МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

**«КУБАНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»**

**(ФГБОУ ВО «КУБГУ»)**

**Факультет компьютерных технологий и прикладной математики**

**Кафедра вычислительных технологий**

**Отчет**

**по лабораторной работе №5 по курсу**

**«МЕТОДЫ ПОИСКОВОЙ ОПТИМИЗАЦИИ»**

Работу выполнили

Студенты 46 группы

Стасюк К.В.

Баева Д.Н.

Преподаватель:

Нигодин Е.А.

Краснодар 2023

**Цель работы:** разработать B-алгоритм минимизации функций Розенброка, Химмельблау, Растригина.

**Ход работы:**

Пчелиным алгоритмом называют (В-алгоритм) *Bees algorithm*, предложенный в работах Фама (D. Т. Pham) с соавторами в 2005 г. Концептуально, схема В-алгоритма имеет следующий вид. Сначала из улья вылетает в случайном направлении некоторое число пчел-разведчиков, которые пытаются отыскать участки, где есть нектар. Через какое-то время пчелы возвращаются в улей и сообщают другим пчелам, где и сколько они нашли нектара. После этого на найденные участки отправляются рекрутированные (рабочие) пчелы, причем чем больше на данном участке предполагается найти нектара, тем больше пчел летит к этому участку. Пчелы-разведчики опять улетают искать другие участки, после чего процесс повторяется.

Пусть дана задача минимизации фитнес-функции. Количество нектара в некоторой точке пространства поиска полагаем пропорциональным значению фитнес-функции в этой точке. Одна итерация алгоритма включает в себя следующие основные шаги.

1. В случайные точки пространства поиска отправляем пчел-разведчиков.

2. На основании значений фитнес-функции, вычисленных в указанных точках, выделяем некоторое число элитных (лучших) участков - подобластей пространства поиска, соответствующих минимальным значениям этой функции. Аналогично определяем некоторое число перспективных участков, которые соответствуют значениям фитнес-функции, близким к минимальным.

