Санкт–Петербургский государственный университет

Самедов Нихад Рагим оглы

Выпускная квалификационная работа

Численное сравнение алгоритма наискорейшего градиентного спуска с регуляризованной версией Иусема и Свайтера

Уровень образования: бакалавриат
Направление 02.03.02 «Фундаментальная информатика и информационные технологии»
Основная образовательная программа СВ.5003.2017 «Программирование и информационные технологии»
Профиль «Автоматизация научных исследований»

Научный руководитель:
профессор, кафедра математического моделирования энергетических систем,
д.ф. - м.н. Крылатов Александр Юрьевич

Санкт-Петербург
2021 г.

Содержание

Введение

При решении конкретной задачи оптимизации необходимо прежде всего выбрать математический метод, который приводил бы к конечным результатам с наименьшими затратами на вычисления или же давал возможность получить наибольший объем информации об искомом решении. Выбор того или иного метода в значительной степени определяется постановкой оптимальной задачи, а также используемой математической моделью объекта оптимизации.

Выбор наилучшего метода под задачу достаточно существенен. Для этого рассмотрим два градиентных метода и исследуем их на эффективность. Градиентные методы используются во многих алгоритмах, где требуется найти экстремум функции — нейронные сети, SVM, k-средних, регрессии.

Постановка задачи

При помощи численных экспериментов на тестовых примерах сравнить классический метод градиентного спуска с его регулуризованной версией, предложенной Иусемом и Свайтером.

Обзор литературы

В [3] рассмотрены методы оптимизации: градиентные и неградиентные методы. Рассмотрены такие алгоритмы как: алгоритм наискорейшего спуска, метод вторых производных (метод Ньютона), прямой поиск, поиск по деформируемому многограннику и остальные. В целом, данная литература способствует глубокому пониманию работ оптимизационных алгоритмов. В [2] содержатся важные понятия и определения по методам оптимизации для краткого ознакомления, включая конкретные примеры и пошаговое выполнение градиентных, и не только, алгоритмов. Приведен необходимый минимум теоретических сведений, подробно разобраны примеры, что впоследствии помогло при написании программной реализации алгоритмов. В [4] от Университета Карнеги наглядно проиллюстрирован метод наискорейшего спуска с подробными доказательствами вспомогательных теорем, что помогло улучшить теоретическую часть работы. В [1] описывается сам модифицированный регуляризованный алгоритм наискорейшего спуска, предложенный Иусемом и Свайтером, который и сравнивается в данной выпускной квалификационной работе. В публикации подробно рассмотрен алгоритм, а также даны теоретические обоснования вспомогательных теорем и лемм.

Для реализации программной части были использованы документации библиотек на языке программирования Python. [5], [6] были использованы для написания вычислительной части программы. Документация [7] включает в себя библиотеку для отображения графических элементов на экране, с помощью которого и были получены изображения работ алгоритмов. Документации [8], [9] и [10] добавляют поддержку многомерных массивов и матриц вместе с большой библиотекой высокоуровневых и достаточно быстрых математических функций. В основном данные библиотеки были использованы для численных преобразований. В [11], [12], [13] представлены тестовые функции для задачи оптимизации. На данных функциях были проведены численные эксперименты для алгоритмов классического и регуляризованного спусков.

Глава 1. Математическая модель

4.1 Описание

Выпишем разложение функции f в окрестности точки x_k в ряд Тейлора [2]:

$$f(x) = f(x^k) + \langle f'(x^k), x - x^k \rangle + o(x - x^k), \tag{1}$$

где $f'(x^k)$ – градиент функции f в точке x^k, а

$$\lim_{x \to x^k} \frac{o(x - x^k)}{\|x - x^k\|} = 0. \tag{2}$$

Ясно, что при достаточно малом $\|x - x^k\|$ значение разности $f(x) - f(x^k)$ определяется главной частью приращения $\langle f'(x^k), x - x^k \rangle$. С другой стороны, скалярное произведение двух векторов принимает наименьшее значение тогда, когда векторы разнонаправлены. Поэтому при $x - x^k = -\alpha f'(x^k)$, где $\alpha > 0$ – достаточно малое положительное число, получим

$$f(x) - f(x^k) \approx \langle f'(x^k), x - x^k \rangle = \langle f'(x^k), -\alpha f'(x^k) \rangle = -\alpha \|f'(x^k)\|^2 < 0, \tag{3}$$

Поэтому приняв $x^{k+1} = x^k - \alpha f'(x^k)$, при том же самом α, получим $f(x^{k+1}) < f(x^k)$ и сможем таким образом построить последовательность $\{x^k\}$, на которой значения нашей функции будут убывать. Число α представляет собой длину шага, который необходимо сделать в направлении антиградиента.

Существует несколько варинатов этого метода, отличающиеся способом выбора длины шага.

4.2 Классический метод наискорейшего градиентного спуска

Можно искать оптимальную длину шага, то есть α_k, доставляющее минимум функции в направлении антиградиента [2]:

$$f(x^k - \alpha_k f'(x^k)) = \min_{\alpha \geq 0} f(x^k - \alpha f'(x^k)). \tag{4}$$

По необходимому условию экстремума оптимальное α должно удовлетворять

$$\frac{d}{d\alpha} f(x^k - \alpha f'(x^k)) = 0, \tag{5}$$

то есть

$$\langle -f'(x^k), f'(x^k - \alpha f'(x^k)) \rangle = 0, \tag{6}$$

Определив отсюда α_k, получим:

$$\langle -f'(x^k), f'(x^{k+1}) \rangle = 0, \tag{7}$$

то есть направления, вдоль которых ведется минимизация в методе наискорейшего градиентного спуска на соседних шагах ортогональны.

