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

Факультет Программной Инженерии и Компьютерной Техники

Лабораторная работа №5

«Интерполяция функции»

по дисциплине «Вычислительная математика»

Вариант: **9**

Преподаватель:
Машина Екатерина Алексеевна

Выполнил:
Пронкин Алексей Дмитриевич
Группа: P3208

Санкт-Петербург, 2025 г.

Цель работы

Цель лабораторной работы: решить задачу интерполяции, найти значения функции при заданных значениях аргумента, отличных от узловых точек.

Исходная таблица $y = f(x)$

i	x	y	Δy	$\Delta^2 y$	$\Delta^3 y$	$\Delta^4 y$	$\Delta^5 y$	$\Delta^6 y$
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	
2	1,25	2,1459	1,0106	−0,0100	0,0342	−0,0682		
3	1,35	3,1565	1,0006	0,0242	−0,0340			
4	1,45	4,1571	1,0248	−0,0098				
5	1,55	5,1819	1,0150					
6	1,65	6,1969						

Шаг сетки: $h = 0{,}10$.

1. Интерполяция в точке $X_1 = 1{,}562$ (формула Ньютона обратно-направленная)

Параметр $$p = \frac{X_1 - x_n}{h} = \frac{1{,}562 - 1{,}65}{0{,}10} = -0{,}88.$$

$$\begin{aligned} f(X_1) \approx y_n + p\,\nabla y_n + \frac{p(p+1)}{2!}\,\nabla^2 y_n + \frac{p(p+1)(p+2)}{3!}\,\nabla^3 y_n \\ + \frac{p(p+1)(p+2)(p+3)}{4!}\,\nabla^4 y_n + \frac{p(p+1)(p+2)(p+3)(p+4)}{5!}\,\nabla^5 y_n \\ + \frac{p(p+1)\dots(p+5)}{6!}\,\nabla^6 y_n. \end{aligned}$$

Подстановка значений

$$\begin{aligned} f(1{,}562) \approx 6{,}1969\ &\underbrace{+\,(-0{,}88)\,1{,}0150}_{-0{,}8932} \underbrace{+\,\tfrac{-0{,}88\cdot 0{,}12}{2}\,(-0{,}0098)}_{+0{,}00052} \underbrace{+\,\tfrac{-0{,}88\cdot 0{,}12\cdot 1{,}12}{6}\,(-0{,}0340)}_{+0{,}00067} \\ &\underbrace{+\,\tfrac{-0{,}88\cdot 0{,}12\cdot 1{,}12\cdot 2{,}12}{24}\,(-0{,}0682)}_{+0{,}00071} \underbrace{+\,\tfrac{-0{,}88\cdot\dots\cdot 3{,}12}{120}\,(-0{,}1087)}_{+0{,}00071} \underbrace{+\,\tfrac{-0{,}88\cdot\dots\cdot 4{,}12}{720}\,(-0{,}1478)}_{+0{,}00066} \\ &\approx \boxed{5{,}30697}. \end{aligned}$$

Погрешность после учёта члена $\nabla^6 y$ — менее $3 \cdot 10^{-4}$.

2. Интерполяция в точке $X_2 = 1{,}362$ (первая формула Гаусса)

Выбираем центральный узел $x_0 = 1{,}35$ (строка $i = 3$),

$$p = \frac{X_2 - x_0}{h} = \frac{1{,}362 - 1{,}35}{0{,}10} = 0{,}12.$$

$$
\begin{aligned}
f(X_2) \approx y_0 + p\,\Delta y_0 + \frac{p(p-1)}{2!}\,\Delta^2 y_{-1/2} + \frac{p(p+1)(p-1)}{3!}\,\Delta^3 y_{-1} \\
+ \frac{p(p+1)(p-1)(p-2)}{4!}\,\Delta^4 y_{-3/2} + \frac{p(p+1)(p-1)(p-2)(p+2)}{5!}\,\Delta^5 y_{-2} \\
+ \frac{p(p+1)\dots(p-3)}{6!}\,\Delta^6 y_{-5/2}.
\end{aligned}
$$

