|  |
| --- |
| МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ |
| ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ |
| УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ |
| **«СЕВАСТОПОЛЬСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»** |
|  |
| Институт информационных технологий и управления в технических системах |
| (полное название института) |
|  |
| кафедра «Информационные системы» |
| (полное название кафедры) |
|  |
|  |
| **Лабораторные работы** |
| по дисциплине «Информационный анализ данных» |

Выполнил:

студент группы

ИС/б-24-1-о

Таран Юлия Сергеевна

Принял:

старший преподаватель

Сырых Ольга Анатольевна

|  |  |  |  |
| --- | --- | --- | --- |
| 20 | 24 | г. | Г. |

Лабораторная работа № 1

«Исследование возможностей языка R для статистического анализа данных»

1.1. Цель работы

Исследовать возможности языка R для визуализации данных в R.

1.2. Постановка задачи

1) Выполнить все примеры, содержащие в методических рекомендация, изменяя параметры функций.

2) Смоделировать независимые случайные векторы (X, Y), имеющие гауссовское распределение с заданным математическим ожиданием α и корреляционной матрицей R заданными по варианту (рисунок 1). Визуализировать данные на плоскости в виде точек.

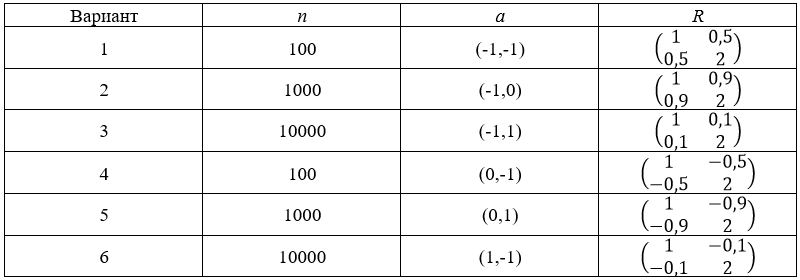


Рисунок 1 – Варианты задания для моделирования данных в задании 2.

3) Используя реальные статистические данные из заданного набора Ирисы Фишера:

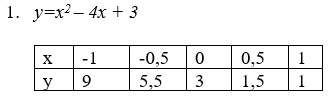
Ссылка: <https://archive.ics.uci.edu/ml/datasets/Iris>

Первый признак: Sepal.Length (столбец № 2)

Второй признак: Sepal.Width (столбец № 3)

визуализировать данные на плоскости, используя 4-5 разных видов графиков.

4) Построить график функции y = f(x), заданной по варианту:



5) Построить в том же графическом окне график функции y=f(x), заданной таблично.

6) Постройте в новом графическом окне графики функций реализации запчастей 1, 2 и 3 групп за 2023 год по месяцам:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Группа** | **Январь** | **Февраль** | **Март** | **Апрель** | **Май** | **Июнь** |
| I группа | 30350,4 | 30350,4 | 94351,8 | 75810,6 | 57860,4 | 104943,6 |
| II группа | 6525 | 6525 | 73198,8 | 14310 | 29550 | 18716,4 |
| III группа | 10230 | 10230 | 69480 | 18874,8 | 43176,6 | 70900,2 |
| **Группа** | **Июль** | **Август** | **Сентябрь** | **Октябрь** | **Ноябрь** | **Декабрь** |
| I группа | 80710,5 | 91177,2 | 113397,6 | 89148 | 60709,2 | 49272,3 |
| II группа | 26282,4 | 39085,8 | 96808,8 | 215320,8 | 73904,4 | 75894 |
| III группа | 50358 | 77707,2 | 87267 | 86396,4 | 24232,8 | 17842,8 |

1.3. Ход работы

1.3.1. Выполнение примеров

1) Использование функции plot(x, y,…) для построения графика функции y = cos(2x), с разным значением параметра type:

