MINISTERUL EDUCAŢIEI
Universitatea Tehnică a Moldovei
Facultatea Calculatoare Informatică și Microelectronică
Tehnologia Informației

Raport

Disciplina: Metode numerice

Lucrarea de laborator nr. 1

Tema: “REZOLVAREA NUMERICĂ A ECUAŢIILOR ALGEBRICE ŞI TRANSCENDENTE”

Varianta: 23

Student: ____________ **Raevschi Grigore TI-231**

Coordonator: ____________ **Conf. univ. Pațiuc Vladimir**

Chișinău 2024

Cuprins

Scopul lucrării

1. Să se separe toate rădăcinile reale ale ecuației f(x)=0 unde y=f(x) este o funcție reală de variabilă reală.
2. Să se determine o rădăcină reală a ecuației date cu ajutorul metodei înjumătățirii intervalului cu o eroare mai mică decât ε=10^{-2}
3. Să se precizeze rădăcina obținută cu exactitatea ε=10^{-6} utilizând
   - metoda aproximațiilor succesive
   - metoda tangentelor (Newton)
   - metoda secantelor.
4. Să se compare rezultatele luând în considerație numărul de iterații, evaluările pentru funcția și derivată.

Varianta: 23

$a)\ lg(2x+3)+2x-1$

$b)\ x^3+7x-2$

Funcția a

1) $f(c)-f(b)<0$

$y=\lg(2x+3)+2x-1=0$

$Calculăm\ pe\ intervalul\ [0;1]$

$f1=lg3-1;<0$

$f2=lg5+1;>0$

Din faptul că avem o expresie >0 rezultă că pe acest interval există soluții.

2) $f'(x)>0\ sau\ f'(x)<0$

$f'(x)=\frac{1}{2x+3}\ ln10+2; \qquad x\in[0;1]$

$2x+3>0; \qquad x\in[0;1] <=>$

$f'(x)>0\ ; \qquad \forall\ x\ \in[0;1]$

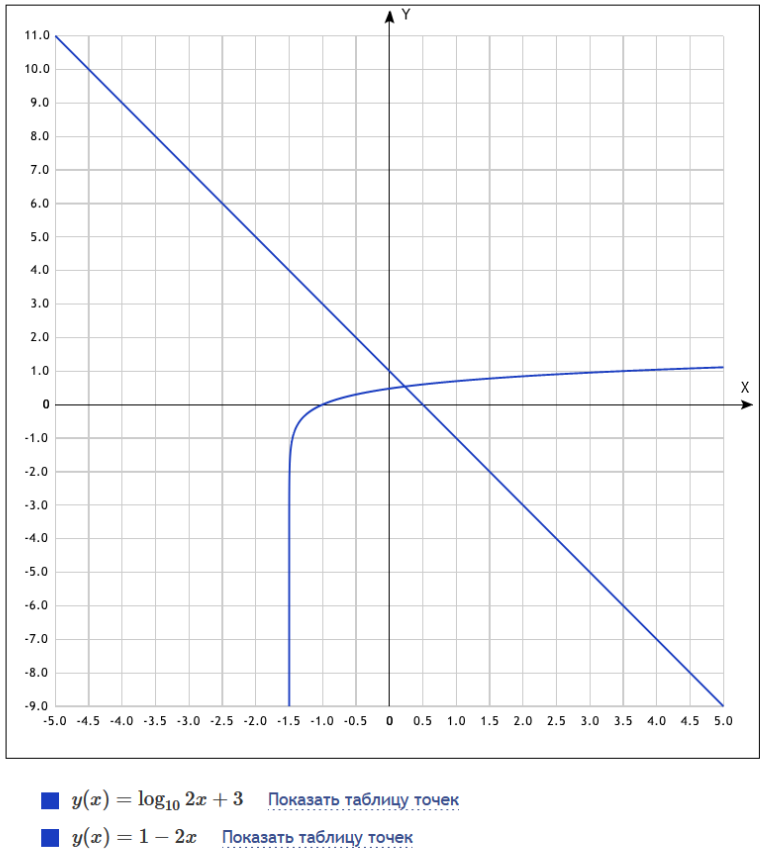

Figura 1*:Graficul funcției lg(2x+3)+2x-1*

Funcția b

$f(x) = x^3 + 7x - 2$

$f'(x) = 3x^2 + 7 = 0 \qquad x^2 = -\frac{7}{3} \qquad x_{1,2} = \pm i\sqrt{\frac{7}{3}}$

x_i	$a = 0$	$b = 1$
$y = f(x)$	-2	$+6$

Observăm că semnele diferă (+ și -) [0,1] $\exists!$

Aproximare succesivă $x^3 + 7x - 2 = 0 \qquad x^2 = \frac{2-x^2}{7} = \boldsymbol{\varphi(x)}$

