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

Мегафакультет компьютерных технологий и управления
Факультет программной инженерии и компьютерной техники

Отчёт
по лабораторной работе №5
по дисциплине «Вычислительная математика»

Вариант №4

Группа: P3218

Студент: Горло Евгений Николаевич

Преподаватель: Бострикова Дарья Константиновна

Санкт-Петербург
2024

Содержание

1 Цель

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

2 Задание

Программная реализация

1. Исходные данные задаются тремя способами:
   (a) в виде набора данных (таблицы x, y), пользователь вводит значения с клавиатуры;
   (b) в виде сформированных в файле данных (подготовить не менее трех тестовых вариантов);
   (c) на основе выбранной функции, из тех, которые предлагает программа, например, $\sin x$. Пользователь выбирает уравнение, исследуемый интервал и количество точек на интервале (не менее двух функций).
2. Сформировать и вывести таблицу конечных разностей;
3. Вычислить приближенное значение функции для заданного значения аргумента, введенного с клавиатуры, указанными методами (см. табл. 2). Сравнить полученные значения;
4. Построить графики заданной функции с отмеченными узлами интерполяции и интерполяционного многочлена Ньютона/Гаусса (разными цветами);
5. Программа должна быть протестирована на различных наборах данных, в том числе и некорректных;
6. Проанализировать результаты работы программы.

Методы для реализации в программе по варианту:

1. Многочлен Лагранжа;
2. Многочлен Ньютона с разделенными разностями;
3. Многочлен Ньютона с конечными разностями;

Вычислительная реализация

1. Выбрать из табл. 1 заданную по варианту таблицу $y = f(x)$ (таблица 1.1 – таблица 1.5);

2. Построить таблицу конечных разностей для заданной таблицы. Таблицу отразить в отчете;

3. Вычислить значения функции для аргумента X_1 (см. табл.1), используя первую или вторую интерполяционную формулу Ньютона. Обратить внимание какой конкретно формулой необходимо воспользоваться;

4. Вычислить значения функции для аргумента X_2 (см. табл. 1), используя первую или вторую интерполяционную формулу Гаусса. Обратить внимание какой конкретно формулой необходимо воспользоваться;

5. Подробные вычисления привести в отчете.

Таблица по варианту:

x	y
1,05	0,1213
1,15	1,1316
1,25	2,1459
1,35	3,1565
1,45	4,1571
1,55	5,1819
1,65	6,1969

$X_1 = 1,051$

$X_2 = 1,277$

3 Рабочие формулы

Многочлен Лагранжа

$$L_n(x) = \sum_{i=0}^{n} y_i \prod_{j=0, j\neq i}^{n} \frac{x - x_j}{x_i - x_j}$$

Первая интерполяционная формула Ньютона:

$$N_n(x) = y_0 + t\Delta y_0 + \frac{t(t-1)}{2!}\Delta^2 y_0 + ... + \frac{t(t-1)...(t-n+1)}{n!}\Delta^n y_0$$

Вторая интерполяционная формула Ньютона:

$$N_n(x) = y_n + t\Delta y_{n-1} + \frac{t(t+1)}{2!}\Delta^2 y_{n-2} + ... + \frac{t(t+1)...(t+n-1)}{n!}\Delta^n y_0$$

Первая интерполяционная формула Гаусса:

$$P_n(x) = y_0 + t\Delta y_0 + \frac{t(t-1)}{2!}\Delta^2 y_{-1} + \frac{(t+1)t(t-1)}{3!}\Delta^3 y_{-1} + \quad ... \quad +$$
$$+ \frac{(t+n-1)...(t-n+1)}{(2n-1)!}\Delta^{2n-1} y_{-(n-1)} + \frac{(t+n-1)...(t-n)}{(2n)!}\Delta^{2n} y_{-n}$$

Вторая интерполяционная формула Гаусса:

$$P_n(x) = y_0 + t\Delta y_0 + \frac{t(t-1)}{2!}\Delta^2 y_{-1} + \frac{(t+1)t(t-1)}{3!}\Delta^3 y_{-1} + \quad ... \quad +$$
$$+ \frac{(t+n-1)...(t-n+1)}{(2n-1)!}\Delta^{2n-1} y_{-(n-1)} + \frac{(t+n)(t+n-1)...(t-n+1)}{(2n)!}\Delta^{2n} y_{-n}$$

