МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение  
высшего образования   
**«Национальный исследовательский   
Нижегородский государственный университет им. Н.И. Лобачевского»**

**(ННГУ)**

**Институт информационных технологий, математики и механики**

**Кафедра информатики и автоматизации научных исследований**

Направление подготовки: «Прикладная информатика»

Профиль подготовки: «Проектирование и автоматизация производства изделий микроэлектроники»

**ОТЧЕТ**

по лабораторной работе

«Реализация приложения для оптимизации методом Нелдера-Мида»

**Выполнили:** студенты группы 3822Б1ПИмэ1:

Голубев Роман Александрович

Пудков Максим Алексеевич

Самохин Владимир Владимирович

Нижний Новгород  
2025 г.

1. Введение
2. **Наша команда:**

* Голубев Р. - поиск библиотек, CMake (ломаем, потом 3 дня чиним), алгоритм оптимизации, тестирование.
* Пудков М. - frontend, предобработка входящей функции, тестирование.
* Самохин В. - алгоритм вычисления функции в точке, критика вычислительной сложности, исправление багов за всеми остальными, тестирование.

1. **О приложении:**

Мы реализовали приложение под Linux, оптимизирующее функцию, заданную аналитически, с помощью эвристического алгоритма Нелдера-Мида, дополнительно реализована [Web версия (запускается в контейнере)](https://github.com/Golubeff-RA/Nelder-web)

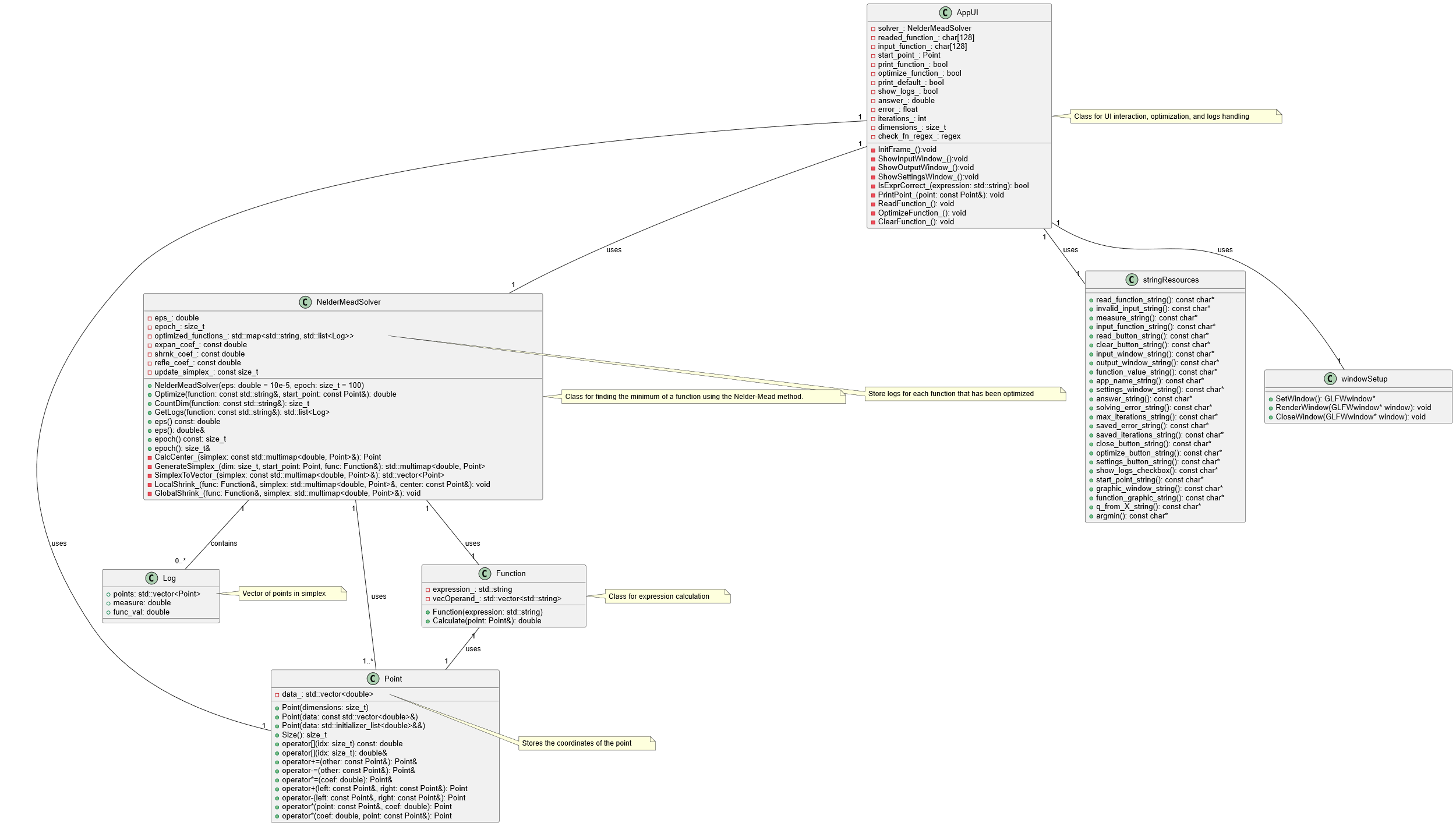
Frontend (UI) нашего приложения написан с помощью открытой библиотеки [ImGUI](https://github.com/ocornut/imgui). Она достаточно простая и легковесная, по сравнению с Qt Framework.

