

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ**

**ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ  
«ДОНСКОЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

**(ДГТУ)**

Факультет «Информатика и вычислительная техника»

наименование факультета

Кафедра «Математика и информатика»

наименование кафедры

**Отчет**

по «Численным методам интегрирования»

Выполнил студент

группы МСК11

Леонавичус Даниил

Ростов-на-Дону

2022

Задача: решить интеграл с помощью методов численного интегрирования.

1. Метод прямоугольников

Метод численного 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. Алгебраический порядок точности равен 0. (Для формулы средних прямоугольников равен 1).

1. 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. Алгебраический порядок точности равен 1.

1. Метод Симпсона

Он определяет замену

Программная реализация данных методов

Метод прямоугольников

import 'dart:math';

// Функция, которую мы интегрируем

double f(double x) {

return x \* x \* sqrt(16 - x \* x);

}

// Метод прямоугольников: левые прямоугольники

double leftRectangleMethod(double a, double b, int n) {

// Вычисляем ширину каждого прямоугольника

double h = (b - a) / n;

double integral = 0;

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

double x = a + i \* h;

integral += f(x) \* h;

}

return integral;

}

// Метод прямоугольников: правые прямоугольники

double rightRectangleMethod(double a, double b, int n) {

// Вычисляем ширину каждого прямоугольника

double h = (b - a) / n;

double integral = 0;

for (int i = 1; i <= n; i++) {

double x = a + i \* h;

integral += f(x) \* h;

}

return integral;

}

// Метод прямоугольников: центральные прямоугольники

double centralRectangleMethod(double a, double b, int n) {

// Вычисляем ширину каждого прямоугольника

double h = (b - a) / n;

double integral = 0;

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

double x = a + (i + 0.5) \* h;

integral += f(x) \* h;

}

return integral;

}

void main() {

// Границы интегрирования

double a = 2;

double b = 4;

// Количество прямоугольников (чем больше, тем точнее результат)

int n = 1000;

// Вычисляем интегралы с помощью трех разных методов

double leftIntegral = leftRectangleMethod(a, b, n);

double rightIntegral = rightRectangleMethod(a, b, n);

double centralIntegral = centralRectangleMethod(a, b, n);

// Выводим результаты на экран

print('Значение интеграла (левые прямоугольники): $leftIntegral');

print('Значение интеграла (правые прямоугольники): $rightIntegral');

print('Значение интеграла (центральные прямоугольники): $centralIntegral');

// Вычисление погрешности

double h = (b - a) / n;

double xi = a + Random().nextDouble() \* (b - a); // Случайная точка на отрезке

double secondDerivative = 16.0 \* xi \* xi / (pow(16 - xi \* xi, 1.5)) - 2.0; // Вторая производная функции

double error = (((b - a) \* h \* h \* secondDerivative) / 24.0).abs();

print('Погрешност: ${error.toStringAsFixed(6)}');

}

Метод трапеций

import 'dart:math';

// Функция, которую мы интегрируем

double f(double x) {

return x \* x \* sqrt(16 - x \* x);

}

// Метод трапеций для вычисления интеграла

double trapezoidalMethod(double a, double b, int n) {

// Вычисляем ширину каждой трапеции

double h = (b - a) / n;

double integral = 0;

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

double x0 = a + i \* h;

double x1 = a + (i + 1) \* h;

integral += (f(x0) + f(x1)) / 2 \* h;

}

return integral;

}

void main() {

// Границы интегрирования

double a = 2;

double b = 4;

// Количество трапеций (чем больше, тем точнее результат)

int n = 1000;

// Вычисляем интеграл с помощью метода трапеций

double integral = trapezoidalMethod(a, b, n);

// Выводим результат на экран

print('Значение интеграла (метод трапеций): $integral');

// Вычисление погрешности

double h = (b - a) / n;

double xi = a + Random().nextDouble() \* (b - a); // Случайная точка на отрезке

double secondDerivative = 16.0 \* xi \* xi / pow(16 - xi \* xi, 1.5) - 2.0; // Вторая производная функции

double error = (((b - a) \* h \* h \* secondDerivative) / 12.0).abs();

print('Погрешность метода трапеций: ${error.toStringAsFixed(6)}');

}

Метод Симпсона

import 'dart:math';

// Функция, которую мы интегрируем

double f(double x) {

return x \* x \* sqrt(16 - x \* x);

}

// Метод Симпсона для вычисления интеграла

double simpsonsRule(double a, double b, int n) {

// Проверка на четность количества подинтервалов n

if (n % 2 != 0) {

throw ArgumentError("Количество подинтервалов n должно быть четным.");

}

// Вычисляем ширину каждого подинтервала

double h = (b - a) / n;

double integral = 0;

// Вычисляем сумму по формуле Симпсона

for (int i = 1; i < n; i++) {

double x = a + i \* h;

integral += (i % 2 == 0) ? 2 \* f(x) : 4 \* f(x);

}

// Умножаем на h/3 по формуле Симпсона, чтобы получить интеграл

integral \*= h / 3;

return integral;

}

void main() {

// Границы интегрирования

double a = 2;

double b = 4;

// Количество подинтервалов (чем больше, тем точнее результат)

int n = 1000;

// Вычисляем интеграл с помощью метода Симпсона

double integral = simpsonsRule(a, b, n);

// Выводим результат на экран

print('Значение интеграла (метод Симпсона): $integral');

// Вычисление погрешности

double h = (b - a) / n;

double xi = a + Random().nextDouble() \* (b - a); // Случайная точка на отрезке

double fourthDerivative =

-16.0 \* (3 \* xi \* xi - 8) \* (3 \* xi \* xi - 4) / pow(16 - xi \* xi, 2.5); // Четвертая производная функции

double error = (((b - a) \* h \* h \* h \* h \* fourthDerivative) / 2880.0).abs();

print('Погрешность метода Симпсона: ${error.toStringAsFixed(6)}');

}

Вывод: были написаны программы, которые реализуют вычисление интеграла с помощь методов численного интегрирования, они могут помочь в реализации тяжелых интегралов и дать максимально точный ответ.