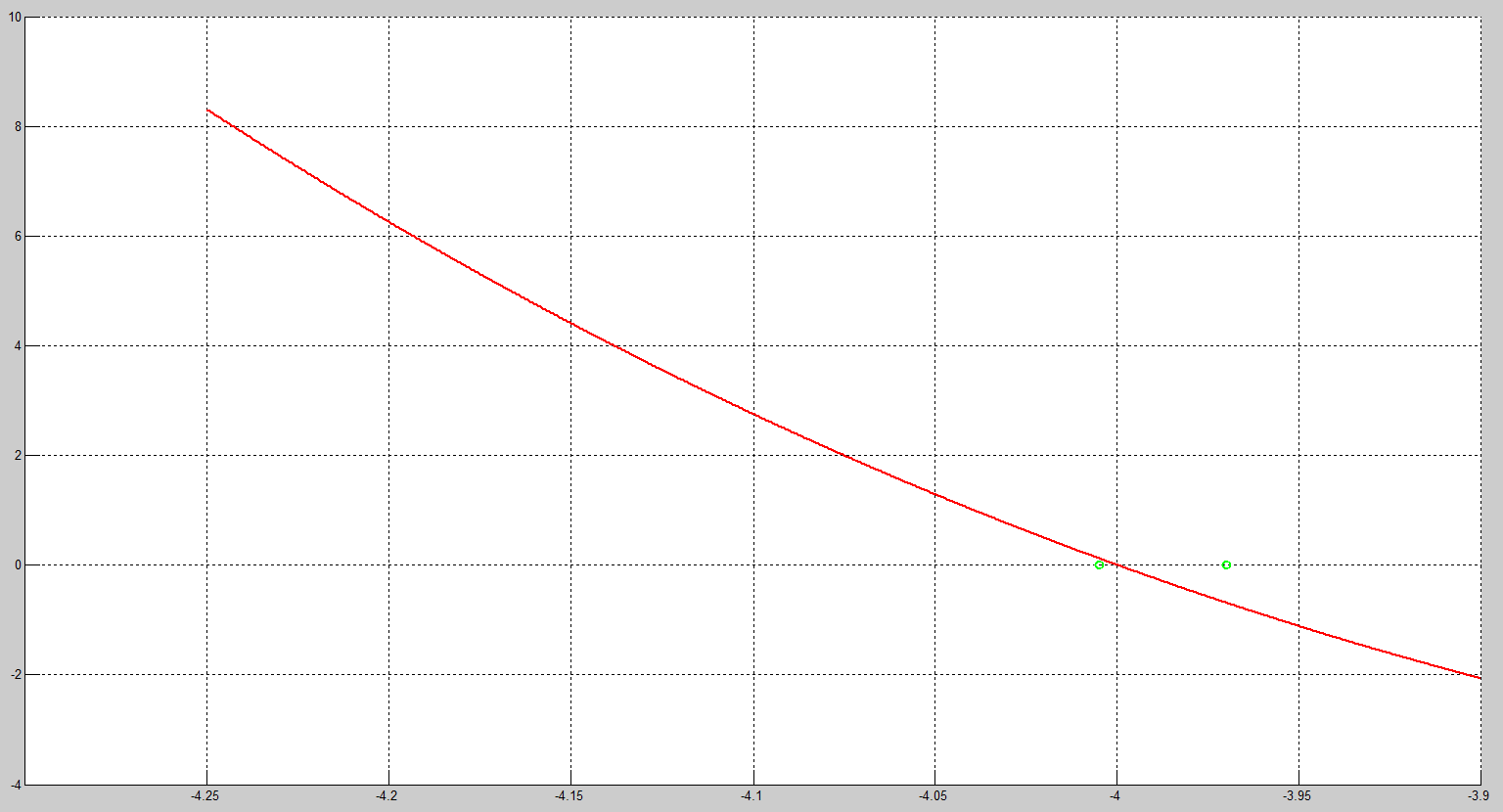
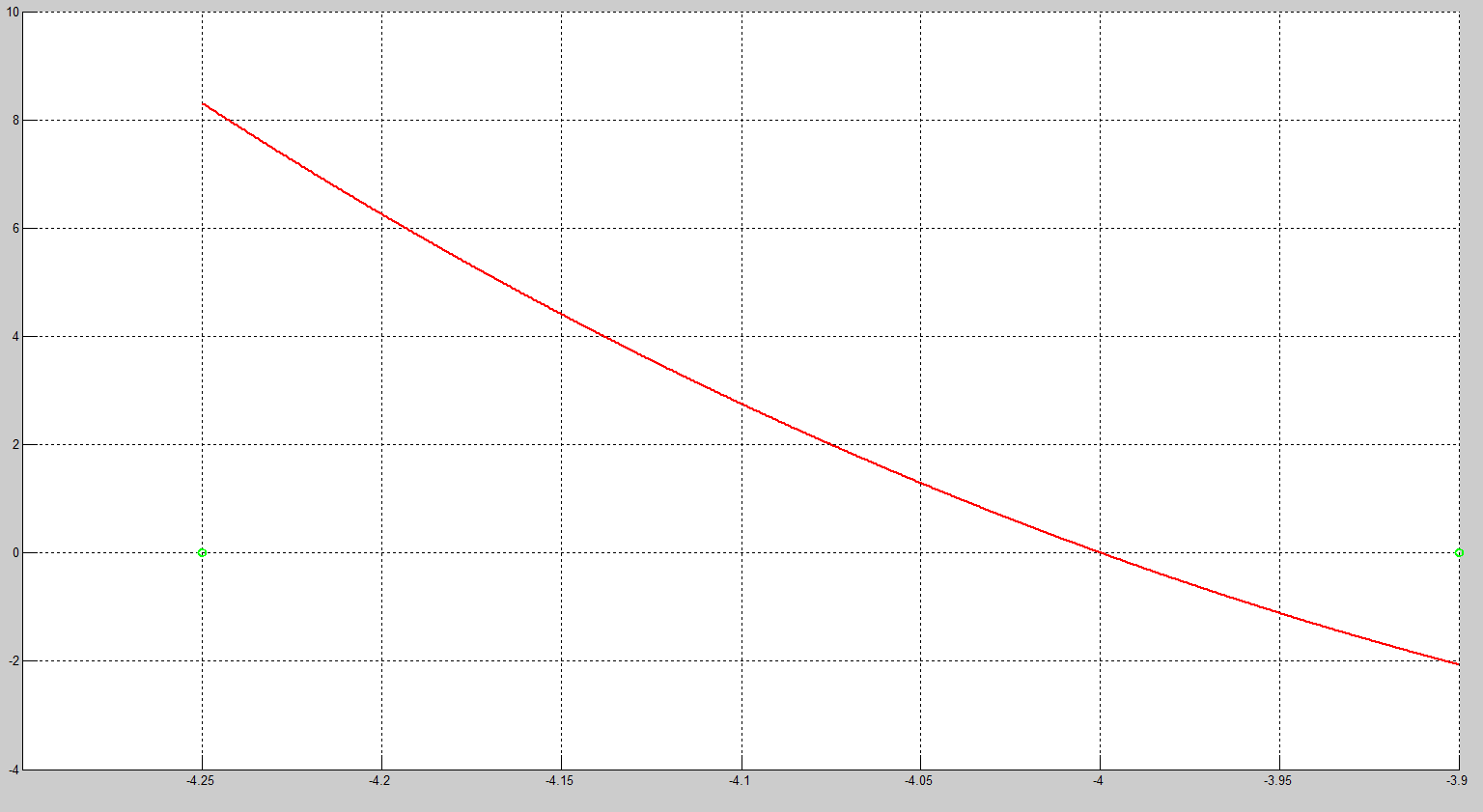
## Šaknų tikslinimas skenavimo, stygų, Kvazi-Niutono (kirstinių) metodais.

Tariama, kad xg yra šaknis (stabdomi skaičiavimai), jei |f(xg)| < 1e-9. Skaičiavimuose naudojamas šaknies tikslumo įvertis |f(xg)|.