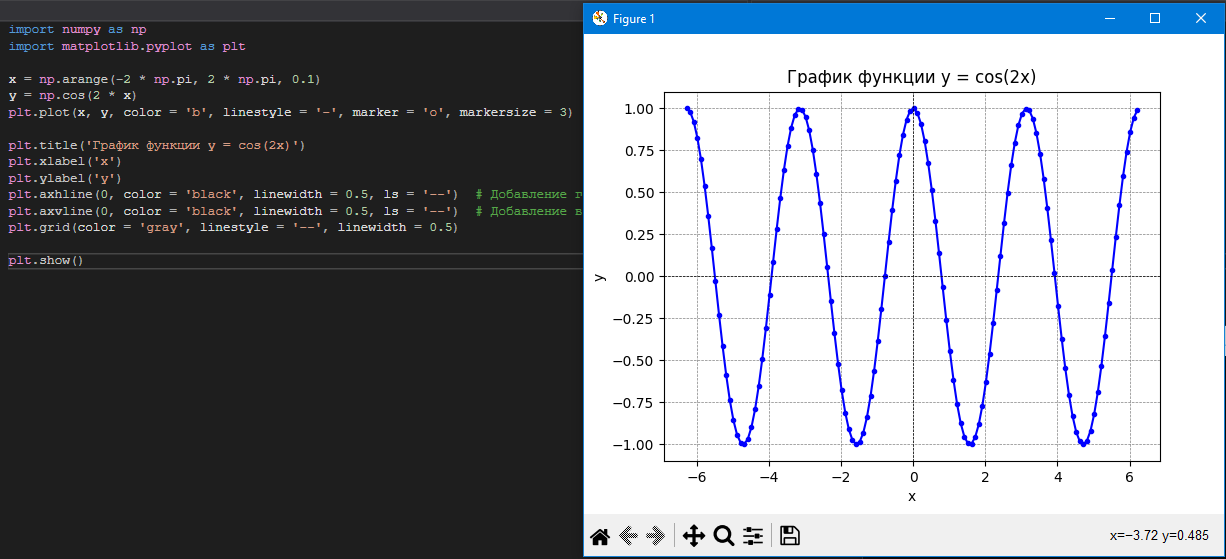
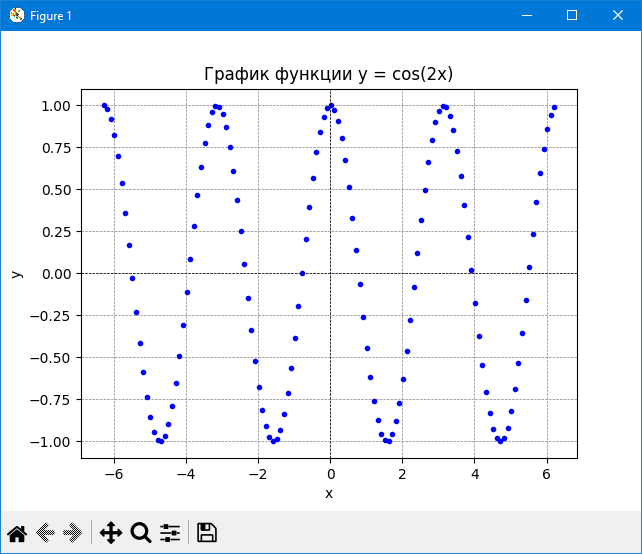
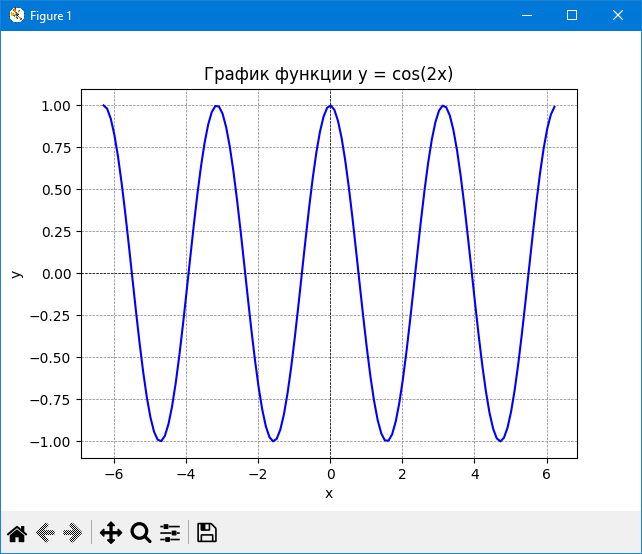


Рисунок 1.1 – Пример использования функции plot(x, y,…) для построения графика.

Рисунок 1.2 – Пример использования функции plot(x, y,…) для построения графика.

2) Использование функции pie() для построения круговых диаграмм.

