Дипломная работа по теме:

**Сравнение различных библиотек для визуализации данных**

Автор: Шарый Владимир Петрович

Urban University

2024г

Содержание

[**1. Введение** ……………………………………………………………………………………………………………………………………………3](#_Toc186137050)

[**Обоснование выбора темы** 3](#_Toc186137051)

[**Определение цели и задач исследования** 4](#_Toc186137052)

[**2. Краткое сравнение библиотек** …………………………………………………………………………………………………………5](#_Toc186137053)

[**3. Подход к разработке** ………………………………………………………………………………………………………………………..6](#_Toc186137054)

[Matplotlib 7](#_Toc186137055)

[Seaborn 10](#_Toc186137056)

[Plotly 13](#_Toc186137057)

[**4. Анализ и интерпретация результатов** ……………………………………………………………………………………………17](#_Toc186137058)

[Сравнение библиотек 17](#_Toc186137059)

[Интерпретация результатов и рекомендации к использованию 20](#_Toc186137060)

[**5. Структура программы** …………………………………………………………………………………………………………………….22](#_Toc186137061)

[**6. Заключение** ……………………………………………………………………………………………………………………………………..23](#_Toc186137062)

[Обзор выполненной работы 23](#_Toc186137063)

[Дальнейшие планы 23](#_Toc186137064)

[**7. Список литературы** ………………………………………………………………………………………………………………………….24](#_Toc186137065)

# **1. Введение**

## **Обоснование выбора темы**

1. **Актуальность визуализации данных**

Современный мир переполнен данными, которые важно не только собирать, но и эффективно интерпретировать. Визуализация играет ключевую роль в анализе данных, помогая исследователям, аналитикам и разработчикам быстро находить закономерности, выявлять аномалии и доносить информацию до аудитории. Тема сравнения библиотек актуальна, так как выбор подходящего инструмента для визуализации может значительно упростить процесс создания и анализа полученных данных.

1. **Широкий выбор инструментов**

В Python существует множество библиотек для визуализации, таких как Matplotlib, Seaborn, Plotly и другие. Каждая из них обладает уникальными возможностями, преимуществами и ограничениями, что создает необходимость их сравнения для определения лучшего решения под конкретные задачи.

1. **Универсальность применения**

Визуализация данных используется в различных областях: науке, бизнесе, маркетинге, образовании и других. Выбор подходящего инструмента может существенно повлиять на качество анализа и принятие решений. Поэтому важно понять, в каких случаях предпочтительнее использовать ту или иную библиотеку.

1. **Образовательная ценность**

Сравнение библиотек — полезная тема для студентов, аналитиков и разработчиков, которые только начинают работать с визуализацией данных. Это позволяет быстро ознакомиться с ключевыми инструментами и выбрать наиболее подходящий метод для визуализации текущих и будущих проектов.

1. **Развитие навыков**

Тема способствует не только изучению основ визуализации данных, но и позволяет освоить особенности работы с различными типами графиков в библиотеках, настройки параметров, а также познакомиться с разными подходами в библиотеках к построению графиков, получить сравнение затраченных памяти и времени на отрисовку графиков (если много графиков чем это может грозить серверу)

1. **Помощь в оптимизации процессов**

Правильный выбор библиотеки может ускорить процессы анализа данных, сделать графики более читаемыми и улучшить взаимодействие с командой или клиентами. Поэтому важно понять, какой инструмент лучше всего справляется с конкретной задачей и возможностями сервера.

## **Определение цели и задач исследования**

Цель исследования: В данной работе будут рассмотрены три популярные библиотеки для визуализации данных в Python: **Matplotlib**, **Seaborn** и **Plotly**. Сравним их по ключевым параметрам, включая простоту использования, возможности настройки, типы поддерживаемых графиков и интерактивность, чтобы помочь определить, какой из них лучше подходит для конкретных задач и аппаратных возможностей.

Задачи исследования:

1. Краткий обзор библиотек **Matplotlib**, **Seaborn** и **Plotly.** Описание их преимуществ и недостатков.
2. Создание кода с использованием этих библиотек с различными настройками, данными и типами графиков.
3. Реализация возможности последовательного запуска примеров с этими библиотеками.
4. Сбор данных, затраченных памяти и времени, вывод результатов в итоговом графике.
5. Написание дипломной работы: составить дипломную работу, включающую в себя введение, обзор материалов, методологию и результаты исследования, анализ результатов, выводы и рекомендации.

Цели и задачи исследования направлены на получение практических результатов, которые позволят показать преимущества и более подходящие места использования обозначенных выше библиотек.

# **2. Краткое сравнение библиотек**

1. **Matplotlib**

* Это базовая библиотека для визуализации данных в Python, которая предоставляет гибкость и полный контроль над графиками. Она служит основой для многих других библиотек, включая Seaborn.

1. **Seaborn**

* Библиотека, построенная на основе Matplotlib, предназначенная для более удобного и эстетичного создания визуализаций. Она отлично подходит для работы с данными в формате DataFrame (например, из pandas).

1. **Plotly**

* Библиотека для интерактивной визуализации данных, которая поддерживает как веб-приложения, так и локальные графики. Отлично подходит для сложных и динамических визуализаций.

# **3. Подход к разработке**

Для сравнения способов визуализации, создадим простое приложение. Работа приложения выполняется с одного запуска с выводом всех графиков, реализованных при помощи сравниваемых библиотек. При завершении в консоль выводится надпись: «Работа программы сравнения Matplotlib, Seaborn, Plotly завершена!!!» а браузере итоговый график с использования памяти и затраченного времени для каждой библиотеки при выполнении каждого графика.

Для сбора дынных создадим файл для работы с базой данных и саму базу данных.

Для получения данных о работе создадим файл с функцией «декоратор», которая будет получать данные при построении графиков и заносить их в базу данных.

Для демонстрации визуализации будем использовать файл с функцией создания необходимых данных вычисленных на базе библиотек numpy и pandas.

Создадим три отдельных файла, каждый из которых будет содержать функции отображения графиков, одинаковые для каждой из библиотек визуализации.

Список графиков:

Загрузка\*

Разброс

График

Гистограмма

Ящики

Круговая диаграмма\*\*

График функции

График двух функций

Двумерная гистограмма

Разброс 3D

Каркасная поверхность\*\*

Просто красота\*\*

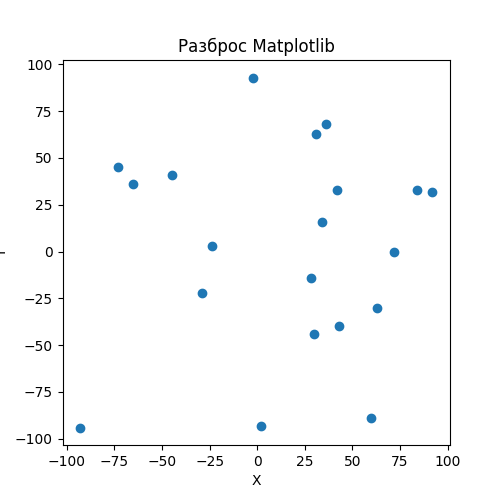
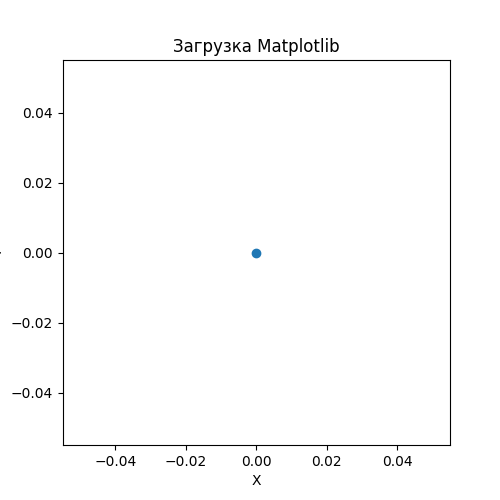
Для демонстрации итогов создадим файл, содержащий функцию извлечения полученных данных из базы данных, разбивку по каждой библиотеке и создании итогового графика.

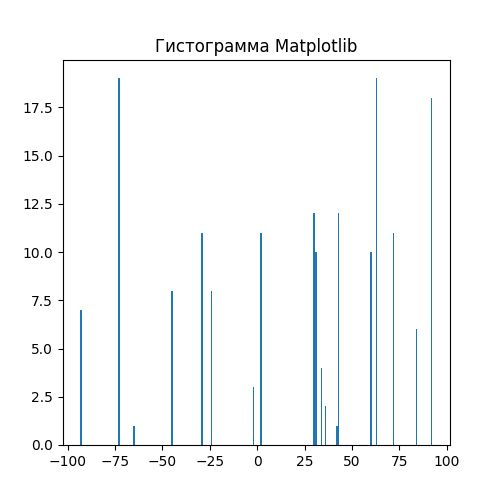
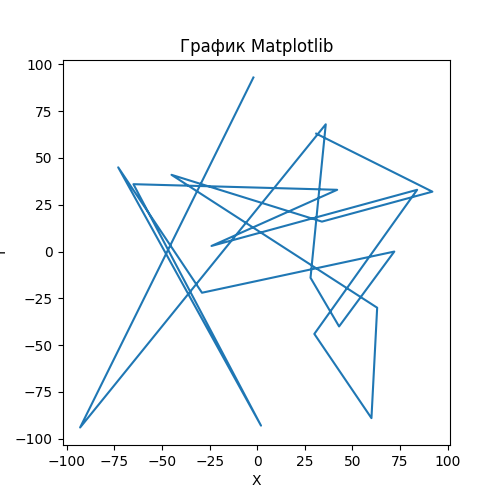
\*При запуске происходит загрузка библиотеки в память, что намного больше чем при последующей работе.

\*\*Ввиду того что данных графиков нет в библиотеке Seaborn, для чистоты эксперимента, эти функции в тестовом файле возьмем из родительской библиотеки Matplotlib.

## Matplotlib

Создаём файл связанный с Matplotlib. Передаем в него данные для построения. Получаем графики представленные на рис.1 - 3