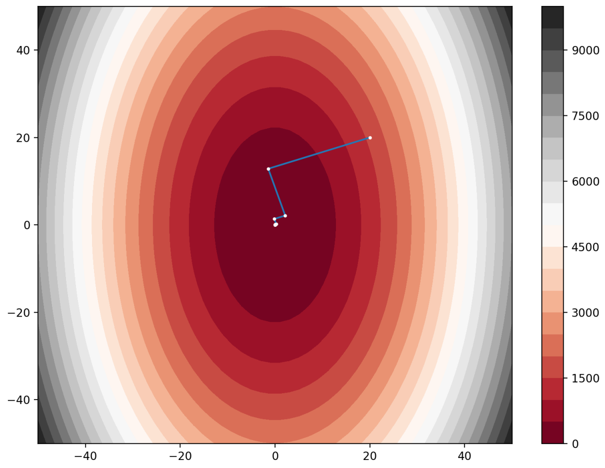


Рис. 1: Применение метода наискорейшего спуска на функции сферы $f(x) = \sum_{i=1}^{n} x_i^2$

4.3 Регуляризованный метод наискорейшего спуска Иусема и Свайтера

Данный метод в основном заключается в модификации способа выбора α, то есть длины шага [1]. Находится α в данном методе следующим образом:

$$a_k = \arg\min_{\alpha \geq 0} \{ f(x^k - \alpha \nabla f(x^k)) + \alpha^2 \lambda_k \| \nabla f(x^k) \|^2 \}, \tag{8}$$

где $\| \cdot \|$ евклидова норма и множество $\{\lambda_k\} \subset \mathbb{R}$ удовлетворяет:

$$\hat{\lambda} \leq \lambda_k \leq \tilde{\lambda}, \tag{9}$$

для некоторых $\hat{\lambda}$, $\tilde{\lambda}$ так, что $0 < \hat{\lambda} \leq \tilde{\lambda}$

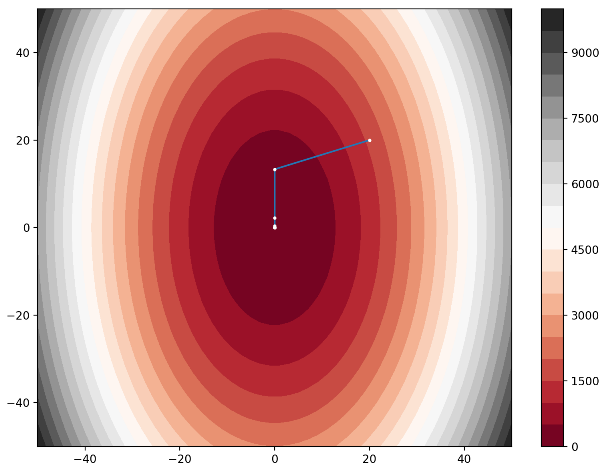


Рис. 2: Применение модифицированного регуляризованного метода

Глава 2. Алгоритм

5.1 Алгоритм классического метода наискорейшего градиентного спуска

1. Задаём начальное приближение x^0 и точность расчёта ε
2. Вычисляем градиент функции f в точке x^0: $\nabla f(x^0)$
3. Вычисляем длину α шага в направлении антиградиента:
   $\alpha_k = \arg\min\limits_{\alpha \geq 0} f(x^k - \alpha \nabla f(x^k))$
4. Вычисляем $x^{k+1} = x^k - \alpha^k \nabla f(x^k)$
   (a) Если $\|\nabla f(x^{k+1})\| > \varepsilon$ (можно выбрать и другое условие), то $k = k+1$ и переход к шагу 3
   (b) Иначе $x_{min} = x^{k+1}$ и останавливаемся

5.2 Алгоритм модифицированного регуляризованного градиентного спуска

1. Задаём начальное приближение x^0 и точность расчёта ε
2. Вычисляем градиент функции f в точке x^0: $\nabla f(x^0)$
3. Вычисляем длину α шага в направлении антиградиента **с добавочным параметром** $\lambda_k > 0$:
   $a_k = \arg\min\limits_{\alpha \geq 0} \{f(x^k - \alpha \nabla f(x^k)) + \alpha^2 \lambda_k \|\nabla f(x^k)\|^2\}$
4. Вычисляем $x^{k+1} = x^k - \alpha^k \nabla f(x^k)$
   (a) Если $\|\nabla f(x^{k+1})\| > \varepsilon$ (можно выбрать и другое условие), то $k = k+1$ и переход к шагу 3
   (b) Иначе $x_{min} = x^{k+1}$ и останавливаемся

Глава 3. Полученные результаты

6.1 Описание тестовых примеров

1. **Функция Леви и Монтавло:** $f(x) = \frac{\pi}{n}\Big(10\sin^2(\pi y_1) + \sum_{i=1}^{n-1}(y_i - 1)^2[1 + 10\sin^2(\pi y_{i+1})] + (y_n - 1)^2\Big), y_i = 1 + \frac{1}{4}(x_i + 1)$
   Глобальный минимум: $f(-1, -1, \ldots, -1) = 0$
   Метод поиска: $-10 \leq x_i \leq 10, i \in \{1, 2, \ldots, n\}$

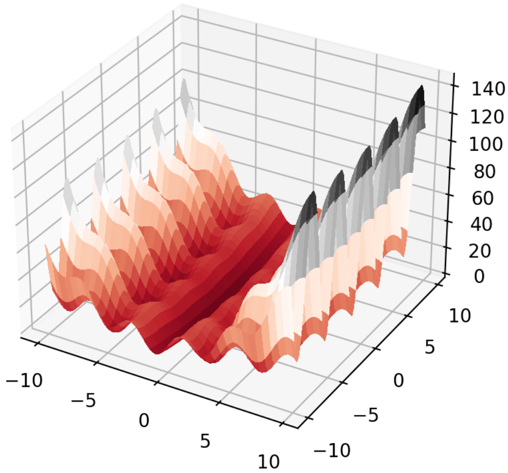


Рис. 3: Функция Леви и Монтавло

2. **Функция Ньюмара:** $f(x) = \sum_{i=1}^{n}(x_i - 1)^2 - \sum_{i=2}^{n} x_i x_{i-1}$
   Глобальный минимум: $f(x^*) = -\frac{n(n+4)(n-1)}{6}, x_i^* = i(n + 1 - i)$
   Метод поиска: $-n^2 \leq x_i \leq n^2, i \in \{1, 2, \ldots, n\}$

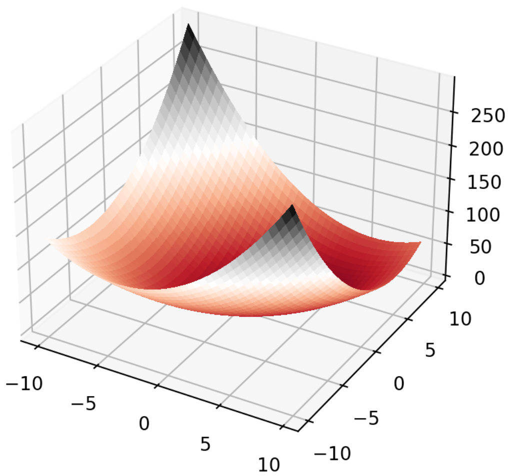


Рис. 4: Функция Ньюмара

3. **Функция Соломона:** $f(x) = 1 - \cos(2\pi\|x\|) + 0.1\|x\|, \|x\| = \sqrt{\sum_{i=1}^{n} x_i^2}$
   Глобальный минимум: $f(x^*) = 0, x_i^* = 0$
   Метод поиска: $-100 \leq x_i \leq 100, i \in \{1, 2, \ldots, n\}$

