Міністерство освіти і науки України
Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського» Інститут прикладного системного аналізу

Лабораторна робота № 5
з курсу «Чисельні методи 1»
з теми «Інтерполювання фцнкцій»
Варіант № 16

Виконав студент 2 курсу групи КА-91
Панченко Єгор Станіславович
перевірила старший викладач
Хоменко Ольга Володимирівна

Київ-2021

Завдання: дана функція $f(x) = x + \frac{8}{1+e^{\frac{x}{4}}}$. Побудувати інтерполяційний многочлен Лагранжа, перший та другий інтерполяційні поліноми Ньютона.

1. **Текст програм**

```
#include <vector>
#include <math.h>
#include <cstdio>
#include <assert.h>

using namespace std;

namespace Laba5 {
    double f(double x) {
        return (x + 8.0 / (1 + exp(x / 4)));
    }


}

double gnLangrange(double x, vector < double > points) {
    double res = 0;
    for (int i = 0; i < points.size(); i++) {

        double mlt = 1;
        for (int j = 0; j < points.size(); j++) {

            if (i == j) {
                continue;
            }

            mlt *= (x - points[j]) / (points[i] - points[j]);
        }

        res += Laba5::f(points[i]) * mlt;
    }

    return res;
}

void PolLangrange() {

    printf("Interval is [-10, 10]\n");
    printf("Langrange\n");

    vector < double > points = {-4, -2, 0, 2, 4};

    vector < double > another_points = {-10, -5, -3, -1, 1, 3, 5, 10};

    printf("Values in points: f g\n");
    for (auto i : points) {
        printf("\t%10.5lf %10.5lf %10.5lf\n", i, Laba5::f(i), gnLangrange(i,
points));
    }
```

```
    printf("Values in another_points: f g\n");
    for (auto i : another_points) {
        printf("\t%10.5lf %10.5lf %10.5lf\n", i, Laba5::f(i), gnLangrange(i,
points));
    }

}

double diff(vector < double > x) {
    double res = 0;
    for (int j = 0; j < x.size(); j++) {

        double dn = 1;
        for (int i = 0; i < x.size(); i++) {

            if (i == j) {
                continue;
            }
            dn *= x[j] - x[i];
        }

        res += Laba5::f(x[j]) / dn;
    }

    return res;
}

double delta_y(double i, double k, vector < double > &points) {
    if (k == 0 /*&& i >= 0 && i < points.size()*/) {
        return Laba5::f(points[i]);
    }
    /*if (i < 0 || i >= points.size()) {
        return 0;
    }*/
    return (delta_y(i + 1, k - 1, points) - delta_y(i, k - 1, points));
}

double gnNewtonLection(double x, vector < double > points) {
    /*for (int i = 0; i <= 5; i++) {
        printf("%lf\n", delta_y(0, i, points));
    }*/
    double res = 0;
    double mlt = 1;

    for (int i = 0; i < points.size(); i++) {
        res += mlt * delta_y(0, i, points);
        if (i + 1 == points.size()) {
            break;
        }
        mlt *= x - points[i];
        mlt /= i + 1;
        mlt /= 2;
    }

    return res;
}
```

```
double gnNewtonLectionSecond(double x, vector < double > points) {
    /*for (int i = 0; i <= 5; i++) {
        printf("%lf\n", delta_y(0, i, points));
    }*/
    double res = 0;
    double mlt = 1;

    for (int i = 0; i < points.size(); i++) {
        res += mlt * delta_y(points.size() - 1 - i, i, points);
        if (i + 1 == points.size()) {
            break;
        }
        mlt *= x - points[points.size() - 1 - i];
        mlt /= i + 1;
        mlt /= 2;
    }

    return res;
}

double gnNewton(double x, vector < double > points) {
    double res = 0;

    vector < double > curr;
    double mlt = 1;
    for (int i = 0; i < points.size(); i++) {

        curr.push_back(points[i]);
        res += diff(curr) * mlt;
        if (i == points.size()) {
            break;
        }
        mlt *= x - points[i];

    }

    return res;
}

void PolNewtonFirst() {
    printf("Interval is [-10, 10]\n");
    printf("Newton\n");

    vector < double > points = {-4, -2, 0, 2, 4};

    vector < double > another_points = {-10, -5, -3, -1, 1, 3, 5, 10};

    printf("Values in points: f g\n");
    for (auto i : points) {
        printf("\t%10.5lf %10.5lf %10.5lf\n", i, Laba5::f(i), gnNewtonLection(i,
points));
    }

    printf("Values in another_points: f g\n");
    for (auto i : another_points) {
        printf("\t%10.5lf %10.5lf %10.5lf\n", i, Laba5::f(i), gnNewtonLection(i,
points));
```

```
    }
}

void PolNewtonSecond() {
    printf("Interval is [-10, 10]\n");
    printf("Newton\n");

    vector < double > points = {-4, -2, 0, 2, 4};

    vector < double > another_points = {-10, -5, -3, -1, 1, 3, 5, 10};

    printf("Values in points: f g\n");
    for (auto i : points) {
        printf("\t%10.5lf %10.5lf %10.5lf\n", i, Laba5::f(i),
gnNewtonLectionSecond(i, points));
    }

    printf("Values in another_points: f g\n");
    for (auto i : another_points) {
        printf("\t%10.5lf %10.5lf %10.5lf\n", i, Laba5::f(i),
gnNewtonLectionSecond(i, points));
    }
}
```

