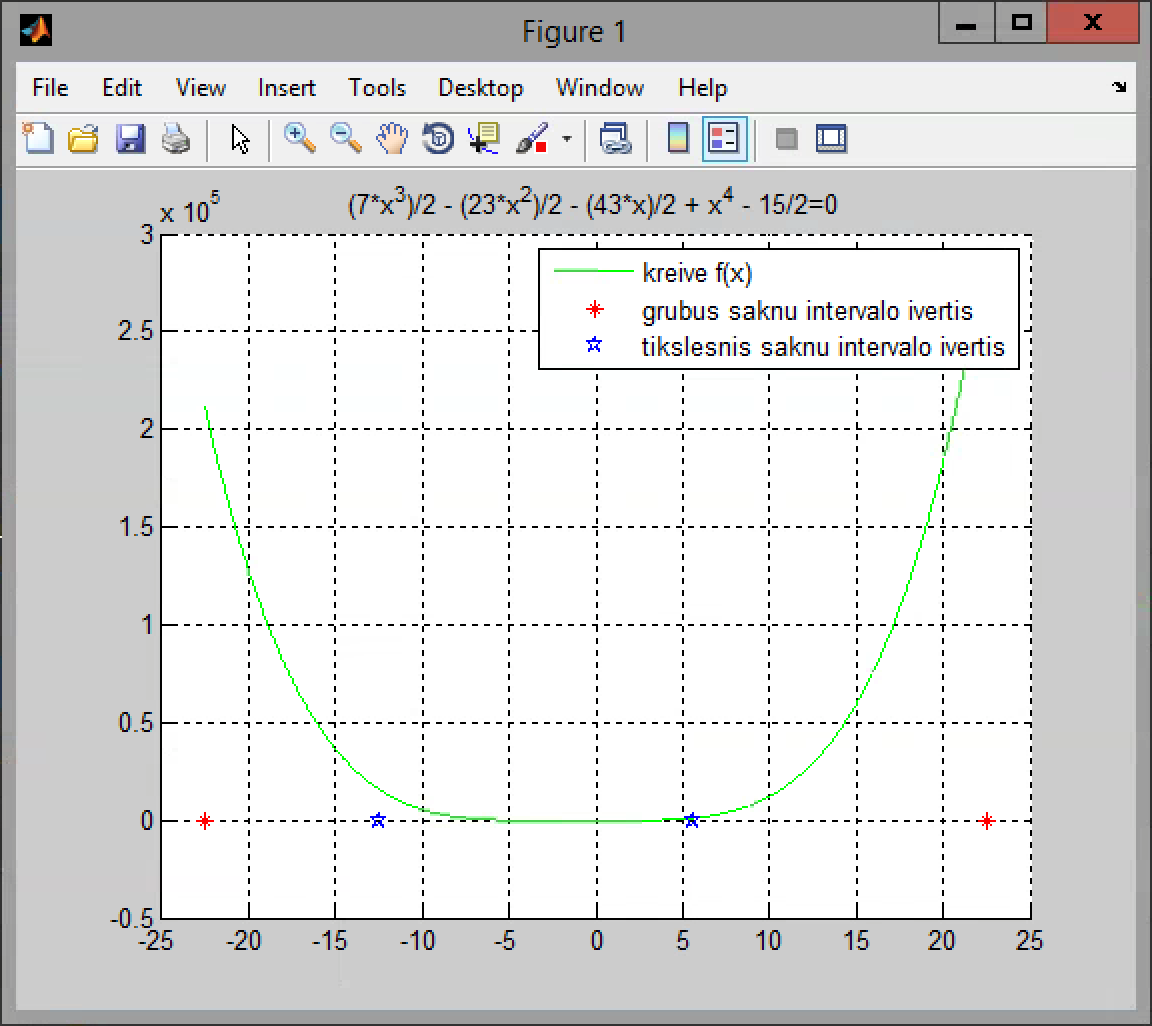
# Netiesinių lygčių sprendimas

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

|  |  |  |
| --- | --- | --- |
| Nr. | Daugianaris f(x) | Funkcija g(x) |
| 1 | 𝑥4 + 7/2𝑥3 – 23/2𝑥2 – 43/2𝑥 -15/2 | 1,9𝑥 sin(𝑥) − (𝑥/1,5-3)2; -10 <= x <= 10 |
| Sprendimo metodai: skenavimo, stygų ir Kvazi-Niutono (kirstinių) | | |

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

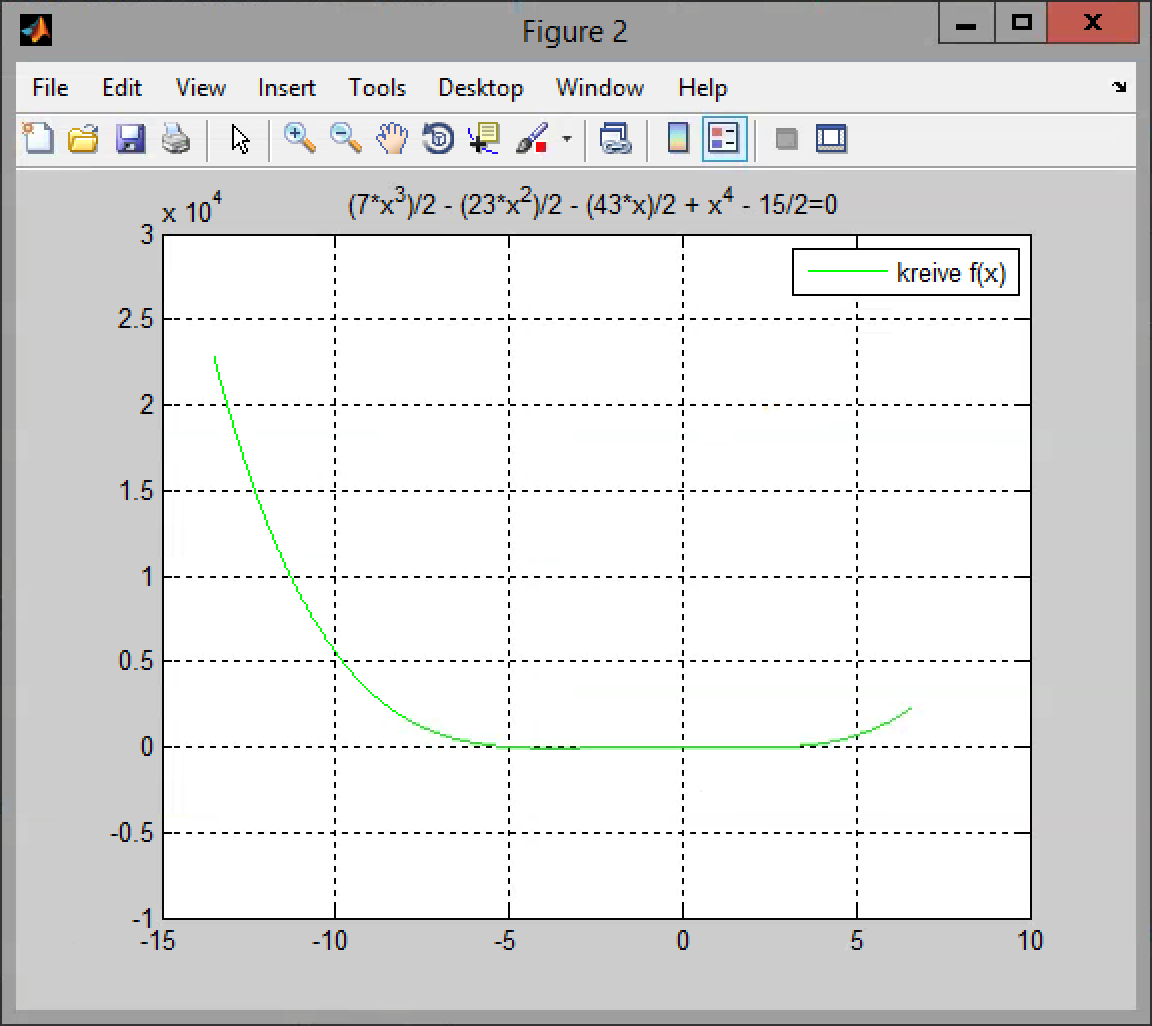
* Daugianario šaknų intervalo įverčiai



Figūra 1 Daugianrio šaknų intervalo įverčiai (grafinis vaizdas)

|  |  |
| --- | --- |
| Grubus lygties f(x) = 0 šaknų intervalo įvertis | [-22.5;22.5] |
| Tikslesnis lygties f(x) = 0 šaknų intervalo įvertis | [-12.5; 5.6368] |

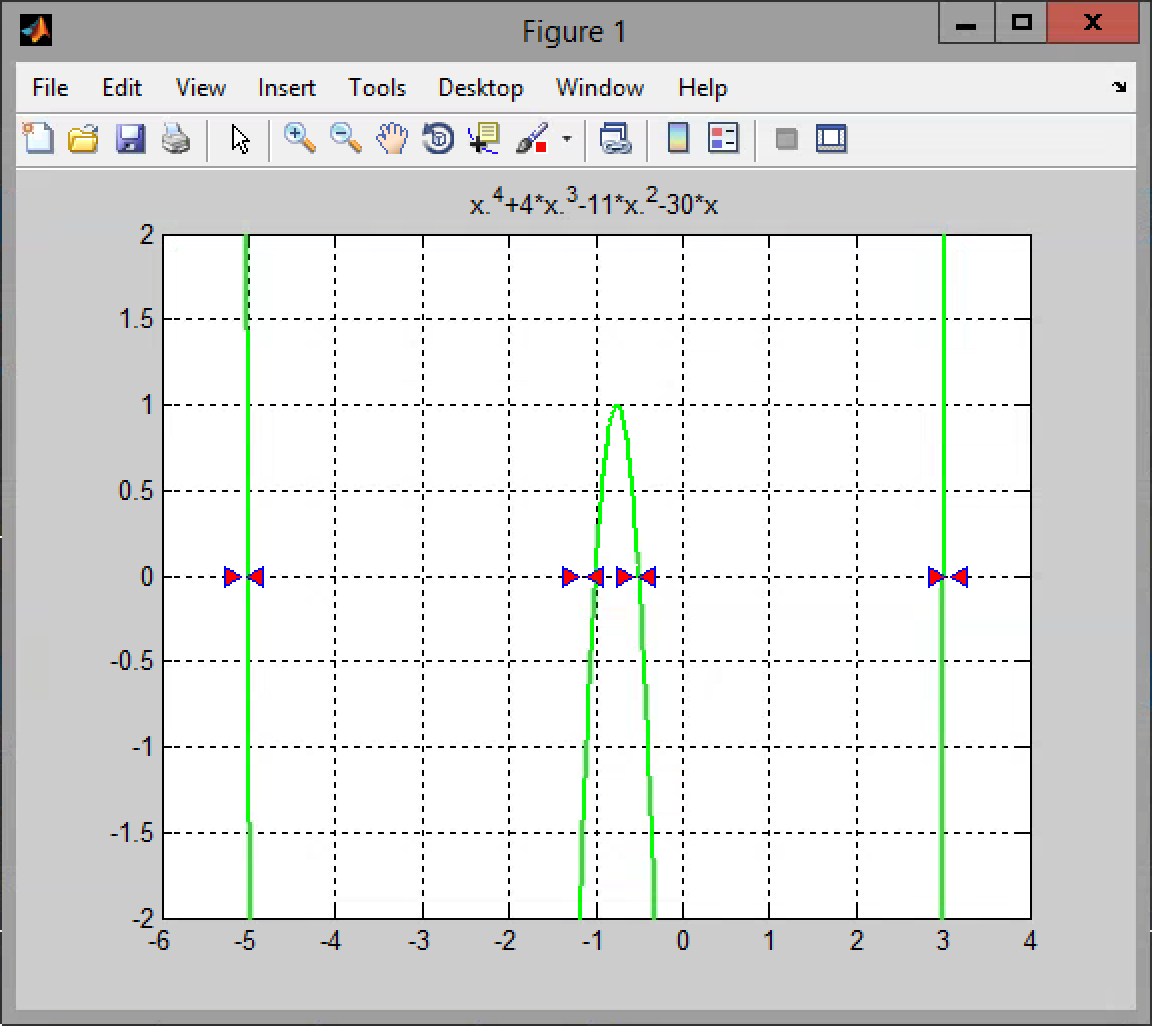
* Daugianario grafinis vaizdas nustatytame intervale



