МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

НИЖЕГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ им. Р.Е.АЛЕКСЕЕВА

Институт радиоэлектроники и информационных технологий

Кафедра информатики и систем управления

Лабораторная работа №4

«Численное интегрирование функций»

по дисциплине

Вычислительная математика

РУКОВОДИТЕЛЬ:

________________ Суркова А.С.

СТУДЕНТ:

________________ Сухоруков В.А.

19-ИВТ-3

Работа защищена «___» ____________

С оценкой _______________________

Нижний Новгород 2021

Оглавление

Цель

Закрепление знаний и умений по численному интегрированию функций.

Постановка задачи

Вычислить интеграл по формулам центральных (средних) прямоугольников, трапеций и формуле Симпсона, при n=8 и n=20; оценить погрешность результата.

$$19.\ \int_{2.5}^{3.3} \frac{\lg(x^2+0.8)}{x-1}\,dx$$

Теоретические сведения

Метод средних (центральных) прямоугольников.

Метод прямоугольников- метод интегрирования функции одной переменной, заключающийся в замене подынтегральной функции на многочлен нулевой степени, то есть константу, на каждом элементарном отрезке. Если рассмотреть график подынтегральной функции, то метод будет заключаться в приближённом вычислении площади под графиком суммированием площадей конечного числа прямоугольников, ширина которых будет определяться расстоянием между соответствующими соседними узлами интегрирования, а высота — значением подынтегральной функции в этих узлах.

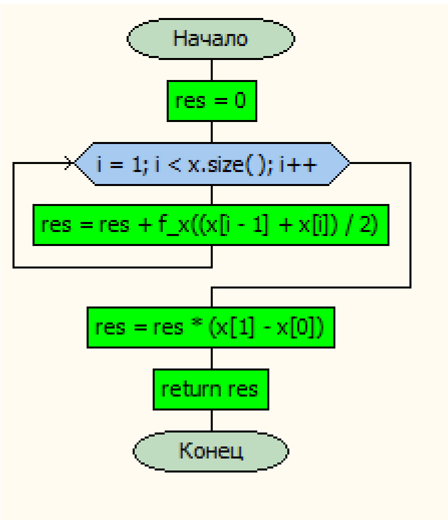
Составная квадратурная формула для метода средних (центральных) прямоугольников.

$$h = \frac{b-a}{n}$$

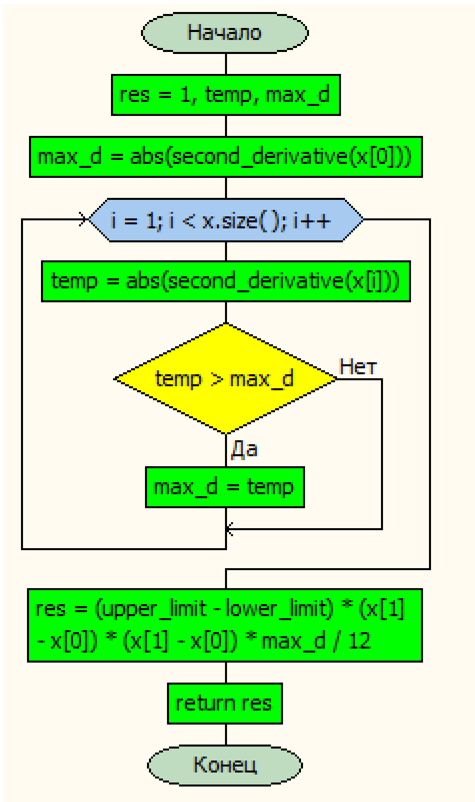
$$\int_a^b f(x)dx = h\sum_{i=1}^{n} f\left(x_{i-\frac{1}{2}}\right)$$