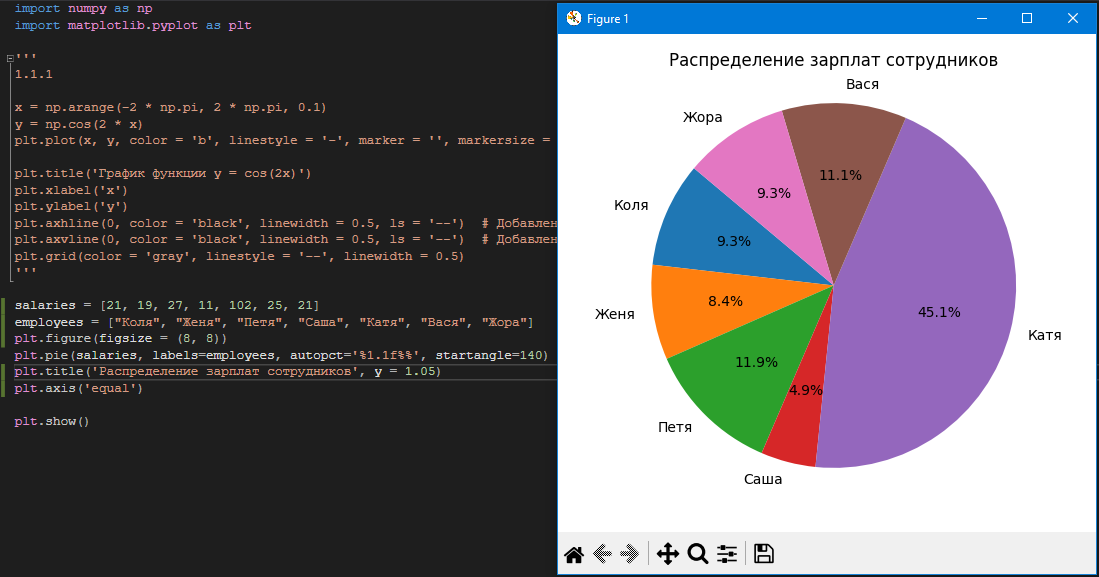


Рисунок 2 – Пример построения круговой диаграммы.

3) Разбиение графического окна с использованием функции par().

import numpy as np

import matplotlib.pyplot as plt

x = np.arange(-np.pi, np.pi, 0.1)

y1 = np.sin(2 \* x)

y2 = np.sin(x)

fig, axs = plt.subplots(2, 1, figsize=(8, 8)) # 2 строки, 1 столбец

# Построение первого графика

axs[0].plot(x, y1, color='brown', label='sin(2x)')

axs[0].set\_title('График функции y = sin(2x)')

axs[0].set\_xlabel('x')

axs[0].set\_ylabel('y')

axs[0].grid()

axs[0].legend()

# Построение второго графика

axs[1].plot(x, y2, color='blue', label='sin(x)')

axs[1].set\_title('График функции y = sin(x)')

axs[1].set\_xlabel('x')

axs[1].set\_ylabel('y')

axs[1].grid()

axs[1].legend()

plt.tight\_layout()

plt.show()

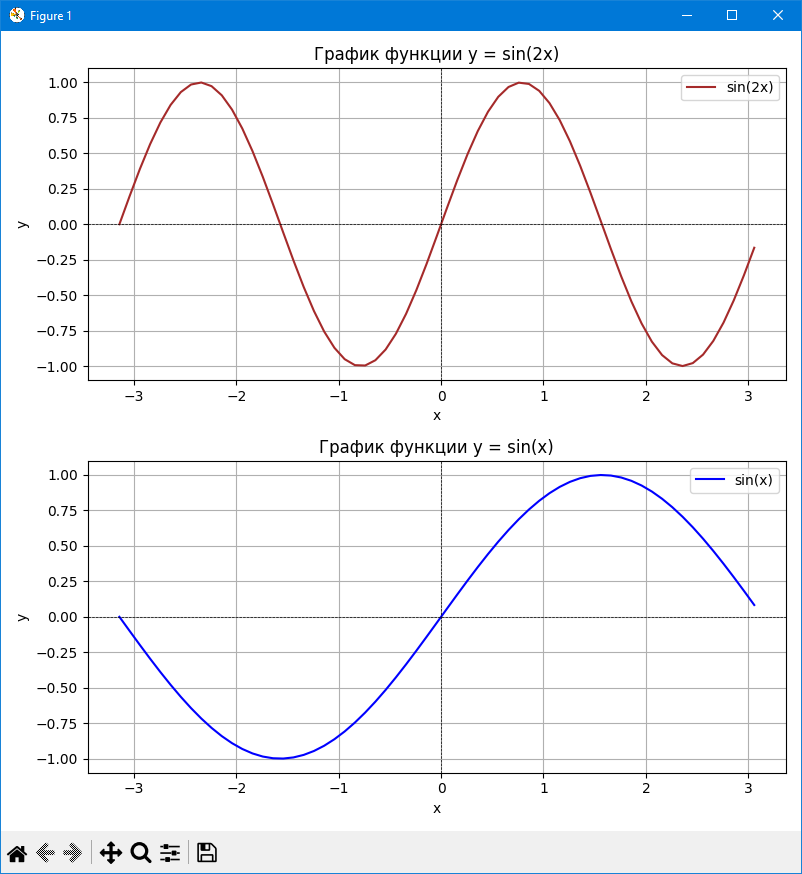


Рисунок 3 – Пример позиционирования нескольких графиков в одном окне.

