

Министерство науки и высшего образования Российской Федерации  
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ  
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ  
НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО  
ITMO University

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА  
GRADUATION THESIS

Разработка сервиса интерполяции и экстраполяции данных

Обучающийся / Student Попов Александр Витальевич

Факультет/институт/кластер/ Faculty/Institute/Cluster факультет информационных технологий и программирования

Группа/Group М34351

Направление подготовки/ Subject area 01.03.02 Прикладная математика и информатика

Образовательная программа / Educational program Информатика и программирование  
2019

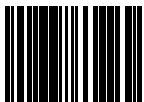
Язык реализации ОП / Language of the educational program Русский

Статус ОП / Status of educational program

Квалификация/ Degree level Бакалавр

Руководитель ВКР/ Thesis supervisor Лукин Михаил Андреевич, кандидат технических наук, Университет ИТМО, факультет информационных технологий и программирования, доцент (квалификационная категория "доцент практики")

Обучающийся/Student


Документ подписан	
Попов Александр Витальевич	
17.05.2023	

(эл. подпись/ signature)

Попов  
Александр  
Витальевич

(Фамилия И.О./ name  
and surname)

Руководитель ВКР/  
Thesis supervisor

Документ подписан	
Лукин Михаил Андреевич	
17.05.2023	

(эл. подпись/ signature)

Лукин Михаил  
Андреевич

(Фамилия И.О./ name  
and surname)

**Министерство науки и высшего образования Российской Федерации  
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ  
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ  
НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО  
ITMO University**

**ЗАДАНИЕ НА ВЫПУСКНУЮ КВАЛИФИКАЦИОННУЮ РАБОТУ /  
OBJECTIVES FOR A GRADUATION THESIS**

**Обучающийся / Student** Попов Александр Витальевич  
**Факультет/институт/кластер/ Faculty/Institute/Cluster** факультет информационных технологий и программирования  
**Группа/Group** М34351  
**Направление подготовки/ Subject area** 01.03.02 Прикладная математика и информатика  
**Образовательная программа / Educational program** Информатика и программирование 2019  
**Язык реализации ОП / Language of the educational program** Русский  
**Статус ОП / Status of educational program**  
**Квалификация/ Degree level** Бакалавр  
**Тема ВКР/ Thesis topic** Разработка сервиса интерполяции и экстраполяции данных  
**Руководитель ВКР/ Thesis supervisor** Лукин Михаил Андреевич, кандидат технических наук, Университет ИТМО, факультет информационных технологий и программирования, доцент (квалификационная категория "доцент практики")

**Основные вопросы, подлежащие разработке / Key issues to be analyzed**

Разработать библиотеку для языка Python, содержащую оптимальные для работы с большими наборами данных методы интерполяции и унифицирующую работу с ними. Она должна включать в себя такие методы, как Inverse Distance Weighting, Natural Neighbor, Radial Basis Function, Linear, Simple Kriging, Ordinary Kriging, Universal Kriging.

На основе созданной библиотеки разработать сервис для построения графиков пространственной интерполяции и экстраполяции данных. Он должен содержать весь функционал из предыдущей версии сервиса Uncorr, а также настраиваемые графики, сравнимые по качеству с создаваемыми программой Surfer от Golden Software. Кроме того, должна быть возможность интегрировать сервис в другие продукты компании Sudo.

**Форма представления материалов ВКР / Format(s) of thesis materials:**

Пояснительная записка, презентация

**Дата выдачи задания / Assignment issued on:** 01.02.2022

**Срок представления готовой ВКР / Deadline for final edition of the thesis** 15.05.2023

**Характеристика темы ВКР / Description of thesis subject (topic)**

**Название организации-партнера / Name of partner organization:** ООО Судо

**Тема в области фундаментальных исследований / Subject of fundamental research:** нет / not

**Тема в области прикладных исследований / Subject of applied research:** да / yes

**СОГЛАСОВАНО / AGREED:**

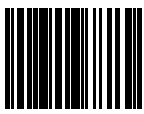
Руководитель ВКР/  
Thesis supervisor

Документ подписан	
Лукин Михаил Андреевич	
16.05.2023	

(эл. подпись)

Лукин Михаил  
Андреевич

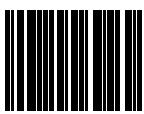
Задание принял к  
исполнению/ Objectives  
assumed BY

Документ подписан	
Попов Александр Витальевич	
16.05.2023	

(эл. подпись)

Попов  
Александр  
Витальевич

Руководитель ОП/ Head  
of educational program

Документ подписан	
Станкевич Андрей Сергеевич	
22.05.2023	

(эл. подпись)

Станкевич  
Андрей  
Сергеевич

**Министерство науки и высшего образования Российской Федерации  
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ  
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ  
НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО  
ITMO University**

**АННОТАЦИЯ  
ВЫПУСКНОЙ КВАЛИФИКАЦИОННОЙ РАБОТЫ  
SUMMARY OF A GRADUATION THESIS**

**Обучающийся / Student** Попов Александр Витальевич  
**Факультет/институт/кластер/ Faculty/Institute/Cluster** факультет информационных технологий и программирования  
**Группа/Group** М34351  
**Направление подготовки/ Subject area** 01.03.02 Прикладная математика и информатика  
**Образовательная программа / Educational program** Информатика и программирование 2019  
**Язык реализации ОП / Language of the educational program** Русский  
**Статус ОП / Status of educational program**  
**Квалификация/ Degree level** Бакалавр  
**Тема ВКР/ Thesis topic** Разработка сервиса интерполяции и экстраполяции данных  
**Руководитель ВКР/ Thesis supervisor** Лукин Михаил Андреевич, кандидат технических наук, Университет ИТМО, факультет информационных технологий и программирования, доцент (квалификационная категория "доцент практики")

**ХАРАКТЕРИСТИКА ВЫПУСКНОЙ КВАЛИФИКАЦИОННОЙ РАБОТЫ  
DESCRIPTION OF THE GRADUATION THESIS**

**Цель исследования / Research goal**

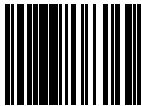
Разработать сервис интерполяции и экстраполяции данных

**Задачи, решаемые в ВКР / Research tasks**

Разработка архитектуры приложения; разработка и реализация библиотеки с методами интерполяции; разработка и реализация серверной части; разработка и реализация клиентской части; добавление возможности интеграции с другими сервисами; анализ решения и сравнение с аналогами.

**Краткая характеристика полученных результатов / Short summary of results/findings**

Была создана библиотека для языка Python, стандартизирующая работу с методами интерполяции и экстраполяции, реализованными в других библиотеках. Метод Inverse distance weighting был реализован самостоятельно. Библиотека была задокументирована, покрыта тестами и выложена в открытый доступ. На ее основе был разработан сервис, который позволяет обрабатывать большие наборы данных, создавать и редактировать графики интерполяции и экстраполяции. Качество графиков сравнимо с получаемыми в программах-аналогах и подходит для использования в научных работах. Добавлена возможность для последующей интеграции в другие продукты. Сервис внедрен в компанию Sudo и доступен по адресу [uncorr.com](http://uncorr.com).

подписан	
Попов Александр Витальевич	
17.05.2023	

(эл. подпись/ signature)

Попов  
Александр  
Витальевич

(Фамилия И.О./ name  
and surname)

Руководитель ВКР/  
Thesis supervisor

Документ подписан	
Лукин Михаил Андреевич	
17.05.2023	

(эл. подпись/ signature)

Лукин Михаил  
Андреевич

(Фамилия И.О./ name  
and surname)

## СОДЕРЖАНИЕ

СОДЕРЖАНИЕ.....	4
ВВЕДЕНИЕ .....	6
ГЛАВА 1. ОБЗОР И СРАВНЕНИЕ СУЩЕСТВУЮЩИХ ИНСТРУМЕНТОВ ПОСТРОЕНИЯ ИНТЕРПОЛЯЦИИ.....	7
1.1. Задача интерполяции и экстраполяции .....	7
1.2. Методы интерполяции .....	7
1.3. Обзор существующих инструментов .....	8
1.3.1. Surfer .....	8
1.3.2. Методы программирования.....	9
1.3.3. Microsoft Excel и Google Spreadsheets .....	10
1.3.4. Другие онлайн сервисы.....	10
1.3.5. Первая версия сервиса Uncorr .....	11
1.3.6. Сравнение существующих инструментов.....	13
1.4. Постановка ЦЕЛЕЙ И ЗАДАЧ РАБОТЫ .....	14
Выводы по главе 1 .....	15
ГЛАВА 2. ОПИСАНИЕ РАЗРАБОТКИ НОВОГО РЕШЕНИЯ .....	16
2.1. РАЗБИЕНИЕ НА ПОДЗАДАЧИ.....	16
2.2. АНАЛИЗ И РЕАЛИЗАЦИЯ ВЫДЕЛЕННЫХ ПОДЗАДАЧ .....	16
2.2.1. Разработка архитектуры приложения .....	16
2.2.2. Разработка библиотеки с методами интерполяции.....	17
2.2.3. Разработка серверной части .....	19
2.2.4. Разработка клиентской части .....	23
2.2.5. Интерфейс и типичный пользовательский сценарий .....	24
2.2.6. Интеграция с другими сервисами.....	29

Выводы по главе 2 .....	30
ГЛАВА 3. АНАЛИЗ РАЗРАБОТАННОГО СЕРВИСА .....	31
3.1. Анализ выполнения задач.....	31
3.2. Сравнение качества графиков с аналогами.....	32
3.3. График экстраполяции.....	35
3.4. Внедрение в компанию <i>SUDO</i> .....	36
3.5. Возможные направления развития сервиса в будущем .....	36
Выводы по главе 3 .....	36
ЗАКЛЮЧЕНИЕ.....	38
СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ .....	39

## ВВЕДЕНИЕ

Представленная работа опирается на результаты работы «Разработка сервиса для построения карт пространственной корреляции данных» [1]. Во время эксплуатации представленного в ней решения были найдены недостатки и выдвинуты новые требования от компании и пользователей, такие как: изменить набор технологий на стандартный для компании, добавить дополнительные методы интерполяции, добавить экстраполяцию данных, улучшить качество и добавить возможность интерактивного исследования и гибкой настройки полученных графиков, возможность обрабатывать большие наборы данных. Все эти требования невыполнимы в рамках существующего решения. Более подробно о предметной области, существующих аналогах, набору требований и обосновании причин создания нового решения рассказано в главе 1.

Цель работы – разработка нового сервиса для анализа пространственных данных и построения карт интерполяции и библиотеки методов интерполяции для него.

Во второй главе описывается процесс проектирования и реализации сервиса и библиотеки, решенные инженерные проблемы.

В третьей главе представлены результаты работы, производится сравнение с предшествующей версией сервиса и аналогами. Также указаны возможные направления развития в будущем.



# ГЛАВА 1. ОБЗОР И СРАВНЕНИЕ СУЩЕСТВУЮЩИХ ИНСТРУМЕНТОВ ПОСТРОЕНИЯ ИНТЕРПОЛЯЦИИ

## 1.1. ЗАДАЧА ИНТЕРПОЛЯЦИИ И ЭКСТРАПОЛЯЦИИ

Пространственная интерполяция – это использование известных значений величины в определенных точках для оценки неизвестных значений в неизвестных промежуточных точках [2].

В отличие от интерполяции, пространственная экстраполяция ставит перед собой задачу распространения вывода, полученного в результате анализа одной части процесса, на другую его часть или на процесс в целом [3].

Эти методы анализа популярны во многих статистических науках, связанных с землей, таких как геостатистика, нефтяная промышленность, метеорология, геохимия, география и других.

На практике вычисление значений в географических процессах производят методами интерполяции, где эти два подхода смешиваются вместе [4].

## 1.2. МЕТОДЫ ИНТЕРПОЛЯЦИИ

Существует множество методов построения интерполяции. В данной работе сравнивается работа следующих, реализованных в различных приложениях или библиотеках для языков программирования, методов:

- *Inverse Distance Weighting (IDW)* – метод обратных взвешенных расстояний;
- *Linear* – линейный метод;
- *Nearest Neighbor* – метод ближайшего соседа;
- *Radial Basis Function (RBF)* – метод радиальной базисной функции;
- *Simple Kriging* – простой кригинг;
- *Ordinary Kriging* – обычный кригинг;
- *Universal Kriging* – универсальный кригинг;

Они были выбраны как часто используемые специалистами, работающими в области геостатистики [5].

### 1.3. ОБЗОР СУЩЕСТВУЮЩИХ ИНСТРУМЕНТОВ

Географические карты с интерполированными значениями называются «картами интерполяции». Существуют различные инструменты для их построения, каждый из которых хорошо подходит для выполнения узкого круга задач. В этом разделе разберем, почему уже существующие аналоги не решают в полной мере задачу быстрого и доступного построения качественных и интерактивных карт интерполяции для больших наборов данных.

