УНИВЕРСИТЕТ ИТМО

Факультет программной инженерии и компьютерной техники

Вычислительная математика

Лабораторная работа №3

Численное интегрирование

Преподаватель: Малышева Татьяна Алексеевна

Выполнил: Цыпандин Николай Петрович

Группа: P3210

Вариант: 24

Санкт-Петербург
2022

Цель работы

Реализовать Метод прямоугольников (все) и Симпсона для нахождения приближенного значения определенного интеграла с требуемой точностью

Описание методов

Метод прямоугольников

$\int_a^b f(x)dx \approx h_1 y_0 + h_2 y_1 + \cdots + h_n y_{n-1} = \sum_{i=1}^{n} h_i\, y_{i-1}$ - левые прямоугольники

$\int_a^b f(x)dx \approx h_1 y_1 + h_2 y_2 + \cdots + h_n y_n = \sum_{i=1}^{n} h_i y_i$ - правые прямоугольники

При $h_i = h = \frac{b-a}{n} = const$:

$$\int_a^b f(x)dx = h \sum_{i=1}^{n} y_{i-1}$$

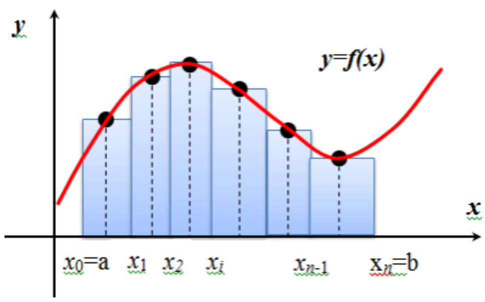
$$\int_a^b f(x)dx = h \sum_{i=1}^{n} y_i$$

Метод прямоугольников. Метод средних

Для аналитически заданных функций более точным является использование значений в средних точках элементарных отрезков (полуцелых узлах):

$$\int_a^b f(x)dx \approx \sum_{i=1}^{n} h_i\, f(x_{i-1/2})$$

$$x_{i-1/2} = \frac{x_{i-1} + x_i}{2} = x_{i-1} + \frac{h_i}{2}, i = 1, 2, \ldots n$$



При $h_i = h = \frac{b-a}{n} = const$:

$$\int_a^b f(x)dx = h \sum_{i=1}^{n} f(x_{i-1/2})$$

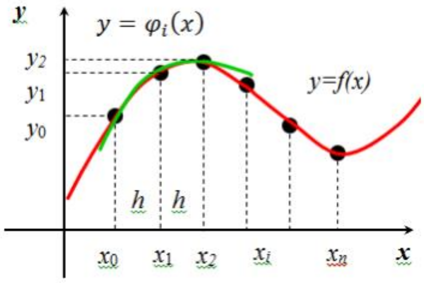
Метод Симпсона (**Симпсон Томас**(20.8.1710–14.5.1751) – английский математик)

Разобьем отрезок интегрирования $[a, b]$ на четное число n равных частей с шагом h. На каждом отрезке $[x_0, x_2]$, $[x_2, x_4]$, …, $[x_{i-1}, x_{i+1}]$, … , $[x_{n-2}, x_n]$ подынтегральную функцию заменим интерполяционным многочленом второй степени:

$$f(x) \approx \varphi_i(x) = a_i x^2 + b_i x + c_i, \quad x_{i-1} \le x \le x_{i+1}$$

Коэффициенты этих квадратных трехчленов могут быть найдены из условий равенства многочлена и подынтегральной функции в узловых точках.

В качестве $\varphi_i(x)$ можно принять интерполяционный многочлен Лагранжа второй степени, проходящий через точки $(x_{i-1}, y_{i-1}), (x_i, y_i), (x_{i+1}, y_{i+1})$.

