МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

**«МОСКОВСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

**(МОСКОВСКИЙ ПОЛИТЕХ)**

«Интеграция и программирование в САПР»

Кафедра «СМАРТ технологии»

ОТЧЕТ

по дисциплине:

**Проектная деятельность**

на тему:

Решение интегральных уравнений методами: сумма Римана, правило трапеций, правило Симпсона

Построение изолиний для функций двух неизвестных.

Преподаватель: / Толстиков А.В., к.т.н. /

*подпись ФИО, уч. звание и степень*

Студент: / Пересторонин А.М. 201-324 /

*подпись ФИО, группа*

Москва, 2022 г.

**ТЕОРЕТИЧЕСКАЯ ЧАСТЬ**

Сумма Римана — один из механизмов определения интеграла через сумму вида Используется в определении интеграла Римана. Названа в честь первооткрывателя, Бернхарда Римана.

Пусть является функцией определённой на подмножестве D на вещественной прямой R. — замкнутый интервал содержащийся в D. [ является разбиением I, в котором

Сумма Римана функции F с разбиением P определяется следующим образом:

где  Выбор  в данном интервале является произвольным. Если для всех I тогда S называется левой суммой Римана. Еслитогда S называется правой суммой Римана. Если тогда S называется средней суммой Римана. Усреднённое значение левой и правой суммы Римана называется трапециевидной суммой. Если Сумма

Римана представляется в виде:

где  является точной верхней границей множества f на интервале , то S называется верхней суммой Римана. Аналогично, если является точной нижней границей множества f интервале то S называется нижней суммой Римана.

Любая сумма Римана с заданным разбиением (при выборе любого значения из интервала ) находится между нижней и верхней суммами Римана.

Если для функции f и отрезка [a,b] существует предел сумм Римана, когда шаг разбиения стремится к нулю (независимо от выбора ), то этот предел называют интегралом Римана функции f на отрезке [a,b] и обозначается Визуализация представлена на рисунке 1.1.

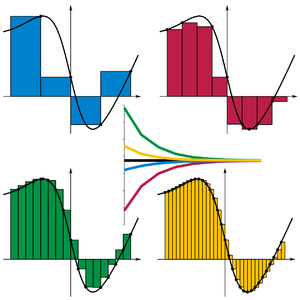


Рисунок 1.1 – Сумма Римана

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.

Если отрезок [a,b] является элементарным и не подвергается дальнейшему разбиению, значение интеграла можно найти по формуле.

Это простое применение формулы для площади трапеции — произведение полусуммы оснований, которыми в данном случае являются значения функции в крайних точках отрезка, на высоту (длину отрезка интегрирования). Погрешность аппроксимации для элементарного отрезка можно оценить через максимум второй производной (для случаев разбиения отрезка на *n* частей см. составные формулы ниже). Визуализация представлена на рисунке 1.2.

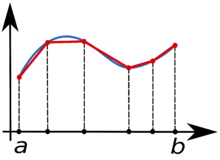


Рисунок 1.2 – Метод трапеций

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 , то есть приближение графика функции на отрезке параболой. Метод Симпсона имеет порядок погрешности 4 и алгебраический порядок точности 3.

Формула

Формулой Симпсона называется интеграл от интерполяционного многочлена второй степени на отрезке [a,b].

где  и — значения функции в соответствующих точках (на концах отрезка и в его середине).

Погрешность

При условии, что у функции  на отрезке [a,b] существует четвёртая производная, погрешность согласно найденной Джузеппе Пеано формуле, равна:

В связи с тем, что значение зачастую неизвестно, для оценки погрешности используется следующее неравенство:

Визуализация представлена на рисунке 1.3.

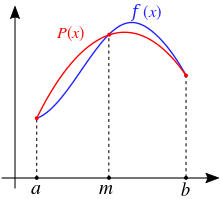


Рисунок 1.3 – Формула Симпсона

**ПРАКТИЧЕСКАЯ ЧАСТЬ**

Программа была выполнена в консольном приложении, это связано с тем что методы описанные выше не могут применятся как конечный продукт в связи с тем что они представляют из себя математические методы компьютерного анализа математических формул.