4) Построение столбчатых диаграмм.

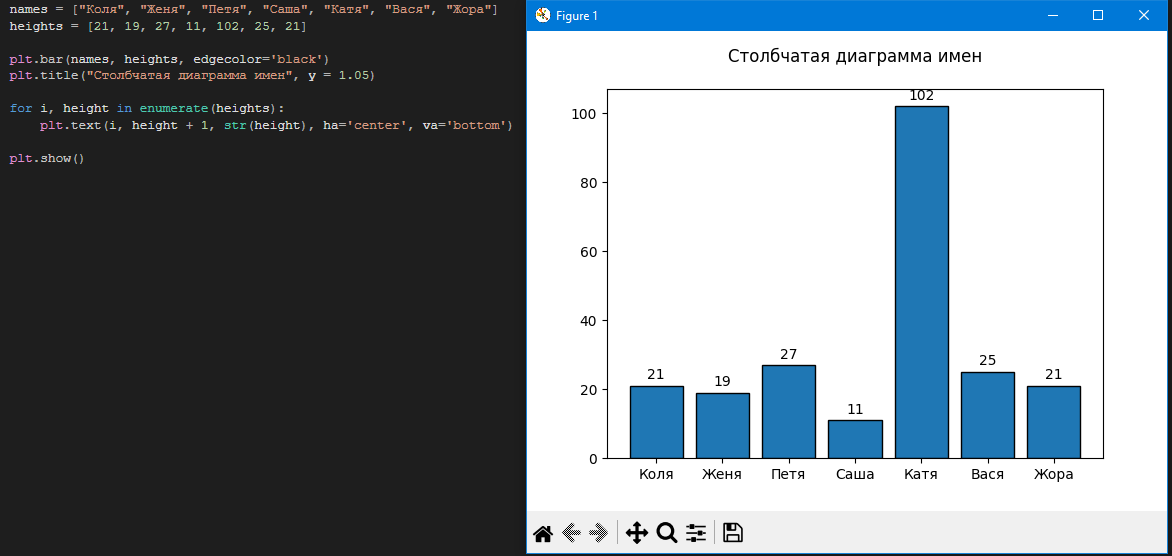


Рисунок 4 – Пример построения столбчатой диаграммы без настройки подписей к осям.

5) Построение диаграмм при помощи функции barplot() с использованием аргументов а xlab, ylab, main, col.main и font.main.

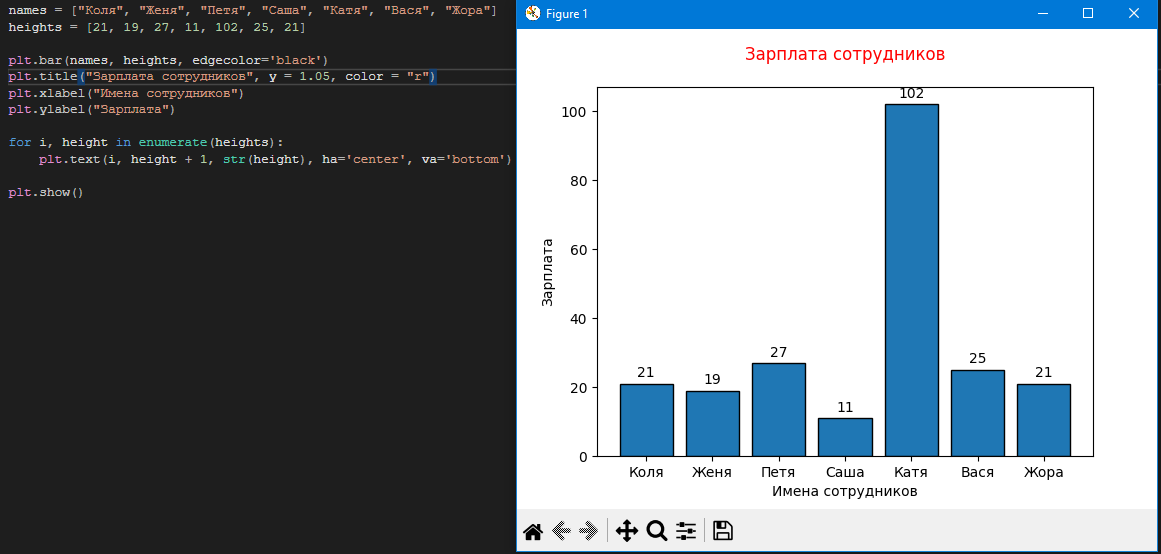


Рисунок 5 – Пример построения столбчатой диаграммы с настройкой подписей.