Figūra 2 Daugianario grafinis vaizdas nustatytame interval

* Šaknų atskyrimas skenavimo metodu

Skenavimas atliekamas intervale [-5.1; 3.1], skenavimo žingsnis lygus 0.14



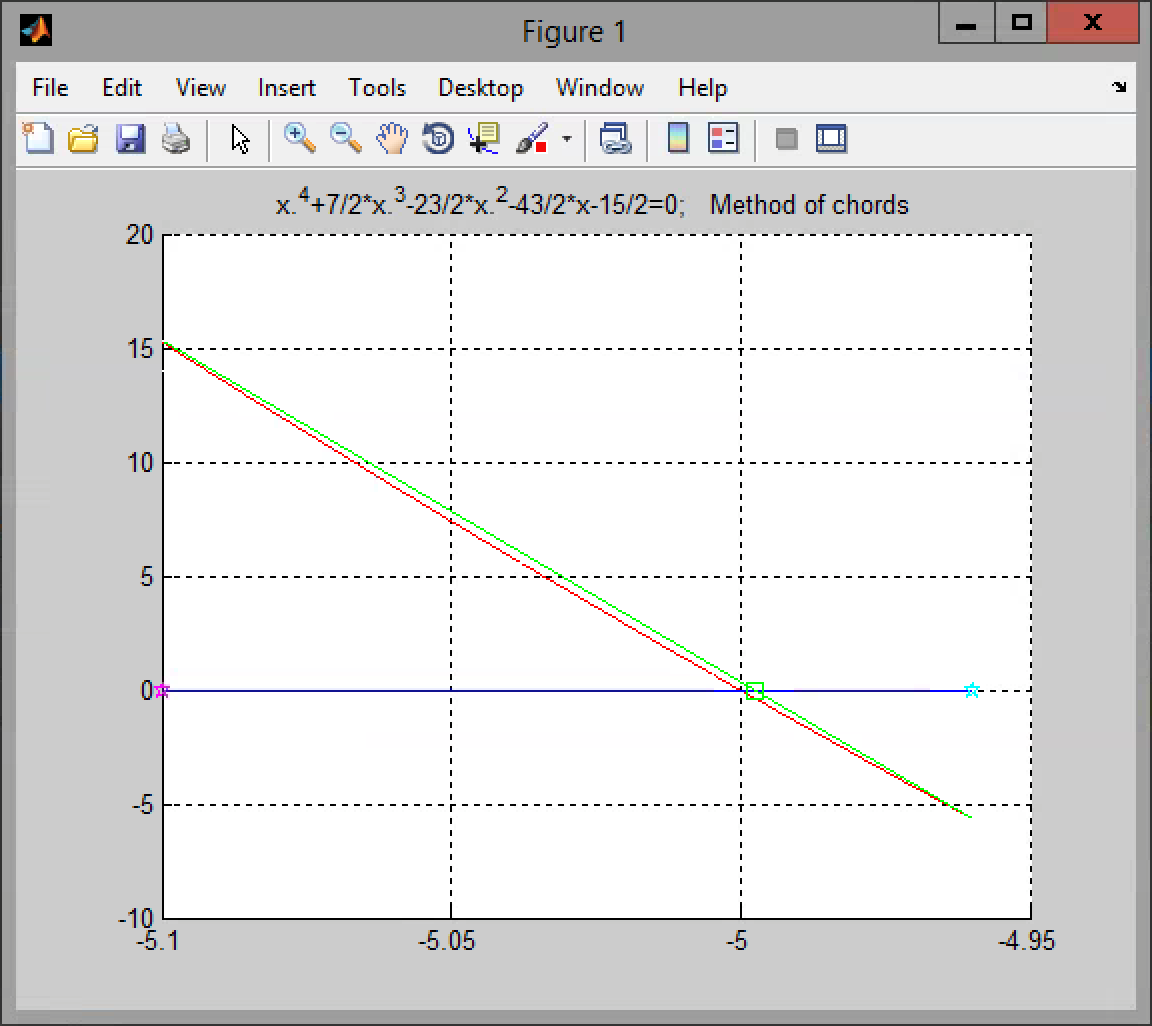
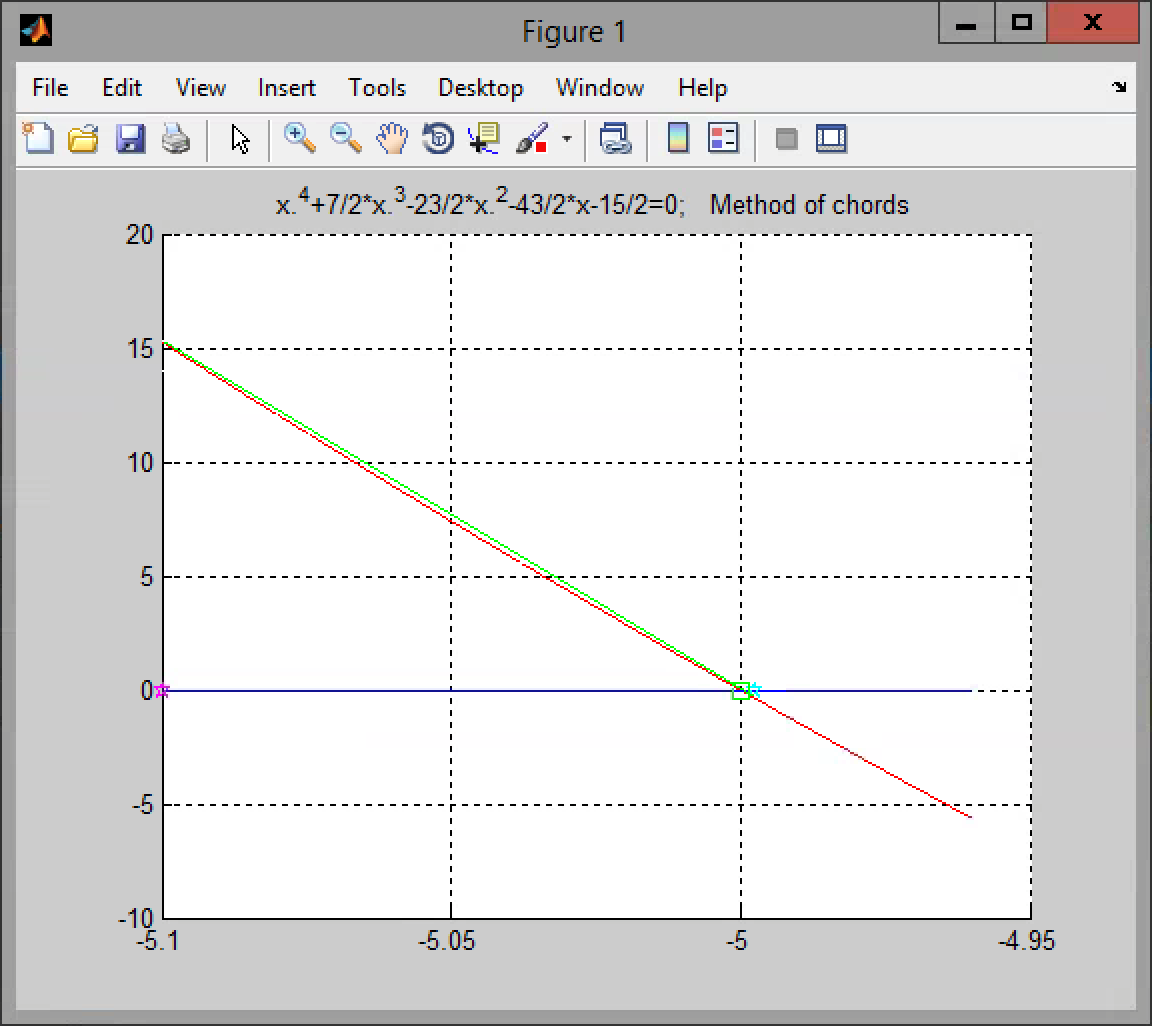
Figūra 3 Daugianario šaknų atskyrimo intervalai

|  |  |
| --- | --- |
| Intervalo Nr. | Intervalas |
| 1 | [-5.10 ; -4.96 ] |
| 2 | [-1.04 ; -0.90 ] |
| 3 | [-0.62 ; -0.48 ] |
| 4 | [2.88 ; 3.02 ] |

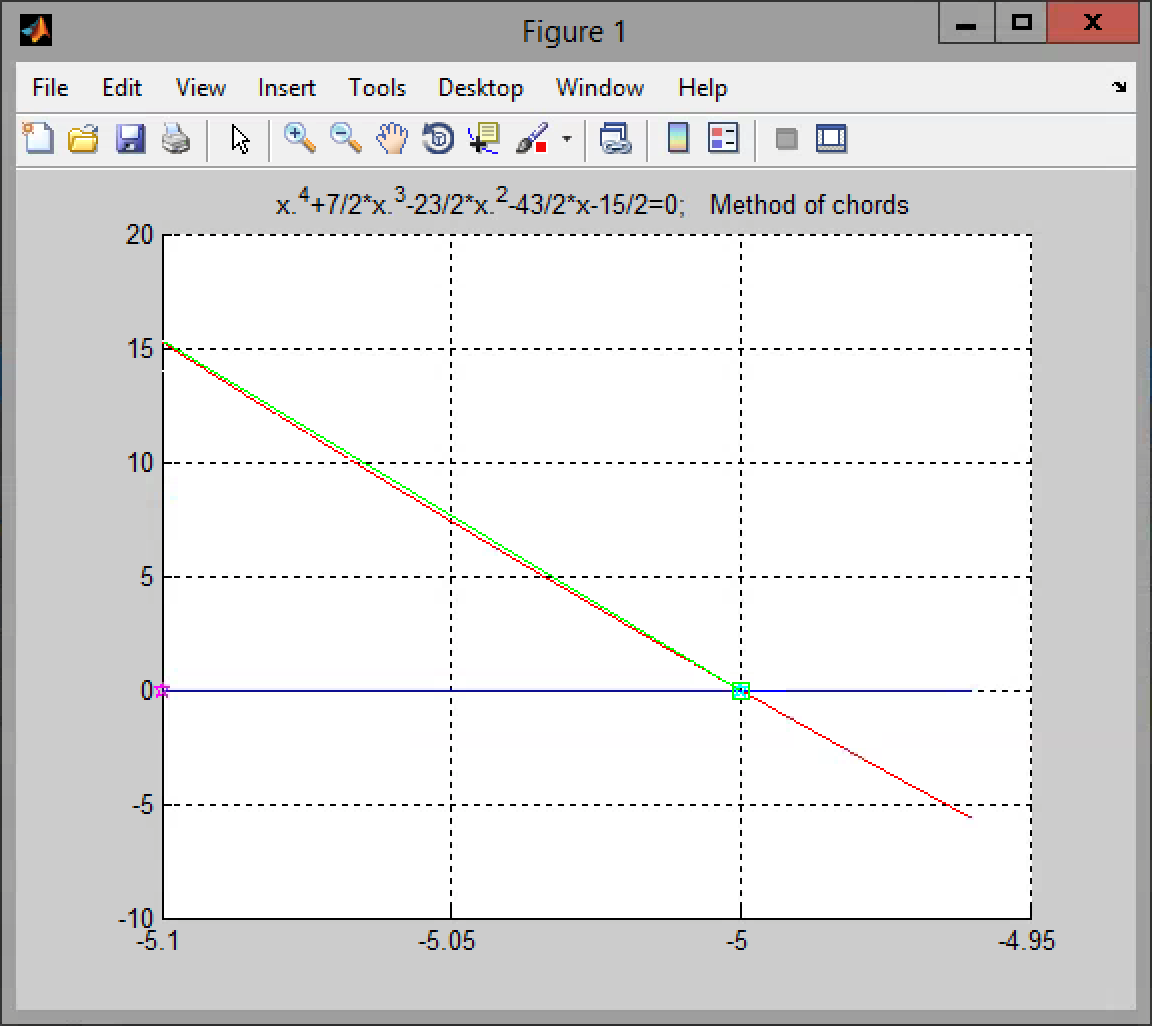
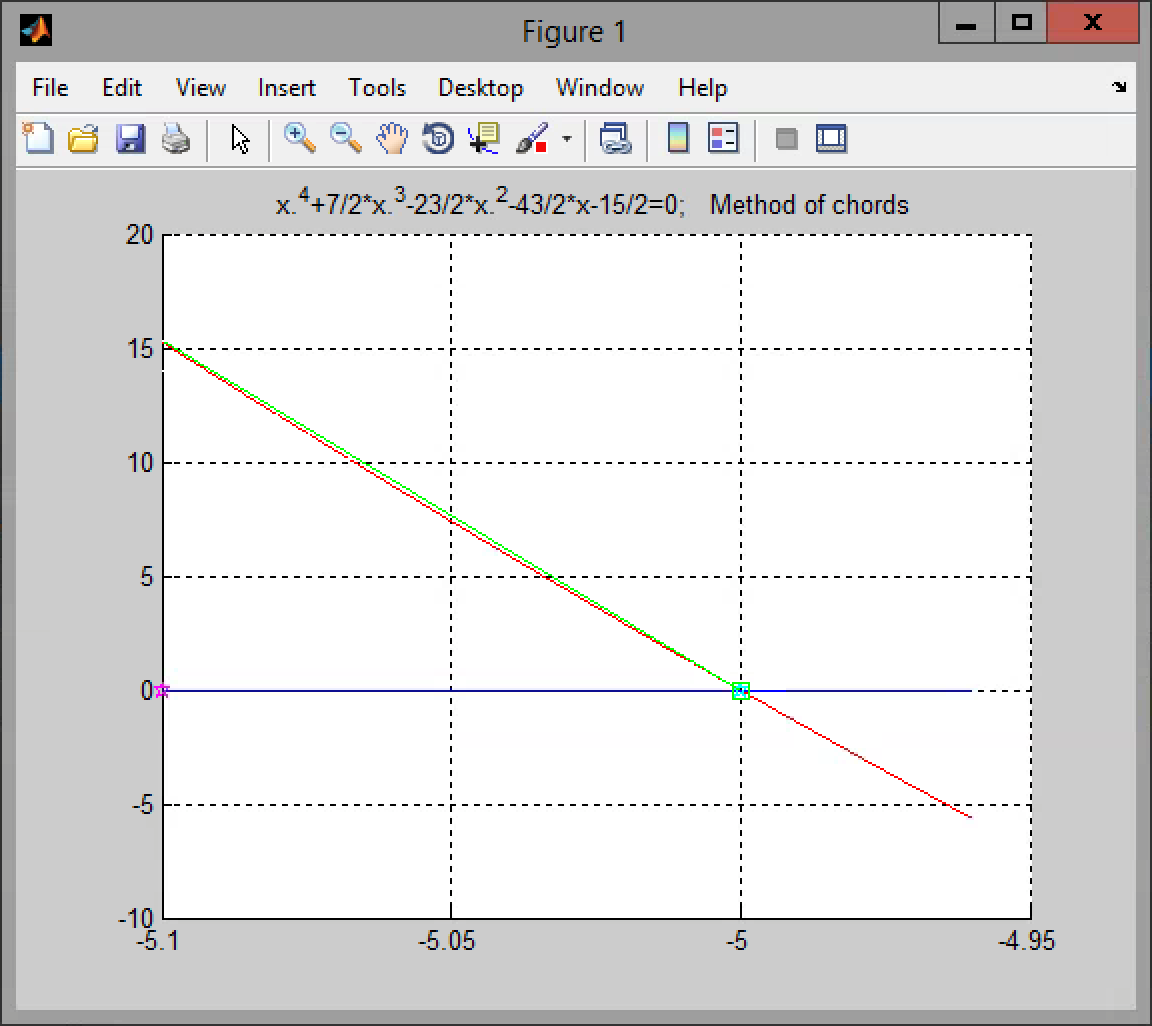
* Šaknų tikslinimas skenavimo, stygų ir Kvazi-Niutono (kirstinių) metodais