Для начала оптимизации выбирается стартовая точка, вокруг неё строится симплекс из линейно независимых векторов. Пример старт = (0, 0, 0), симплекс (1, 0, 0), (0, -1, 0), (0, 0, 1) + стартовая (0, 0, 0). Т. е. к координатам стартовой точки прибавляем 1 или -1 (случайно) в каждом измерении. Остановка оптимизации по числу шагов или Евклидовой мере симплекса (объёму).

1. **Основные классы:**

* Point - обёртка над std::vector для удобной работы с n-мерными точками
* Function - переводит исходную строку функции в польскую запись при создании и считает значение в точке с помощью метода Calculate.
* Solver - оптимизатор, в котором и реализован сам алгоритм Нелдера-Мида.

1. **UML диаграмма:**



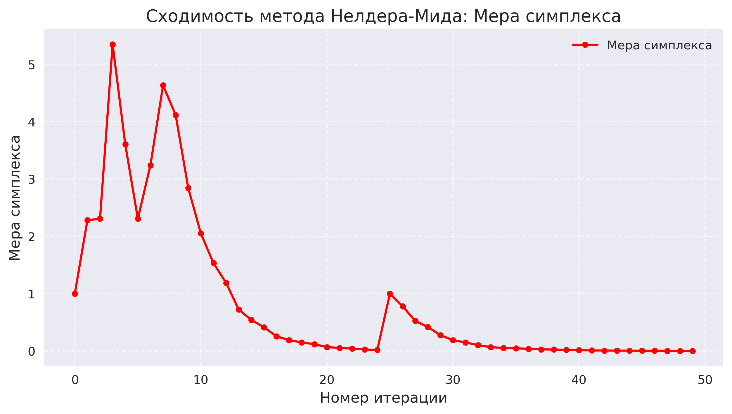
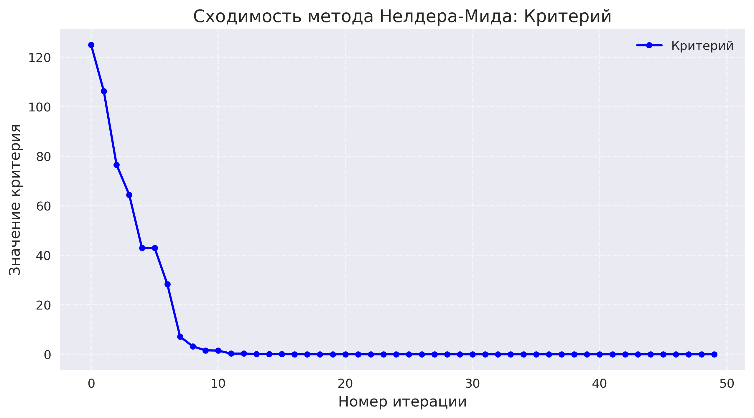
Подробнее с проектом можно ознакомиться в [github репозитории проекта](https://github.com/Golubeff-RA/Nelder-mead-optimization)

1. Исследование различных критериев
2. Квадратичная функция (параболоид)

*f*(*x*,*y*)=*x*2+*y*2

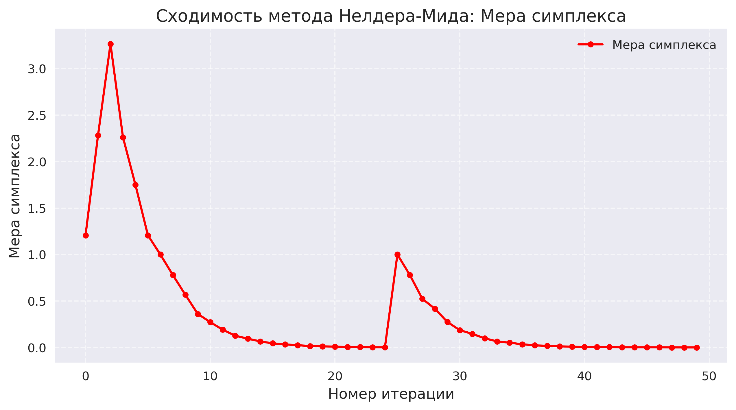
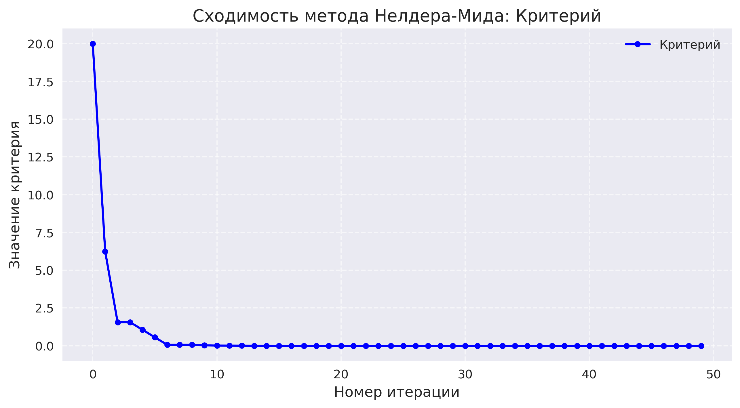
1. Запуск из точки (-5, 10) на 50 итераций:

Результат работы приложения: Q(X) = 0.0, точка: (0.000183, 0.000236})