6) Построение диаграмм при помощи функции hist().

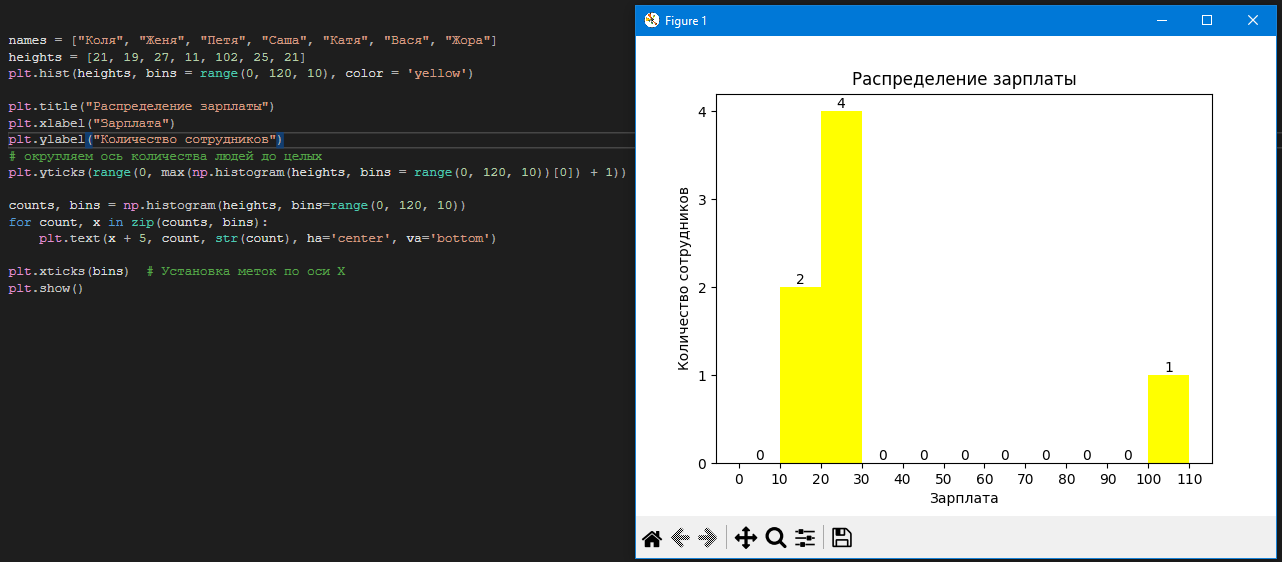


Рисунок 6 – Пример построения гистограммы.

7) Построение трехмерного графика при помощи функции persp().

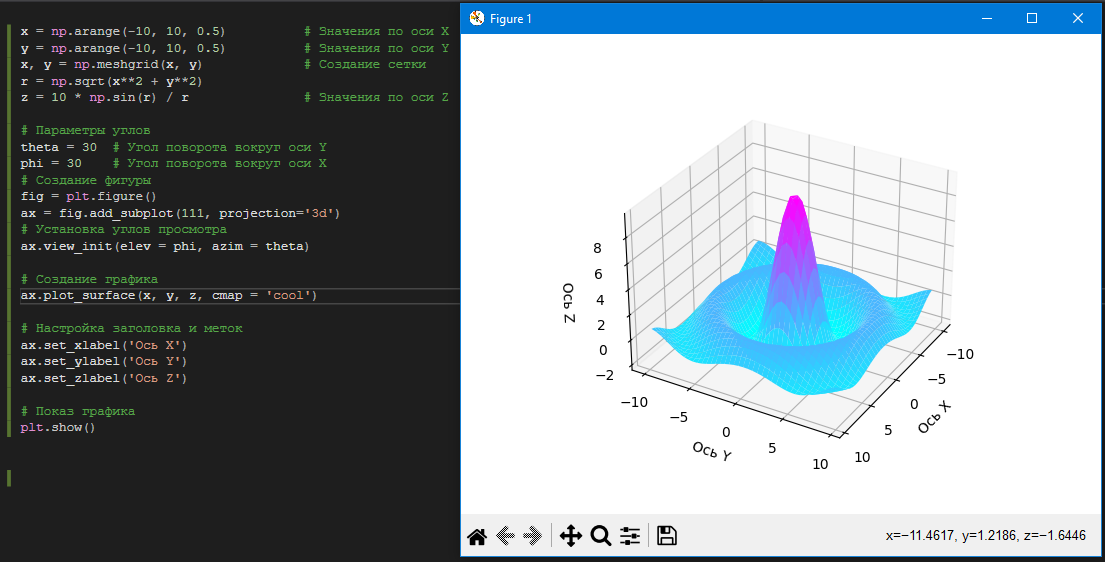


Рисунок 7 – Пример построения трёхмерного графика.