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

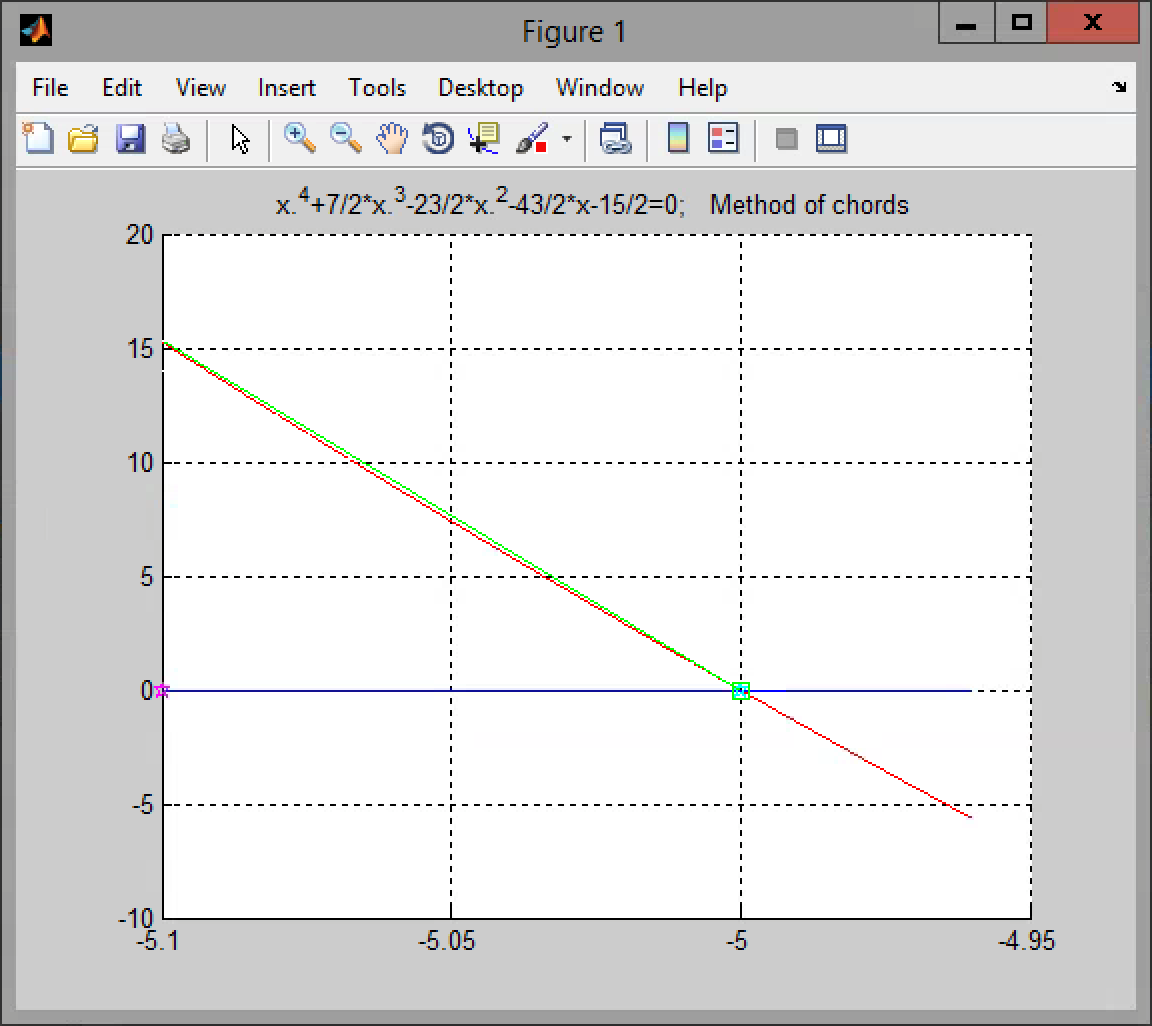
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Stygų | Pradinis intervalas | Šaknis | Tikslumas | Iteracijų sk. |
| [-5.10 ; -4.96 ] | -5.0000000000000000 | 0.000000000688317 | 8 |
| [-1.04 ; -0.90 ] | -1.0000000000000000 | 0.00000000090048 | 8 |
| [-0.62 ; -0.48 ] | -0.5000000000000000 | 0.00000000012345 | 7 |
| [2.88 ; 3.02 ] | 3.0000000000000000 | 0.00000000007.16085 | 6 |
| Skenavimo | Pradinis intervalas | Šaknis | Tikslumas | Iteracijų sk. |
| [-5.10 ; -4.96 ] | -5.0000000000000000 | 0.0000000006704113 | 48 |
| [-1.04 ; -0.90 ] | -1.0000000000000000 | 0.0000000002980212 | 30 |
| [-0.62 ; -0.48 ] | -0.5000000000000000 | 0.0000000001466818 | 45 |
| [2.88 ; 3.02 ] | 3.0000000000000000 | 0.000000000260641 | 51 |
| Kvazi-Niutono | Pradinis intervalas | Šaknis | Tikslumas | Iteracijų sk. |
| [-5.10 ; -4.96 ] | -5.0000000000000000 | 0.00000000000012789 | 5 |
| [-1.04 ; -0.90 ] | -1.0000000000000000 | 0.000000000000149214 | 6 |
| [-0.62 ; -0.48 ] | -0.5000000000000000 | 0.0000000000022939 | 5 |
| [2.88 ; 3.02 ] | 3.0000000000000000 | 0.000000000572555 | 5 |
| MATLAB funkcijos | Pradinis artinys | (Fzero) | (Roots) |  |
| -5.0 | -5.00000000000000 | -5.00000000000000 |  |
| 3.0 | -3.00000000000000 | -3.00000000000000 |  |
| -1.0 | 1.00000000000000 | 1.00000000000000 |  |
| -0.5 | 0.50000000000000 | 0.50000000000000 |  |

1 iteracija 2 iteracija

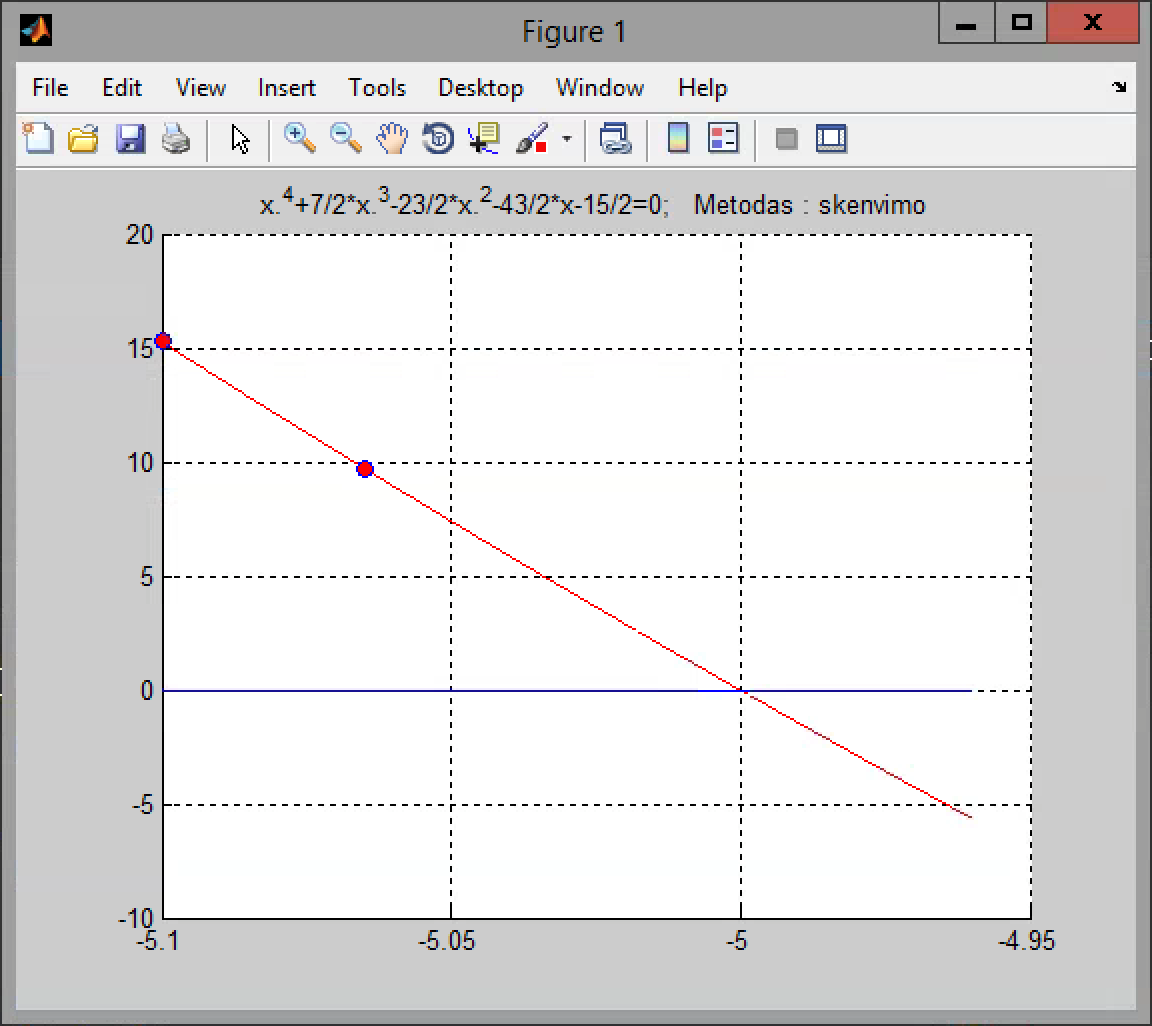
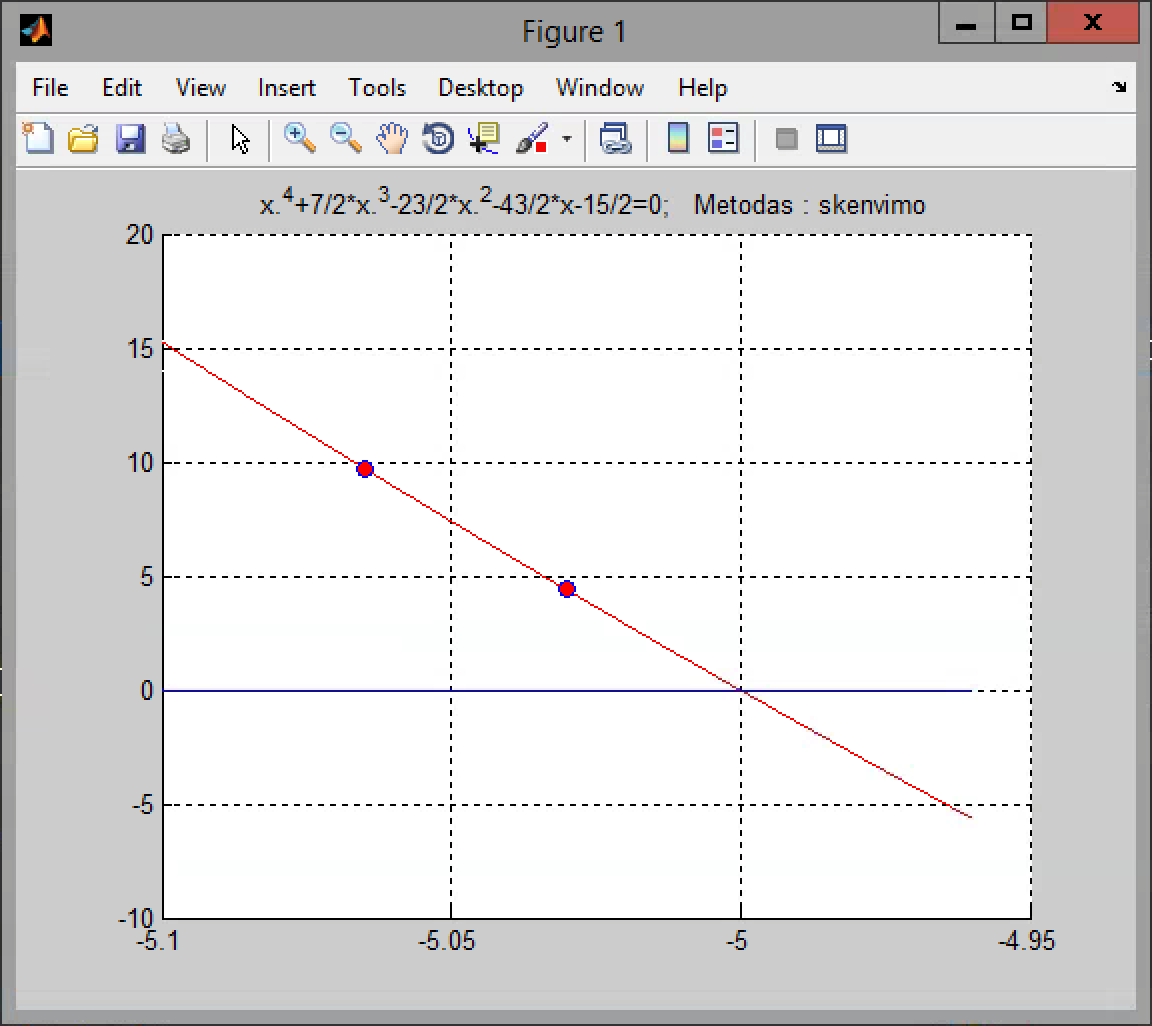
 