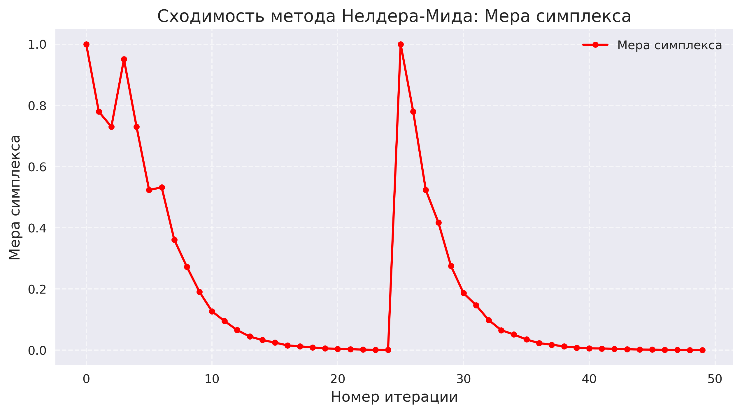
1. Запуск из точки (3, -4) на 50 итераций:

Результат работы приложения: Q(X) = 0.0, точка: (0.000095, 0.000116)



1. Запуск из точки (0.5, 0.5) на 50 итераций:

Результат работы приложения: Q(X) = 0.0, точка: (0.000102, -0.000193)



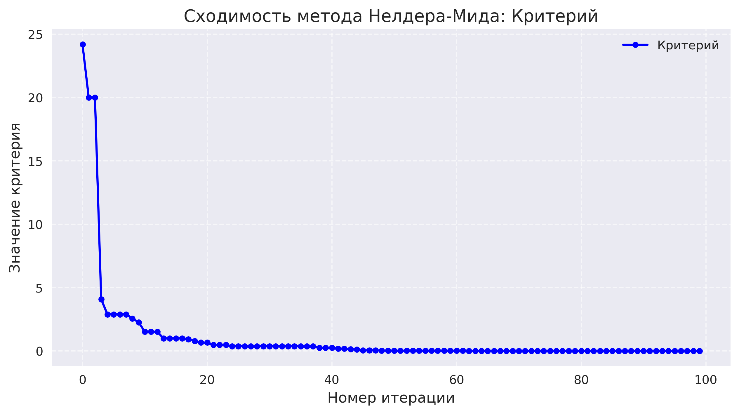
Вывод: данная функция быстро сходится при различных начальных точках

1. Функция Розенброка (сложный овраг):

*f*(*x*,*y*)=(1−*x*)2+100(*y*−*x*2)2

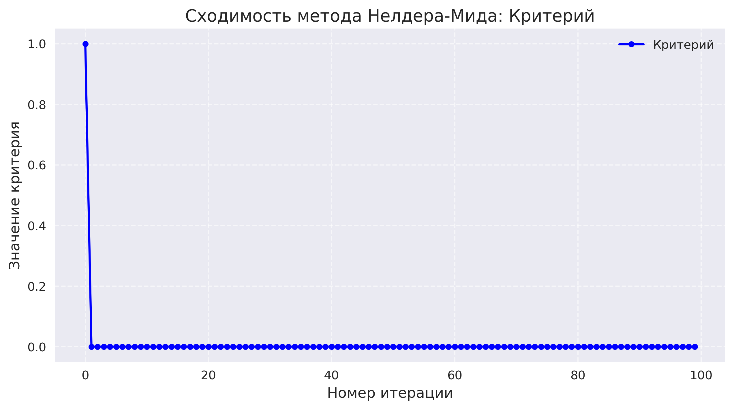
1. Запуск из точки (-1.2, 1) на 100 итераций:

Результат работы приложения: Q(X) = 0.0, точка: (0.999707, 0.999358)



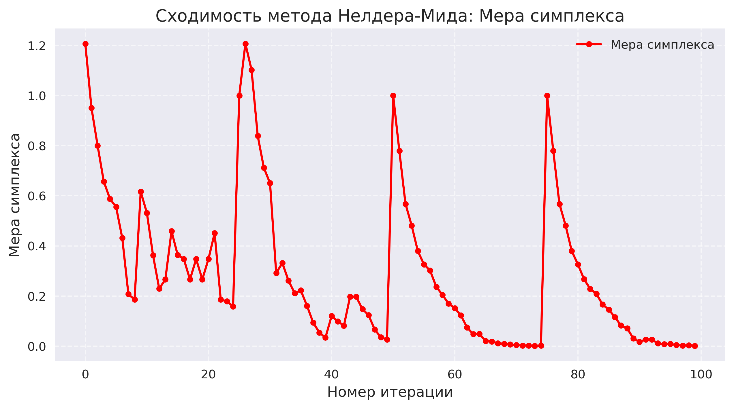
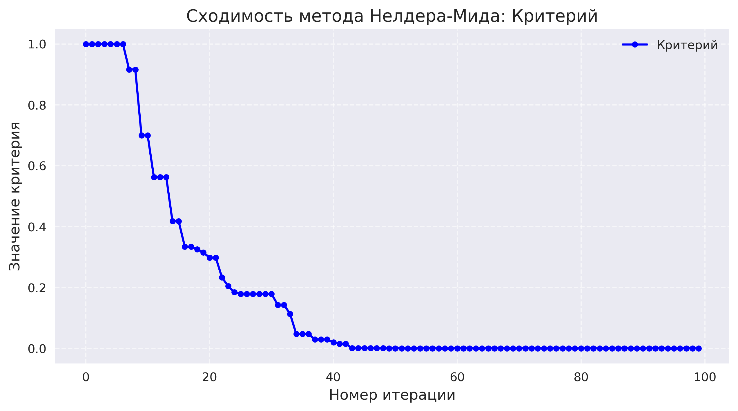
1. Запуск из точки (0, 0) на 100 итераций:

Результат работы приложения: Q(X) = 0.0, точка: (1.000000, 1.000000)