#### 1.3.1. Surfer

Стандартом индустрии считается программное обеспечение *Surfer* [6] от *Golden Software*. Приложение предоставляет широкие возможности по обработке и визуализации данных. В нем доступны практически все существующие на данный момент методы интерполяции, в том числе и обозначенные в данной работе. При помощи приложения можно не только тщательно настроить обработку данных и вычисления под свою задачу, но и построить карты интерполяции, исследовать их и адаптировать под использование в статьях (рисунок 1).

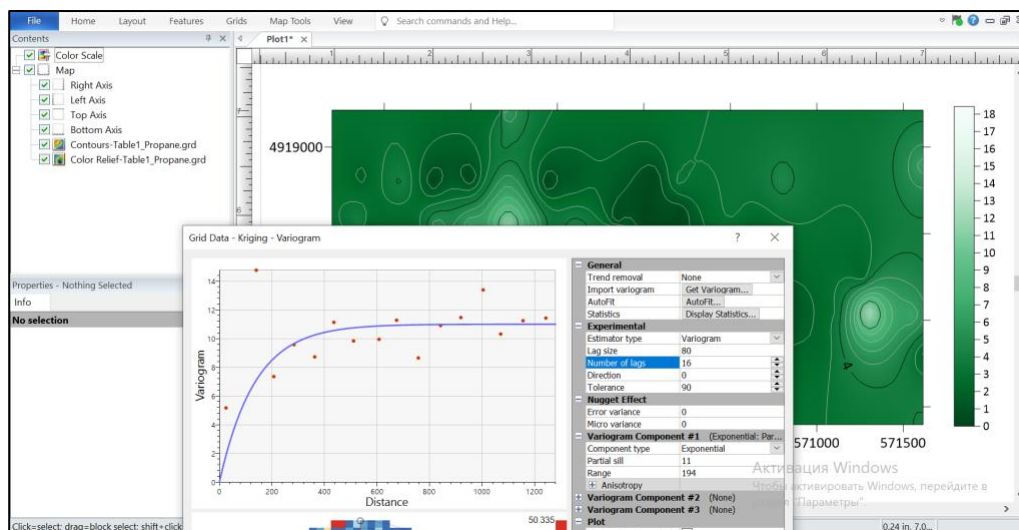


Рисунок 1 – Интерфейс *Surfer* версии 22.3.218

Но у этого решения есть и минусы. Использовать его можно только по платной лицензии, что делает его использование возможным только среди ограниченного числа работников из профильных организаций или состоявшихся ученых. Кроме них, существует большое количество людей, занимающихся исследованиями в области геостатистики, которые вынуждены пользоваться другими инструментами.

Также, обилие функционала и не самый современный дизайн *Surfer* делает его неудобным для использования новичкам. Необходимо потратить значительное количество времени на изучения программы, перед тем как начать ей пользоваться.

Инструмент имеет большое сообщество и богатую документацию, активно поддерживается и улучшается, последняя версия на момент написания этой работы была выпущена 04.04.2023.

### **1.3.2. Методы программирования**

Для получения наибольших возможностей к настройке и достижения наилучшего контроля за процессом обработки данных и построения карт интерполяции можно воспользоваться методами, в основе которых лежит программирование. Все методы интерполяции, построение интерактивных графиков уже реализовано в библиотеках. Они очень популярны, имеют большую поддержку сообщества, постоянно развиваются и исправляют найденные проблемы.

Самым популярным языком программирования для математических расчетов, согласно рейтингу TIOBE [7], является *Python*. Используя его библиотеки *Numpy* и *Pandas* можно обработать данные, при помощи *Scipy*, *GSTools* и *PyKrig* – воспользоваться любым из перечисленных выше методов интерполяции, а библиотека *Plotly* умеет создавать разнообразные графики.

Также существует и другой очень популярный подходящий здесь язык программирования *R*. Для него тоже было создано много библиотек для обработки данных, использования методов интерполяции и отображения данных на графиках.

Несмотря на обширные возможности и диапазон возможных решаемых задач, этот способ подходит не всегда. То, что все реализовано в библиотеках, еще не значит, что этим просто пользоваться. Большинство людей, работающих с географическими данными, не умеет работать с языками программирования. Поэтому данный инструмент для них недоступен.

### **1.3.3. Microsoft Excel и Google Spreadsheets**

Еще одним популярным методом для математических расчетов является использование интерактивных таблиц *Microsoft Excel* [10] или *Google Spreadsheets*. Они предоставляют обширные возможности по обработке табличных данных и построению графиков. Они существуют в виде отдельных приложений или интернет-сервисов.


Однако эти инструменты не предоставляют напрямую функционал интерполяции данных, его необходимо реализовать самостоятельно. Это очень серьезный недостаток, учитывая с требование довольно специфичных навыков и опыта работы с приложениями. Также, в них невозможно создание удобного шаблона, который бы подошел ко всем наборам данных. Соответственно, и автоматизация создания интерполяции также невозможна.

### **1.3.4. Другие онлайн сервисы**

Альтернативой предыдущим решениям являются всевозможные онлайн сервисы. Это удобный, бесплатный и быстрый способ выполнить несложные вычисления и построить карту интерполяции.

Низкая их популярность обусловлена отсутствием возможности предобработки данных, наличием только нескольких простейших методов, в основном только линейного или ближайшего соседа, и недостаточной возможностью настройки полученных графиков.

**Билинейная интерполяция** (Двойная линейная интерполяция)

[Онлайн сервисы](#) > Билинейная интерполяция (Двойная линейная интерполяция) 

$x_0$	<input type="text"/>	$y_0$	<input type="text"/>	$f(x_0, y_0)$	<input type="text"/>
$x_1$	<input type="text"/>	$y_1$	<input type="text"/>	$f(x_1, y_1)$	?
$x_2$	<input type="text"/>	$y_2$	<input type="text"/>	$f(x_2, y_2)$	<input type="text"/>
$x_0$	<input type="text"/>	$y_2$	<input type="text"/>	$f(x_0, y_2)$	<input type="text"/>
$x_2$	<input type="text"/>	$y_0$	<input type="text"/>	$f(x_2, y_0)$	<input type="text"/>
$x_1$	<input type="text"/>	$y_0$	<input type="text"/>	$f(x_1, y_0)$	?
$x_1$	<input type="text"/>	$y_2$	<input type="text"/>	$f(x_1, y_2)$	?
$x_0$	<input type="text"/>	$y_1$	<input type="text"/>	$f(x_0, y_1)$	?
$x_2$	<input type="text"/>	$y_1$	<input type="text"/>	$f(x_2, y_1)$	?

☐ Рассчитать

Рисунок 2 – Пример интерполяции в онлайн сервисе BL2.ru [8]

Примеры сервисов, которые при использовании вместе позволяют построить карту интерполяции:

- Contour Map Creator [9] – сервис для построения карт интерполяции.
- BL2.ru [8] - сервис для использования билинейной интерполяции.

### 1.3.5. Первая версия сервиса Uncorr

Как золотая середина между разнообразием функциональности, сложностью и удобством использования должен был стать сервис *Uncorr* от компании *Sudo* [1].

Он предоставляет возможности для базовой обработки данных, такой как нормализация и выбор осей координат. В нем поддержано несколько параметризуемых методов: *IDW*, *Ordinary kriging* и *Universal kriging*. После указания всех необходимых данных, просмотра основной статистики по набору данных, просмотра графика точек и гистограммы распределения можно выбрать метод интерполяции и задать основные параметры метода.

После этого строится карта интерполяции. Для нее можно задать бинарную цветовую палитру, количество и цвет изолиний.

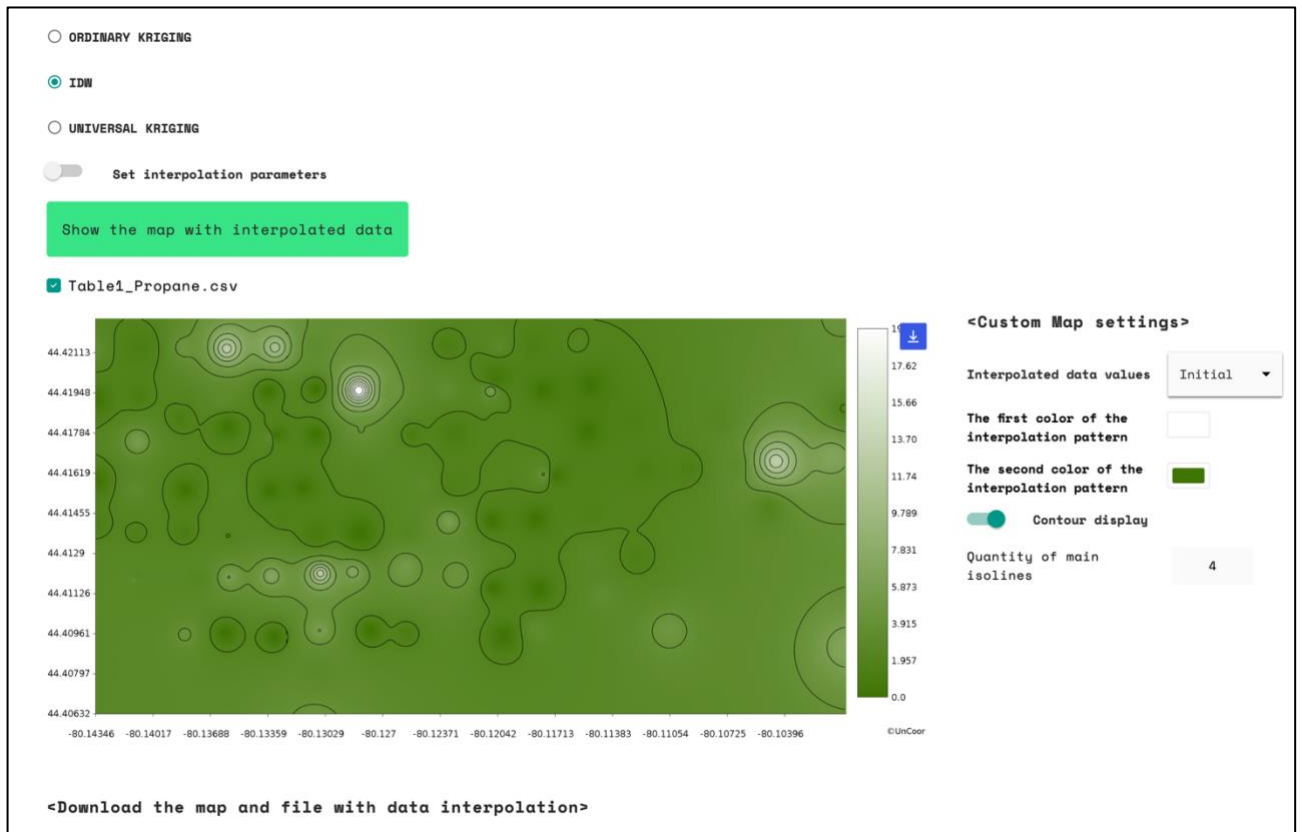


Рисунок 3 - Интерфейс сервиса *Uncorr* версии 1

Во время эксплуатации *Uncorr 1.0* были обнаружены следующие недостатки:

- Реализовано недостаточно методов интерполяции. Необходимо добавить методы *Simple kriging*, *Nearest neighbor*, *Linear* и *RBF*. Существующие методы были реализованы самостоятельно без использования вспомогательных библиотек и не всегда работают. Их сложно поддерживать из-за математической сложности методов.
- Были выявлены проблемы с алгоритмами отрисовки изолиний, возникают артефакты.
- Сервис не задумывался для работы с большими наборами данных. Методы поддерживают до тысячи точек.

- Внешний вид и формат графиков не подходит для того, чтобы вставлять их в статьи.
- Все графики клиенту сервиса показываются в формате картинки. Чтобы поменять параметры ее отображения, необходимо запросить с сервера новую картинку. Такое архитектурное решение создает дополнительную нагрузку на сервер, не позволяет пользователю исследовать график.
- Использованный набор технологий не является стандартным для компании *Sudo*. Возникли сложности с поддержкой сервиса, после того как его создатель закончил работу над ним. Компания выдвинула требования заменить язык программирования *Java* на *Python*.

### 1.3.6. Сравнение существующих инструментов

Ниже приведена сравнительная таблица четырех популярных решений.