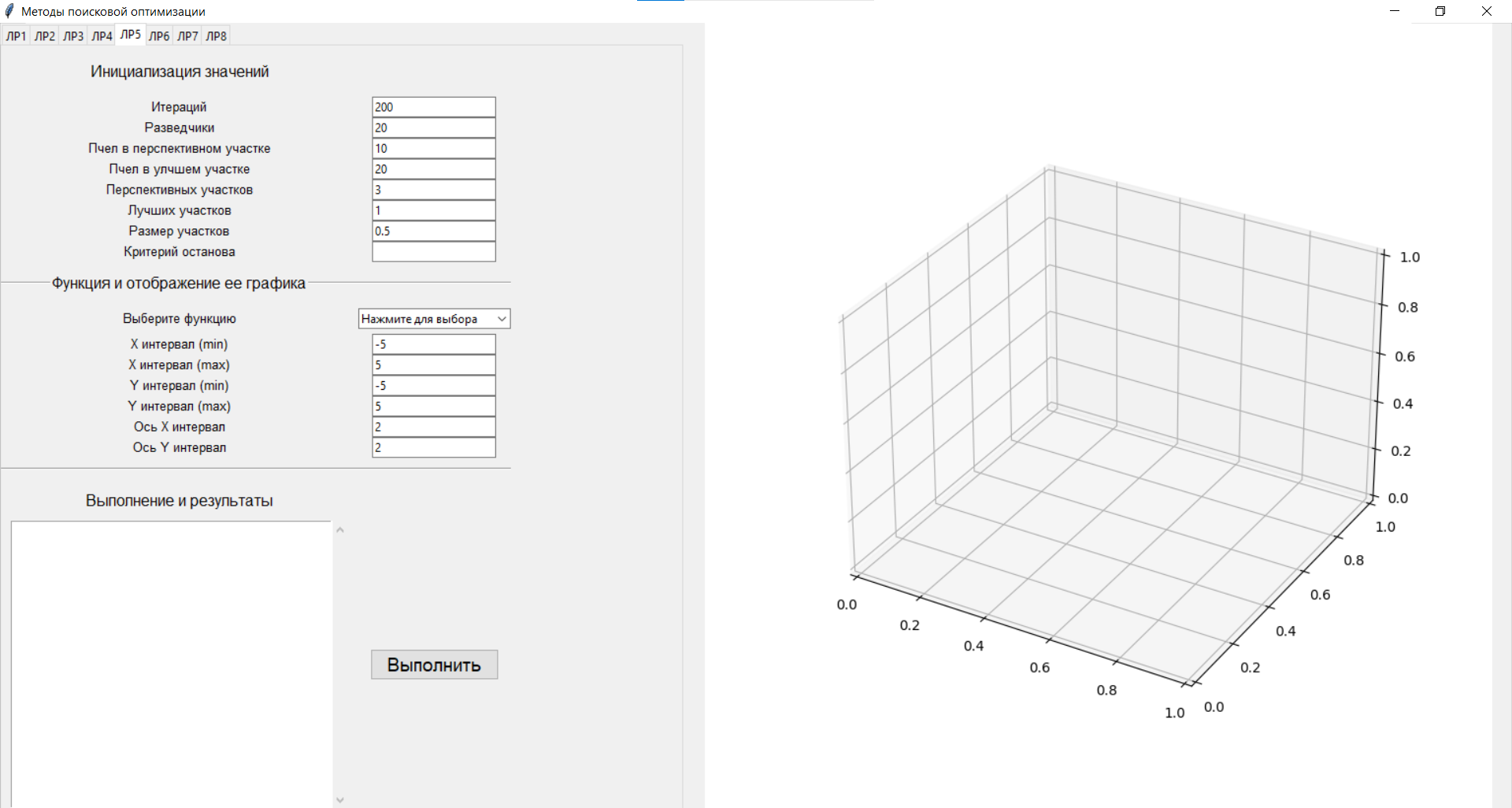
3. На каждый из элитных и перспективных участков посылаем определенное число рабочих пчел. Используя соответствующие значения фитнес-функции, находим новые элитные и перспективные участки. При этом выборе учитываем результаты, полученные как пчелами-разведчиками, так и рабочими пчелами.

В качестве текущего приближения к решению задачи принимаем точку с минимальным значением фитнес-функции. Размеры элитных и перспективных участков уменьшаем с ростом числа итераций, так что на завершающих итерациях поиск ведется только в окрестностях минимумов целевой функции.

**Особенности реализации генетического алгоритма.**

Для создания программы использовался язык программирования Python 3.10 и среда разработки PyCharm Professional 2023.2. Для графической визуализации были подключены графические фреймворки Tkinter и Matplotlib.

Интерфейс программы имеет вид:

Рисунок 1 – Главное окно программы.

В данном окне можно выбрать необходимую вкладку соответствующей лабораторной работы, в данном случае «ЛР5», ввести количество итераций, количество пчёл-разведчиков, пчёл в перспективном участке, пчёл в лучшем участке, количество перспективных участков, количество лучших участков, размер участков и критерий останова. В поле «Выполнение и результаты» динамически выводятся результаты работы алгоритма в виде шагов, представленных координатами и значениями оптимизируемой функции в этих координатах. На рисунке 2 показано поле «Выполнение и результаты» до запуска программы.

Изображение выглядит как текст, снимок экрана, Прямоугольник

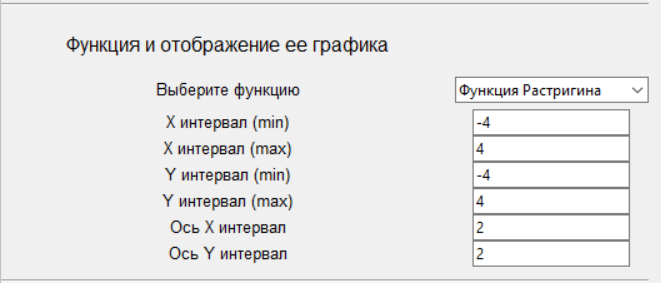
Автоматически созданное описание

Рисунок 2 – Выполнение и результаты.