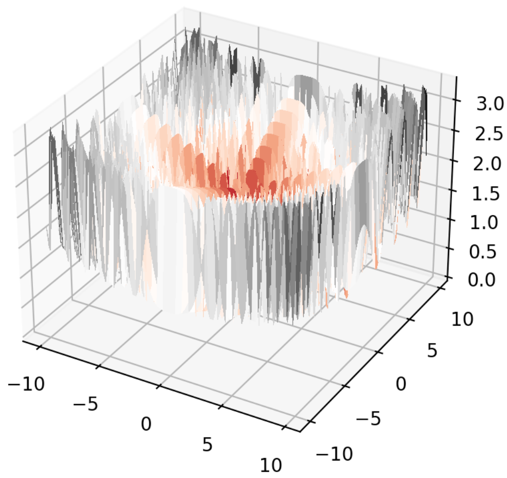


Рис. 5: Функция Соломона

4. **Функция Шефеля:** $f(x) = -\sum_{i=1}^{n} x_i \sin(\sqrt{|x_i|})$
   Глобальный минимум: $f(x^*) = -418.9829n, x_i^* \approx 420.97$
   Метод поиска: $-500 \leq x_i \leq 500, i \in \{1, 2, \ldots, n\}$

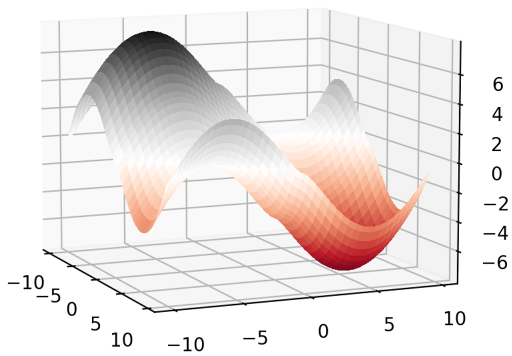


Рис. 6: Функция Шефеля

5. **Функция Шуберта:** $f(x) = \prod_{i=1}^{n}(\sum_{j=1}^{5} j\cos((j+1)x_i + j))$
   Глобальный минимум: $f^* \approx -186.7309$
   Метод поиска: $-10 \le x_i \le 10, i \in \{1, 2\}$

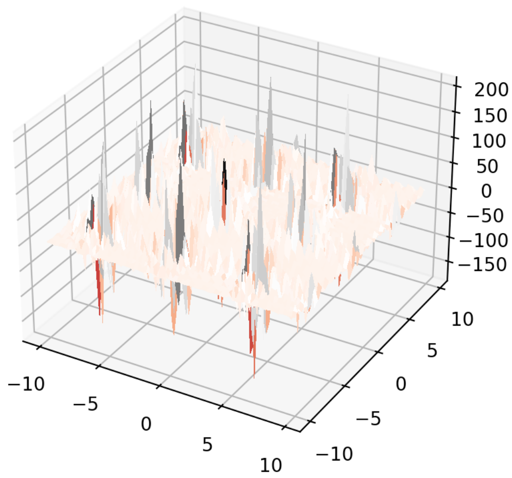


Рис. 7: Функция Шуберта

6. **Функция Griванка:** $f(x) = 1 + \frac{1}{4000}\sum_{i=1}^{n} x_1^2 - \prod_{i=1}^{n}\cos(\frac{x_i}{\sqrt{i}})$
   Глобальный минимум: $f(0, 0, \ldots, 0) = 0$
   Метод поиска: $-600 \le x_i \le 600, i \in \{1, 2, \ldots, n\}$

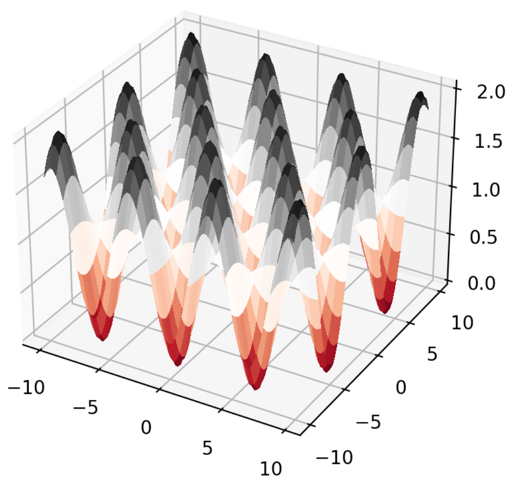


Рис. 8: Функция Гриванка

7. **Функция Экли:** $f(x,y) = -20\exp\big(-0.2\sqrt{0.5(x^2+y^2)}\big) - \exp\big(0.5(\cos 2\pi x + \cos 2\pi y)\big) + e + 20$
   Глобальный минимум: $f(0,0) = 0$
   Метод поиска: $-5 \leq x, y \leq 5$

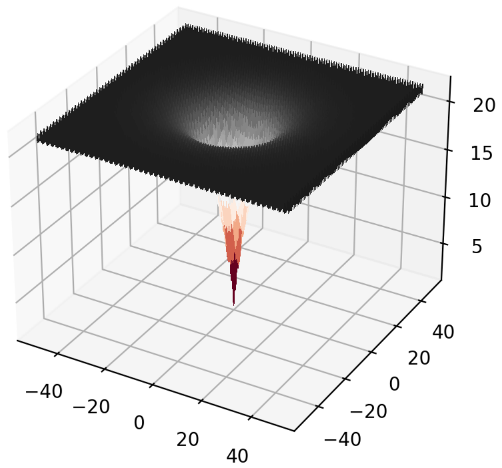


Рис. 9: Функция Экли

8. **Функция Сферы:** $f(x,y) = x^2 + y^2$
   Глобальный минимум: $f(0,0) = 0$
   Метод поиска: $-\infty \leq x, y \leq \infty$

