Общее

Все эксперименты реализованы в Experiments.py. Для вызова какого -либо из опытов, нужно найти функцию all_experiments. Первым аргументов передать номер эксперимента, второй аргумент – номер датасета для эксперимента 3, начиная с 0. Датасеты упорядочены по описанию лабораторной работы 3-го эксперимента. Запуск через командную строку не требовался, но при желании можно реализовать. За помощь с подсчетом градиента логистическй функции (численно, аналитически спокойно выводится) благодарю Алима Ханмурзина.

Работы выполнил студент магистратуры Сергаев Ярослав Сергеевич с направления «Программирование и анализ данных».

Эксперимент 1*: Траектория градиентного спуска (ГС) на квадратичной функции*

Описание: в данном эксперименте нужно рассмотреть траектории ГС на линиях уровня генерируемых квадратичных ораклов. Для каждого опыта выбиралась начальная точка, из которой нужно достигнуть глобального минимума функции, ввиду её липшецивости, а также исследовалась зависимость от метода линейного поиска шага алгоритма ГС.

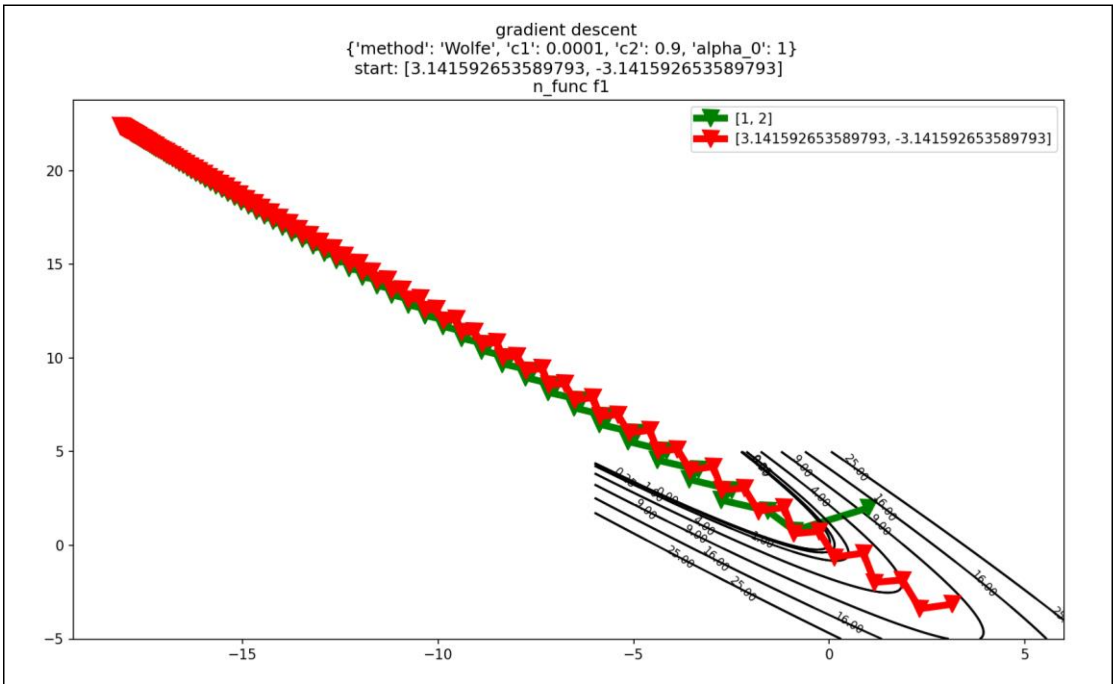

Рис 1. Метод Вульфа для первой генерируемой функции

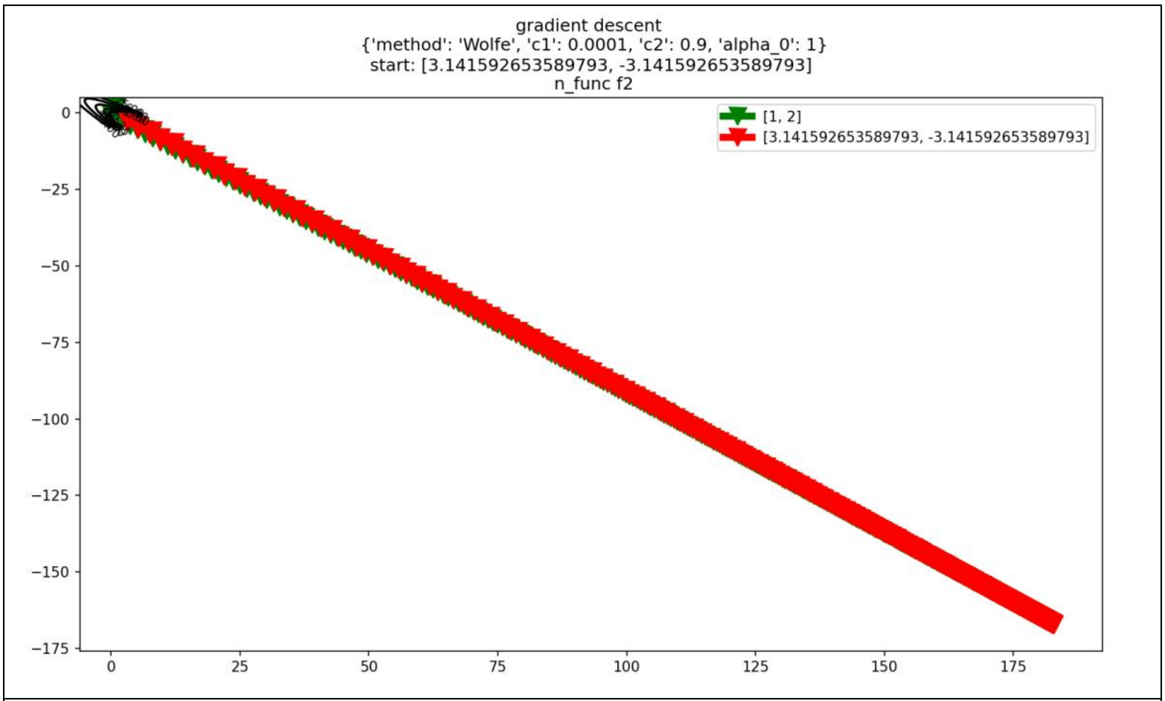

Рис 2. Метод Вульфа для второй генерируемой функции

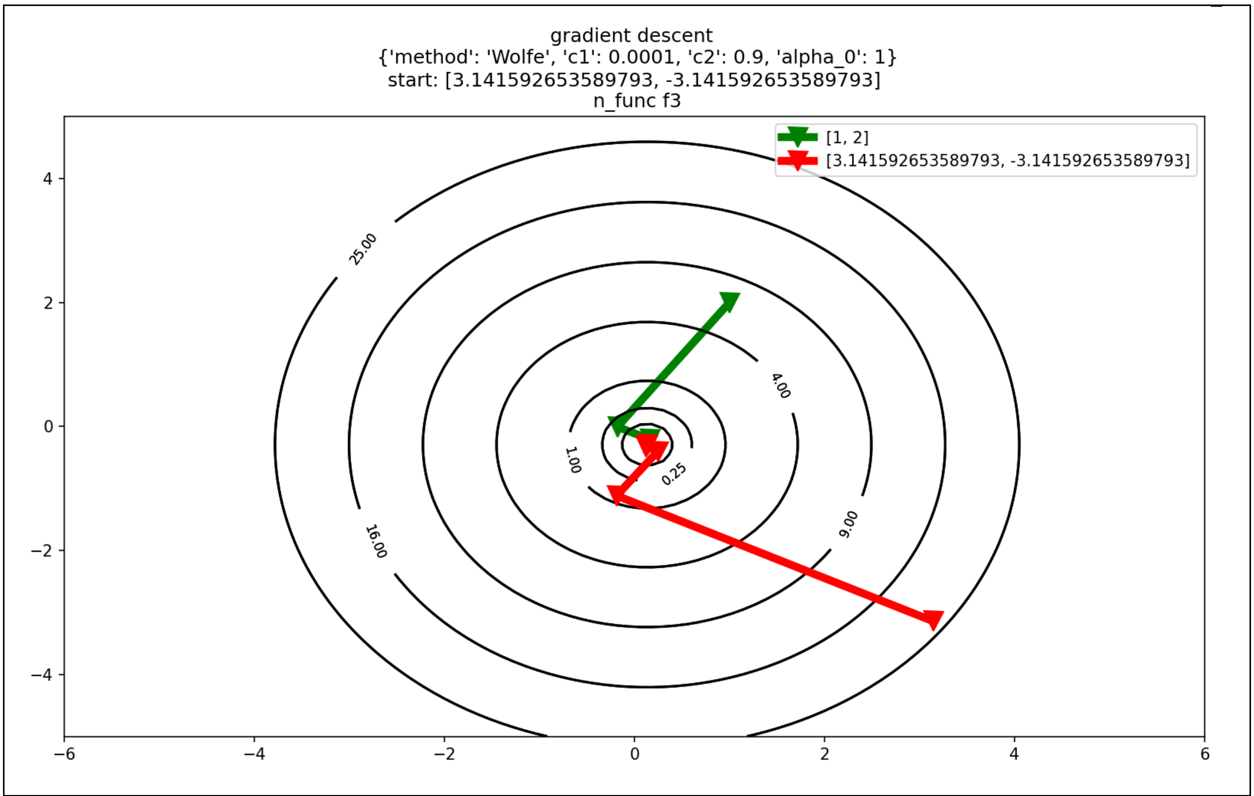

Рис 3. Метод Вульфа для третей генерируемой функции

По первым результатам можно сказать, что когда линии уровня напоминают эллипсы, метод сходится, однако, можно заметить, что для функции с параболическим видом линий уровня метод метод расходится