3 iteracija 4 iteracija

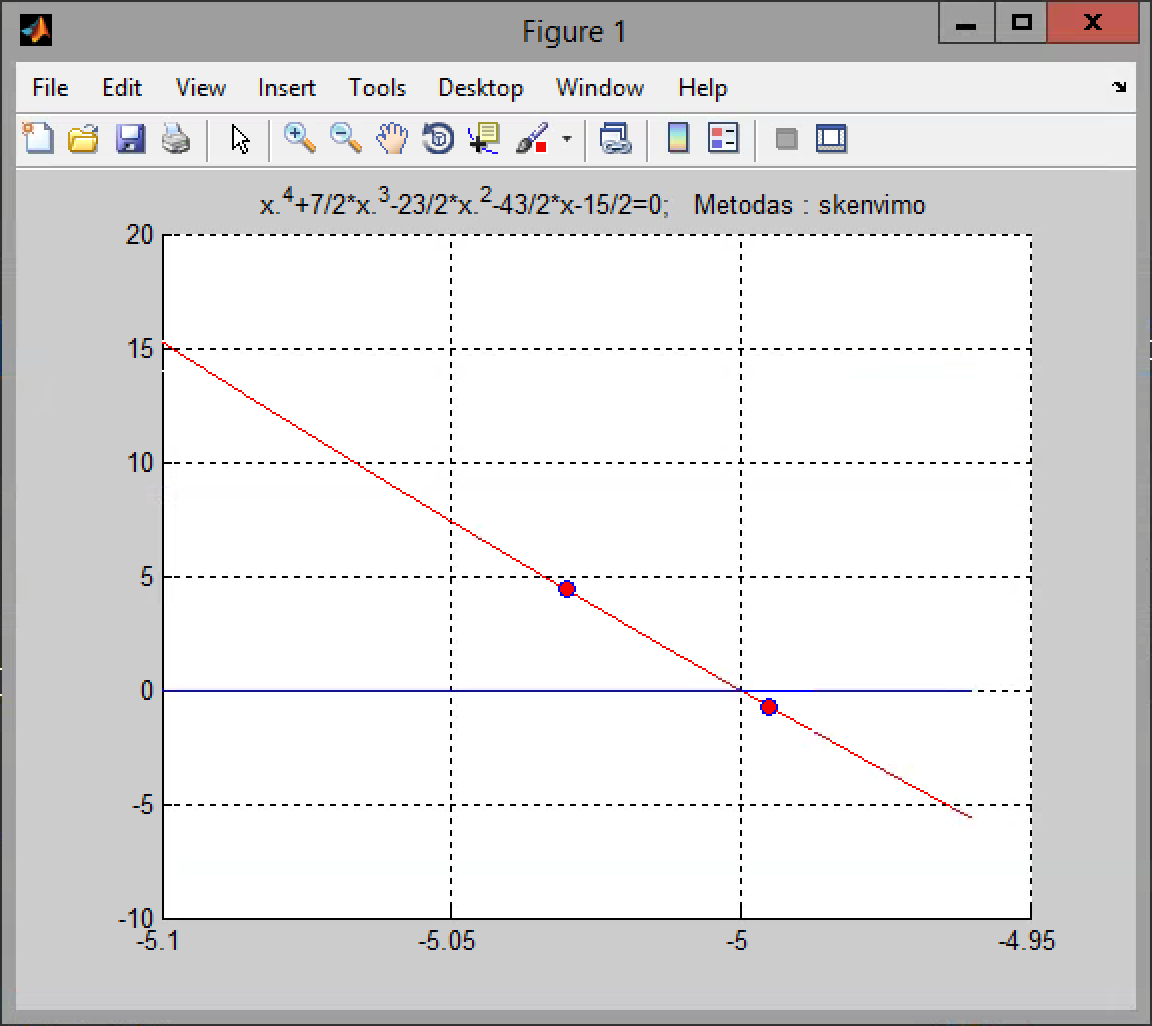
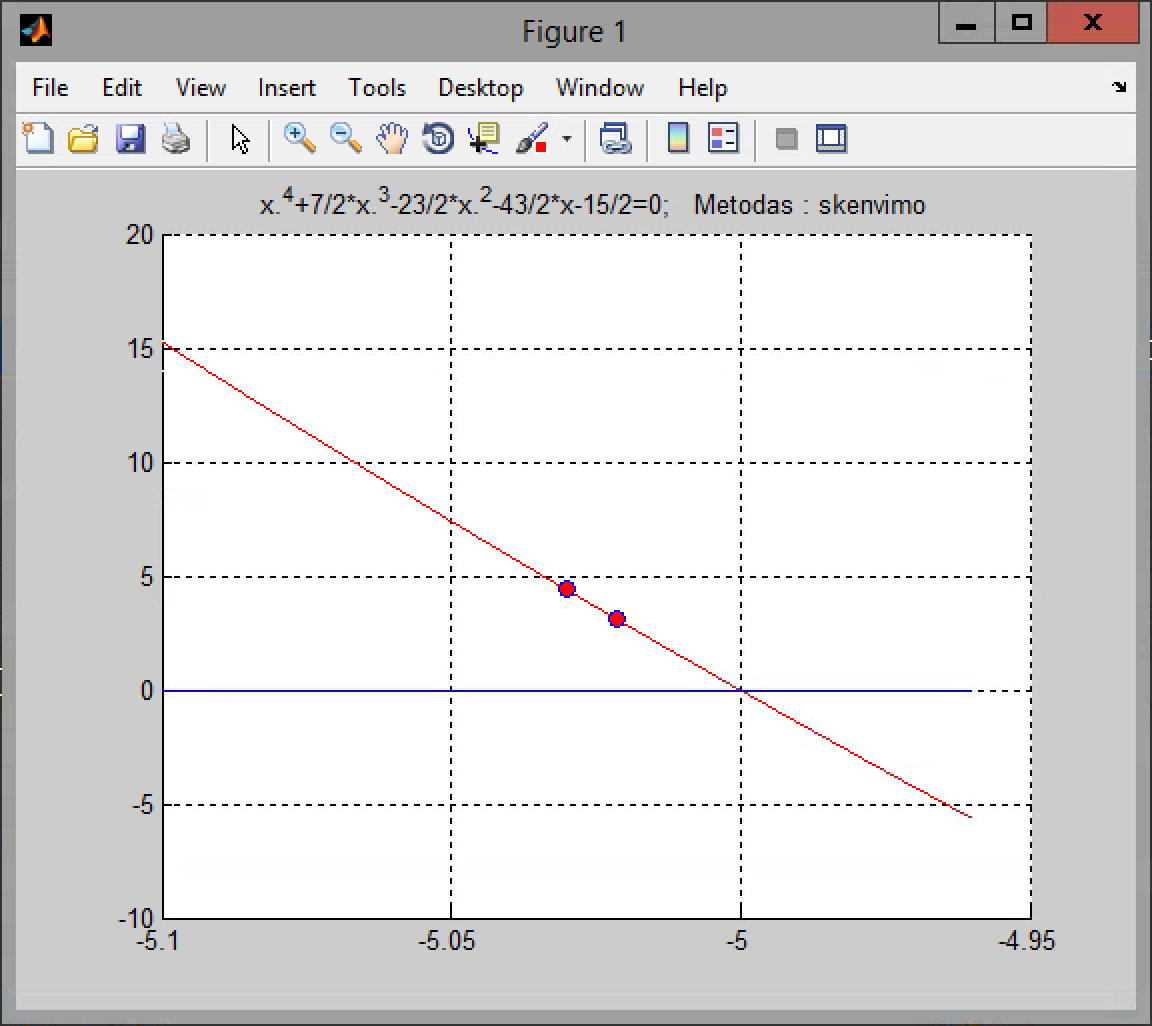


5 iteracija

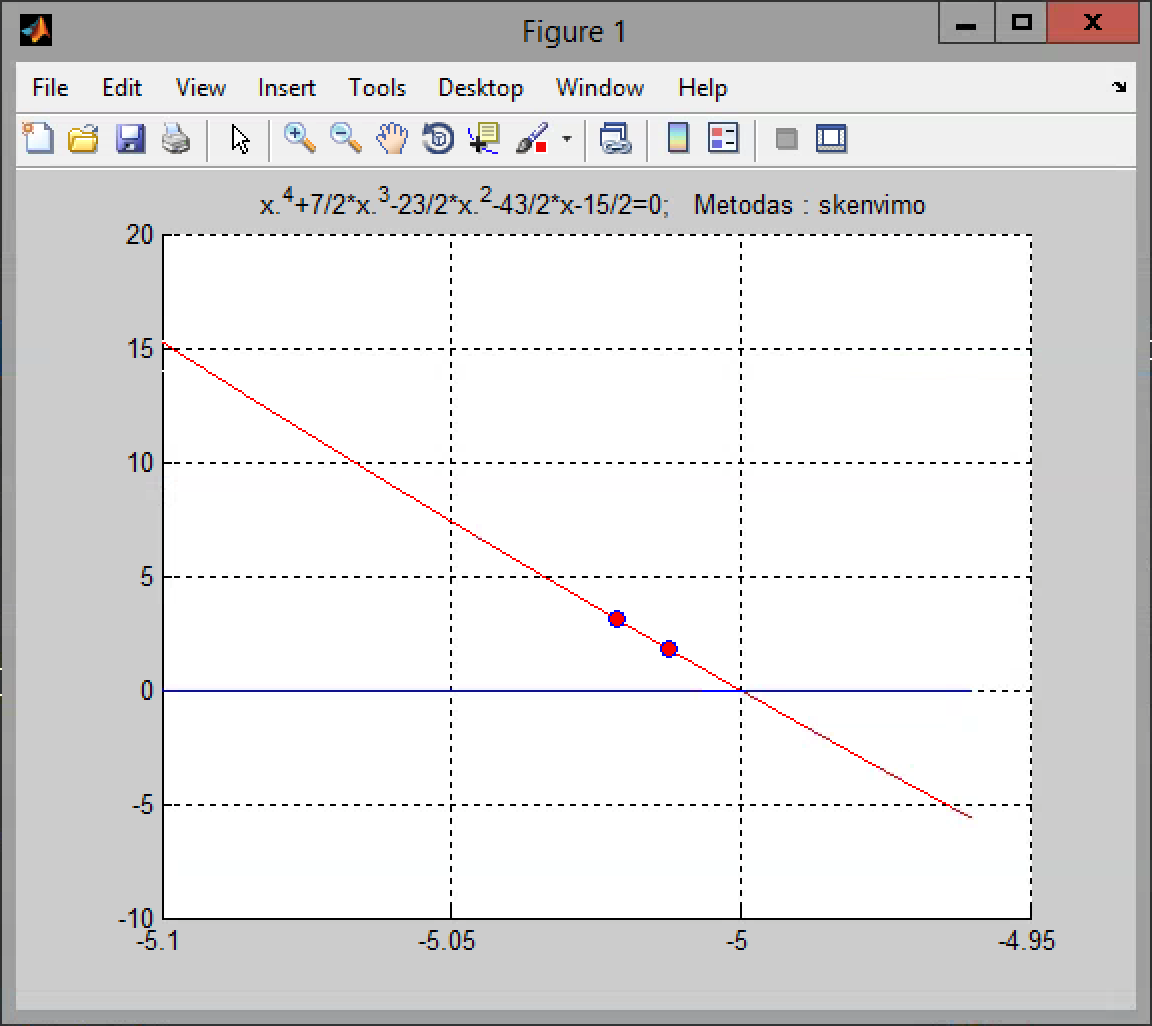
Šaknies xg = -5 tikslinimo **stygų** metodu vizualizacija. Raudona linija brėžiama funkcija, žalia linija - pagalbinė, taškais žymimi iteracijoje nagrinėjamo intervalo galai.