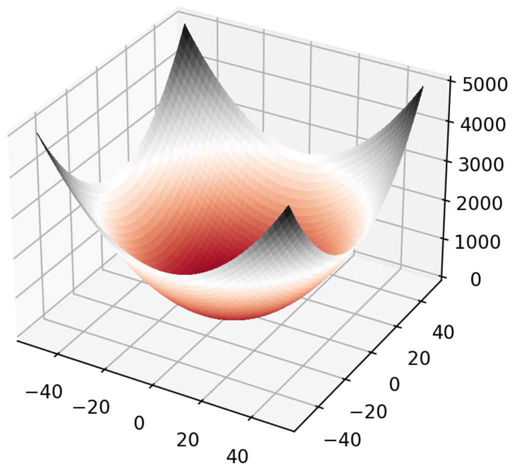


Рис. 10: Функция Сферы

9. **Функция Розенброка:** $f(x, y) = 100(y - x^2)^2 + (x - 1)^2$
   Глобальный минимум: $f(1, 1) = 0$
   Метод поиска: $-\infty \leq x, y \leq \infty$

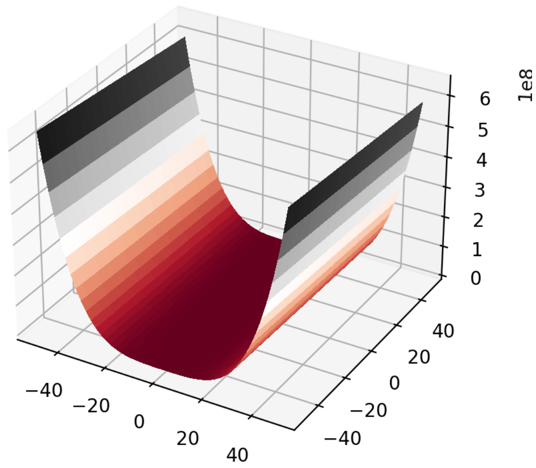


Рис. 11: Функция Розенброка

10. **Функция Била:** $f(x, y) = (1.5 - x + xy)^2 + (2.25 - x + xy^2)^2 + (2.625 - x + xy^3)^2$
    Глобальный минимум: $f(3, 0.5) = 0$
    Метод поиска: $-4.5 \leq x, y \leq 4.5$

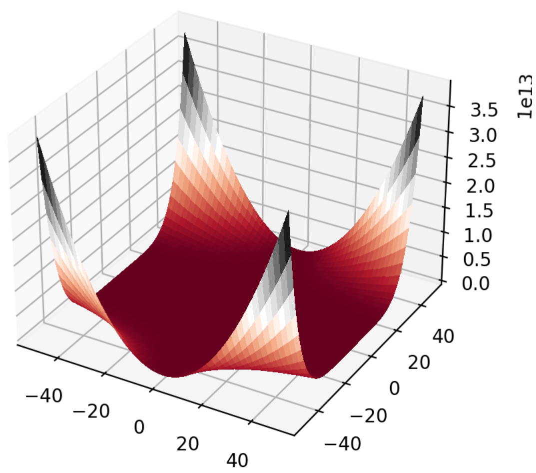


Рис. 12: Функция Била

11. **Функция Гольдман-Прайса:** $\Big(1 + (x + y + 1)^2(19 - 14x + 3x^2 - 14y + 6xy + 3y^2)\Big)\Big(30 + (2x - 3y)^2(18 - 32x + 12x^2 + 48y - 36xy + 27y^2)\Big)$
Глобальный минимум: $f(0, -1) = 3$
Метод поиска: $-2 \leq x, y \leq 2$

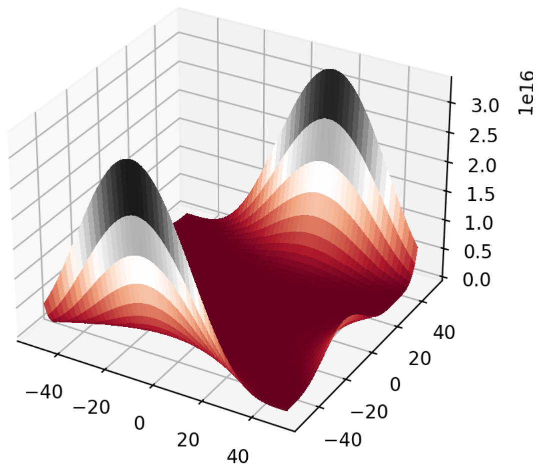


Рис. 13: Функция Гольдман-Прайса

12. **Функция Бута:** $f(x, y) = (x + 2y - 7)^2 + (2x + y - 5)^2$
Глобальный минимум: $f(1, 3) = 0$
Метод поиска: $-10 \leq x, y \leq 10$

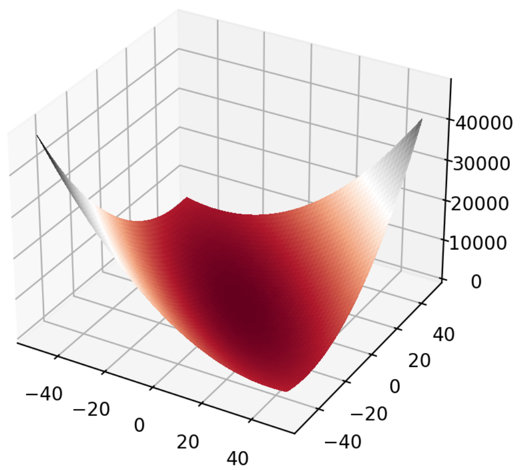


Рис. 14: Функция Бута

13. **Функция Букина №6:** $f(x,y) = 100\sqrt{|y - 0.01x^2|} + 0.01|x + 10|$
    Глобальный минимум: $f(-10, 1) = 0$
    Метод поиска: $-15 \le x \le -5, \quad -3 \le y \le 3$

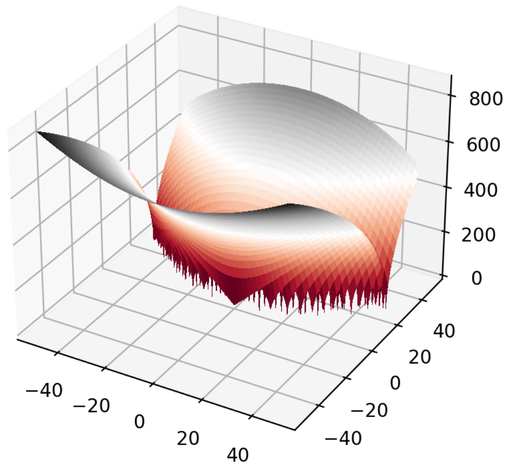


Рис. 15: Функция Букина №6

14. **Функция Матьяса:** $f(x,y) = 0.26(x^2 + y^2) - 0.48xy$
    Глобальный минимум: $f(0, 0) = 0$
    Метод поиска: $-10 \le x, y \le 10$