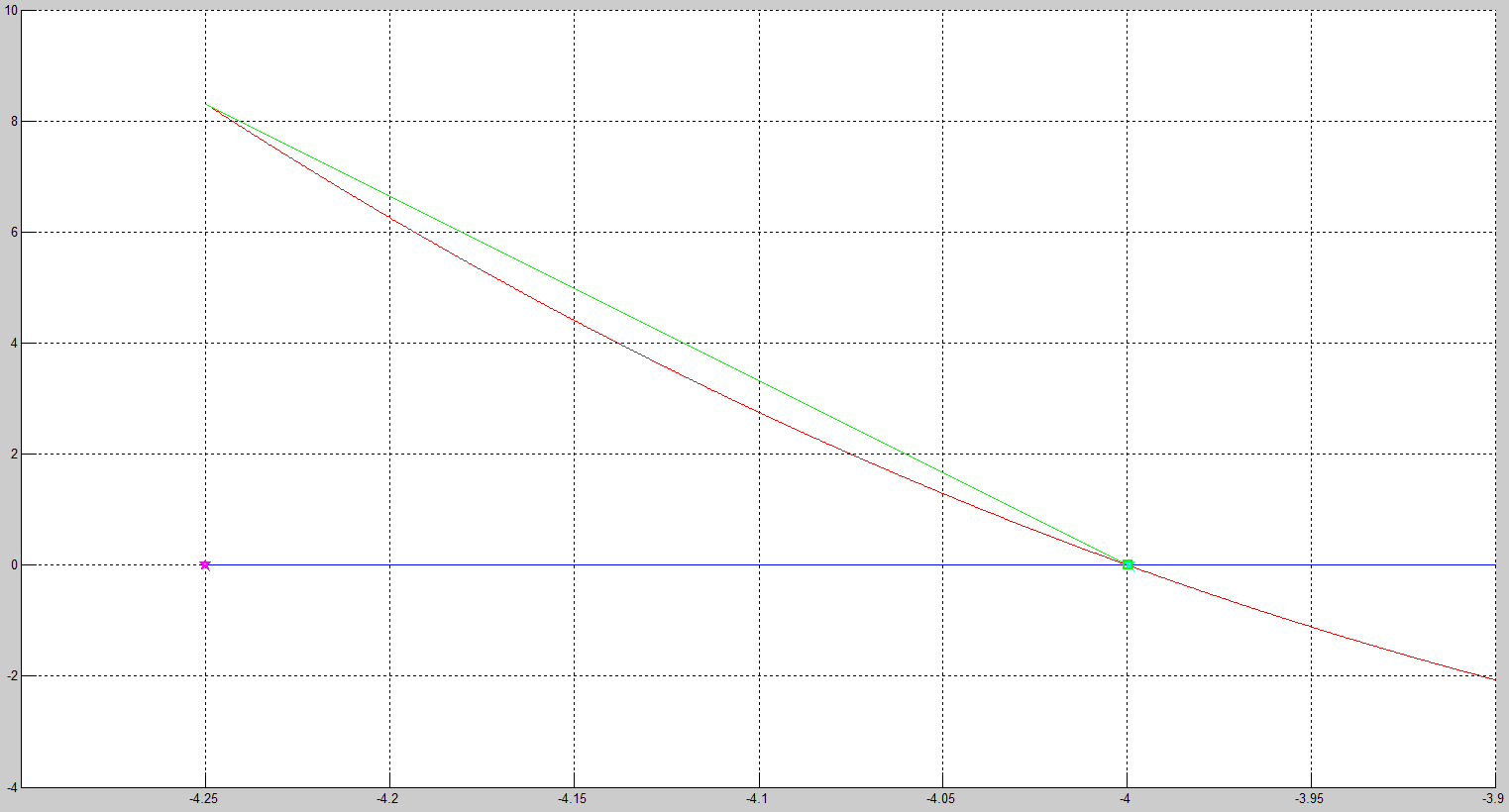
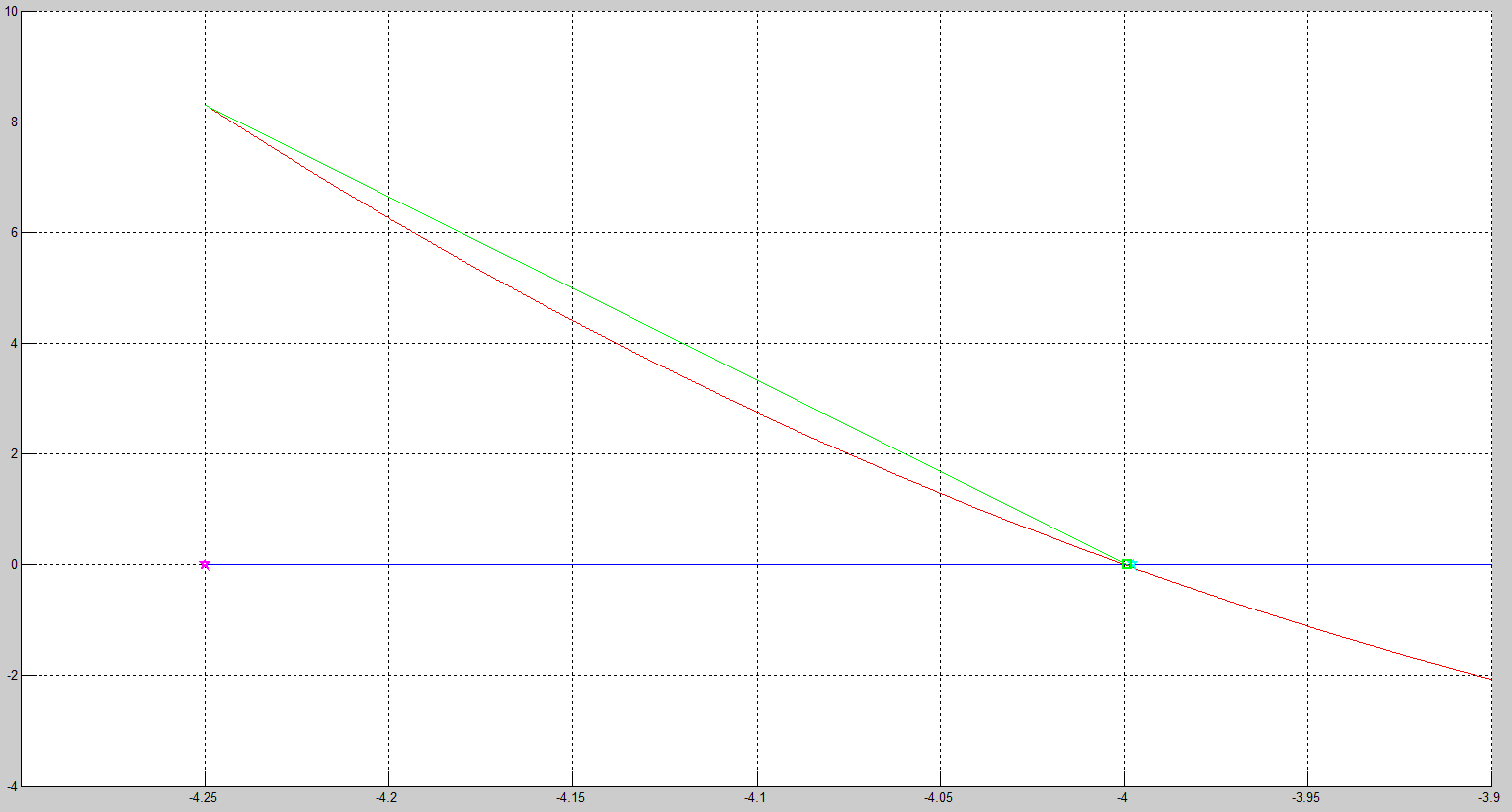
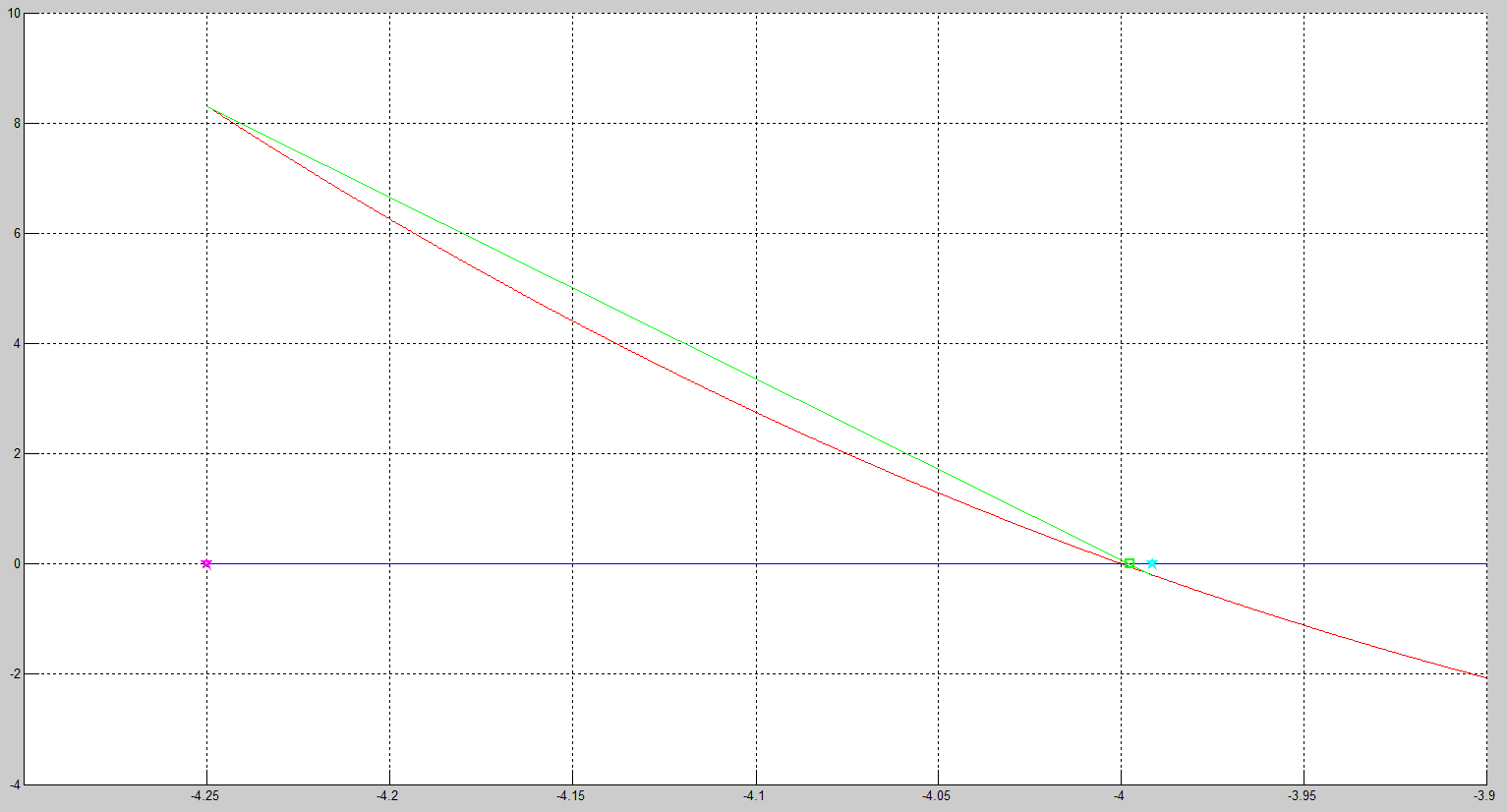
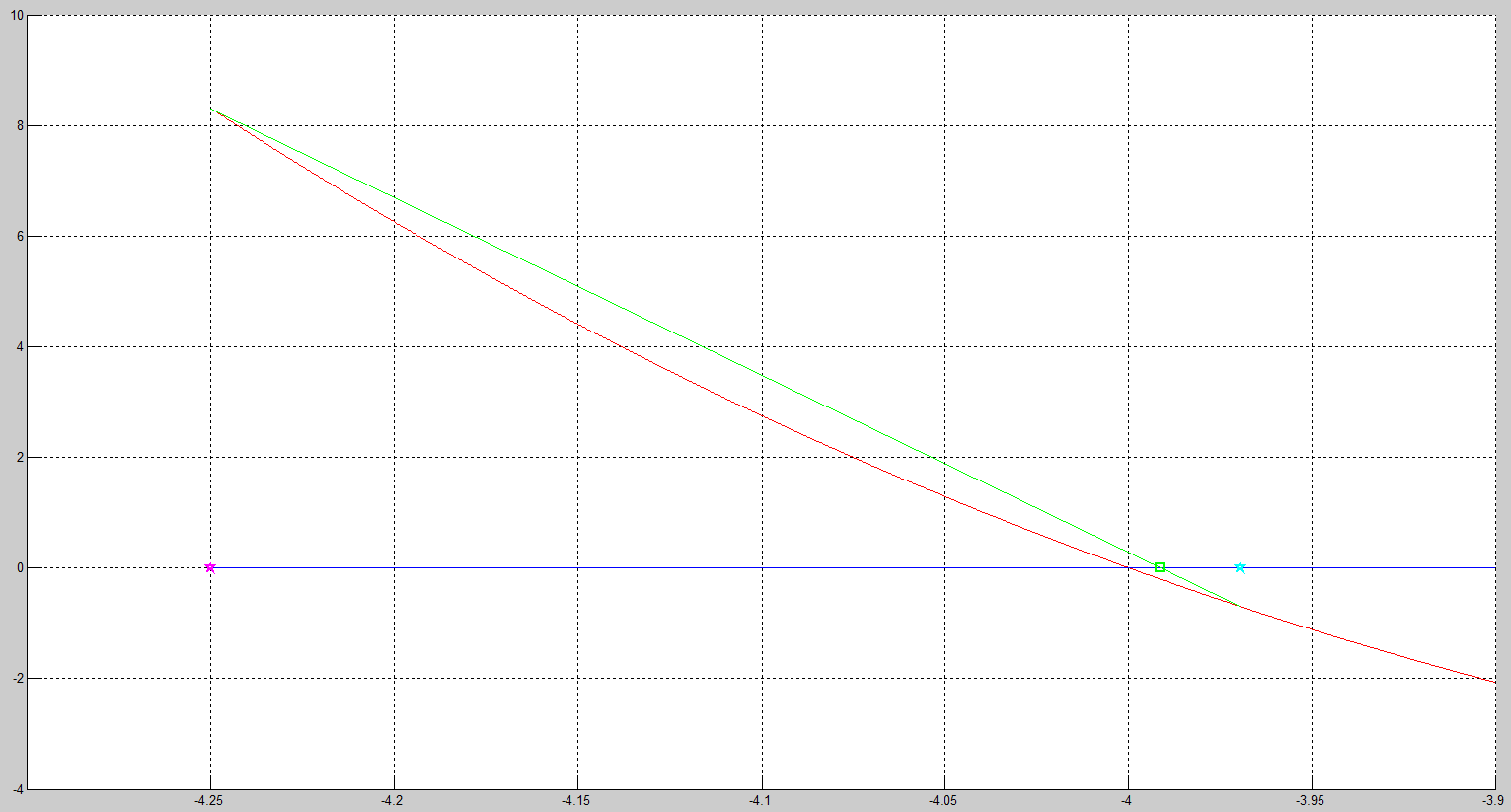
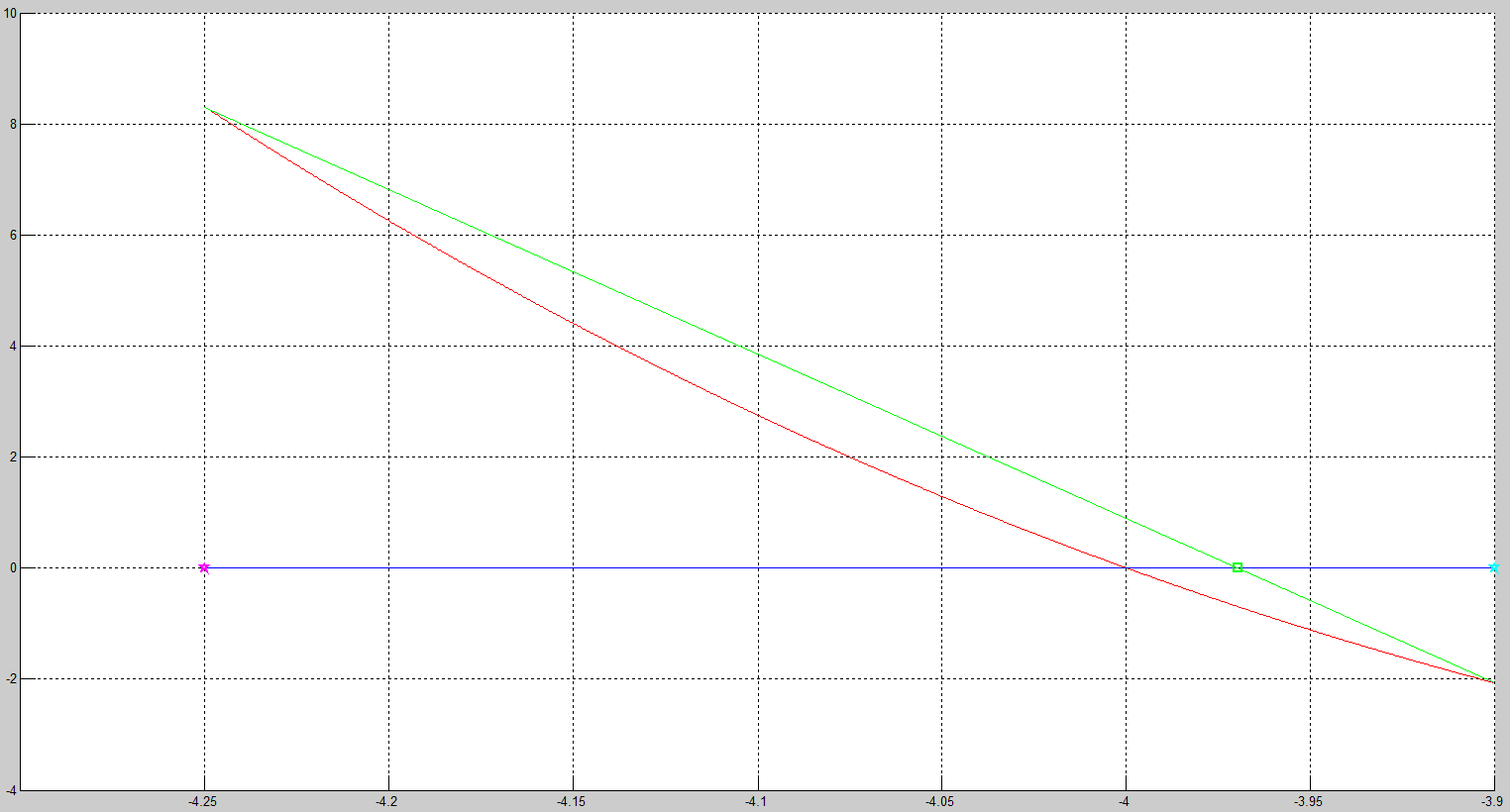
3 lentelė. Rezultatų lentelė.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Skenavimo metodas | Pradinis intervalas | Šaknis | Tikslumas | Iteracijų skaičius |
| [-4.250000000000002; -3.900000000000002] | -3.999999999750001 | 0.000000003500000 | 10 |
| [-3.200000000000002; -2.850000000000001] | -2.999999998750003 | 0.000000003500000 | 10 |
| [-0.050000000000001; 0.299999999999999] | 0.000000000749999 | 0.000000003500000 | 10 |
| [1.699999999999999; 2.049999999999999] | 1.999999999250000 | 0.000000003500000 | 10 |
| Stygų metodas | Pradinis intervalas | Šaknis | Tikslumas | Iteracijų skaičius |
| [-4.250000000000002; -3.900000000000002] | -3.999999999962187 | 0.000000000907548 | 17 |
| [-3.200000000000002; -2.850000000000001] | -3.000000000028439 | 0.000000000426581 | 8 |
| [-0.050000000000001; 0.299999999999999] | -0.000000000000285 | 0.000000000006830 | 5 |
| [1.699999999999999; 2.049999999999999] | 1.999999999996428 | 0.000000000214314 | 8 |
| Kvazi-Niutono (kirstinių) metodas | Pradiniai artiniai | Šaknis | Tikslumas | Iteracijų skaičius |
| [-4.250000000000002; -3.900000000000002] | -4.000000000000015 | 0.000000000000355 | 6 |
| [-3.200000000000002; -2.850000000000001] | -2.999999999999908 | 0.000000000001393 | 5 |
| [-0.050000000000001; 0.299999999999999] | 0.000000000000000 | 0.000000000000000 | 4 |
| [1.699999999999999; 2.049999999999999] | 2.000000000015661 | 0.000000000939650 | 5 |
| MATLAB funkcijos | Pradinis intervalas | Šaknis (fzero) | Šaknis (roots) |  |
| -4.250000000000002 | -4 | -4.000000000000001 |  |
| -3.200000000000002 | -3.000000000000001 | -3.000000000000000 |  |
| -0.050000000000001 | -7.948517295054264e-22 | 0 |  |
| 1.699999999999999 | 2 | 2.000000000000003 |  |

