Московский авиационный институт
(Национальный исследовательский университет)

Институт: «Информационные технологии и прикладная математика»
Кафедра: 806 «Вычислительная математика и программирование»
Дисциплина: «Компьютерная графика»

Лабораторная работа № 1

Тема: Построение изображений 2D-кривых

Студент: Дубровин Дмитрий
Группа: М8О-307Б-21
Преподаватель: Чернышов Л.Н.
Дата:
Оценка:

Москва, 2023

1. **Постановка задачи**

Написать и отладить программу, строящую изображение заданной замечательной кривой.

Вариант №16:

$$16.\ y^2 = x^2(a-x)/(a+x)\ ,\ -a<A<=x<=B<a$$

2. **Описание программы**

Для выполнения поставленной задачи было принято решение использовать язык программирования Python и его модули. Этот код создаёт графический пользовательский интерфейс (GUI) с помощью библиотеки Tkinter в Python, который позволяет пользователю вводить значения параметров (a), (A), и (B), и отображает соответствующий график функции, заданной как:

$$16.\ y^2 = x^2(a-x)/(a+x)\ ,\ -a<A<=x<=B<a$$

Описание Элементов:

1. Tkinter Widgets:
   - Labels: Для отображения текстовых меток "Введите a (a > 0):", "Введите A (A > -a):", "Введите B (B < a):".
   - Entry Widgets: Для ввода значений параметров (a), (A), и (B).
   - Buttons: Кнопки "Отрисовать", "Уменьшить" и "Увеличить" для отрисовки графика и изменения масштаба.

2. Matplotlib:
   - Используется для создания графика на основе введенных значений и отображения его в пользовательском интерфейсе.

3. Глобальные Переменные:
   - `zoom_factor:` Используется для масштабирования графика при нажатии кнопок увеличенияуменьшения масштаба.

4. Функции:
   - `on_draw():` Отрисовывает график на основе текущих значений параметров и фактора масштабирования.
   - `zoom_in():` Увеличивает масштаб графика на 10% и перерисовывает

его.

- `zoom_out():` Уменьшает масштаб графика на 10% и перерисовывает его.

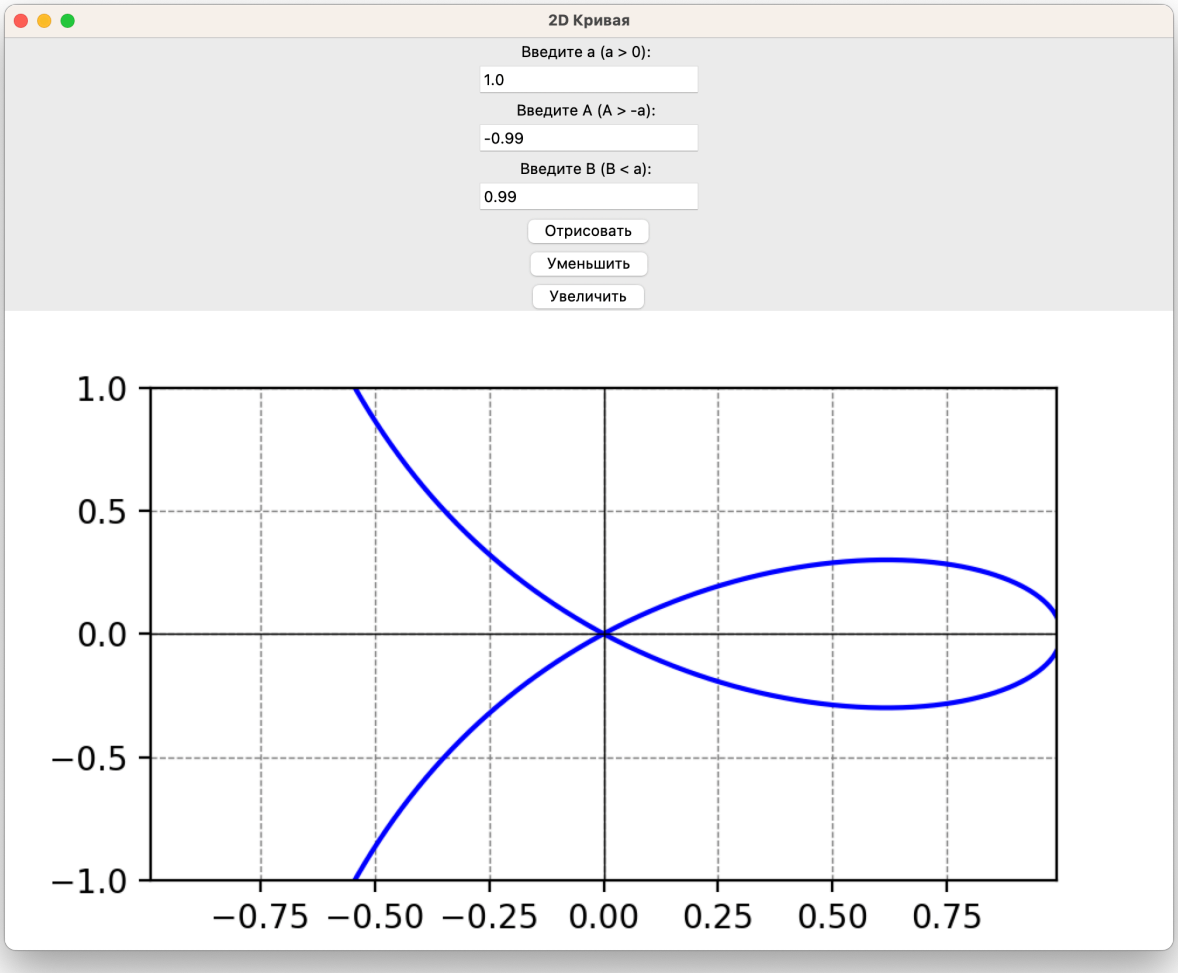
Рабочий Процесс:
- Пользователь вводит значения параметров (a), (A), и (B).
- При нажатии на кнопку "Отрисовать", программа строит график в соответствии с введенными значениями.
- Пользователь может изменять масштаб графика с использованием кнопок "Увеличить" и "Уменьшить".
- Если введены некорректные значения, программа выводит сообщение об ошибке в консоль.