1. Запуск из точки (2, 3) на 100 итераций:

Результат работы приложения: Q(X) = 0.0, точка: (1.000197, 1.000378)



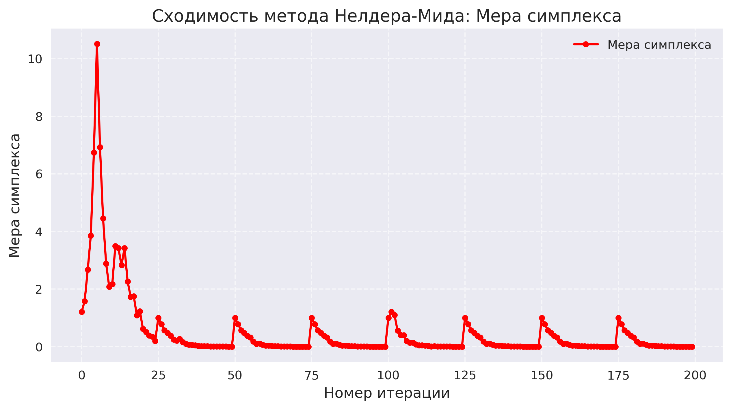
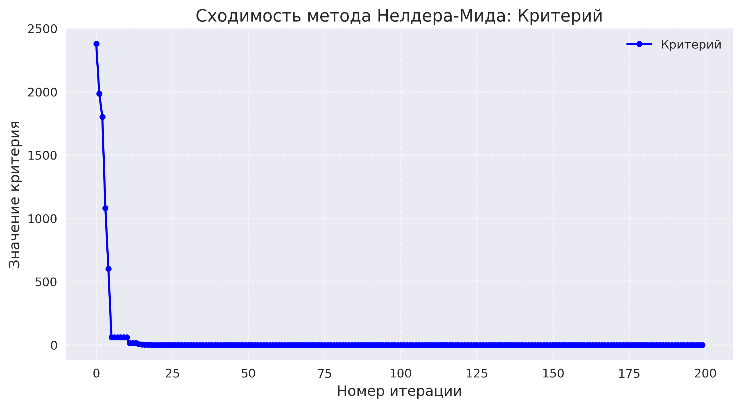
Вывод: сходимость функции зависит от начальной точки больше по сравнению с предыдущей функцией.

1. Функция Бута (простая выпуклая):

*f(x,y)=(x+2y−7)2+(2x+y−5)2*

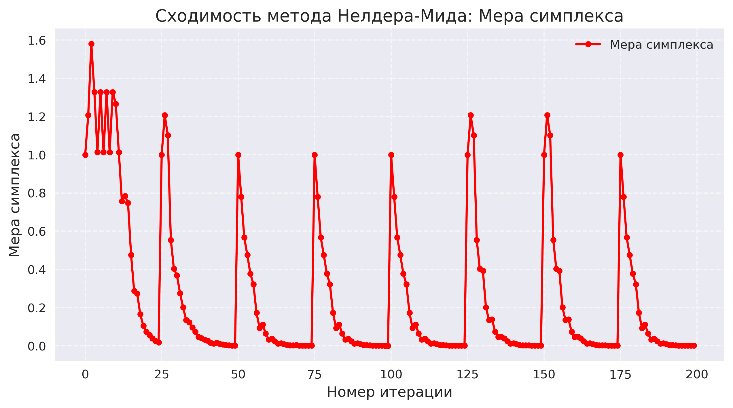
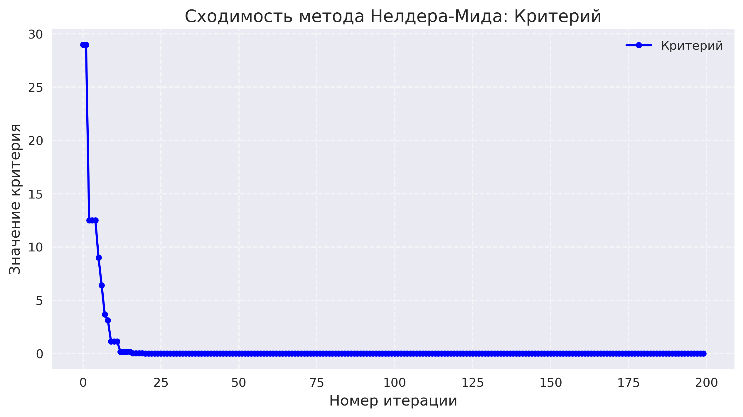
1. Запуск из точки (-10, -10) на 200 итераций:

Результат работы приложения: Q(X) = 0.0, точка: (1.000137, 2.999825)