2. Графіки поліномів

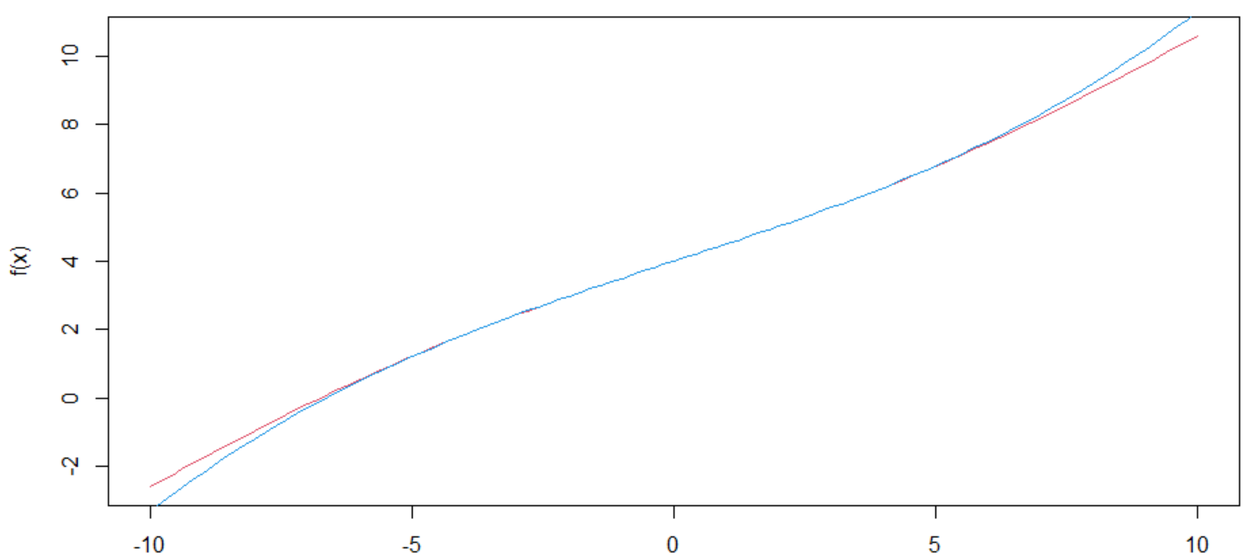
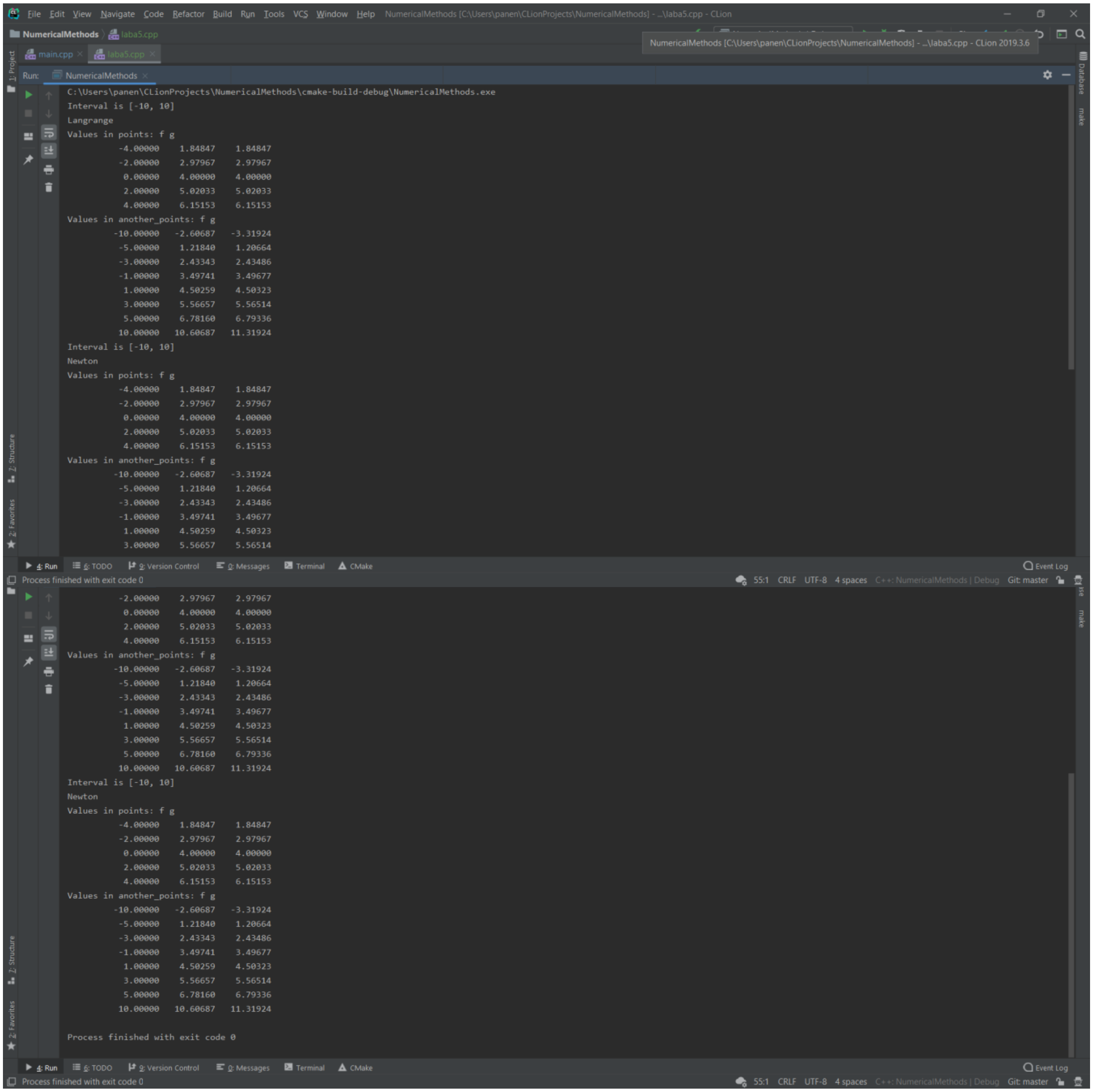


Рис.3.3 1 Червона крива - f(x); Синя крива - інтерполяційні многочлени

3. Результат роботи програми



4. Висновок

У ході виконання лабораторної роботи було програмно реалізовано метод побудови інтерполяційного поліному Лагранжа, першого та другого інтерполяційних многочленів Ньютона.