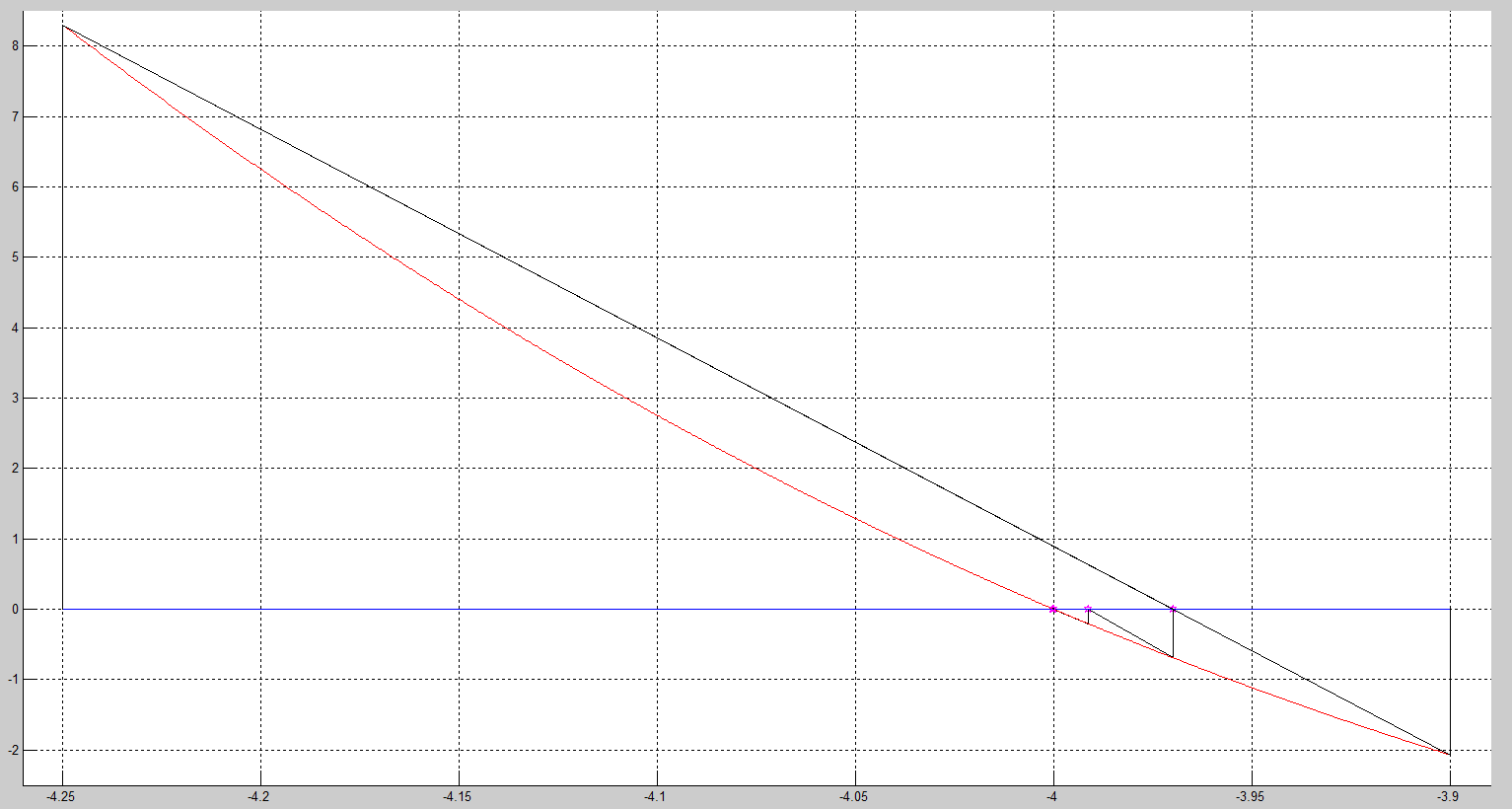
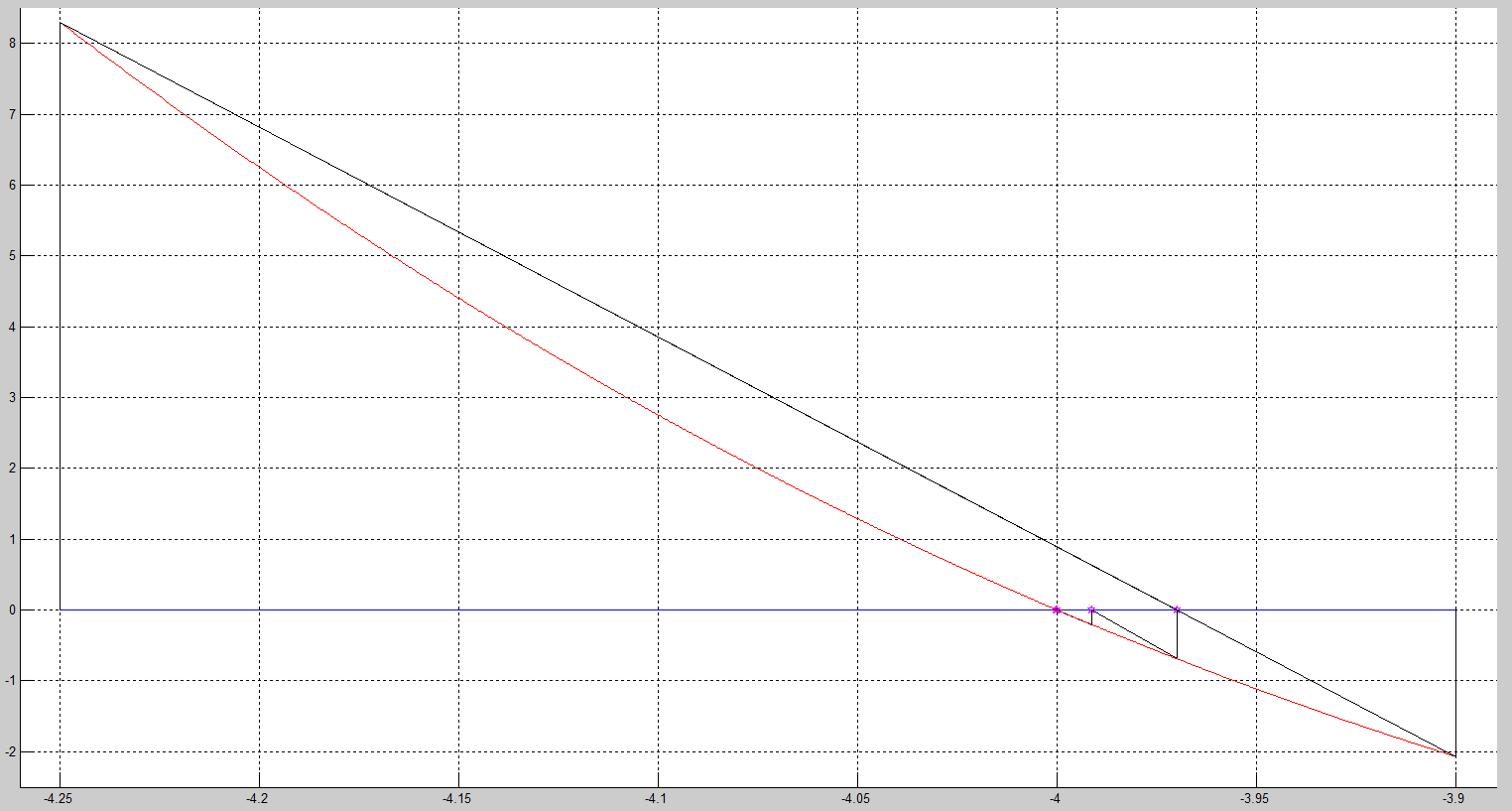
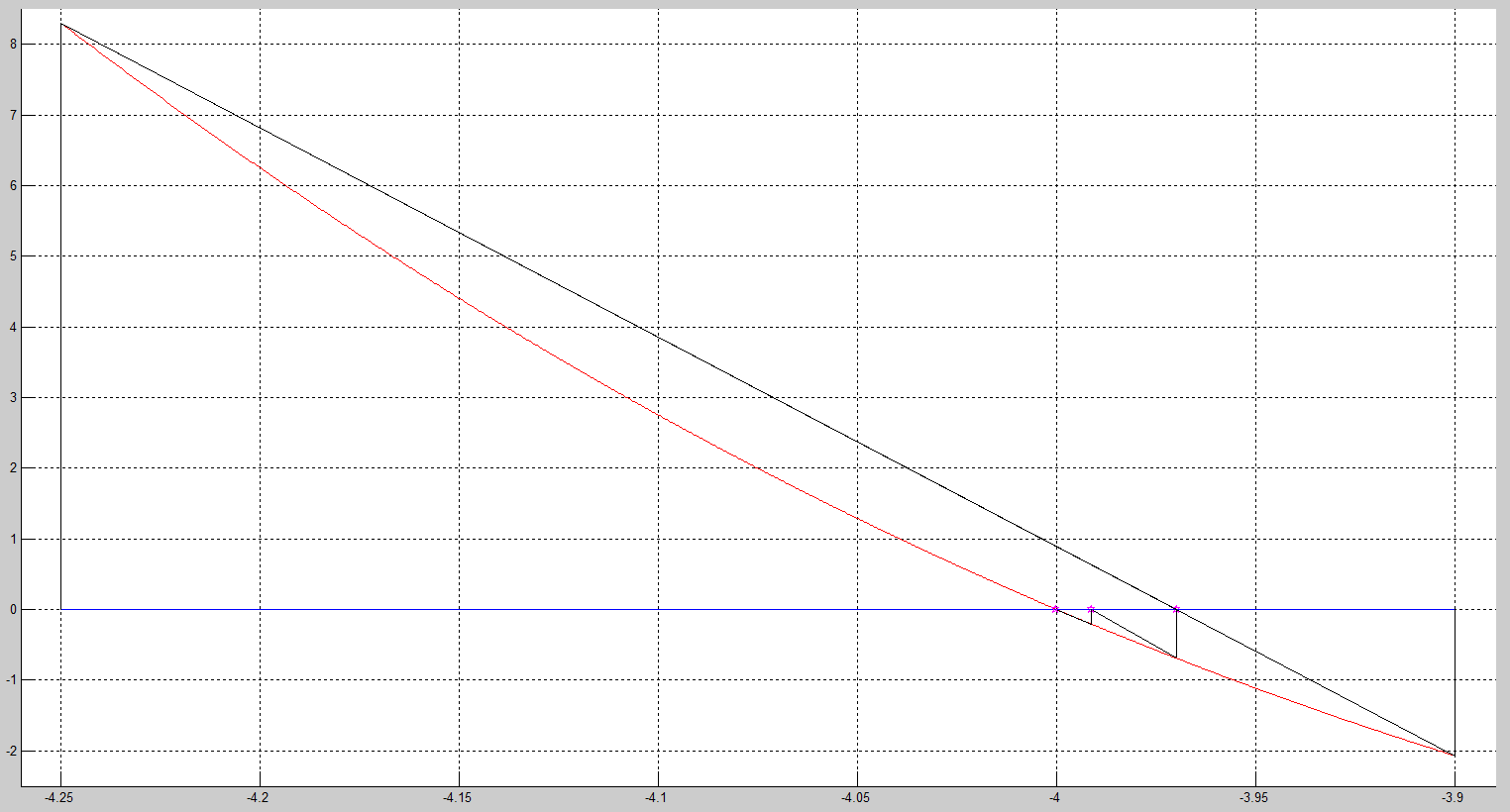
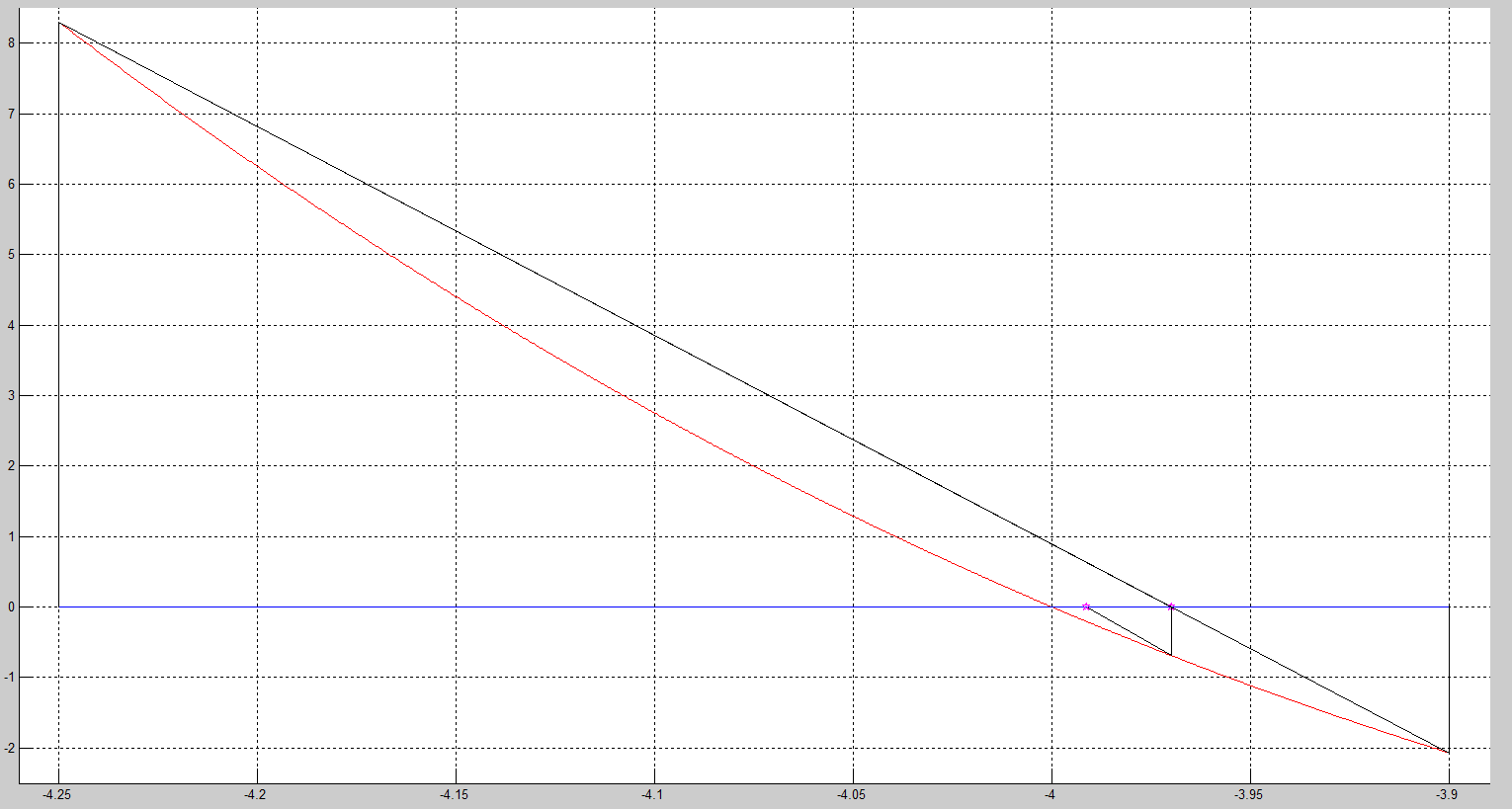
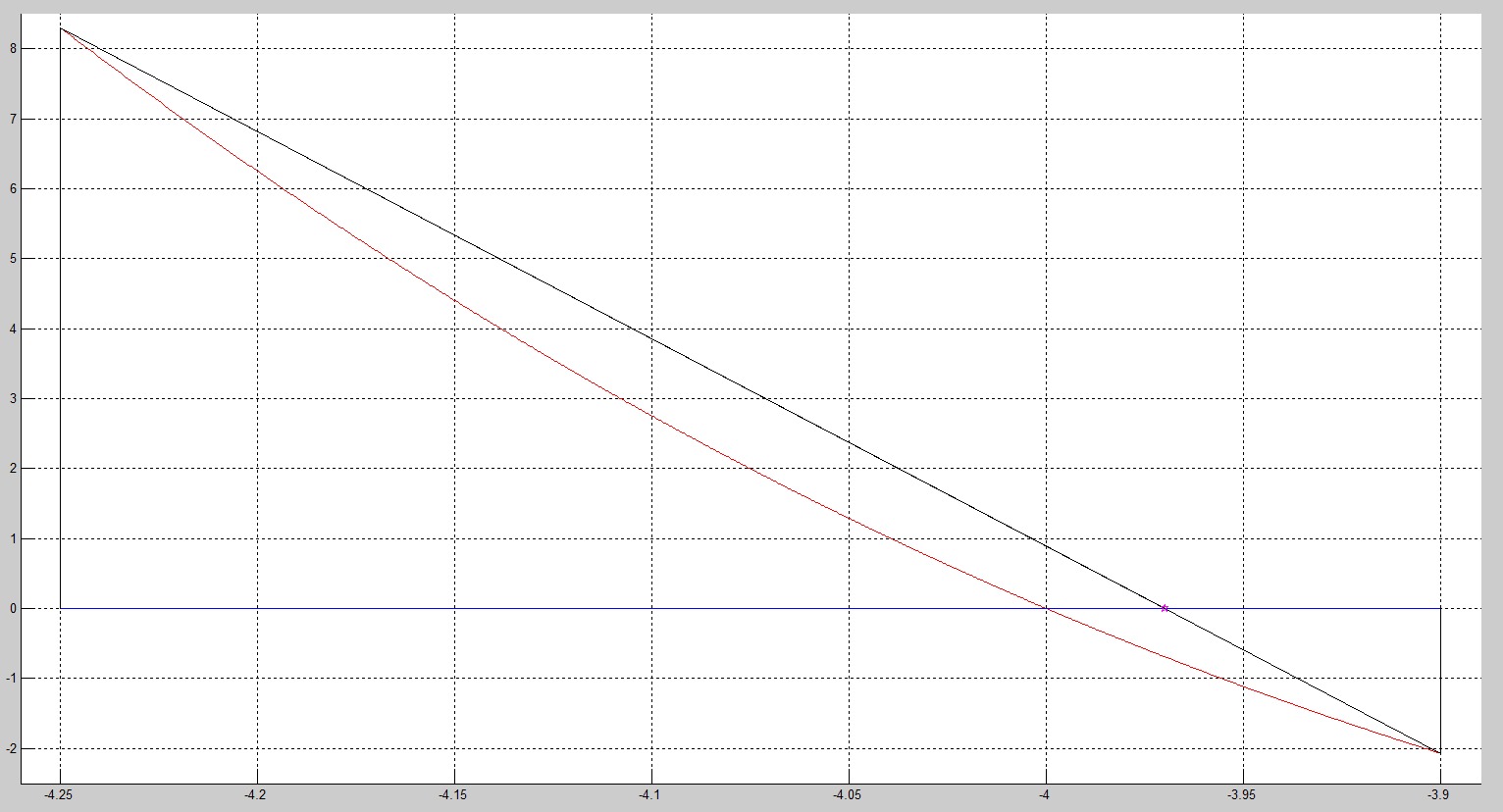
**Išvados:** Iš rezultatų lentelės (3 lentelė) matome, jog visais metodais rastos tos pačios šaknys, todėl galime teigti, jog metodai veikia teisingai. Iš išbandytų metodų mažiausiai iteracijų atlieka Kvazi-Niutono metodas. Verta paminėti, jog naudotas rekursinis skenavimo mažėjančiu žingsniu metodas, todėl iteracijų skaičius visų šaknų paieškai buvo vienodas, nes nagrinėti tokio paties dydžio šaknų intervalai. Taip pat verta paminėti ir tai, jog Kvazi-Niutono metodas atlieka ne tik mažiausiai iteracijų, tačiau ir šio metodo tikslumas taip pat lenkia tiek skenavimo, tiek stygų metodus. MATLAB funkcijos fzero ir roots taip pat rado tas pačias polinomo šaknis, todėl galime teigti, kad metodų algoritmai veikia teisingai.