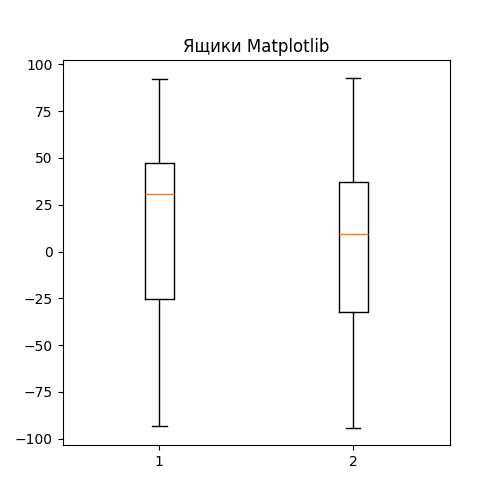
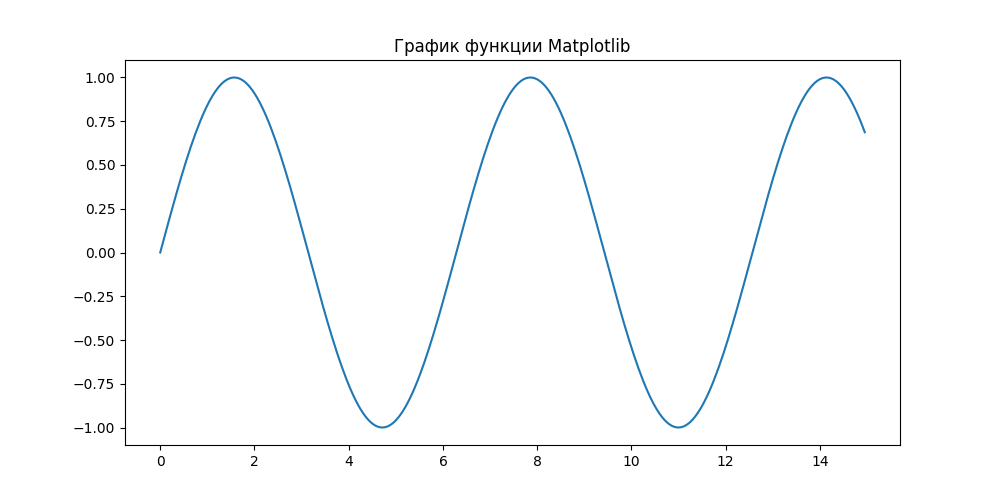
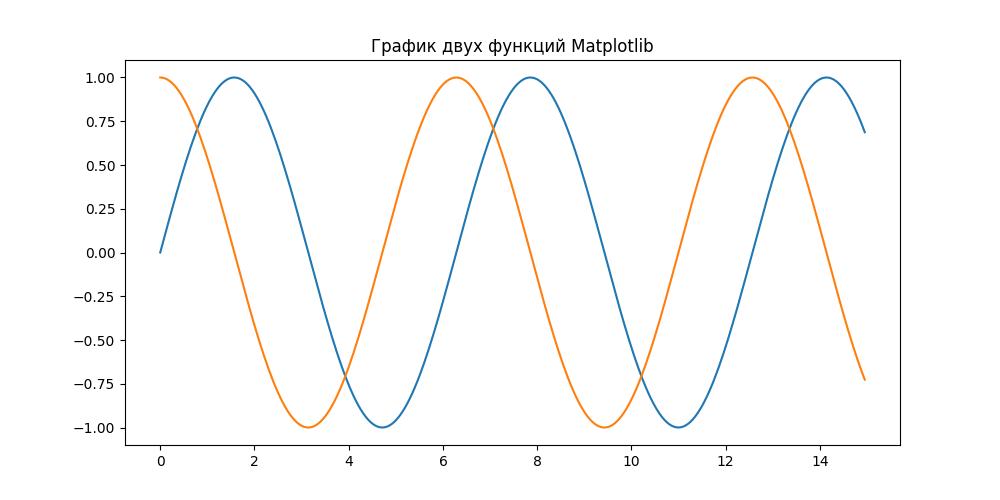


Рис. 1 Графики Загрузка, Разброс, График, Гистограмма, Ящики и Круговая диаграмма выполненные функциями библиотеки Matplotlib.





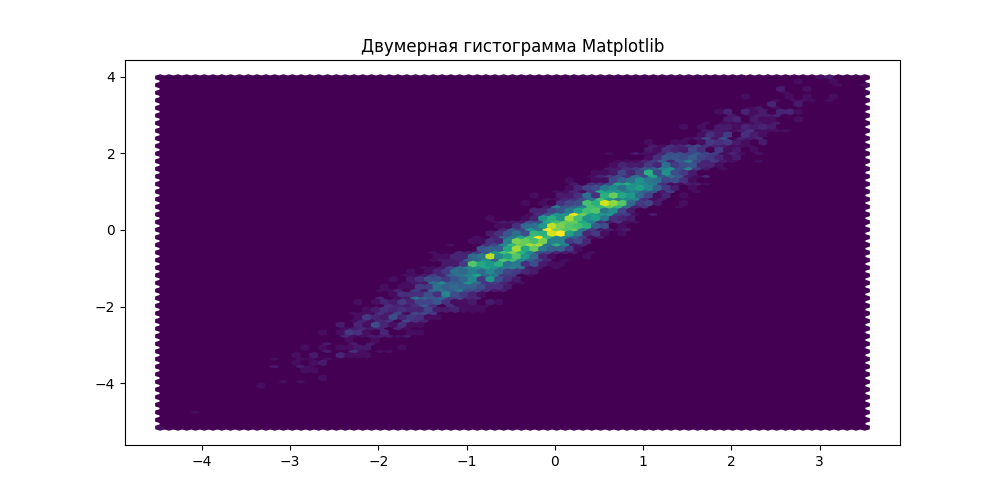
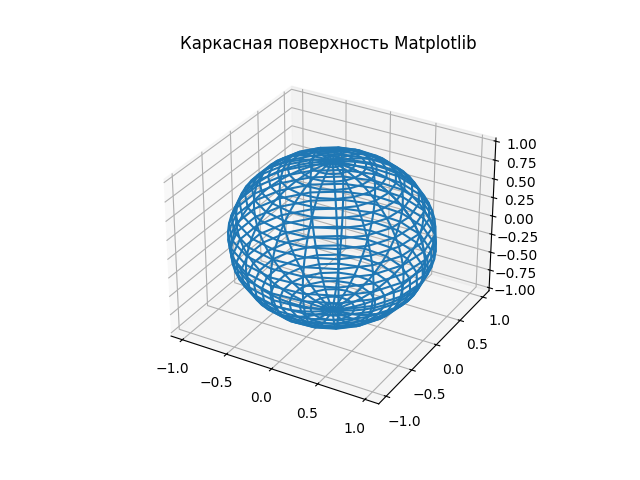
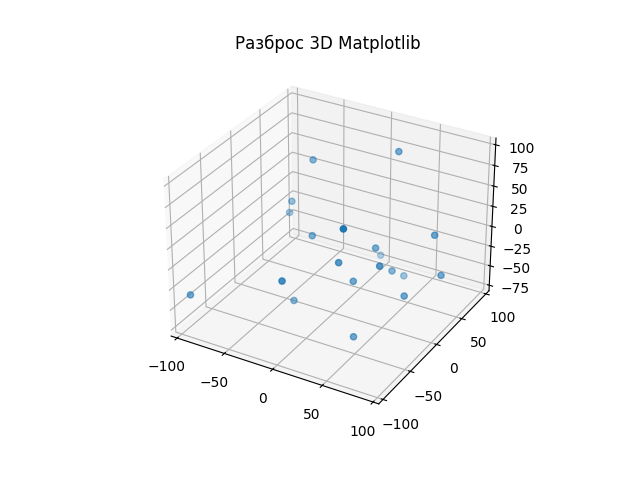


Рис. 2 График функции, График двух функций, Двумерная гистограмма выполненные функциями библиотеки Matplotlib.



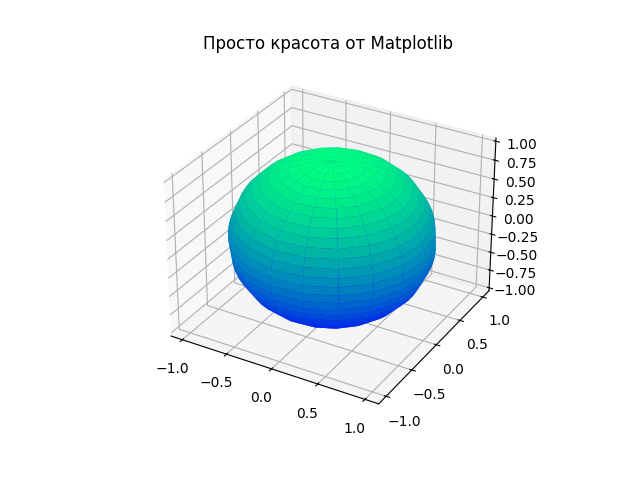
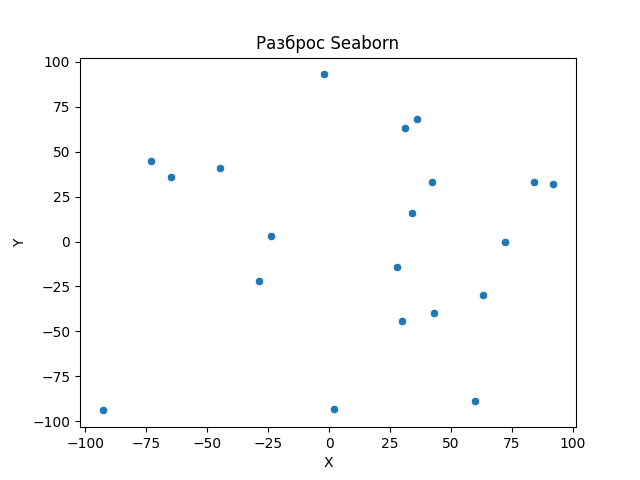
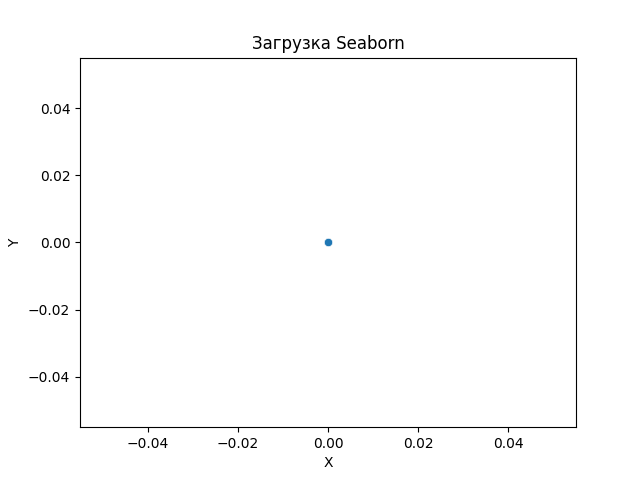


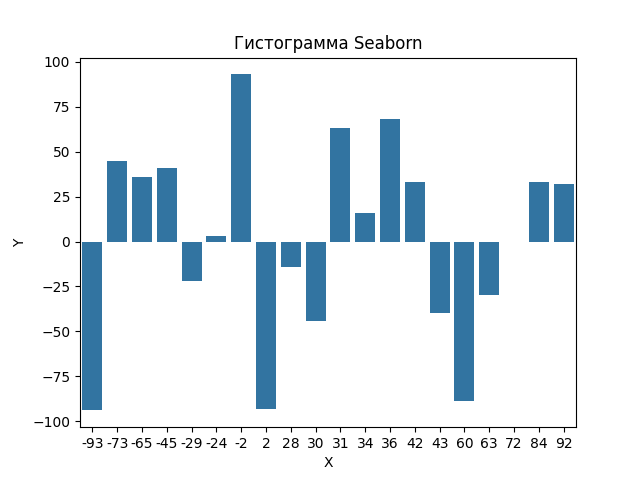
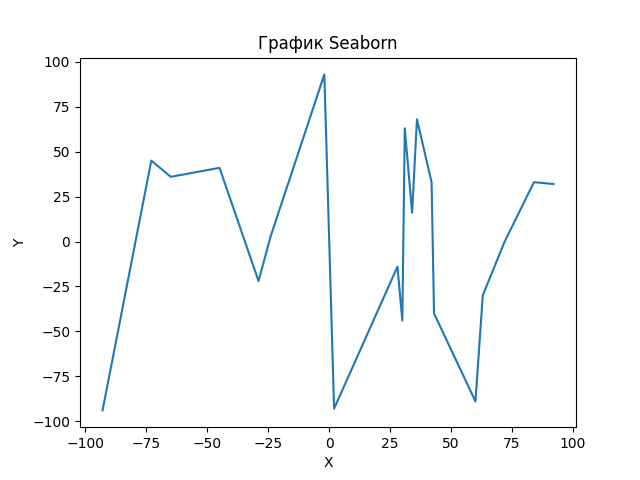
Рис. 3 Разброс 3D, Каркасная поверхность 3D, Просто красота (поверхность 3D) выполненные функциями библиотеки Matplotlib

Для создания визуализации использовались простые случайные данные, такие как списки, кортежи, двумерные списки вычисленные прямо в программе. При каждом запуске программы получаем новые данные.