Таблица 1 – Сравнение инструментов построения интерполяции

	<i>Surfer</i>	Методы программирования, <i>Microsoft Excel</i> / <i>Google Spreadsheets</i>	<i>Uncorr 1.0</i>	Другие онлайн сервисы
Количество методов	>10	>10	3	1
Возможности настройки	Да	Да	Да	Нет
Качественные графики	Да	Да	Нет	Нет
Поддерживает большие наборы данных	Да	Да	Нет	Нет

	<i>Surfer</i>	Методы программирования, <i>Microsoft Excel</i> / <i>Google Spreadsheets</i>	<i>Uncorr 1.0</i>	Другие онлайн сервисы
Платное	Да	Нет	Нет	Нет
Сложность использования	Средняя	Высокая	Низкая	Средняя

Из таблицы 1 видно, что создание более продвинутой версии *Uncorr* актуально и необходимо. Каждый из этих вариантов имеет свои минусы: большая цена за лицензионную копию, сложность в изучении и использовании, неудобство в работе, недостаточный или некачественно реализованный функционал. Все это не позволяет выделить инструмент без указанных выше недостатков, а потребность в нем у пользователей есть.

#### 1.4. ПОСТАНОВКА ЦЕЛЕЙ И ЗАДАЧ РАБОТЫ

Необходимо разработать веб-приложение, удовлетворяющее указанным требованиям:

- В нем реализован весь функционал из предыдущей версии сервиса по обработке наборов данных, просмотру статистики и графиков, построению и настройке карт интерполяции.
- Поддержаны методы интерполяции *IDW*, *Nearest neighbor*, *Linear*, *Radial basis function*, *Simple kriging*, *Ordinary kriging*, *Universal Kriging*.
- Должна быть возможность обрабатывать большие наборы данных (до миллиона точек) за умеренное время (до шестидесяти секунд на запрос).
- Качество графиков, интерактивность, возможности их обработки сравнимы с программой *Surfer* от *Golden Software*, а формат получаемых изображений должен быть *svg*.



- Поддержать возможность интеграции с другим сервисом компании *Sudo forctool.com*. Должна быть возможность обрабатывать в новом сервисе получаемые оттуда наборы данных.
- Приложение должно быть реализовано с использованием стандартных для компании *Sudo* технологий. Использовать для разработки серверной части язык программирования *Python*, а для клиентской части – *Typescript* и библиотеку *React*.

## ВЫВОДЫ ПО ГЛАВЕ 1

В данной главе проведен обзор предметной области, существующих методов интерполяции, инструментов построения карт интерполяции и проведено их сравнение. Были выделены проблемы в существующей реализации сервиса, доказывающие необходимость создания нового решения, исправляющего эти недостатки.

## ГЛАВА 2. ОПИСАНИЕ РАЗРАБОТКИ НОВОГО РЕШЕНИЯ

Чтобы достичь поставленной задачи необходимо разбить ее на независимые подзадачи и решить каждую из них.

### 2.1. РАЗБИЕНИЕ НА ПОДЗАДАЧИ

Были выделены следующие подзадачи:

1. Разработка архитектуры приложения.
2. Разработка и реализация библиотеки с методами интерполяции.
3. Разработка и реализация серверной части.
4. Разработка и реализация клиентской части.
5. Добавление возможности интеграции с другими сервисами.

### 2.2. АНАЛИЗ И РЕАЛИЗАЦИЯ ВЫДЕЛЕННЫХ ПОДЗАДАЧ

#### 2.2.1. Разработка архитектуры приложения

В сравнении с предыдущей версией сервиса (рисунок 4), архитектура приложения претерпела несколько не очень больших, но важных улучшений.

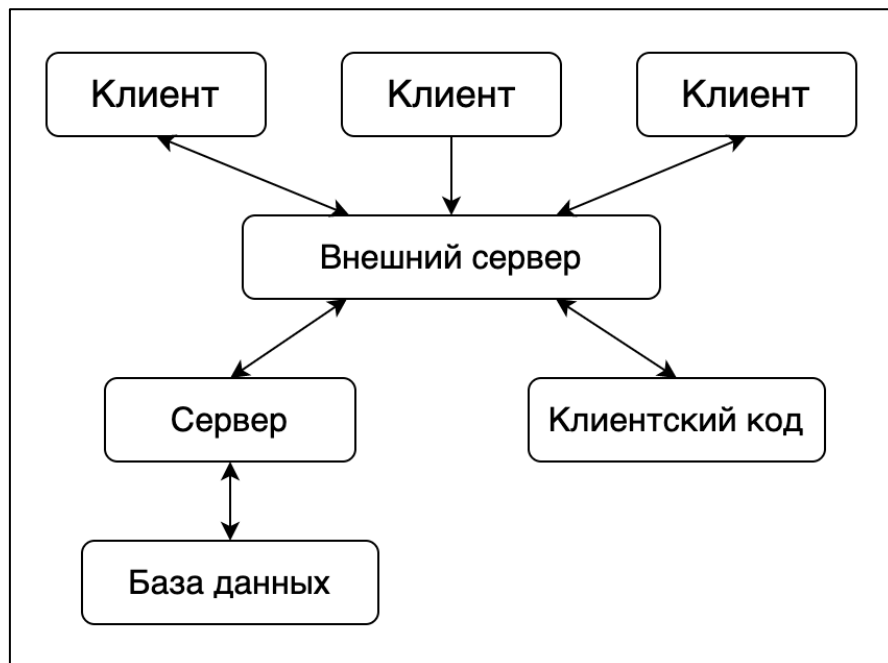


Рисунок 4 – Архитектура старой версии *Uncorr*

Модулей стало больше, и они стали более изолированы друг от друга. Это дает лучшую масштабируемость для задачи со сложными вычислениями

на сервере. Каждый запрос может обрабатываться до нескольких десятков секунд, занимая при этом поток обработки сервера. Можно сделать более многопоточный сервер, то есть заниматься вертикальным масштабированием системы, но более выгодно и перспективно именно горизонтальное масштабирование – увеличение количество независимых друг от друга серверов. Для задачи расчета интерполяции больших наборов данных это важное улучшение.

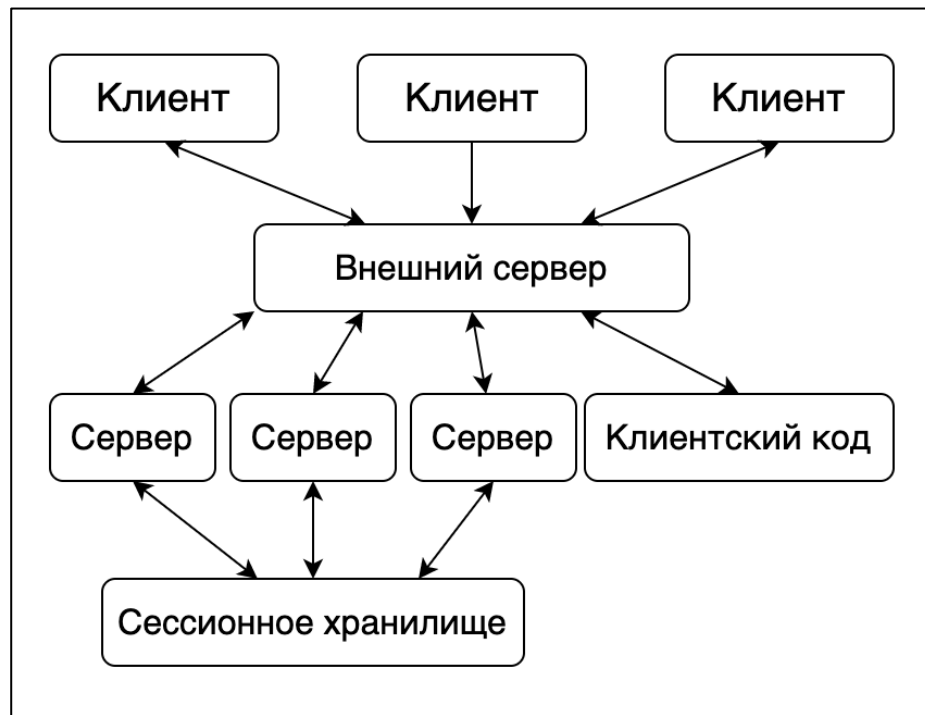


Рисунок 5 – Архитектура новой версии *Uncorr*

### 2.2.2. Разработка библиотеки с методами интерполяции

Как показала практика, реализация методов интерполяции самостоятельно невыгодна, потому что это дополнительный сложный алгоритмический код, который нуждается в тщательном тестировании и постоянной поддержке. Таких возможностей у проекта *Uncorr* нет, что привело к проблемам во время эксплуатации первой версии сервиса, где все алгоритмы были реализованы самостоятельно с использованием только стандартных библиотек.

Язык программирования *Python* популярен своими библиотеками для математических расчетов. Было принято решение найти как можно больше

реализаций методов интерполяции в популярных, хорошо покрытых тестами и активно поддерживаемых библиотеках. Также проверялась и оптимальность работы алгоритма для больших наборов данных.

Методы кригинга, функции для расчета эмпирических вариограмм и модели ковариации были взяты из пакета *GSTools*. Методы ближайшего соседа (*Nearest neighbor*), линейный метод (*Linear*) и метод радиальной базисной функции (*RBF*) были найдены в пакете *Scipy*. Удалось найти все методы, кроме метода обратных взвешенных расстояний (*IDW*) [11,12], который был реализован самостоятельно на основе библиотек *Scipy* и *Numpy*, протестирован и затем покрыт тестами.

Не было найдено библиотеки, которая бы предоставляла удобные методы для вычисления интерполяции и, вместе с тем, обладала достаточным разнообразием методов и параметров к ним. Было решено создать новую библиотеку интерполяции, в которой стандартизируется работа со всеми методами. Она была дополнительно покрыта тестами и документацией и выложена в общий индекс пакетов *PyPI* под названием *sdinterp* [13].

Структура библиотеки следующая: есть один общий класс с методом `__call__(points)`, который вызывается при передаче координат в уже созданную модель. В этом методе вызывается функция, которая должна быть задана в конструкторе каждого класса, наследующегося от базового. Для всех методов были созданы такие классы-наследники, в которых необходимо только реализовать конвертацию переданных параметров из стандартизированного вида в необходимый для текущего метода.

Это значит, что методы имеют две фазы: обучение и предсказание. На втором этапе можно получить аппроксимацию для произвольного интервала. Таким образом одновременно решаются задачи интерполяции и экстраполяции.

Ранее упоминалось, что метод *IDW* был реализован самостоятельно. Это было сделано потому, что все найденные реализации из библиотек плохо работают с большими наборами данных. Чтобы это исправить, было решено

использовать модифицированный *метод Шенарда* [14], ограничение в 20 обрабатываемых соседей для каждой точки. Для поиска ближайших соседей к текущей обрабатываемой точке использованы KD-деревья из пакета *Scipy*. В сумме это дало значительный прирост производительности метода.

В таблице 2 приведено время работы каждого метода из библиотеки для процессора *Intel Core i7-10850H 2.70GHz x12*.

Метод	Время работы, с
<i>IDW</i>	$10,0 \pm 0,5$
<i>Linear</i>	$8,3 \pm 0,1$
<i>Nearest Neighbour</i>	$3,0 \pm 0,1$
<i>RBF</i>	$26,0 \pm 1,0$
<i>Simple Kriging</i>	$33,2 \pm 2,0$
<i>Ordinary Kriging</i>	$37,0 \pm 2,0$
<i>Universal Kriging</i>	$39,5 \pm 2,0$

Таблица 2 - Время работы методов для набора данных размером 314000 точек

### 2.2.3. Разработка серверной части

Серверная часть предоставляет публичный интерфейс (*API*) для использования созданной библиотеки. Она была разработана с использованием *Python* библиотек *FastAPI*, *Numpy*, *Pandas* и еще нескольких вспомогательных пакетов. Использование только популярных поддерживаемых библиотек и языка *Python* позволило сделать более простую поддержку серверной части в будущем.

Все операции разбиты на три группы: операции с файлами, операции с вариограммами и вызовы методов интерполяции. Каждая из них имеет свою группу обработчиков. При разработке был применен архитектурный принцип *Clean* [15], который обеспечивает построение оптимальных систем объектов.

Архитектура представлена на рисунке 6.