Таблиця скінченних різниць побудована не була – замість неї реалізовано функцію $delta\_y()$, яка дозволяє вирахувати скінченну різницю k-го порядку з індексом i лише за числами iта k.

З графіків можна зробити висновок, що функція дуже гарно апроксимується в околі нуля. При більших значеннях функції починають відрізнятися все більше і більше.

Письмова частина

При обрахунках було допущено округлення з метою спрощення розрахунків. Виявилося, що кубічні сплайни та інтерполяційний многочлен Лагранжа приймають один й той самий вигляд.

x	-2	0	2
y	2.97	4	5.02

$$y = x + \frac{8}{1+e^{x/4}}$$

Будуємо:

$$g(x) = \begin{cases} a_1 + b_1\cdot(x) + c_1\cdot x^2 + d_1\cdot x^3, & x\in[-2;0] : g_1 \\ a_2 + b_2\cdot(x-2) + c_2\cdot(x-2)^2 + d_2\cdot(x-2)^3, & x\in[0;2] : g_2 \end{cases}$$

Має виконуватися система

$$\begin{cases} g_1(-2) = 2.97 \\ g_1(0) = 4 \\ g_2(0) = 4 \\ g_2(2) = 5.02 \\ g_2'(0) = g_1'(0) \quad ; g_1''(-2) = 0 \\ g_2''(0) = g_1''(0) \quad g_2''(2) = 0 \end{cases}$$

$$g_1' = 3d_1x^2 + 2c_1x + b_1$$
$$g_1'' = 6d_1x + 2c_1$$
$$g_2' = 3d_2(x-2)^2 + 2c_2(x-2) + b_2$$
$$g_2'' = 6d_2(x-2) + 2c_2$$

Для спрощення розрахунків приймемо $g_1(-2)=3$ і $g_2(2)=5$. Тоді:

$$\begin{cases} a_1 - 2b_1 + 4c_1 - 8d_1 = 3 \\ a_1 = 4 \\ a_2 - 2b_2 + 4c_2 - 8d_2 = 4 \\ a_2 = 5 \\ b_1 = 12d_2 - 4c_2 + b_2 \\ -12d_2 + 2c_2 = 2c_1 \\ -12d_1 + 2c_1 = 0 \\ 2c_2 = 0 \end{cases} \Rightarrow \begin{cases} a_1 = 4 \;;\; c_2 = 0 \\ a_2 = 5 \;;\; c_1 = 6d_1 \\ 2b_1 - 4c_1 + 8d_1 = 1 \\ 2b_2 - 4c_2 + 8d_2 = 1 \\ b_1 = 12d_2 - 4c_2 + b_2 \\ c_1 = c_2 - 6d_2 \end{cases} \Rightarrow$$

$$\Rightarrow \begin{cases} 2(12d_2 - 4c_2 + b_2) - 4(c_2 - 6d_2) + 8d_1 = 1 \\ 2b_2 - 4c_2 + 8d_2 = 1 \\ c_2 - 6d_2 = 6d_1 \end{cases} \Rightarrow \begin{cases} 36d_2 \cancel{-12c_2} + 2b_2 + 8d_1 = 1 \\ 2b_2 + 8d_2 = 1 \\ c_2 - 6d_2 = 6d_1 \end{cases}$$

$a_1=4; a_2=5; c_2=0$

$$\Rightarrow \begin{cases} 2b_1 - 4c_1 + 8d_1 = 1 \\ 2b_2 + 8d_2 = 1 \\ b_1 = 12d_2 + b_2 \\ c_1 = -6d_2 \\ c_1 = 6d_1 \end{cases} \Rightarrow d_1 = -d_2 \quad \Rightarrow \begin{cases} 2b_1 - 24d_1 + 8d_1 = 1 \\ 2b_2 - 8d_1 = 1 \\ b_1 = -12d_1 + b_2 \\ d_1 = -d_2 \end{cases} \Rightarrow \begin{cases} -24d_1 + 2b_2 - 24d_1 + 8d_1 = 1 \\ 2b_2 - 8d_1 = 1 \end{cases}$$