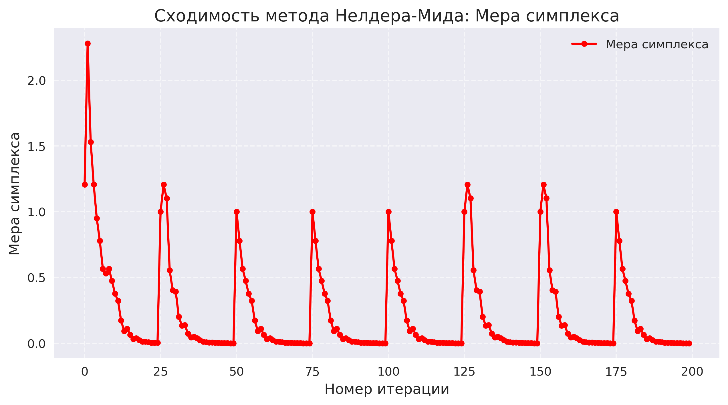
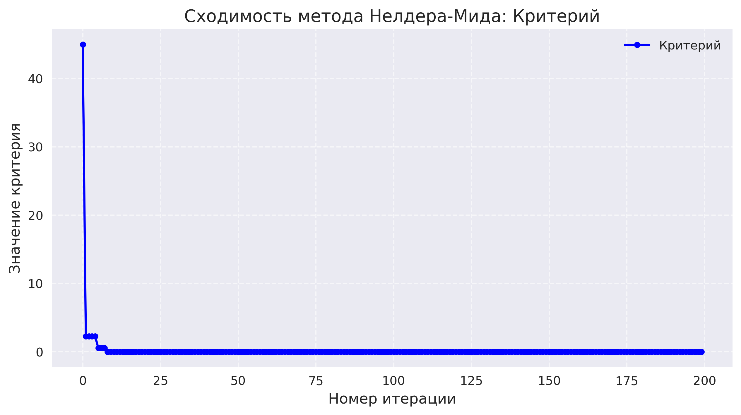
1. Запуск из точки (5, 0) на 200 итераций:

Результат работы приложения: Q(X) = 0.0, точка: (0.999844,3.000197)



1. Запуск из точки (0, 0) на 200 итераций:

Результат работы приложения: Q(X) = 0.0, точка: (1.000000,3.000000)



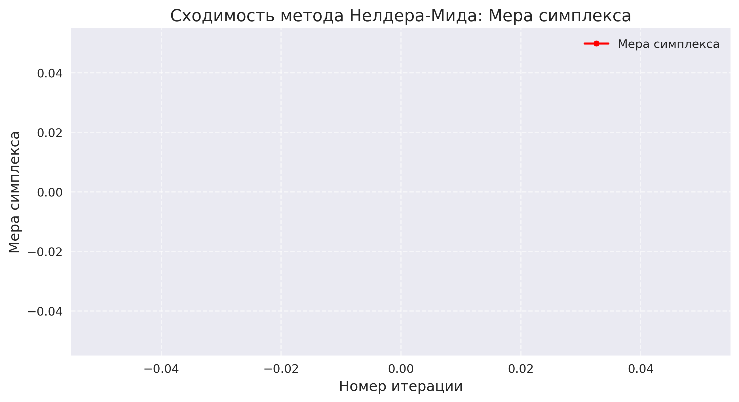
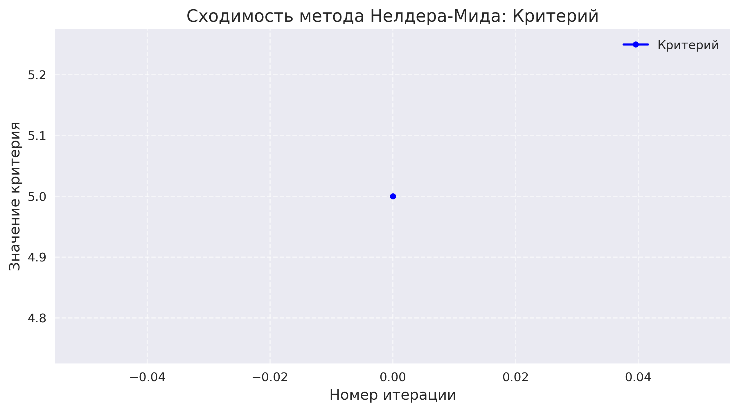
Вывод: функция хорошо сходится из различных начальных точек

1. Постоянная функция:

*f*(*x*,*y*)=5

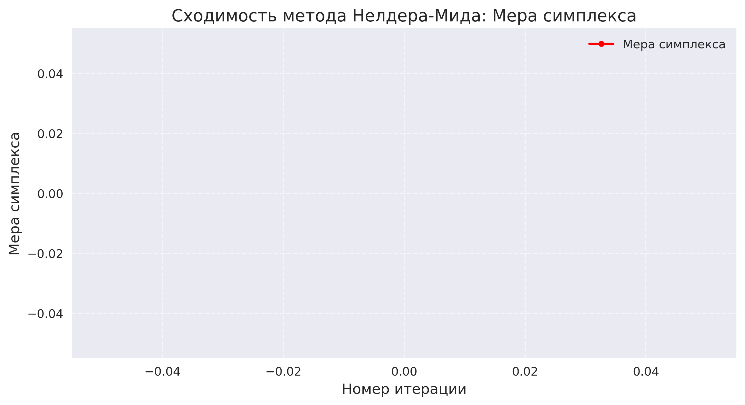
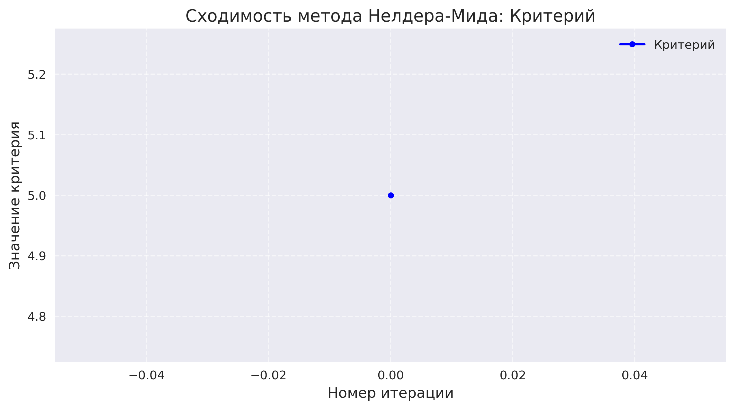
1. Запуск из точки (1, 2) на 200 итераций:

Результат работы приложения: Q(X) = 5.0, точка: (1.000000, 2.000000)



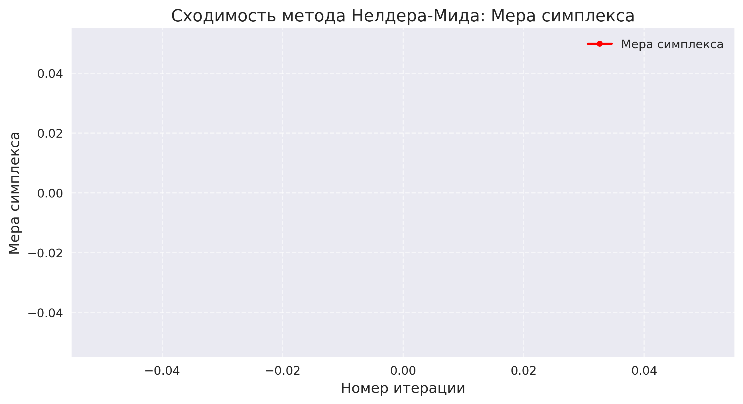
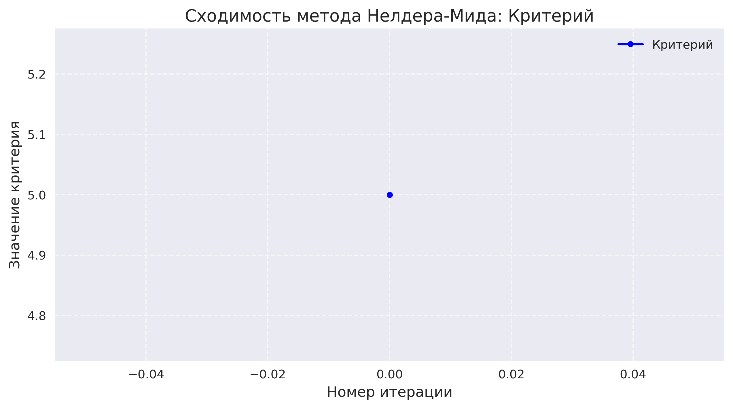
1. Запуск из точки (-3, 4) на 200 итераций:

Результат работы приложения: Q(X) = 5.0, точка: (-3.000000, 4.000000)



1. Запуск из точки (0, 0) на 200 итераций:

Результат работы приложения: Q(X) = 5.0, точка: (0.000000, 0.000000)



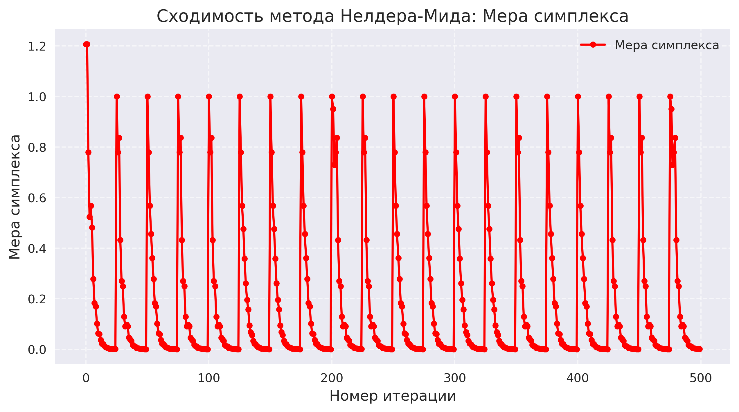
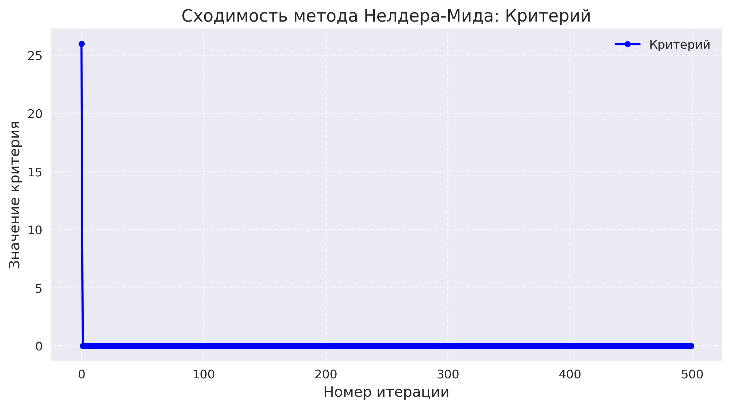
Вывод: из-за постоянности функции метод Нелдера-Мида не может построить симплекс и завершается преждевременно.

1. Функция Химмельблау:

