Kauno Technologijos Universitetas

**Skaitiniai metodai ir algoritmai**

Namų darbas Nr. 1

Parengė: Tautvydas Petkus IFF-1

KAUNAS

2013

## 1a dalis

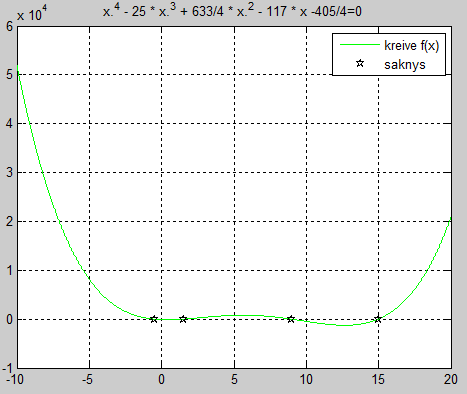
Variantas Nr. 3

Reikia rasti:

1. Daugianario *f(x)=x4-25x3-633/4x2-117x+405/4* šaknis skenavimo ir stygų metodu.
2. Funkcijos *g(x)=e-(x²)\*sin(x²)\*(x+2); -3≤x≤3* šaknis skenavimo ir Niutono metodu.

**Daugianario skenavimo metodas**

***Daugianario sprendinio grafikai***



***Programinis kodas***

clc;close all;clear all;

x = -10:.001:20;

f='x.^4 - 25 \* x.^3 + 633/4 \* x.^2 - 117 \* x -405/4';

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

grid on; hold on;

n=2000;

prad = -10;

pab = 20;

i=1;

while(prad <= 20)

x = prad;

fx=eval(f);

x = prad + 0.01;

fxpb = eval(f);

if sign(fx)~= sign(fxpb);

saknys(i) = fzero(f,prad) % kelios saknys

i=i+1;

end

prad = prad + 0.01;

end

plot(saknys,0\*saknys,'kp');%saknys

legend('kreive f(x)', 'saknys');

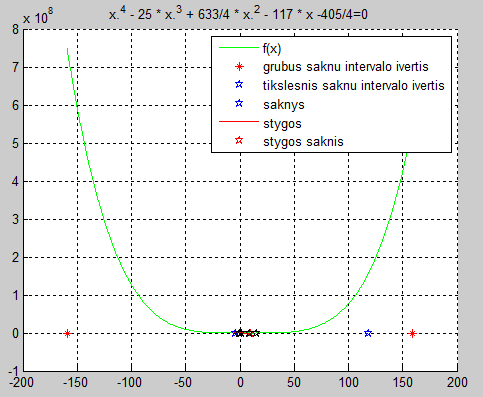
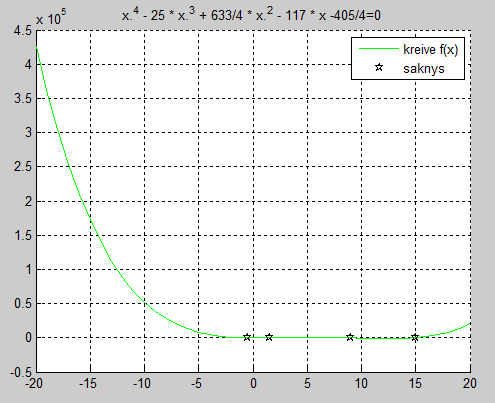
title([char(f),'=0']);

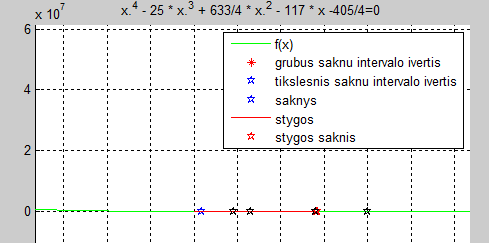
saknys =

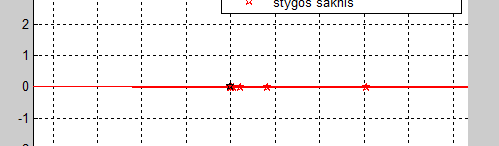
-0.5000 1.5000 9.0000 15.0000

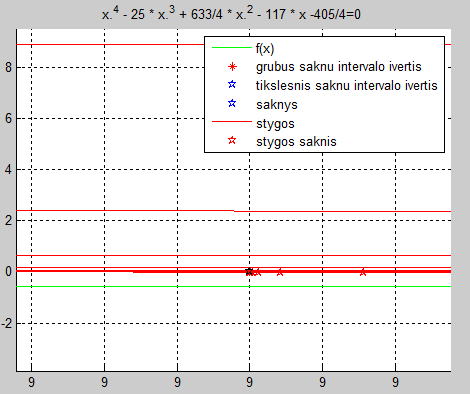
**Daugianario stygų metodas**

***Daugianario sprendinio grafikai***









***Rezultatų lentelė***

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Šaknis** | **Tikslumas** | **Iteracijų sk.** | **Iteracijų pabaigos sąlygos** | **Pradinis artinys** | **Fzero()**  **Šaknys** |
| 15 | 3.7835e-10 | 21 | f(x)<1e-9 | 14 | 15 |
| 9 | 9.45874e-11 | 8 | f(x)<1e-9 | 8 | 9 |
| 1.5 | 4.52076e-10 | 12 | f(x)<1e-9 | 1 | 1.5 |
| -0.5 | 5.32566e-10 | 20 | f(x)<1e-9 | -1 | -0.5 |