4 Вычислительная часть

Таблица конечных разностей

i	x_i	y_i	Δy_i	$\Delta^2 y_i$	$\Delta^3 y_i$	$\Delta^4 y_i$	$\Delta^5 y_i$	$\Delta^6 y_i$
0	1,05	0,1213	1,0103	0,0040	-0,0077	0,0014	0,0391	-0,1478
1	1,15	1,1316	1,0143	-0,0037	-0,0063	0,0405	-0,1087	0
2	1,25	2,1459	1,0106	-0,0100	0,0342	-0,0682	0	0
3	1,35	3,1565	1,0006	0,0242	-0,0340	0	0	0
4	1,45	4,1571	1,0248	-0,0098	0	0	0	0
5	1,55	5,1819	1,0150	0	0	0	0	0
6	1,65	6,1969	0	0	0	0	0	0

Интерполяционная формула Ньютона для равноотстоящих узлов
$X_1 = 1,051$

$$t = \frac{x - x_0}{h} = \frac{1,051 - 1,05}{0,1} = 0,1$$

$$N_n(x) = y_0 + t\Delta y_0 + \frac{t(t-1)}{2!}\Delta^2 y_0 + \cdots + \frac{t(t-1)\ldots(t-n+1)}{n!}\Delta^n y_0$$

$$N_6(1.051) = 0.1213 + 0.01 \cdot 1.0103 + \frac{0.01(0.01-1)}{2!} \cdot 0.0040$$

$$+\frac{0.01(0.01-1)(0.01-2)}{3!} \cdot (-0.0077)$$

$$N_6(1.051) \approx 0.1314$$

Интерполяционный многочлен Гаусса
$X_2 = 1,277$
$x < a$, используем вторую интерполяционную формулу Гаусса

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

$$+\frac{(t+2)(t+1)t(t-1)}{4!}\Delta^4 y_{-2} + \ldots + \frac{(t+n-1)\ldots(t-n+1)}{(2n-1)!}\Delta^{2n-1} y_{-n} +$$

$$\frac{(t+n)(t+n-1)\ldots(t-n+1)}{(2n)!}\Delta^{2n} y_{-n}$$

$$t = \frac{1.277 - 1,25}{0.1} = 0.27$$

$$P_n(1.277) = 2.1459 + 0.27 \cdot 1.0106 + \frac{0.27(0.27+1)}{2!} \cdot (-0.0037) +$$
$$+ \frac{0.27(0.27+1)(0.27-1)}{3!} \cdot 0.0342$$
$$P_n(1.277) = 2.1459 + 0.272862 - 0.000636 - 0.001167$$
$$P_n(1.277) \approx 2.417$$

5 Листинг программы

Ссылка на Github репозиторий

Метод Лагранжа:

```
def lagrange_interpolation(x, y, value):
    name = "          ␣                    "
    n = len(x)
    result = 0.0
    for i in range(n):
        term = y[i]
        for j in range(n):
            if i != j:
                term *= (value - x[j]) / (x[i] - x[j])
        result += term
    return result
```

Метод Ньютона с разделенными разностями:

```
def newton_interpolation_with_shared_difference(x, y, value):
    name = "           ␣              ␣  ␣                        ␣
                                     "
    n = len(x)
    result = y[0]
    temp = 1.0
    shared_differences_table = create_shared_differences_table(x, y)
    for i in range(1, n):
        temp *= (value - x[i - 1])
        result += temp * (shared_differences_table[0, i])
    return result
```

Метод Ньютона с конечными разностями:

```
def newton_interpolation_with_finite_difference(x, y, inter_point):
name = "          ␣              ␣  ␣                    ␣                     "
    diff_table = create_finite_differences_table(x, y)
    n = len(x)
    h = x[1] - x[0]
    def t_forward(x_val):
        return (x_val - x[0]) / h
    def t_backward(x_val):
      return (x_val - x[-1]) / h
    def forward(x_val):
        result = y[0]
        for i in range(1, n):
            term = diff_table[0][i]
            prod = 1
            for j in range(i):
                prod *= (t_forward(x_val) - j)
            term *= prod / math.factorial(i)
            result += term
        return result
    def backward(x_val):
        result = y[-1]
        for i in range(1, n):
            term = diff_table[-i - 1][i]
            prod = 1
            for j in range(i):
                prod *= (t_backward(x_val) + j)
            term *= prod / math.factorial(i)
            result += term
        return result
    if (x[-1] - x[0]) / 2 < inter_point:
        return forward(inter_point)
    else:
        return backward(inter_point)
```