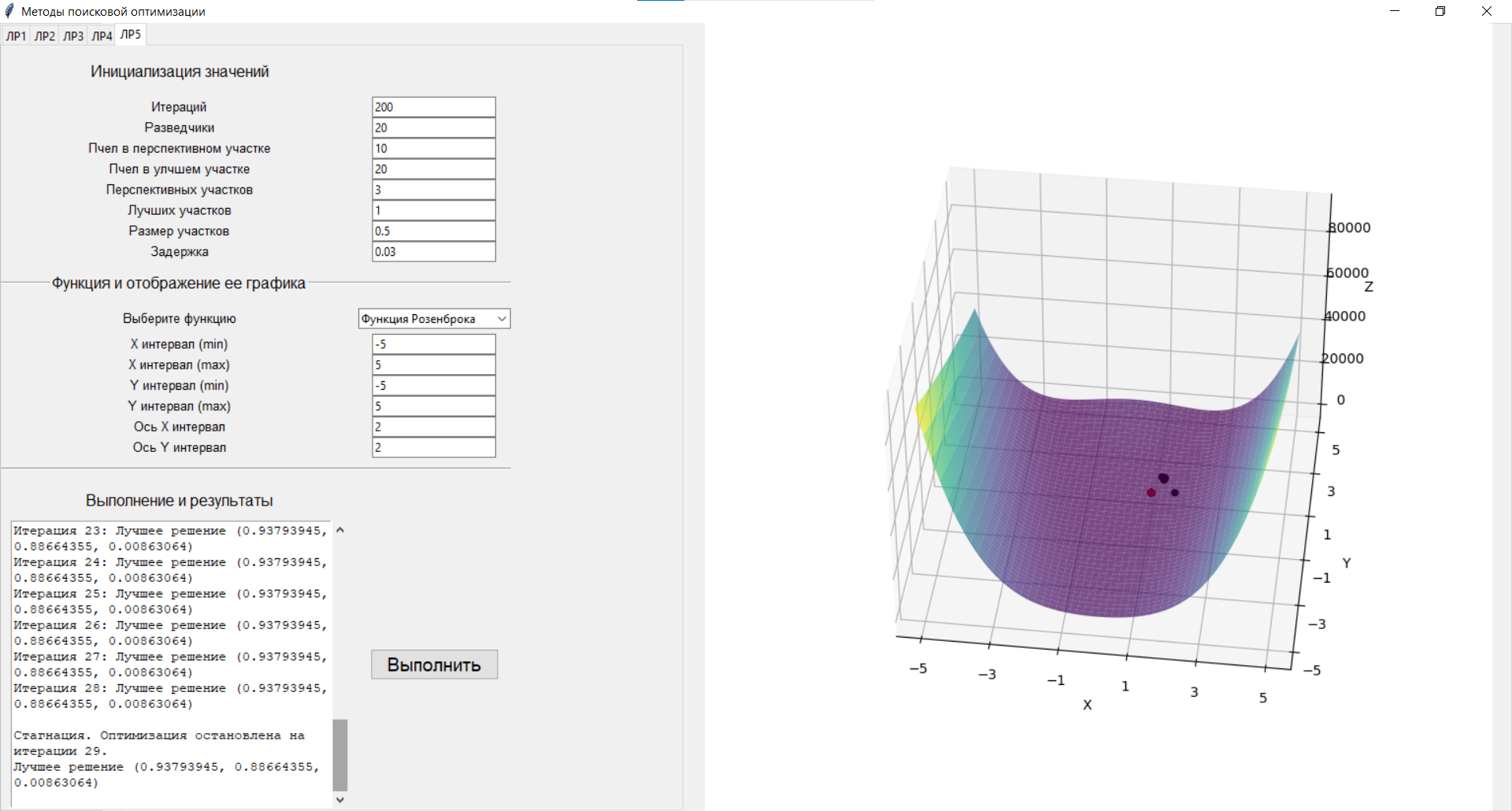
Также, в интерфейсе программы есть панель управления «Функция и отображение её графика», в котором можно выбрать необходимую функцию и задать её параметры:

1. X интервал (интервал на котором строится и минимизируется функция);
2. Y интервал (интервал на котором строится и минимизируется функция);
3. Ось X интервал (размер отображаемой сетки по оси X);
4. Ось Y интервал (размер отображаемой сетки по оси Y).

На рисунке 3 показана панель «Функция и отображение её графика»

Рисунок 3 – Функция и отображение её графика.

Для запуска алгоритма необходимо нажать кнопку «Выполнить», при этом поле «Выполнение и результаты» начнёт динамически заполняться точками, а на отображаемой функции можно наглядно увидеть функционирование алгоритма - искомые точки, которые также отображаются динамически как показано на рисунке 4. Результирующая точка выделена на графике крестом. Значения результирующей функции имеют разный окрас в зависимости от положения: самые высокие точки показаны ярко-жёлтым цветом, а самые низкие – тёмно-синим.

Рисунок 4 – Результат работы программы.

**Вывод:** в ходе работы были изучен пчелиный алгоритм (B-алгоритм), а затем разработан B-алгоритм минимизации функций Розенброка, Химмельблау, Растригина.

**Листинг программы**

Файл LR5.py

import tkinter as tk

from tkinter import ttk

from tkinter import scrolledtext

import numpy as np

import math

from LR5.BeeAlgorithm import BeeAlgorithm

def BeesAlgorithm(frame,root,ax,canvas):

# Функция Розенброка для оптимизации

def himel\_function(x\_arr):

x, y = x\_arr[0], x\_arr[1]

return (x \*\* 2 + y - 11) \*\* 2 + (x + y \*\* 2 - 7) \*\* 2

def rosenbrock\_function(x\_arr):

x, y = x\_arr[0], x\_arr[1]

return (1 - x) \*\* 2 + 100 \* (y - x \*\* 2) \*\* 2

def rastrigin(x\_arr):

size = len(x\_arr)

return 10 \* size + np.sum(x\_arr \*\* 2 - 10 \* np.cos(2 \* np.pi \* x\_arr))

def run\_optimization():

ax.cla()

function\_choice = function\_var.get()

target\_func = himel\_function

if function\_choice == "Функция Химмельблау":

target\_func = himel\_function

elif function\_choice == "Функция Розенброка":

target\_func = rosenbrock\_function

elif function\_choice == "Функция Растригина":

target\_func = rastrigin

iterations=int(iteration.get())

scout = int(scouts.get()) # разведчики

perspective\_B = int(perspective\_b.get())

best\_B = int(best\_b.get()) # лучшие пчелы

perspective\_A = int(perspective\_a.get())

best\_A = int(best\_a.get())

size\_A = int(size\_a.get())

bound\_start = float(x\_interval\_min.get())

bound\_end = float(x\_interval\_max.get())

bounds = [(bound\_start, bound\_end) for i in range(2)]

x\_range = np.linspace(bound\_start, bound\_end, 100)

y\_range = np.linspace(bound\_start, bound\_end, 100)

X, Y = np.meshgrid(x\_range, y\_range)

Z = np.zeros\_like(X)

for i in range(X.shape[0]):

for j in range(X.shape[1]):