1 iteracija 2 iteracija

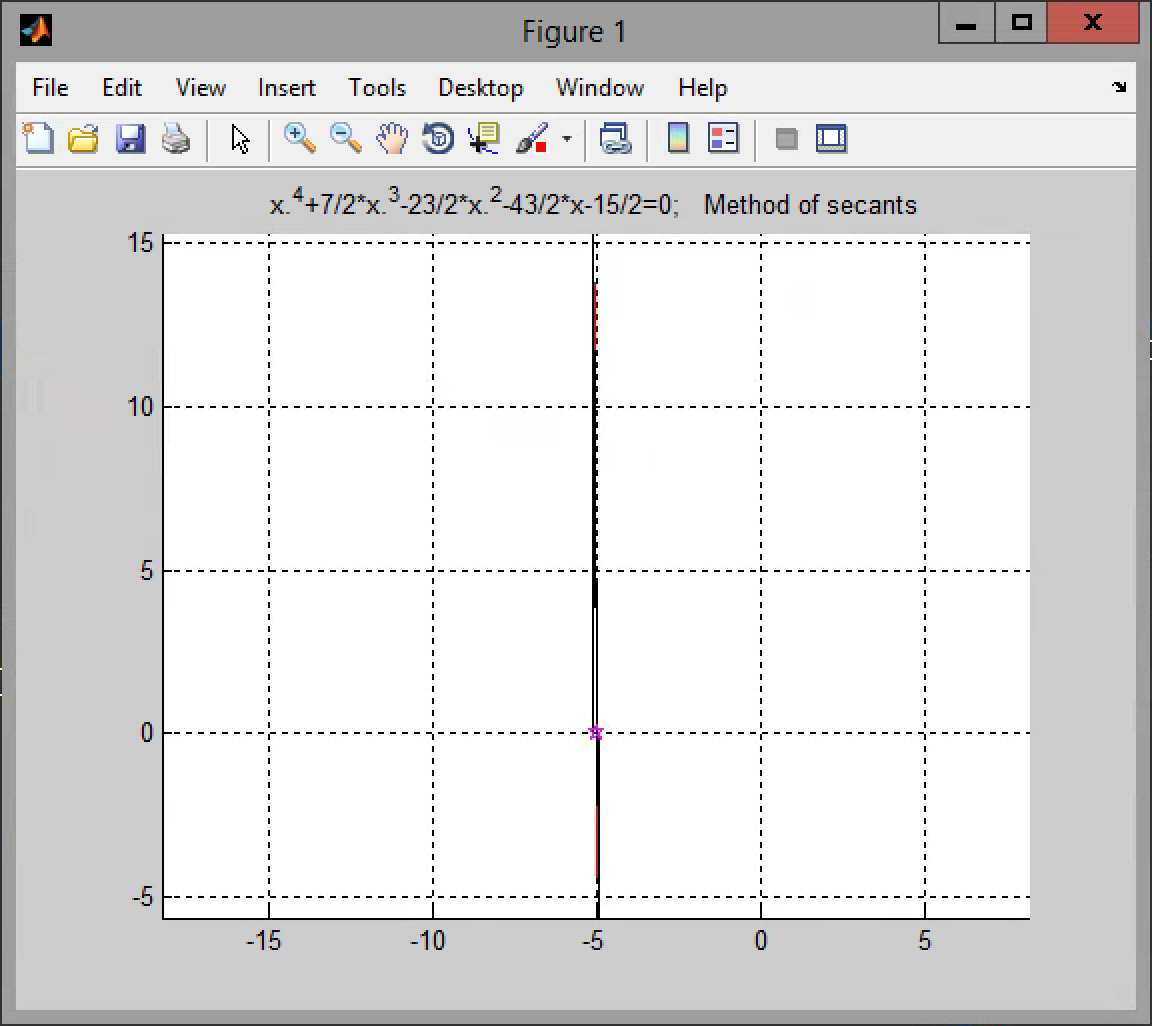
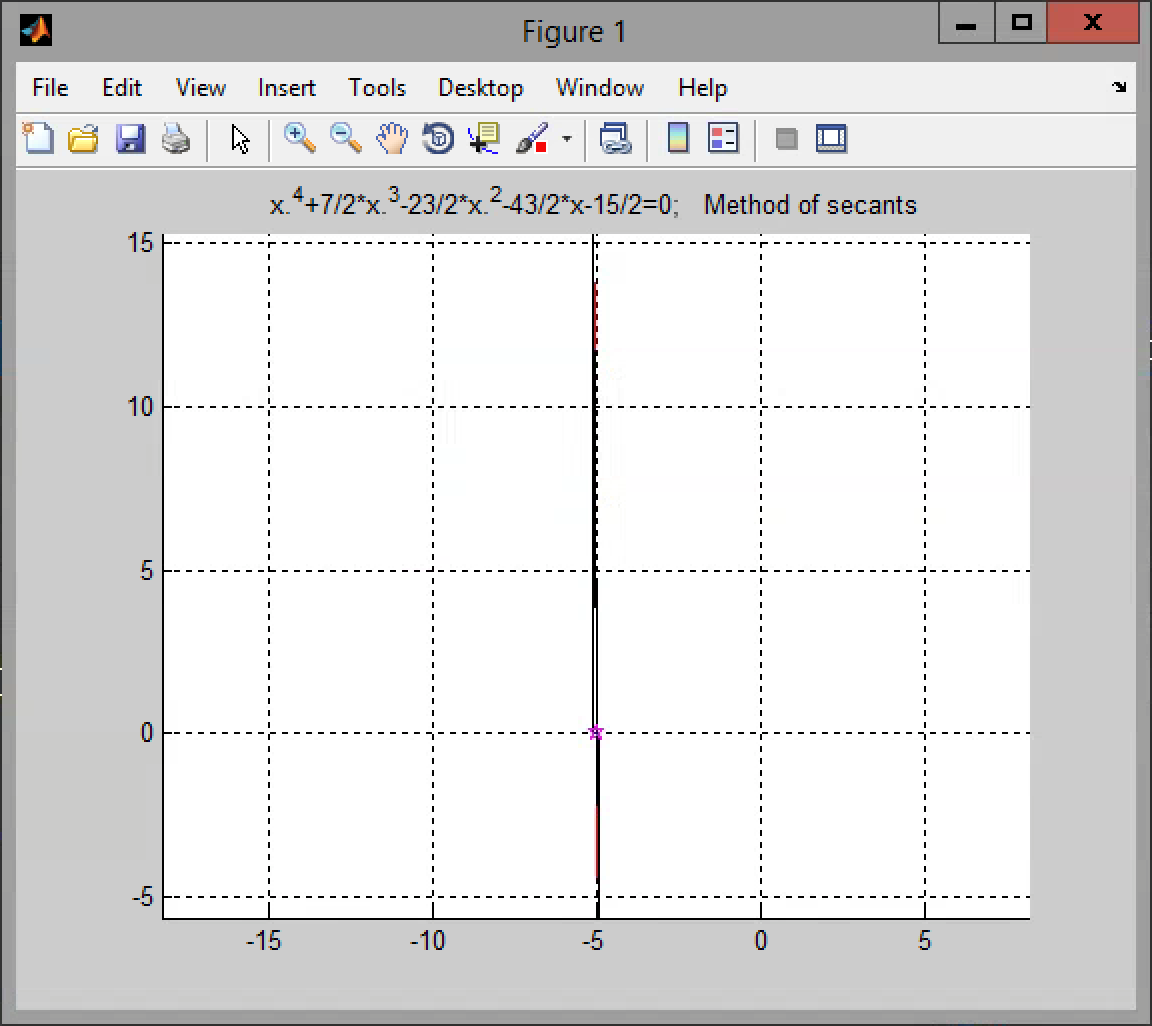
 

3 iteracija 4 iteracija

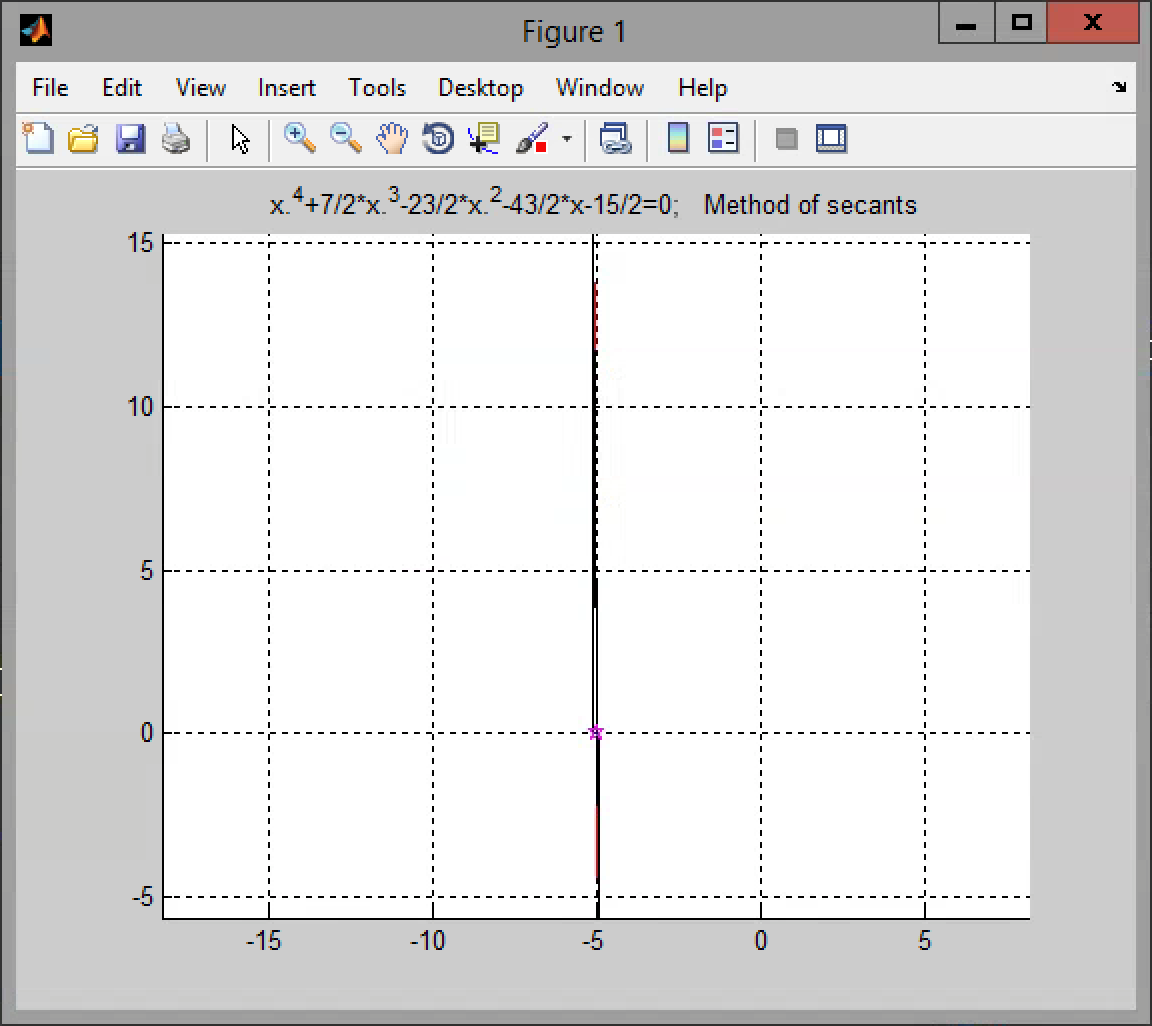
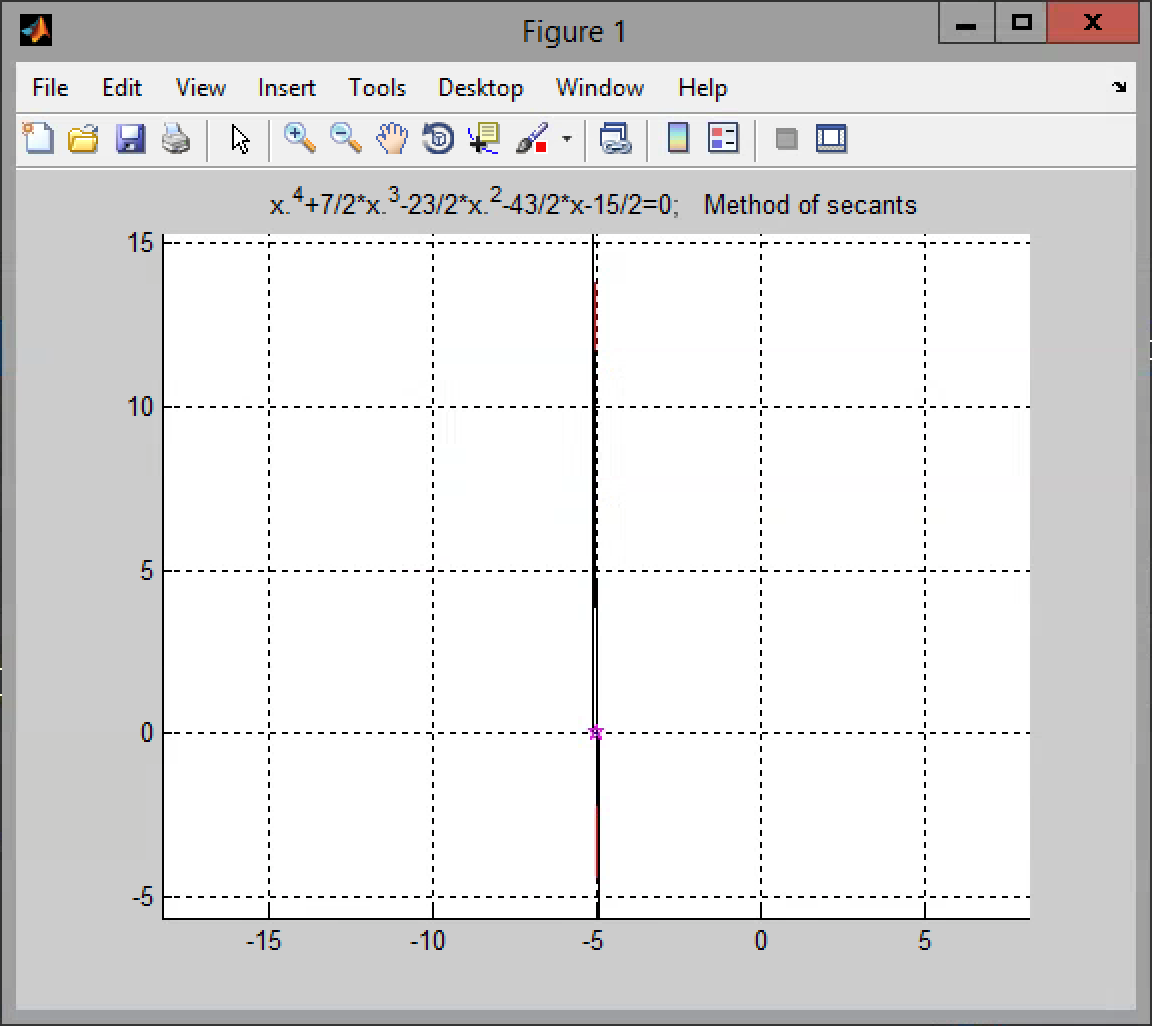