4 pav. Šaknies **xg = - 4** tikslinimo **skenavimo** metodu vizualizacija. Raudona linija brėžiama funkcija, žaliais taškais žymimi iteracijoje nagrinėjamo intervalo galai.



5 pav. Šaknies **xg = -4** tikslinimo **stygų** metodu vizualizacija. Raudona linija brėžiama funkcija, žalia – pagalbinės linijos, žalias kvadratėlis – vieta, kurioje brėžiama styga kerta X ašį, mėlyna ir purpurinė žvaigždės žymi stygos galų X ašies koordinatę.



6 pav. Šaknies **xg = -4** tikslinimo **Kvazi-Niutono** metodu vizualizacija. Raudona linija brėžiama funkcija, juoda – pagalbinės linijos, purpurinė žvaigždė žymi kirstinės vietą X ašyje.

**Išvados:** iš vizualizacijų matome, kad skenavimo algoritmo nagrinėjamas intervalas labai sparčiai artėja prie tikrosios šaknies reikšmės. Stygų metodo vizualizacija demonstruoja, kaip stygos metodu styga trumpėja bei artėja prie tikrosios reikšmės, o Kvazi-Niutono (kirstinių) metodo vizualizacija demonstruoja, kaip paimama vis kita kirstinės pabaigos (pradžios) reikšmė, artėjanti prie tikrosios šaknies reikšmės.

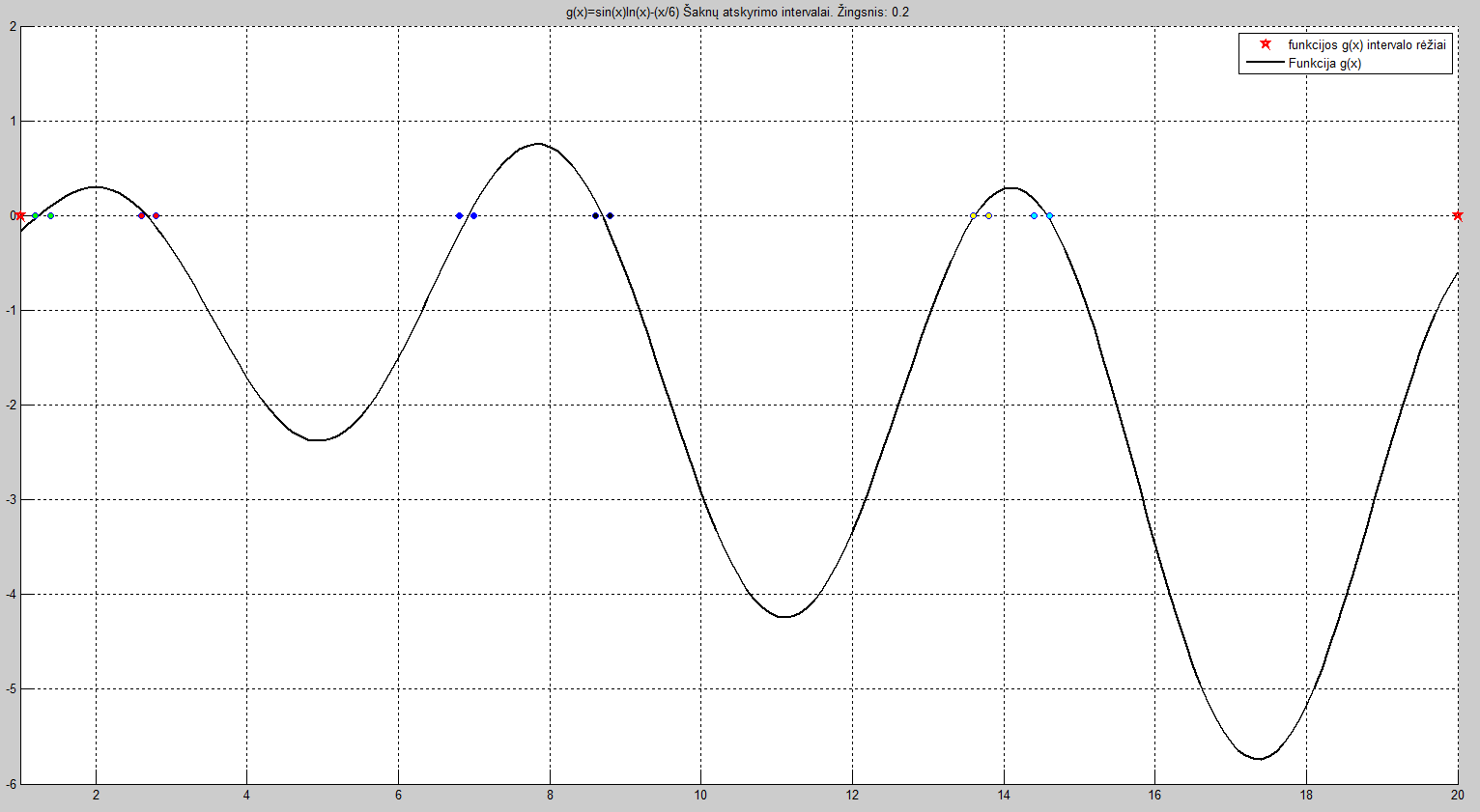
## Lygties g(x) = 0 (g(x) –transcendentinė funkcija) sprendimas

## Šaknų atskyrimas skenavimo metodu

Skenavimas atliekamas intervale [1; 20], skenavimo žingsnis lygus 0,2.