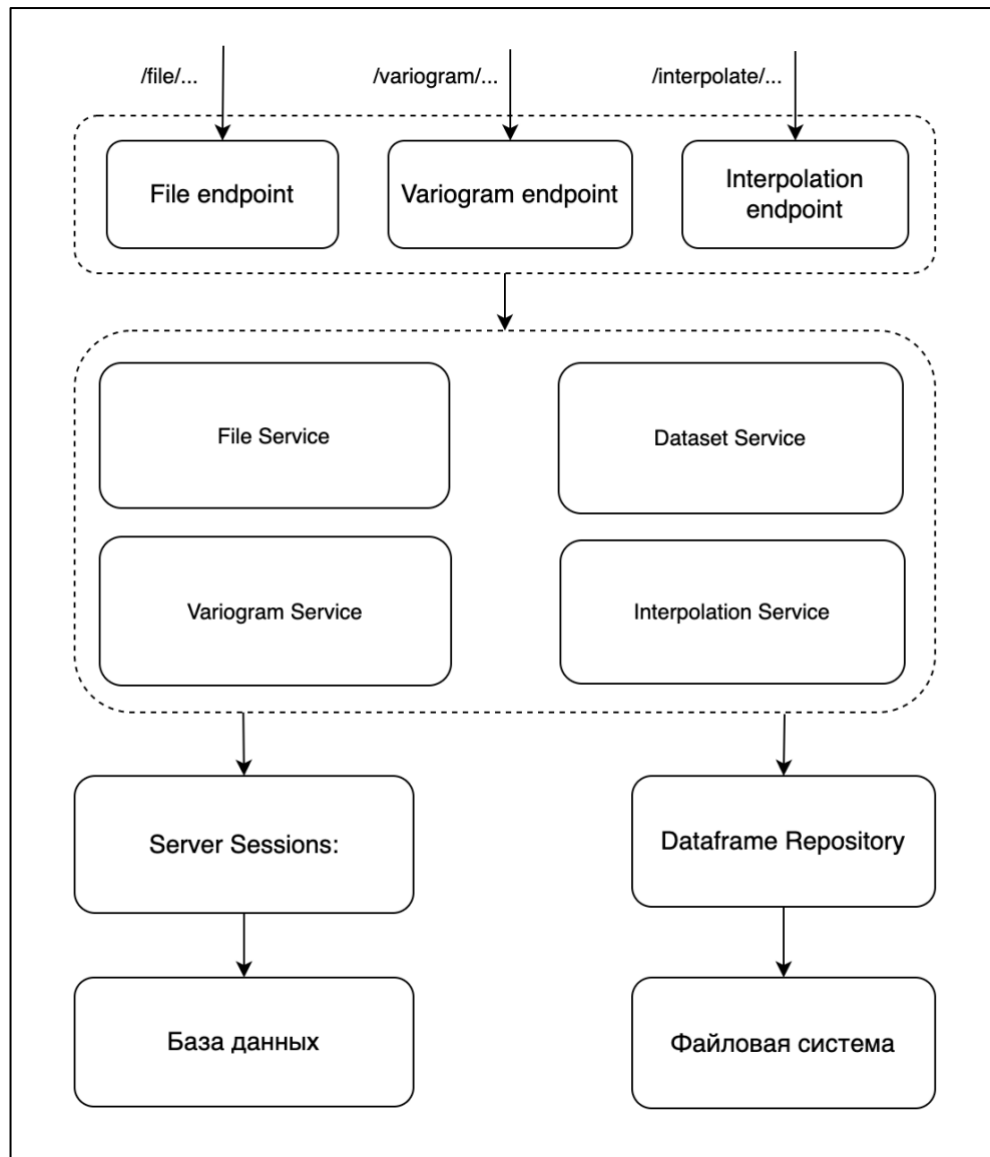


Рисунок 6 – Архитектура серверной части сервиса

Для работы с сервисом пользователю необходим хеш, уникально идентифицирующий набор данных, который стоит использовать для выполнения запроса. Этот хеш можно получить при загрузке файла с набором данных в формате csv. Тогда он сохраняется на сервере и в будущем при необходимости считывается. При загрузке файла два раза, второй раз он не будет сохранен.

Для больших наборов данных необходимо сохранять состояние сервиса, то есть информацию о результатах предыдущих запросов. Если этого не делать, придется на каждый запрос повторно проводить дорогостоящие

математические вычисления. Например, для автоматического расчета параметров выбранной модели ковариации необходимо знать результат расчета эмпирической вариограммы, а для методов кригинга необходима модель ковариации с определенными параметрами.

Хранить эти данные прямо на сервере непрактично, потому что требует наличие системы привязки запросов конкретного пользователя к одному и тому же серверу. Значит нужна быстрая и удобная в работе база данных. Такой является база данных *Redis*.

Но как долго надо хранить данные? Если запросы должны исполняться последовательно, а пользователь сделал большую паузу между запросами и данные на сервере удалились, то выполнить запрос больше не получится. Возникает необходимость передавать вместе с запросом все параметры для его расчета, когда данные предыдущих запросов уже были удалены. Полученная архитектура серверной части уже классифицируется как «без состояния», а все, что хранится в базе данных является кэшем. То есть система также реализует архитектурный принцип *REST* [16].

Будем хранить в кэше наборы данных после обработки (чтобы уменьшить количество считываний файла с диска и обработок данных), эмпирическую вариограмму и модель ковариации (потому что их расчет – дорогая операция). Хранить все данные очень невыгодно, даже если они стираются через какое-то время. Поэтому, после нескольких экспериментов было решено хранить только последние результат последнего расчета, но сделать кэш уникальным для пары пользователь + набор данных.

Идентификация пользователя происходит через технологии *cookie* и серверных сессий. При каждом ответе сервера отправляется уникальный номер в параметре *cookie*. Во время следующих запросов пользователя туда автоматически браузером добавляется этот номер и достается нужный кэш. Все это дает значительное ускорение работы сервиса на больших наборах данных. Стоит также упомянуть, что кэширование в данной архитектуре

необязательно и, если произошла поломка в базе данных или она отключилась, нужные значения просто заново будет рассчитаны.

Использование *FastAPI* также предоставляет автоматически сгенерированные по *OpenAPI* спецификации веб-интерфейсы *Swagger* и *ReDoc*, благодаря которым возможно удобное взаимодействие с приложением из браузера и размещение там документации к использованию *API* (рисунки 7,8).

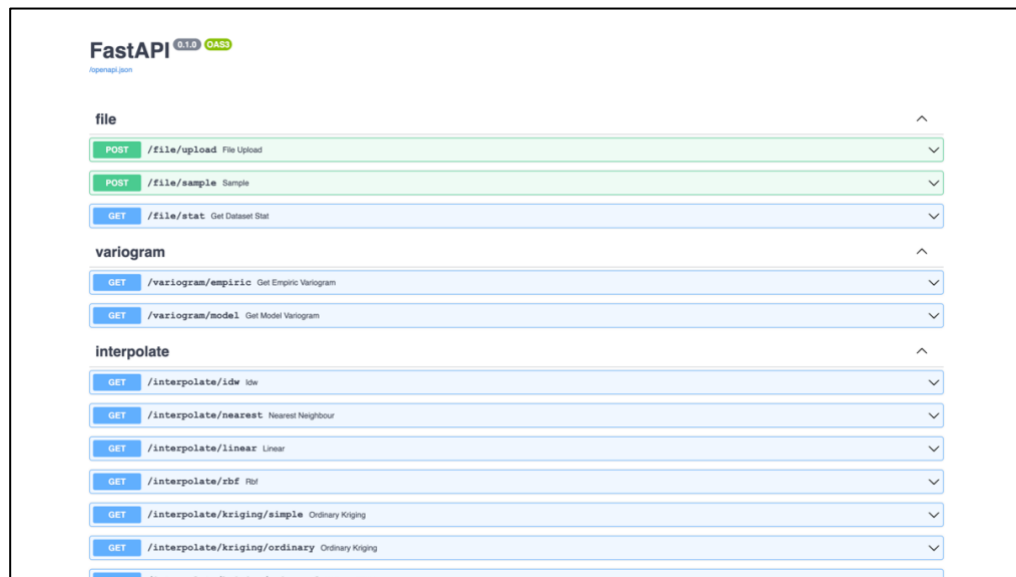


Рисунок 7 – Интерфейс *Swagger*

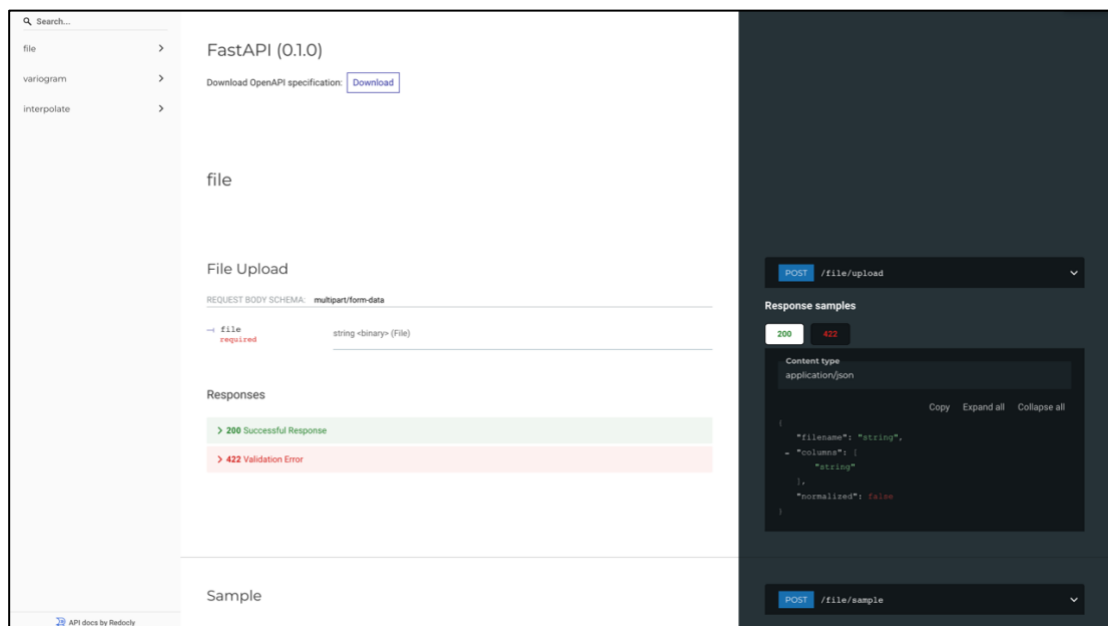


Рисунок 8 – Интерфейс *ReDoc*



## 2.2.4. Разработка клиентской части

Клиентская часть приложения была разработана с использованием популярных на данный момент технологий: язык программирования *Typescript*, библиотеки *React*, *reduxToolkit*, *plotly.js* и сборщик модулей *webpack*.

Создан интерфейс, состоящий из функциональных блоков. Каждый из них содержит специфичную часть функциональности и зачастую влияет на блоки, находящиеся ниже. На отдельной странице находится документация к сервису, страница с интерфейсом и документацией серверной части (*API*), описанная в предыдущем подразделе.

Всего доступно пять графиков:

- простой график точек – на нем можно посмотреть, где находятся точки, их значения и сделать первые выводы о наборе данных;
- график точек на карте, если выбраны географические координаты – то же самое, что и предыдущий график, но точки изображены на карте *OpenStreetMap*;
- гистограмма распределения с линией накопления – более детально показывает распределение данных;
- график эмпирической вариограммы и модели ковариации – нужен только для расчета параметров модели ковариации при использовании методов кригинга;
- карта интерполяции – результат работы метода интерполяции.

Графики строятся в браузере библиотекой *plotly.js*. Это сильно уменьшает нагрузку на сервера и дает возможность гибкой настройки получаемых графиков. Каждый из них может быть адаптирован и настроен пользователем под свои нужды. Для графика интерполяции дополнительно была добавлена форма настроек отображения. Выгрузить готовый график можно в подходящим под использование в научных статьях формате, в данном случае в *svg*.

Во всех полях ввода, где это возможно, были добавлены значения по умолчанию. Получается, чтобы получить результат нужно сделать всего несколько действий. Если подобранные значения не подходят, их можно задать вручную.

При разработке был использован подход *Feature Sliced Design (FSD)*. Он позволяет однозначно определить по назначению элемента интерфейса и использованным в нем элементам его место в проекте.

### 2.2.5. Интерфейс и типичный пользовательский сценарий

Разберем работу сервиса на типичном пользовательском сценарии. Каждый пользователь в начале работы видит только один блок с формой загрузки набора данных и кнопку «Choose sample dataset» (рисунок 9).