## Seaborn

Создаём файл связанный с Seaborn. Передаем в него данные для построения. Получаем графики представленные на рис.4 - 6





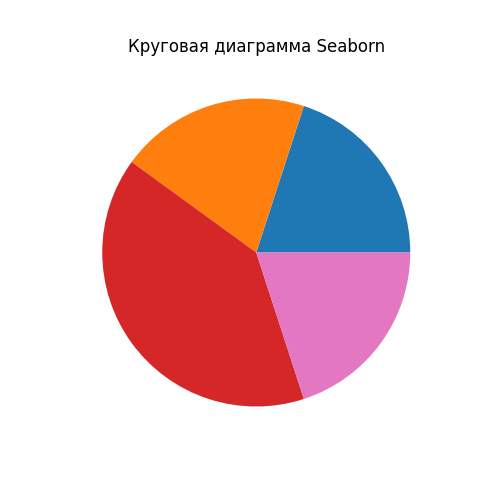
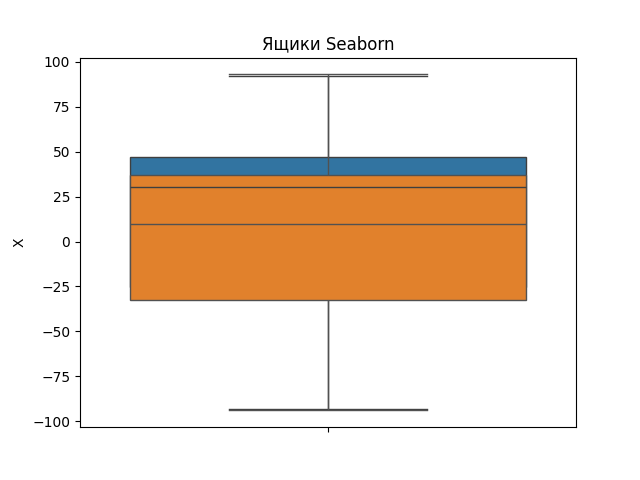
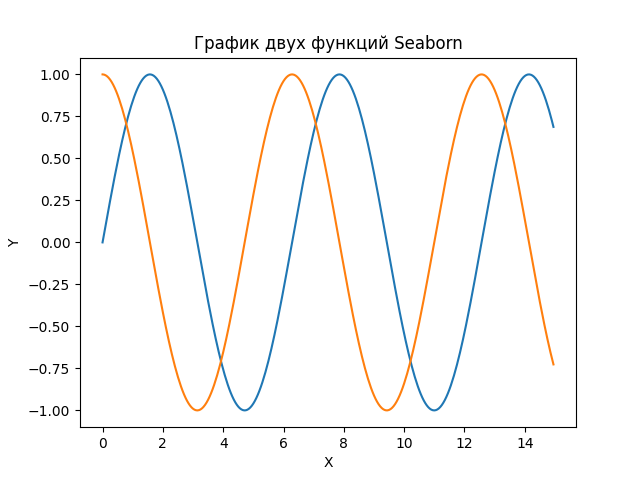
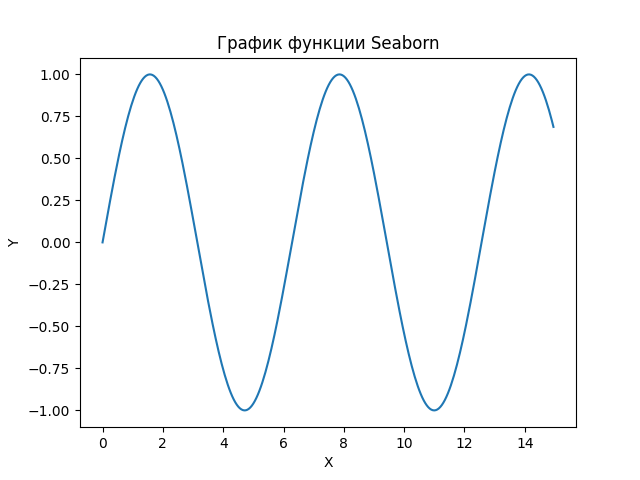


Рис. 4 Графики Загрузка, Разброс, График, Гистограмма, Ящики и Круговая диаграмма выполненные функциями библиотеки Seaborn



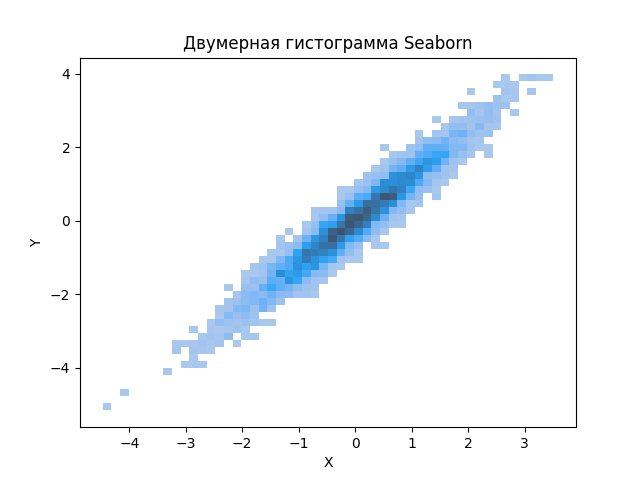
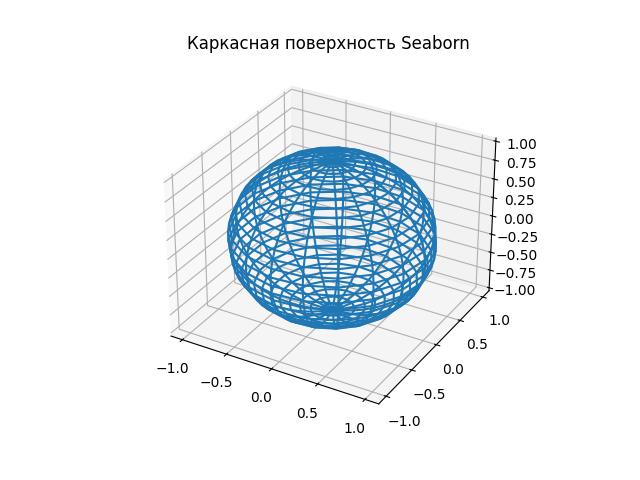
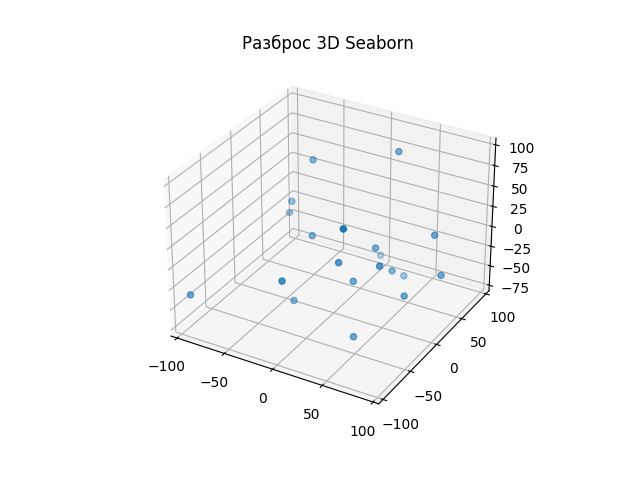


Рис. 5 График функции, График двух функций, Двумерная гистограмма выполненные функциями библиотеки Seaborn.



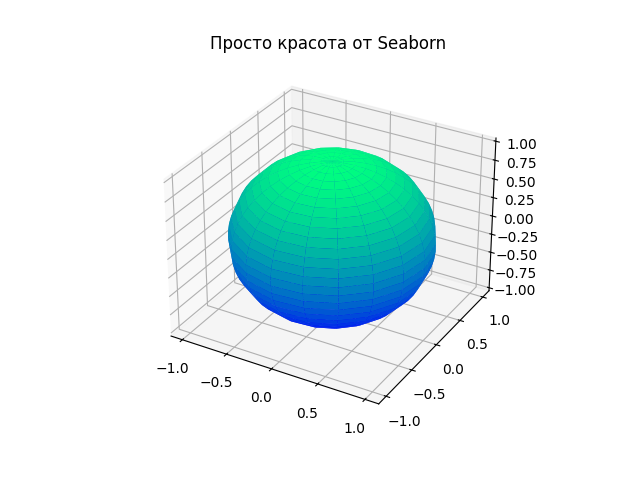
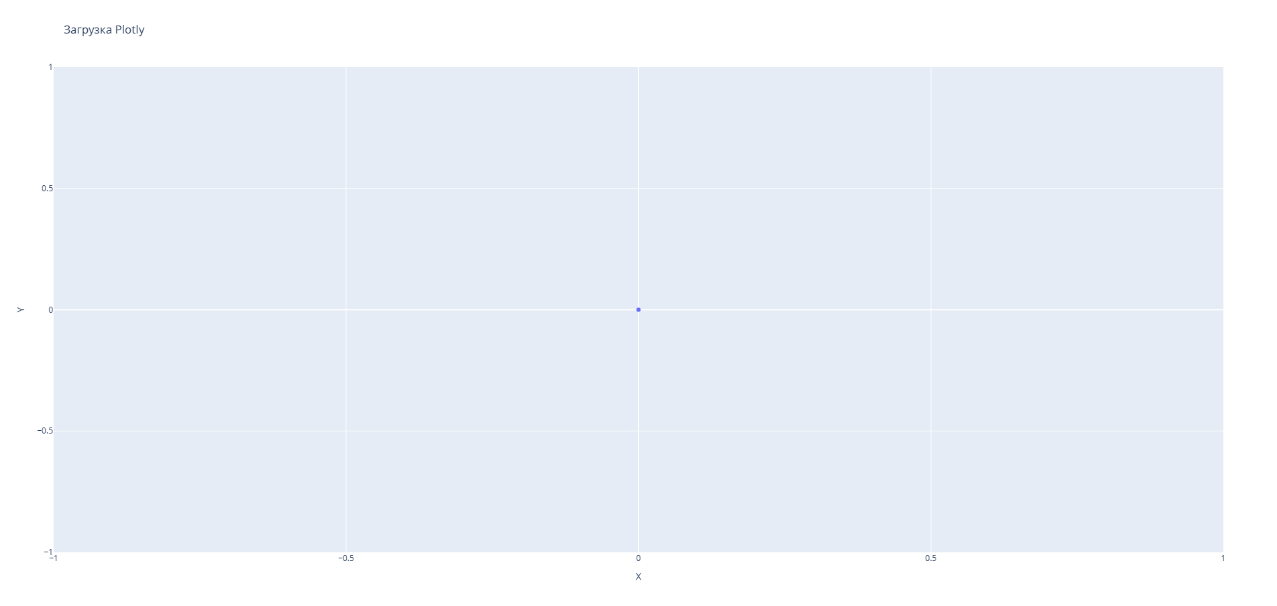


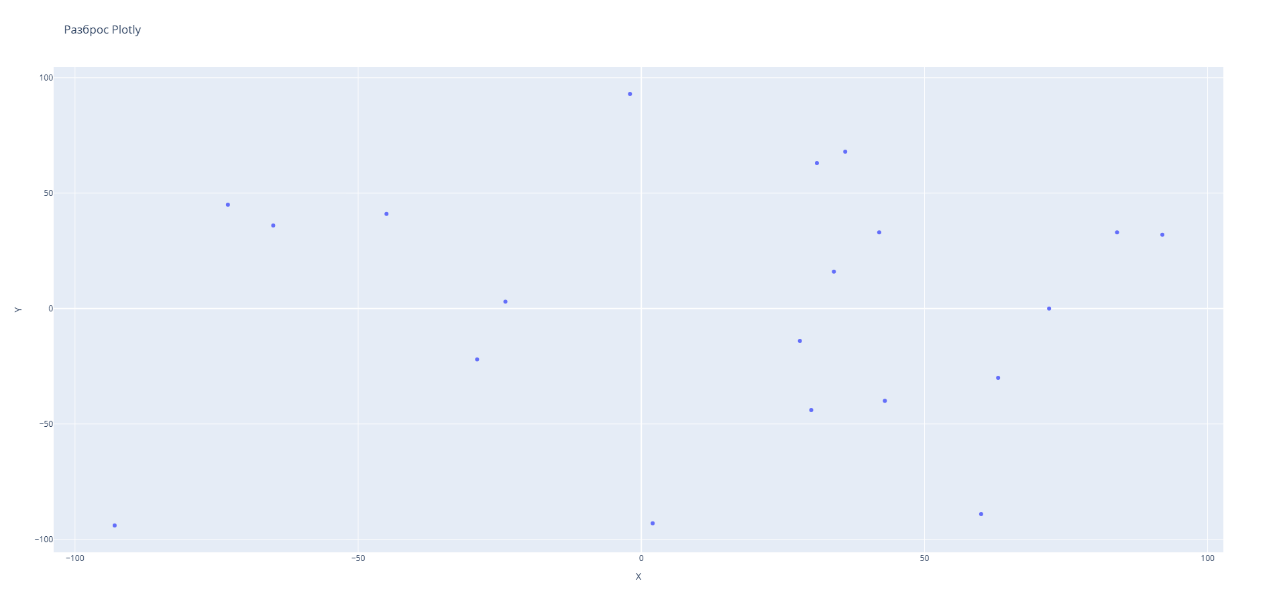
Рис. 6 Разброс 3D, Каркасная поверхность 3D, Просто красота (поверхность 3D) выполненные функциями библиотеки Seaborn (для чистоты собранных статистических данных).

