**Санкт-Петербургский национальный исследовательский университет**

**информационных технологий, механики и оптики**

**Кафедра информатики и прикладной математики**

Вычислительная математика

Лабораторная работа №2

Метод прямоугольников

Выполнил: Гхази Даниэль

Группа P3218

Преподаватель: Кучер Алексей Владимирович

2017 г.

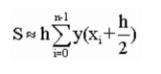
г. Санкт-Петербург

**1. Описание метода**

Метод средних прямоугольников.

Разделим отрезок [a; b] на n равных частей, т.е. на n элементарных отрезков. Длина каждого элементарного отрезка. Точки деления будут: x0=a; x1=a+h; x2=a+2× h, ... , xn-1=a+(n-1)× h; xn=b. Эти числа будем называть узлами. Вычислим значения функции f(x) в узлах, обозначим их y0, y1, y2, ... , yn. Cтало быть, y0=f(a), y1=f(x1), y2=f(x2), ... , yn=f(b). Числа y0, y1, y2, ... , yn являются ординатами точек графика функции, соответствующих абсциссам x0, x1, x2, ... , xn. Площадь криволинейной трапеции приближенно заменяется площадью многоугольника, составленного из n прямоугольников. Таким образом, вычисление определенного интеграла сводится к нахождению суммы n элементарных прямоугольников.

Формула средних прямоугольников:



**2. Блок-схема** (для левых прямоугольников)

****

**3. Реализация метода**

internal class RectanglesComputingMethods

{

public double ComputeUsingLeftRectangles(Function func, double a, double b, double accur, out long numOfPartitions, out double delta)

{

double I1 = 0, I2;

double x;

double s, h, sigma = (1.0 / 3);

int n = 10;

double diff;

do

{

h = (b - a) / n;

x = a;

s = 0;

for (long i = 0; i < n; i++)

{

s = s + func(x);

x += h;

}

I2 = s \* h;

diff = sigma \* Math.Abs(I2 - I1);

if (diff > accur)

{

n = n \* 2;

I1 = I2;

}

} while (diff > accur);

numOfPartitions = n;

delta = Math.Abs(I2 - I1);

return I2;

}

public double ComputeUsingRightRectangles(Function func, double a, double b, double accur, out long numOfPartitions, out double delta)

{

double I1 = 0, I2;

double x1;

double s, h, sigma = (1.0 / 3);

int n = 10;

double diff;

do

{

h = (b - a) / n;

x1 = a;

s = 0;

for (int i = 0; i < n; i++)

{

s = s + func(x1 + h);

x1 += h;

}

I2 = s \* h;

diff = sigma \* Math.Abs(I2 - I1);

if (diff > accur)

{

n = n \* 2;

I1 = I2;

}

} while (diff > accur);

numOfPartitions = n;

delta = Math.Abs(I2 - I1);

return I2;

}

public double ComputeUsingMiddleRectangles(Function func, double a, double b, double accur, out long numOfPartitions, out double delta)

{

double I1 = 0, I2;

double x1;

double s, h, sigma = (1.0 / 3);

int n = 10;

double diff;

do

{

h = (b - a) / n;

x1 = a;

s = 0;

for (int i = 0; i < n; i++)

{

s = s + func((x1 + x1 + h) / 2);

x1 += h;

}

I2 = s \* h;

diff = sigma \* Math.Abs(I2 - I1);

if (diff > accur)

{

n = n \* 2;

I1 = I2;

}

} while (diff > accur);

numOfPartitions = n;

delta = Math.Abs(I2 - I1);

return I2;

}

}

**4. Пример работы программы**

**Ввод:**

Точность: 0,0001

Верхний предел: 10

Нижний предел: 1

Функция: x

Метод: Левых прямоугольников

**Вывод:**

Результат: 49,499752

Количество разбиений: 163840

Погрешность: 0,00024719

**5. Вывод**

В ходе работы мною был изучены и реализованы методы левых, правых и средних прямоугольников. Также мною были дополнительно изучены метод трапеций и метод Симпсона.

Наиболее эффективным является метод Симпсона. Метод прямоугольников наиболее прост в реализации, но метод трапеций является более точным. Комбинация двух последних дает ту же погрешность, что и метод Симпсона, но они менее производительны.