Погрешность формулы интегрирования метода средних (центральных) прямоугольников:

$$|R_n| \leq \frac{(b-a)}{24} h^2 max|f''(x)|$$



Вычисление значения интеграла



Вычисление погрешности

Метод трапеций

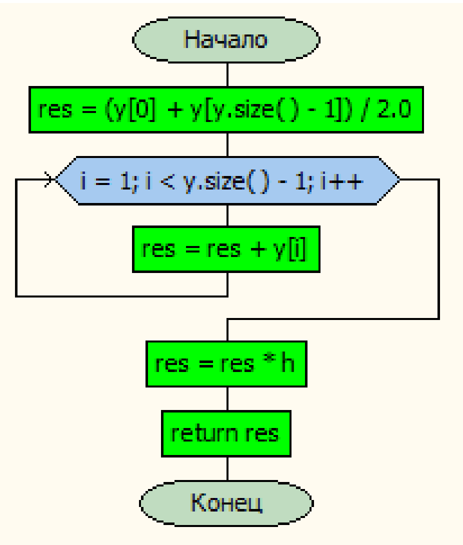
Метод трапеций — метод численного интегрирования функции одной переменной, заключающийся в замене на каждом элементарном отрезке подынтегральной функции на многочлен первой степени, то есть линейную функцию. Площадь под графиком функции аппроксимируется прямоугольными трапециями.

$$h = \frac{b-a}{n}$$

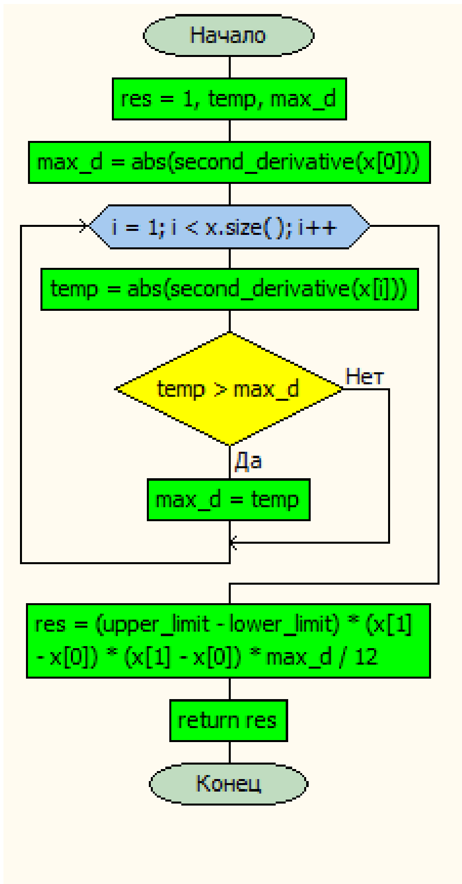
$$\int_a^b f(x)dx = h\left(\frac{y_0 + y_n}{2} + \sum_{i=1}^{n-1} f(x_i)\right)$$

Погрешность формулы интегрирования метода средних (центральных) прямоугольников:

$$|R_n| \leq \frac{(b-a)}{12} h^2 max|f''(x)|$$



Вычисление значения интеграла



Вычисление погрешности

Метод Симсона

Формула Симпсона (также **Ньютона-Симпсона**) относится к приёмам численного интегрирования. Получила название в честь британского математика Томаса Симпсона (1710—1761).

Суть метода заключается в приближении подынтегральной функции на отрезке $[a;b]$ интерполяционным многочленом второй степени $p_2(x)$, то есть приближение графика функции на отрезке параболой.

$$h = \frac{b-a}{n}$$

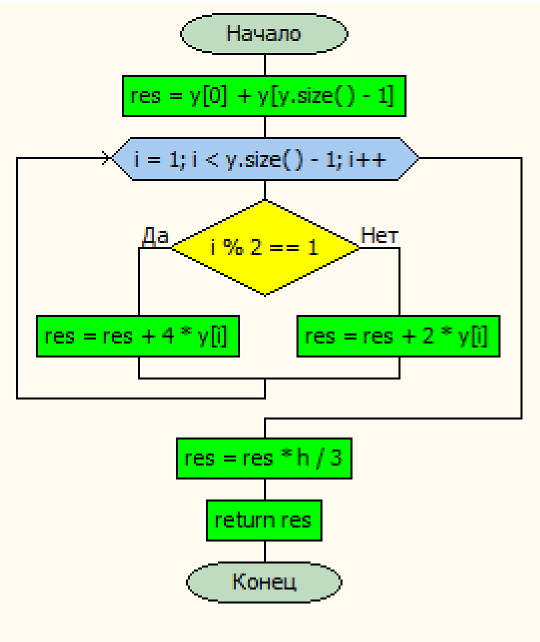
$$\int_a^b f(x)dx = \frac{h}{3}(y_0 + 4(y_1 + y_3 + \cdots + y_{n-1}) + 2(y_2 + y_4 + \cdots + y_{n-2}) + y_n)$$