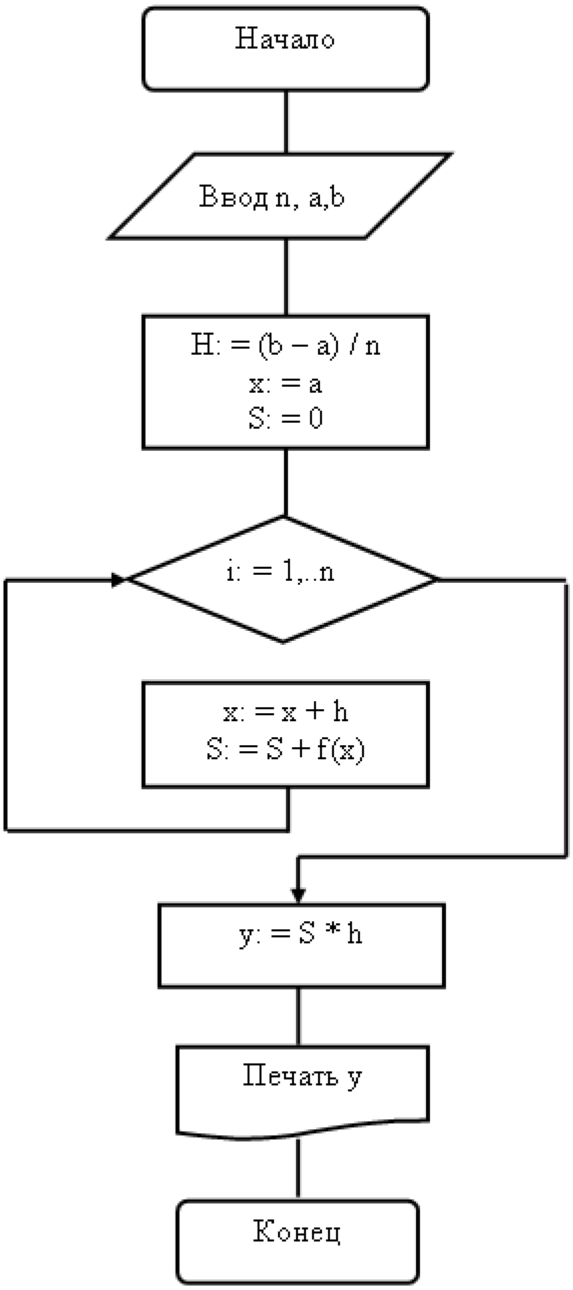


Формула Симпсона

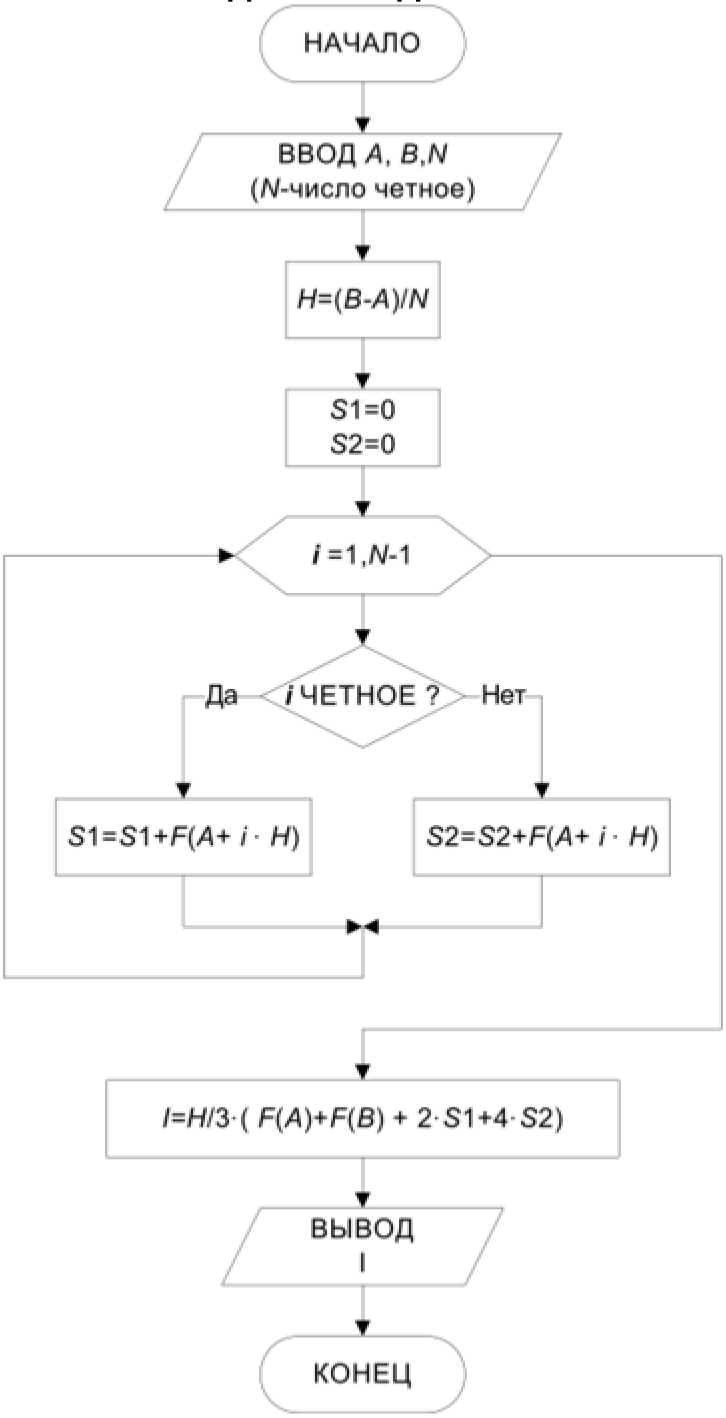
$$\int_a^b f(x) = \frac{h}{3}\left[(y_0 + 4(y_1 + y_3 + \dots + y_{n-1}) + 2(y_2 + y_4 + \dots + y_{n-2}) + y_n)\right]$$