$|\boldsymbol{\varphi}'(\boldsymbol{x})| \leq \boldsymbol{\alpha} < \mathbf{1}, \quad \boldsymbol{x} \in [0,1]$

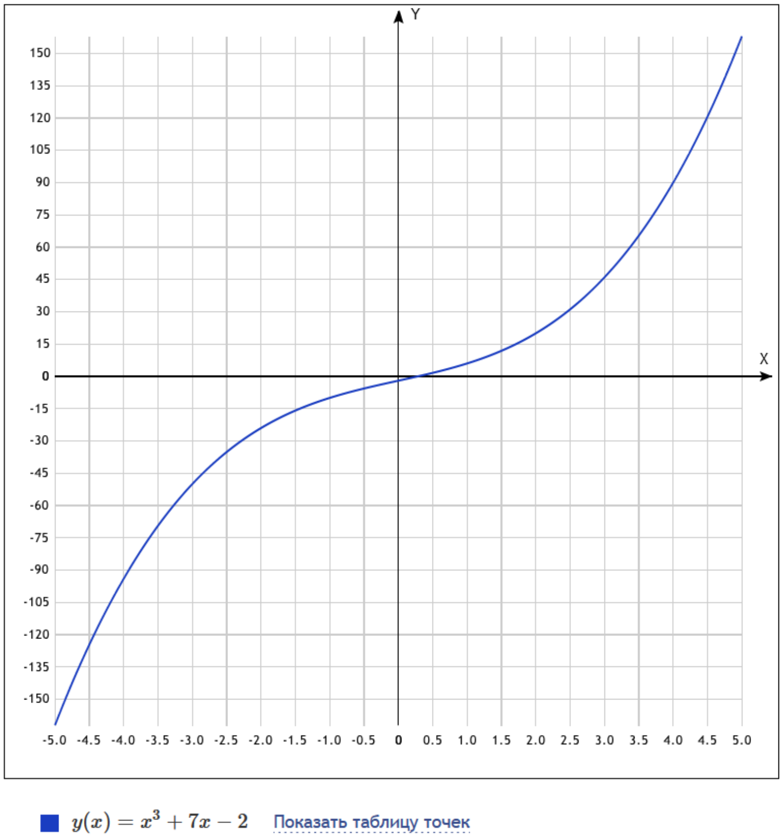

Figura 1*: Graficul funcției* $x^3 + 7x - 2$

Codul $lg(2x + 3) + 2x - 1$

```
#include <stdio.h>

#include <math.h>


#define epsilon 0.0001


int numar_iteratii = 0;


double functie(double x)

{

    return log10(2 * x + 3) + 2 * x - 1;

}


double functie_iterata(double x)

{

    return (log10(2 * x + 3) - 1) / -2;

}


double derivata_functie(double x)

{

    return 2 + 2 / (2 * x + 3);

}


double derivata_doua_functie(double x)

{

    return -2 / pow(2 * x + 3, 2);

}


double metoda_injumatatirii(double a, double b)

{

    double c;

    if (functie(a) * functie(b) < 0)

    {
```

```
        while (fabs(b - a) > epsilon)
        {
            c = (a + b) / 2;
            numar_iteratii++;
            printf("Pasul %d: a = %f, b = %f, c = %f\n", numar_iteratii, a, b, c);
            if (functie(a) * functie(c) < 0)
                b = c;
            else
                a = c;
        }
        return c;
    }
    else
    {
        return -1;
    }
}

double metoda_aproximarilor(double a)
{
    double x = a, y;
    do
    {
        y = functie_iterata(x);
        numar_iteratii++;
        printf("Pasul %d: x = %f, y = %f\n", numar_iteratii, x, y);
        if (fabs(y - x) <= epsilon)
            break;
        x = y;
    } while (1);
    return x;
}

double metoda_newton(double a)
{
```

```
    double x = a, x1;
    do
    {
        x1 = x - functie(x) / derivata_functie(x);
        numar_iteratii++;
        printf("Pasul %d: x = %f, x1 = %f\n", numar_iteratii, x, x1);
        if (fabs(x1 - x) <= epsilon)
            break;
        x = x1;
    } while (1);
    return x1;
}

double metoda_secantelor(double x0, double x1)
{
    double x2;
    do
    {
        x2 = x1 - (functie(x1) * (x1 - x0)) / (functie(x1) - functie(x0));
        numar_iteratii++;
        printf("Pasul %d: x0 = %f, x1 = %f, x2 = %f\n", numar_iteratii, x0, x1, x2);
        if (fabs(x2 - x1) <= epsilon)
            break;
        x0 = x1;
        x1 = x2;
    } while (1);
    return x2;
}

int main()
{
    double a = -1, b = 1;

    printf("Metoda injumatatirii intervalului:\n");
    double radacina_injumatatire = metoda_injumatatirii(a, b);
```

```
    if (radacina_injumatatire != -1)
    {
        printf("Rădăcina: (%f, 0)\n", radacina_injumatatire);
