Федеральное государственное автономное образовательное учреждение высшего образования «Национальный исследовательский университет ИТМО»

Факультет программной инженерии и компьютерной техники

Лабораторная работа №5

По Вычислительной математике

Вариант №9

Выполнил:

Ступин Тимур Русланович

Группа № P3108

Поток № 1.3

Преподаватель:

Санкт-Петербург 2025

Содержание

1 Цель работы

Решить задачу интерполяции, найти значения функции при заданных значениях аргумента, отличных от узловых точек.

2 Порядок выполнения работы

2.1 Вычислительная реализация задачи

Выберем таблицу $y = f(x)$

	x	y	№ варианта	X_1	X_2
Таблица 1.4	1.05	0.1213	9	1.562	1.362
	1.15	1.1316			
	1.25	2.1459			
	1.35	3.1565			
	1.45	4.1571			
	1.55	5.1819			
	1.65	6.1969			

Построим таблицу конечных разностей:

x_i	y_i	$\Delta^1 y_i$	$\Delta^2 y_i$	$\Delta^3 y_i$	$\Delta^4 y_i$	$\Delta^5 y_i$	$\Delta^6 y_i$
1.05	0.1213	1.0103	0.0040	-0.0077	0.0014	0.0391	-0.1478
1.15	1.1316	1.0143	-0.0037	-0.0063	0.0405	-0.1087	
1.25	2.1459	1.0106	-0.0100	0.0342	-0.0682		
1.35	3.1565	1.0006	0.0242	-0.0340			
1.45	4.1571	1.0248	-0.0098				
1.55	5.1819	1.0150					
1.65	6.1969						

Вычислим значения функции для аргумента X_1, используя первую или вторую интерполяционную формулу Ньютона:

Воспользуемся формулой Ньютона для интерполирования назад, так как $X_1 = 1.562$ лежит в правой половине отрезка

Для $X_1 = 1.562$ получаем:

$$t = \frac{(x - x_n)}{h} = \frac{1.562 - 1.65}{0.1} = -0.88$$

Интерполяционная формула:

$$\begin{gathered}
N_6(x) = \\
y_6 \\
+t\Delta^1 y_5 \\
+\frac{t(t+1)}{2!}\Delta^2 y_4 \\
+\frac{t(t+1)(t+2)}{3!}\Delta^3 y_3 \\
+\frac{t(t+1)(t+2)(t+3)}{4!}\Delta^4 y_2 \\
+\frac{t(t+1)(t+2)(t+3)(t+4)}{5!}\Delta^5 y_1 \\
+\frac{t(t+1)(t+2)(t+3)(t+4)(t+5)}{6!}\Delta^6 y_0
\end{gathered}$$

Подставляя значения получаем:

$$\begin{gathered}
y(1.562) = \\
6.1969 \\
+ -0.88 \cdot 1.0150 \\
+\frac{-0.88(-0.88+1)}{2!} \cdot -0.0098 \\
+\frac{-0.88(-0.88+1)(-0.88+2)}{3!} \cdot -0.0340 \\
+\frac{-0.88(-0.88+1)(-0.88+2)(-0.88+3)}{4!} \cdot -0.0682 \\
+\frac{-0.88(-0.88+1)(-0.88+2)(-0.88+3)(-0.88+4)}{5!} \cdot -0.1087 \\
+\frac{-0.88(-0.88+1)(-0.88+2)(-0.88+3)(-0.88+4)(-0.88+5)}{6!} \cdot -0.1478 \\
\approx 5.2993
\end{gathered}$$

Вычислить значения функции для аргумента X_2, используя первую или вторую интерполяционную формулу Гаусса:

Центральная точка:

$$\alpha = 1.35$$

Так как:

$$X_2 = 1.362 > 1.35$$

используем первую интерполяционную формулу Гаусса.