***Programinis kodas***

function stygu\_metodas\_var2

clc, close all

syms f x

f='x.^4 - 25 \* x.^3 + 633/4 \* x.^2 - 117 \* x -405/4';

%Grubus:

R=159.2500

%Tikslesnis:

Rteig=118

Rneig=-4.1721

x=[-R:R/500:R];

figure(1);

grid on;hold on;

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

run('Pvz\_SMA\_1\_2\_Daugianario\_saknu\_reziu\_iverciai');

Stygu\_skaiciavimas(-5, 10);

fprintf(1, '\nPatikrinimas:');

spr=roots([1 -25 633/4 -117 -405/4])

% grafikas ir saknys

plot(spr,zeros(size(spr)),'kp')

legend('f(x)','grubus saknu intervalo ivertis','tikslesnis saknu intervalo ivertis','saknys', 'stygos', 'stygos saknis');

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

figure(2);grid on;hold on;

x=[-20:1:20];

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

plot(spr,zeros(size(spr)),'kp');

legend('kreive f(x)', 'saknys');

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

end

function Stygu\_skaiciavimas(l, r)

syms f x;

f=x.^4 - 25 \* x.^3 + 633/4 \* x.^2 - 117 \* x -405/4;

eps=1e-9;

xn=l; xn1=r; prec=1;

iter=0;

while prec > eps

iter=iter+1;

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

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

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

ff = inline('x.^4 - 25 \* x.^3 + 633/4 \* x.^2 - 117 \* x -405/4');

plot([xn, xn1], [ff(xn), ff(xn1)], 'r-');

plot (xmid, ff(xmid), 'rp');

prec=abs(fxmid);

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

end

return;

end

**Pvz\_SMA\_1\_2\_Daugianario\_saknu\_reziu\_iverciai:**

function Pvz\_SMA\_1\_2\_Daugianario\_saknu\_reziu\_iverciai

clc, close all

syms f x

f=x.^4 - 25 \* x.^3 + 633/4 \* x.^2 - 117 \* x -405/4;

f=expand(f)

fneig=expand(subs(f,x,-x))

[CF1,orders]=coeffs(f,x)

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

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

n=str2num(auksciausias\_x\_laipsnis(nnn+1:end))

[CF1\_neig,orders\_neig]=coeffs(fneig,x)

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

orders

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

CF=CF/CF(1);

CF\_neig=CF\_neig/CF\_neig(1);

CF,CF\_neig

CF\_value=eval(CF) R=max(abs(CF\_value(2:end)))/CF\_value(1)+1

t=-R:R/500:R;

figure(1);grid on;hold on

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

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

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')

CF\_value\_neig=eval(CF\_neig)

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'])

end

function p=fnk(CF,x)

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

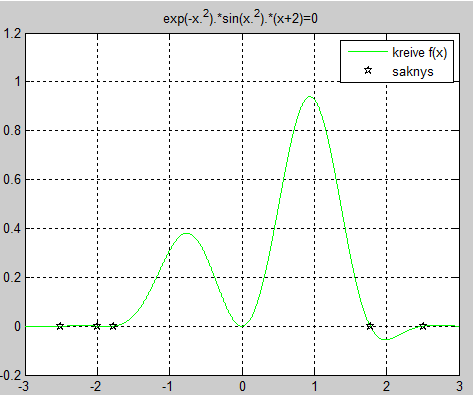
for i=1:length(CF), p=p+CF(i)\*x.^(n-i+1); end

return

end

**Funkcijos skenavimo metodas**

***Funkcijos sprendinio grafikai***



***Programinis kodas***

clc;close all;clear all;

x = -3:.001:3;

f='exp(-x.^2).\*sin(x.^2).\*(x+2)';

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

grid on; hold on;

n=2000;

prad = -3;

pab = 3;

i=1;

while(prad <= 3)

x = prad;

fx=eval(f);

x = prad + 0.01;

fxpb = eval(f);

if sign(fx)~= sign(fxpb);

saknys(i) = fzero(f,prad) % kelios saknys

i=i+1;

end

prad = prad + 0.01;

end

plot(saknys,0\*saknys,'kp');%saknys

legend('kreive f(x)', 'saknys');