$$\begin{cases} -40d_1 + 2b_2 = 1 \\ 2b_2 - 8d_1 = 1 \end{cases} \Rightarrow \begin{cases} d_1 = 0 \\ b_2 = 1/2 \\ d_2 = 0 \\ b_1 = 1/2 \\ c_2 = 0 \\ c_1 = 0 \\ a_1 = 4 \\ a_2 = 5 \end{cases} \Rightarrow$$

$$\Rightarrow \boxed{\begin{cases} g_1 = 4 + \frac{x}{2} \\ g_2 = 5 + \frac{1}{2}(x-2) \end{cases}}$$

$x/2 + 4$

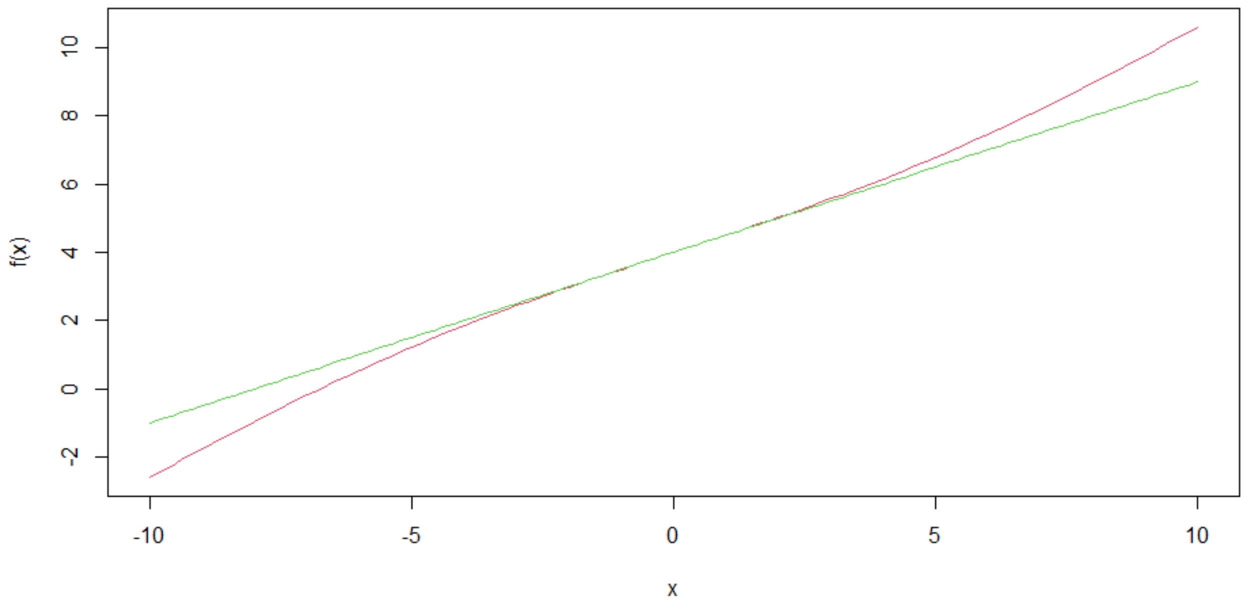
Интерполяционный многочлен Лагранжа:

x	-2 (x_0)	0 (x_1)	2 (x_2)
y	3 (y_0)	4 (y_1)	5 (y_2)

$$L = \frac{x\cdot(x-2)}{-2\cdot -4}\cdot 3 + \frac{(x+2)(x-2)}{2\cdot -2}\cdot 4 + \frac{(x+2)\cdot x}{4\cdot 2}\cdot 5 =$$

$$= \frac{3}{8}(x^2-2x) - (x^2-4) + \frac{5}{8}(x^2+2x) = x^2 - x^2 - \frac{6x}{8} + \frac{10x}{8} + 4 =$$

$$\boxed{= \frac{x}{2} + 4}$$


f(x)
-2
0
2
4
6
8
10
-10
-5
0
5
10
x