Для $X_2 = 1.362$ получаем:

$$t = \frac{(x - x_0)}{h} = \frac{1.362 - 1.35}{0.1} = 0.12$$

Интерполяционная формула:

$$P_6(x) =$$
$$y_0$$
$$+t\Delta^1 y_0$$
$$+\frac{t(t+1)}{2!}\Delta^2 y_{-1}$$
$$+\frac{t(t+1)(t+2)}{3!}\Delta^3 y_{-1}$$
$$+\frac{t(t+1)(t+2)(t+3)}{4!}\Delta^4 y_{-2}$$
$$+\frac{t(t+1)(t+2)(t+3)(t+4)}{5!}\Delta^5 y_{-2}$$
$$+\frac{t(t+1)(t+2)(t+3)(t+4)(t+5)}{6!}\Delta^6 y_{-3}$$

Подставляя значения получаем:

$$
\begin{gathered}
y(1.362) = \\
3.1565 \\
+0.12 \cdot 1.0006 \\
+\frac{0.12(0.12+1)}{2!} \cdot -0.0100 \\
+\frac{0.12(0.12+1)(0.12+2)}{3!} \cdot 0.0342 \\
+\frac{0.12(0.12+1)(0.12+2)(0.12+3)}{4!} \cdot 0.0405 \\
+\frac{0.12(0.12+1)(0.12+2)(0.12+3)(0.12+4)}{5!} \cdot -0.1087 \\
+\frac{0.12(0.12+1)(0.12+2)(0.12+3)(0.12+4)(0.12+5)}{6!} \cdot -0.1478 \\
\approx 3.2767
\end{gathered}
$$

2.2 Программная реализация задачи

2.2.1 Листинг программы

```
def lagrange_polynomial(x, y):
    n = len(x)

    def p(x_):
        result = 0
        for k in range(n):
            nominator = 1
            denominator = 1
            for i in range(n):
                if i == k:
                    continue
                nominator *= x_ - x[i]
                denominator *= x[k] - x[i]
            result += y[k] * (nominator / denominator)
        return result

    return p


def newton_divided_difference_polynomial(x, y):
    n = len(x)
    diffs = calculate_divided_differences(x, y)

    def p(x_):
        result = y[0]
```

```
        for k in range(1, n):
            d = diffs[k]
            for i in range(0, k):
                d *= (x_ - x[i])
            result += d
        return result

    return p


def _first_gauss_polynomial(x, y):
    if len(x) <= 1:
        raise Exception('Должно быть минимум две точки')

    n = len(x)
    h = x[1] - x[0]
    alpha_ind = n // 2
    diffs = calculate_finite_difference_table(y)

    def p(x_):
        t = (x_ - x[alpha_ind]) / h
        result = 0

        for k in range(n):
            m = (k + 1) // 2

            nominator = 1
            for j in range(-(m - 1), m):
                nominator *= t + j
            if k == 2 * m and m != 0: nominator *= t - m

            factorial = 1
            for j in range(1, k + 1):
                factorial *= j

            if k == 2 * m:
                result += diffs[alpha_ind - m][k] * (nominator /
                 ↪  factorial)
            else:
                result += diffs[alpha_ind - (m - 1)][k] * (nominator /
                 ↪  factorial)
        return result

    return p


def _second_gauss_polynomial(x, y):
    if len(x) <= 1:
        raise Exception('Должно быть минимум две точки')
```