Для создания визуализации использовались DataFrame полученные из простых случайных данных, которые получили для использования в функциях библиотеки Matplotlib. При каждом запуске программы получаем новые данные.

## Plotly

Создаём файл связанный с Plotly. Передаем в него данные для построения. Получаем графики представленные на рис.7 - 10





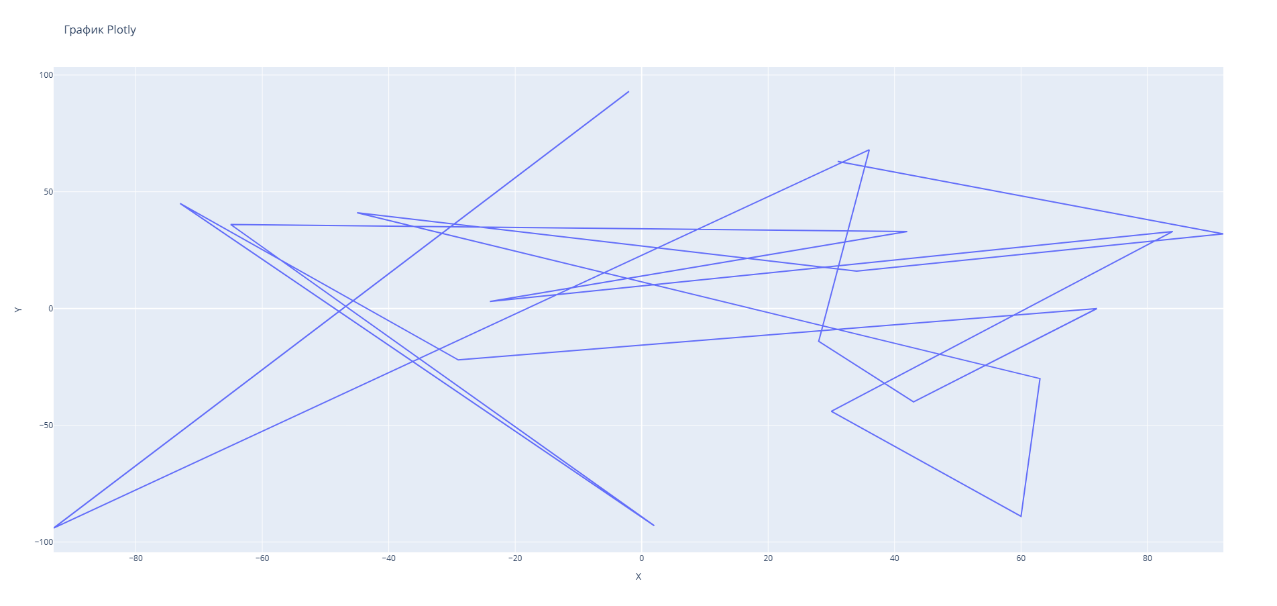
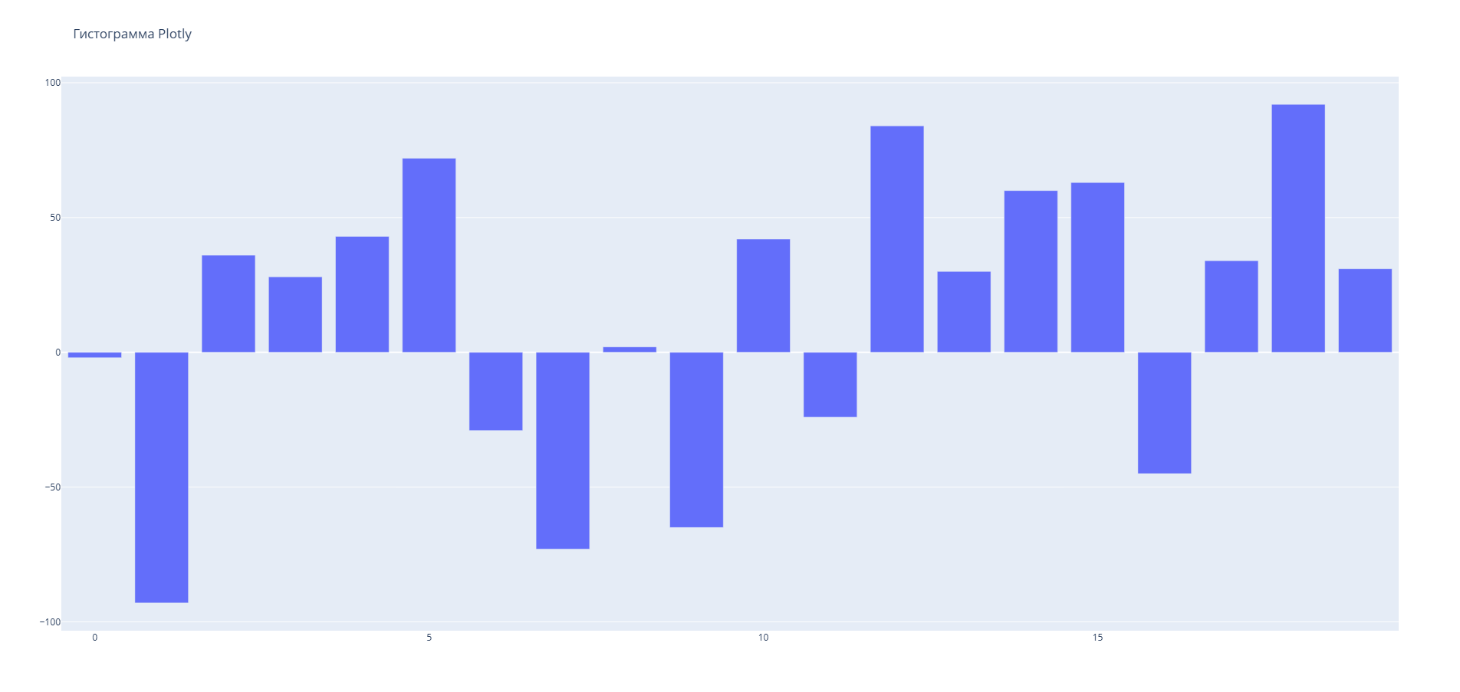
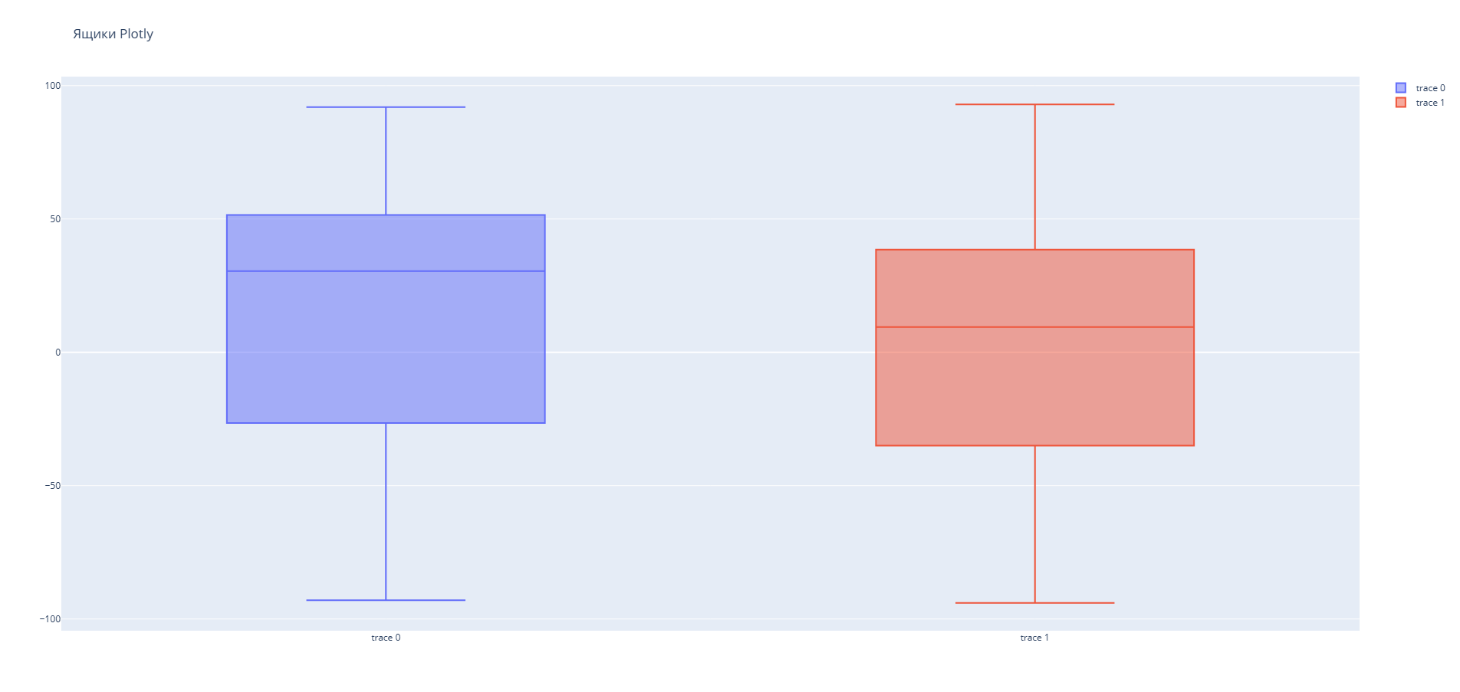