Метод Стирлинга:

```
def stirling_interpolation(x, y, value):
    name = "          ␣                    "
    n = len(x)
    central_differences_table = create_central_differences_table(y)
    alpha_index = n // 2
    h = x[1] - x[0]
    dts = generate_array_offset(n // 2)
    f1 = first_interpolation_gauss_form(n, x, y, alpha_index, dts, h,
        central_differences_table, value)
    f2 = second_interpolation_gauss_form(n, x, y, alpha_index, dts, h,
        central_differences_table, value)
    return (f1 + f2) / 2
```

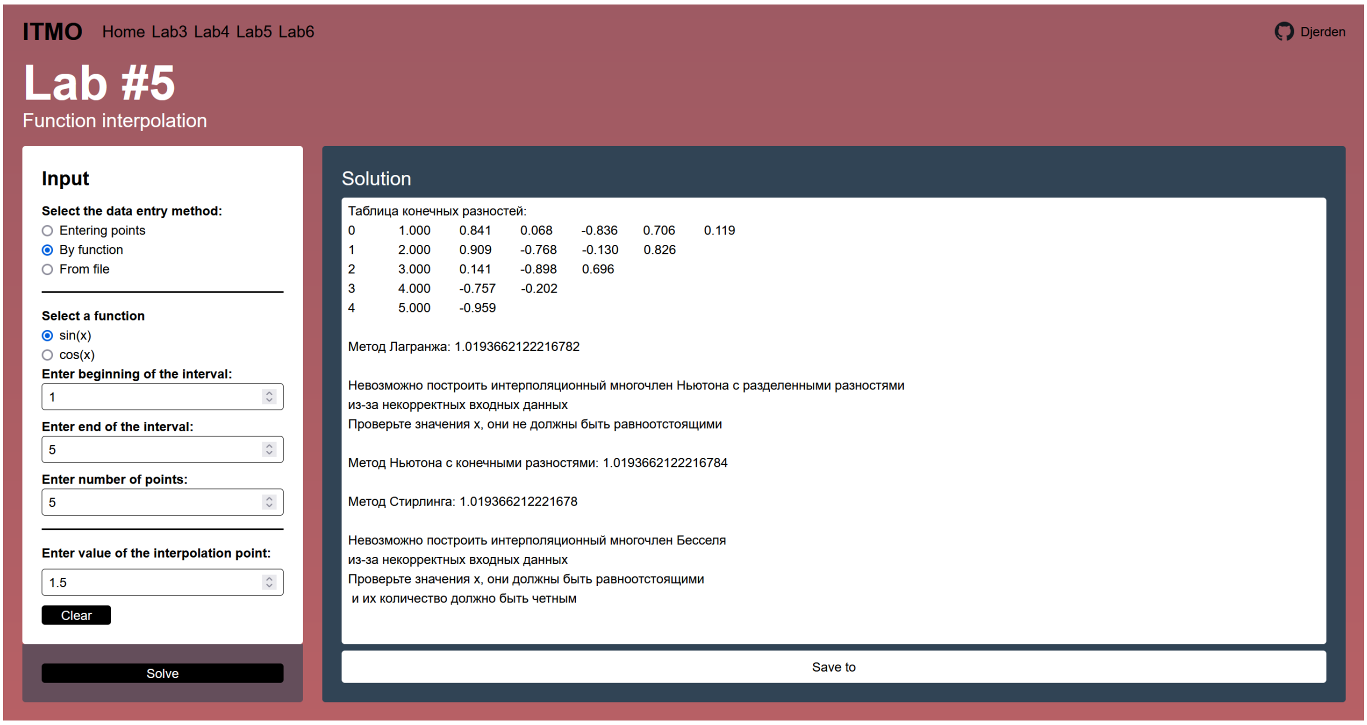
Метод Бесселя:

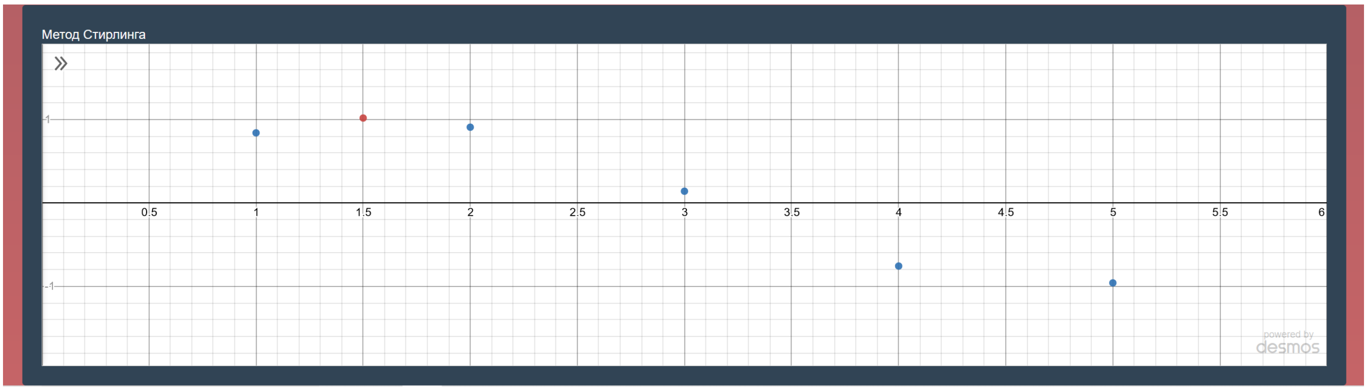
```
def bessel_interpolation(x, y, value):
    name = "          ␣                "
    n = len(x)
    central_differences_table = create_central_differences_table(y)
    alpha_index = n // 2
    h = x[1] - x[0]
    dts = generate_array_offset(n // 2)
    t = (value - x[alpha_index - 1]) / h
    result = (y[alpha_index - 1] + y[alpha_index]) / 2
    k = 0
    l = 0
    for i in range(1, n):
        if i % 2 == 1:
            current = (t - 0.5)
            for j in range(i - 1):
                current = current * (t + dts[j])
            current /= (math.factorial(i))
            current *= central_differences_table[i][alpha_index-1- k]
            result += current
            k += 1
        else:
            current = 1
            for j in range(i):
                current *= (t + dts[j])
            current /= (math.factorial(i))
            current *= (central_differences_table[i][alpha_index-l -2]  +
                central_differences_table[i][
                alpha_index - 1 - l]) / 2
            l += 1
            result += current
    return result
```