Подстановка значений

$$
\begin{aligned}
f(1{,}362) \approx 3{,}1565\ \underbrace{+\,0{,}12\,1{,}0006}_{+0{,}12007} \underbrace{+\,\tfrac{0{,}12(-0{,}88)}{2}\,(-0{,}0100)}_{+0{,}00053} \underbrace{+\,\tfrac{0{,}12\cdot 1{,}12\cdot(-0{,}88)}{6}\,(-0{,}0063)}_{+0{,}00012} \\
\underbrace{+\,\tfrac{0{,}12\cdot 1{,}12\cdot(-0{,}88)\cdot(-1{,}88)}{24}\,(0{,}0405)}_{+0{,}00038} \underbrace{+\,\tfrac{0{,}12\cdot\dots\cdot 2{,}12}{120}\,(0{,}0391)}_{+0{,}00015} \underbrace{+\,\tfrac{0{,}12\cdot\dots\cdot(-2{,}88)}{720}\,(-0{,}1478)}_{+0{,}00028} \\
\approx \boxed{3{,}27803}.
\end{aligned}
$$

Погрешность после члена $\Delta^6 y$ — порядка $6 \cdot 10^{-4}$.

Итоговые значения функции

$$
f(1{,}562) = 5{,}307\ \ (\pm 0{,}0003), \qquad f(1{,}362) = 3{,}278\ \ (\pm 0{,}0006).
$$