Погрешность формулы интегрирования метода Симпсона:

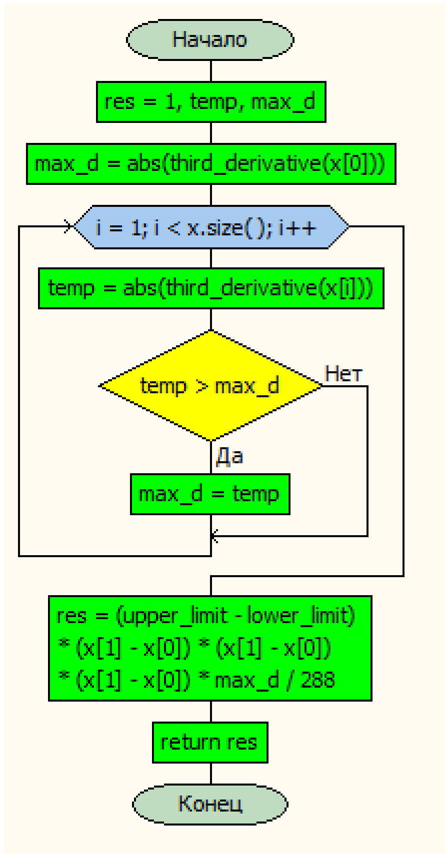
$$|R_n| \le \frac{(b-a)}{2880} h^4 max|f^{(4)}(x)|$$

Погрешность формулы интегрирования метода Симпсона при невозможности ввода производной четвертого порядка:

$$|R_n| \le \frac{(b-a)}{288} h^3 max|f^{(3)}(x)|$$



Вычисление значения интеграла



Вычисление погрешности

Расчетные данные

Исходная функция:

$$19.\ \int_{2.5}^{3.3} \frac{\lg(x^2+0.8)}{x-1}dx$$

Первая производная:

$$\frac{2x}{(x-1)\left(x^2+\frac{4}{5}\right)} - \frac{\log\left(x^2+\frac{4}{5}\right)}{(x-1)^2}$$

Вторая производная:

$$\frac{2\left(-\frac{10x}{(x-1)(5x^2+4)} - \frac{5\left(\frac{10x^2}{5x^2+4}-1\right)}{5x^2+4} + \frac{\log\left(x^2+\frac{4}{5}\right)}{(x-1)^2}\right)}{x-1}$$

Третья производная:

$$\frac{2\left(\frac{50x\left(\frac{20x^2}{5x^2+4}-3\right)}{(5x^2+4)^2} + \frac{30x}{(x-1)^2(5x^2+4)} + \frac{15\left(\frac{10x^2}{5x^2+4}-1\right)}{(x-1)(5x^2+4)} - \frac{3\log\left(x^2+\frac{4}{5}\right)}{(x-1)^3}\right)}{x-1}$$

<u>При n = 8:</u>

	Значение интеграла	Значение погрешности
Метод средних прямоугольников	0.407905	0.0103256
Метод трапеций	0.408001	0.0206513
Метод Симпсона	0.407937	0.000001

<u>При n = 20:</u>

	Значение интеграла	Значение погрешности
Метод средних прямоугольников	0.407932	0.001652
Метод трапеций	0.407947	0.003304
Метод Симпсона	0.407937	0.0000001