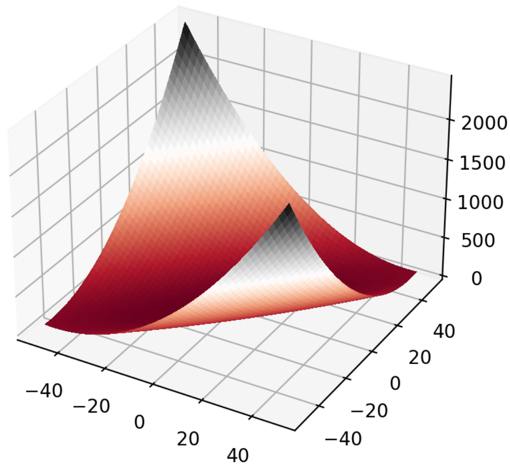


Рис. 16: Функция Матьяса

15. **Функция Леви №13:** $f(x, y) = \sin^2 3\pi x + (x-1)^2(1+\sin^2 3\pi y) + (y-1)^2(1+\sin^2 2\pi y)$

Глобальный минимум: $f(1, 1) = 0$

Метод поиска: $-10 \le x, y \le 10$

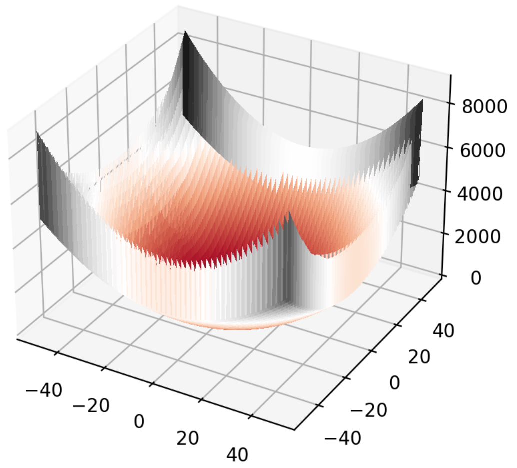


Рис. 17: Функция Леви №13

16. **Функция трехгорбого верблюда:** $f(x, y) = 2x^2 - 1.05x^4 + \frac{x^6}{6} + xy + y^2$

Глобальный минимум: $f(0, 0) = 0$

Метод поиска: $-5 \le x, y \le 5$

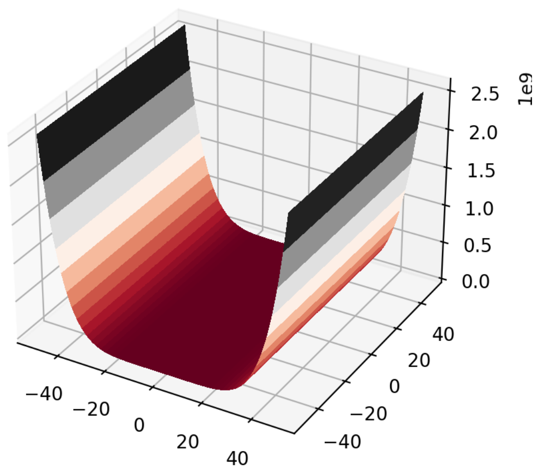


Рис. 18: Функция трехгорбого верблюда

17. **Функция Изома:** $f(x,y) = -\cos(x)\cos(y)\exp\Big(-\big((x-\pi)^2+(y-\pi)^2\big)\Big)$
Глобальный минимум: $f(\pi,\pi) = -1$
Метод поиска: $-100 \le x, y \le 100$

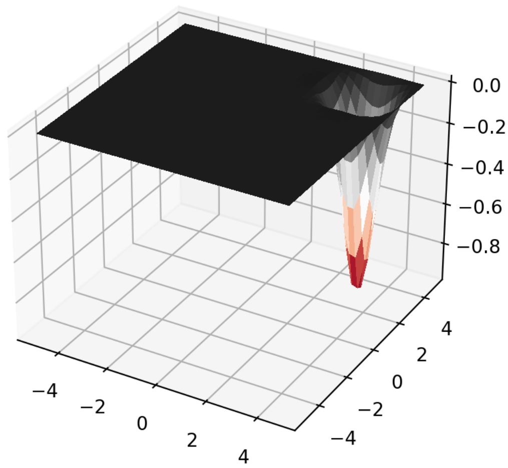


Рис. 19: Функция Изома

18. **Функция МакКормика:** $f(x,y) = \sin(x+y)+(x-y)^2-1.5x+2.5y+1$ **Глобальный минимум:** $f(-0.54719, -1.54719) = -1.9133$
Метод поиска: $-1.5 \le x \le 4, \quad -3 \le y \le 4$

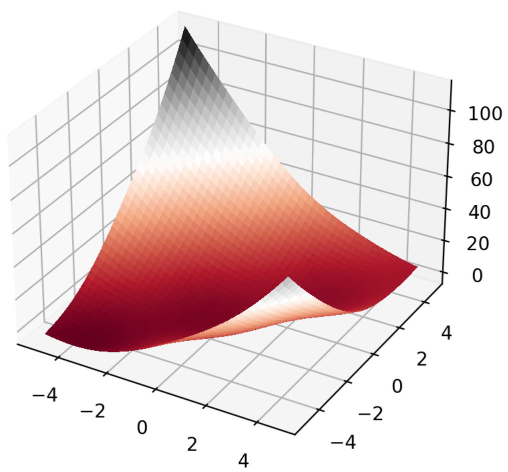


Рис. 20: Функция МакКормика

19. **Функция Шаффера №2:** $f(x,y) = 0.5 + \frac{sin^2(x^2-y^2)-0.5}{\left(1+0.001(x^2+y^2)\right)^2}$

    Глобальный минимум: $f(0,0) = 0$

    Метод поиска: $-100 \leq x, y \leq 100$

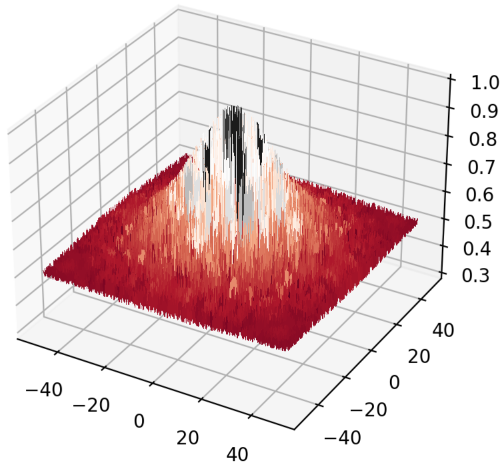


Рис. 21: Функция Шаффера №2

20. **Функция Шаффера №4:** $f(x,y) = 0.5 + \frac{\cos^2\left(\sin(|x^2-y^2|)\right)-0.5}{\left(1+0.001(x^2+y^2)\right)^2}$