Программная реализация задачи:

```python
# ----------------------------------------
# Функции, интегралы которых вычисляем
# ----------------------------------------
def f(x):
    """
    3(x)^2
    """
    return 3 * x ** 2


def f2(x):
    """
    3(x)^3+4x
    """
    return 3 * x ** 3 + 4 * x


def f3(x):
    """
    3(x)^2+4x+5
    """
    return 3 * x ** 2 + 4 * x + 5


# ----------------------------------------
# Методы вычисления интегралов
# ----------------------------------------
```

```python

# ---------------------------------------
# прямоугольники
# ---------------------------------------
def left_rectangle_method(func, a, b, n):
    """
    Вычисление интеграла методом левых прямоугольников.

    Параметры:
      func: интегрируемая функция,
      a, b: пределы интегрирования,
      n: число разбиений.

    Возвращает:
      Приближённое значение интеграла.
    """
    h = (b - a) / n
    total = 0.0
    for i in range(n):
        x_i = a + i * h  # левая точка подинтервала
        total += func(x_i)
    return total * h


def right_rectangle_method(func, a, b, n):
    """
    Вычисление интеграла методом правых прямоугольников.

    Параметры:
      func: интегрируемая функция,
      a, b: пределы интегрирования,
      n: число разбиений.

    Возвращает:
      Приближённое значение интеграла.
    """
    h = (b - a) / n
    total = 0.0
    for i in range(1, n + 1):
        x_i = a + i * h  # правая точка подинтервала
        total += func(x_i)
    return total * h


def middle_rectangle_method(func, a, b, n):
    """
    Вычисление интеграла методом центральных прямоугольников.

    Параметры:
      func: интегрируемая функция,
      a, b: пределы интегрирования,
      n: число разбиений.

    Возвращает:
      Приближённое значение интеграла.
    """
    h = (b - a) / n
    total = 0.0
    for i in range(n):
```

```
            x_i = a + i * h + h / 2  # центральная точка подинтервала
            total += func(x_i)
        return total * h


# ---------------------------------------
# трапеция
# ---------------------------------------
def trapezoid_method(func, a, b, n):
    """
    Вычисление интеграла методом трапеций.

    Параметры:
      func: интегрируемая функция,
      a, b: пределы интегрирования,
      n: число разбиений.

    Возвращает:
      Приближённое значение интеграла.
    """
    h = (b - a) / n
    total = (func(a) + func(b)) / 2.0
    for i in range(1, n):
        x_i = a + i * h  # левая точка подинтервала
        total += func(x_i)
    return total * h


# ---------------------------------------
# симпсон
# ---------------------------------------
def simpson_method(func, a, b, n):
    """
    Вычисление интеграла методом Симпсона.

    Параметры:
      func: интегрируемая функция,
      a, b: пределы интегрирования,
      n: число разбиений.

    Возвращает:
      Приближённое значение интеграла.
    """
    h = (b - a) / n
    total = func(a) + func(b)
    for i in range(1, n, 2):
        x_i = a + i * h  # левая точка подинтервала
        total += 4 * func(x_i)
    for i in range(2, n, 2):
        x_i = a + i * h
        total += 2 * func(x_i)
    return total * h / 3


# ---------------------------------------
# Вычисление интеграла с правилом Рунге
# ---------------------------------------
def integrate_with_runge(method, func, a, b, eps, order=1, initial_n=4):
    """
```

```
    Вычисление интеграла с использованием заданного метода и правила Рунге для
    оценки погрешности.

    Параметры:
      method: функция для вычисления интеграла (например, left_rectangle_method или
       right_rectangle_method),
      func: интегрируемая функция,
      a, b: пределы интегрирования,
      eps: требуемая точность,
      order: порядок метода (для прямоугольников order = 1),
      initial_n: начальное число разбиений (по умолчанию 4).

    Возвращает:
      Кортеж (интегральное приближение, итоговое число разбиений).
    """
    n = initial_n
    I_n = method(func, a, b, n)
    n *= 2
    I_2n = method(func, a, b, n)

    # Правило Рунге: |I(2n) - I(n)| / (2^order - 1) < eps
    while abs(I_2n - I_n) / (2 ** order - 1) > eps:
        I_n = I_2n
        n *= 2
        I_2n = method(func, a, b, n)

    return I_2n, n


def main():
    valid_choices = {'1', '2', '3'}

    function_choice = None
    function_to_compute = None
    print("Вычисление интеграла функции")
    print("Выберите функцию, которую хотите проинтегрировать:")
    print("1 - 3x^2")
    print("2 - 3x^3 + 4x")
    print("3 - 3x^2 + 4x + 5")

    while function_choice not in valid_choices:
        function_choice = input("Введите номер функции: ").strip()
        if function_choice not in valid_choices:
            print("Неверный ввод. Пожалуйста, введите 1, 2 или 3.")

    if function_choice == '1':
        function_to_compute = f
    elif function_choice == '2':
        function_to_compute = f2
    elif function_choice == '3':
        function_to_compute = f3


    print("Выберите метод вычисления интеграла:")
    print("1 - Метод прямоугольников")
    print("2 - Метод трапеций")
    print("3 - Метод Симпсона")

    choice = None
```

```python

    while choice not in valid_choices:
        choice = input("Введите номер метода: ").strip()
        if choice not in valid_choices:
            print("Неверный ввод. Пожалуйста, введите 1, 2 или 3.")

    if choice == "1":
        print("Какой прямоугольник будем использовать для вычислений?")
        print("1 - Левый")
        print("2 - Средний")
        print("3 - Правый")

        choice2 = None

        while choice2 not in valid_choices:
            choice2 = input("Введите номер метода: ").strip()
            if choice2 not in valid_choices:
                print("Неверный ввод. Пожалуйста, введите 1, 2 или 3.")

        if choice2 == "1":
            integration_method = left_rectangle_method
            order = 1  # порядок метода для прямоугольников (левый)
        elif choice2 == "2":
            integration_method = middle_rectangle_method
            order = 2  # порядок метода для прямоугольников (средний)
        elif choice2 == "3":
            integration_method = right_rectangle_method
            order = 1  # порядок метода для прямоугольников (правый)
    elif choice == "2":
        integration_method = trapezoid_method
        order = 2
    elif choice == "3":
        integration_method = simpson_method
        order = 4
    else:
        print("Некорректный выбор метода.")
        return

    try:
        a = float(input("Введите нижний предел интегрирования (a): "))
        b = float(input("Введите верхний предел интегрирования (b): "))
        eps = float(input("Введите требуемую точность (eps): "))
    except ValueError:
        print("Ошибка: необходимо вводить числовые значения.")
        return

    result, subdivisions = integrate_with_runge(
        integration_method, function_to_compute, a, b, eps, initial_n=4, order=order
    )

    print("\nРезультаты вычисления:")
    print(f"Приближённое значение интеграла: {result}")
    print(f"Число разбиений для достижения требуемой точности: {subdivisions}")


if __name__ == "__main__":
    main()
```