5 iteracija

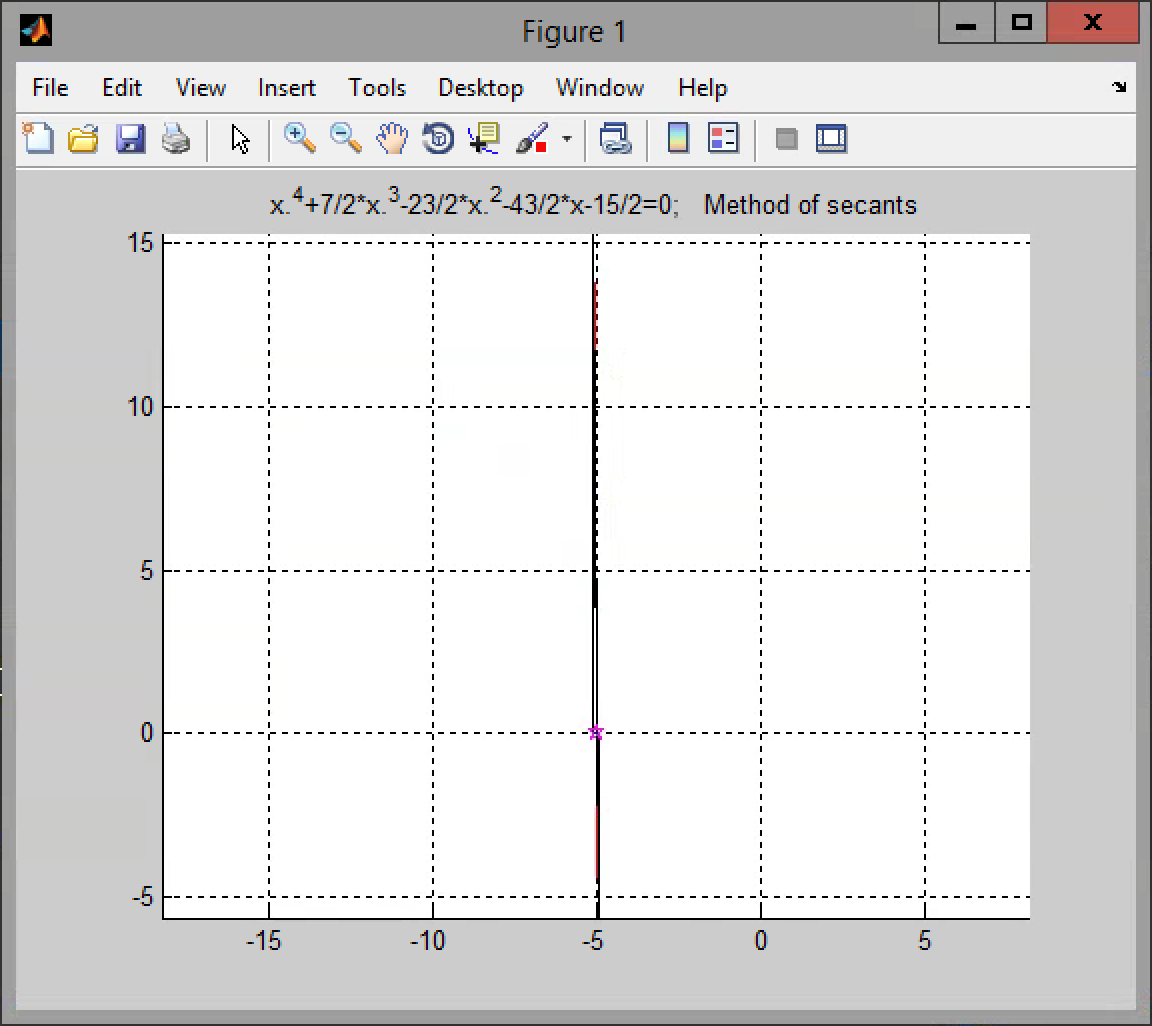
Šaknies xg = -5 tikslinimo **skenavimo** metodu vizualizacija. Linija brėžiama funkcija, taškais žymimi interacijoje nagrinėjamo intervalo galai.

1 iteracija 2 iteracija

3 iteracija 4 iteracija



5 iteracija

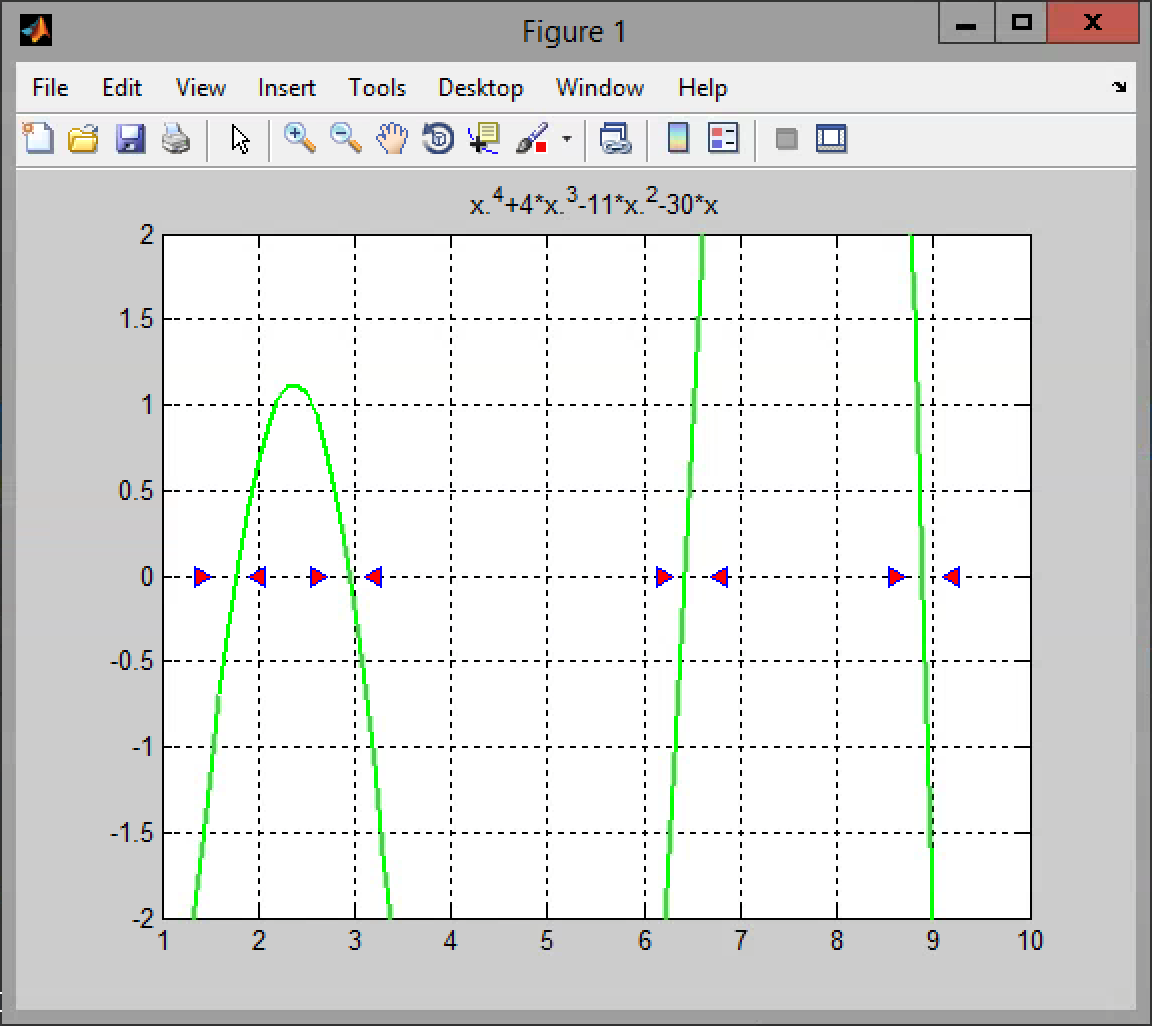
Šaknies xg = -5 tikslinimo **kirstinių** metodu vizualizacija. Raudona linija brėžiama funkcija, kitos – pagalbinės linijos.

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

* Šaknų atskyrimas skenavimo metodu

Skenavimas atliekamas intervale [-10; 10], skenavimo žingsnis lygus 0.6

|  |  |
| --- | --- |
| Intervalo Nr. | Intervalas |
| 1 | [1.40 ; 2.00 ] |
| 2 | [2.60 ; 3.20 ] |
| 3 | [6.20 ; 6.80 ] |
| 4 | [8.60 ; 9.20 ] |



Figūra 4 Trancendentinės funkcijos šaknų atskyrimo intervalai

* Šaknų tikslinimas skenavimo, stygų ir Kvazi-Niutono (kirstinių) metodais

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

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Stygų | Pradinis intervalas | Šaknis | Tikslumas | Iteracijų sk. |
| [1.40 ; 2.00 ] | 1.77440000000000 | 0.000000000267121 | 13 |
| [2.60 ; 3.20 ] | 2.9498800000000 | 0.0000000000245887 | 12 |
| [6.20 ; 6.80 ] | 6.4176200000000 | 0.0000000000000159872 | 4 |
|  | [8.60 ; 9.20 ] | 8.8924700000000 | 0.000000000255381 | 10 |
| Skenavimo | Pradinis intervalas | Šaknis | Tikslumas | Iteracijų sk. |
| [1.40 ; 2.00 ] | 1.77440000000000 | 0.0000000006535918 | 48 |
| [2.60 ; 3.20 ] | 2.9499000000000 | 0.000000000177563 | 30 |
| [6.20 ; 6.80 ] | 6.4176000000000 | 0.000000000341897 | 35 |
| [8.60 ; 9.20 ] | 8.8925000000000 | 0.0000000006666667 | 35 |
| Kvazi-Niutono | Pradinis intervalas | Šaknis | Tikslumas | Iteracijų sk. |
| [1.40 ; 2.00 ] | 1.77440000000000 | 0.00000000000003996 | 6 |
| [2.60 ; 3.20 ] | 2.9498800000000 | 0.0000000000083317 | 6 |
| [6.20 ; 6.80 ] | 6.4176200000000 | 0.00000000000013811 | 4 |
| [8.60 ; 9.20 ] | 8.8924700000000 | 0.000000000026883 | 5 |
| MATLAB funkcijos | Pradinis artinys | (Fzero) |  |  |
| 1 | 1.7744 |  |  |
| 2.5 | 2.9499 |  |  |
| 6 | 6.4176 |  |  |
|  | 8 | 8.8925 |  |  |

## Išvados

I projektinės užduoties.,1 dalies darbo metu ieškojau netiesinių lygčių (daugianario ir funkcijos) šaknų trimis skirtingais metodais: Stygų, Skenavimo ir Kvazi - Niutono (kirstinių).

Realizavau visus tris metodus ir pastebėjau, kad iš gautų rezultatų galima teigti, kad pats neefektyviausias metodas yra Skenavimo, šiuo metodu pasiekiamas ženkliai didesnis iteracijų skaičius. Pats efektyviausias metodas, šiame darbe, yra Kvazi – Niutono (kirtsinių), parinkus tinkamus pirmąjį ir antrąjį artinius jam prireikė mažiausiai iteracijų šaknų radimui.

## Programų tekstai

* Daugianario šaknų  rėžių įverčio nustatymas

function Daugianario\_saknu\_reziu\_iverciai

clc, close all

syms f x

f=x.^4+7/2\*x.^3-23/2\*x.^2-43/2\*x-15/2

fneig=subs(f,x,-x) % daugianario skleistine pakeitus x-> -x

[CF1,orders]=coeffs(f,x) % daugianario f koeficientai ir juos atitinkantys x laipsniai

auksciausias\_x\_laipsnis=char(orders(1));

nnn=strfind(auksciausias\_x\_laipsnis,'^');

n=str2num(auksciausias\_x\_laipsnis(nnn+1:end)) % auksciausias x laipsnio rodiklis daugianaryje (daugianario eile)

[CF1\_neig,orders\_neig]=coeffs(fneig,x) % daugianario fneig koeficientai ir juos atitinkantys x laipsniai

% suformuojama visu x laipsniu eile:

for i=1:n+1, orders\_full(i)=x^(n-i+1); end

% koeficientu eile papildoma nuliniais nariais:

for i=1:n+1

j=find(orders == orders\_full(i));

if j>0, CF(i)=CF1(j);

CF\_neig(i)=CF1\_neig(j);

else, CF(i)=0;

CF\_neig(i)=0;

end

end

% koeficientas prie auksciausio x laipsnio turi buti teigiamas:

CF=CF/CF(1); % f(x) koeficientai

CF\_neig=CF\_neig/CF\_neig(1); % f(-x) koeficientai

% Saknu intervalo iverciai:

% ------------- Grubus ivertis:

CF\_value=eval(CF) % f(x) koeficientu simboliai paverciami skaiciais

R=max(abs(CF\_value(2:end)))/CF\_value(1)+1 % taikoma grubaus ivercio formule

% grafinis funkcijos, saknu ir grubaus ivercio intervalo pavaizdavimas:

t=-R:R/500:R;

figure(1);grid on;hold on

plot(t,fnk(CF\_value,t),'g-')

plot([-R,R],[0 0],'r\*')

% ------------ Tikslesnis ivertis:

% teigiamoms saknims:

neig\_ind=find(CF\_value(2:end) < 0)

if ~isempty(neig\_ind)

B=max(abs(CF\_value(neig\_ind+1)))

k=neig\_ind(1)

Rteig=1+(B/CF\_value(1))^(1/k)

else

Rteig=0

end

plot(min(R,Rteig),0,'bp') % pavaizduojamas teigiamu saknu virsutines ribos ivertis

% neigiamoms saknims:

CF\_value\_neig=eval(CF\_neig) % f(-x) koeficientu simboliai paverciami skaiciais

neig\_ind1=find(CF\_value\_neig(2:end) < 0)

if ~isempty(neig\_ind1)

B=max(abs(CF\_value\_neig(neig\_ind1+1)))

k=neig\_ind1(1)

Rneig=1+(B/CF\_value\_neig(1))^(1/k)

else

Rneig=0

end

plot(-min(R,Rneig),0,'bp')

legend('kreive f(x)','grubus saknu intervalo ivertis','tikslesnis saknu intervalo ivertis');

title([char(f),'=0'])