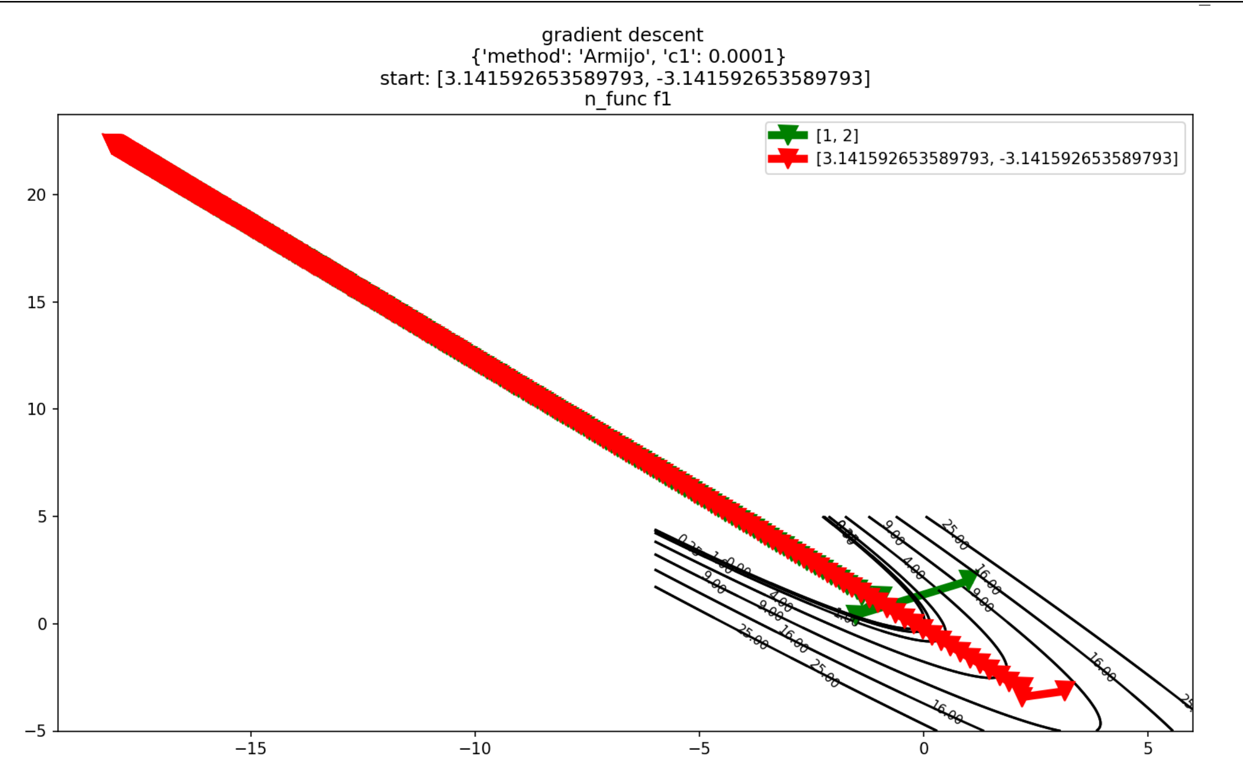

Рис 4. Метод Армихо для четвертой генерируемой функции

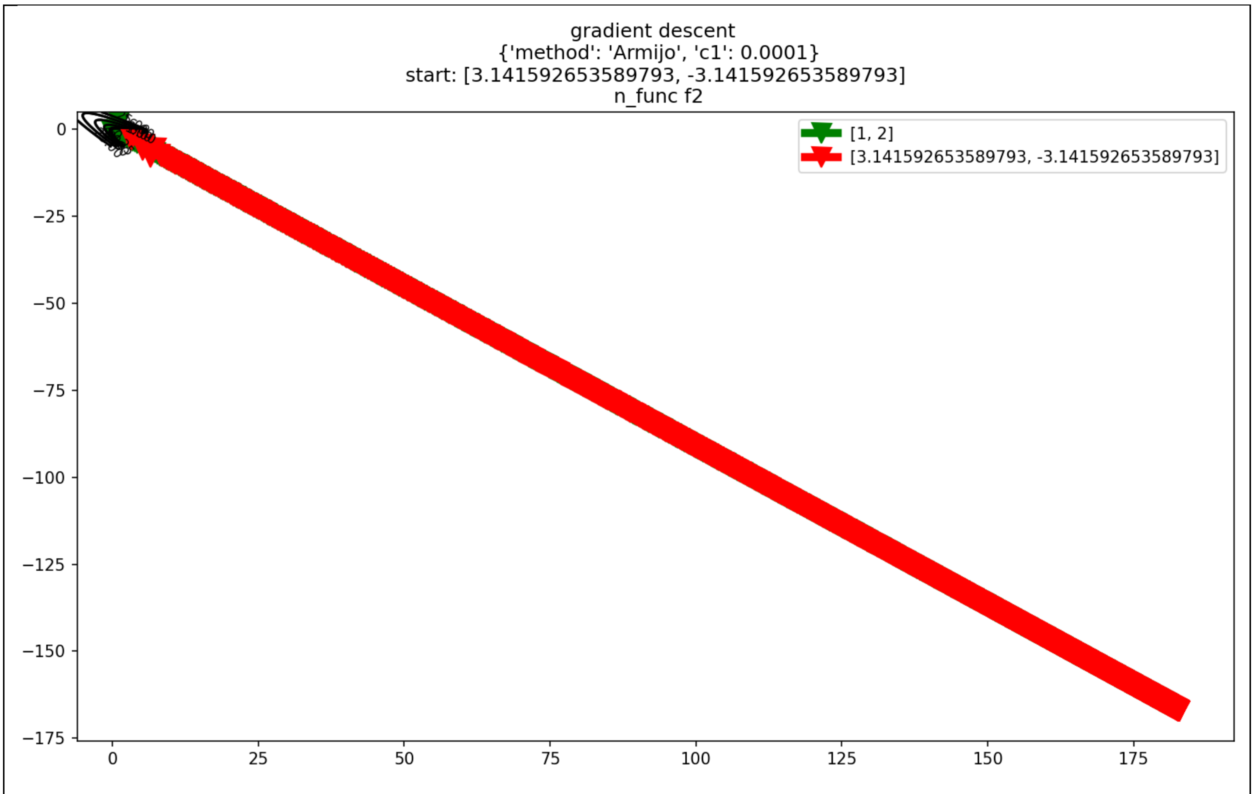

Рис 5. Метод Армихо для второй генерируемой функции

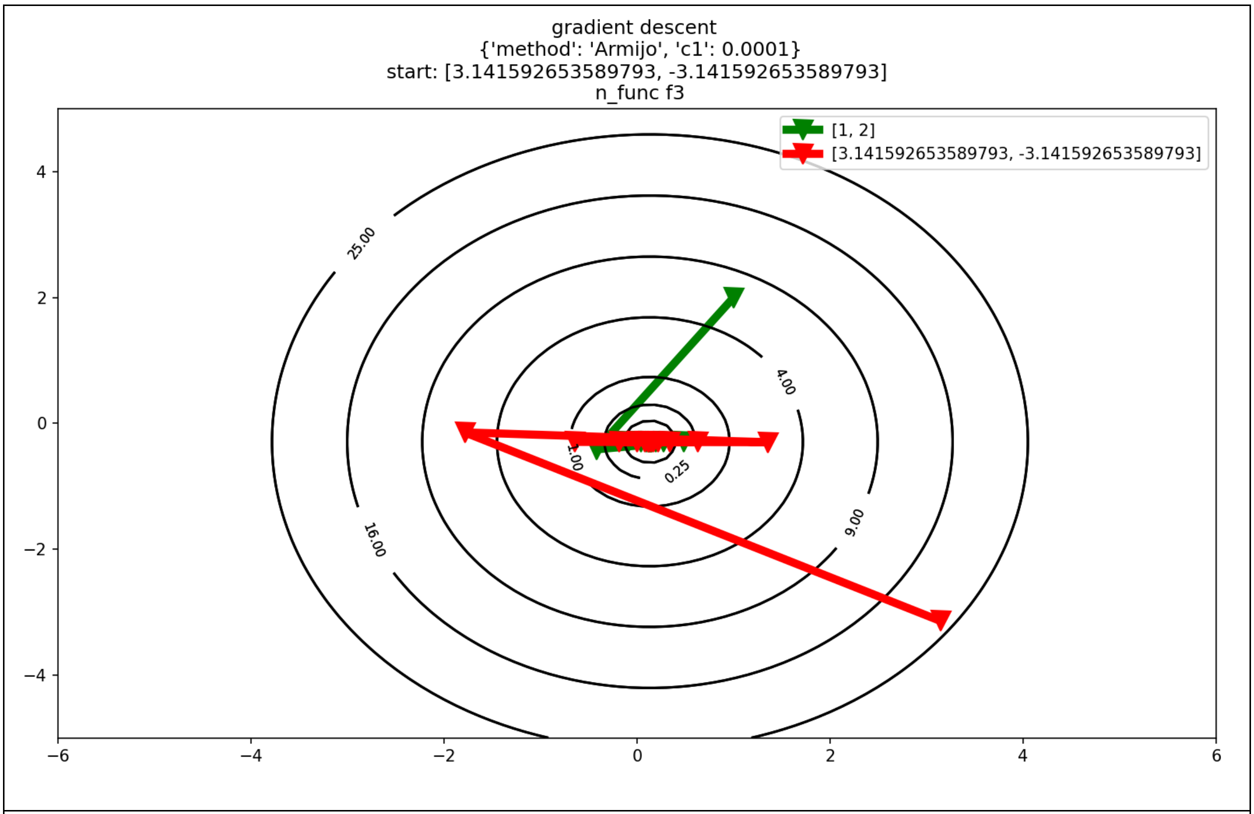

Рис 6. Метод Армихо для третей генерируемой функции

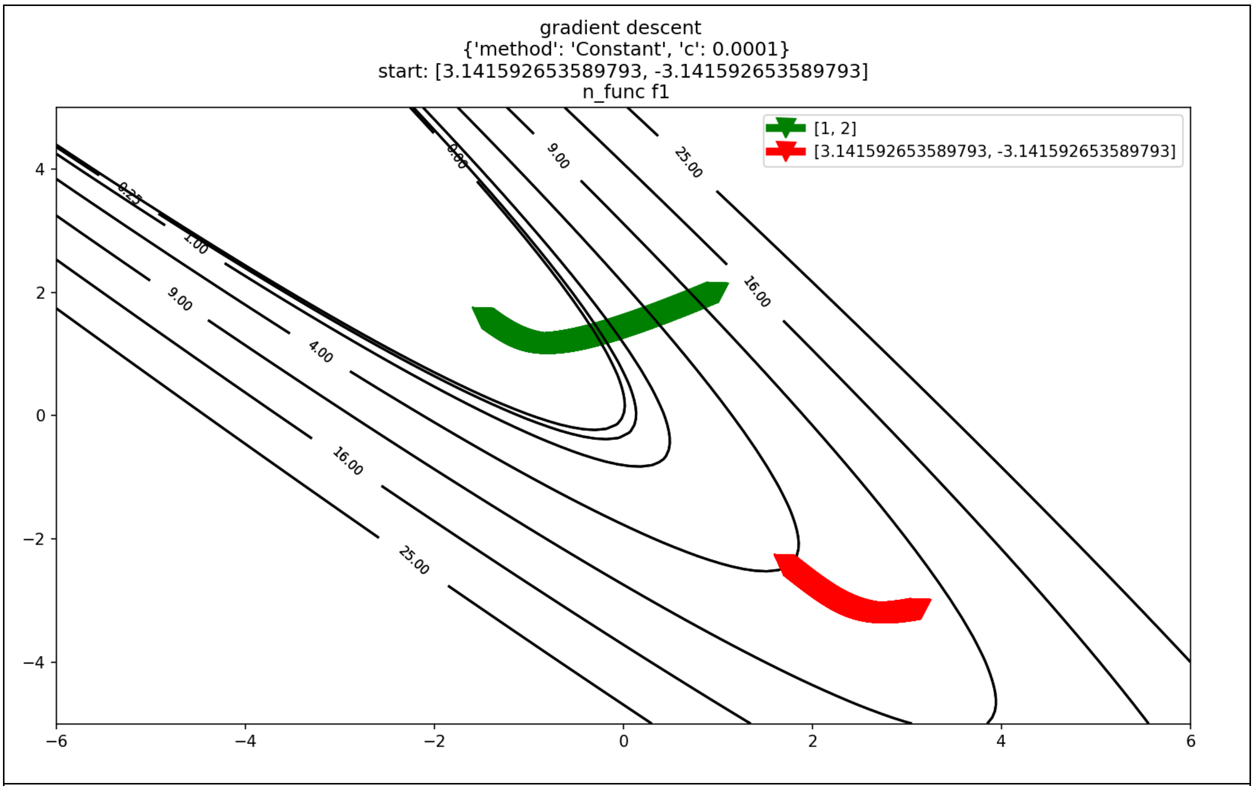

Рис 7. Константный метод для первой генерируемой функции

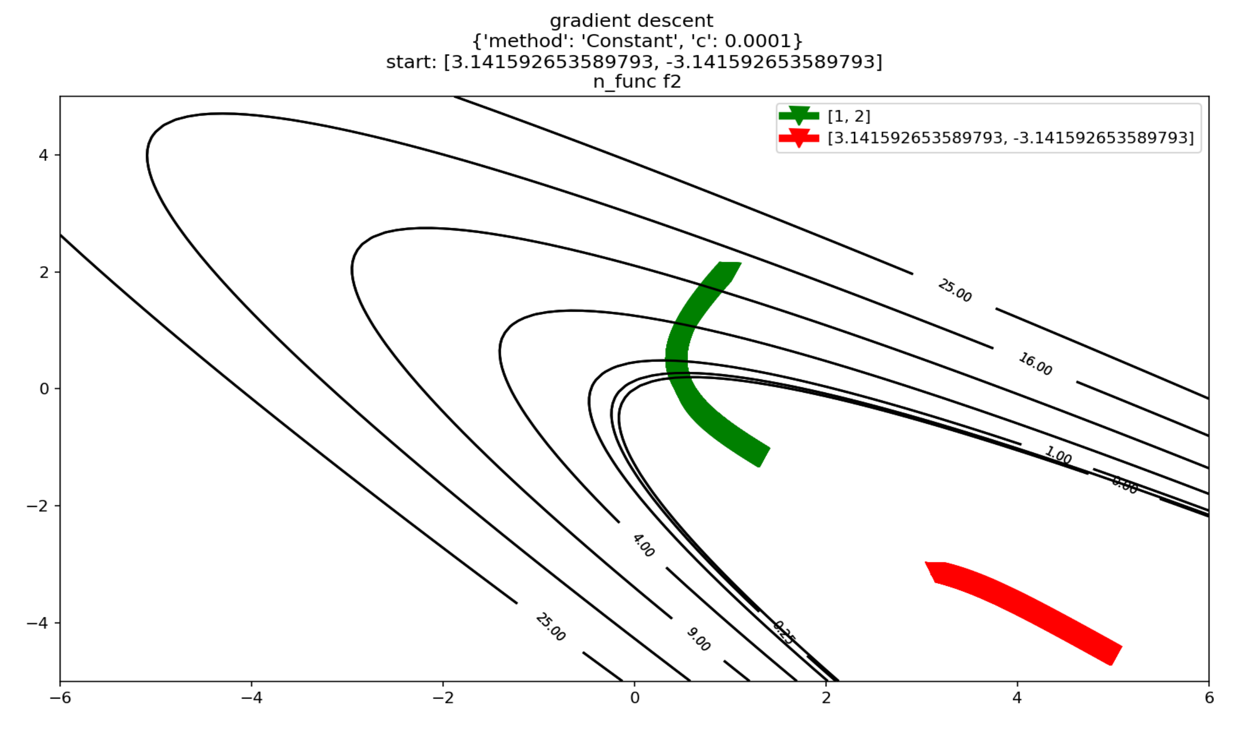

Рис 8. Константный метод для второй генерируемой функции