1) Формулы средних прямоугольников: $|R| \le \max\limits_{x\in[a,b]} |f''(x)| \cdot \frac{(b-a)^3}{24n^2}$: второй порядок точности $O(h^2)$
2) Формула трапеций: $|R| \le \max\limits_{x\in[a,b]} |f''(x)| \cdot \frac{(b-a)^3}{12n^2}$: $O(h^2)$
3) Формула Симпсона: $|R| \le \max\limits_{x\in[a,b]} |f''''(x)| \cdot \frac{(b-a)^5}{180n^4}$: $O(h^4)$

Блок-схема для метода прямоугольников:



Блок-схема для метода Симпсона:



Листинг программы (код численного метода)

```
class RectangleMethod:
    def __init__(self, function):
        self.function = function

    # method may be 0, 1/2 or 1 (left, mid, right respectively)
    def solve(self, a, b, n, method=1 / 2):
        result = 0
        h = (b - a) / n
        start = a

        for i in range(n):
            result += self.function.func(start + h * method) * h
            start += h

        return result

    def solve_while(self, a, b, eps, min_n=4, method=1 / 2, max_n=2 ** 25):
        n = min_n
        result = float('inf')
        last_result = 0

        while abs(result - last_result) / 3 > eps:
            last_result = result
            result = self.solve(a, b, n, method)
            n *= 2
            if n >= max_n:
                print('Добиться требуемой точности за разумное время не
удалось')
                print('eps = ' + str(abs(last_result - result)))
                break

        return {'n': n, 'result': result}


class SimpsonMethod:
    def __init__(self, function):
        self.function = function

    def solve(self, a, b, n):
        if n % 2 != 0:
            return None

        h = (b - a) / n
        x = a + h
        result = 0

        for i in range(n - 1):
            if i % 2 == 0:
                result += 4 * self.function.func(x)
            else:
                result += 2 * self.function.func(x)
            x += h
        result *= h / 3

        return result

    def solve_while(self, a, b, eps, min_n=4, max_n=2 ** 25):
        if min_n % 2 != 0:
            return None

        n = min_n
        result = float('inf')
        last_div = float('inf')
        last_result = 0
```

```
        while abs(result - last_result) / 15 > eps:
            last_result = result
            result = self.solve(a, b, n)
            n *= 2
            if n >= max_n:
                print('Добиться требуемой точности за разумное время не удалось')
                print('eps = ' + str(abs(last_result - result)))
                break
            if abs(last_result - result) > last_div:
                print('Интеграл расходится')
                exit(1)

            last_div = abs(last_result - result)
        return {'n': n, 'result': result}
```