В matplotlib для настройки угла поворота по оси Z можно использовать метод view\_init() для управления углами обзора в трехмерном графике. Однако для конкретного вращения графика вокруг оси Z нет прямого метода, как для theta и phi, но можно изменить представление графика с помощью трансформации координат или настройки угла обзора:

# Создание графика

surf = ax.plot\_surface(x, y, z, cmap = 'cool')

# Функция обновления для анимации

def update(frame):

ax.view\_init(elev=30, azim=frame) # Вращение вокруг оси Z

return surf,

# Создание анимации

ani = animation.FuncAnimation(fig, update, frames=np.arange(0, 360, 1), interval=50)

8) Разбиение графического окна на части командой split.screen().

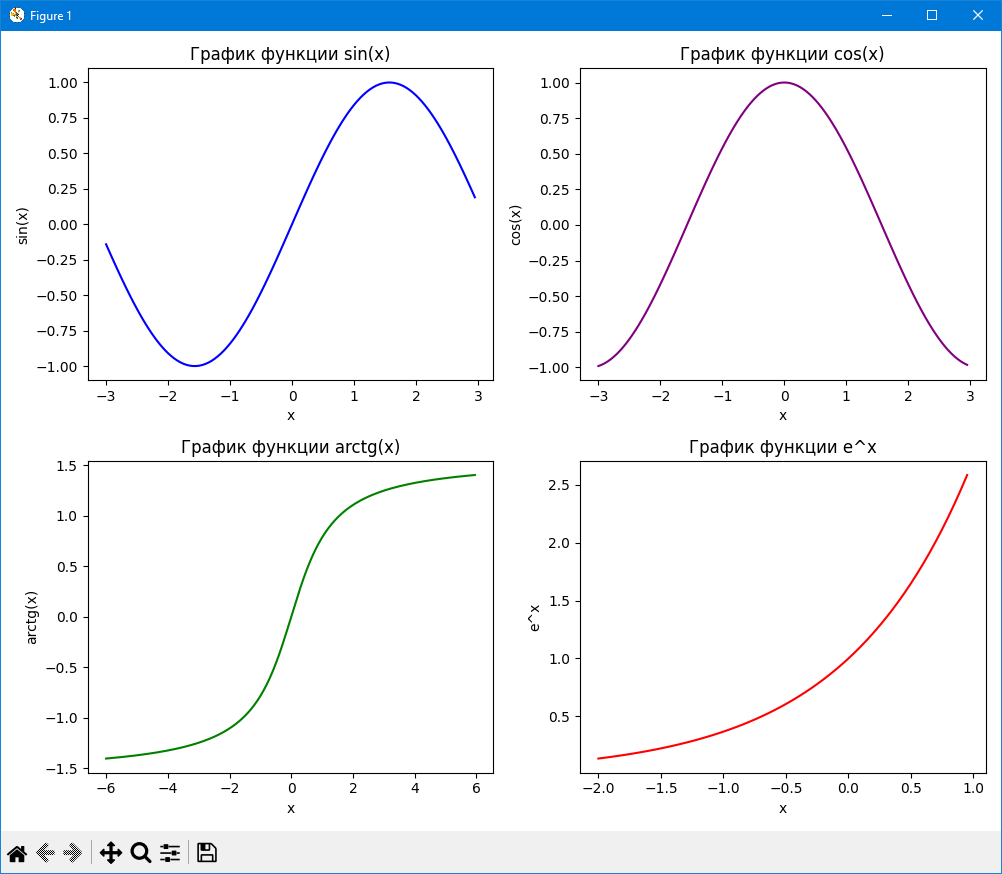


Рисунок 8 – Пример построения нескольких окон графиков на одном фрейме.

9) Построение нескольких функций графиков в одном графическом окне.