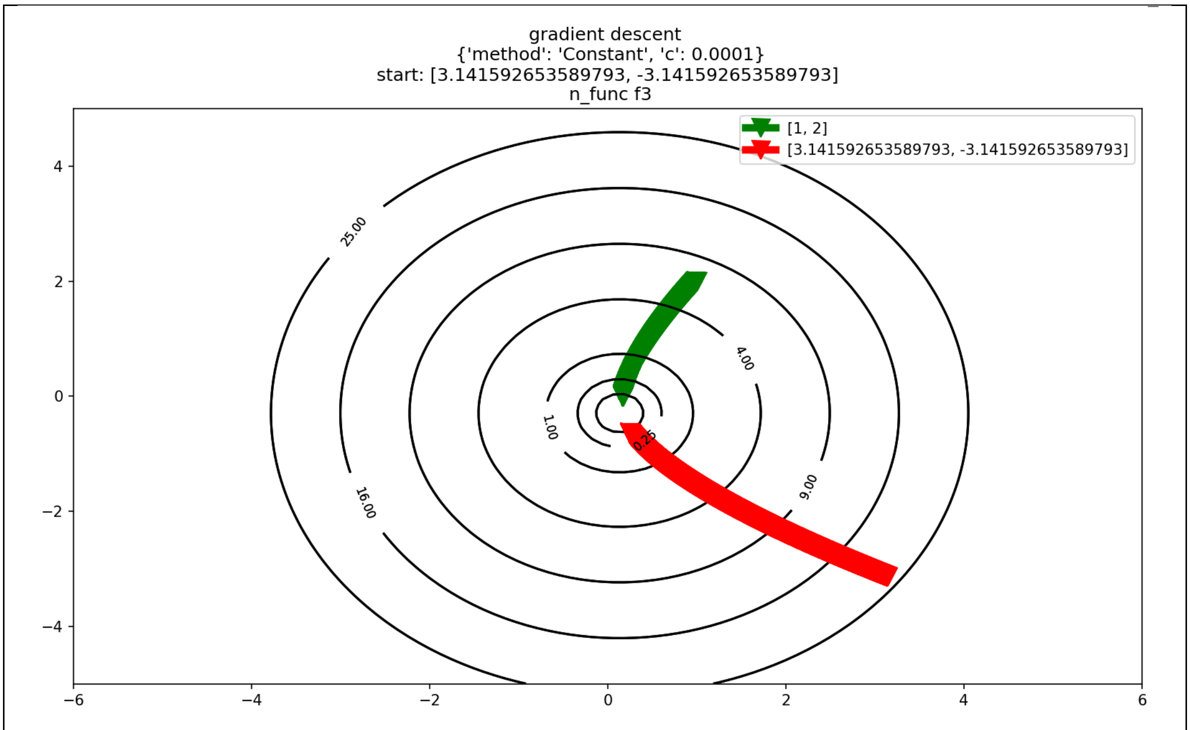

Рис 9. Константный метод для третей генерируемой функции

Видно, что Градиентный спуск на данном наборе данных сходится для всех функций при константном подходе выборе шага спуска alpha. Есть смысл рассмотреть зависимость при более большом выборе шага. Про числа обусловленности задачи: чем больше – тем хуже сходимость методов с применением бэктрэекинга для выполнения условий Армихо, а также, хуже с применение сильных условий Вульфа.