*f*(*x*,*y*)=(*x*2+*y*−11)2+(*x*+*y*2−7)2

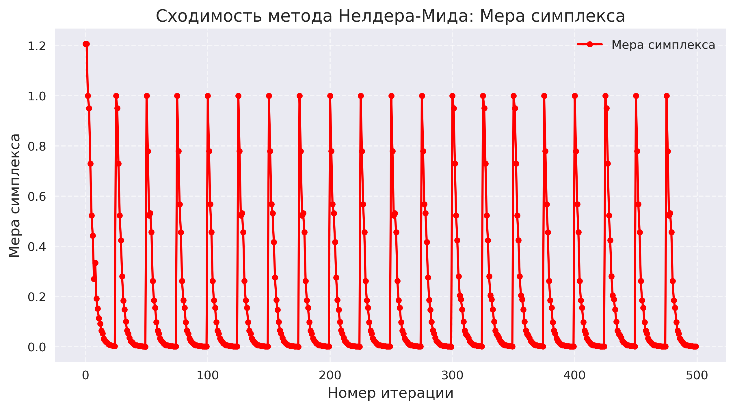
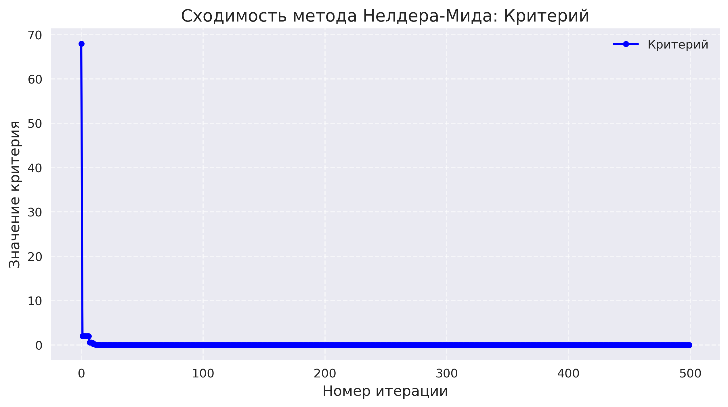
1. Запуск из точки (4, 3) на 500 итераций:

Результат работы приложения: Q(X) = 0.0, точка: (3.000000, 2.000000)



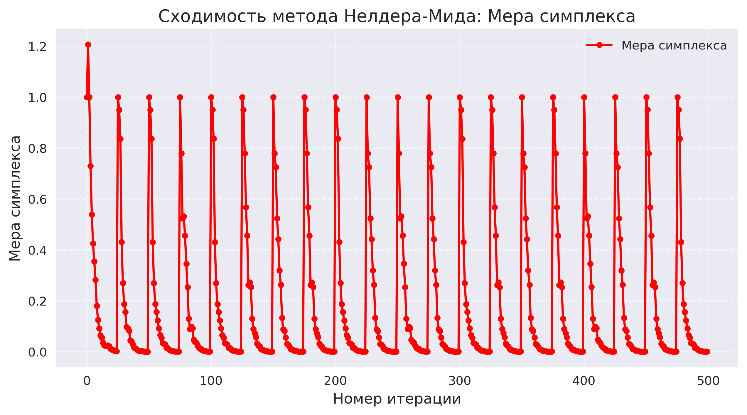
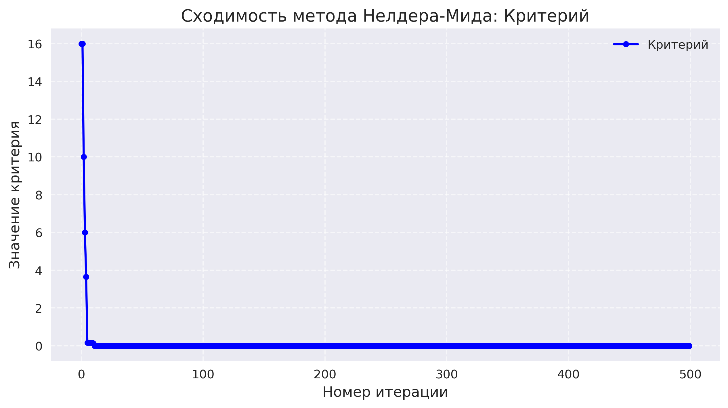
1. Запуск из точки (-4, 4) на 500 итераций:

Результат работы приложения: Q(X) = 0.0, точка: (-2.805273, 3.131184)



1. Запуск из точки (3, -2) на 500 итераций:

Результат работы приложения: Q(X) = 0.0, точка: (3.584351, -1.848205)



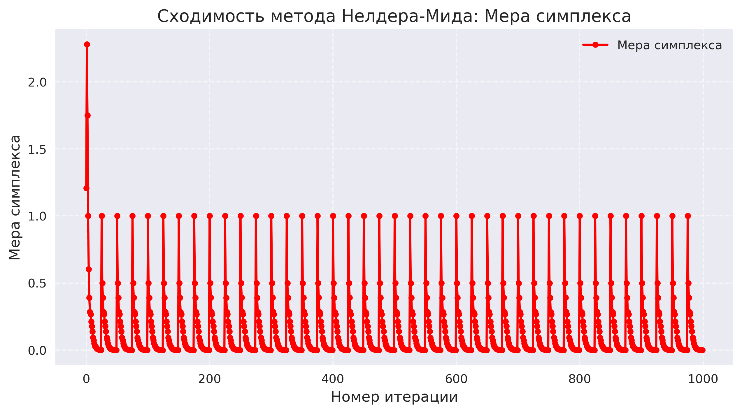
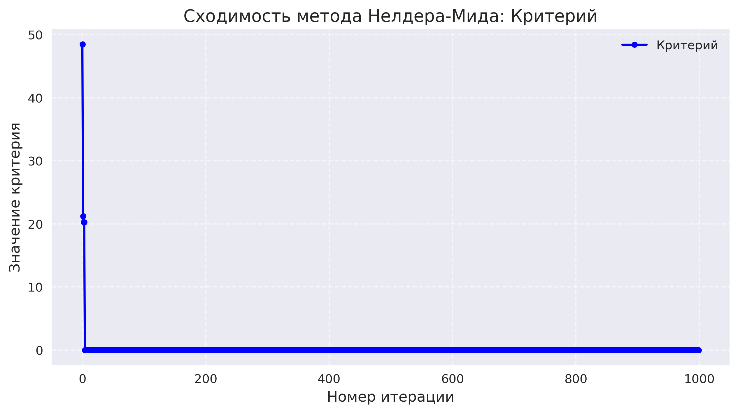
Вывод: данная функция имеет несколько локальных оптимумов и при различных начальных точках метод Нелдера-Мида сходится к разным оптимумам.