Z[i, j] = target\_func(np.array([X[i, j], Y[i, j]]))

ax.cla()

canvas.draw()

ax.plot\_surface(X, Y, Z, cmap='viridis', alpha=0.7)

ax.set\_xlabel('X')

ax.set\_ylabel('Y')

ax.set\_zlabel('Z')

ax.set\_xticks(np.arange(bound\_start, bound\_end + 1, 2))

ax.set\_yticks(np.arange(bound\_start, bound\_end + 1, 2))

results\_text.config(state=tk.NORMAL)

results\_text.delete(1.0, tk.END)

algorithm = BeeAlgorithm(scout, size\_A, size\_A, best\_A, perspective\_A,

perspective\_B, best\_B, bounds, iterations, 20,

target\_func)

algorithm.set\_options(root, ax, canvas, results\_text)

best\_bee = algorithm.optimize()

ax.scatter(best\_bee.coords[0], best\_bee.coords[1], best\_bee.fitness, c="red")

results\_text.insert(tk.END,

f"Лучшее решение ({best\_bee.coords[0]:.8f}, {best\_bee.coords[1]:.8f}, {best\_bee.fitness:.8f})\n")

canvas.draw()

root.update()

param\_frame2 = frame

# Параметры задачи

ttk.Label(param\_frame2, text="Инициализация значений", font=("Helvetica", 12)).grid(row=0, column=0, pady=15)

ttk.Label(param\_frame2, text="Итераций", font=("Helvetica", 10)).grid(row=1, column=0)

ttk.Label(param\_frame2, text="Разведчики", font=("Helvetica", 10)).grid(row=2, column=0)

ttk.Label(param\_frame2, text="Пчел в перспективном участке", font=("Helvetica", 10)).grid(row=3, column=0)

ttk.Label(param\_frame2, text="Пчел в улчшем участке", font=("Helvetica", 10)).grid(row=4, column=0)

ttk.Label(param\_frame2, text="Перспективных участков", font=("Helvetica", 10)).grid(row=5, column=0)

ttk.Label(param\_frame2, text="Лучших участков", font=("Helvetica", 10)).grid(row=6, column=0)

ttk.Label(param\_frame2, text="Размер участков", font=("Helvetica", 10)).grid(row=7, column=0)

ttk.Label(param\_frame2, text="Критерий останова", font=("Helvetica", 10)).grid(row=8, column=0)

iteration = tk.IntVar(value=200)

scouts = tk.IntVar(value=20) #разведчики

perspective\_b = tk.IntVar(value=10) #перспективных пчел

best\_b = tk.IntVar(value=20) #лучшие пчелы

perspective\_a = tk.IntVar(value=3) #перпективных участков

best\_a = tk.IntVar(value=1) # лучших участков

size\_a = tk.DoubleVar(value=0.5) # размер участков

stop = tk.DoubleVar(value=20) # задержка

iteration\_entry = ttk.Entry(param\_frame2, textvariable=iteration)

scouts\_entry = ttk.Entry(param\_frame2, textvariable=scouts)

perspective\_b\_entry = ttk.Entry(param\_frame2, textvariable=perspective\_b)

best\_b\_entry = ttk.Entry(param\_frame2, textvariable=best\_b)

perspective\_a\_entry = ttk.Entry(param\_frame2, textvariable=perspective\_a)

best\_a\_entry = ttk.Entry(param\_frame2, textvariable=best\_a)

size\_a\_entry = ttk.Entry(param\_frame2, textvariable=size\_a)

stop\_entry = ttk.Entry(param\_frame2, textvariable=stop)

iteration\_entry.grid(row=1, column=1)

scouts\_entry.grid(row=2, column=1)

perspective\_b\_entry.grid(row=3, column=1)

best\_b\_entry.grid(row=4, column=1)

perspective\_a\_entry.grid(row=5, column=1)

best\_a\_entry.grid(row=6, column=1)

size\_a\_entry.grid(row=7, column=1)

stop\_entry.grid(row=8, column=1)

separator = ttk.Separator(param\_frame2, orient="horizontal") # Горизонтальная полоса разделения

separator.grid(row=9, column=0, columnspan=2, sticky="ew", pady=10)