Эксперимент 2: *Зависимость числа итераций градиентного спуска от числа обусловленности и размерности пространства*

Описание: этом эксперименте рассматривается число итерация в ГС в зависимость от чисел обусловленности генерируемой задачи и размерности пространства

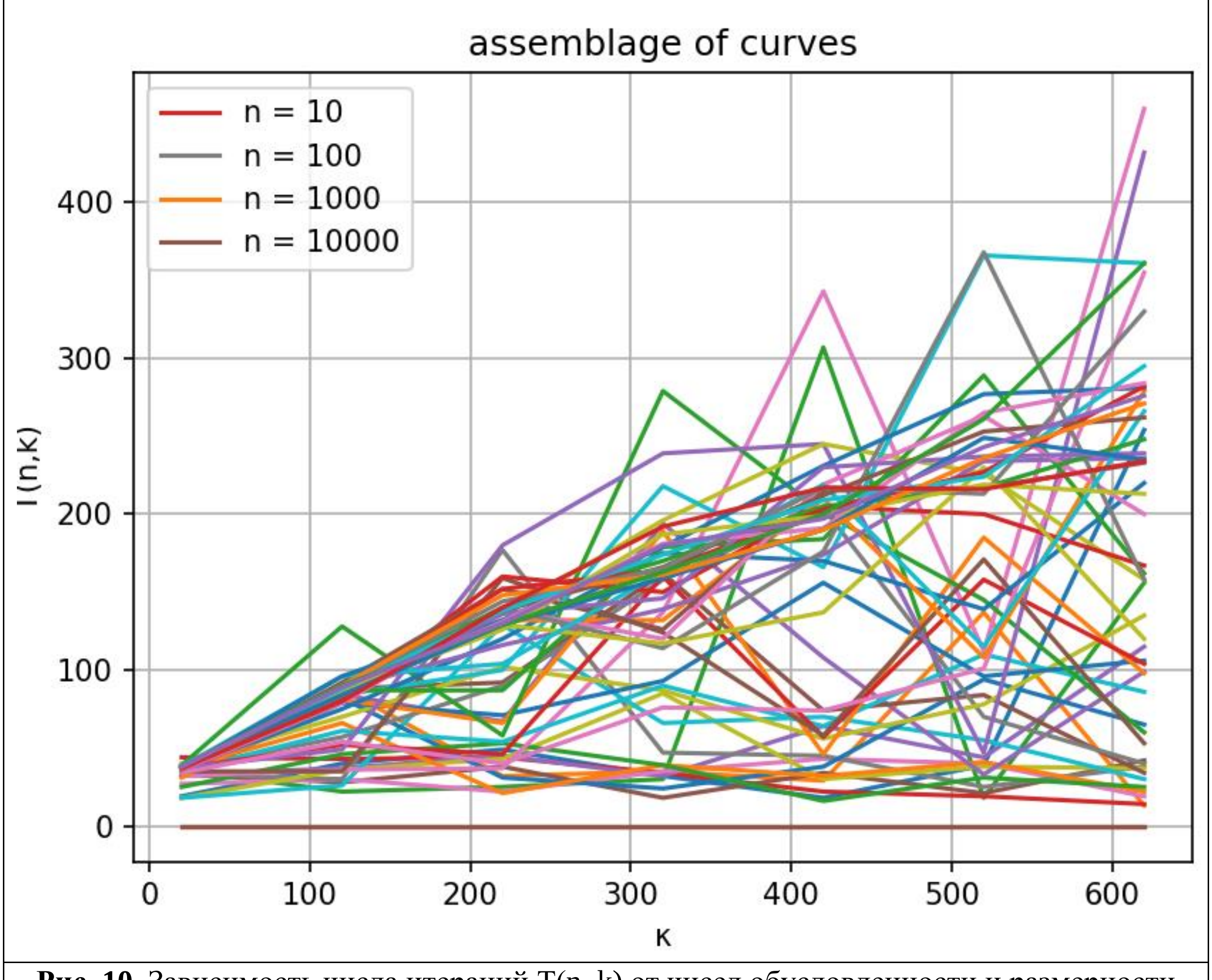

Рис. 10. Зависимость числа итераций T(n, k) от чисел обусловленности и размерности пространства (цвет)

По результатам, можно сказать, что при малых K происходит наиболее быстрая сходимость метода а при малых n требуется меньше итераций для сходимости, чем при больших, это, в целом, логично.

Эксперимент 3*: Сравнение методов градиентного спуска и Ньютона на реальной задаче логистической регрессии*

Описание: для 3 датасетов с svmlight: *w8a, gissete, real_sim* решить задачу бинарной классификации в зависимости с использованием логистической регрессии.
анализ результатов предлагается делать с отрисовки фукции и отношения квадрата модуля градиенов.

Дана задача A размера n x m.
n – размерность признакового пространства, m – число элементов пространства (число признаков)

Операция matvec_Ax работает за O(m), поэлементноное умножение за O(n)

Таким образом первая операция – в экспоненте - работает за O ((nm))

Для подсчета алгоритмической сложности операции возведения в степень, нужно переходить в двоичное представление чисел.
Чтобы это найти, нужно найти длину двоичного представления максимума из результата перемножения матрицы на вектор – назовем эту длину q. Таким образом, сложность O(ln(q)nm). А с учётом ещё одного поэлементного умножения O(ln(q)mn^2). Взятие транспонирования обходится в O(nm), а последующее умножение – O(n). Таким образом, сложность O(nm + ln(q) mn^3) ~ O(ln(q) mn^3). Если не учитывать сложность операций умножения и сложение на одного элемента.

Алгоритм Вольфа, на k итерациях имеет сходимость O(1/k).

Таким образом, сложность прохода градиентного спуска составит O(max_iter/k + max_iter* ln(q) mn^3) .

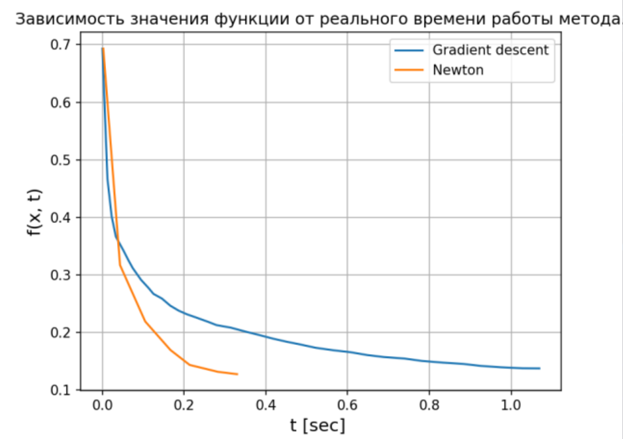

Рис. 11. w8a. Зависимость значения функции от реального времени работы мето-
да.

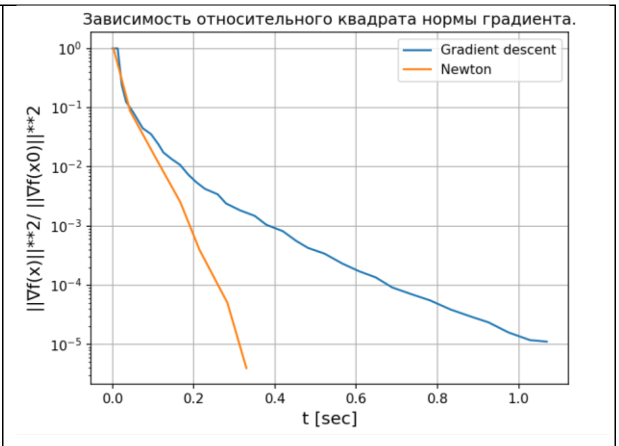

Рис. 12. w8a. Зависимость относительного квадрата нормы градиента (в логарифмической шкале)

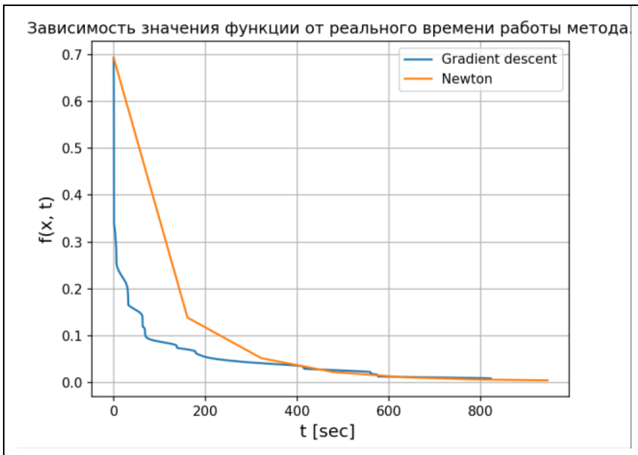

Рис. 13. Gissete. Зависимость значения функции от реального времени работы метода.

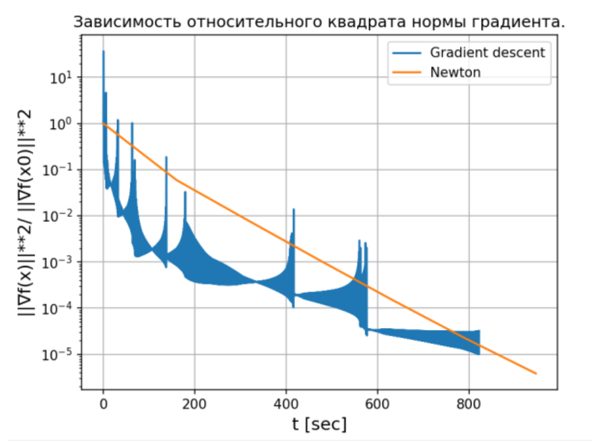

Рис. 14. Gissete. Зависимость относительного квадрата нормы градиента (в логарифмической шкале)

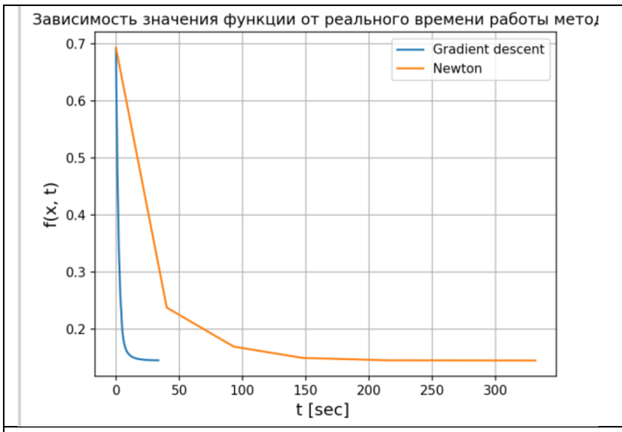

Рис. 15. Real-sem. Зависимость значения функции от реального времени работы метода.

