|  |  |
| --- | --- |
|  | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТ Информатики и систем управления

КАФЕДРА Теоретической информатики и компьютерных технологий

**Лабораторная работа № 2**

«Сравнительный анализ методов численного интегрирования»

по курсу «Численные методы»

Выполнила:

студент группы ИУ9-61Б

Александрова Ольга

Проверила:

Домрачева А. Б.

Москва, 2024

1. **Цель**

Целью данной работы является сравнение по быстродействию методов численного интегрирования:

1. Метод центральных прямоугольников
2. Метод трапеций
3. Метод Симпсона
4. **Постановка задачи**

**Дано:** Интеграл

где – подынтегральная функция, непрерывная на отрезке

**Найти:** Значение интеграла

При заданной точности

**Индивидуальный вариант:** ,

1. **Основные теоретические сведения**
   1. **Метод центральных прямоугольников**

Метод заключается в вычислении площади под графиком подынтегральной функции с помощью суммирования площадей прямоугольников, ширина которых определяется шагом разбиения (расстояние между узлами интегрирования), а высота – значением подынтегральной функции в узле интегрирования.

Пусть требуется определить значение интеграла функции на отрезке . Тогда отрезок разбивается на равных отрезков длиной . Получаем разбиение данного отрезка точками

Тогда приближенное значение интеграла на всем отрезке будет равно:

* 1. **Метод трапеций**

Метод заключается в вычислении площади под графиком подынтегральной функции с помощью суммирования площадей трапеций, высота которых определяется шагом разбиения (расстояние между узлами интегрирования), а высота – значением подынтегральной функции в узле интегрирования.

Пусть требуется определить значение интеграла функции на отрезке . Тогда отрезок разбивается на равных отрезков длиной . Получаем разбиение данного отрезка точками

Тогда приближенное значение интеграла на всем отрезке будет равно:

* 1. **Метод Симпсона**

Метод заключается в приближении функции на отрезке интерполяционным многочленом 2 степени функции

Тогда приближенное значение интеграла на всем отрезке будет равно:

* 1. **Уточнение значения интеграла по Ричардсону**

, где – порядок точности метода, – приближенное значение интеграла, вычисленного с помощью метода с шагом .

Для метода средних прямоугольников и метода трапеций .

Для метода Симпсона .

, где – некоторая константа, – шаг.

Считаем, что вычисления проводятся без вычислительной погрешности, тогда можно записать строгое равенство для шага

для шага

Из равенств получаем уточненное значение интеграла:

Где значение – уточнение по Ричардсону:

Данная величина используется для компенсации методологической погрешности численных методов интегрирования.  
 Чтобы построить процедуру приближенного вычисления интеграла с заданной точностью , используется правило Рунге:

1. **Реализация**

Листинг 1. Численное интегрирование

#include <iostream>  
#include <cmath>  
#include <iomanip>  
  
double myFunction(double x) {  
 return exp(sqrt(x));  
}  
  
double method\_of\_trapezoid(double a = 0, double b = 4, int n = 2) {  
 double h = (b - a) / n;  
 double series\_of\_sum = 0;  
  
 for (int i = 0; i < n - 1; i++) {  
 double x = a + (i + 1) \* h;  
 series\_of\_sum += myFunction(x);  
 }  
  
 return h \* ( myFunction(a) + myFunction(b) / 2 + series\_of\_sum);  
}  
  
  
double method\_of\_simpson(double a = 0, double b = 4, int n = 2) {  
 double h = (b - a) / n;  
 double series\_of\_sum = 0;  
  
 for (int i = 0; i < n; i++) {  
 double x1 = a + (i + 0.5) \* h;  
 double x2 = a + (i + 1) \* h;  
  
 series\_of\_sum += 4 \* myFunction(x1);  
 if (i < n - 1) {  
 series\_of\_sum += 2 \* myFunction(x2);  
 }  
 }  
  
 return h / 3 \* ( myFunction(a) + myFunction(b) + series\_of\_sum);  
}  
  
double method\_of\_rectangles(double a = 0, double b = 4, int n = 2) {  
 double h = (b - a) / n;  
 double result = 0;  
  
