МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«САМАРСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ

ИМЕНИ АКАДЕМИКА С.П.КОРОЛЕВА»

Факультет информатики

Кафедра информационных систем и технологий

Отчет по теме

«Вычисление ортогональных функций Якоби»

Выполнили:

студенты гр. 6224-090401D

Сурков А.В.

Чурсин П.О.

Кузьмин И.В.

Воробьев Д.А.

Проверил:

профессор кафедры ИСТ

Прохоров С.А.

Самара 2017 г. **ИСХОДНЫЕ ДАННЫЕ**

На рисунках 1-3 представлены функции которые были использованы для анализа:

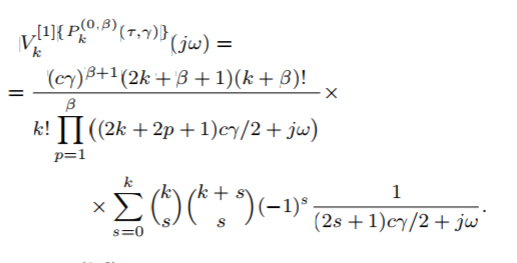


Рисунок 1 – Функция Якоби 5.72

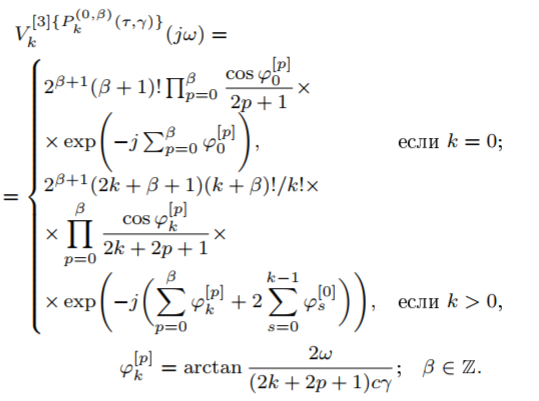


Рисунок 1 – Функция Якоби 5.73

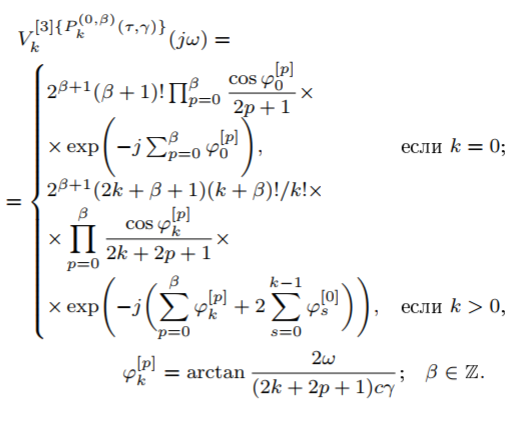


Рисунок 1 – Функция Якоби 5.74

**ОСОБЕННОСТИ РЕАЛИЗАЦИИ ВЫЧИСЛЕНИЙ**

Для работы с комплексными числами был создан класс Complex. Он содержит в себе все необходимые инструменты для удобной работы в конкретном случае и не только.

За заботу с ортонормированными функциями отвечает абстрактный класс Ortho и три класса наследующих его, которые реализуют конкретный метод. Кроме того, в классе Ortho также присутствуют вспомогательные методы, необходимые для решения.

**Вычисление факториала**:

За эту операцию отвечает метод fact (рисунок 4), возвращающий значение типа Double и принимающий в качестве аргументов 2 параметра: числитель и знаменатель. Таким образом появляется возможность сократить определённое количество множителей в зависимости от того, какой параметр больше. Это снижает вычислительные расходы, а также уменьшает вероятность переполнения при больших значениях.

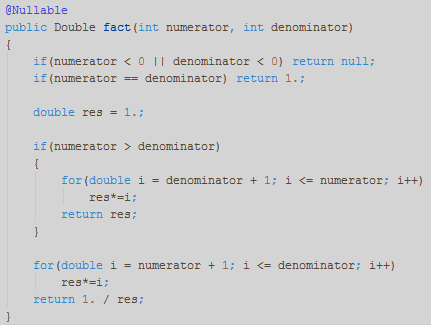


Рисунок 4 – метод вычисляющий частное двух факториалов

**Вычисление произведения сочетаний:**

В одной из формул присутствует следующая операция: . Данное выражение можно переписать сократив отдельные множители в числителе и знаменателе: . Здесь можно увидеть, что в зависимости от значения s и k в обоих частях дроби можно сократить определённое количество множителей. За это отвечает метод combinations (рисунок 5), принимающий в качестве аргументов k и s и возвращающий значение тапа Double.

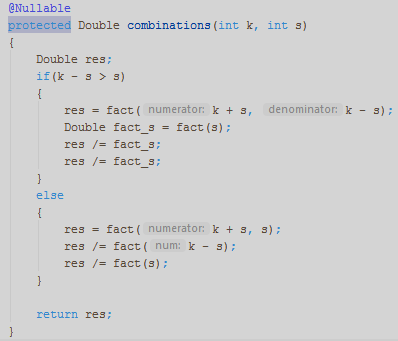


Рисунок 5 – метод, вычисляющий произведение двух конкретных сочетаний

Как видно их кода на рисунке 2, в зависимости от того, какой из делителей больше в метод fact передаются разные аргументы, стремясь уменьшить потенциально большое число в числителе. Затем результат разделяется на оставшиеся делители.

**Вычисление ∆ω и N для конкретных значений k, β, γ и ширины коридора δ**

Для вычисления этих значений сперва требуется найти – максимальное значение , при котором функция попадёт в коридор и больше с него не выйдет. Для этого реализовано 2 алгоритма, которые работают последовательно.