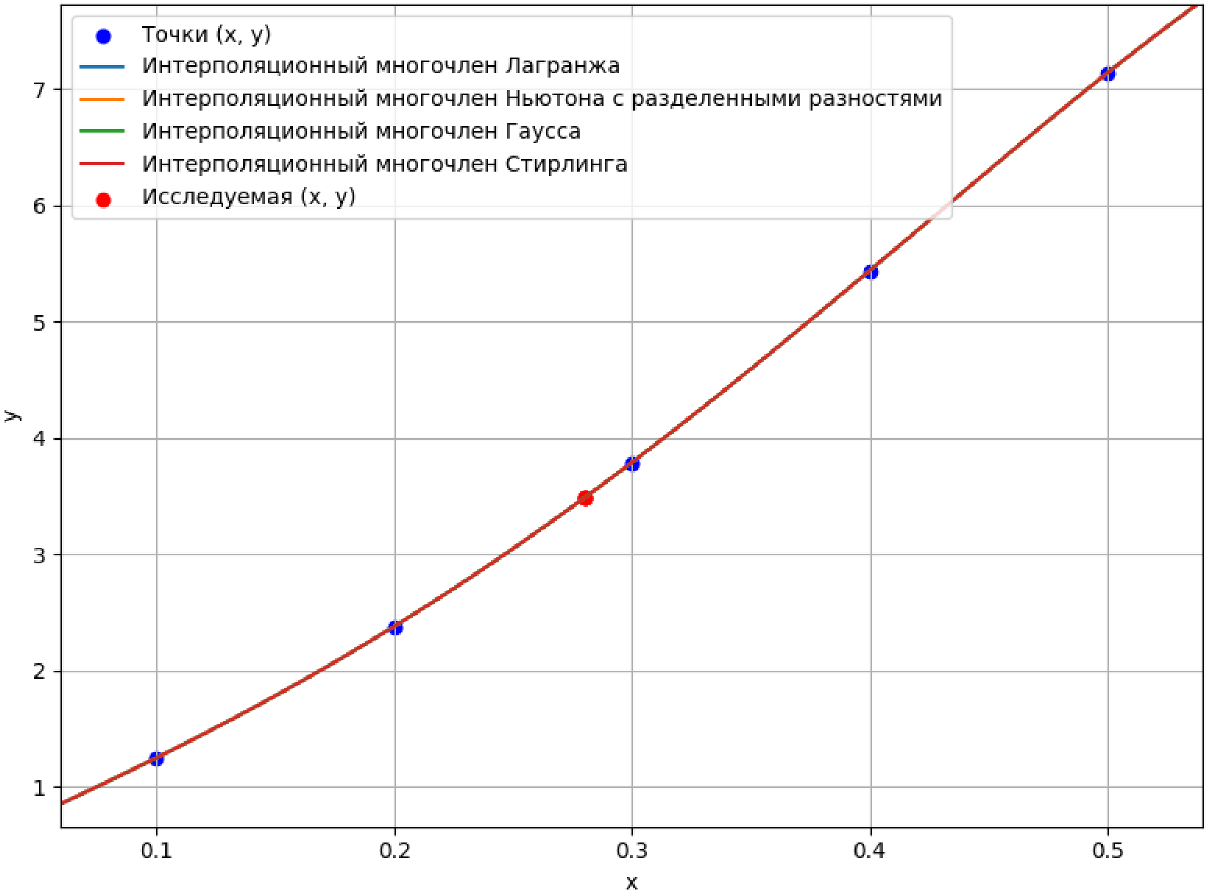
```
    n = len(x)
    h = x[1] - x[0]
    alpha_ind = n // 2
    diffs = calculate_finite_difference_table(y)

    def p(x_):
        t = (x_ - x[alpha_ind]) / h
        result = 0

        for k in range(n):
            m = (k + 1) // 2

            nominator = 1
            for j in range(-(m - 1), m):
                nominator *= t + j
            if k == 2 * m and m != 0: nominator *= t + m

            factorial = 1
            for j in range(1, k + 1):
                factorial *= j

            result += diffs[alpha_ind - m][k] * (nominator / factorial)
        return result

    return p


def gauss_polynomial(x, y):
    if len(x) <= 1:
        raise Exception('Должно быть минимум две точки')
    if not is_finite_difference(x):
        raise Exception('Значения X должны иметь фиксированный шаг!')

    n = len(x)
    alpha_ind = n // 2

    p1 = _first_gauss_polynomial(x, y)
    p2 = _second_gauss_polynomial(x, y)

    p = lambda x_: p1(x_) if x_ > x[alpha_ind] else p2(x_)

    return p


def stirling_polynomial(x, y):
    if len(x) <= 1:
        raise Exception('Должно быть минимум две точки')
    if len(x) % 2 != 1:
        raise Exception('Число узлов должно быть нечетным')
    if not is_finite_difference(x):
```

```
        raise Exception('Значения X должны иметь фиксированный шаг!')

    p1 = _first_gauss_polynomial(x, y)
    p2 = _second_gauss_polynomial(x, y)

    p = lambda x_: (p1(x_) + p2(x_)) / 2

    return p


def bessel_polynomial(x, y):
    if len(x) <= 1:
        raise Exception('Должно быть минимум две точки')
    if len(x) % 2 != 0:
        raise Exception('Число узлов должно быть чётным')
    if not is_finite_difference(x):
        raise Exception('Значения X должны иметь фиксированный шаг')

    n = len(x)
    h = x[1] - x[0]
    alpha_ind = n // 2 - 1
    diffs = calculate_finite_difference_table(y)

    def p(x_):
        t = (x_ - (x[alpha_ind] + x[alpha_ind + 1]) / 2) / h
        result = (y[alpha_ind] + y[alpha_ind + 1]) / 2
        result += (t - 0.5) * diffs[alpha_ind][1]
        if alpha_ind > 0:
            result += (t * (t - 1) / (1 * 2)) * ((diffs[alpha_ind - 1][2]
             ↪  + diffs[alpha_ind][2]) / 2)
            result += ((t - 0.5) * t * (t - 1) / (1 * 2 * 3)) *
             ↪  diffs[alpha_ind - 1][3]
        if alpha_ind > 1:
            result += ((t * (t - 1) * (t + 1) * (t - 2) / (1 * 2 * 3 * 4))
             ↪  *
                        ((diffs[alpha_ind - 2][4] + diffs[alpha_ind -
                         ↪  1][4]) / 2))
        return result

    return p
```