% vaizduoja tikslesniam iveryje

figure(2)

aa=-Rneig-1:0.01:Rteig+1;

plot(aa, fnk(CF\_value, aa), 'g-');

xlim([-Rneig-2,Rteig+2])

ylim([-0.1\*10^5,0.3\*10^5])

grid on;

hold on

legend('kreive f(x)','tikslesnis saknu intervalo ivertis');

title([char(f),'=0'])

% funkcija g

figure(3)

g = @(x)1.9\*x.\*sin(x)-(x/1.5-3).^2

x = -10:0.1:10;

s(1)=fzero('1.9\*x.\*sin(x)-(x/1.5-3).^2',1)

plot(x, g(x), 'g-')

ylim([-120,30])

grid on;

legend('kreive g(x)');

title(['1.9\*x.\*sin(x)-(x/1.5-3).^2=0'])

end

function p=fnk(CF,x)

% Apskaiciuoja daugianario reiksmes, kai argumentas yra x

% Kai x yra reiksmiu vektorius, p taip pat yra atitinkamu funkcijos reiksmiu vektorius

p=0; n=length(CF)-1;

for i=1:length(CF), p=p+CF(i)\*x.^(n-i+1); end % veiksmas < .^ > reiskia, kad laipsniu keliami visi vektoriaus x elementai

return

end

* Daugianario šaknų intervalų nustatymas

clear all; close all ; clc;

f = @(x)x.^4+7/2\*x.^3-23/2\*x.^2-43/2\*x-15/2;

x = -5.1:0.01:3.1;

% f = @(x)1.9\*x.\*sin(x)-(x/1.5-3).^2;

% x = -10:0.1:10;

plot(x, f(x), 'g', 'LineWidth', 2);

ylim([-2,2])

grid on; hold on;

i = -10;

zingsnis = 0.14;

%zingsnis = 0.6;

while (i <= 10)

ats=f(i);

ats2=f(i+zingsnis);

P =sprintf('i=%d, ats=%d, i2=%d, ats2=%d',i,ats,i+zingsnis,ats2);

if (ats > 0 && ats2 < 0) || (ats < 0 && ats2 > 0) || ( ats == 0 || ats2 == 0)

G =sprintf('[%0.2f ; %0.2f ]',i,i+zingsnis);

disp(G);

plot(i, 0, '>' , ...

'MarkerFaceColor', 'r', 'MarkerSize' , 7)

plot(i+zingsnis, 0, '<' , ...

'MarkerFaceColor', 'r', 'MarkerSize' , 7)

end

i = i + zingsnis;

end

* Stygų metodas

clc,close all

%------------------------ PRADINIAI DUOMENYS ---------------------------

%f='x.^4+7/2\*x.^3-23/2\*x.^2-43/2\*x-15/2'

%range=[-5.10 ; -4.96 ]

%range=[-1.04 ; -0.90 ]

%range=[-0.62 ; -0.48 ]

%range=[2.88 ; 3.02 ]

f = '1.9\*x.\*sin(x)-(x/1.5-3).^2'

range=[1.40 ; 2.00 ]

% range=[2.60 ; 3.20 ]

% range=[6.20 ; 6.80 ]

%range=[8.60 ; 9.20 ]

eps=1e-9; % parenkame tikslumo reiksme

nitmax=100;% parenkame didziausia leistina iteraciju skaiciu

method='chords';

% braizomas funkcijos grafikas

npoints=1000; x=range(1): (range(2)-range(1))/(npoints-1) :range(2);

figure(1); grid on; hold on;

str=[f,'=0; Method of ',method]; title(str);

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

plot(range,[0 0],'b-');

%------------------------ SPRENDIMAS -----------------------------------

xn=range(1);xn1=range(2);prec=1;

nit=0;

while prec > eps

nit=nit+1;

if nit > nitmax, fprintf('Virsytas leistinas iteraciju skaicius');break;end

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

% paskutinio grafinio objekto valdiklis irasomas handle masyvo priekyje

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

x=xn;fxn=eval(f);x=xn1;fxn1=eval(f);

k=abs(fxn/fxn1);xmid=(xn+k\*xn1)/(1+k);

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

x=xmid;fxmid=eval(f);

% jeigu pradzioje tikriname kairi taska

x=xn;fxn=eval(f);

if sign(fxmid) == sign(fxn), xn=xmid;

else, xn1=xmid;

end

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

% skaiciavimas stabdomas iki bus paspaustas Enter klavisas

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

prec=abs(fxmid);

fprintf(1,'iteracija %d tikslumas= %g \n',nit,prec);

end

plot(xmid,0,'k\*');plot(xmid,0,'ko');

fprintf(1,'\n tikslumas pasiektas, saknis xmid=%g\n\n',xmid);

* Skenavimo metodas

clc,close all

%------------------------ PRADINIAI DUOMENYS ---------------------------

%f = 'x.^4+7/2\*x.^3-23/2\*x.^2-43/2\*x-15/2'

%ff = @(x)x.^4+7/2\*x.^3-23/2\*x.^2-43/2\*x-15/2

%range=[-5.10 ; -4.96 ]

%range=[-1.04 ; -0.90 ]

%range=[-0.62 ; -0.48 ]

%range=[2.88 ; 3.02 ]

f = '1.9\*x.\*sin(x)-(x/1.5-3).^2'

ff = @(x)1.9\*x.\*sin(x)-(x/1.5-3).^2

%range=[1.40 ; 2.00 ]

% range=[2.60 ; 3.20 ]

% range=[6.20 ; 6.80 ]

range=[8.60 ; 9.20 ]

eps=1e-9; % parenkame tikslumo reiksme

method='skenavimo';

% braizomas funkcijos grafikas

npoints=1000; x=range(1): (range(2)-range(1))/(npoints-1) :range(2);

figure(1); grid on; hold on;

str=[f,'=0; Metodas : ',method]; title(str);

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

plot(range,[0 0],'b-');

riba1 = range(1);

riba2 = range(2);

interacijos = 1;

rez = 100;

while rez > eps

zingsnis = riba1 - riba2;

zingsnis = zingsnis/4;

zingsnis = abs(zingsnis);

i = riba1;

while (i <= riba2)

ats=ff(i);

ats2=ff(i+zingsnis);

if(abs(ats) < abs(ats2))

rez = abs(ats);

saknis = i;

end

if(abs(ats2) < abs(ats))

rez = abs(ats2);

saknis = i+zingsnis;

end

G =sprintf('interacija %d, tikslumas %d, rezis : [%d ; %d ]',interacijos,rez,i,i+zingsnis);

disp(G);

interacijos = interacijos + 1;

plot(i, ats, 'o' , ...

'MarkerFaceColor', 'r', 'MarkerSize' , 7);h = findobj(gca,'Type','line');h1=h(1);

plot(i+zingsnis, ats2, 'o' , ...

'MarkerFaceColor', 'r', 'MarkerSize' , 7);h = findobj(gca,'Type','line');h2=h(1);

input('Press Enter'), figure(1); % skaiciavimas stabdomas iki bus paspaustas Enter klavisas

delete(h1);delete(h2);

if (ats > 0 && ats2 < 0) || (ats < 0 && ats2 > 0) || ( ats == 0 || ats2 == 0)

riba1 = i;

riba2 = i + zingsnis;

break;

end

i = i + zingsnis;

end

end

disp(saknis);

* Kvazi – Niutono (kirstinių) metodas

clc, close all

syms f x

x0=-5.10; % parenkame pradini artini

x01=-1.90; % kirstiniu metodui parenkame antra pradini artini

deltax=0.3; % parenkame pradine zingsnio reiksme (reikalinga tik kirstiniu metodui)

nitmax=100; % parenkame didziausia leistina iteraciju skaiciu

%f='x.^4+7/2\*x.^3-23/2\*x.^2-43/2\*x-15/2'

% range=[-5.10 ; -4.96 ]

% x0=-5.10;

% x01=-4.96;

%range=[-1.04 ; -0.90 ]

% x0=-1.04;

% x01=1.0;

% range=[-0.62 ; -0.48 ]

% x0=-0.62;

% x01=-0.48;

% range=[2.88 ; 3.02 ]

% x0=2.88;

% x01=3.5;

f = '1.9\*x.\*sin(x)-(x/1.5-3).^2'

% range=[1.40 ; 2.00 ]

% x0=1.40;

% x01=2.00;

% range=[2.60 ; 3.20 ]

% x0=2.60;

% x01=3.20;

% range=[6.20 ; 6.80 ]

% x0=6.20;

% x01=6.80;

range=[8.60 ; 9.20 ]

x0=8.60;

x01=9.20;

eps=1e-9; % Parenkame tiksluma

method='secants';

if strcmp(method,'secants'),

x=x01;fxn1=eval(f);x=x0;fxn=eval(f);

dfxn=(fxn1-fxn)/(x01-x0); end % Taikant kirstiniu metoda, reiks apskaiciuoti ,

% pradines kirstines krypti pagal du pradinius artinius

% braizomas funkcijos grafikas:

npoints=1000; xrange=range(1): (range(2)-range(1))/(npoints-1) :range(2);

figure(1); grid on; hold on; axis equal; str=[char(f),'=0; Method of ',method];title(str);

x=xrange; % simbolinis x keiciamas reiksmemis is parinkto funkcijos vaizdavimo intervalo

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

plot(range,[0 0],'b-');

plot(x0,0,'mp');

h = findobj(gca,'Type','line');

h1=h(1);

% input('Press Enter'), figure(1); % skaiciavimas stabdomas iki bus paspaustas Enter klavisas

%------------------------ SPRENDIMAS -----------------------------------

xn=x0;prec=1;nit=0;

if strcmp(method,'secants'),

xn1=x01;

plot([xn,xn,xn1,xn1],[0,fxn,fxn1,0],'k-');

end % antras pradinis artinys

while prec > eps % iteracijos

nit=nit+1;

if nit > nitmax, fprintf('Virsytas leistinas iteraciju skaicius');break;end

xn1=xn-fxn/dfxn;

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

delete(h1);plot(xn1,0,'mp');h = findobj(gca,'Type','line');h1=h(1);

x=xn1;fxn1=eval(f);dfxn=(fxn1-fxn)/(xn1-xn);

xn=xn1;

fxn=fxn1;

input('Press Enter'), figure(1); % skaiciavimas stabdomas iki bus paspaustas Enter klavisas

x=xn;fxn=eval(f);prec=abs(fxn);

fprintf(1,'iteracija %d x= %g prec= %g \n',nit,xn,prec);

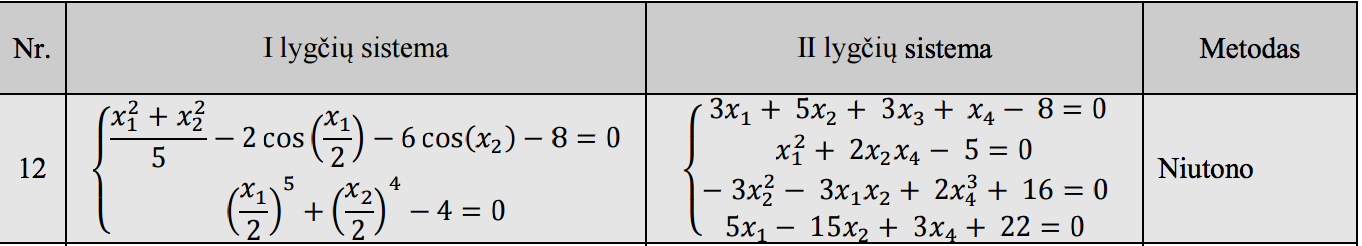
end

plot(xn,fxn,'k\*');plot(xn,fxn,'ko');