Код программы

function.h

```
#pragma once
#include<iostream>
#include<vector>
#include<cmath>
#include"colors.h"

using namespace std;
/*Библиотека для работы с исходной функцией и её производными */


/*Функция считывания пределов интегрирования*/
void get_limits(double* lower_limit, double* upper_limit) {
	cout << Green << "Введите пределы интегрирования \n"
		<< Yellow << "\tНижний предел ";
	cin >> *lower_limit;
	cout << "\tВерхний предел ";
	cin >> *upper_limit;
}

/*Функция считывания n*/
int get_n() {
	int n;
	cout << Green << "Введите n - число отрезков разбиения ";
	cin >> n;
	return n;
}

/*Функция вывода меню*/
char menu_item_selection() {
	char c;
	cout << Green << "\nВыберите, что нужно сделать :\n" <<
Yellow
		<< "\t{1} - Найти интеграл методом центральных
прямоугольников\n"
		<< "\t{2} - Найти интеграл методом трапеций\n"
```

```
            << "\t{3} - Найти интеграл по формуле Симпсона\n"
            << "\t{n} - Сменить n\n"
            << "\t{q} - Завершить программу\n";
    cin >> c;
    return c;
}

/*Функция нахождения значений x на каждом отрезке
 *Параметры:
 *   1)lower_limit-нижний предел
 *   2)upper_limit-верхний предел
 *   3)n-число отрезков
 */
vector <double> get_x(double lower_limit, double upper_limit,
int n) {
    vector <double> x(n + 1);

    double h = (upper_limit - lower_limit) / n;  //Шаг
    x[0] = lower_limit;

    for (int i = 1; i < (n + 1); i++) {
        x[i] = x[i - 1] + h;
    }

    return x;
}

/*Функция нахождения значения функции в заданной точке*/
double f_x(double x) {
    double y = (log10(x * x + 0.8)) / (x - 1);
    return y;
}

/*Функция нахождения y на каждом отрезке
 *Параметры:
 *   1)x - вектор значений x
 */
vector <double> get_y(vector<double>x) {
    vector <double> y(x.size());

    for (int i = 0; i < (x.size()); i++) {
        y[i] = f_x(x[i]);
    }

    return y;
}

/*Функция нахождения значения второй производной в заданной
точке
*/
double second_derivative(double x) {
    double temp1, temp2, temp3, res;
```

```
	temp1 = -2*x / (x - 1) * (x * x + 0.8);
	temp2 = -((2 * x * x) / (x * x + 0.8) - 1) / (x * x + 0.8);
	temp3 = log10(x * x + 0.8) / ((x - 1) * (x - 1));

	res = 2 * (temp1 + temp2 + temp3) / (x - 1);

	return res;
}

/*Функция нахождения значения третьей производной в заданной точке
*/
double third_derivative(double x) {
	double temp1, temp2, temp3,temp4, res;

	temp1 = 3 * (2 * x * x / (x * x + 0.8) - 1) /
		   ((x - 1) * (x * x + 0.8));

	temp2 = 6 * x / ((x - 1) * (x - 1) * (x * x + 0.8));

	temp3 = 2 * x * (4 * x / (x * x + 0.8) - 3) /
		  ((x * x + 0.8) * (x * x + 0.8));

	temp4 = -3 * log10(x * x + 0.8) /
		  ((x - 1) * (x - 1) * (x - 1));

	res = 2 * (temp1 + temp2 + temp3+temp4) / (x - 1);

	return res;
}
```