# 1. Определение значений переменной x

x = np.array([-1, 0, 1, 1.5, 2])

# 2. Определение значений переменной y

y = np.array([0.8, 0.1, 1.05, 2.3, 3.8])

# 3. Вычисление значений функции y = x^2

y\_function = x\*\*2

# Создание графического окна

plt.figure(figsize=(10, 6))

# 4. Построение графика экспериментальных данных

plt.scatter(x, y, color = 'red', label = 'Экспериментальные данные', linewidth = 2) # Кружки

# 5. Разрешение построения в графическом окне еще одного графика (в matplotlib это делается автоматически)

# 6. Построение графика функции y = x^2

plt.plot(x, y\_function, color = 'blue', label = 'y = x^2', linewidth = 2) # Сплошная линия

# Настройка заголовка и меток

plt.title('График экспериментальных данных и функции y = x^2')

plt.xlabel('x')

plt.ylabel('y')

plt.axhline(0, color='black',linewidth=0.5, ls='--') # Горизонтальная линия на уровне y=0

plt.axvline(0, color='black',linewidth=0.5, ls='--') # Вертикальная линия на уровне x=0

plt.grid(color = 'gray', linestyle = '--', linewidth = 0.5)

plt.legend()

plt.tight\_layout()

plt.show()

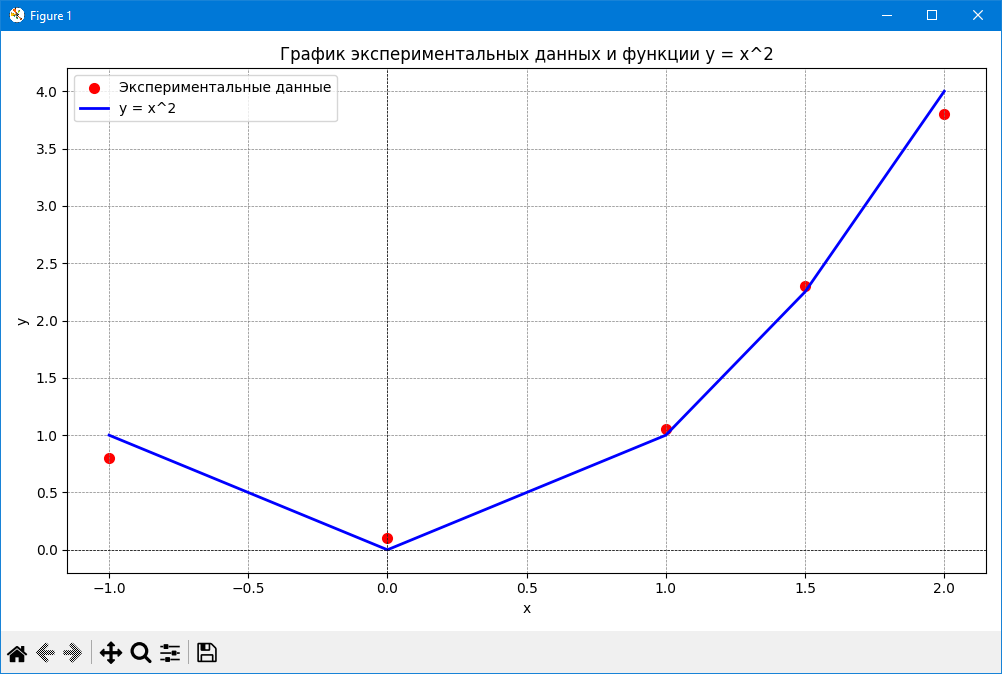


Рисунок 9.1 – Пример построения нескольких графиков в одном окне.