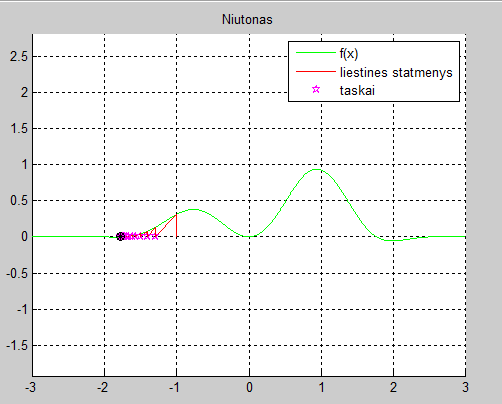
title([char(f),'=0']);

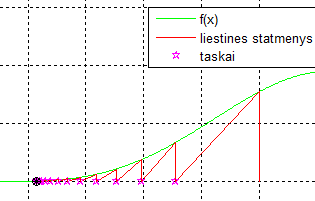
saknys =

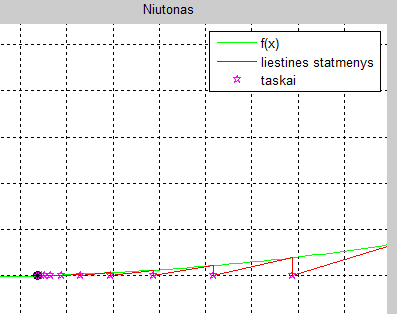
-2.5066 -2.0000 -1.7725 1.7725 2.5066

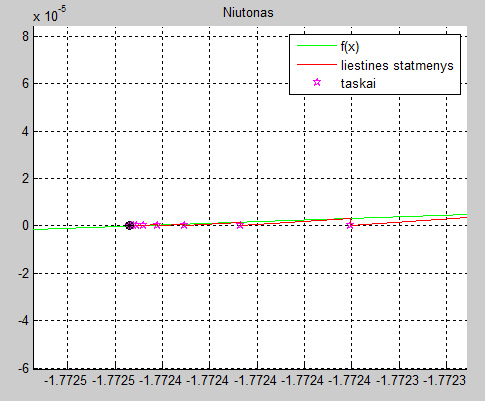
**Funkcijos Niutono metodas**

***Daugianario sprendinio grafikai***









***Rezultatų lentelė***

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| *-3≤x≤3* | | | | | |
| **Šaknis** | **Pradinis Artinys** | **Tikslumas** | **Iteracijų Skaičius** | **Iteracijos pabaigos sąlygos** | **Fzero rezultatas** |
| -2.5066 | -2.5 | 9.95178e-10 | 15 | |f(x)<1e-9| | -2.5066 |
| -2.0000 | -1.9 | 5.45184e-10 | 20 | |f(x)<1e-9| | -2.0000 |
| -1.7725 | -1 | 7.94496e-10 | 30 | |f(x)<1e-9| | -1.7725 |
| 1.7725 | 1.8 | 7.94292e-10 | 24 | |f(x)<1e-9| | 1.7725 |
| 2.5066 | 2.5 | 5.66205e-10 | 19 | |f(x)<1e-9| | 2.5066 |

***Programinis kodas***

function Niutono\_metodas

clc, close all

syms f x

f='exp(-x.^2).\*sin(x.^2).\*(x+2)';

x0=2.5;

nitmax=100;

range=[-3,3];

eps=1e-9;

beta=0.5 ;

df=diff(eval(f),x);

x=-3:0.01:3;

npoints=2000; xrange=range(1): (range(2)-range(1))/(npoints-1) :range(2);

figure(1); grid on; hold on; axis equal; title('Niutonas');

x=xrange;

plot(x,eval(f),'g-');h = findobj(gca,'Type','line');h1=h(1);

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

while prec > eps % iteracijos

nit=nit+1;

if nit > nitmax, fprintf('Iteraciju skaicius virsytas');break;end

x=xn;fxn=eval(f);dfxn=eval(df);

xn1=xn-beta\*fxn/dfxn;

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

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

xn=xn1;

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

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

end

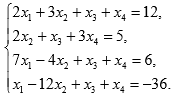
plot(xn,0,'k\*');plot(xn,0,'ko');

legend('f(x)','liestines statmenys','taskai');

## 2a dalis

Reikia išspręsti:

1. Tiesinių lygčių sistemą LU sklaidos metodu.



*x =*

*3.0000*

*3.0000*

*-4.0000*

*1.0000*

*LU=  
7.0000 -4.0000 1.0000 1.0000*

*1.0000 -12.0000 1.0000 1.0000*

*0 2.0000 1.0000 3.0000*

***Programinis kodas***

%LU

%2 3 1 1 = 12

%0 2 1 3 = 5

%7 -4 1 1 = 6

%1 -12 1 1 -36

function LU\_skaida

clc, close all, clear all

A=[2 3 1 1;

