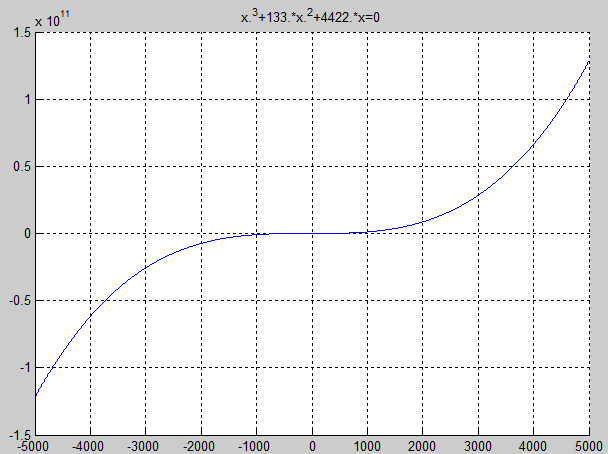
**Kolokviumo užduotys**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Grupė** | **Pavardė Vardas** | **Savarankiško darbo Nr.** | **Lygčių Nr.** | **Sprendimo metodas** | |
| **daugianario** | **sistemos** |
| IF-0/7  IFF0 | Šarūnas Intas | 9 | 3 | Pusiaukirtos | Lu Skaidos |

1. *Vienos lygties sprendimo algoritmai*. **Duota** daugianario lygtis f(x)=0.
   1. Grafiškai pavaizduota funkcija.



Programos kodas:

function Atvaizdavimas

clc, close all

syms x f g

f='x.^3+133.\*x.^2+4422.\*x'

figure(1);

grid on;

hold on;

xmin=-5000; xmax=5000; nst=100000; % vaizdavimo reziai ir tasku skaicius

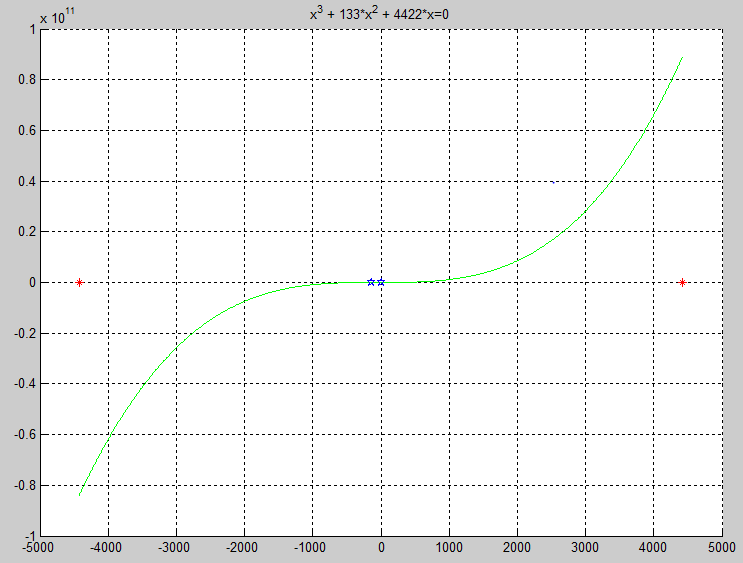
x=[xmin :(xmax-xmin)/nst : xmax];

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

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

end

* 1. Nustatyti daugianario f(x) šaknų intervalai, taikant „grubų“ ir tikslesnį įverčius.



Programos kodas:

function iverciai

clc, close all

syms f x

f=x.^3+133.\*x^2+4422.\*x;

f=expand(f) % daugianario skleistine

fneig=expand(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

orders

% koeficientu eile papildoma nuliniais nariais (jeigu eil?je n?ra visi nuosekl?s x laipsniai):

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

CF,CF\_neig

% 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,subs(f,t),'g-')

grubusIverciai=[-R,R]

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

RteigIvert=[min(R,Rteig),0]

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

Rneigvert=[-min(R,Rneig),0]

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

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

title([char(f),'=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

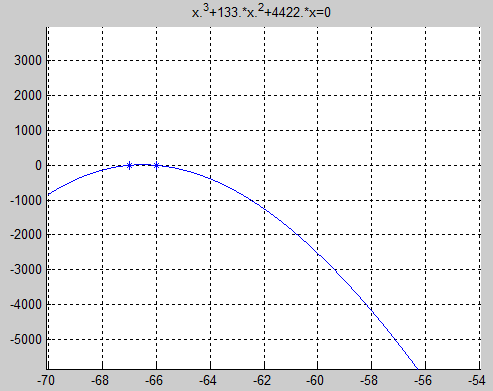
p=0; n=length(CF)-1;

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

* 1. Daugianario grafikas su pažymėtomis šaknimis.