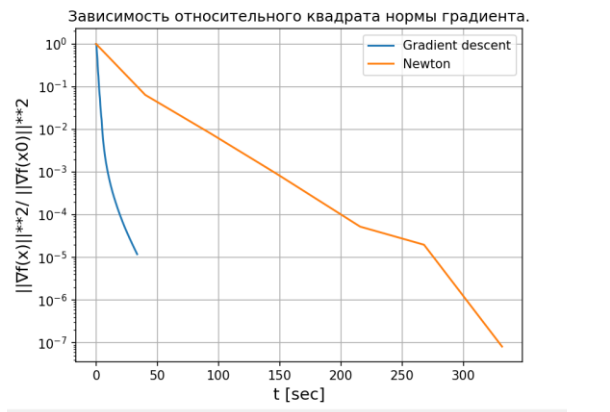

Рис. 16. Real-sim Зависимость относительного квадрата нормы градиента (в логарифмической шкале)

На разных наборах данных заметно, что метод Ньютона обеспечивает наиболее быструю сходимость, чем метод Градиентного спуска. Во всех стратегиях для линейного поиска использовался метод Вульфа по умолчанию.

Эксперимент 4*: Стратегия выбора длины шага в градиентном спуске.*

Описание: для каждого из метода оптимизации требовалось Исследовать, как зависит поведение данного метода от стратегии подбора шага константный шаг (попробовать различные значения), бэктрэкинг (попробовать различные константы c), условия Вульфа (попробовать различные пара-
метры c2).

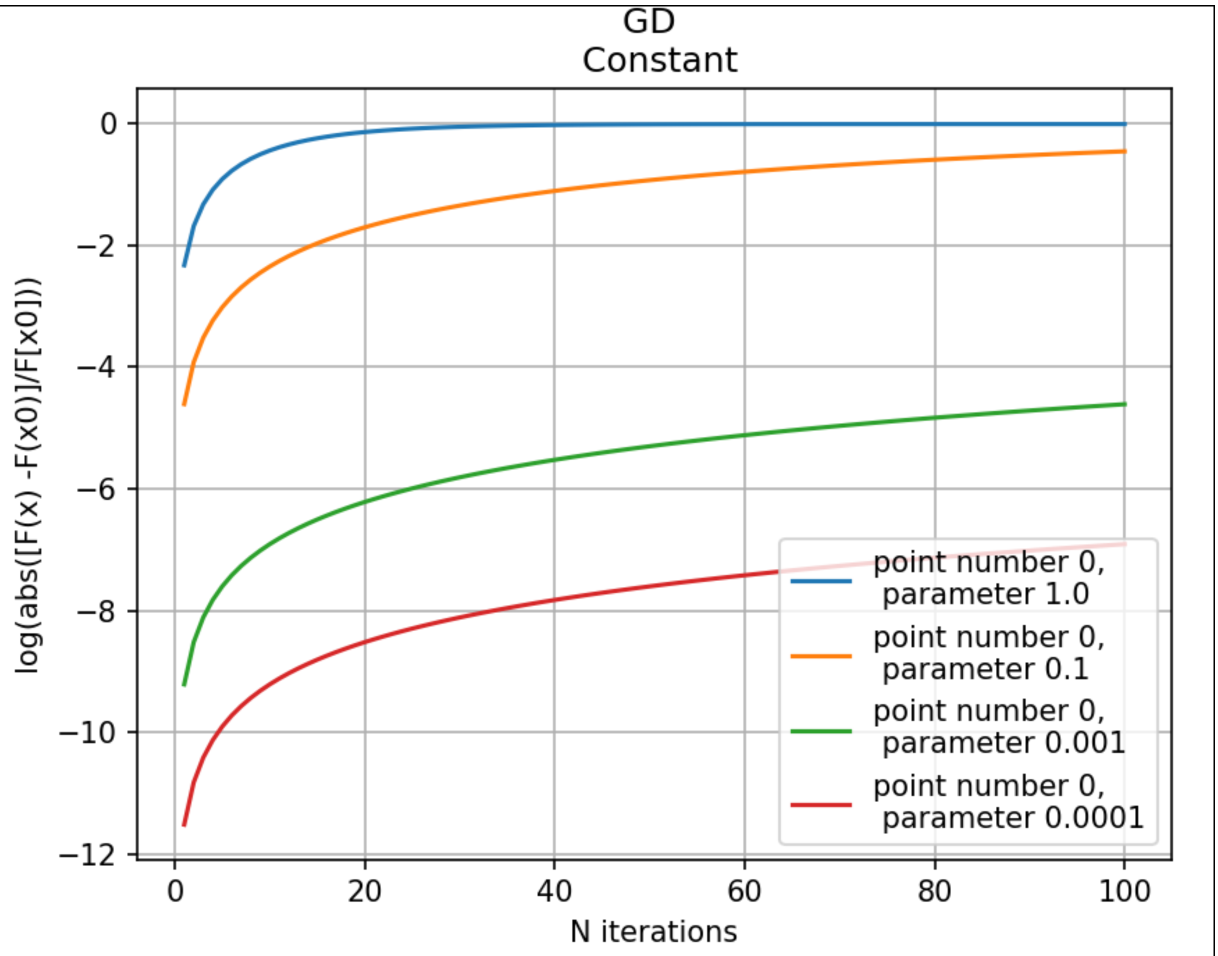

Рис. 17. Квадратичный оракул. Метод оптимизации: Градиентный спуск; линейный поиск: с постоянным шагом

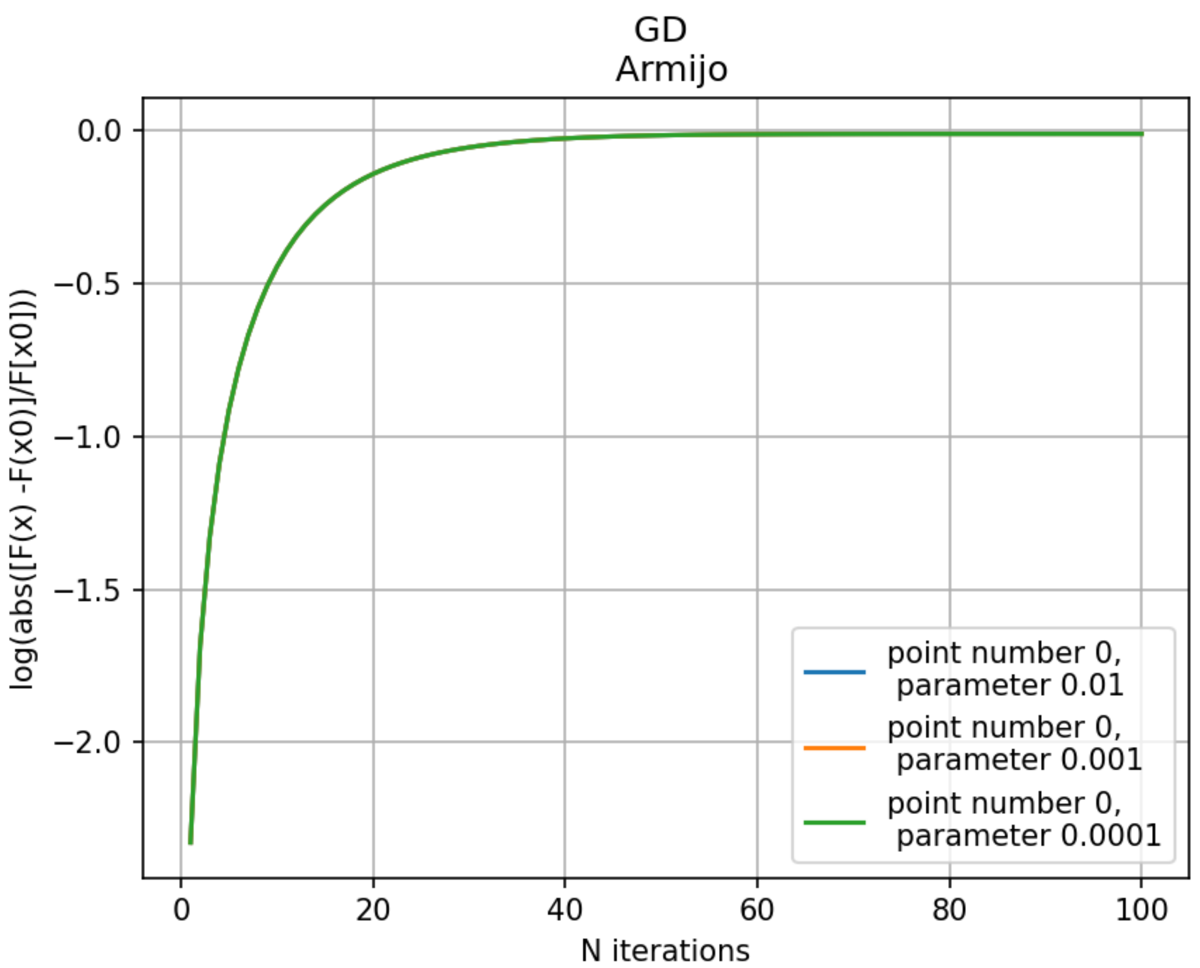

Рис. 18. Квадратичный оракул. Метод оптимизации: Градиентный спуск; линейный поиск: Армихо