Полный код можно найти по ссылке:

https://github.com/kkkooolllyyyaaa/computational_math/tree/master/comp-math-3

Пример 1

Выберите метод решения:

1 — Метод прямоугольников

2 — Метод Симпсона

1

Выберите функцию:

1 — x^3 - 2x^2 - 5x+24

2 — sin^2(x)

3 — ln(x^2)

4 — 1/x^2 + 7*x

5 — x^2

1

Введите пределы интегрирования:

a: 3

b: 5

Введите погрешность вычисления:

eps: 1e-2

1 - левое, 2 - среднее, 3 - правое: 2

Результат: 78.66015625

Количество разбиений: 64

Пример 2

Выберите метод решения:

1 — Метод прямоугольников

2 — Метод Симпсона

2

Выберите функцию:

1 — x^3 - 2x^2 - 5x+24

2 — sin^2(x)

3 — ln(x^2)

4 — 1/x^2 + 7*x

5 — x^2

1

Введите пределы интегрирования:

a: 3

b: 5

Введите погрешность вычисления:

eps: 1e-5

Результат: 78.6665496826

Количество разбиений: 1048576

Сравнение методов:

• Интеграл точно

$$\int_3^5 (x^3 - 2x^2 - 5x + 24)\,dx = \frac{x^4}{4} - \frac{2x^3}{3} - \frac{5x^2}{2} + 24x \Big|_3^5 =$$

$$= \frac{236}{3} \approx 78,(6)$$

• Ньютон – Котес

$$I = C_6^0\, f(a) + C_6^1\, f\left(a\tfrac{2}{6}\right) + C_6^2\, f\left(a\tfrac{4}{6}\right) + \ldots + C_6^6 \cdot f(2a)$$

$$C_6^0 = C_6^6 = \frac{41(2)}{840} \qquad C_6^1 = C_6^5 = \frac{9\cdot 2}{35} \qquad C_6^2 = C_6^4 = \frac{9\cdot 2}{280}$$