3. Результаты выполнения тестов

a =1.0

A = -0.99

B = 0.99



4. Листинг программы

```
import tkinter as tk
from tkinter import DoubleVar
import matplotlib.pyplot as plt
from matplotlib.backends.backend_tkagg import FigureCanvasTkAgg
import numpy as np

# Глобальная переменная для фактора масштабирования
zoom_factor = 1.0

def on_draw():
    global zoom_factor

    # Считываем значение константы a, A и B
    a = a_var.get()
    A = A_var.get()
    B = B_var.get()

    # Проверяем, что значения A, B и a корректны
    if A >= a or B <= -a or B <= A:
        print("Некорректные значения A, B или a")
        return

    # Создаем массив x в пределах от A до B
    x = np.linspace(A, B, 1000)

    # Вычисляем y
    y_squared = x**2 * ((a - x) / (a + x))  # Квадрат y
        y_squared = np.maximum(y_squared, 0)    # Убедимся, что значения
неотрицательны
    y = np.sqrt(y_squared)  # Вычислим корень для получения y

    # Обновляем график с учетом масштабирования
    ax.cla()

    ax.plot(x, y, 'b')
    ax.plot(x, -y, 'b')
    ax.axhline(0, color='black', linewidth=0.5)
```

```
    ax.axvline(0, color='black', linewidth=0.5)
    ax.grid(color='gray', linestyle='--', linewidth=0.5)

    # Устанавливаем новые пределы осей
    ax.set_xlim(A * zoom_factor, B * zoom_factor)
    ax.set_ylim(-a * zoom_factor, a * zoom_factor)

    canvas.draw()


def zoom_in():
    global zoom_factor
    zoom_factor *= 1.1  # Увеличиваем фактор масштабирования на 10%
    on_draw()  # Перерисовываем график после зума


def zoom_out():
    global zoom_factor
    zoom_factor /= 1.1  # Уменьшаем фактор масштабирования на 10%
    on_draw()  # Перерисовываем график после зума


root = tk.Tk()
root.title("2D Кривая")

# Создаем tkinter переменные для констант a, A и B
a_var = DoubleVar(value=1.0)
A_var = DoubleVar(value=-0.99)
B_var = DoubleVar(value=0.99)

# Создаем поля для ввода констант a, A и B
a_label = tk.Label(root, text="Введите a (a > 0): ")
a_label.pack()
a_entry = tk.Entry(root, textvariable=a_var)
a_entry.pack()

A_label = tk.Label(root, text="Введите A (A > -a): ")
A_label.pack()
```

```
A_entry = tk.Entry(root, textvariable=A_var)
A_entry.pack()

B_label = tk.Label(root, text="Введите B (B < a): ")
B_label.pack()
B_entry = tk.Entry(root, textvariable=B_var)
B_entry.pack()

# Создаем кнопки для отрисовки и управления масштабом
draw_button = tk.Button(root, text="Отрисовать", command=on_draw)
draw_button.pack()

zoom_in_button = tk.Button(root, text="Уменьшить", command=zoom_in)
zoom_in_button.pack()

zoom_out_button = tk.Button(root, text="Увеличить", command=zoom_out)
zoom_out_button.pack()

# Настраиваем фигуру и область рисования
fig, ax = plt.subplots(figsize=(5, 5))
canvas = FigureCanvasTkAgg(fig, master=root)
canvas.get_tk_widget().pack()

# Запускаем главный цикл tkinter
root.mainloop()
```

5. Вывод

В ходе данной лабораторной работы я изучил несколько полезных библиотек Python, с которыми только пересекался ранее. Лабораторная работа была успешно выполнена, все поставленные задачи были решены, а цели достигнуты. Программа демонстрирует корректную работу и предоставляет пользователям возможность визуализации графика на основе введенных параметров, а также проведения анализа графика путем изменения масштаба. Эта работа демонстрирует, как можно сочетать графический интерфейс и визуализацию данных для создания полезных и эффективных инструментов анализа.

Литература

Numpy documentation [Электронный ресурс] URL: https://numpy.org
Matplotlib tutorials [Электронный ресурс] URL: https://matplotlib.org
Tkinter documentation [Электронный ресурс] URL: https://docs.python.org/3/library/tkinter.html