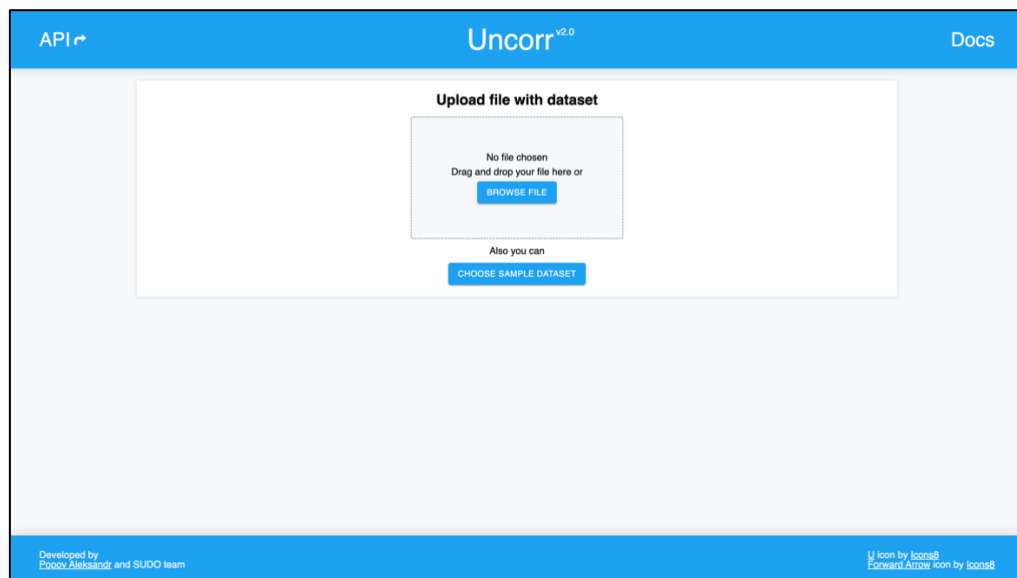


Рисунок 9 – Начальное состояние

После выбора набора данных и его загрузки на сервер, появляется возможность поменять местами координаты в данных или нормализовать целевое значение. Также показывается стандартная статистика по файлу: первые 10 строк, количество точек, минимальное и максимальное значение координат, а для целевого признака на сервере рассчитывается медиана и среднее значение. Ниже строятся графики точек и гистограммы распределения. Если указать систему географических координат (LatLon или

UTM), можно также увидеть *OpenStreetMap* [17] карту под точками (рисунки 10-13).

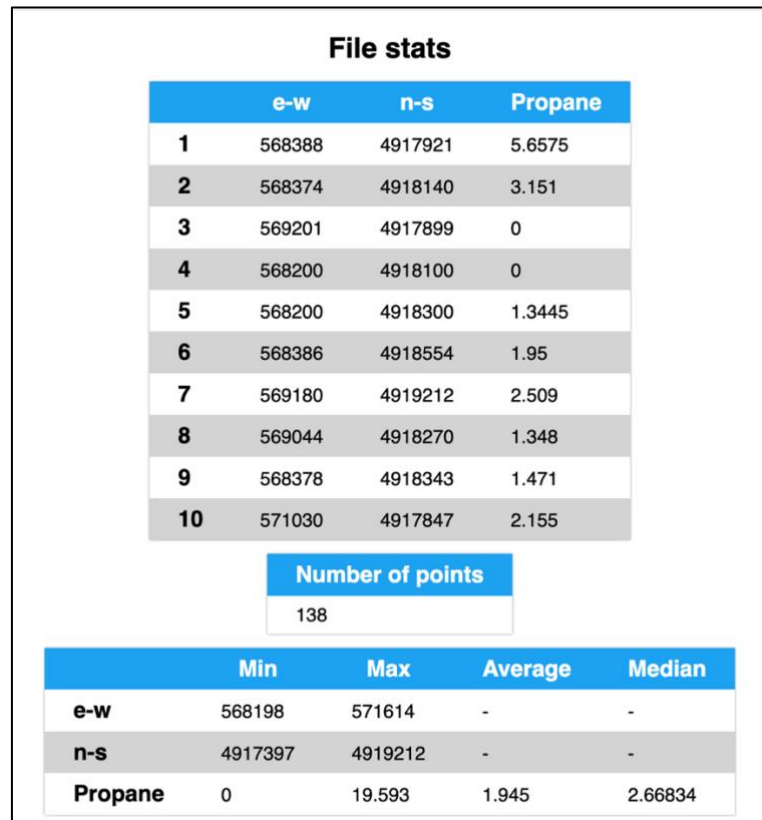


Рисунок 10 – Пример статистики по файлу

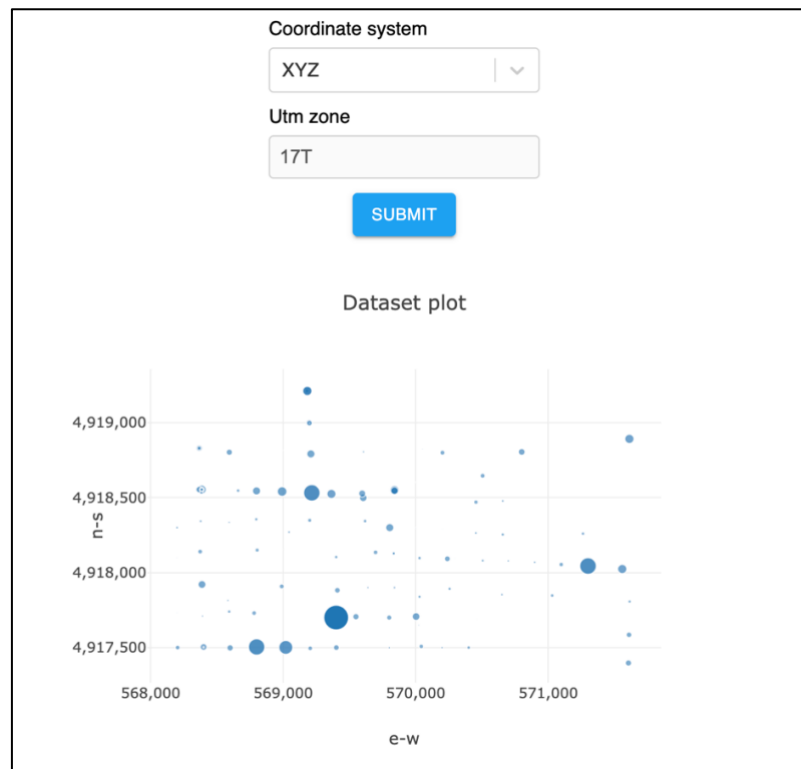


Рисунок 11 – Пример графика точек по набору данных

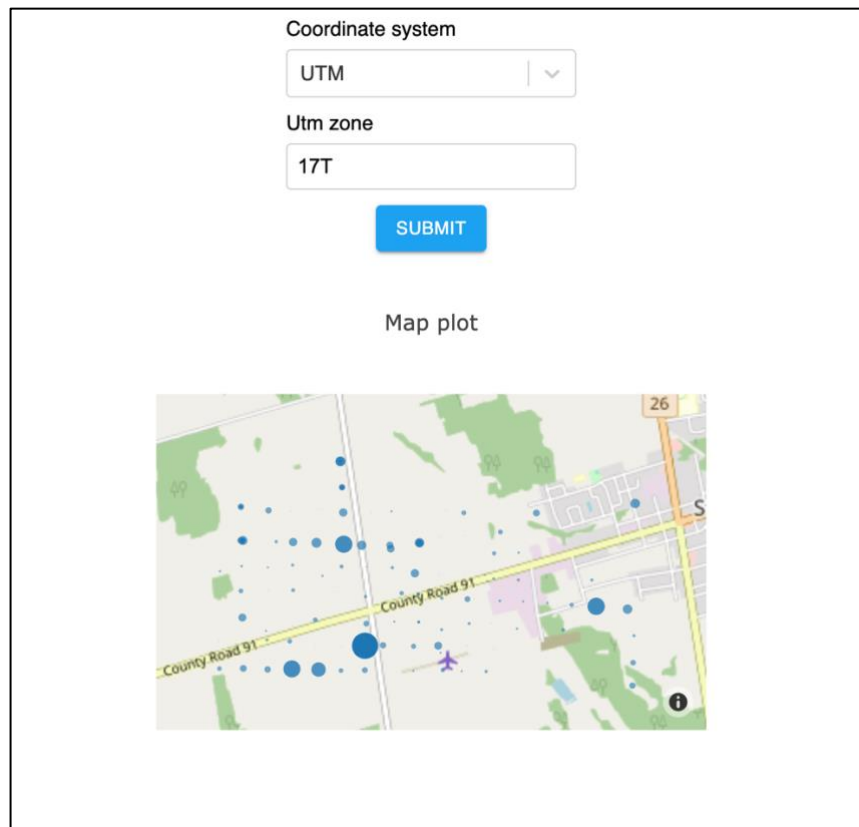


Рисунок 12 – Пример графика точек на карте

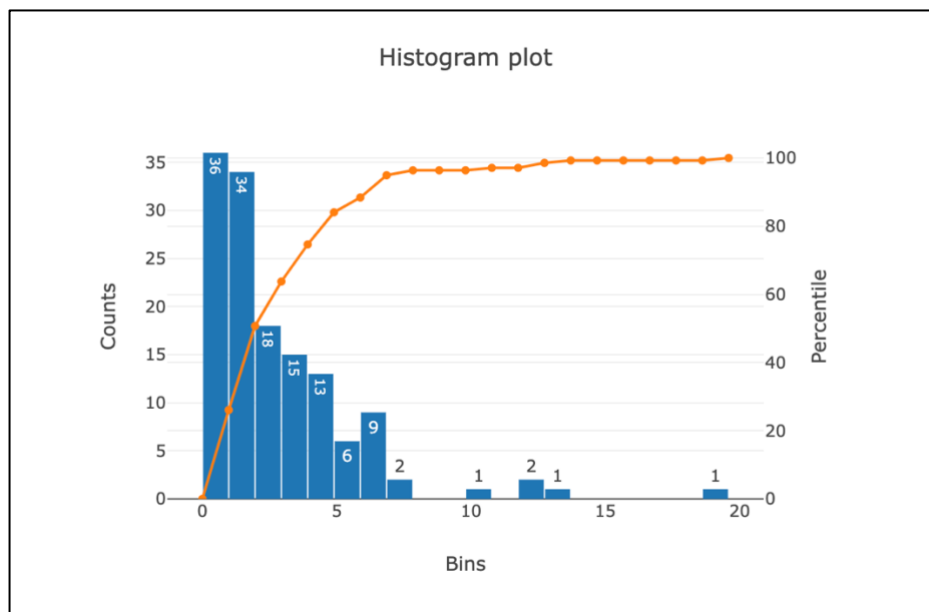


Рисунок 13 – Пример гистограммы распределения

Следующим этапом является выбор метода. При выборе кригинга необходимо также предоставить параметры модели ковариации. Для того, чтобы подобрать их, вначале рассчитывается эмпирическая вариограмма. По ее графику можно задать основные параметры (variance, range, nugget) самому

или не заполнять их и довериться автоподбору параметров. Все введенные пользователем параметры будут зафиксированы, а пустые поля будут подобраны (рисунки 14,15).

The screenshot shows a web-based interface with two main sections. The top section, titled "Choose interpolation method", contains a dropdown menu set to "Universal Kriging" and a blue "SUBMIT" button. The bottom section, titled "Configure kriging model", contains several input fields: "Model" (dropdown set to "Exponential"), "Variance" (text input with value "9.60015860585138"), "Range" (text input with value "85.32146805706518"), and "Nugget" (text input with value "Empty for autofit"). A blue "SUBMIT" button is located at the bottom of this section.

Рисунок 14 – Пример настройки метода и вариограммы

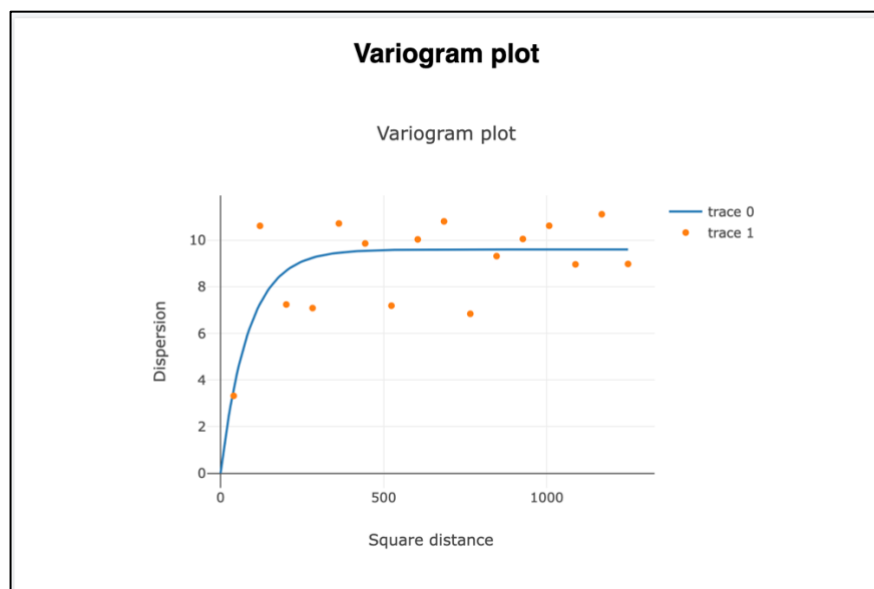
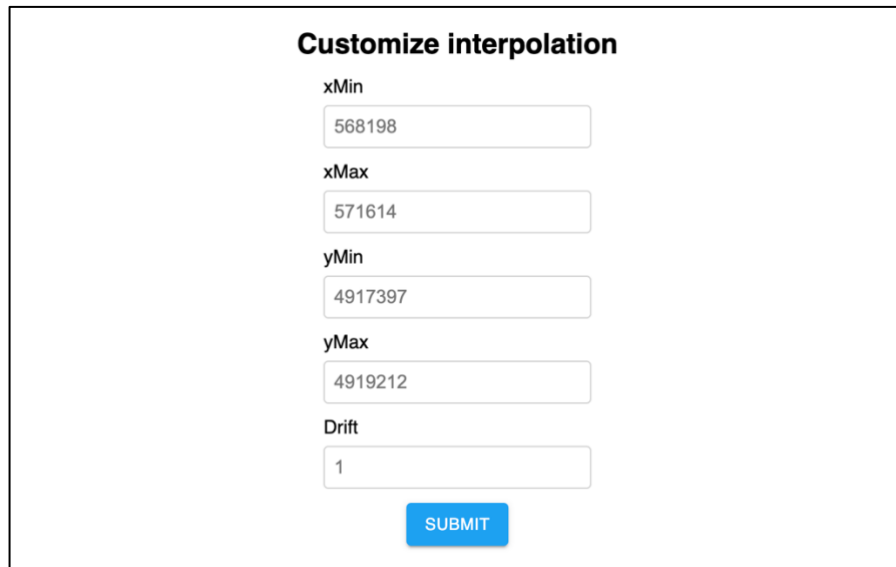


Рисунок 15 – Пример графика вариограммы

Далее для всех методов можно задать параметры методу и выбрать диапазон расчетов: минимум и максимум координат. По умолчанию в них берутся минимальное и максимальное значения из набора данных, но можно задать и свои значения. Это очень полезно, когда хочется получить лучшую

детализацию в небольшой ограниченной части пространства или посмотреть результат вне изучаемого процесса, то есть провести экстраполяцию данных (рисунки 16,17).