Solutions.h

```
#pragma once
#include<iostream>
#include<vector>
#include"colors.h"

using namespace std;

/*Библиотека, содержащая методы нахождения интеграла и погрешности этих методов*/

/*Функция поиска интеграла методом центральных прямоугольников
 *Параметры
 *	1)x - вектор значений x
 */
double central_rectangles(vector<double>x) {
	double res=0;

	for (int i = 1; i < x.size(); i++){
		res = res + f_x((x[i - 1] + x[i]) / 2);
```

```
    }
    res = res * (x[1] - x[0]);

    cout<<Blue << "Значение интеграла, вычисленное методом"
        <<"центральных прямоугольников при n="
           << x.size() - 1 << " равно " << res << "\n";

    return res;
}

/*Функция поиска интеграла методом трапеций
 *Параметры
 *   1)h - длина отрезка (шаг)
 *   2)y - вектор значений y
 */
double trapeze(double h, vector <double> y) {
    double res = (y[0]+y[y.size()-1])/2.0;

    for (int i = 1; i < y.size()-1; i++){
         res = res + y[i];
    }
    res = res * h;

    cout<<Blue << "Значение интеграла, вычисленное методом"
        <<" трапеций  при n="<< y.size() - 1 << " равно "
        << res << "\n";

    return res;
}

/*Функция поиска интеграла методом Симпосна
 *Параметры
 *   1)h - длина отрезка (шаг)
 *   2)y - вектор значений y
 */
double Simpson(double h, vector <double> y) {
    double res=y[0]+y[y.size()-1];

    for (int i = 1; i < y.size()-1; i++){
         if (i % 2 == 1) {res = res + 4 * y[i];}
         else { res = res + 2 * y[i];}
    }
    res = res * h / 3;

    cout<<Blue << "Значение интеграла, вычисленное методом"
        <<" Симпсона  при n="<< y.size() - 1 << " равно "
        << res << "\n";

    return res;
}

/*Функция поиска погрешности метода центральных прямоугольников
 *Параметры
```

```
 *  1)lower_limit - нижний предел
 *  2)upper_limit - верхний предел
 *   3)x - вектор значений x
 */
double central_rectangles_errror(double lower_limit, double
upper_limit, vector<double>x) {
     double res=1,temp, max_d;

     max_d=abs(second_derivative(x[0]));
     for (int i = 1; i < x.size(); i++){
          temp = abs(second_derivative(x[i]));
          if (temp > max_d) { max_d = temp; }
     }

     res = (upper_limit - lower_limit) * (x[1]-x[0])
           * (x[1] - x[0]) * max_d / 24;

     cout << Blue << "Значение погрешности при вычислении"
          <<" методом центральных прямоугольников при n="
          << x.size() - 1 << " равно " << res << "\n";

     return res;
}

/*Функция поиска погрешности метода трапеций
 *Параметры
 *   1)lower_limit - нижний предел
 *  2)upper_limit - верхний предел
 *   3)x - вектор значений x
 */
double trapeze_errror(double lower_limit, double upper_limit,
vector<double>x) {
     double res = 1, temp, max_d;

     max_d = abs(second_derivative(x[0]));
     for (int i = 1; i < x.size(); i++) {
          temp = abs(second_derivative(x[i]));
          if (temp > max_d) { max_d = temp; }
     }

     res = (upper_limit - lower_limit) * (x[1] - x[0])
           * (x[1] - x[0]) * max_d / 12;

     cout << Blue << "Значение погрешности при вычислении"
          <<" трапеций  при n="
          << x.size() - 1 << " равно " << res << "\n";

     return res;
}

/*Функция поиска погрешности метода Симсона
 *Параметры
 *   1)lower_limit - нижний предел
```

```
 *  2)upper_limit - верхний предел
 *   3)x - вектор значений x
 */
double Simpson_errror(double lower_limit, double upper_limit,
vector<double>x) {
     double res = 1, temp, max_d;

     max_d = abs(third_derivative(x[0]));
     for (int i = 1; i < x.size(); i++) {
          temp = abs(third_derivative(x[i]));
          if (temp > max_d) { max_d = temp; }
     }

     res = (upper_limit - lower_limit) * (x[1] - x[0])
           * (x[1] - x[0]) * (x[1] - x[0]) * max_d / 288;

     cout << Blue << "Значение погрешности при вычислении”
          <<“ Симпсона  при n="
          << x.size() - 1 << " равно " << res << "\n";

     return res;
}
```

Main.cpp

```
#include<iostream>
#include<vector>
#include"colors.h"
#include"function.h"
#include"solutions.h"
using namespace std;
int main() {
     setlocale(LC_ALL, "rus");

     double lower_limit, upper_limit; //Пределы интегрирования

     int n;                           //Число отрезков, на
                                       //которые разделится
                                      //исходный

     vector<double>x, y;              //Значения x и y на каждом
                                      //отрезке

     char c;                          //Выбор пункта меню

     cout << Green << "Программа для вычисления интеграла функции
          "lg(x^2+0.8)/(x-1)\n";

     get_limits(&lower_limit, &upper_limit);
     n = get_n();
     x=get_x(lower_limit,upper_limit,n);
     y = get_y(x);

     bool end = false;
```

```
	while (end == false) {
		c = menu_item_selection();
		switch (c) {
			case '1':
				central_rectangles(x);
				central_rectangles_errror(lower_limit,
							upper_limit, x);
				break;
			case '2':
				trapeze(x[1]-x[0], y);
				trapeze_errror(lower_limit, upper_limit, x);
				break;
			case '3':
				Simpson(x[1] - x[0], y);
				Simpson_errror(lower_limit, upper_limit, x);
				break;
			case'n':
				n = get_n();
				x = get_x(lower_limit, upper_limit, n);
				y = get_y(x);
				break;
			case 'q':
				end = true;
				break;
			default:
				cout<< Red
					<< "Неверный ввод, повторите попытку!\n";
		}
	}

	return 0;
}
```