    Глобальный минимум: $f(0, 1.25313) = 0.292579$

    Метод поиска: $-100 \leq x, y \leq 100$

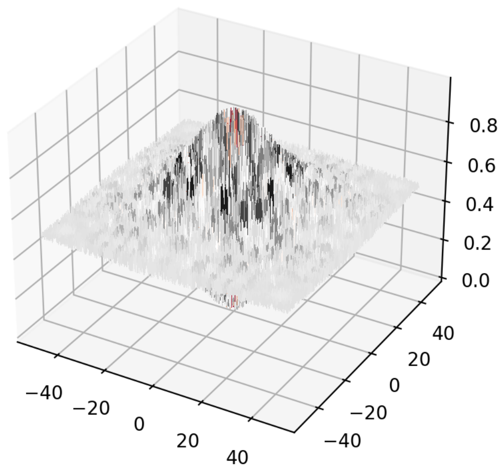


Рис. 22: Функция Шаффера №4

6.2 Программа, реализующая алгоритмы на языке Python

```
import numpy as np
from scipy.optimize import minimize
import numdifftools as nd
import time

np.set_printoptions(suppress=True)

def xa(a, x0, gr):
    return x0[0] - a * gr[0]

def ya(a, x0, gr):
    return x0[1] - a * gr[1]

def f(a, x0, gr, function):
    return function([xa(a, x0, gr), ya(a, x0, gr)])

def fr(a, x0, gr, function, l):
    return f(a, x0, gr, function) + a * a * l * np.inner(gr, gr)


def gradientDescent(function, x0=[0, 0], l = 1, regularized = False):
    startTime = time.time()
    x_history = []
    y_history = []
    a_history = []
    i_counter = 1
    x_k = x0
    x_history.append(x_k[0])
    y_history.append(x_k[1])
    gr = nd.Gradient(function)([x_k])
    a = minimize(fr, 0, args=(x_k, gr, function, l), method='Nelder-Mead'
    a_history.append(a)

    while True:
        x_k = [xa(a, x_k, gr), ya(a, x_k, gr)]
        x_history.append(x_k[0])
        y_history.append(x_k[1])
```

```
        gr = nd.Gradient(function)([x_k])
        a = minimize(fr, 0, args=(x_k, gr, function, l), method='Nelder-M
        a_history.append(a)
        i_counter += 1
        if np.linalg.norm(gr) < 1e-5:
            break
    endTime = time.time()
    timeElapsed = endTime - startTime
    return x_history, y_history, a_history, i_counter, timeElapsed
```

6.3 Результаты вычислений

1. **Лямбда** – параметр λ в регуляризованном методе
2. **Итераций** – количество итераций, за которые алгоритм находит минимум
3. **Время** – длительность работы алгоритма, выраженная в секундах
4. **(x_min y_min)** – найденный алгоритмом минимум (x_{min}, y_{min}) функции $f(x, y)$

Таблица 1: Результаты вычислений. Функция Леви и Монтавло

Функция Леви и Монтавло

Метод	Лямбда	Итераций	Время	(x_min, y_min)
Классический		5	0.034	(2.95943986, 2.95903229)
Регуляризованный	0.2	6	0.037	(2.95943987, 2.95943987)
Регуляризованный	0.5	7	0.034	(2.95943989, 2.95903234)
Регуляризованный	1	8	0.032	(2.95943994, 2.95903241)

Таблица 2: Результаты вычислений. Функция Ньюмара

Функция Ньюмара

Метод	Лямбда	Итераций	Время	(x_min, y_min)
Классический		2	0.009	(2, 2)
Регуляризованный	0.2	12	0.009	(2.00000209, 2.00000209)
Регуляризованный	0.5	20	0.011	(2.00000379, 2.00000379)
Регуляризованный	1	32	0.008	(2.00000694, 2.00000694)

Таблица 3: Результаты вычислений. Функция Соломона

Функция Соломона

Метод	Лямбда	Итераций	Время	(x_min, y_min)
Классический		3	0.013	(4.24084949, 4.24084949)
Регуляризованный	0.2	5	0.013	(4.24084949, 4.24084949)
Регуляризованный	0.5	5	0.013	(4.24084949, 4.24084949)
Регуляризованный	1	6	0.014	(4.24084949, 4.24084949)

Таблица 4: Результаты вычислений. Функция Шефеля

Функция Шефеля

Метод	Лямбда	Итераций	Время	(x_min, y_min)
Классический		2	0.009	(5.2391993, 5.2391993)
Регуляризованный	0.2	18	0.012	(5.23919061, 5.23919061)
Регуляризованный	0.5	34	0.012	(5.23918225, 5.23918225)
Регуляризованный	1	62	0.01	(5.23918264, 5.23918264)

Таблица 5: Результаты вычислений. Функция Шуберта

Функция Шуберта

Метод	Лямбда	Итераций	Время	(x_min, y_min)
Классический		3	0.02	(3.99024797, 3.99024797)
Регуляризованный	0.2	3	0.02	(3.99024797, 3.99024797)
Регуляризованный	0.5	4	0.025	(3.99024797, 3.99024797)
Регуляризованный	1	4	0.022	(3.99024797, 3.99024797)

Таблица 6: Результаты вычислений. Функция Гриванка

Функция Гриванка

Метод	Лямбда	Итераций	Время	(x_min, y_min)
Классический		7	0.038	(3.14002968, 4.43844174)
Регуляризованный	0.2	15	0.039	(3.14002263, 4.43843189)
Регуляризованный	0.5	27	0.039	(3.14002263, 4.43842461)
Регуляризованный	1	48	0.039	(3.14002263, 4.43842696)

Таблица 7: Результаты вычислений. Функция Экли

Функция Экли

Метод	Лямбда	Итераций	Время	(x_min, y_min)
Классический		222	1.477	(3.92904642, 3.79214463)
Регуляризованный	0.2	264	1.584	(3.92904707, 3.792144)
Регуляризованный	0.5	267	1.603	(3.92904712, 3.79214366)
Регуляризованный	1	253	1.513	(3.92904677, 3.79214402)