2.2.2 Результаты выполнения программы

```
Выберите способ задания функции:
1 -> Консоль
2 -> Файл
3 -> Функция
1
Вводите точки, по одной в строке. По окончании ввода введите q
```

```
0.1 1.25
0.2 2.38
0.3 3.79
0.4 5.44
0.5 7.14
q
Введите точку интерполяции: 0.28
Таблица конечных разностей:
1.2500        1.1300        0.2800        -0.0400        -0.1500
2.3800        1.4100        0.2400        -0.1900
3.7900        1.6500        0.0500
5.4400        1.7000
7.1400

==================================================

Интерполирующая функция: Интерполяционный многочлен Лагранжа
P(0.28) = 3.485360000000001

==================================================

Интерполирующая функция: Интерполяционный многочлен Ньютона с разделенными
↪  разностями
P(0.28) = 3.485360000000001

==================================================

Интерполирующая функция: Интерполяционный многочлен Гаусса
P(0.28) = 3.4853600000000005

==================================================

Интерполирующая функция: Интерполяционный многочлен Стирлинга
P(0.28) = 3.4853600000000005

==================================================

Интерполирующая функция: Интерполяционный многочлен Бесселя
ОШИБКА: Число узлов должно быть чётным

==================================================
```



3 Вывод

В ходе работы я изучил основные методы интерполяции: метода Лагранжа, методы Ньютона, метода Гаусса, а также Стирлинга и Бесселя. Также я реализовал программу, которая используя данные методы решает задачу интерполяции функции по точкам и строит её график. В программе реализованы методы интерполяции Ньютона, Гаусса, Лагранжа, Бесселя и Стирлинга. Программы была запущена на различных входных данных, после чего результаты были проанализированы. В процессе проверки было обнаружено, что методы Бесселя и Стирлинга очень чувствительны к проверяемой точке и работают хорошо только для точек приближенных к центру. Это связано с быстрым ростом ошибки при использовании этих формул. Однако при взятии близких к центру точек они обеспечивают большую точность чем другие рассмотренные методы.