4 lentelė. Šaknies atskyrimo intervalai.

|  |  |
| --- | --- |
| Intervalo Nr. | Intervalas |
| 1 | [1.200000000000000; 1.400000000000000] |
| 2 | [2.600000000000000; 2.800000000000000] |
| 3 | [6.800000000000003; 7.000000000000004] |
| 4 | [8.600000000000001; 8.800000000000001] |
| 5 | [13.599999999999984; 13.799999999999983] |
| 6 | [14.399999999999981; 14.599999999999980] |



7 pav. Funkcijos šaknų atskyrimo intervalai.

**Išvados:** Iš grafiko (7 pav.) bei šaknies atskyrimo intervalų matome, kad funkcija g(x) turi 6 šaknis (6 kartus duotame intervale kertama X ašis). Taip pat gauti ir 6 šaknų intervalai, , todėl galime spręsti, jog žingsnis buvo parinktas teisingai – rastas reikiamas skaičius šaknų intervalų.

## Šaknų tikslinimas skenavimo, stygų, Kvazi-Niutono (kirstinių) metodais.

Tariama, kad xg yra šaknis (stabdomi skaičiavimai), jei |f(xg)| < 1e-9. Skaičiavimuose naudojamas šaknies tikslumo įvertis |f(xg)|.

5 lentelė. Rezultatų lentelė.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Skenavimo metodas | Pradinis intervalas | Šaknis | Tikslumas | Iteracijų skaičius |
| [1.200000000000000; 1.400000000000000] | 1.244837223000000 | 0.000000002000000 | 10 |
| [2.600000000000000; 2.800000000000000] | 2.671334376999999 | 0.000000002000000 | 10 |
| [6.800000000000003; 7.000000000000004] | 6.922074701000001 | 0.000000002000000 | 10 |
| [8.600000000000001; 8.800000000000001] | 8.690729397000002 | 0.000000002000000 | 10 |
| [13.599999999999984; 13.799999999999983] | 13.619990714999982 | 0.000000002000000 | 10 |
| [14.399999999999981; 14.599999999999980] | 14.572906702999978 | 0.000000002000000 | 10 |
| Stygų metodas | Pradinis intervalas | Šaknis | Tikslumas | Iteracijų skaičius |
| [1.200000000000000; 1.400000000000000] | 1.244837223621610 | 0.000000000009593 | 5 |
| [2.600000000000000; 2.800000000000000] | 2.671334377700895 | 0.000000000086623 | 8 |
| [6.800000000000003; 7.000000000000004] | 6.922074700337777 | 0.000000000203741 | 6 |
| [8.600000000000001; 8.800000000000001] | 8.690729395960940 | 0.000000000101704 | 7 |
| [13.599999999999984; 13.799999999999983] | 13.619990716081094 | 0.000000000522431 | 5 |
| [14.399999999999981; 14.599999999999980] | 14.572906703010368 | 0.000000000091756 | 6 |
| Kvazi-Niutono (kirstinių) metodas | Pradiniai artiniai | Šaknis | Tikslumas | Iteracijų skaičius |
| [1.000000000000000; 1.100000000000000] | 1.244837223607163 | 0.000000000000007 | 4 |
| [2.400000000000000; 2.500000000000000] | 2.671334377132789 | 0.000000000582560 | 5 |
| [6.600000000000003; 6.700000000000004] | 6.922074700199420 | 0.000000000000002 | 5 |
| [8.400000000000001; 8.500000000000001] | 8.690729395995058 | 0.000000000043874 | 5 |
| [13.399999999999984; 13.499999999999983] | 13.619990715637094 | 0.000000000005206 | 5 |
| [14.199999999999981; 14.299999999999980] | 14.572906703413357 | 0.000000000406050 | 7 |
| MATLAB funkcijos | Pradinis intervalas | Šaknis (fzero) |  |  |
| 1.200000000000000; | 1.244837223607174 |  |  |
| 2.600000000000000; | 2.671334377800123 |  |  |
| 6.800000000000003; | 6.922074700199421 |  |  |
| 8.600000000000001; | 8.690729396020943 |  |  |
| 13.599999999999984; | 13.619990715641475 |  |  |
| 14.399999999999981; | 14.572906703084644 |  |  |

**Išvados:** Iš rezultatų lentelės (5 lentelė) matome, jog visais metodais rastos tos pačios šaknys, todėl galime teigti, jog metodai veikia teisingai. Iš išbandytų metodų mažiausiai iteracijų atlieka Kvazi-Niutono metodas, tačiau skirtumas nėra toks didelis, koks buvo nagrinėjant polinomą. Kaip ir nagrinėjant daugianarį, rekursinis skenavimo metodas atliko tiek pat iteracijų visuose nagrinėtuose funkcijos šaknų intervaluose. Taip pat verta paminėti ir tai, jog Kvazi-Niutono metodas atlieka ne tik mažiausiai iteracijų, tačiau ir šio metodo tikslumas taip pat lenkia tiek skenavimo, tiek stygų metodus. MATLAB funkcijos fzero taip pat rado tas pačias funkcijos šaknis, todėl galime teigti, kad metodų algoritmai veikia teisingai ir su funkcija f(x).

## Išvados

Šios užduoties metu buvo nagrinėjami 3 netiesinių lygčių šaknų radimo metodai: skenavimo, stygų bei Kvazi-Niutono (kirstinių). Kaip parodė grafikai bei rezultatų lentelės, visi metodai veikia tinkamai, randa tas pačias šaknis, skiriasi tik metodų atliktų iteracijų skaičius bei šaknų tikslumas. Apibendrinus rezultatus galima teigti, kad iš nagrinėtųjų metodų tiksliausias bei mažiausiai iteracijų atliekantis buvo Kvazi-Niutono metodas.

## Programų tekstai

## Daugianario šaknų intervalo įverčių nustatymas

function iverciai\_ir\_grafikai

clc, close all, clear all;

format long;

% ------------------------------------

% Daugianaris f(x)

f = @(x)x.^4+5\*x.^3-2\*x.^2-24\*x;

f\_name = 'x^4+5x^3-2x^2-24x';

% ------------------------------------

% Funkcija g(x)

g = @(x)sin(x).\*log(x)-(x/6);

g\_name = 'sin(x)ln(x)-(x/6)';

% ------------------------------------

% f(x) intervalo nustatymas

% f(x) = x.^4+5\*x.^3-2\*x.^2-24\*x

% n = 4

% a4 = 1

% a3 = 5

% a2 = -2

% a1 = -24

a = [1 5 -2 -24];

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 = 1;

g\_max = 20;

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

## Šaknų intervalų išskyrimas

function saknu\_intervalai

clc, close all, clear all;

format long;

% ------------------------------------

% Daugianaris f(x)

f = @(x)x.^4+5\*x.^3-2\*x.^2-24\*x;

f\_name = 'x^4+5x^3-2x^2-24x';

% ------------------------------------

% Funkcija g(x)

g = @(x)sin(x).\*log(x)-(x/6);

g\_name = 'sin(x)ln(x)-(x/6)';

% ------------------------------------

a = [1 5 -2 -24];

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 = 1;

g\_max = 20;

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

## Šaknų tikslinimo metodai (skenavimo mažėjančiu žingsniu, stygų, Kvazi-Niutono)

function saknu\_tikslinimas

clc, close all, clear all;

format long;

% ------------------------------------

% Daugianaris f(x)

f = @(x)x.^4+5\*x.^3-2\*x.^2-24\*x;

f\_name = 'x^4+5x^3-2x^2-24x';

% ------------------------------------

% Funkcija g(x)

g = @(x)sin(x).\*log(x)-(x/6);

g\_name = 'sin(x)ln(x)-(x/6)';

% ------------------------------------

a = [1 5 -2 -24 0];

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)

% ------------------------------------

disp( sprintf( '----------------------------------------------------'));

disp( sprintf( 'Šaknų tikslinimas skenavimo metodu, mažinant žingsnį'));

disp( sprintf( 'Daugianaris f(x)=x^4+5x^3-2x^2-24x'));

disp( sprintf( '----------------------------------------------------'));

disp( sprintf( 'Stulpelių reikšmės:'));

disp( sprintf( '1:2 - pradiniai šaknų tikslinimo intervalai'));

disp( sprintf( '3 - šaknis'));

disp( sprintf( '4 - tikslumas'));

disp( sprintf( '5 - atliktų iteracijų kiekis'));

disp( sprintf( '----------------------------------------------------'));