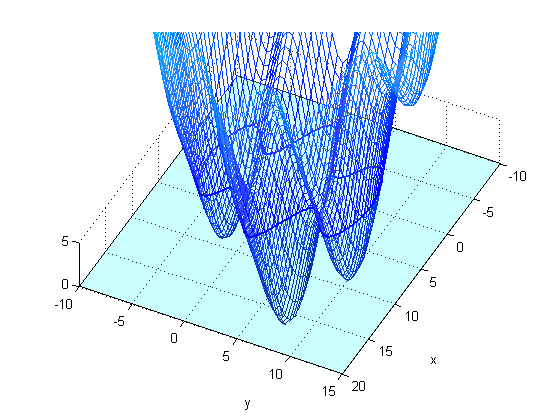
xn

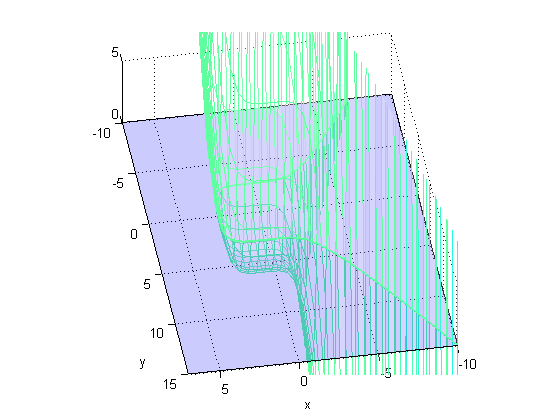
nit

# Netiesinių lygčių sistemų sprendimas



## 2.1 I-os netiesinių lygčių sistemos sprendimas

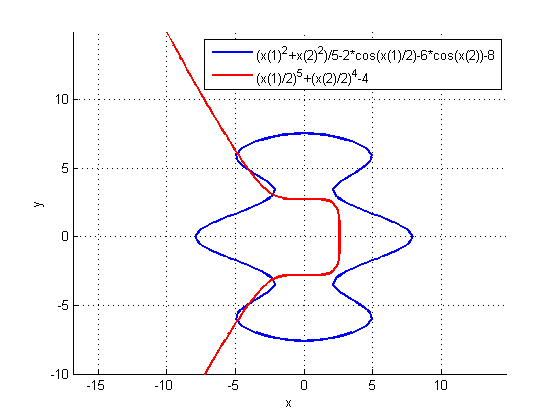
* Paviršių grafinis vaizdas



a b

Figūra 5 Paviršių grafinis vaizdas

* Sprendimas grafiniu būdu

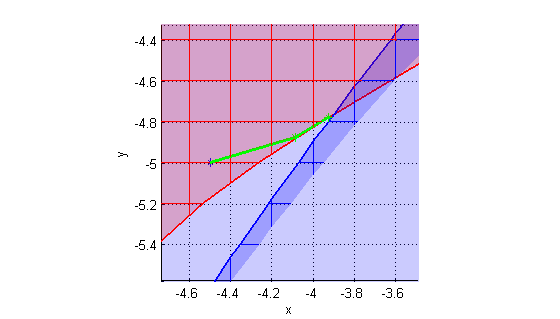
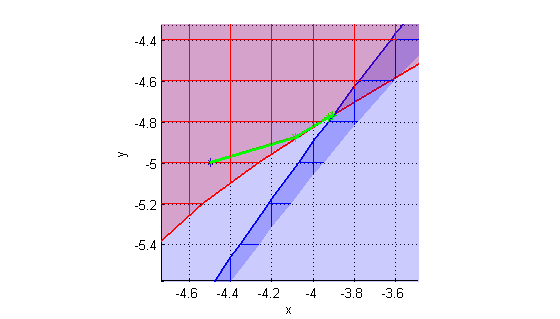
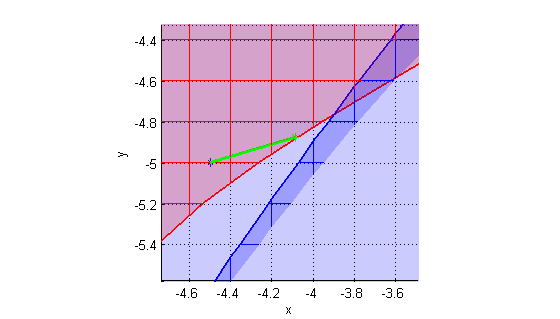


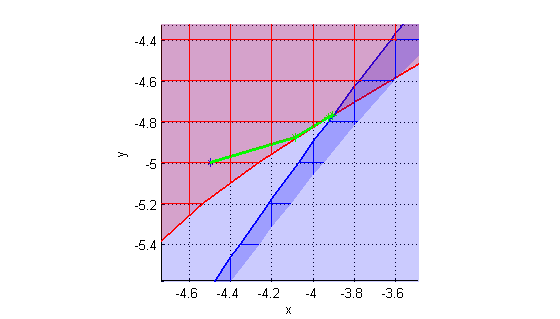
Figūra 6 Lygčių sistemos sprendiniai nustatomi grafikų susikirtimo taškuose.

* Sprendimas Niutono metodu

Tariama, kad 𝑥𝑔 yra sprendinys (stabdomi skaičiavimai), jei |𝑓(𝑥𝑔)| < 1𝑒 − 5, čia 𝑓(𝑥) – tikslo funkcija. Pradinis metodo žingsnis 1.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Pradinis artinys** | **Sprendinys Niutono metodu** | **Tikslumas** | **Iteraciju skaičius** | **Sprendinys MATLAB funkcija fsolve** |
| [-5 ; 6.3] | [-4.89124 ; 6.18541] | 8.5271e-6 | 3 | [-4.8912 ; 6.1854] |
| [-4.5 ; 5] | [-3.90394 ; -4.76933] | 1.29585e-8 | 5 | [-3.9039; -4.7693] |
| [-3.5 ; 3.2] | [-2.22182 ; -3.0892] | 1.35433e-8 | 6 | [-2.2218; -3.0892] |
| [-5 ; -6.3] | [-4.89124 ; -6.18541] | 8.5271e-6 | 3 | [-4.8912; -6.1854] |
| [-4.5 ; -5] | [-3.90394 ; -4.76933] | 1.29585e-8 | 5 | [-3.9039; -4.7693] |
| [-3.5 ; -3.2] | [-2.22182 ; -3.0892] | 1.35433e-8 | 6 | [-2.2218; -3.0892] |





## 

Figūra 7 Niutono metodo vizualizacija nuo pradinio artinio [-4.5;-5]

## 2.2 II-os netiesinių lygčių sistemos sprendimas

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Pradinis artinys** | **Sprendinys Niutono metodu** | **Tikslumas** | **Iteraciju skaičius** | **Sprendinys MATLAB funkcija fsolve** |
| [1;1;1;1] | 1 2 -2 1 | 1.39986e-11 | 21 | 1.0000  2.0000  -2.0000  1.0000 |

## 2.3 Išvados

I projektinės užduoties.,2 dalies darbo metu ieškojau netiesinių lygčių sistemų sprendinių Niutono metodu, ir rezultatus lyginau su MATLAB funkcijos fsolve rezultatais.

Pastebėjau, kad Niutono metodas yra tikslesnis negu fsolve funkcija. Taip pat išmokau grafiškai pavaizduoti lygčių sistemas.

## 2.2 Programų tekstai

* Fsolve metodas

function fsolveDemo

% x0=[-5;6.3];

% x0=[-4.5;-5];

% x0=[-3.5;-3.2];

% x0=[-5;-6.3];

% x0=[-4.5;-5];

% x0=[-3.5;-3.2]; % Make a starting guess at the solution

x0=[1;1;1;1]

options=optimset('Display','iter'); % Option to display output

[x,fval] = fsolve(@myfun,x0,options) % Call solver

end

function F = myfun(x)

% Rewrite the equation in the form F(x) = 0

% F=[(x(1)^2+x(2)^2)/5-2\*cos(x(1)/2)-6\*cos(x(2))-8;

% (x(1)/2)^5+(x(2)/2)^4-4];

F=[3\*x(1)+5\*x(2)+3\*x(3)+x(4)-8;

x(1)^2+2\*x(2)\*x(4)-5;

-3\*x(2)^2-3\*x(1)\*x(2)+2\*x(4)^3+16;

5\*x(1)-15\*x(2)+3\*x(4)+22];

end

* Grafinis metodas lygčių sistemai

%Grafinis metodas lygciu sistemai

function pagrindine

clc,close all

x=[-10:0.5:20];y=[-10:0.5:45];

Z=pavirsius(@f,x,y);

figure(1),hold on,grid on,axis equal,axis([min(x) max(x) min(y) max(y) 0 5]);view([1 1 1]);

xlabel('x'),ylabel('y');

mesh(x,y,Z(:,:,1)','FaceAlpha',0.2);contour(x,y,Z(:,:,1)',[0,0 ],'LineWidth',1.5);

xx=axis;

fill([xx(1),xx(1),xx(2),xx(2)],[xx(3),xx(4),xx(4),xx(3)],'c','FaceAlpha',0.2);

figure(2),hold on,grid on,axis equal,axis([min(x) max(x) min(y) max(y) 0 5]);view([1 1 1]);

xlabel('x'),ylabel('y')

mesh(x,y,Z(:,:,2)','FaceAlpha',0.2);contour(x,y,Z(:,:,2)',[0 0],'LineWidth',1.5)

xx=axis;

fill([xx(1),xx(1),xx(2),xx(2)],[xx(3),xx(4),xx(4),xx(3)],'b','FaceAlpha',0.2);

figure(3),hold on,grid on,axis equal

contour(x,y,Z(:,:,1)',[0 0],'LineWidth',1.5,'LineColor','b')

contour(x,y,Z(:,:,2)',[0 0],'LineWidth',1.5,'LineColor','r')

% contour(x,y,Z(:,:,1)')

% contour(x,y,Z(:,:,2)')

% ylim([-5 0])

% xlim([-3 0])

xlabel('x'),ylabel('y')

legend('(x(1)^2+x(2)^2)/5-2\*cos(x(1)/2)-6\*cos(x(2))-8','(x(1)/2)^5+(x(2)/2)^4-4')

return

end

% Lygciu sistemos funkcija

function fff=f(x)

fff=[(x(1)^2+x(2)^2)/5-2\*cos(x(1)/2)-6\*cos(x(2))-8;

(x(1)/2)^5+(x(2)/2)^4-4];

return

end

function Z=pavirsius(funk,x,y)

for i=1:length(x)

for j=1:length(y)

Z(i,j,1:2)=funk([x(i),y(j)]);

end

end

return

end

* Niutono metodas 2 lygčių sistemai

% Niutono metodas

function pagrindine

clc,close all

scrsz = get(0,'ScreenSize')

% x=[-7:0.2:0];y=[0:0.2:10];

x=[-7:0.2:0];y=[-10:0.2:0];

% x=[-20:0.2:20];y=[-20:0.2:20];

Z=pavirsius(@f,x,y);

fig1=figure(1);set(fig1,'Position',[50 scrsz(4)/1.8 scrsz(3)/3 scrsz(4)/3],'Color','w');

hold on,grid on,axis equal,axis([min(x) max(x) min(y) max(y) 0 5]);view([0 0 1]);xlabel('x'),ylabel('y');

mesh(x,y,Z(:,:,1)','FaceAlpha',0.2,'FaceColor','r','EdgeColor','r');contour(x,y,Z(:,:,1)',[0 0],'LineWidth',1.5,'LineColor','r');

mesh(x,y,Z(:,:,2)','FaceAlpha',0.22,'FaceColor','b','EdgeColor','b');contour(x,y,Z(:,:,2)',[0 0],'LineWidth',1.5,'LineColor','b');

xx=axis; fill([xx(1),xx(1),xx(2),xx(2)],[xx(3),xx(4),xx(4),xx(3)],'b','FaceAlpha',0.2);

eps=1e-5;itmax=200;

% x=[-5;6.3];

% x=[-4.5;-5];

% x=[-3.5;-3.2];

% x=[-5;-6.3];

% x=[-4.5;-5];

x=[-3.5;-3.2];

ff=f(x); dff=df(x);

figure(1);plot3(x(1),x(2),0,'b\*');line([x(1),x(1)],[x(2),x(2)],[0,ff(1)],'Color','black');

alpha=1 %1 %0.9 %0.8; % zingsnio sumazinimo koeficientas

for iii=1:itmax

dff=df(x); deltax=-dff\ff; x1=x+alpha\*deltax; ff1=f(x1);

figure(1);plot3(x1(1),x1(2),0,'r\*');line([x(1),x1(1)],[x(2),x1(2)],[0,0],'Color','red');

line([x(1),x1(1)],[x(2),x1(2)],[ff(1),0\*ff1(1)],'Color','magenta','LineWidth',2.5);

line([x1(1),x1(1)],[x1(2),x1(2)],[0,ff1(1)],'Color','black');

pause;

tikslumas=norm(deltax)/(norm(x)+norm(deltax));

fprintf(1,'\n iteracija %d tikslumas %g',iii,tikslumas);

if tikslumas < eps, fprintf(1,'\n sprendinys x ='); fprintf(1,' %g ',x);plot3(x(1),x(2),0,'rp'); break;

elseif iii == itmax,fprintf(1,'\n \*\*\*\*tikslumas nepasiektas. Paskutinis artinys x = %g',x'); plot3(x(1),x(2),0,'gp'); break;

end

x=x1;ff=ff1;

end

fprintf(1,'\n');

return

end

% Lygciu sistemos funkcija

function fff=f(x)

fff=[(x(1)^2+x(2)^2)/5-2\*cos(x(1)/2)-6\*cos(x(2))-8;

(x(1)/2)^5+(x(2)/2)^4-4];

return

end

% Jakobio matrica

function dfff=df(x)

dfff=[(2\*x(1))/5 + sin(x(1)/2), (2\*x(2))/5 + 6\*sin(x(2));

(5\*x(1)^4)/32, x(2)^3/4];

return

end

function Z=pavirsius(funk,x,y)

for i=1:length(x)

for j=1:length(y)

Z(i,j,1:2)=funk([x(i),y(j)]);

end

end

return

end

* Niutono metodas 4 lygčių sistemai

% Niutono metodas

function pagrindine

clc,close all

eps=1e-10

itmax=100

x=[1;1;1;1];

% x=[-0.828147 6.95332 -4.33552 2.98925]'

for iii=1:itmax

deltax=-df(x)\f(x);

x=x+deltax;

tikslumas=norm(deltax)/(norm(x)+norm(deltax));

fprintf(1,'\n iteracija %d tikslumas %g',iii,tikslumas);

if tikslumas < eps

fprintf(1,'\n sprendinys x ='); fprintf(1,' %g',x);

fprintf(1,'\n funkcijos reiksme f ='); fprintf(1,' %g',f(x));

break

elseif iii == itmax

fprintf(1,'\n \*\*\*\*tikslumas nepasiektas. Paskutinis artinys x =');

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

fprintf(1,'\n funkcijos reiksme f =');

fprintf(1,' %g',f(x));

break

end

end

return

end

% Lygciu sistemos funkcija

function F=f(X)

F(1)=3\*X(1)+5\*X(2)+3\*X(3)+X(4)-8;

F(2)=X(1)^2+2\*X(2)\*X(4)-5;

F(3)=-3\*X(2)^2-3\*X(1)\*X(2)+2\*X(4)^3+16;

F(4)=5\*X(1)-15\*X(2)+3\*X(4)+22;

F=F(:);

return

end

% Jakobio matrica

function DF=df(X)

DF(1,1)=3; DF(1,2)=5; DF(1,3)=3; DF(1,4)=1;

DF(2,1)=2\*X(1); DF(2,2)=2\*X(4); DF(2,3)=0; DF(2,4)=2\*X(4);

DF(3,1)=-3\*X(1); DF(3,2)=-3\*X(1)-6\*X(2); DF(3,3)=0; DF(3,4)=6\*X(3)^2;

DF(4,1)=5; DF(4,2)=-15; DF(4,3)=0; DF(4,4)=3;

return

end