Рис. 7 Графики Загрузка, Разброс, График выполненные функциями библиотеки Plotly





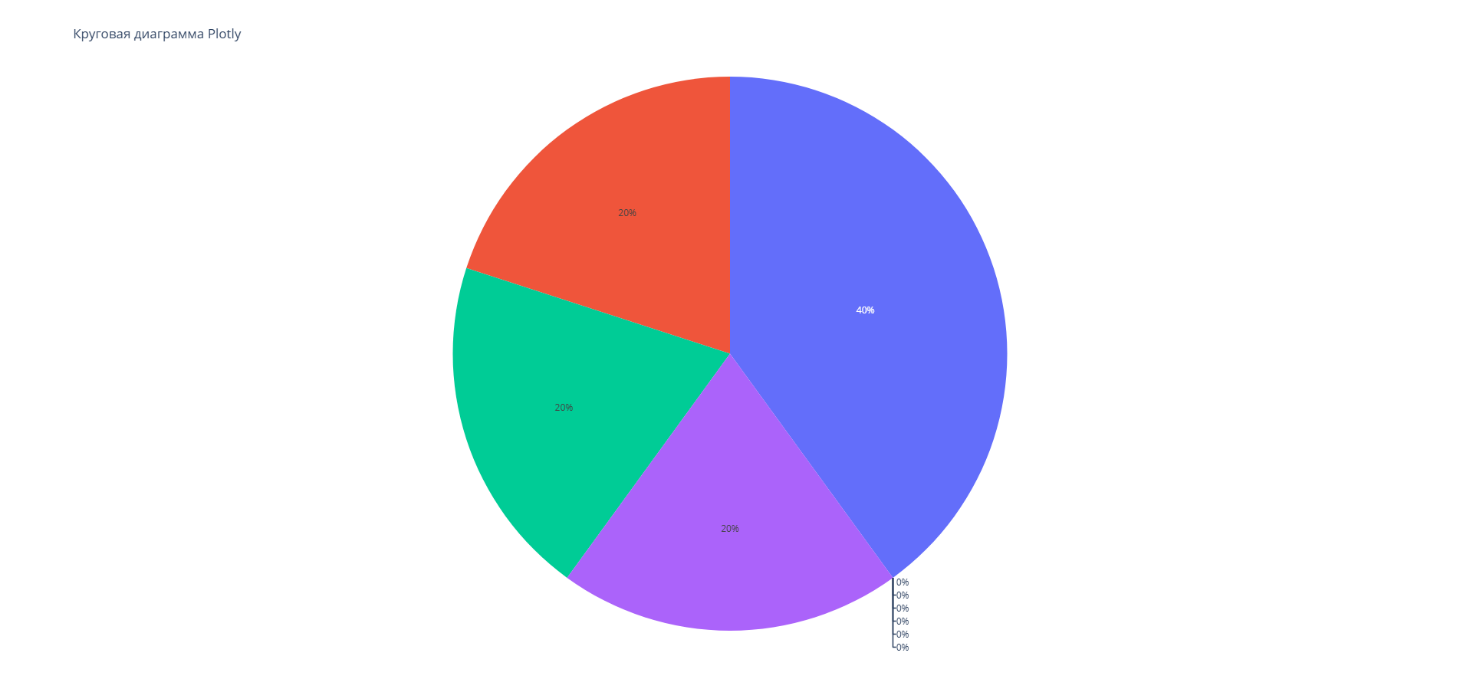
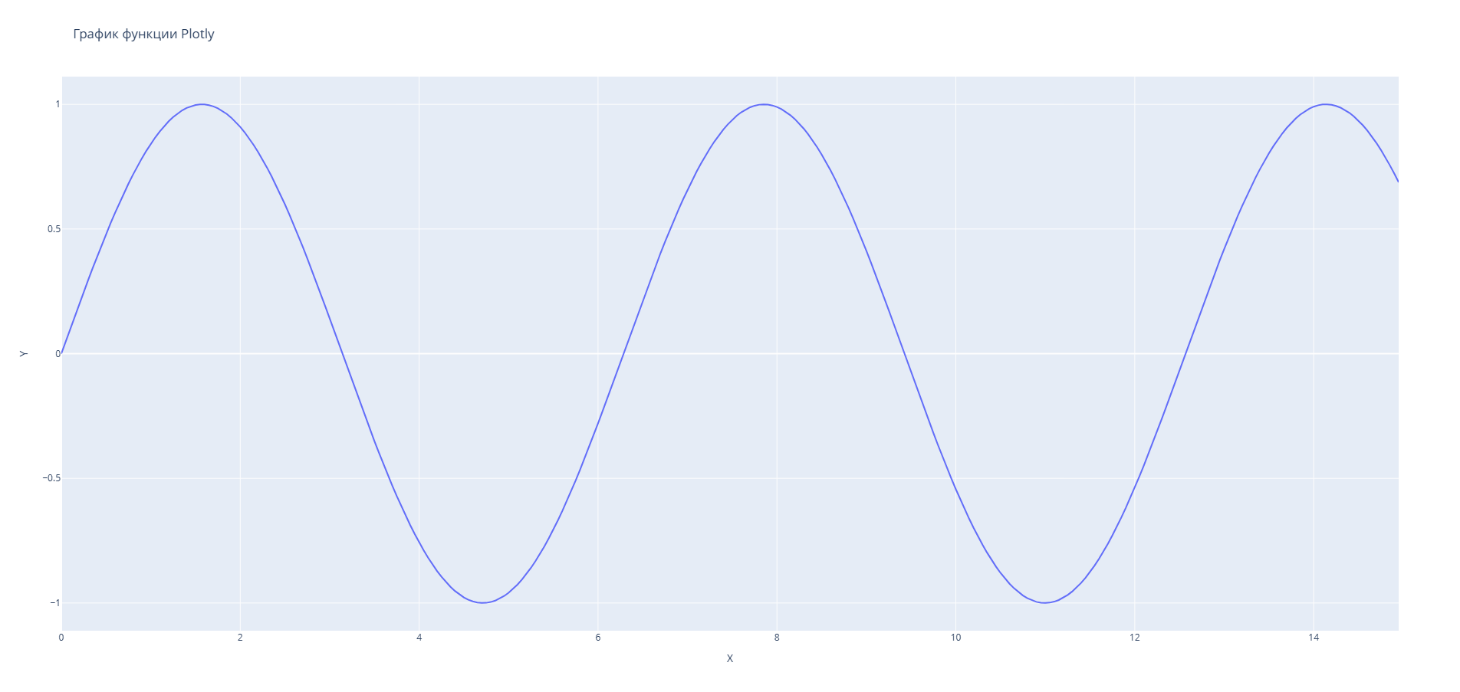
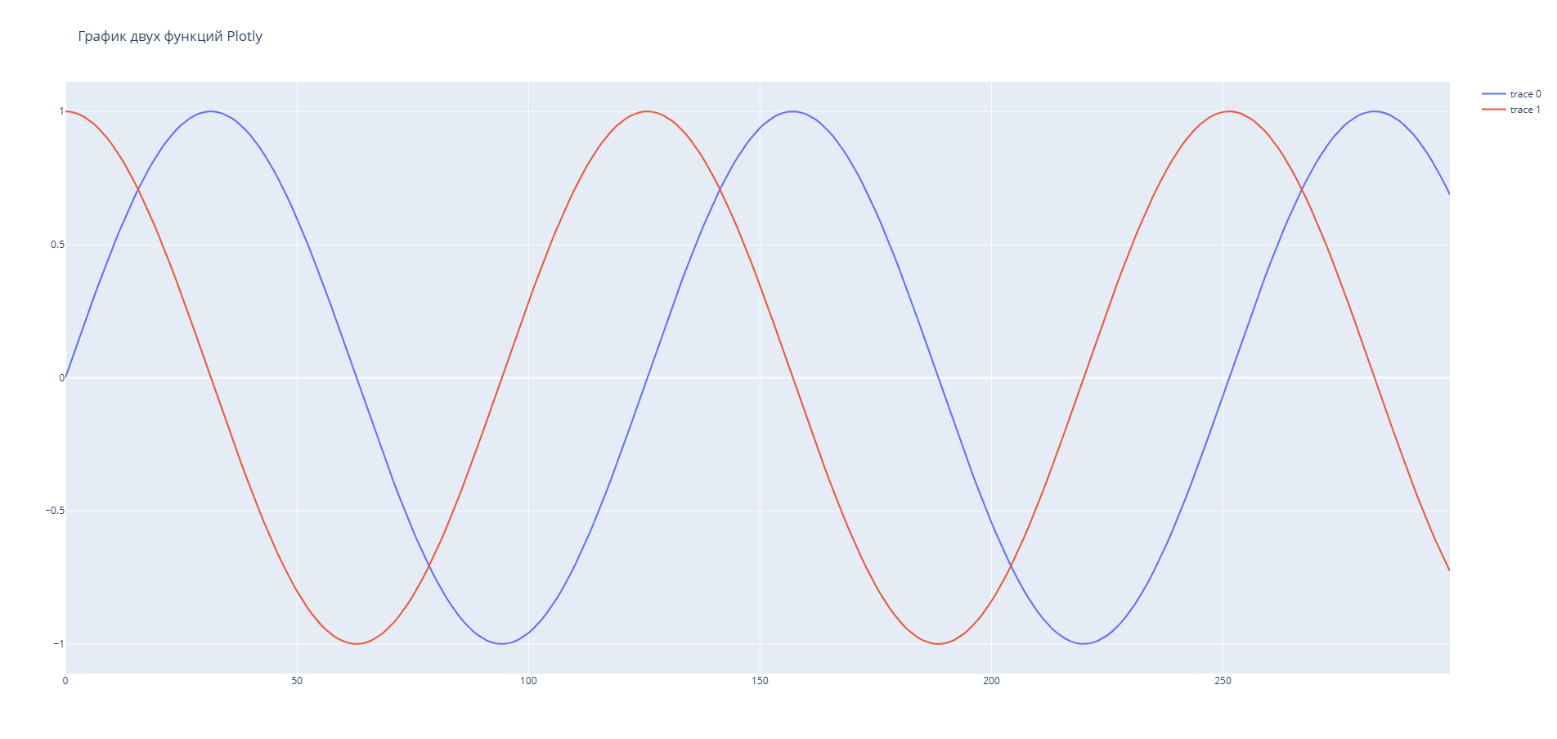


Рис. 8 Графики Гистограмма, Ящики и Круговая диаграмма выполненные функциями библиотеки Plotly