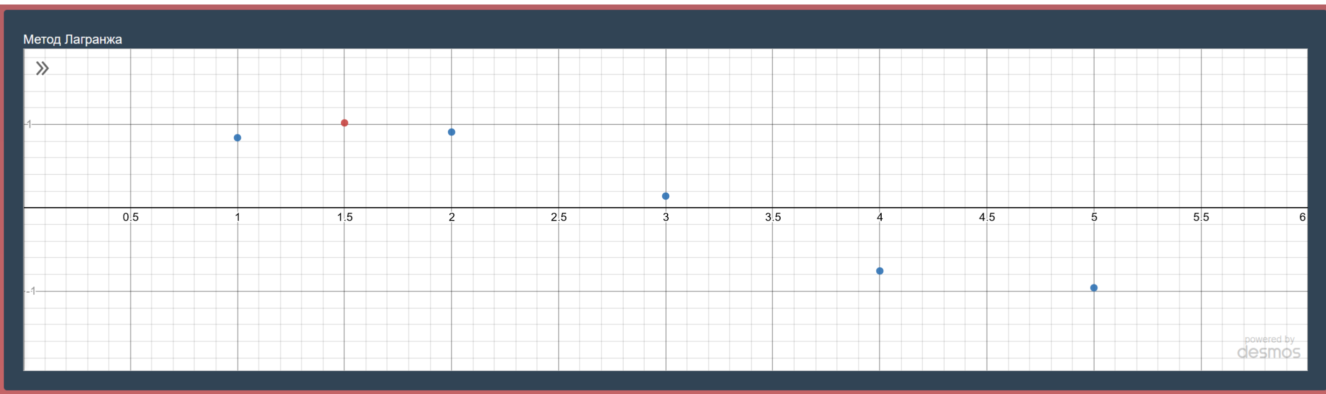
6 Результаты выполнения программы

6.1 Пример 1

Функция: sin(x)
Интервал: [1,5]
Количество точек: 5
Точка интерполяции: 1.5







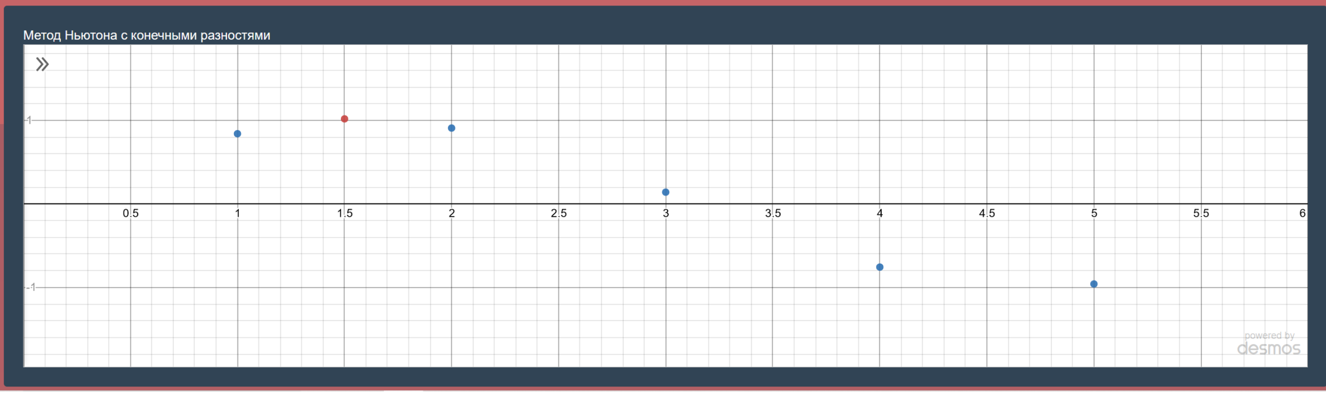


Таблица конечных разностей:
0 1.000 0.841 0.068 -0.836 0.706 0.119
1 2.000 0.909 -0.768 -0.130 0.826
2 3.000 0.141 -0.898 0.696
3 4.000 -0.757 -0.202
4 5.000 -0.959

Метод Лагранжа: 1.0193662122216782

Невозможно построить интерполяционный многочлен Ньютона с разделенными разностями из-за некорректных входных данных
Проверьте значения x, они не должны быть равноотстоящими

Метод Ньютона с конечными разностями: 1.0193662122216784

Метод Стирлинга: 1.019366212221678

Невозможно построить интерполяционный многочлен Бесселя из-за некорректных входных данных Проверьте значения x, они должны быть равноотстоящими и их количество должно быть четным

7 Вывод

В ходе данной лабораторной работы, было изучено несколько численных методов интерполирования, а именно:
Метод Лагранжа:
Преимущества: Простота реализации и понятность.
Недостатки: При добавлении новых узлов требуется пересчет всего многочлена, что делает его менее эффективным при большом количестве точек.
Применение: Хорошо подходит для небольшого количества узлов (до 20), после чего вычислительная сложность значительно увеличивается.
Метод Ньютона:
Преимущества: Позволяет легко добавлять новые узлы без пересчета предыдущих значений. Метод более гибкий по сравнению с Лагранжем.
Недостатки: Сложность реализации из-за необходимости использования нескольких формул в зависимости от структуры данных.
Применение: Удобен для последовательного добавления новых узлов, что делает его эффективным для задач, где требуется динамическое обновление интерполяции.
Метод Гаусса:
Преимущества: Хорошо подходит для интерполяции в случае, когда узлы находятся ближе к центральной точке отрезка. Метод позволяет улучшить точность интерполяции относительно опорной точки.
Недостатки: Требует более сложных вычислений и не всегда применим, если узлы не симметричны относительно центра.
Применение: Подходит для узлов, расположенных ближе к центру интервала, обеспечивая более точную интерполяцию для таких задач.
Методы Бесселя и Стирлинга:
Преимущества: Эти методы учитывают симметричное расположение узлов относительно центральной точки, что делает их точными для задач с равномерно распределенными узлами.
Недостатки: Требуют симметрии узлов и могут быть менее точными для неравномерно расположенных данных.
Применение: Используются для интерполяции в задачах с симметричными узлами, где важно минимизировать погрешности при центральных значениях.