# Параметры функции

ttk.Label(param\_frame2, text="Функция и отображение ее графика", font=("Helvetica", 12)).grid(row=9, column=0, pady=10)

ttk.Label(param\_frame2, text="Выберите функцию", font=("Helvetica", 10)).grid(row=10, column=0)

function\_choices = ["Нажмите для выбора","Функция Химмельблау", "Функция Розенброка",

"Функция Растригина"]

function\_var = tk.StringVar(value=function\_choices[0])

function\_menu = ttk.Combobox(param\_frame2, textvariable=function\_var, values=function\_choices, width=22)

function\_menu.grid(row=10, column=1, pady=5)

ttk.Label(param\_frame2, text="X интервал (min)", font=("Helvetica", 10)).grid(row=11, column=0)

ttk.Label(param\_frame2, text="X интервал (max)", font=("Helvetica", 10)).grid(row=12, column=0)

ttk.Label(param\_frame2, text="Y интервал (min)", font=("Helvetica", 10)).grid(row=13, column=0)

ttk.Label(param\_frame2, text="Y интервал (max)", font=("Helvetica", 10)).grid(row=14, column=0)

ttk.Label(param\_frame2, text="Ось X интервал", font=("Helvetica", 10)).grid(row=16, column=0)

ttk.Label(param\_frame2, text="Ось Y интервал", font=("Helvetica", 10)).grid(row=17, column=0)

separator = ttk.Separator(param\_frame2, orient="horizontal") # Горизонтальная полоса разделения

separator.grid(row=18, column=0, columnspan=2, sticky="ew", pady=10)

x\_interval\_min = tk.DoubleVar(value=-5)

x\_interval\_max = tk.DoubleVar(value=5)

y\_interval\_min = tk.DoubleVar(value=-5)

y\_interval\_max = tk.DoubleVar(value=5)

x\_axis\_interval = tk.IntVar(value=2)

y\_axis\_interval = tk.IntVar(value=2)

x\_interval\_min\_entry = ttk.Entry(param\_frame2, textvariable=x\_interval\_min)

x\_interval\_max\_entry = ttk.Entry(param\_frame2, textvariable=x\_interval\_max)

y\_interval\_min\_entry = ttk.Entry(param\_frame2, textvariable=y\_interval\_min)

y\_interval\_max\_entry = ttk.Entry(param\_frame2, textvariable=y\_interval\_max)

x\_axis\_interval\_entry = ttk.Entry(param\_frame2, textvariable=x\_axis\_interval)

y\_axis\_interval\_entry = ttk.Entry(param\_frame2, textvariable=y\_axis\_interval)

x\_interval\_min\_entry.grid(row=11, column=1)

x\_interval\_max\_entry.grid(row=12, column=1)

y\_interval\_min\_entry.grid(row=13, column=1)

y\_interval\_max\_entry.grid(row=14, column=1)

x\_axis\_interval\_entry.grid(row=16, column=1)

y\_axis\_interval\_entry.grid(row=17, column=1)

# Создание кнопки Выполнить

button\_style = ttk.Style()

button\_style.configure("My.TButton", font=("Helvetica", 14))

# Создание кнопки Выполнить

apply\_settings\_button = ttk.Button(param\_frame2, text="Выполнить",command=run\_optimization, style="My.TButton")

apply\_settings\_button.grid(row=21, column=1, padx=10, pady=10)

ttk.Label(param\_frame2, text="Выполнение и результаты", font=("Helvetica", 12)).grid(row=20, column=0, pady=10)

results\_text = scrolledtext.ScrolledText(param\_frame2, wrap=tk.WORD, height=18, width=40, padx=2, state=tk.DISABLED)

results\_text.grid(row=21, column=0, padx=10)

root.mainloop()