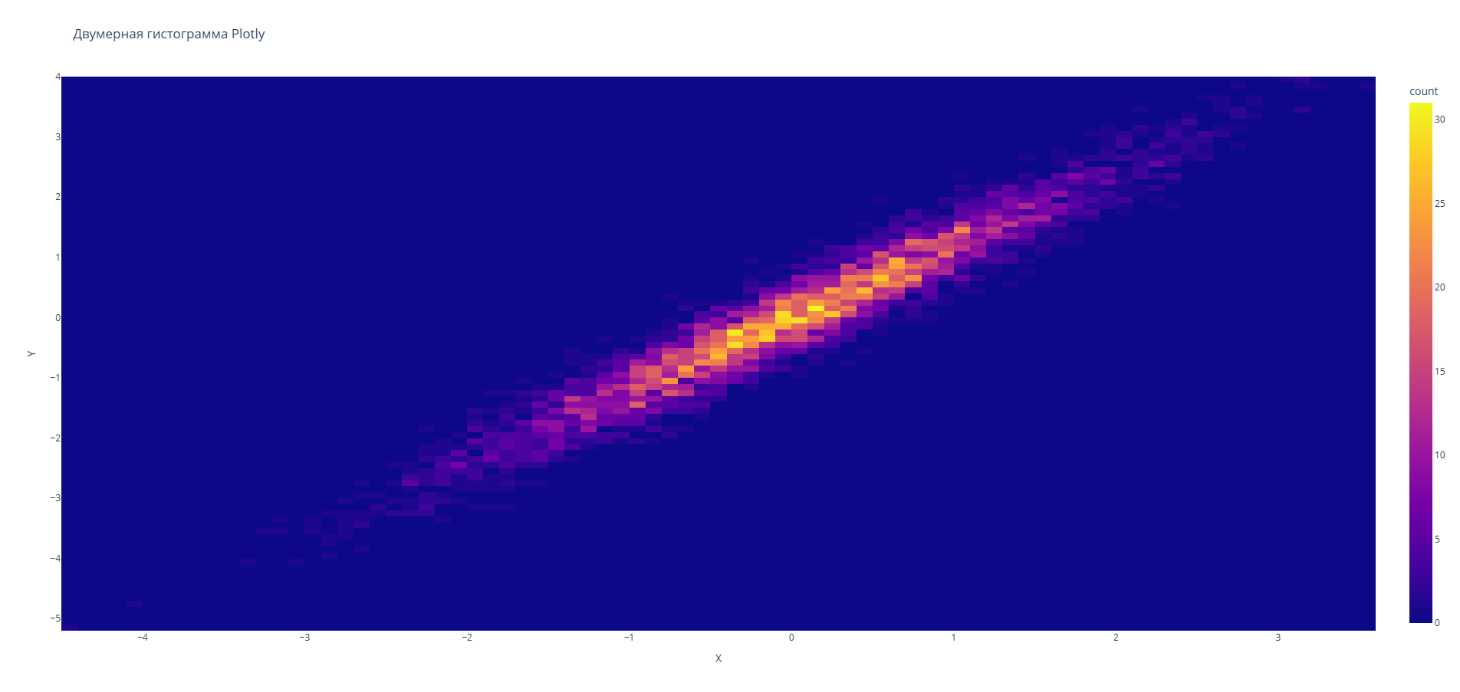
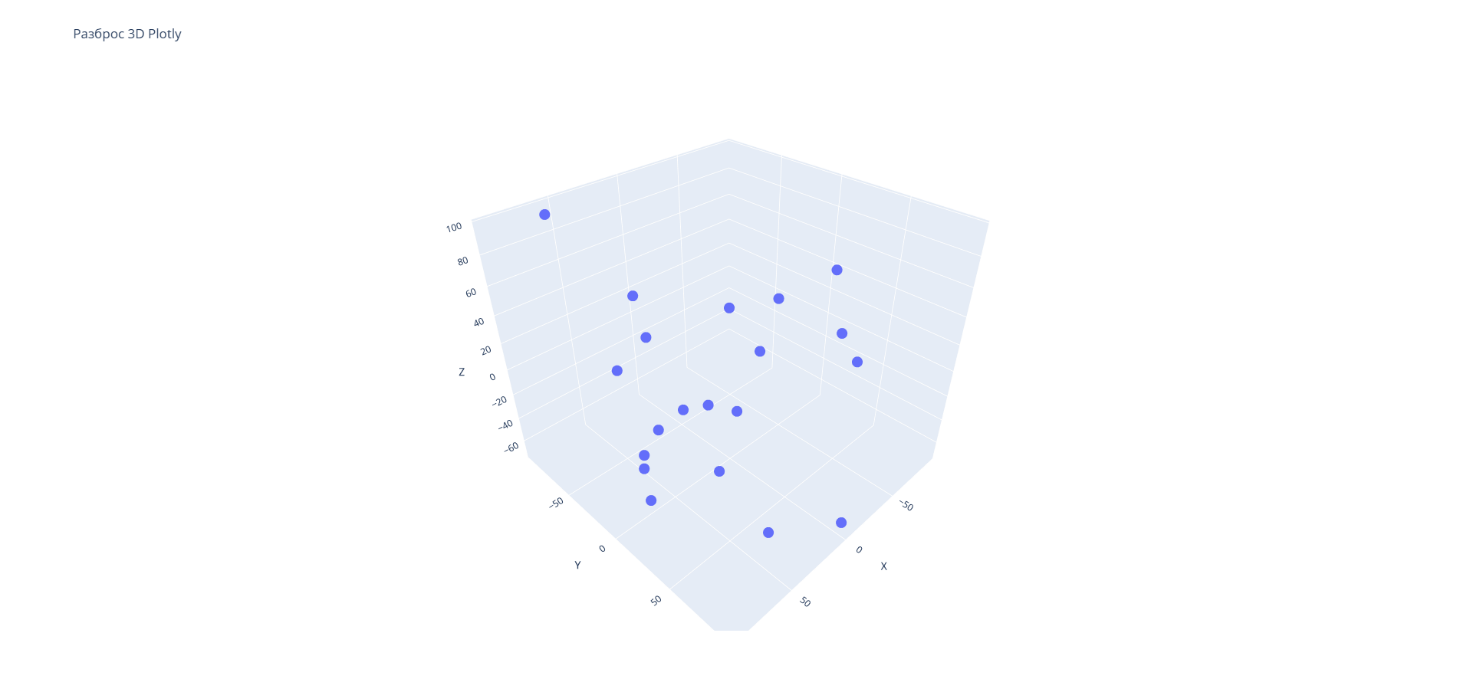
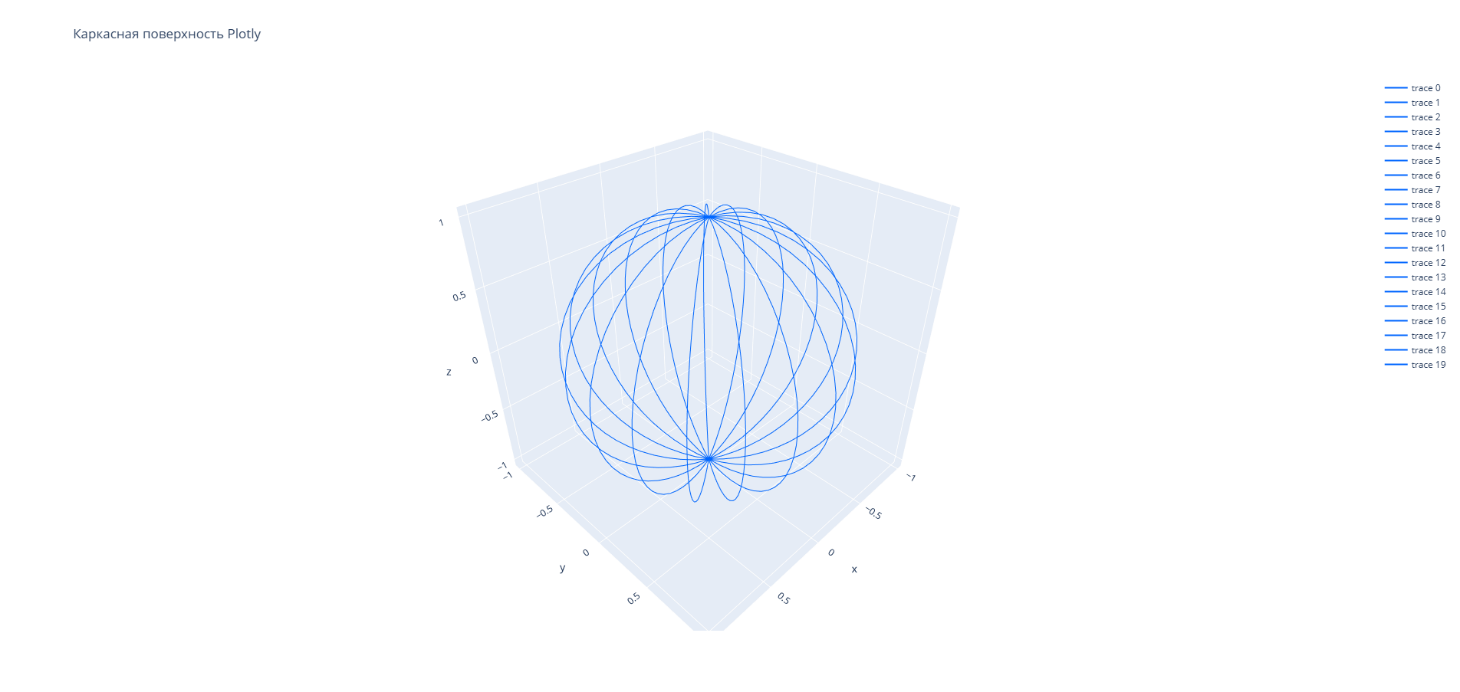


Рис. 9 График функции, График двух функций, Двумерная гистограмма выполненные функциями библиотеки Plotly





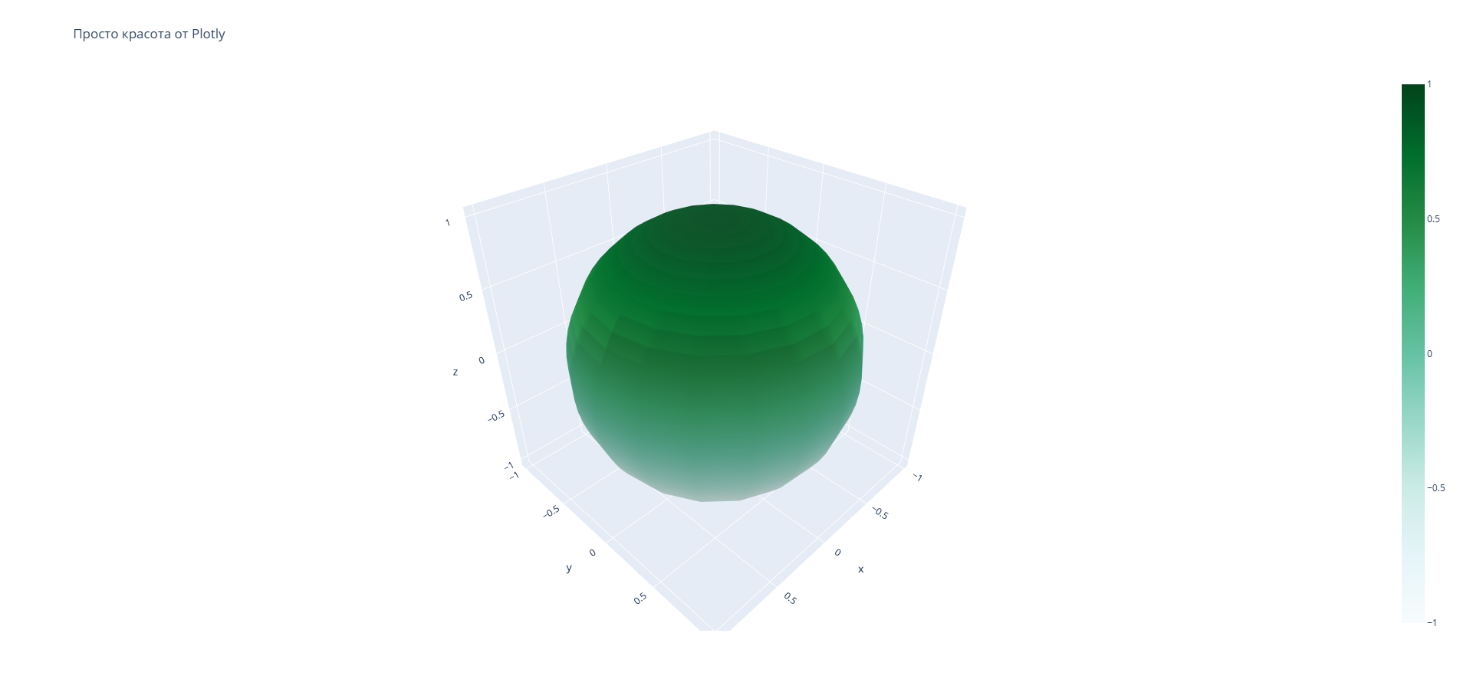


Рис. 10 Разброс 3D, Каркасная поверхность 3D, Просто красота (поверхность 3D) выполненные функциями библиотеки Plotly

Здесь немного другая логика построения графиков. Создаём графики через plotly.express, plotly.graph\_objects. В созданном окне можно изучить возможности масштабирования графиков и остальные возможности, включая интерактивное включение и отключение элементов.

Данная библиотека обладает гораздо более сложным синтаксисом, но большей функциональностью и встроенной поддержкой веба. Для создания визуализации использовались как DataFrame полученные из простых случайных данных, так и данные, которые получили для использования в функциях библиотеки Matplotlib. При каждом запуске программы получаем новые данные.

# **4. Анализ и интерпретация результатов**

## Сравнение библиотек

В ходе работы, были продемонстрированы аналогичные инструменты из библиотек Matplotlib, Seaborn, Plotly. Были собраны статистические данные работы всех трех библиотек. Таблица с данными в базе данных представлена на рис.11, графики с результатами пяти запусков программы представлены на рис.12-16.

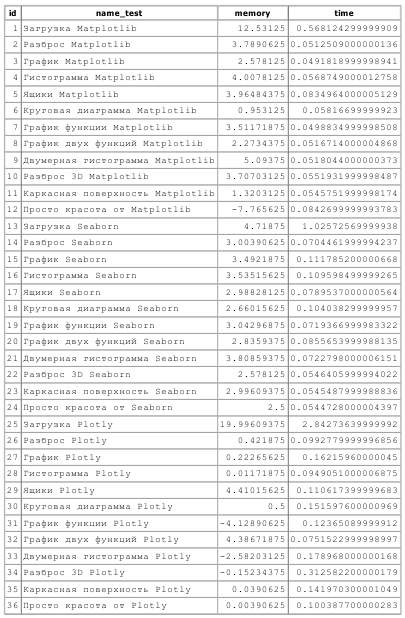


Рис. 11 Таблица результатов тестов полученная из data.db

При каждом запуске программы таблица очищается и заполняется заново.