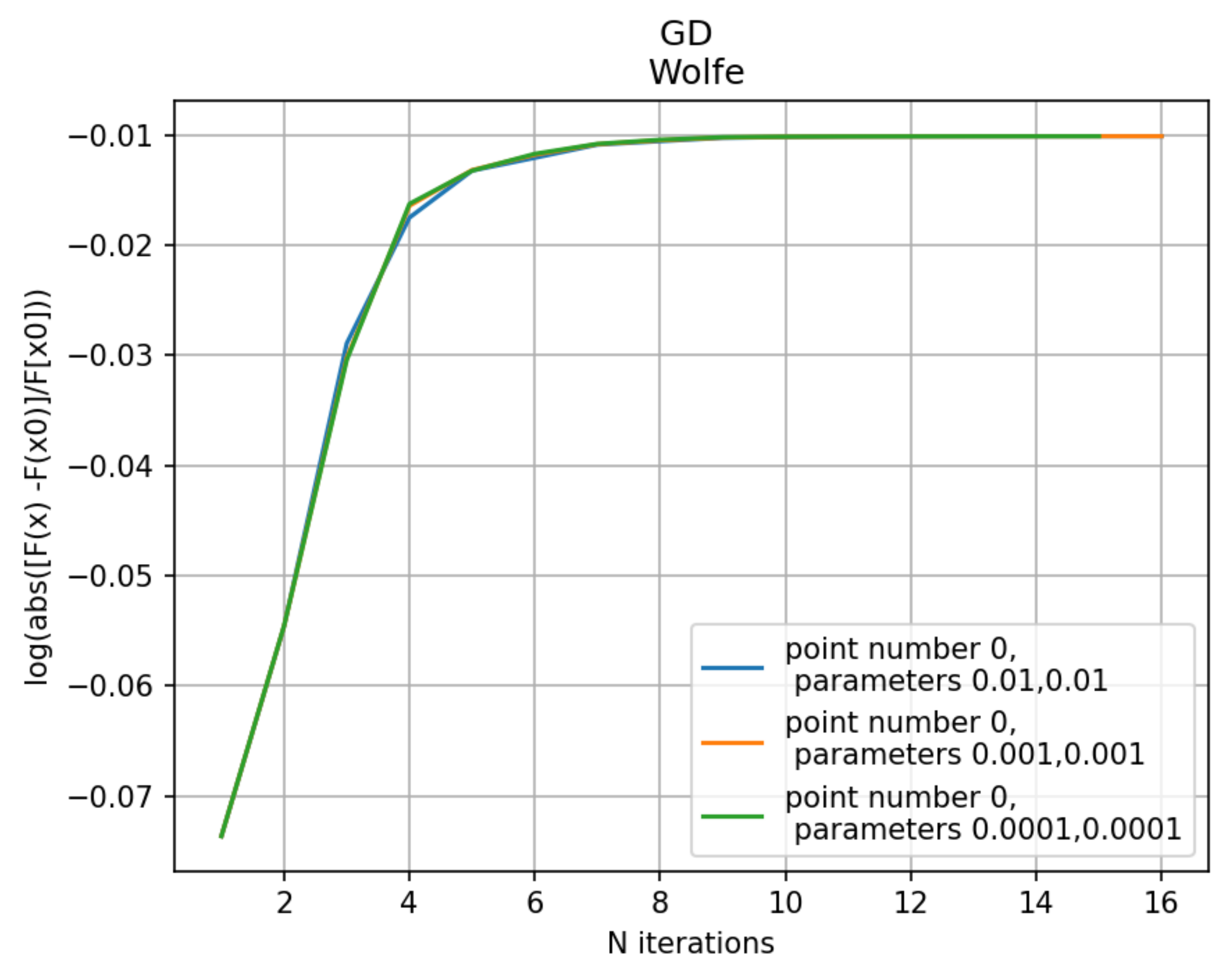

Рис. 19. Квадратичный оракул. Метод оптимизации: Градиентный спуск; линейный поиск: Вульф

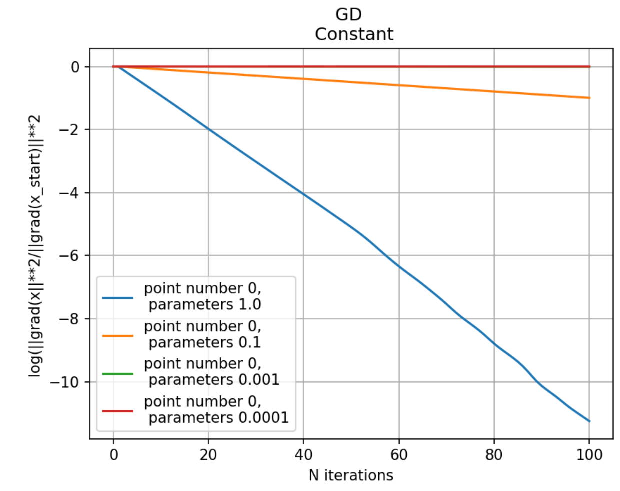

Рис. 20. Логистическая Регрессия. Метод оптимизации: Градиентный спуск; линейный поиск: с постоянным шагом

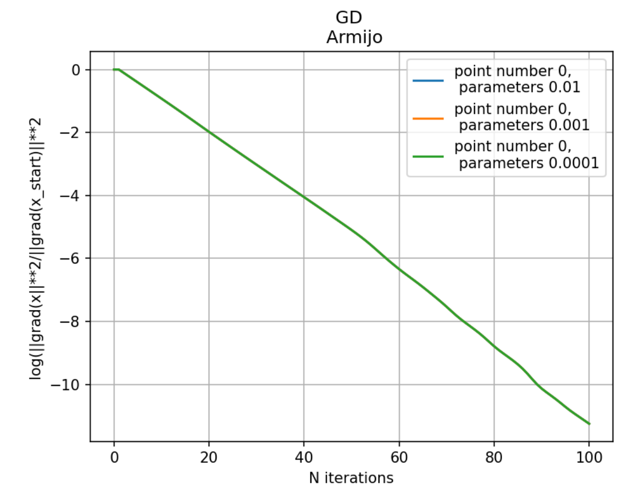

Рис. 21. Логистическая Регрессия. Метод оптимизации: Градиентный спуск; линейный поиск: Армихо

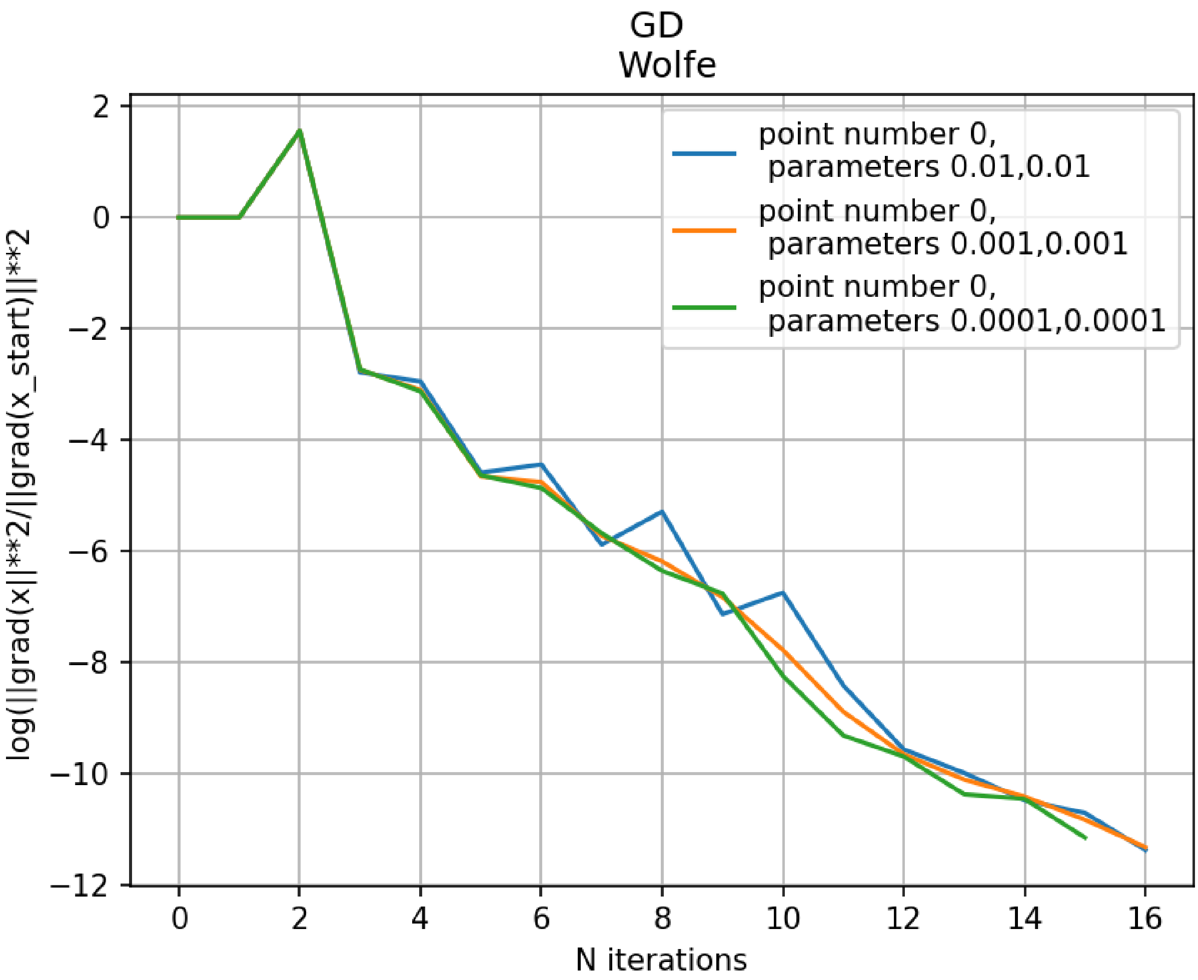

Рис. 22. Логистическая Регрессия. Метод оптимизации: Градиентный спуск; линейный поиск: Вульф

Для квадратичного оракула: из приведённых выше результатов лучшие результаты показывает линейный поиск с постоянным шагом c = 0.0001

Для логистической регрессии: лучшая сходимость из графиков достигается при применении линейный поиск с сильными условиями Вульфа, более того, требуется меньшее число итераций, чем в других методах линейного поиска.