* 1. Rezultatų lentelė.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Metodas** | | **Lygtis** | | | **Šaknų intervalo įverčiai** | |
| Pusiaukirtos | |  | | | Grubus: -4423 ir 4423  Tikslesnis: -134 ir 0 | |
| **Šaknis** | **Intervalas**  ***arba* artinys** | | **Tikslumas** | **Iteracijų skaičius** | **Iteracijų pabaigos sąlygos** | ***roots* funkcijos rezultatas** |
| -67 | Xn = -68  Xn1 = -66.5 | | 9.8953e-010 | 35 | |f(x)|<1e-9 | -67  -66 |
| -66 | Xn = -66.5  Xn1 = -65 | | 8.73115e-010 | 35 | |f(x)|<1e-9 |

1. Programos kodą.

%

% Vienos lygties sprendimas: pusiaukirtos ir stygu metodai

%

function Pvz\_SMA\_1\_3\_Viena\_lygtis\_bisection\_chords

clc,close all

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

% f='2\*x.\*cos(2\*x)-(x+1).^2' % funkcijos simboline israiska

f='x.^3+133.\*x.^2+4422.\*x'

% f='x^2' % tokiai funkcijai siais metodais saknu nerasime

range=[-80,-50];

range=[-68,0];

eps=1e-9; % parenkame tikslumo reiksme

nitmax=100;% parenkame didziausia leistina iteraciju skaiciu

% 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;'; title(str);

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

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

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

xn=-66.5;

xn1=-65;

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);

% pasirenkame, ar spresti pusiaukirtos, ar stygu metodu:

xmid=(xn+xn1)/2;

plot(xmid,0,'gs');

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

h3=h(1);

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

% jeigu pradzioje tikriname kairi taska

%x=xn1;

%fxn1=eval(f);

%if sign(fxmid) == sign(fxn1), xn1=xmid;

%else, xn=xmid;

% end

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);

% ................................................................................

disp('.... Patikriname saknies reiksme, naudodami MATLAB funkcija roots: ....')

fprintf('\n')

roots([1 133 4422])

end

1. *Tiesinių lygčių sistemų sprendimo algoritmai*. **Duota** tiesinių lyčių sistema [A][X]=[B].
   1. Išspręskime šias sistemas, naudodami Lu skaidos metodą.
   2. Patikrinkite gautus sprendinius, naudodami MATLAB lygčių sistemų sprendimo (visiems metodams) ir matricų skaidos funkcijas (skaidos metodams).

Tame pačiame *Word* faile pateikite **rezultatus**:

1. Rezultatų lentelę.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sprendinys [X]** | **Skaidos rezultatai (skaidos metodams)** | **Sprendinio patikrinimas** | | |
| **[A][X]-[B]** | **Lygčių sistemos sprendimo funkcija (nurodykite konkrečią)** | **Skaidos funkcija (nurodykite konkrečią)** |
| -1.0108  4.2628  -2.9697  1.1945  0.4541 |  | 1.0e-014 \*  0.1776  0.7105  -0.1776  0.3553  0.0888 | linsolve(A,b)  Rezultatai:  -1.0108  4.2628  -2.9697  1.1945  0.4541 |  |

1. Programos kodą.

% LU skaida

function laoras\_15\_LU\_skaida

clc, close all, clear all

A=[4 3 2 1 0;

5 2 8 1 -2;

0 1 2 14 1;

9 6 2 1 16;

16 3 -1 4 -3]

b=[4; -20; 15.5; 19; 3]

A1=A; b1=b;

n=size(A, 1)

P=[1:n]

%tiesioginis zingsnis

for i=1:n-1

[a, iii]=max(abs(A1(i:n,i)));

A1([i,iii+i-1], :)=A1([iii+i-1,i], :);

P([i,iii+i-1])=P([iii+i-1,i])

for j=i+1:n

r = A1(j, i)/A1(i, i);

A1(j,i+1:end)=A1(j,i+1:end)-A1(i,i+1:end)\*r;

A1(j,i)=r;

end

end

%atvirkstinis zingsnis

b1=b1(P);

for i=2:n

b1(i)=(b1(i)-A1(i, 1:i-1)\*b1(1:i-1));

end

for i=n:-1:1

b1(i)=(b1(i)-A1(i, i+1:n)\*b1(i+1:n))/A1(i,i);

end

A1

atsakymas=b1

% tikslumo tikrinimas

liekana=A\*b1-b;

liekana

%tikrinimas su MatLab funkcija

X = linsolve(A,b)

end