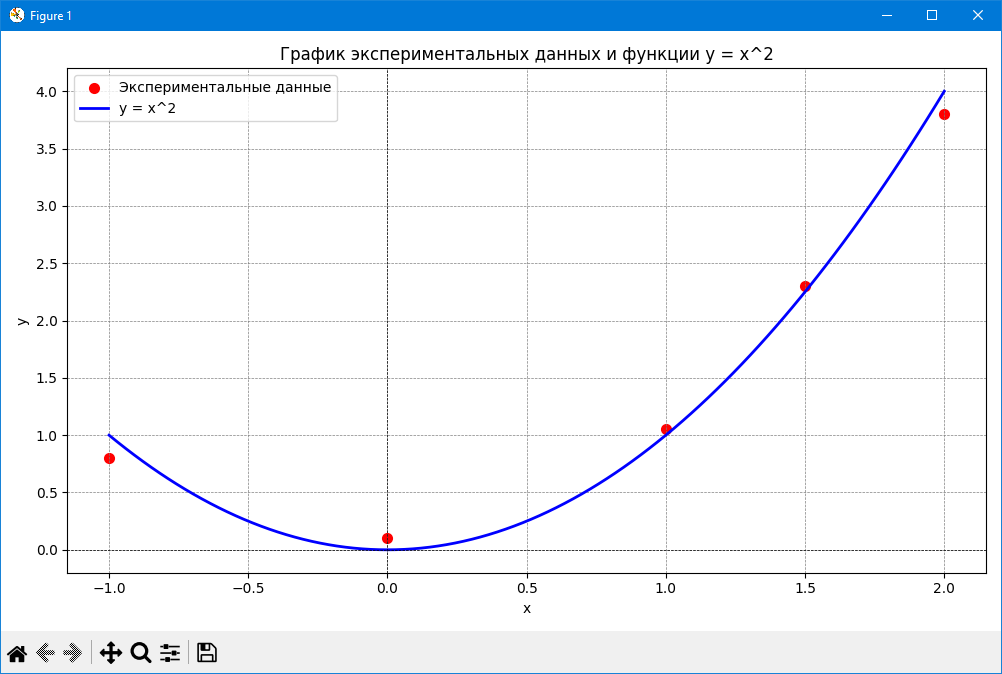


Рисунок 9.2 – Пример построения нескольких графиков в одном окне.

10) Построение графики с помощью пакета ggplot2.

Вывод

В ходе выполнения данной лабораторной работы были исследованы возможности языка R для визуализации данных в R, а также возможности визуализации данных в Python.

Контрольные вопросы

1. Особенности языка R.

– эффективная обработка данных и простые средства для сохранения результатов;

– набор операторов для обработки массивов, матриц, и других сложных конструкций;

– большая, последовательная, интегрированная коллекция инструментальных средств

для проведения статистического анализа,

– многочисленные графические средства;

– простой и эффективный язык программирования, который включает много

возможностей.

2. Команда для получения подробной информации о функции в R.

Команда 'help()' используется для получения справки.

Также можно использовать синтаксис '? '.

3. Структура и особенности команды round() в R.

round(*<округляемое значение>*, *[<точность>]*)

Команды round() имеет два аргумента: число, которое нужно округлить, и значение digits, сообщающее, до какого знака округлять. По умолчанию digits = 0.

4. Команды для работы с векторами в R (изучить команды, не представленные в методических указаниях).

с() – создаёт вектор, объединяя несколько однотипных элементов;

а1:а2 – задаёт регулярную последовательность значений от а1 до а2;

seq(а1, а2) – задаёт регулярную последовательность значений от а1 до а2;

rep(а1, а2) – повторяет последовательность а1 количество раз, равное а2;

crossprod(v1, v2) – вычисляет скалярное произведение подобных векторов;

v[a] – возвращает а-й элемент вектора;

v[a > 0] – возвращает все элементы вектора, удовлетворяющие условию;

length(v1) – возвращает длину вектора v1;

mean(v1) – возвращает среднее значение элементов вектора v1;

var(v1) – возвращает дисперсию элементов вектора v1;

Для работы с векторами доступны все простые арифметические команды.

5. Команды для работы с матрицами в R (изучить команды, не представленные в методических указаниях).

matrix*(<последовательность или значение, которым будет заполнена матрица>, <число>, < число >, [<TRUE/FALSE>], [<NULL / список, длиной больше 2>]*), по умолчанию data = NA, nrow = 1, ncol = 1, byrow = FALSE, dimnames = NULL;

cbind(<с1>, [с2, …]) – создаёт матрицу, путём объединения строк с1, с2, …;

rbind(<r1>, [r2, …]) – создаёт матрицу, путём объединения столбцов r1, r2, …;

dim(m) – возвращает размерность матрицы m;

6. Работа с графикой в R (изучить команды, не представленные в методических указаниях).

Основной функцией для рисования объектов в R является функция plot(x, y, …):

x – х-координаты точек графика, либо некоторая графическая структура, функция или объект, содержащий методы рисования.

𝑦 – 𝑦-координаты точек графика, если 𝑥 – соответствующего типа.

– остальные графические параметры. Перечислим некоторые из них:

– параметр type позволяет изменять внешний вид точек на графике и может принимать

одно из следующих значений:

– "p" – точки (points; используется по умолчанию);

– "l" – линии (lines);

– "b" – изображаются и точки, и линии (both points and lines);

– "o" – точки изображаются поверх линий (points over lines);

– "h" – гистограмма (histogram);

– "s" – ступенчатая кривая (steps);

– "n" – данные не отображаются (no points).

– параметры xlab и ylab задают название осей абсцисс и ординат, соответственно;

– параметр main задаёт заголовок графика.