Эксперимент 5*: Стратегия выбора длины шага в методе Ньютона*

Описание метода аналогично эксперименту 4, только для оптимизации используется метод Ньютона.

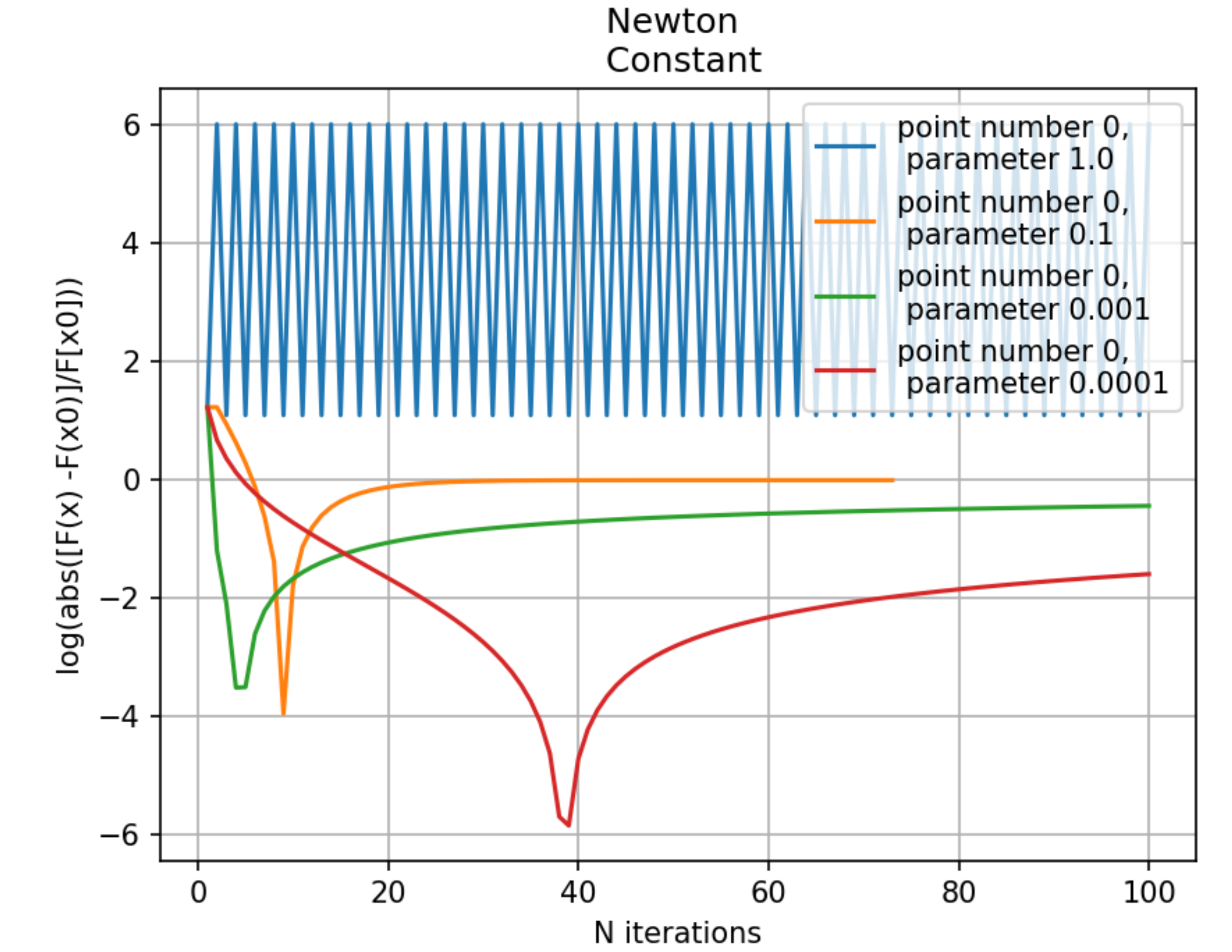

Рис. 23. Квадратичный оракул. Метод оптимизации: Ньютон; линейный поиск: постоянный шаг