    }
    else
    {
        printf("Metoda injumatatirii nu a reusit.\n");
    }
    printf("Numar iteratii: %d\n", numar_iteratii);
    numar_iteratii = 0;

    printf("Metoda aproximarilor succesive:\n");
    printf("Rădăcina: %f\n", metoda_aproximarilor(a));
    printf("Numar iteratii: %d\n", numar_iteratii);
    numar_iteratii = 0;

    printf("Metoda Newton:\n");
    printf("Rădăcina: %f\n", metoda_newton(a));
    printf("Numar iteratii: %d\n", numar_iteratii);
    numar_iteratii = 0;

    printf("Metoda secantelor:\n");
    printf("Rădăcina: %f\n", metoda_secantelor(a, b));
    printf("Numar iteratii: %d\n", numar_iteratii);

    return 0;
}
```

```
Metoda injumatatirii intervalului:
Pasul 1: a = -1.000000, b = 1.000000, c = 0.000000
Pasul 2: a = 0.000000, b = 1.000000, c = 0.500000
Pasul 3: a = 0.000000, b = 0.500000, c = 0.250000
Pasul 4: a = 0.000000, b = 0.250000, c = 0.125000
Pasul 5: a = 0.125000, b = 0.250000, c = 0.187500
Pasul 6: a = 0.187500, b = 0.250000, c = 0.218750
Pasul 7: a = 0.218750, b = 0.250000, c = 0.234375
Pasul 8: a = 0.218750, b = 0.234375, c = 0.226563
Pasul 9: a = 0.226563, b = 0.234375, c = 0.230469
Pasul 10: a = 0.226563, b = 0.230469, c = 0.228516
Pasul 11: a = 0.228516, b = 0.230469, c = 0.229492
Pasul 12: a = 0.229492, b = 0.230469, c = 0.229980
Pasul 13: a = 0.229980, b = 0.230469, c = 0.230225
Pasul 14: a = 0.230225, b = 0.230469, c = 0.230347
Pasul 15: a = 0.230347, b = 0.230469, c = 0.230408
Rădăcina: (0.230408, 0)
Numar iteratii: 15
Metoda aproximarilor succesive:
Pasul 1: x = -1.000000, y = 0.500000
Pasul 2: x = 0.500000, y = 0.198970
Pasul 3: x = 0.198970, y = 0.234392
Pasul 4: x = 0.234392, y = 0.229911
Pasul 5: x = 0.229911, y = 0.230473
Pasul 6: x = 0.230473, y = 0.230403
Rădăcina: 0.230473
Numar iteratii: 6
Metoda Newton:
Pasul 1: x = -1.000000, x1 = -0.250000
Pasul 2: x = -0.250000, x1 = 0.143593
Pasul 3: x = 0.143593, x1 = 0.218730
Pasul 4: x = 0.218730, x1 = 0.228918
Pasul 5: x = 0.228918, x1 = 0.230221
Pasul 6: x = 0.230221, x1 = 0.230386
Pasul 7: x = 0.230386, x1 = 0.230407
Rădăcina: 0.230407
Numar iteratii: 7
Metoda secantelor:
Pasul 1: x0 = -1.000000, x1 = 1.000000, x2 = 0.276876
Pasul 2: x0 = 1.000000, x1 = 0.276876, x2 = 0.229513
Pasul 3: x0 = 0.276876, x1 = 0.229513, x2 = 0.230412
Pasul 4: x0 = 0.229513, x1 = 0.230412, x2 = 0.230410
Rădăcina: 0.230410
Numar iteratii: 4
PS C:\Users\grigo\Desktop\Metode numerice>
```