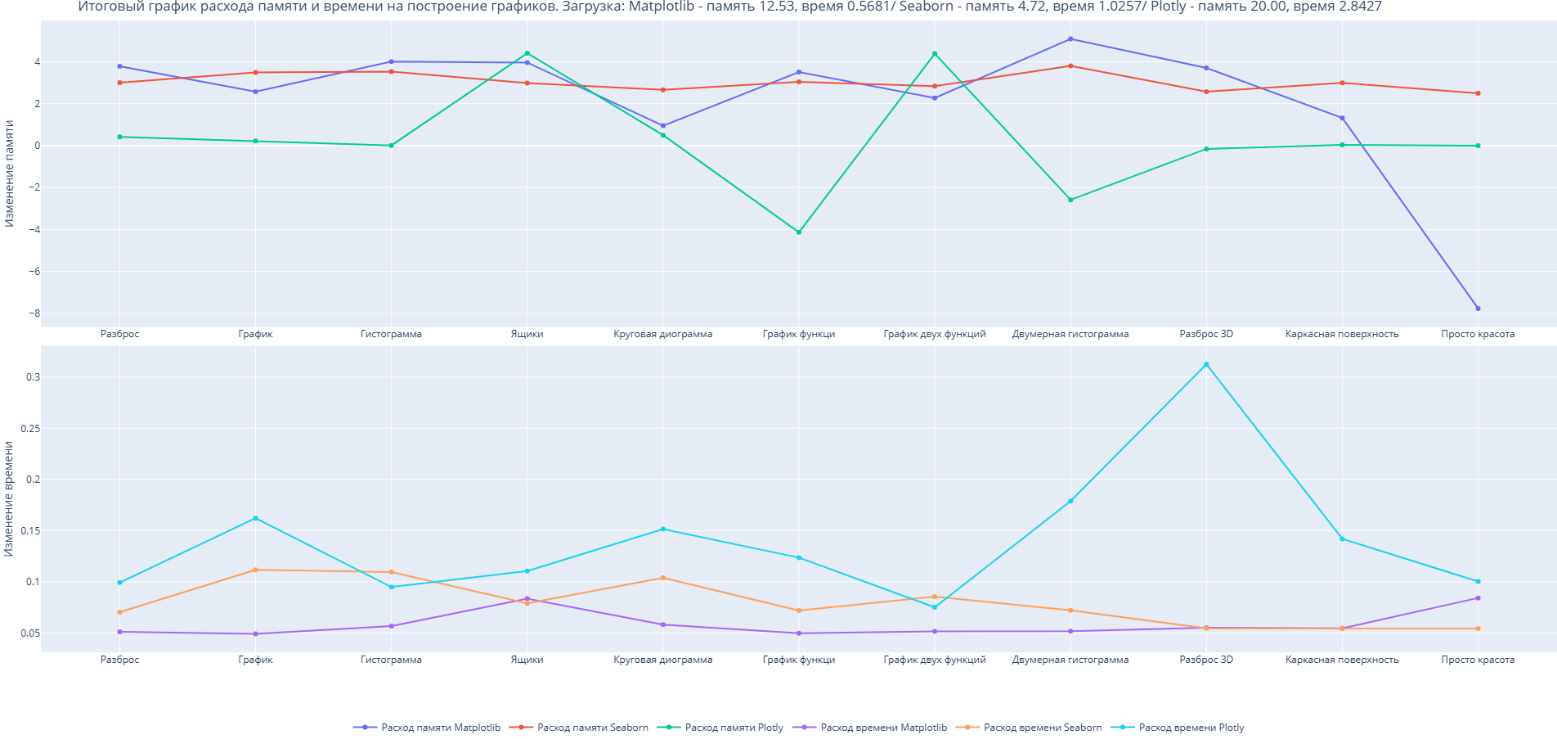


Рис. 12 График результатов после первого запуска программы. В название графика внесены данные из загрузки библиотек – память и время.

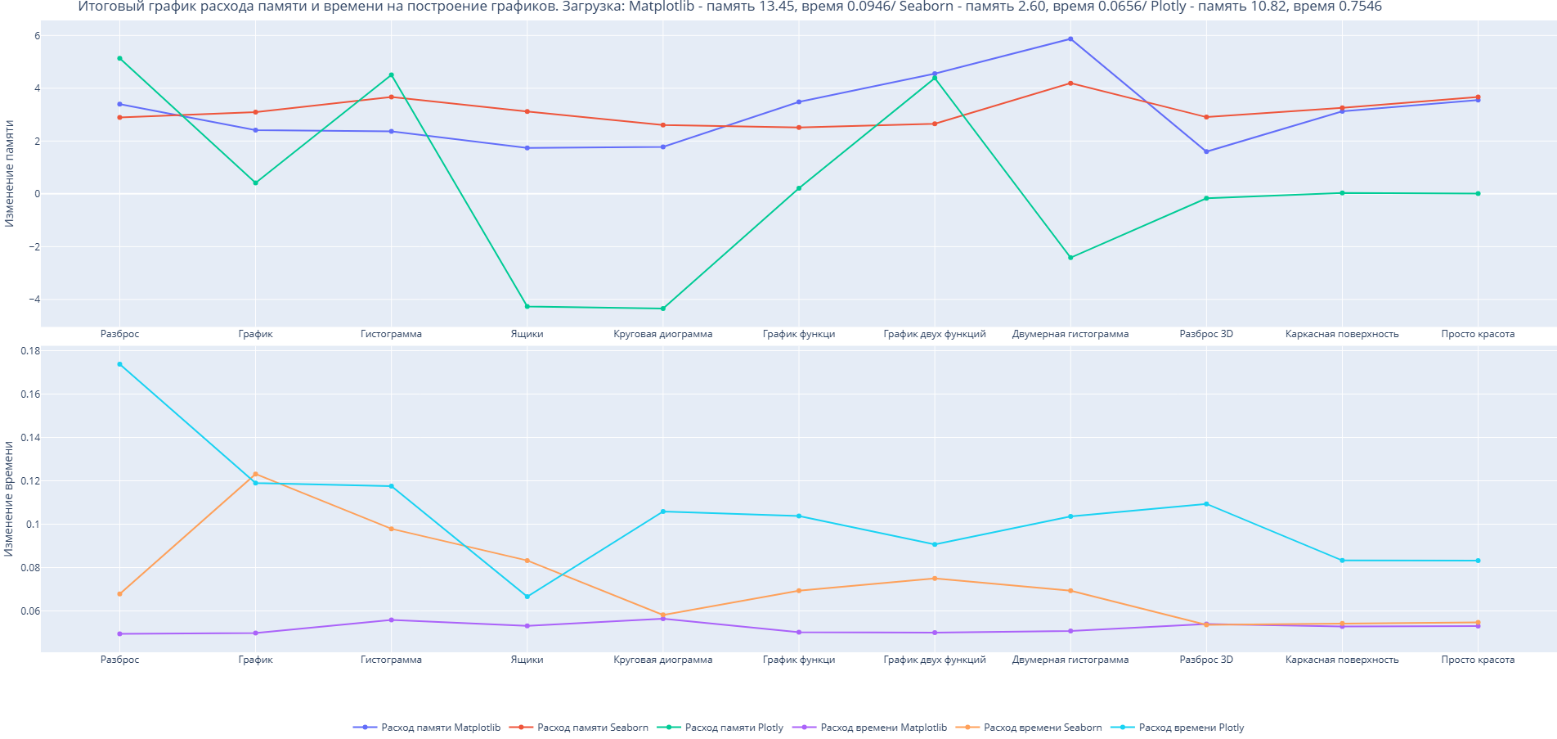


Рис. 13 График результатов после второго запуска программы. В название графика внесены данные из загрузки библиотек – память и время.

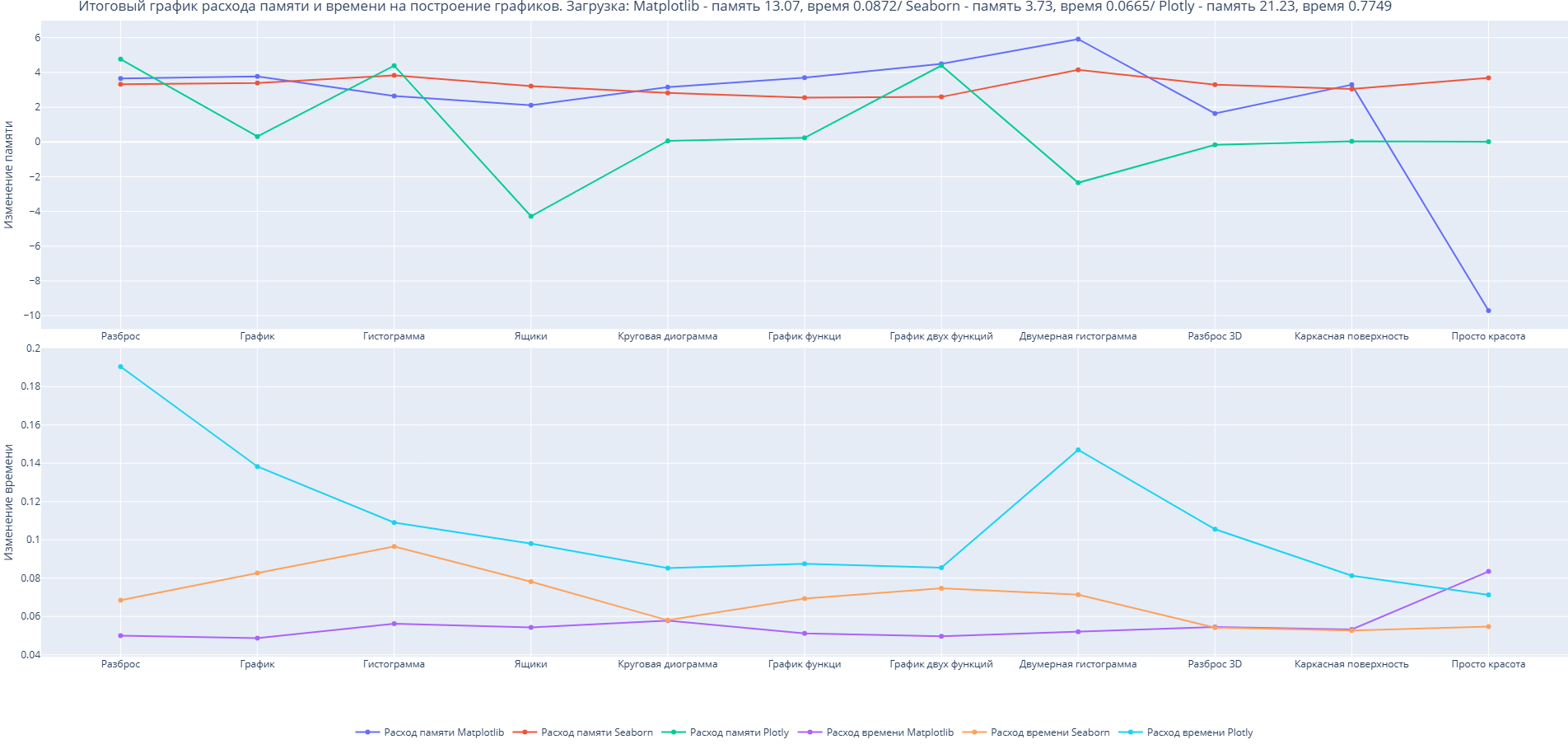


Рис. 14 График результатов после третьего запуска программы. В название графика внесены данные из загрузки библиотек – память и время.