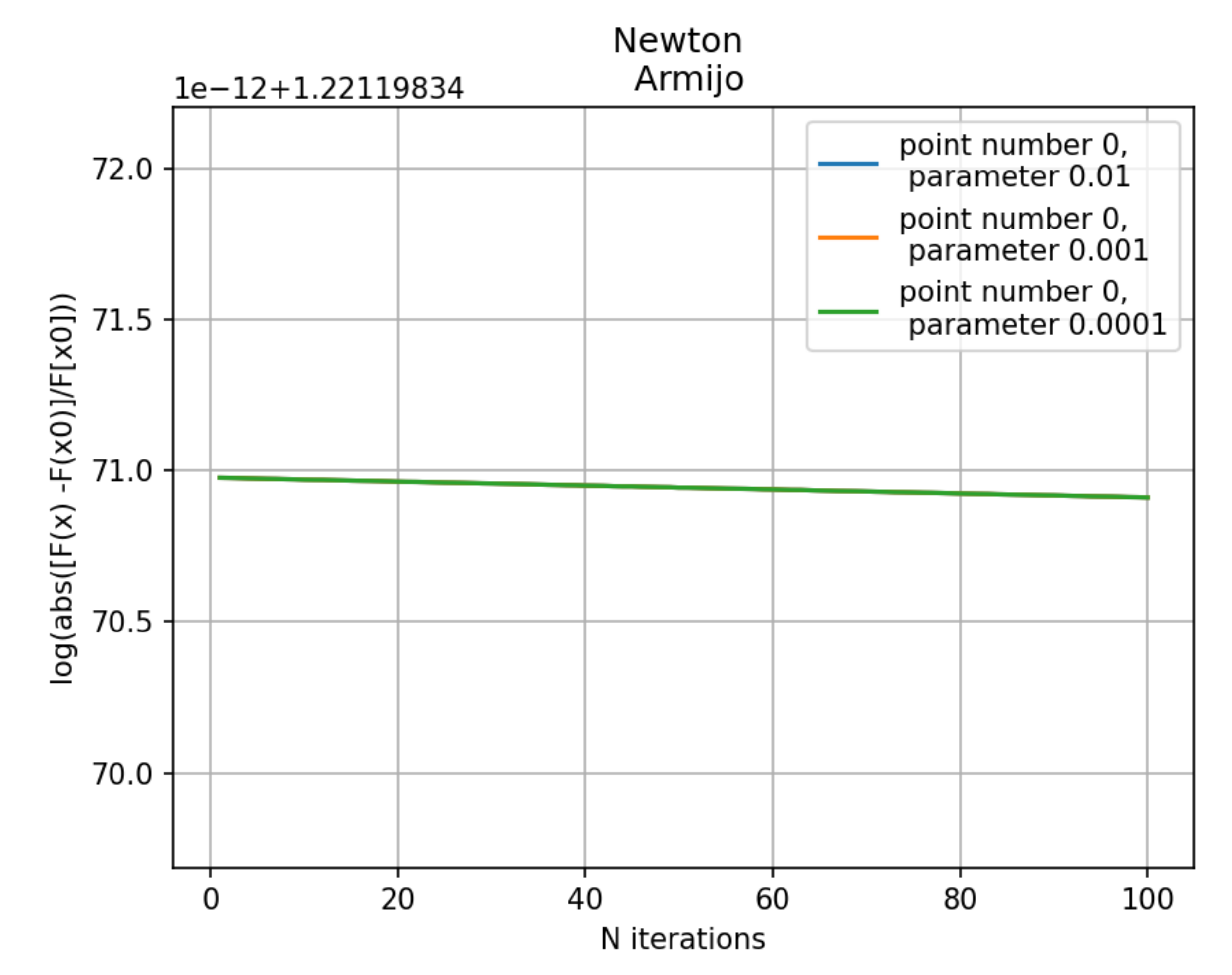

Рис. 24. Квадратичный оракул. Метод оптимизации: Ньютон; линейный поиск: Армихо

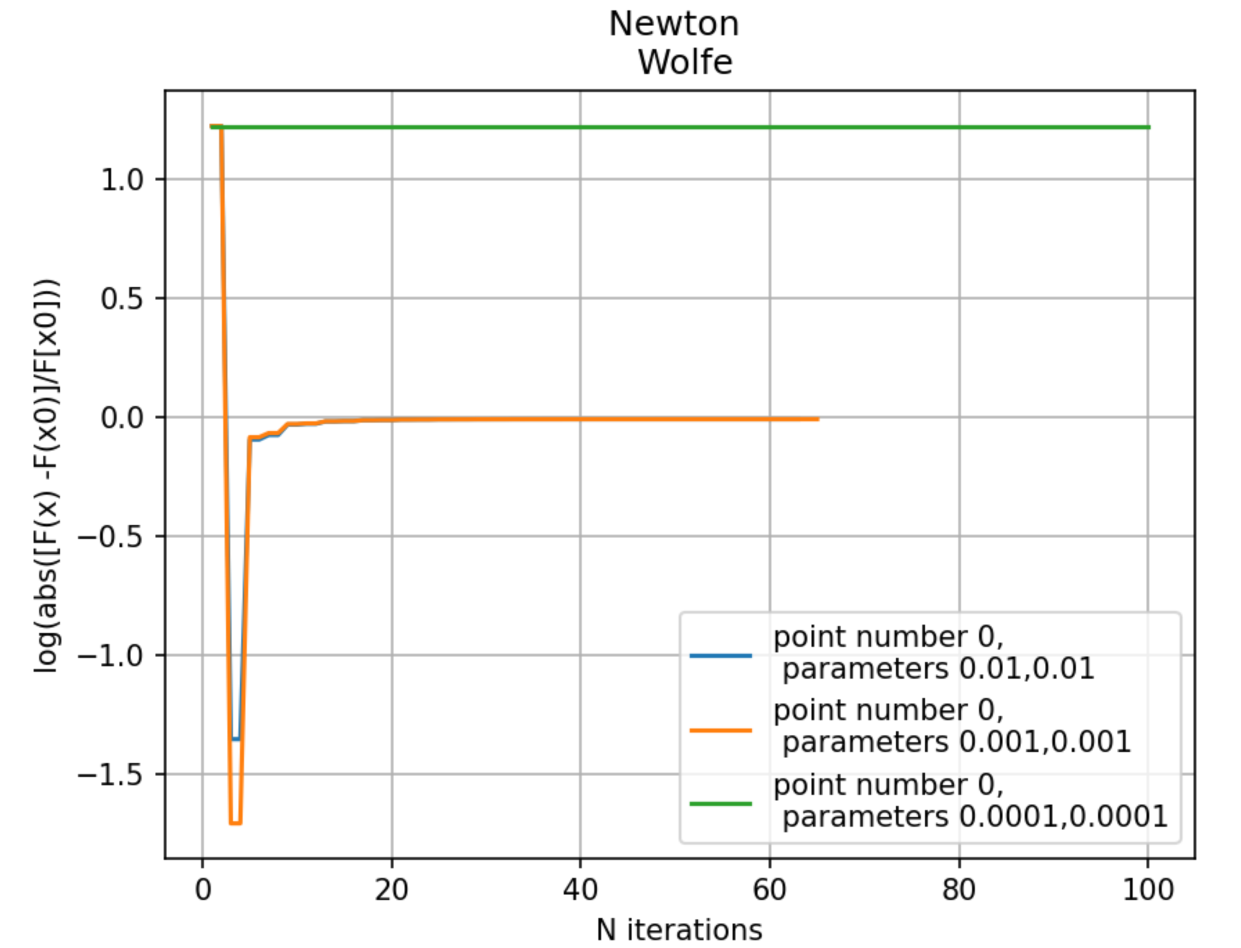

Рис. 25. Квадратичный оракул. Метод оптимизации: Ньютон; линейный поиск: Вульф

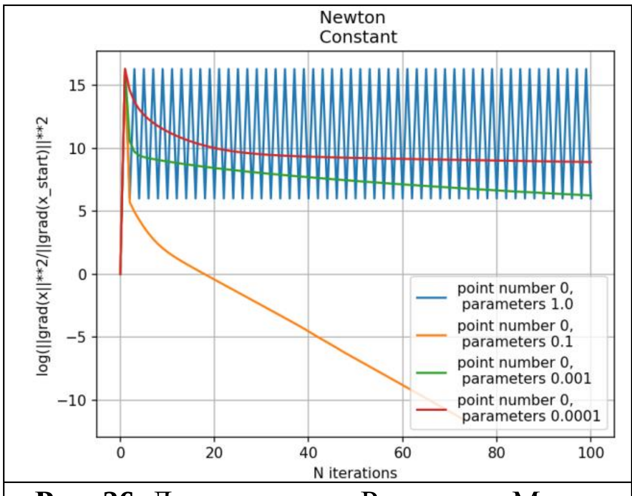

Рис. 26. Логистическая Регрессия. Метод оптимизации: Ньютон; линейный поиск: с постоянным шагом

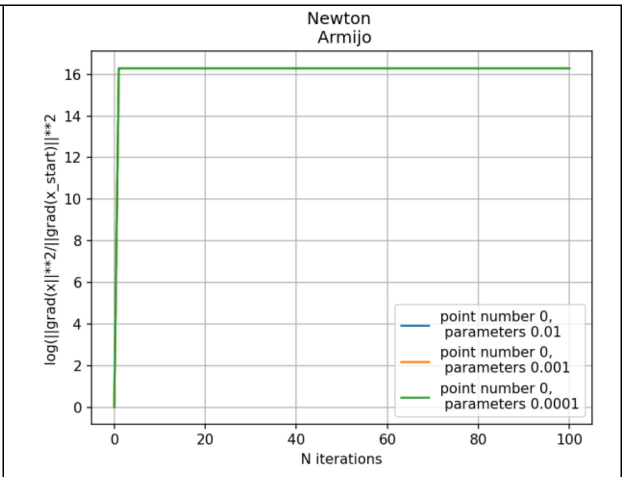

Рис. 27. Логистическая Регрессия. Метод оптимизации: Ньютон; линейный поиск: Армихо

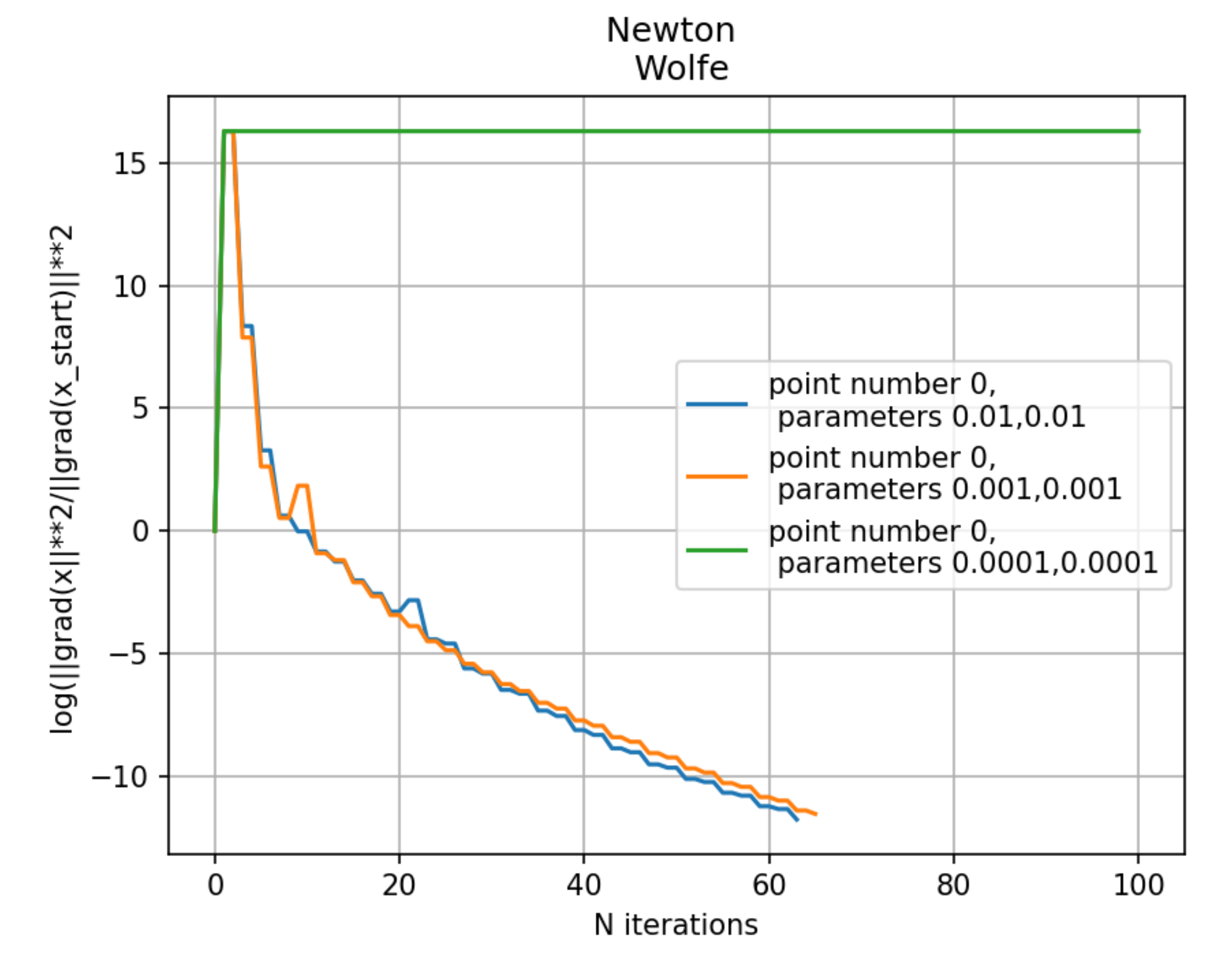

Рис. 28. Логистическая Регрессия. Метод оптимизации: Ньютон; линейный поиск: Вульф

При рассмотрении метода Ньютона результаты для логистической регресии и для квадратичного оракула примерно одинаковые, в смысле,что лучшие результаты у одних и тех же алгоритмов линейного поиска. Для квадратичного оракула – поиск с постонным шагом, а для логистической регрессии – метод Вульфа.

Сравнивая результаты, метод градиентого спуска с использованием сильных условий Вульфа, кажется интереснее для логистической регрессии, так как требуетс меньше число итераций.