***Figura 2**: Rezultatul obținut a funcției lg(2x+3)+2x-1în program*

Codul $x^3 + 7x - 2$

```
#include <stdio.h>

#include <math.h>


#define epsilon 0.0001


int numar_iteratii = 0;


double functie(double x)

{

    return pow(x, 3) + 7 * x - 2;

}


double derivata_functie(double x)

{

    return 3 * pow(x, 3) + 7;

}


double metoda_injumatatirii(double a, double b)

{

    double c;

    if (functie(a) * functie(b) < 0)

    {

        while (fabs(b - a) > epsilon)

        {

            c = (a + b) / 2;

            numar_iteratii++;

            printf("Pasul %d: a = %f, b = %f, c = %f\n", numar_iteratii, a, b, c);

            if (functie(a) * functie(c) < 0)

                b = c;

            else

                a = c;

        }

        return c;
```

```
    }
    else
    {
        return -1;
    }
}

double metoda_newton(double a)
{
    double x = a, x1;
    do
    {
        x1 = x - functie(x) / derivata_functie(x);
        numar_iteratii++;
        printf("Pasul %d: x = %f, x1 = %f\n", numar_iteratii, x, x1);
        if (fabs(x1 - x) <= epsilon)
            break;
        x = x1;
    } while (1);
    return x1;
}

double metoda_secantelor(double x0, double x1)
{
    double x2;
    do
    {
        x2 = x1 - (functie(x1) * (x1 - x0)) / (functie(x1) - functie(x0));
        numar_iteratii++;
        printf("Pasul %d: x0 = %f, x1 = %f, x2 = %f\n", numar_iteratii, x0, x1, x2);
        if (fabs(x2 - x1) <= epsilon)
            break;
        x0 = x1;
        x1 = x2;
    } while (1);
```

```
    return x2;
}

int main()
{
    double a = -1, b = 1;

    printf("Metoda injumatatirii intervalului:\n");
    double radacina_injumatatire = metoda_injumatatirii(a, b);
    if (radacina_injumatatire != -1)
    {
        printf("Rădăcina: (%f, 0)\n", radacina_injumatatire);
    }
    else
    {
        printf("Metoda injumatatirii nu a reusit.\n");
    }
    printf("Numar iteratii: %d\n", numar_iteratii);
    numar_iteratii = 0;

    printf("Metoda Newton:\n");
    printf("Rădăcina: %f\n", metoda_newton(a));
    printf("Numar iteratii: %d\n", numar_iteratii);
    numar_iteratii = 0;

    printf("Metoda secantelor:\n");
    printf("Rădăcina: %f\n", metoda_secantelor(a, b));
    printf("Numar iteratii: %d\n", numar_iteratii);

    return 0;
}
```

```
Metoda injumatatirii intervalului:
Pasul 1: a = -1.000000, b = 1.000000, c = 0.000000
Pasul 2: a = 0.000000, b = 1.000000, c = 0.500000
Pasul 3: a = 0.000000, b = 0.500000, c = 0.250000
Pasul 4: a = 0.250000, b = 0.500000, c = 0.375000
Pasul 5: a = 0.250000, b = 0.375000, c = 0.312500
Pasul 6: a = 0.250000, b = 0.312500, c = 0.281250
Pasul 7: a = 0.281250, b = 0.312500, c = 0.296875
Pasul 8: a = 0.281250, b = 0.296875, c = 0.289063
Pasul 9: a = 0.281250, b = 0.289063, c = 0.285156
Pasul 10: a = 0.281250, b = 0.285156, c = 0.283203
Pasul 11: a = 0.281250, b = 0.283203, c = 0.282227
Pasul 12: a = 0.282227, b = 0.283203, c = 0.282715
Pasul 13: a = 0.282227, b = 0.282715, c = 0.282471
Pasul 14: a = 0.282471, b = 0.282715, c = 0.282593
Pasul 15: a = 0.282471, b = 0.282593, c = 0.282532
Rădăcina: (0.282532, 0)
Numar iteratii: 15
Metoda Newton:
Pasul 1: x = -1.000000, x1 = 1.500000
Pasul 2: x = 1.500000, x1 = 0.806569
Pasul 3: x = 0.806569, x1 = 0.320142
Pasul 4: x = 0.320142, x1 = 0.281569
Pasul 5: x = 0.281569, x1 = 0.282516
Pasul 6: x = 0.282516, x1 = 0.282493
Rădăcina: 0.282493
Numar iteratii: 6
Metoda secantelor:
Pasul 1: x0 = -1.000000, x1 = 1.000000, x2 = 0.250000
Pasul 2: x0 = 1.000000, x1 = 0.250000, x2 = 0.278195
Pasul 3: x0 = 0.250000, x1 = 0.278195, x2 = 0.282509
Pasul 4: x0 = 0.278195, x1 = 0.282509, x2 = 0.282494
Rădăcina: 0.282494
Numar iteratii: 4
PS C:\Users\grigo\Desktop\Metode numerice>
```

***Figura 3**: Rezultatul obținut a funcției $x^3 + 7x - 2$ în program*

Concluzie

În urma efectuării acestei lucrări de laborator s-a determinat soluția ecuației transcendente f(x) =0, prin precizarea rădăcinii obținute cu exactitatea ε utilizând cele $\varepsilon = 10 - 4$, metode: înjumătățirii intervalului, aproximărilor succesive, Newton și secantelor. În cazul ambelor ecuații metoda Newton s-a dovedit a fi mai eficientă, obținându-se o soluție exactă și cu o complexitatea în timp mai mică.