Результаты работы программы

```
Программа для вычисления интеграла функции lg(x^2+0.8)/(x-1)
Введите пределы интегрирования
        Нижний предел 2.5
        Верхний предел 3.3
Введите n - число отрезков разбиения 8

Выберите, что нужно сделать :
        {1} - Найти интеграл методом центральных прямоугольников
        {2} - Найти интеграл методом трапеций
        {3} - Найти интеграл по формуле Симпсона
        {n} - Сменить n
        {q} - Завершить программу
1
Значение интеграла, вычисленное методом центральных прямоугольников при n=8 равно 0.407905
Значение погрешности при вычислении методом центральных прямоугольников при n=8 равно 0.0103256

Выберите, что нужно сделать :
        {1} - Найти интеграл методом центральных прямоугольников
        {2} - Найти интеграл методом трапеций
        {3} - Найти интеграл по формуле Симпсона
        {n} - Сменить n
        {q} - Завершить программу
2
Значение интеграла, вычисленное методом трапеций при n=8 равно 0.408001
Значение погрешности при вычислении методом трапеций при n=8 равно 0.0206513

Выберите, что нужно сделать :
        {1} - Найти интеграл методом центральных прямоугольников
        {2} - Найти интеграл методом трапеций
        {3} - Найти интеграл по формуле Симпсона
        {n} - Сменить n
        {q} - Завершить программу
3
Значение интеграла, вычисленное методом Симпсона при n=8 равно 0.407937
Значение погрешности при вычислении методом Симсона при n=8 равно 0.000001

Выберите, что нужно сделать :
        {1} - Найти интеграл методом центральных прямоугольников
        {2} - Найти интеграл методом трапеций
        {3} - Найти интеграл по формуле Симпсона
        {n} - Сменить n
        {q} - Завершить программу
n
Введите n - число отрезков разбиения 20

Выберите, что нужно сделать :
        {1} - Найти интеграл методом центральных прямоугольников
        {2} - Найти интеграл методом трапеций
        {3} - Найти интеграл по формуле Симпсона
        {n} - Сменить n
        {q} - Завершить программу
1
Значение интеграла, вычисленное методом центральных прямоугольников при n=20 равно 0.407932
Значение погрешности при вычислении методом центральных прямоугольников при n=20 равно 0.001652
```

```
Выберите, что нужно сделать :
        {1} - Найти интеграл методом центральных прямоугольников
        {2} - Найти интеграл методом трапеций
        {3} - Найти интеграл по формуле Симпсона
        {n} - Сменить n
        {q} - Завершить программу
2
Значение интеграла, вычисленное методом трапеций при n=20 равно 0.407947
Значение погрешности при вычислении методом трапеций при n=20 равно 0.003304

Выберите, что нужно сделать :
        {1} - Найти интеграл методом центральных прямоугольников
        {2} - Найти интеграл методом трапеций
        {3} - Найти интеграл по формуле Симпсона
        {n} - Сменить n
        {q} - Завершить программу
3
Значение интеграла, вычисленное методом Симпсона при n=20 равно 0.407937
Значение погрешности при вычислении методом Симсона при n=20 равно 0.000000

Выберите, что нужно сделать :
        {1} - Найти интеграл методом центральных прямоугольников
        {2} - Найти интеграл методом трапеций
        {3} - Найти интеграл по формуле Симпсона
        {n} - Сменить n
        {q} - Завершить программу
q

C:\Users\Валерий\Desktop\Учёба\Дз\вычмат\ЛР4\Project1\Debug\Project1.exe (пр
Нажмите любую клавишу, чтобы закрыть это окно…
```

Вывод

В ходе данной работы были закреплены знания и умения по вычислению интеграла при помощи методов средних прямоугольников, трапеция и методу Симпсона.

Самым точным является метод Симпсона, а самым не точным метод трапеций. При увеличении кол-во промежутков разбиения n точность результатов возрастает.