Для начала нам необходимо создать функцию уравнения. То есть функцию на вход которой подается значение соответствующие значению одной переменой, а на выходе возвращается значение функции в точке входного параметра.

double f(double x)  
{  
return Math.Pow(x, 2) / (Math.Pow(x, 2) + 1);  
}

Примечание: для введения своей функции можно использовать динамическую компиляцию кода в стеке. Этот механизм был продемонстрирован в отчете про “изолинии”. Однако такой подход сопровождается существенной задержкой в расчетах, решение которого представляется в оптимизации средств компиляции.

Функция с кодом Сумма Риманапредставлена ниже**.** Описание кода не требуется, так как код полностью копирует математическую формулу, описанную в теоретической части выше.

double riemann\_sum(double n, double a, double b)  
{  
double h = Math.Abs(b - a) / n;  
double f\_sum = f(a);  
for (int i = 1; i < n; i++)  
{  
f\_sum += f(a + i \* h);  
}  
return h \* f\_sum;  
}

Примечание: для представления понимания входных данных: a, b => диапазоны функции, n => шаг функции. Чем ниже шаг, тем точнее результат выполнения функции.

Функция с кодом Правило трапеций аналогично с предыдущей функцией описана выше и полностью копирует формулу из теоретического аспекта отчета.

double trapezoidal\_rule(double n, double a, double b)  
{  
double h = Math.Abs(b - a) / n;  
double f\_sum = f(a) + f(b);  
for (int i = 1; i < n; i++)  
{  
f\_sum += 2 \* f(a + i \* h);  
}  
return h \* f\_sum / 2;  
}

Примечание: входные параметры аналогично предыдущей функции и для удобства пользованием приведены в аналогичном порядке.

Функция с кодом правило Симпсона по аналогии с предыдущим не требует дополнительных пояснений.

double simpsons\_rule(double n, double a, double b)  
{  
double h = Math.Abs(b - a) / n;  
double f\_sum = f(a) + f(b);  
for (int i = 1; i < n; i++)  
{  
f\_sum += (2 + 2 \* (i % 2)) \* f(a + i \* h);  
}  
return h \* f\_sum / 3;  
}

Пример работы программы представлен на рисунке 1.

Для его работы были написаны дополнительные функции вызова математических методов.

double A = = Convert.ToDouble(Console.ReadLine());  
double B = = Convert.ToDouble(Console.ReadLine());  
double n = Convert.ToDouble(Console.ReadLine());  
  
Console.WriteLine(riemann\_sum(n, A, B)); // сумма Римана  
Console.WriteLine(trapezoidal\_rule(n, A, B)); // Правило трапеций  
Console.WriteLine(simpsons\_rule(n, A, B)); // правило Симпсона

Пример работы программы представлен на рисунке 1.

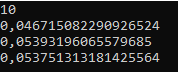


Рисунок 1. – Пример работы программы.

**РЕЗУЛЬТАТЫ**

В результате работы сделана программа по расчету уравнения методом: сумма Римана, Правило трапеций, правило Симпсона. На экран выводятся результаты расчета математических методов.

ПРИЛОЖЕНИЕ 1. Код файла \*.cs

|  |
| --- |
| double f(double x) {  return Math.Pow(x, 2) / (Math.Pow(x, 2) + 1); }  double riemann\_sum(double n, double a, double b) {  double h = Math.Abs(b - a) / n;  double f\_sum = f(a);  for (int i = 1; i < n; i++)  {  f\_sum += f(a + i \* h);  }  return h \* f\_sum; }  double trapezoidal\_rule(double n, double a, double b) {  double h = Math.Abs(b - a) / n;  double f\_sum = f(a) + f(b);  for (int i = 1; i < n; i++)  {  f\_sum += 2 \* f(a + i \* h);  }  return h \* f\_sum / 2; }  double simpsons\_rule(double n, double a, double b) {  double h = Math.Abs(b - a) / n;  double f\_sum = f(a) + f(b);  for (int i = 1; i < n; i++)  {  f\_sum += (2 + 2 \* (i % 2)) \* f(a + i \* h);  }  return h \* f\_sum / 3; }  double A = 0; double B = 1 / Math.Sqrt(3);  double n = Convert.ToDouble(Console.ReadLine());  Console.WriteLine(riemann\_sum(n, A, B)); // сумма Римана Console.WriteLine(trapezoidal\_rule(n, A, B)); // Правило трапеций Console.WriteLine(simpsons\_rule(n, A, B)); // правило Симпсона |