 for (int i = 0; i < n; i++) {  
 double x = a + (i + 0.5) \* h;  
 result += h \* myFunction(x);  
 }  
  
 return result;  
}  
  
double get\_richardson(double i\_n, double i\_n2, int k) {  
 return (i\_n - i\_n2) / (2 \* k - 1);  
}  
  
  
int main() {  
 double eps = 1e-3;  
 std::cout << std::setw(20) << "Метод трапеций " << std::setw(20) << "Метод прямоугольников" << std::setw(30) << "Метод Симпсона" << std::endl;  
  
 std::vector<double> N;  
 std::vector<double> Result;  
 std::vector<double> Richardson;  
  
 int count = 0;  
  
 for (int j = 0; j < 3; j++) {  
 int n = 1;  
 double richardson = INFINITY;  
 int iteration = 0;  
 double i\_n = 0, i\_n2 = 0;  
  
 while (std::abs(richardson) >= eps) {  
 n \*= 2;  
 i\_n2 = i\_n;  
 if (count == 1) {  
 i\_n = method\_of\_simpson(0, 4, 4);  
 richardson = get\_richardson(i\_n, i\_n2, 4);  
 } else if (count == 0) {  
 i\_n = method\_of\_trapezoid(0, 4, 2);  
 richardson = get\_richardson(i\_n, i\_n2, 2);  
 } else {  
 i\_n = method\_of\_rectangles(0, 4, 2);  
 richardson = get\_richardson(i\_n, i\_n2, 2);  
 }  
 iteration++;  
 }  
 count++;  
 N.push\_back(iteration);  
 Result.push\_back(i\_n);  
 Richardson.push\_back(i\_n + richardson);  
 }  
 std::cout << std::setw(2) << std::fixed << std::setprecision(3) << "N)" << std::setw(9) << N[0] << std::setw(15) << N[1] << std::setw(25) << N[2]<< std::setw(30) << std::endl;  
 std::cout << std::setw(2) << std::fixed << std::setprecision(3) << "Res)" << std::setw(7) << Result[0] << std::setw(15) << Result[1] << std::setw(25) << Result[2]<< std::setw(30) << std::endl;  
 std::cout << std::setw(2) << std::fixed << std::setprecision(3) << "Rich)" << std::setw(6) << Richardson[0] << std::setw(15) << Richardson[1] << std::setw(25) << Richardson[2]<< std::setw(30) << std::endl;  
  
  
 return 0;  
}

1. **Результаты**

Для тестирования выбран интеграл:

В качестве были выбраны следующие значения:

Таблица 1 – Результаты программы

|  |  |  |  |
| --- | --- | --- | --- |
| Метод | Кол-во итераций | Значение интеграла | Значение интеграла с уточнением по Ричардсону |
|  | | | |
| Метод центральных прямоугольников | |  |  | | --- | --- | |  |  |   4 | 0.36725322733841864 | 0.36684480994171986 |
| Метод трапеций | 4 | 0.36604553395806505 | 0.3668565158148817 |
| Метод Симпсона | 2 | 0.3669524667895847 | 0.3668393717934952 |
|  | | | |
| Метод центральных прямоугольников | 6 | 0.36687538006255127 | 0.3668502538890432 |
| Метод трапеций | 6 | 0.36680006961565864 | 0.36685029927146423 |
| Метод Симпсона | 3 | 0.3668565158148816 | 0.3668501190832347 |
|  | | | |
| Метод центральных прямоугольников | 7 | 0.366856550324328 | 0.36685027374492024 |
| Метод трапеций | 8 | 0.3668471375817167 | 0.3668502751625873 |
| Метод Симпсона | 3 | 0.3668565158148816 | 0.3668501190832347 |
|  |  |  |  |

1. **Вывод**

В ходе выполнения лабораторной работы были рассмотрены 3 метода численного интегрирования: метод центральных прямоугольников, метод трапеций и метод Симпсона. Данные методы были реализованы на языке программирования С++.

Самым точным среди рассмотренных трех методов оказался метод Симпсона.

Анализируя оставшиеся два метода, приходим к выводу, что метод средних прямоугольников точнее метода трапеций.