**Customize interpolation**

xMin  
568198

xMax  
571614

yMin  
4917397

yMax  
4919212

Drift  
1

**SUBMIT**

Рисунок 16 – Пример настройки интерполяции

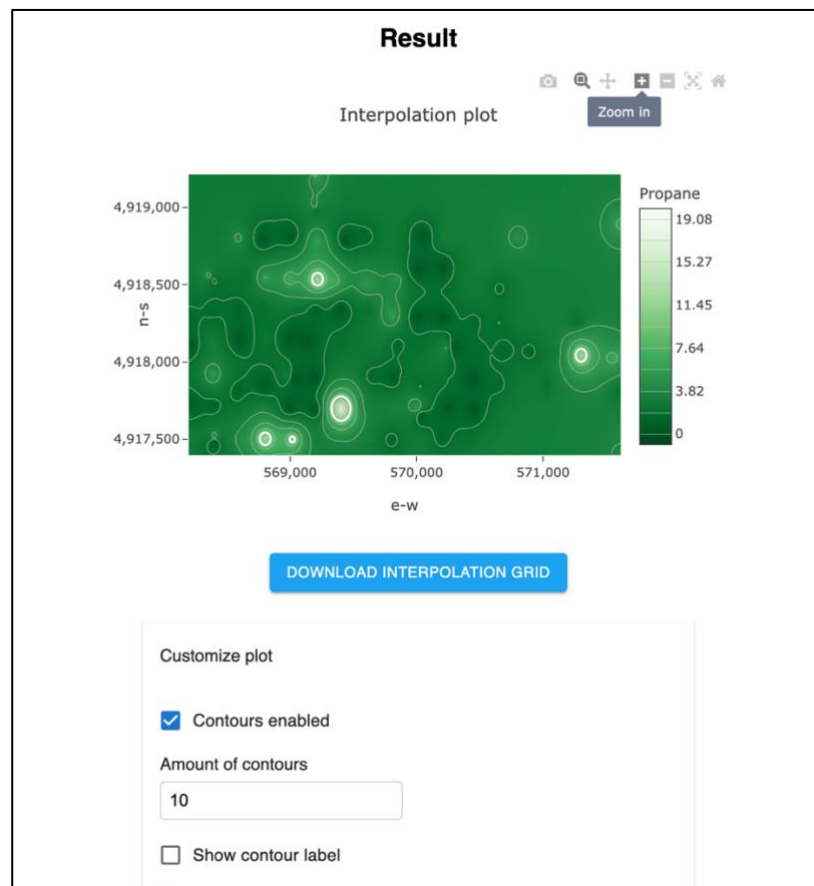


Рисунок 17 – Пример графика интерполяции и формы настройки отображения

### 2.2.6. Интеграция с другими сервисами

Несмотря на то, что получить приемлемый результат в сервисе просто и быстро, на правильную настройку параметров обычно уходит много времени. Необходимо добавить возможность поделиться финальным результатом. Кроме того, был обнаружен типичный пользовательский сценарий: после захода на страницу, пользователь загружает набор данных, двигается по всем полям ввода сверху вниз и, возможно, указывает значение параметра.

С целью удовлетворить пользовательские потребности была добавлена возможность интеграции через прямую ссылку. При указании в ссылке пути «/cmd» будет загружена обычная страница, но параметры форм будут извлечены из ссылки. Если в форму введено достаточно данных – она отправляется. Таким образом, для воспроизведения результата достаточно сохранить ссылку на него. Создать ее можно в любой момент, нажав на кнопку «Share result».

Также была добавлена другая возможность интеграции – вставка своего сервиса внутри другого (рисунок 18). По ссылке, заканчивающейся на «/embedded», открывается не привычная версия сайта, а только последний блок с интерполяцией. Параметры также собираются из ссылки. Если указано достаточно параметров, будет автоматически рассчитана интерполяция. Это очень удобно, так как пользователю не надо покидать веб-страницу, в которую интегрирован график. Но в этом подходе есть и минусы, такие как невозможность явно поменять параметры интерполяции.

## Какая-то другая страница

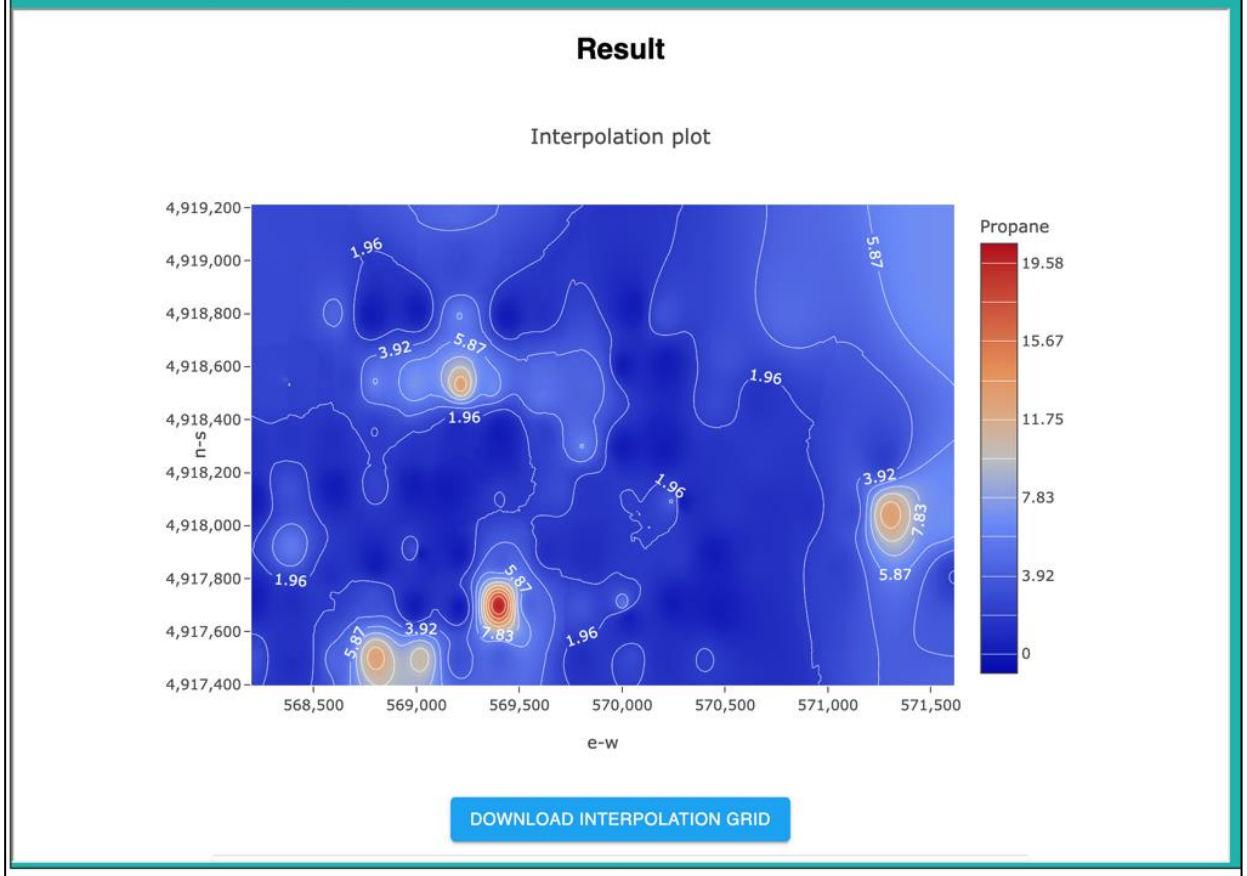


Рисунок 18 – Пример интеграции в другую веб-страницу

## ВЫВОДЫ ПО ГЛАВЕ 2

В данной главе было предложено разбиение задачи на меньшие подзадачи, приведено описание решения и процесса его реализации. Для каждой подзадачи указаны альтернативные варианты с аргументацией, почему выбранное решение является наилучшим. В результате были созданы библиотека с методами интерполяции для языка *Python* и веб сервис для построения карт интерполяции.



## ГЛАВА 3. АНАЛИЗ РАЗРАБОТАННОГО СЕРВИСА

Проанализируем созданный сервис на соответствие требованиям и конкурентоспособность по сравнению с аналогами и предшественником.

### 3.1. АНАЛИЗ ВЫПОЛНЕНИЯ ЗАДАЧ

Созданное решение позволяет анализировать большие наборы данных благодаря использованию более адаптированных под это реализаций алгоритмов. Также это сильно ускорило работу методов для небольших наборов данных. Созданная библиотека не имеет прямых аналогов, потому что обладает двумя свойствами одновременно: наличие большого разнообразия методов интерполяции данных и стандарта работы с ними. Кроме того, в ней реализован оптимальный для больших данных алгоритм *IDW*, не имеющий аналогов среди других библиотек. По сравнению с предыдущей версией сервиса, доступно большее количество методов.

Добавлена возможность интегрировать сервис в любые другие. Это открывает множество возможностей к синергии сервисов компании и построению экосистемы.

Сервис был реализован с использованием стандартных для компании технологий. Это удешевит и упростит поддержку сервиса и его дальнейшее развитие.

Реализован не только функционал из предыдущей версии сервиса, но и добавлен новый, такой как построение графика вариограммы. Перешедший по наследству функционал также был переработан и значительно улучшен. Подробное сравнение качества графиков приведено в следующем разделе.

Таблица 3 – Сравнительная таблица реализованного функционала

	<i>Surfer</i>	<i>Uncorr 1.0</i>	<i>Uncorr 2.0</i>
Обработка больших наборов данных	Да	Нет	Да
Интерактивные графики	Да	Нет	Да
График точек	Да	Да	Да

	<i>Surfer</i>	<i>Uncorr 1.0</i>	<i>Uncorr 2.0</i>
График точек на географической карте	Да	Да	Да
График гистограммы распределения	Да	Нет	Да
График эмпирической вариограммы и модели ковариации	Да	Нет	Да
График интерполяции	Да	Да	Да
Документация по использованию	Да	Нет	Да
Возможность интеграции в другие сервисы	Нет	Нет	Да
Возможность экстраполяции данных	Да	Нет	Да

### 3.2. СРАВНЕНИЕ КАЧЕСТВА ГРАФИКОВ С АНАЛОГАМИ

Очень важной задачей работы является построение графиков, которые не будут уступать по качеству другим сервисам. В данном разделе проведем сравнение, чтобы в этом удостовериться.

Возьмем один и тот же набор данных размером 138 точек и построим для него графики в *Surfer*, *Uncorr 1.0* и *Uncorr 2.0*. Использован метод *Ordinary kriging* с экспоненциальной моделью ковариации и параметрами  $nugget=0$ ,  $variance=11$ ,  $range=194$ . Изолинии на картинках отличаются, потому что сервисы используют разные способы задания высоты изолиний.