0 2 1 3;

7 -4 1 1;

1 -12 1 1]

b=[12; 5; 6; -36]

n=size(A,1)

P=[1:n]

Aold=A

bold=b

for i=1:n-1

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

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

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

for j=i+1:n

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

A(j,i+1:n)=A(j,i+1:n)-A(i,i+1:n)\*r;

A(j,i)=r;

end

end

b=b(P)

%Ly=b, y->b

for i=2:n, b(i,:)=b(i,:)-A(i,1:i-1)\*b(1:i-1); end

%Ux=b, x->b

for i=n:-1:1, b(i)=(b(i)-A(i,i+1:n)\*b(i+1:n))/A(i,i); end

fprintf(1, 'Sprendinys:');

x=b

fprintf(1, 'AX-B:');

Aold\*x-bold

fprintf(1, 'Patikrinimas:');

linsolve(Aold, bold)

% L ir U matricos

L=[];

U=[];

for i=1:n

U=[U; zeros(1, i-1), A(i,i:n)];

end

for i=1:n

L=[L; A(i,1:i-1) 1 zeros(1, n-i)];

end

fprintf(1, 'Gauta skaida');

U

L

fprintf(1, 'Atstatyta pradin? matrica');

A=L\*U;

A=A(P,:)

fprintf(1, 'Skaidos patikrinimas:');

[L,U,P]=lu(Aold);

U

L

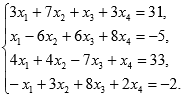
L\*U

end

*2.0000 3.0000 1.0000 1.0000*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sprendinys** | **Skaidos rezultatai (skaidos metodams)** | **Sprendinio patikrinimas** | | |
| **[A][X]-[B]** | **Lygčių sistemos sprendimo funkcija (nurodykite konkrečią)** | **Skaidos funkcija (nurodykite konkrečią)** |
| *3.0000*  *3.0000*  *4.0000*  *1.0000* | U = 7.0000 -4.0000 1.0000 1.0000  0 -11.4286 0.8571 0.8571  0 0 1.1500 3.1500  0 0 0 -1.7826  L = 1.0000 0 0 0  0.1429 1.0000 0 0  0 -0.1750 1.0000 0  0.2857 -0.3625 0.8913 1.0000 | 1.0e-14 \*    0.1776  0  -0.2665  0.7105 | linsolve(Aold, bold)  3.0000  3.0000  -4.0000  1.0000 | [L,U,P]=lu(Aold);  U =  7.0000 -4.0000 1.0000 1.0000  0 -11.4286 0.8571 0.8571  0 0 1.1500 3.1500  0 0 0 -1.7826  L =  1.0000 0 0 0  0.1429 1.0000 0 0  0 -0.1750 1.0000 0  0.2857 -0.3625 0.8913 1.0000 |

1. Tiesinių lygčių sistemą Gauso metodu.



**Pradinė lygtis:**

A1 =

3 7 1 3 31

1 -6 6 8 -5

4 4 -7 1 33

-1 3 8 2 -2

**Sprendimo eiga:**

A1 =

3.0000 7.0000 1.0000 3.0000 31.0000

0 -8.3333 5.6667 7.0000 -15.3333

0 -5.3333 -8.3333 -3.0000 -8.3333

0 5.3333 8.3333 3.0000 8.3333

A1 =

3.0000 7.0000 1.0000 3.0000 31.0000

0 -8.3333 5.6667 7.0000 -15.3333

0 0 -11.9600 -7.4800 1.4800

0 0 11.9600 7.4800 -1.4800

A1 =

3.0000 7.0000 1.0000 3.0000 31.0000

0 -8.3333 5.6667 7.0000 -15.3333

0 0 -11.9600 -7.4800 1.4800

0 0 0 0 -0.0000

**Rezultatas:**

x =

NaN

NaN

Inf

-Inf

**Patikrinimas:**

liekana =

NaN

NaN

NaN

NaN

**Paklaida:**

NaN

**SPRENDINIŲ SISTEMA TURI BE GALO DAUG**

***Programinis kodas***

%Gausas

function Gausas

clc, close all, clear all

A=[3 7 1 3;

1 -6 6 8;

4 4 -7 1;

-1 3 8 2]

b=[31;-5;33;-2]

n=size(A,1), nb=size(b,2)

A1=[A,b]

%Tiesioginis

for i=1:n-1

for j=i+1:n,

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

A1(j,i)=0;

end

A1

end

% Atvirkstinis

x=zeros(n,nb);

for i=n:-1:1

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

end

disp('sprendinys x='),x

disp('Patikrinimas:'),liekana=A\*x-b

disp('Paklaida:'),disp(norm(liekana)/norm(x))

end