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 = 1;

g\_max = 20;

[SaknuIntervalai\_gx]=SkenavimasPastoviu(g\_min, g\_max, zingsnis, g);

disp( sprintf( '----------------------------------------------------'));

disp( sprintf( 'Šaknų tikslinimas skenavimo metodu, mažinant žingsnį'));

disp( sprintf( 'Funkcija g(x)=sin(x)ln(x)-(x/6)'));

disp( sprintf( '----------------------------------------------------'));

disp( sprintf( 'Stulpelių reikšmės:'));

disp( sprintf( '1:2 - pradiniai šaknų tikslinimo intervalai'));

disp( sprintf( '3 - šaknis'));

disp( sprintf( '4 - tikslumas'));

disp( sprintf( '5 - atliktų iteracijų kiekis'));

disp( sprintf( '----------------------------------------------------'));

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

% ------------------------------------

% STYGŲ METODAS

% ------------------------------------

eps = 1e-9;

% šaknų tikslinimas daugianariui f(x)

% ------------------------------------

disp( sprintf( '----------------------------------------------------'));

disp( sprintf( 'Šaknų tikslinimas stygų metodu'));

disp( sprintf( 'Daugianaris f(x)=x^4+5x^3-2x^2-24x'));

disp( sprintf( '----------------------------------------------------'));

disp( sprintf( 'Stulpelių reikšmės:'));

disp( sprintf( '1:2 - pradiniai šaknų tikslinimo intervalai'));

disp( sprintf( '3 - šaknis'));

disp( sprintf( '4 - tikslumas'));

disp( sprintf( '5 - atliktų iteracijų kiekis'));

disp( sprintf( '----------------------------------------------------'));

Tikslumai = [];

Iteracijos = [];

Saknys\_fx = [];

iteracijos\_sk\_max = 200; % maksimalus leistinas iteracijų skaičius

figure(1); grid on; hold on;

for i=1:length(SaknuIntervalai\_fx)

xn = SaknuIntervalai\_fx(i, 1);

xn1 = SaknuIntervalai\_fx(i, 2);

npoints= 1000; x = xn:(xn1-xn)/(npoints - 1):xn1;

iteracijos\_sk = 0;

tikslumas = 1;

while tikslumas > eps

iteracijos\_sk = iteracijos\_sk + 1;

if (iteracijos\_sk > iteracijos\_sk\_max)

fprintf('Virsytas leistinas iteraciju skaicius');

break;

end

% ieškomos k ir xmid reikšmės

fxn = f(xn);

fxn1 = f(xn1);

k=abs(fxn/fxn1);

xmid=(xn+k\*xn1)/(1+k);

%----------------------------------------------------

% vizualizacija

%----------------------------------------------------

if (i == 1 && iteracijos\_sk < 7)

plot(x, f(x), 'r-');

plot([xn xn1], [0 0], 'b-');

plot(xn, 0, 'mp', 'Linewidth', 2); h = findobj(gca,'Type','line');h1=h(1);

plot(xn1,0,'cp', 'Linewidth', 2);h = findobj(gca,'Type','line');h2=h(1);

plot(xmid,0,'gs', 'Linewidth', 2);plot([xn,xn1],[fxn,fxn1],'g-');h = findobj(gca,'Type','line');h3=h(1:2);

input('Press Enter'), figure(1);

delete(h1);delete(h2);delete(h3);

end

%----------------------------------------------------

fxmid=f(xmid);

if (sign(fxmid) == sign(fxn))

xn=xmid;

else

xn1=xmid;

end

tikslumas = abs(fxmid);

end

Iteracijos = [Iteracijos; iteracijos\_sk];

Tikslumai = [Tikslumai; tikslumas];

Saknys\_fx = [Saknys\_fx; xmid];

end

close all;

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)

% ------------------------------------

disp( sprintf( '----------------------------------------------------'));

disp( sprintf( 'Šaknų tikslinimas stygų metodu'));

disp( sprintf( 'Funkcija g(x)=sin(x)ln(x)-(x/6)'));

disp( sprintf( '----------------------------------------------------'));

disp( sprintf( 'Stulpelių reikšmės:'));

disp( sprintf( '1:2 - pradiniai šaknų tikslinimo intervalai'));

disp( sprintf( '3 - šaknis'));

disp( sprintf( '4 - tikslumas'));

disp( sprintf( '5 - atliktų iteracijų kiekis'));

disp( sprintf( '----------------------------------------------------'));

Tikslumai = [];

Iteracijos = [];

Saknys\_gx = [];

iteracijos\_sk\_max = 200; % maksimalus leistinas iteracijų skaičius

for i=1:length(SaknuIntervalai\_gx)

xn = SaknuIntervalai\_gx(i, 1);

xn1 = SaknuIntervalai\_gx(i, 2);

iteracijos\_sk = 0;

tikslumas = 1;

while tikslumas > eps

iteracijos\_sk = iteracijos\_sk + 1;

if (iteracijos\_sk > iteracijos\_sk\_max)

fprintf('Virsytas leistinas iteraciju skaicius');

break;

end

% ieškomos k ir xmid reikšmės

fxn = g(xn);

fxn1 = g(xn1);

k=abs(fxn/fxn1);

xmid=(xn+k\*xn1)/(1+k);

fxmid=g(xmid);

if (sign(fxmid) == sign(fxn))

xn=xmid;

else

xn1=xmid;

end

tikslumas = abs(fxmid);

end

Iteracijos = [Iteracijos; iteracijos\_sk];

Tikslumai = [Tikslumai; tikslumas];

Saknys\_gx = [Saknys\_gx; xmid];

end

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)

% ------------------------------------

disp( sprintf( '----------------------------------------------------'));

disp( sprintf( 'Šaknų tikslinimas Kvazi-Niutono (kirstinių) metodu'));

disp( sprintf( 'Daugianaris f(x)=x^4+5x^3-2x^2-24x'));

disp( sprintf( '----------------------------------------------------'));

disp( sprintf( 'Stulpelių reikšmės:'));

disp( sprintf( '1 - pirmasis pradinis artinys'));

disp( sprintf( '2 - antrasis pradinis artinys'));

disp( sprintf( '3 - šaknis'));

disp( sprintf( '4 - tikslumas'));

disp( sprintf( '5 - atliktų iteracijų kiekis'));

disp( sprintf( '----------------------------------------------------'));

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)

% ------------------------------------

disp( sprintf( '----------------------------------------------------'));

disp( sprintf( 'Šaknų tikslinimas Kvazi-Niutono (kirstinių) metodu'));

disp( sprintf( 'Funkcija g(x)=sin(x)ln(x)-(x/6)'));

disp( sprintf( '----------------------------------------------------'));

disp( sprintf( 'Stulpelių reikšmės:'));

disp( sprintf( '1 - pirmasis pradinis artinys'));

disp( sprintf( '2 - antrasis pradinis artinys'));

disp( sprintf( '3 - šaknis'));

disp( sprintf( '4 - tikslumas'));

disp( sprintf( '5 - atliktų iteracijų kiekis'));

disp( sprintf( '----------------------------------------------------'));

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)

disp( sprintf( '----------------------------------------------------'));

disp( sprintf( 'MATLAB funkcijos'));

disp( sprintf( '----------------------------------------------------'));

disp( sprintf( 'Daugianaris f(x)'));

disp( sprintf( '----------------------------------------------------'));

a = [1 5 -2 -24 0];

saknys\_roots = roots(a)

for i=1:length(SaknuIntervalai\_fx)

fzero(f, SaknuIntervalai\_fx(i, 1))

end

% Matlab funkcijos

% Funkcija g(x)

disp( sprintf( '----------------------------------------------------'));

disp( sprintf( 'Funkcija g(x)'));

disp( sprintf( '----------------------------------------------------'));

for i=1:length(SaknuIntervalai\_gx)

fzero(g, SaknuIntervalai\_gx(i, 1))

end

end

function [a, b, i, tikslumas]=SkenavimasRekursija(xmin, xmax, zingsnis, tol, f, iteracijos\_sk, draw)

x = xmin;

iteracijos\_sk = iteracijos\_sk + 1;

while x < xmax

if zingsnis < tol

a = xmin;

b = xmax;

i = iteracijos\_sk;

tikslumas = zingsnis\*10;

return;

end

if (sign(f(x))~=sign(f(x+zingsnis)))

if (draw == 1 && iteracijos\_sk < 7)

plot(xmin, 0, 'go', 'LineWidth', 2); h = findobj(gca,'Type','line');h1=h(1);

plot(xmax, 0, 'go', 'LineWidth', 2); h = findobj(gca,'Type','line');h2=h(1);

input('Press Enter'), figure(1);

delete(h1); delete(h2);

end

[a,b,i,tikslumas]=SkenavimasRekursija(x, x+zingsnis, zingsnis/10, tol, f, iteracijos\_sk, draw);

end

x=x+zingsnis;

end

end