$$C_6^3 = \frac{34\cdot 2}{105}$$

$f(3) = 18$

$f\left(3\tfrac{2}{6}\right) = 22,148$

$f\left(3\tfrac{4}{6}\right) = 28,074$

$f(4) = 36$

$f\left(4\tfrac{2}{6}\right) = 46,148$

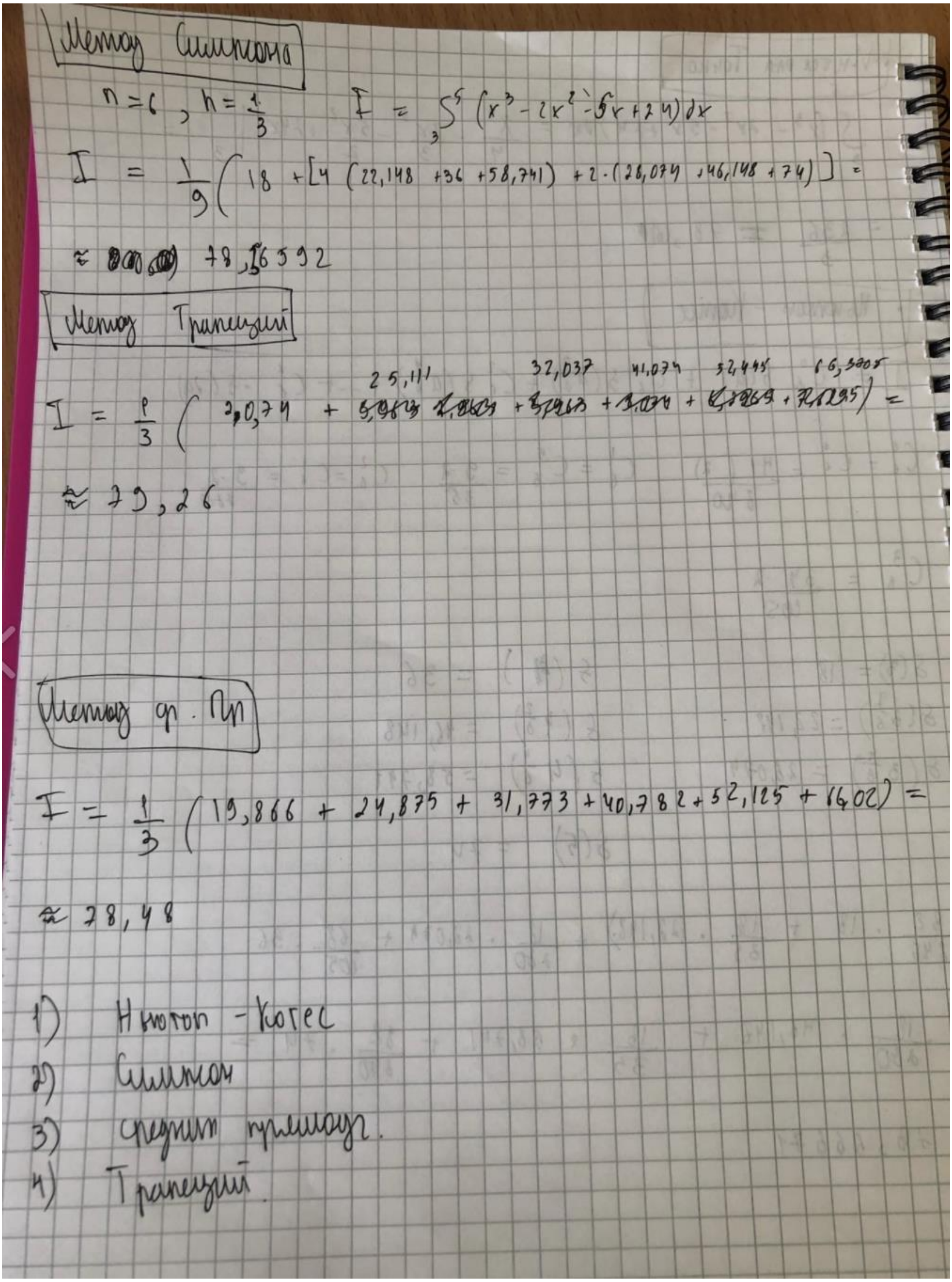
$f\left(4\tfrac{4}{6}\right) = 58,741$

$f(5) = 74$

$$\frac{82}{840}\cdot 18 + \frac{18}{35}\cdot(22,148) + \frac{18}{280}\cdot 28,074 + \frac{68}{105}\cdot 36$$

$$+ \frac{18}{280}\cdot 46,148 + \frac{18}{35}\cdot 58,741 + \frac{82}{840}\cdot 74 =$$

$$\approx 78,66671$$



Метод Симпсона

$n = 6$, $h = \frac{1}{3}$ $\quad I = \int_3^5 (x^3 - 2x^2 - 5x + 24)\,dx$

$I = \frac{1}{9}\left(18 + [4(22{,}148 + 36 + 58{,}741) + 2\cdot(28{,}074 + 46{,}148 + 74)]\right) =$

$\approx 78{,}56592$

Метод Трапеций

$I = \frac{1}{3}\left(20{,}74 + 25{,}111 + 32{,}037 + 41{,}074 + 52{,}445 + 66{,}3305\right) =$

$\approx 79{,}26$

Метод ср. Пр

$I = \frac{1}{3}\left(19{,}866 + 24{,}875 + 31{,}773 + 40{,}782 + 52{,}125 + 66{,}02\right) =$

$\approx 78{,}48$

1) Ньютон - Котес
2) Симпсон
3) средних прямоуг.
4) Трапеций.

Вывод

Написав реализации двух методов решения интегралов, можно сделать вывод, что метод Симпсона быстрее и точнее нежели метод прямоугольников. Также, метод средних прямоугольников, как правило, точнее методов левых и правых прямоугольников.