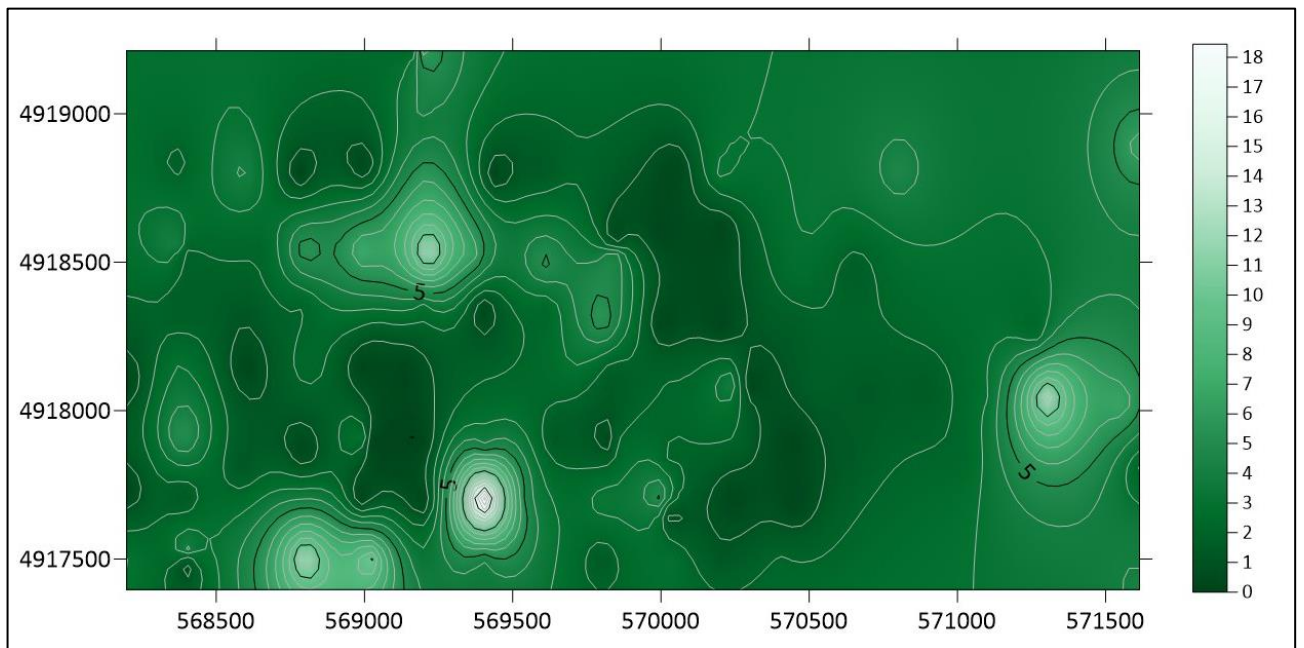


Рисунок 19 – Карта интерполяции первого набора данных в *Surfer*

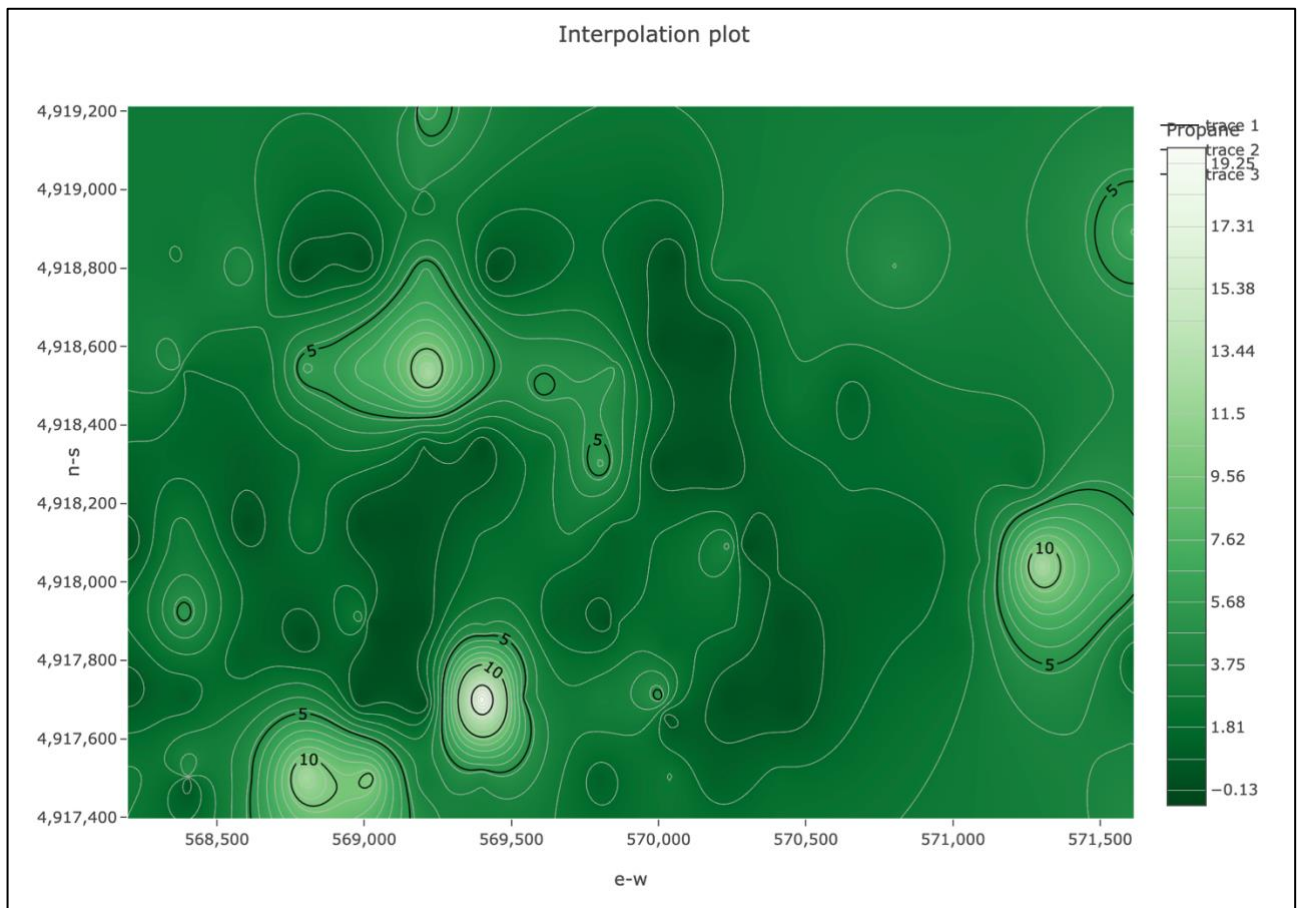


Рисунок 20 – Карта интерполяции первого набора данных в *Uncorr 2.0*

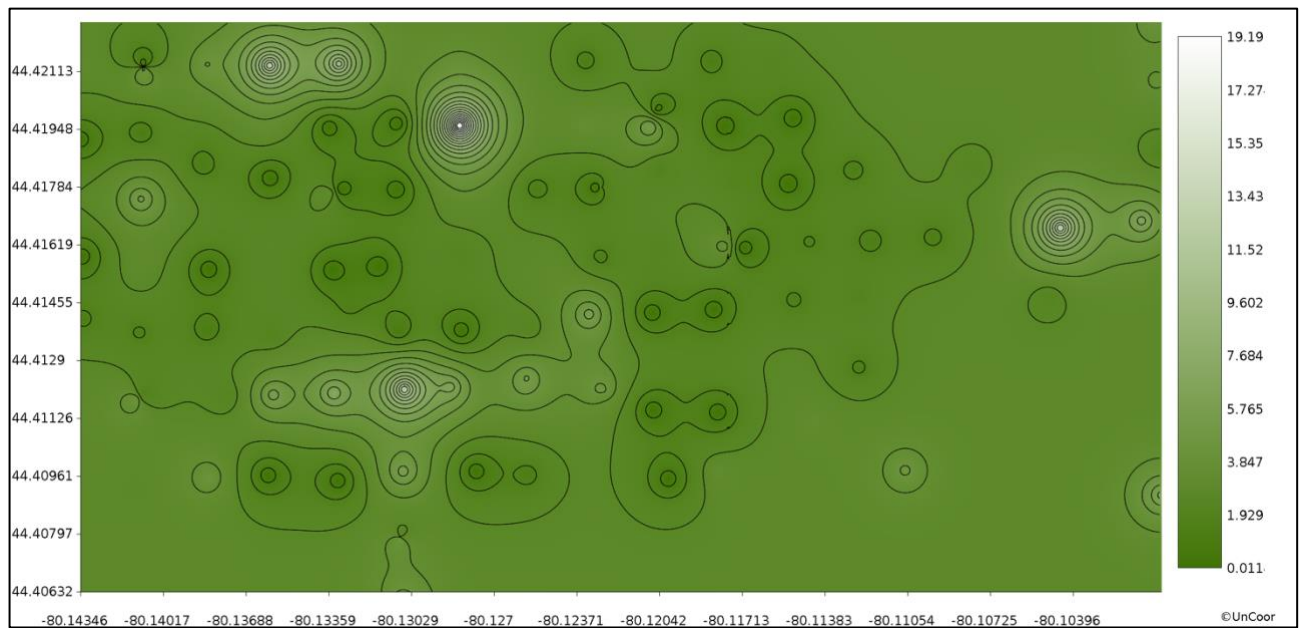


Рисунок 21 – Карта интерполяции первого набора данных в *Uncorr 1.0*

А теперь возьмем набор данных размером 313443 точки и посмотрим, как сервисы справятся с ним.