Таблица 8: Результаты вычислений. Функция Розенброка

Функция Розенброка

Метод	Лямбда	Итераций	Время	(x_min, y_min)
Классический		22641	93.45	(1.00000815, 1.00001632)
Регуляризованный	0.2	22425	96.596	(1.00000791, 1.00001587)
Регуляризованный	0.5	5217	23.951	(1.00001066, 1.00002138)
Регуляризованный	0.4	10561	47.82	(1.00000999, 1.00002002)
Регуляризованный	0.6	14149	63.605	(1.00000899, 1.00001804)
Регуляризованный	1	17177	80	(1.00000834, 1.00001674)

Таблица 9: Результаты вычислений. Функция Била

Функция Била

Метод	Лямбда	Итераций	Время	(x_min, y_min)
Классический		604	3.325	(2.99997355, 0.49999352)
Регуляризованный	0.2	307	1.783	(2.99997355, 0.49999352)
Регуляризованный	0.5	646	3.948	(2.99997642, 0.49999397)
Регуляризованный	1	643	3.741	(2.99997664, 0.49999403)

Таблица 10: Результаты вычислений. Функция сферы

Функция сферы

Метод	Лямбда	Итераций	Время	(x_min, y_min)
Классический		2	0.01	(0, 0)
Регуляризованный	0.2	9	0.043	(0.00000238, 0.00000238)
Регуляризованный	0.5	14	0.074	(0.00000251, 0.00000251)
Регуляризованный	1	22	0.1	(0.00000191, 0.00000191)

Таблица 11: Результаты вычислений. Функция Гольдман-Прайса

Функция Гольдман-Прайса

Метод	Лямбда	Итераций	Время	(x_min, y_min)
Классический		0	0	Не сходится
Регуляризованный	0.2	0	0	Не сходится
Регуляризованный	0.5	0	0	Не сходится
Регуляризованный	1	0	0	Не сходится

Таблица 12: Результаты вычислений. Функция Бута

Функция Бута

Метод	Лямбда	Итераций	Время	(x_min, y_min)
Классический		10	0.54	(1.00000018, 2.99999981)
Регуляризованный	0.2	42	0.237	(1.00000323, 2.99999713)
Регуляризованный	0.5	59	0.352	(1.00000234, 2.99999717)
Регуляризованный	1	58	0.343	(1.00000275, 2.99999769)

Таблица 13: Результаты вычислений. Функция Букина

Функция Букина №6

Метод	Лямбда	Итераций	Время	(x_min, y_min)
Классический		0	0	Не сходится
Регуляризованный	0.2	0	0	Не сходится
Регуляризованный	0.5	0	0	Не сходится
Регуляризованный	1	0	0	Не сходится

Таблица 14: Результаты вычислений. Функция Матьяса

Функция Матьяса

Метод	Лямбда	Итераций	Время	(x_min, y_min)
Классический		2	0.011	(0, 0)
Регуляризованный	0.2	111	0.65	(0.00015488, 0.00015866)
Регуляризованный	0.5	257	1.373	(0.00017439, 0.00017439)
Регуляризованный	1	508	2.624	(0.00017449, 0.00017449)

Таблица 15: Результаты вычислений. Функция Леви №13

Функция Леви №13

Метод	Лямбда	Итераций	Время	(x_min, y_min)
Классический		109	0.565	(3.96388716, 3.9973575)
Регуляризованный	0.2	109	0.574	(3.96388716, 3.9973575)
Регуляризованный	0.5	109	0.581	(3.96388716, 3.9973575)
Регуляризованный	1	107	0.541	(3.96388717, 3.9973575)
Регуляризованный	1.1	105	0.54	(3.96388717, 3.9973575)
Регуляризованный	1.5	109	0.585	(3.96388716, 3.9973575)

Таблица 16: Результаты вычислений. Функция трёхгорбого верблюда

Функция трёхгорбого верблюда

Метод	Лямбда	Итераций	Время	(x_min, y_min)
Классический		10	0.594	(1.74755245, 0.87377753)
Регуляризованный	0.2	42	0.267	(-1.74755233, 0.8737799)
Регуляризованный	0.5	32	0.187	(-1.74755309, 0.8737795)
Регуляризованный	0.9	23	0.136	(-0.00000233, 0.00000563)
Регуляризованный	0.7	40	0.238	(-1.74755165, 0.873773)
Регуляризованный	1	25	0.157	(-0.00000197, 0.00000476)

Таблица 17: Результаты вычислений. Функция Изома

Функция Изома

Метод	Лямбда	Итераций	Время	(x_min, y_min)
Классический		2	0.011	(3.14159268, 3.14159268)
Регуляризованный	0.2	7	0.041	(3.14159499, 3.14159499)
Регуляризованный	0.5	11	0.071	(3.14159359, 3.14159359)
Регуляризованный	1	16	0.09	(3.14159412, 3.14159412)

Таблица 18: Результаты вычислений. Функция МакКормика

Функция МакКормика

Метод	Лямбда	Итераций	Время	(x_min, y_min)
Классический		10	0.492	(2.59439571, 1.59439563)
Регуляризованный	0.2	16	0.086	(2.59439629, 1.59439727)
Регуляризованный	0.5	18	0.103	(2.59439665, 1.59439702)
Регуляризованный	1	24	0.129	(2.59439829, 1.59439829)

Таблица 19: Результаты вычислений. Функция Шаффера №2

Функция Шаффера №2

Метод	Лямбда	Итераций	Время	(x_min, y_min)
Классический		2	0.172	(250096.543, 250096.543)
Регуляризованный	0.2	0	0	Не сходится
Регуляризованный	0.5	0	0	Не сходится
Регуляризованный	1	0	0	Не сходится

Таблица 20: Результаты вычислений. Функция Шаффера №4

Функция Шаффера №4

Метод	Лямбда	Итераций	Время	(x_min, y_min)
Классический		2	0.013	(0.00000007, 0.00000007)
Регуляризованный	0.2	1421	9.48	(0.00352597, 0.00352597)
Регуляризованный	0.5	3545	24.956	(0.00353137, 0.00353137)
Регуляризованный	1	7085	45.324	(0.00353372, 0.00353372)

Выводы

В функциях Розенброка и Била регуляризованный метод показал результаты значительно лучше, чем классический. В функции Розенброка регулуризованный метод сошелся за 5217 итераций при $\lambda = 0.5$ за 23.951 секунд, в то время как классический за 22425 итераций за 96.596 секунд (Таблица 8), то есть удалось в 4 раза сократить скорость нахождения минимума. В функции Била классический метод нашел ответ за 604 итераций за 3.325 секунд, однако регуляризованный метод сошелся за 307 итераций при $\lambda = 0.2$ за 1.783 секунд (Таблица 9). Тем не менее в остальных случаях настройка параметра λ привела к медленной работе алгоритма, а с некоторыми функциями алгоритм вовсе не сошелся.

