|  |  |
| --- | --- |
|  | **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»

Выполнил:

студент группы ИУ9-62Б

Марченко Андрей

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 <vector>  
#include <iomanip>  
  
double func(double x) {  
 return std::log(x) \* std::log(x) / x;  
}  
  
const double exact\_value = 2.0 / 3.0;  
  
double rect\_method(double a, double b, int n) {  
 double h = (b - a) / n;  
 double res = 0;  
 for (int i = 1; i <= n; i++) {  
 res += func(a + i \* h + h / 2);  
 }  
 return res \* h;  
}  
  
double trapezoid\_method(double a, double b, int n) {  
 double h = (b - a) / n;  
 double res = 0;  
 for (int i = 1; i <= n - 1; i++) {  
 res += func(a + i \* h);  
 }  
 return ((func(a) + func(b)) / 2 + res) \* h;  
}  
  
double simpson\_method(double a, double b, int n) {  
 double h = (b - a) / n;  
 double sum = 0;  
 for (int i = 1; i <= n - 1; i++) {  
 sum += 2 \* func((a + h \* i)) \* ((i % 2) + 1);  
 }  
  
 return (h / 3.0) \* (func(a) + func(b) + sum);  
}  
  
double richardson(double Ih, double Ih2, int p) {  
 return (Ih - Ih2) / (pow(2, p) - 1);  
}  
  
  
int main() {  
 double a = 1.0 / std::numbers::e;  
 double b = std::numbers::e;  
 double eps = 0.001;  
  
 std::vector <int> vector\_n;  
 std::vector <double> vector\_Ih2;  
 std::vector <double> vector\_R;  
  
 double error = 1000;  
 int n = 1;  
 double Ih = 0, Ih2 = 0;  
 while (std::abs(error) > eps) {  
 n \*= 2;  
 Ih = Ih2;  
 Ih2 = rect\_method(a, b, n);  
 error = richardson(Ih, Ih2, 2);  
 }  
 vector\_n.push\_back(n);  
 vector\_Ih2.push\_back(Ih2);  
 vector\_R.push\_back(error);  
  
 error = 1000;  
 n = 1;  
 Ih = 0, Ih2 = 0;  
 while (std::abs(error) > eps) {  
 n \*= 2;  
 Ih2 = Ih;  
 Ih = trapezoid\_method(a, b, n);  
 error = richardson(Ih, Ih2, 2);  
 }  
 vector\_n.push\_back(n);  
 vector\_Ih2.push\_back(Ih2);  
 vector\_R.push\_back(error);  
  
 error = 1000;  
 n = 1;  
 Ih = 0, Ih2 = 0;  
 while (std::abs(error) >= eps) {  
 n \*= 2;  
 Ih2 = Ih;  
 Ih = simpson\_method(a, b, n);  
 error = richardson(Ih, Ih2, 4);  
 }  
  
 vector\_n.push\_back(n);  
 vector\_Ih2.push\_back(Ih2);  
 vector\_R.push\_back(error);  
  
 int width = 15;  
  
 std::cout << '\t' << std::setw(width) << "Rectangle method" << '\t' << "Trapezoid method" << '\t' << "Simpson method" << std::endl;  
 std::cout << "n: ";  
 for (int i = 0; i < vector\_n.size(); i++) {  
 std::cout << std::setw(width) << vector\_n[i];  
 }  
 std::cout << std::endl << "I(h/2): ";  
 for (int i = 0; i < vector\_Ih2.size(); i++) {  
 std::cout << std::setw(width) << vector\_Ih2[i];  
 }  
 std::cout << std::endl << "R: ";  
 for (int i = 0; i < vector\_R.size(); i++) {  
 std::cout << std::setw(width) << vector\_R[i];  
 }  
 std::cout << std::endl << "I(h/2) + R: ";  
 for (int i = 0; i < vector\_R.size(); i++) {  
 std::cout << std::setw(width) << vector\_R[i] + vector\_Ih2[i];  
 }  
 std::cout << std::endl << "Expected: ";  
 for (int i = 0; i < vector\_n.size(); i++) {  
 std::cout << std::setw(width) << exact\_value;  
 }  
  
}

1. **Результаты**

Для тестирования выбран интеграл:

В качестве были выбраны следующие значение:

**Таблица 1 – Результаты программы**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Метод | n | Значение интеграла I(h/2) | R | I(h/2) + R | Аналитически вычисленное значение |
|  |  | | | |  |
| Метод центральных прямоугольников | |  |  | | --- | --- | |  |  |   2048 | 0.663983 | -0.000886 | 0.663097 | 0.666667 |
| Метод трапеций | 128 | 0.669168 | -0.000624 | 0.668543 | 0.666667 |
| Метод Симпсона | 32 | 0.670555 | -0.000238 | 0.670317 | 0.666667 |

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.