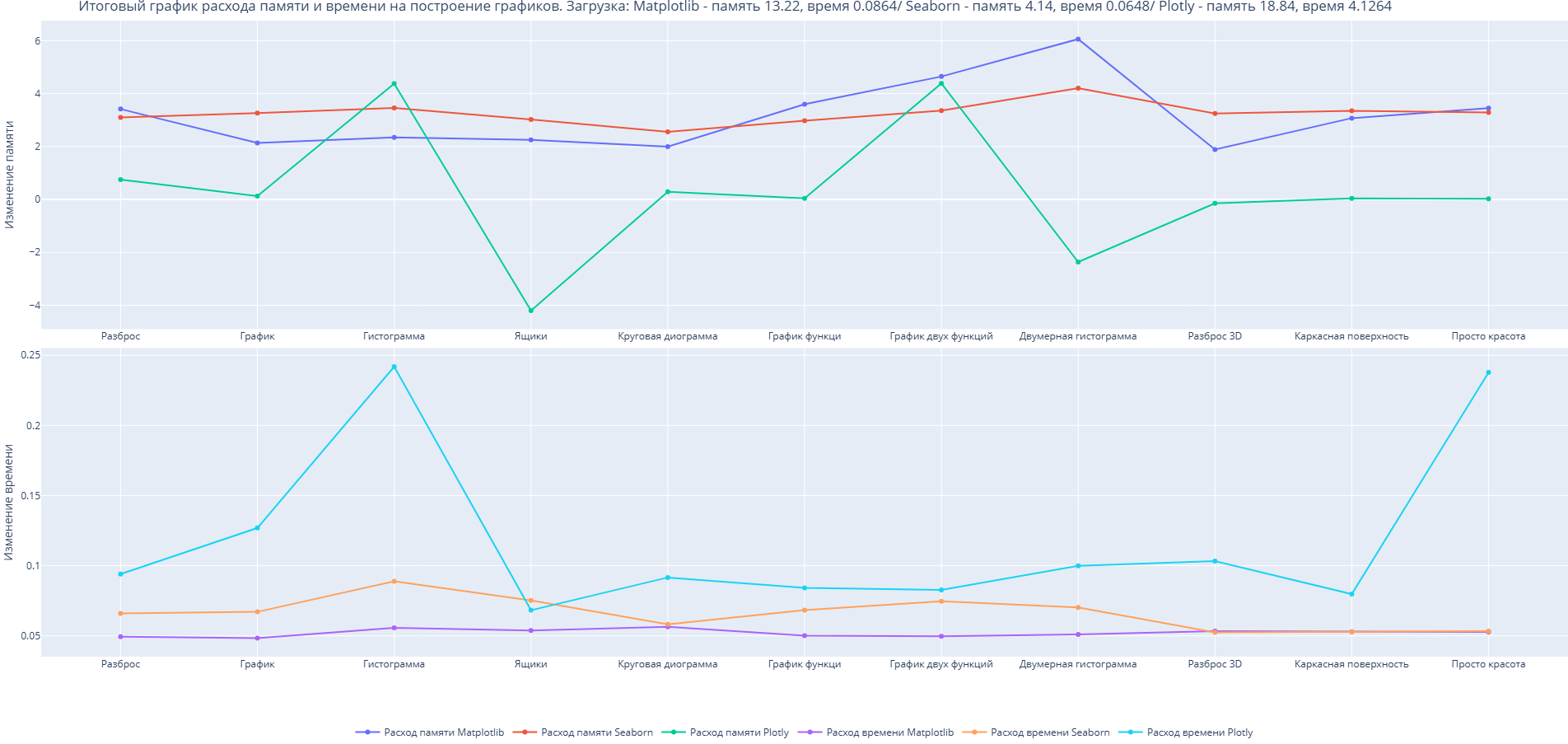


Рис. 15 График результатов после четвертого запуска программы. В название графика внесены данные из загрузки библиотек – память и время.

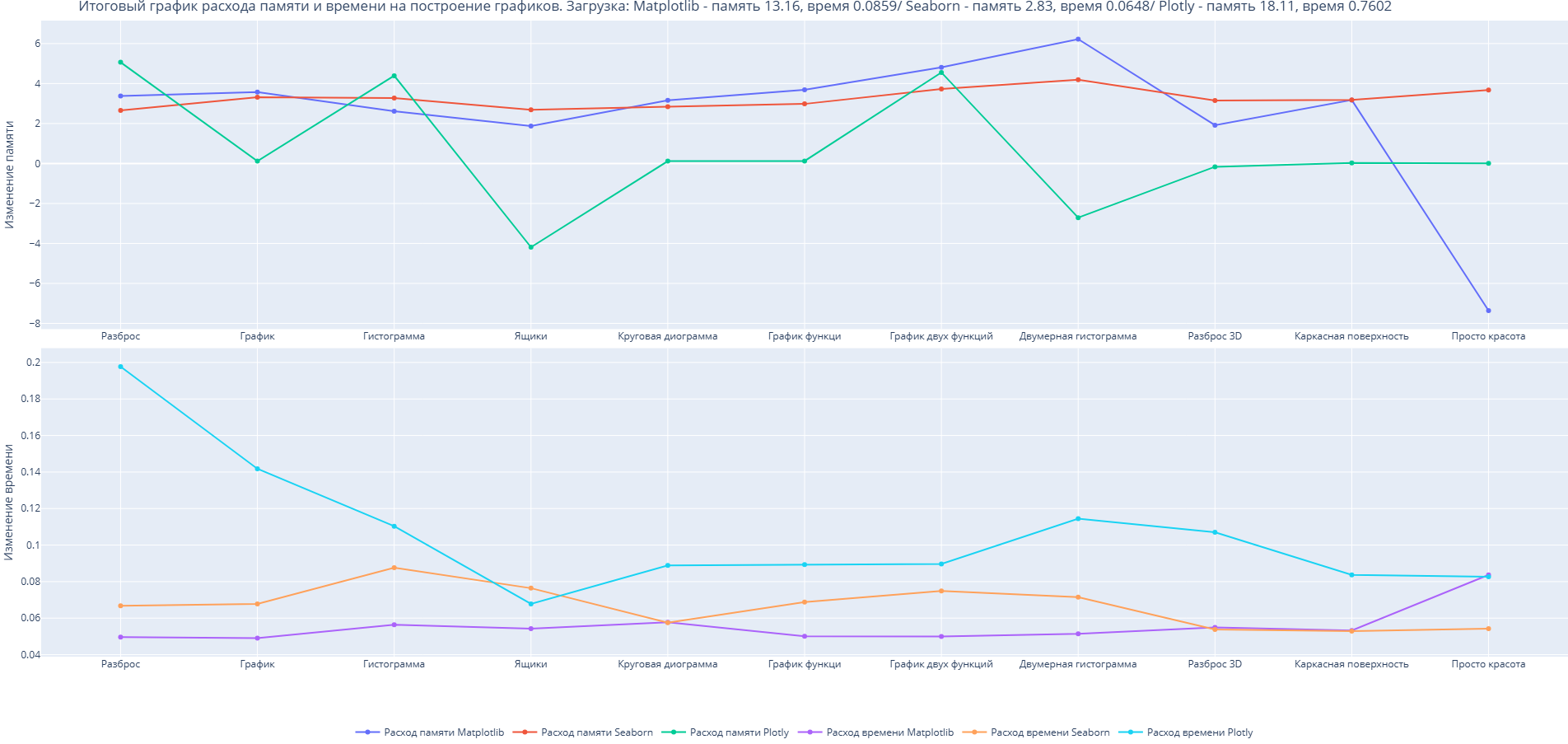


Рис. 16 График результатов после пятого запуска программы. В название графика внесены данные из загрузки библиотек – память и время.

## Интерпретация результатов и рекомендации к использованию

Как видно из графиков при первой загрузке любой библиотеки происходит самая большая загрузка памяти и самая большая потеря времени. При использовании на маломощной конфигурации компьютера или сервера это надо учитывать.

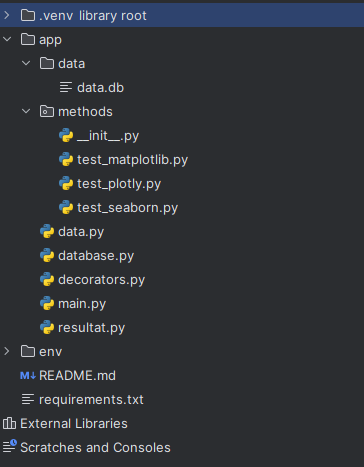
Бесспорным лидером по потерям при загрузке является библиотека Plotly, на втором месте идет библиотека Seaborn так как при её загрузке также загружается материнская библиотека Matplotlib (это видно по отсутствию скачка памяти на круговой диаграмме и 3D графиках, которые отсутствуют в библиотеке Seaborn). Самой быстрой и производительной библиотекой при загрузке является Matplotlib.

При дальнейшей работе самой затратной по памяти библиотекой является Seaborn, второй Matplotlib и соответственно третьей Plotly. По быстродействию на первом месте Matplotlib, на втором Seaborn, на третьем Plotly.

1. **Matplotlib**  
   Функциональность: Основной инструмент для создания визуализаций в Python с гибкой настройкой. Поддерживает 2D- и 3D-графики, такие, как гистограммы, графики рассеяния и контурные диаграммы.  
   Удобство использования: требует больше кода, особенно для сложных визуализаций, и имеет простой, но менее эстетичный дизайн по умолчанию. Использование прямых расчетов без конвертации в базу данных и обратно также ускоряет работу и упрощает написание кода.  
   Лучшее применение: подходит для научных и исследовательских проектов, где требуется строгий контроль над деталями.
2. **Seaborn**  
   Функциональность: Построен поверх Matplotlib, предлагая высокоуровневые API для быстрого создания стильных графиков и работы с данными. Включает встроенные статистические графики (например, boxplots, violinplots, heatmaps).  
   Удобство использования: упрощает создание графиков и легко интегрируется с Pandas. Меньше необходимости в настройке стилей, так как Seaborn использует красивые, готовые по умолчанию стили. Удобен при использовании DataFrame – считывание данных прямо из баз данных.  
   Лучшее применение: Идеален для исследовательского анализа баз данных (EDA), особенно когда нужны быстро и легко создаваемые статистические графики.
3. **Plotly**  
   Функциональность: поддерживает интерактивные графики, 3D-диаграммы, карты и другие сложные визуализации. Можно встраивать в веб-приложения и взаимодействовать с графиками в режиме реального времени.  
   Удобство использования: Графики не так просты в создании, но поддерживают интерактивность. Библиотека может быть сложнее для адаптации по сравнению с Seaborn или Matlibplot. Универсальна при обработке разных типов данных, вычисляемых и готовых баз данных.  
   Лучшее применение: прекрасно подходит для дашбордов и аналитики, особенно если визуализации должны быть интерактивными или отображаться в веб-приложениях.

Из выше сказанного можно сделать вывод: если нет необходимости учитывать производительность, но есть желание увидеть красивые и интерактивные графики прямо на веб странице то можно использовать библиотеку визуализации Plotly. В случае ограниченных возможностях используем Matplotlib, мене эффектную, но быстрою. Seaborn универсал для баз данных - красивый но плохо настраиваемый, со своей логикой построения графиков, что можно победить но придётся постараться.

# **5. Структура программы**



# **6. Заключение**

## Обзор выполненной работы

Подводя итоги, все три библиотеки приемлемы для визуализации данных. Каждая библиотека имеет свои преимущества в зависимости от входных данных и целей: Matplotlib быстро (можно сказать на лету), Seaborn быстро для внешних данных и Plotly красиво для интерактивных приложений с разными входными данными.

## Дальнейшие планы

Возможны улучшения программы: увеличение списка исследуемых графиков, добавление выбора стилистики отдельных графиков и групп, возможность их объединения, реализация добавления пользовательских данных, вывода визуализаций через веб-сервер для анализа быстродействия в интерактивном режиме.

# **7. Список литературы**

<https://matplotlib.org/stable/users/index.html> - документация Matplotlib

<https://seaborn.pydata.org/tutorial.html> - документация Seaborn

<https://plotly.com/python/> - документация Plotly