Заключение

В данной работе было рассмотрено два алгоритма; был проведен вычислительный эксперимент, включающий в себя численное сравнение двух алгоритмов. На большинстве функций классический метод сходился быстрее, чем регуляризованный. Стоит отметить, однако, что на нескольких функциях регуляризованный метод сошелся значительно быстрее, чем классический.

Список литературы

[1] A. N. Iusem B.F.Svaiter A proximal regularization of the steepest descent method, 1995г. URL: http://www.numdam.org/article/RO_1995__29_2_123_0.pdf

[2] М. Э. Аббасов Методы оптимизации, 2014г. URL: http://www.apmath.spbu.ru/ru/staff/abbasov_m_e/publ/publ5.pdf

[3] Химмельблау Д.М. Прикладное нелинейное программирование, 1972г.

[4] Ryan Tibshirani Convex Optimization Lecture 5: Gradient Descent, 2015г. URL: https://www.stat.cmu.edu/~ryantibs/convexopt-S15/scribes/05-grad-descent-scribed.pdf

[5] Programming for Computations – Python: A Gentle Introduction to Numerical Simulations with Python, by Svein Linge, Hans Petter Langtangen, 2016г.

[6] Introduction to Computation and Programming Using Python, Second Edition, by John V. Guttag, Julie Sussman, 2016г.

[7] Документация matplotlib.pyplot URL: https://matplotlib.org/2.0.2/api/pyplot_api.html
(дата обращения: 01.04.21)

[8] Документация NumPy URL: https://numpy.org/doc/stable/reference/
(дата обращения: 02.04.21)

[9] Документация scipy.optimize URL: https://docs.scipy.org/doc/scipy/reference/optimize.html
(дата обращения: 02.04.21)

[10] Документация numdifftools URL: https://numdifftools.readthedocs.io/en/latest/
(дата обращения: 03.04.21)

[11] Тестовые функции URL: https://ru.wikipedia.org/wiki/%D0%A2%D0%B5%D1%81%D1%82%D0%BE%D0%B2%D1%8B%D0%B5_%D1%84%D1%83%D0%BD%D0%BA%D1%86%D0%B8%D0%B8_%D0%B4%D0%BB%D1%8F_%D0%BE%D0%BF%D1%82%D0%B8%D0%BC%D0%B8%D0%B7%D0%B0%D1%86%D0%B8%D0%B8
(дата обращения: 03.04.21)

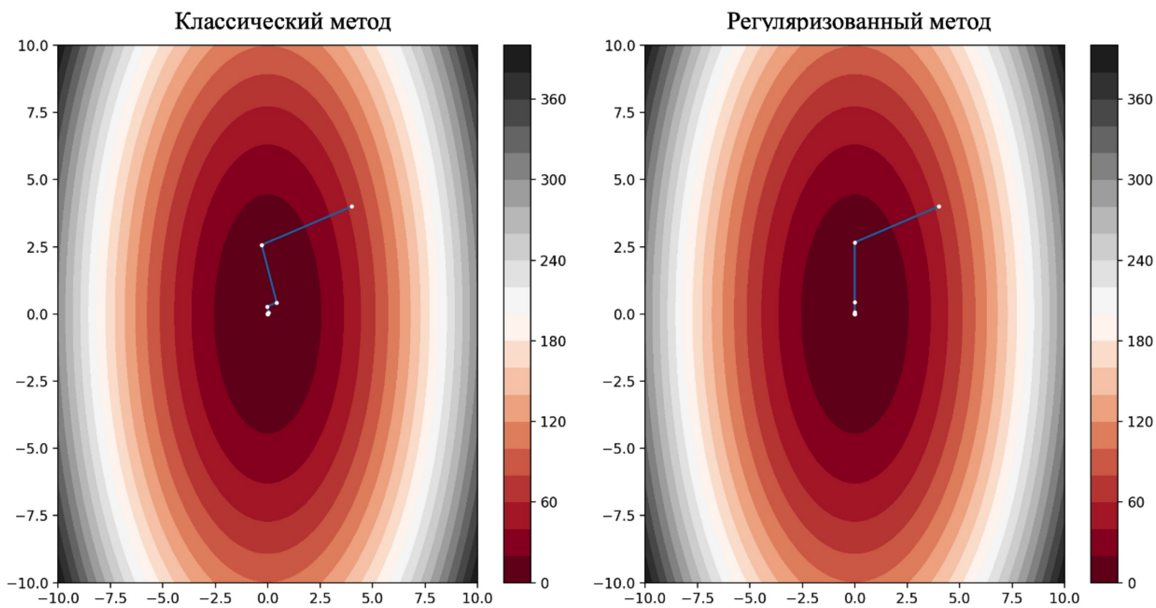
[12] Тестовые функции URL: https://raw.githubusercontent.com/Harrix/HarrixTestFunctions/master/_HarrixTestFunctions.pdf
(дата обращения: 03.04.21)

[13] Тестовые функции URL: http://hpc-education.unn.ru/ru/globopt/%D1%82%D0%B5%D1%81%D1%82%D0%BE%D0%B2%D1%8B%D0%B5-%D0%B7%D0%B0%D0%B4%D0%B0%D1%87%D0%B8-%D0%B3%D0%BB%D0%BE%D0%B1%D0%B0%D0%BB%D1%8C%D0%BD%D0%BE%D0%B9-%D0%BE%D0%BF%D1%82%D0%B8%D0%BC%D0%B8%D0%B7%D0%B0
(дата обращения: 03.04.21)

ПРИЛОЖЕНИЕ А. GitHub репозиторий программы

https://github.com/N-ihad/gradient-descent/

Численное сравнение классического градиентного спуска с его регуляризованной версией



Запуск

Для запуска программы необходимо прописать в терминале (находясь внутри директории проекта):

```
python -m pip install -r requirements.txt
```

Затем:

```
python run.py
```

Описание входных данных

В файле run.py необходимо указать следующие параметры:
x0 — начальная точка
l — параметр лямбда
a, b — (x, y) интервалы для отрисовки графика
f — функция на которой тестируются алгоритмы

Рис. 23: README. Описание репозитория