Первый алгоритм сначала проверяет лежит ли текущее значение функции в заданном коридоре, и если это так, то необходимо запомнить знак производной функции. Если в течение последующих шагов значения не выйдут за пределы коридора и производная поменяет свой знак – значить функция вошла в коридор.

Второй алгоритм считает количество пересечений функции нуля. Если оно больше или равно k, и при этом значение функции лежит в пределах коридора, значит она уже не выйдет за его пределы.

Эффективность первого алгоритма проявляется при больших значениях k, потому как функция может продолжать колебаться уже зайдя в коридор. Второй же алгоритм напротив более эффективен при малых k, потому как изменение функции очень плавное, и уже после пересечения ей нуля k раз знак производной может поменяться ещё не скоро.

Как только один из алгоритмов обнаруживает вхождение, считается найденным и другие искомые величины рассчитываются по формулам.

Исходный код этого метода представлен на рисунке 6. Он принимает в качестве аргумента значение d, задающее ширину коридора, и возвращает объект класса Threshold. Он содержит в себе 3 поля (рисунок 7), два из которых и являются искомыми величинами.

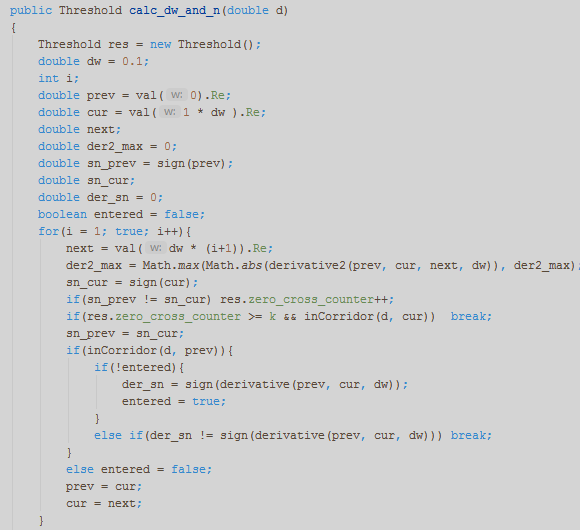


Рисунок 6 – исходный код метода, вычисляющего ∆ω и N

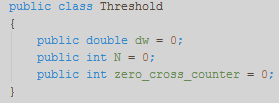


Рисунок 7 – структура класса Threshold

**РЕАЛИЗАЦИЯ**

**Язык программирования**: Java

Web-интерфейс представляет собой html-страницу. Обмен данными с сервером осуществляется с помощью REST-API.

**Характеристики компьютера:**

* Архитектура операционной системы: 64-bit
* Количество ядер процессора: 4
* Количество потоков: 8
* Объем ОЗУ: 12 ГБ
* Процессор: Intel Core i7 3610QM
* Частота процессора: 2.30 GHz
* Операционная система: Windows 10

**Распараллеливание:**

Распределенные алгоритмы были реализованы при помощи библиотеки Apache Spark. Каждая реализация ортогональной функции проходит процедуру Map, которая для каждого k ставит в соответствие функцию, которая вычисляет значение для функции Якоби. Таким образом мы получаем кластер SIMD (Single Instruction Multiple Data) для обработки матрицы, элементами которых является порядки функции, а инструкция, выполняемая над ними, является функция Якоби.

**РЕЗУЛЬТАТЫ ВЫЧИСЛЕНИЙ**

Результаты вычислений представлены в виде таблиц 1-6 и в виде графиков 8-10. Значения функций представлены в графиках 11-13 Время вычислений указано в миллисекундах. Назовём функции 5.72 [1], 5.73 и 5.74 функциями Якоби №1, Якоби №2 и Якоби №3.

Таблица 1 - Последовательное вычисление функции Якоби №1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Порядок | 10 | 15 | 20 | 25 |
| Время, мс | 1 | 4 | 10 | 32 |

Таблица 2 - Последовательное вычисление функции Якоби №2

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Порядок | 10 | 15 | 30 | 60 |
| Время, мс | 1 | 2 | 25 | 214 |

Таблица 3 - Последовательное вычисление функции Якоби №3

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Порядок | 10 | 20 | 40 | 80 |
| Время, мс | 4 | 21 | 229 | 1466 |

Таблица 4 - Параллельное вычисление функции Якоби №1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Порядок | 10 | 15 | 20 | 25 |
| Время, мс | 57 | 50 | 69 | 42 |

Таблица 5 - Параллельное вычисление функции Якоби №2

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Порядок | 10 | 15 | 30 | 80 |
| Время, мс | 32 | 78 | 45 | 98 |

Таблица 6 - Последовательное вычисление функции Якоби №3

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Порядок | 10 | 20 | 40 | 80 |
| Время, мс | 38 | 47 | 117 | 613 |

Рисунок 8 – график зависимости времени выполнения от порядка функции Якоби №1

Рисунок 9 – график зависимости времени выполнения от порядка функции Якоби №2

Рисунок 10 – график зависимости времени выполнения от порядка функции Якоби №3

Рисунок 11 – график значений функции Якоби №1

Рисунок 12 – график значений функции Якоби №2

Рисунок 13 – график значений функции Якоби №3

РАСПРЕДЕЛЕНИЕ РОЛЕЙ В КОМАНДЕ

|  |  |
| --- | --- |
| Участник | Роль |
| Дмитрий Воробьев | серверная часть |
| Илья Кузьмин | клиентская часть |
| Андрей Сурков | реализация математических формул |
| Павел Чурсин | параллельный алгоритм |

СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ

1. Прохоров А.C, Basic orthogonal functions and its applications. Part I. Orthogonal function of exponential type[Текст]/ А.С. Прохоров, И.М Куликовских. -2013. -192 с.