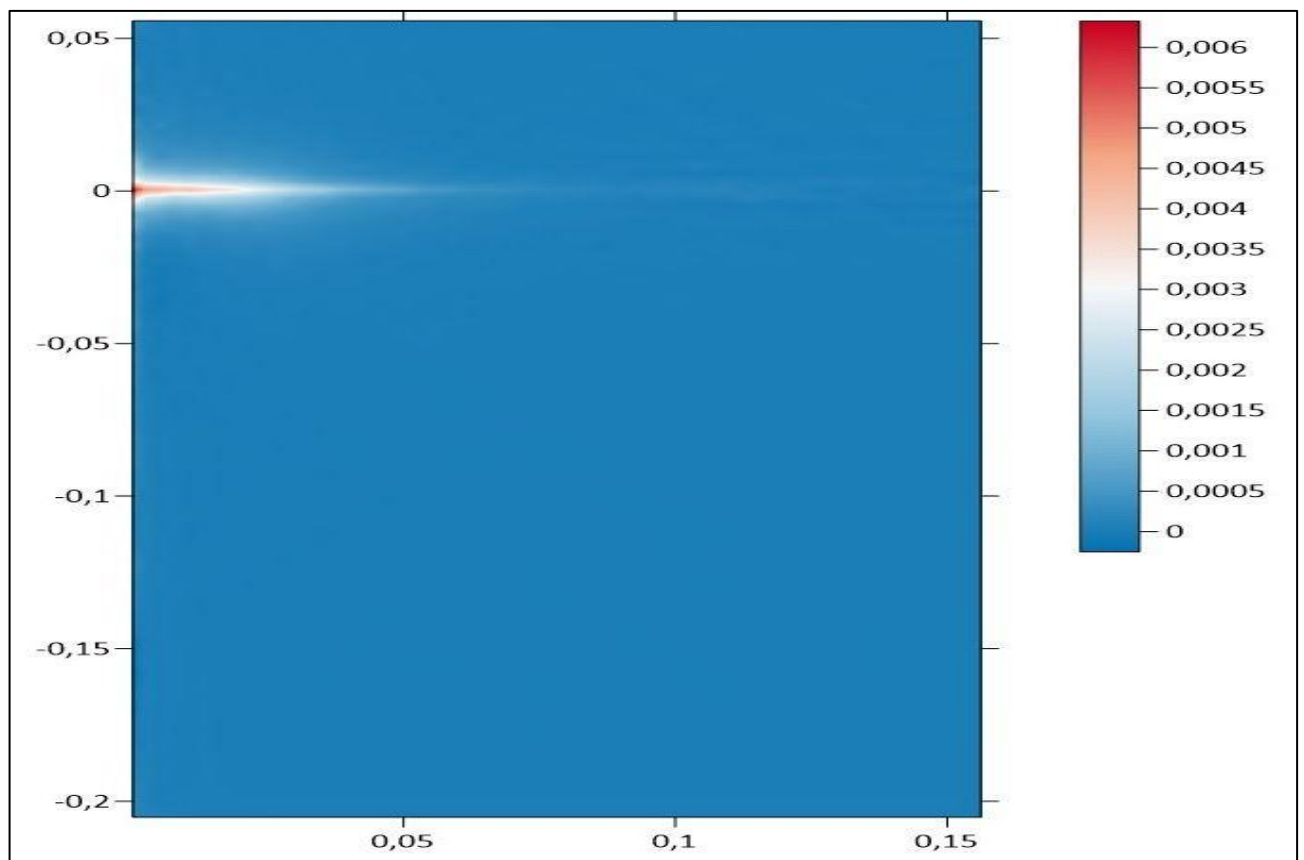


Рисунок 22 – Карта интерполяции второго набора данных в *Surfer*

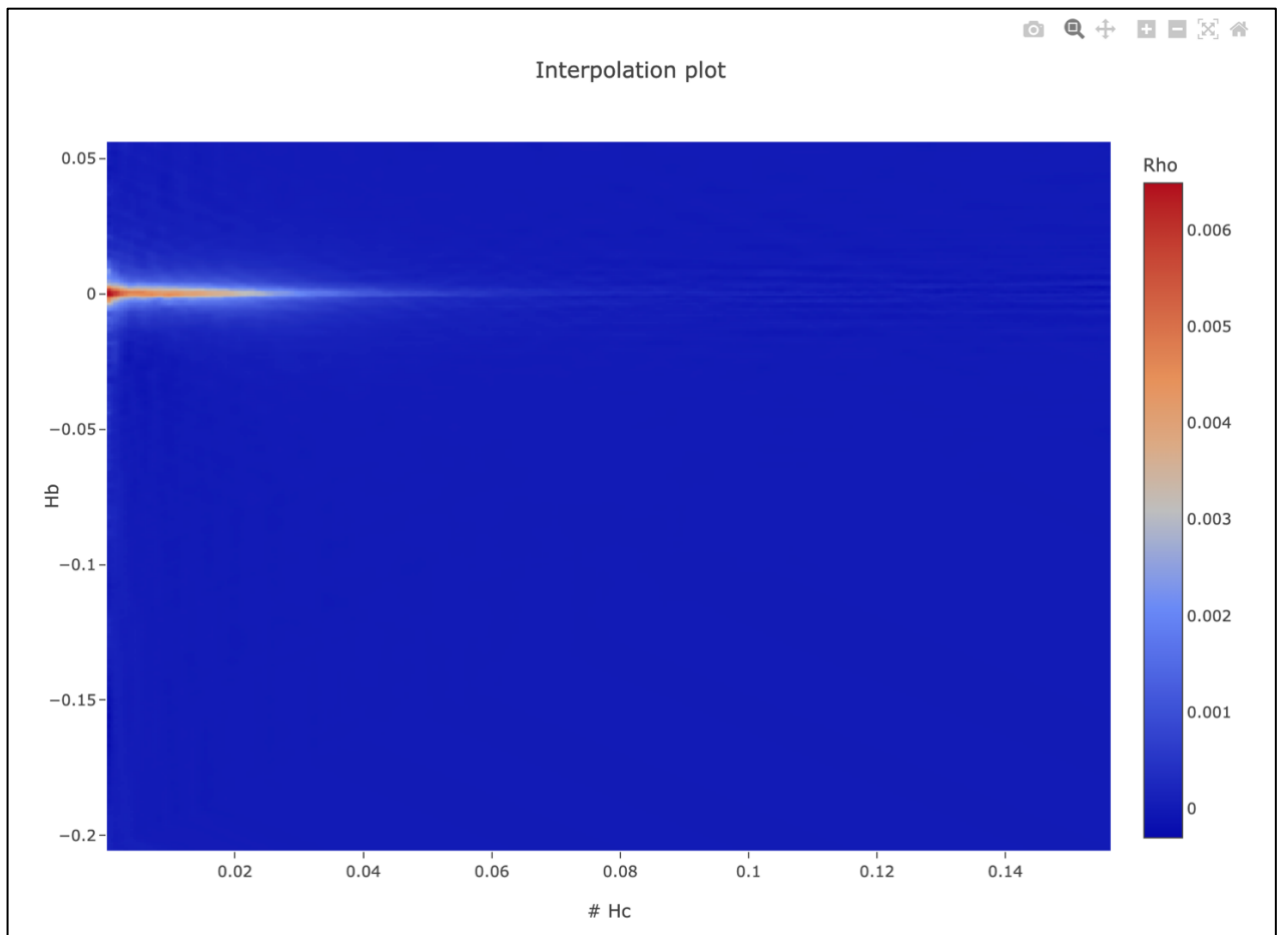


Рисунок 23 – Карта интерполяции второго набора данных в *Uncorr 2.0*

На графиках видно, что качество одинаково достойное и в *Surfer*, и в новом *Uncorr*. Старая версия сервиса не поддерживает такие большие наборы данных. Поэтому можно сделать вывод, что исходное требование соблюдено.

### 3.3. ГРАФИК ЭКСТРАПОЛЯЦИИ

Экстраполяция данных поддержана в разработанной библиотеке с методами интерполяции. Значит, построить график экстраполяции набора данных можно просто задав необходимый диапазон для вычисления пространства.

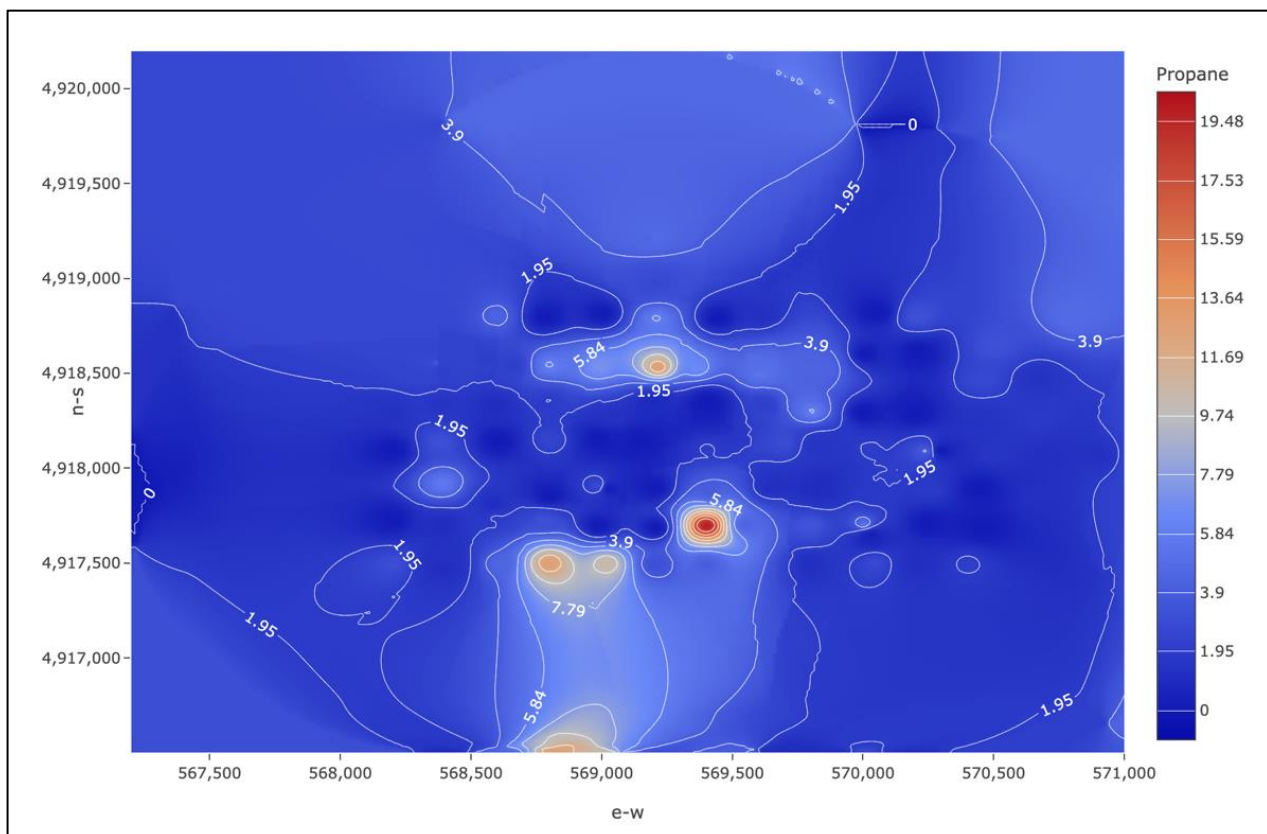


Рисунок 24 – Пример экстраполяции набора данных

### 3.4. ВНЕДРЕНИЕ В КОМПАНИЮ *SUDO*

Проект был внедрен в экосистему компании *Sudo* и заменил собой предыдущую версию сервиса *Uncorr*. На данный момент он доступен по ссылке <http://uncorr.com>.

### 3.5. ВОЗМОЖНЫЕ НАПРАВЛЕНИЯ РАЗВИТИЯ СЕРВИСА В БУДУЩЕМ

Обозначим также возможные направления развития сервиса в будущем:

- Добавление методов интерполяции через триангуляцию Делоне, и байесовский кригинг. Найти или реализовать оптимальный алгоритм и интегрировать в сервис.
- использовать алгоритмы машинного обучения для определения наилучшего метода и параметров интерполяции.

### ВЫВОДЫ ПО ГЛАВЕ 3

В данной главе был сделан анализ сервиса на соответствие поставленным в первой главе требованиям и проведено сравнение его с

другими сервисами, в том числе с его предыдущей версией. Указана информация о внедрении сервиса в компанию и обозначены возможные направления его развития в будущем.

## ЗАКЛЮЧЕНИЕ

В результате работы была создана библиотека для языка *Python*, стандартизирующая работу с методами интерполяции. Она была покрыта документацией, тестами и выложена в открытый доступ.

На ее основе был разработан сервис по интерполяции данных, который позволяет обрабатывать большие наборы данных, создавать и редактировать графики интерполяции. Качество графиков стало сравнимо с аналогами и подходит для использования в научных работах.

Первая версия сервиса уже имеет популярность в узких кругах исследователей и студентов, из чего можно сделать вывод о еще большей востребованности новой улучшенной версии.

Сервис внедрен в компанию *Sudo* и доступен по адресу <http://uncorr.com>.



## СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Кривопальцев Д.М. Разработка сервиса для построения карт пространственной корреляции данных: ВКР бакалавра. Прикладная математика и информатика: 01.03.02. - СПб., 2021. - 43 с.
2. Краткое введение в ГИС. Часть 10: Пространственный анализ растровых данных: интерполяция // GIS-lab URL: <https://gis-lab.info/qa/gentle-intro-gis-10.html> (дата обращения: 01.04.2023).
3. Экстраполяция // Wikipedia URL: <https://ru.wikipedia.org/wiki/Экстраполяция> (дата обращения: 01.04.2023).
4. M. A. Wahab, «Interpolation and extrapolation», *Proc. Topics Syst. Eng. Winter Term*. 2017. Vol. 17, P. 1–6.
5. Multivariate interpolation // Wikipedia URL: [https://en.wikipedia.org/wiki/Multivariate\\_interpolation](https://en.wikipedia.org/wiki/Multivariate_interpolation) (дата обращения: 01.04.2023).
6. Yang C. S. et al. Twelve different interpolation methods: A case study of Surfer 8.0 // *Proceedings of the XXth ISPRS congress*. 2004. Vol. 35. P. 778-785.
7. TIOBE Index for May 2023 // TIOBE URL: <https://www.tiobe.com/tiobe-index/> (дата обращения: 01.04.2023).
8. Билинейная интерполяция (Двойная линейная интерполяция) // BL2.ru URL: <https://www.bl2.ru/matematic/interpolation-double.html> (дата обращения: 01.04.2023).
9. Contour Map Creator // Contour Map Creator URL: <https://contourmapcreator.urgr8.ch> (дата обращения: 01.04.2023).
10. Salkind N.J., Frey B.B. *Statistics for people who (think they) hate statistics: Using Microsoft Excel*. - 5 edition. - University of Kansas, USA: SAGE Publications, 2021. - 512 pp.
11. Inverse Distance Weighted (IDW) Interpolation with Python // Stackoverflow URL: <https://stackoverflow.com/questions/3104781/inverse-distance-weighted-idw-interpolation-with-python> (дата обращения: 01.04.2023).

12. Inverse distance weighting // Wikipedia URL: [https://en.wikipedia.org/wiki/Inverse\\_distance\\_weighting](https://en.wikipedia.org/wiki/Inverse_distance_weighting) (дата обращения: 01.04.2023).
13. sdinterp 0.0.1 // PyPI URL: <https://pypi.org/project/sdinterp/> (дата обращения: 29.04.2023).
14. Shepard D. A two-dimensional interpolation function for irregularly-spaced data // Proceedings of the 1968 23rd ACM national conference. 1968. P. 517–524.
15. Robert Martin Clean Architecture: A Craftsman's Guide to Software Structure and Design. - 1 edition. - Pearson, 2017. - 431 pp.
16. Feng X., Shen J., Fan Y. REST: An alternative to RPC for Web services architecture // 2009 First International Conference on Future Information Networks. 2009. P. 7–10.
17. Haklay M., Weber P. Openstreetmap: User-generated street maps // IEEE Pervasive Comput. 2008. Vol. 7, № 4. P. 12–18.