Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное  
учреждение высшего образования

Новосибирский государственный технический университет

Кафедра Автоматизированных систем управления

**Отчет по лабораторной работе № 5**

по дисциплине «Вычислительная математика»

Выполнили:

Букова Анна

Ланг Татьяна

Пустовских Дмитрий

Группа: АВТ-813

Преподаватель:

Лукоянычев

Алексей Викторович

Новосибирск

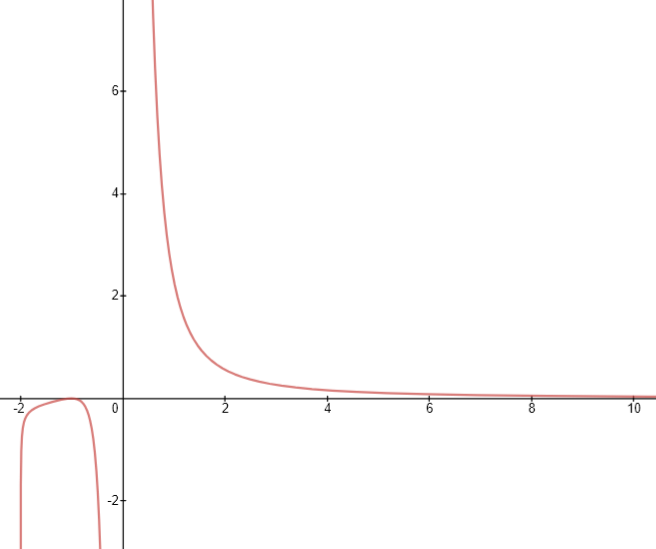
2020

1. **Исходные данные**

Функция:

Интервал интегрирования: [ 1.4 ; 5.0]

График:



1. **Результаты вычислений**

**2.1. Средних прямоугольников.**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Точность (e) | Шаг | Кол.интервалов | Значения | Кол.итераций |
| 0.1 | 0.9 | 4 | 1.151 | 2 |
| 0.01 | 0.225 | 16 | 1.19524 | 4 |
| 0.001 | 0.05625 | 64 | 1.19841 | 6 |

**2.2. Трапеций.**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Точность (e) | Шаг | Кол.интервалов | Значения | Кол.итераций |
| 0.1 | 0.45 | 8 | 1.22536 | 3 |
| 0.01 | 0.1125 | 32 | 1.20033 | 5 |
| 0.001 | 0.028125 | 128 | 1.19873 | 7 |

**2.3. Ньютона-Котеса 3-го порядка.**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Точность (e) | Шаг | Кол.интервалов | Значения | Кол.итераций |
| 0.1 | 0.6 | 6 | 1.20764 | 1 |
| 0.01 | 0.3 | 12 | 1.19952 | 2 |
| 0.001 | 0.15 | 24 | 1.19869 | 3 |

1. **Выводы**

В данной лабораторной работе мы познакомились с различными методами численного интегрирования. Также практически реализовали методы на языке С++. Рассмотрели скорость методов при разных точностях. Из полученных данных можно сделать вывод, что при увеличении степени интерполяции погрешность становится меньше. Практически убедились, что ошибка в методе трапеций в 2 раза больше , чем у метода средних прямоугольников.

1. **Тексты программ**

#include <iostream>

#include <Math.h>

#include <locale>

using namespace std;

int iteration;

double Fx(double x)

{

double y;

y = pow(x + 1, 2) / (pow(x, 3) \* sqrt(2 + x));

return y;

}

double mediumRectangles(double (\*F)(double), double a, double b, double e)

{

double h = 2, step, S1, S2, sum, x1 , x2;

iteration = 0;

S1 = (b - a) \* F((a + b) / 2);

while (true)

{

iteration++;

sum = 0;

x1 = a;

step = (b - a) / h;

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

{

x2 = x1 + step;

sum += F((x1 + x2) / 2);

x1 = x2;

}

S2 = step \* sum;

if (abs(S2 - S1) < e)

{

cout << "Шаг : " << step << " кол. интервалов : " << h << endl;

return S2;

}

S1 = S2;

h = h \* 2;

}

}

double trapezoid(double (\*F)(double), double a, double b, double e)

{

double h = 2, step, S1, S2, sum, x1;

iteration = 0;

S1 = ((b - a)/2) \*(F(a) + F(b));

while (true)

{

iteration++;

sum = 0;

x1 = a;

step = (b - a) / h;

for (int i = 0;i < h - 1; ++i)

{

x1 += step;

sum += F(x1);

}

S2 = (step / 2) \* (F(a) + 2 \* sum + F(b));

if (abs(S2 - S1) < e)

{

cout << "Шаг : " << step <<" кол. интервалов : " << h << endl;

return S2;

}

S1 = S2;

h = h \* 2;

}

}

double Simpson(double (\*F)(double), double a, double b, double e)

{

double h = 3, step, S1, S2, sum, x1;

double C0 = 3. / 8.;

iteration = 0;

step = (b - a) / h;

sum = (F(a) + 3 \* F(a + step) + 3 \* F(a + step + step) + F(b));

S1 = C0 \* step \* sum;

h = h \* 2;

while (true)

{

iteration++;

x1 = a;

sum = 0;

step = (b - a) / h;

for (int i = 0; i < h / 3; ++i)

{

sum += (F(x1) + 3 \* F(x1 + step) + 3 \* F(x1 + step \* 2) + F(x1 + step \* 3));

x1 = x1 + step \* 3;

}

S2 = C0 \*step \*sum;

if (abs(S2 - S1) < e)

{

cout << "Шаг : " << step << " кол. интервалов : " << h << endl;

return S2;

}

S1 = S2;

h = h \* 2;

}

}

int main()

{

setlocale(LC\_ALL, "ru");

double e = 0.1;

cout << "Трапеций : " << trapezoid(Fx, 1.4, 5, e) << endl;

cout << "Количество итераций : " << iteration << endl << endl;

cout << "Средних прямоугольников : " << mediumRectangles(Fx, 1.4, 5, e) << endl;

cout << "Количество итераций : " << iteration << endl << endl;

cout << "Ньютона-Котеса 3-го порядка : " << Simpson(Fx, 1.4, 5, e) << endl;

cout << "Количество итераций : " << iteration << endl << endl;

cout << endl;

}