1. Функция Растригина:

*f*(*x*,*y*)=20+(*x*2−10cos(2*πx*))+(*y*2−10cos(2*πy*))

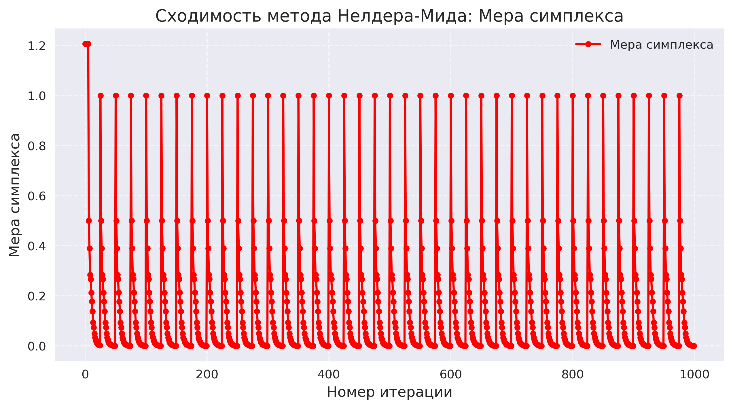
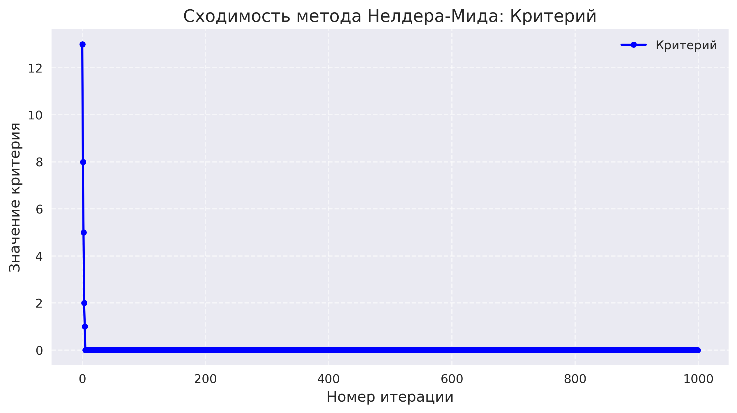
1. Запуск из точки (2.5, 2.5) на 1000 итераций:

Результат работы приложения: Q(X) = 0.0, точка: (0.000000, 0.000000)



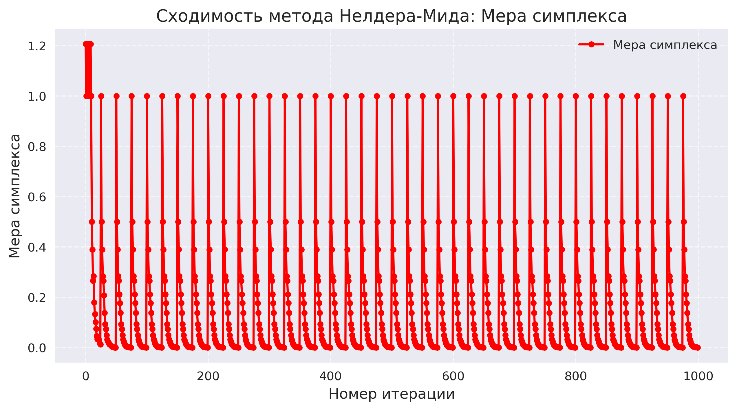
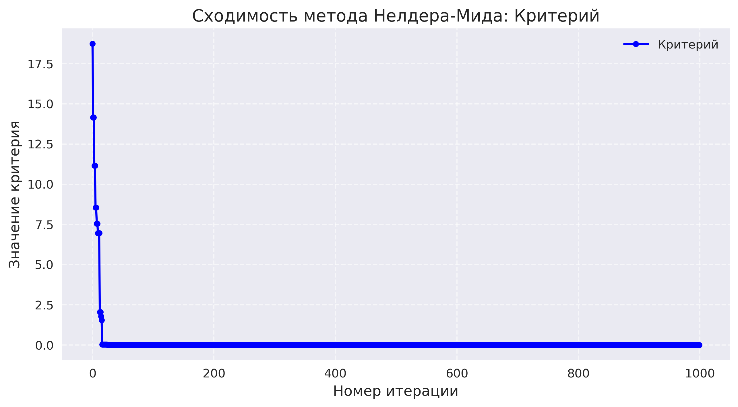
1. Запуск из точки (-3, -3) на 1000 итераций:

Результат работы приложения: Q(X) = 0.0, точка: (0.000000, 0.000000)



1. Запуск из точки (-3, -2.8) на 1000 итераций:

Результат работы приложения: Q(X) = 0.0, точка: (0.000000, 0.000000)



Вывод: несмотря на то, что функция Растригина имеет бесконечное множество локальных минимумов, метод Нелдера-Мида из различных начальных точек сходится к глобальному оптимуму.

1. Общий вывод

При достаточном количестве итераций метод Нелдера-Мида сходится к оптимуму (к одному из, если их несколько) даже при наличии локальных минимумов, метод не “застревает” в них. Метод зависит от построения симплекса, поэтому при невозможности его построения завершается преждевременно (как было с постоянной функцией).