

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
РОССИЙСКИЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ

**А.Б. СОРОКИН, О.В. ПЛАТОНОВА,
Л.М. ЖЕЛЕЗНЯК**

БЕЗУСЛОВНАЯ ОПТИМИЗАЦИЯ
УЧЕБНО-МЕТОДИЧЕСКОЕ ПОСОБИЕ

Москва — 2020

УДК 519.852.3
ББК 22.18
С 65

Сорокин А.Б. Безусловная оптимизация [Электронный ресурс]: Учебно-методическое пособие / А.Б. Сорокин, О.В. Платонова, Л.М. Железняк — М.: РТУ МИРЭА, 2020. — 1 электрон. опт. диск (CD-ROM).

Учебно-методическое пособие содержит теоретические основы описание методов и реализация алгоритмов безусловной минимизации функций многих переменных. Предназначено для студентов 3-го курса квалификации бакалавр, обучающихся по направлению 09.03.04 «Программная инженерия» по профилю «Системы поддержки принятия решений».

Учебно-методическое пособие издается в авторской редакции.

Авторский коллектив: Сорокин Алексей Борисович, Платонова Ольга Владимировна, Железняк Лилия Михайловна.

Рецензенты:

Коваленко Сергей Михайлович, к.т.н., доцент, профессор кафедры вычислительной техники, института ИТ, МИРЭА – Российский технологический университет

Варов Евгений Борисович, к.т.н., доцент, зам. зав. кафедрой разведочной геофизики и компьютерных систем по учебной работе, Российский Государственный Университет Нефти и Газа им. И.М. Губкина

Минимальные системные требования:

Наличие операционной системы Windows, поддерживаемой производителем.

Наличие свободного места в оперативной памяти не менее 128 Мб.

Наличие свободного места в памяти хранения (на жестком диске) не менее 30 Мб.

Наличие интерфейса ввода информации.

Дополнительные программные средства: программа для чтения pdf-файлов (Adobe Reader).

Подписано к использованию по решению Редакционно-издательского совета

Московского технологического университета от _____ 2020 г.

Тираж 10

© Сорокин А.Б., 2020

© Российский технологический
университет (МИРЭА), 2020

Оглавление

Введение.....	5
Постановка задачи.....	6
Глава 1. Методы нулевого порядка.....	7
1.1. Симплексный метод.....	7
1.1.1. Практический расчет задачи	11
1.1.2. Программная реализация	24
1.1.3. Контрольные вопросы	34
1.2. Метод Нелдера-Мида.....	35
1.2.1. Практический расчет задачи	38
1.2.2. Программная реализация	44
1.2.3. Контрольные вопросы	50
1.3. Метод Хука-Дживса.....	51
1.3.1. Практический расчет задачи	54
1.3.2. Программная реализация	59
1.3.3. Контрольные вопросы	63
Глава 2. Методы первого порядка.....	64
2.1. Метод градиентного спуска с постоянным шагом.....	64
2.1.1. Практический расчет задачи	66
2.1.2. Программная реализация	68
2.1.3. Контрольные вопросы	74
2.2. Метод наискорейшего градиентного спуска.....	75
2.2.1. Практический расчет задачи	78
2.2.2. Программная реализация	79
2.2.3. Контрольные вопросы	81
2.3. Метод покоординатного спуска.....	82
2.3.1. Практический расчет задачи	83
2.3.2. Программная реализация	86
2.3.3. Контрольные вопросы	88
2.4. Метод Флетчера-Ривса.....	89
2.4.1. Практический расчет задачи	91
2.4.2. Программная реализация	93
2.4.3. Контрольные вопросы	97
Глава 3. Методы второго порядка.....	98

3.1. Метод Ньютона.	98
3.1.1. Практический расчет задачи	99
3.1.2. Программная реализация	102
3.1.3. Контрольные вопросы	104
3.2. Метод Ньютона-Рафсона.....	105
3.2.1. Практический расчет задачи	106
3.2.2. Программная реализация	110
3.2.3. Контрольные вопросы	113
Практические задания.....	114
Список литературы	115

Введение

Любое исследование основывается на применении научного метода, предоставляет научную информацию и теории для объяснения природы, и свойств окружающего мира. При это не существует неприступных границ – и подходы, и методы могут переплетаться в едином исследовании. При этом научные подходы нацеливают на научные разработки и открытия, а методы помогают совершать их.

Когда наш знаменитый соотечественник Пётр Леонидович Капица работал в Великобритании, с ним произошла забавная история. Получив звание доктора и вступив в профсоюз научных работников, он подписал договор, по которому не имел права бесплатно консультировать промышленных клиентов. Более того, вознаграждение за консультацию не должно было опускаться ниже определенной суммы, соответствующей ученому званию. Как-то раз одна из фирм попросила Пётра Леонидовича, чей авторитет был очень высок, помочь устранить вибрацию новой модели турбины. Осмотрев ротор разобранной турбины и провернув его, Капица ударил куда-то молотком. Когда турбину вновь собрали и запустили, она уже не вибрировала. Благодарные клиенты спросили, сколько стоит оказанная услуга. Ответ был предельно четок: «1000 фунтов». Довольные клиенты не возражали, но спросили, что им лучше написать в финансовом отчете. «Напишите так — сказал знаменитый физик. — Удар молотком — 1 фунт; знание того, куда ударить, — 999 фунтов». Понимающие толк в юморе англичане согласились...

Только благодаря надежным методам возникают новые плодотворные теории, а наша практическая деятельность становится более эффективной. Хороший специалист, вооруженный научной методологией, не только знает правильные ответы на сложные вчерашние вопросы и умеет самостоятельно находить грамотные ответы на сегодняшние, но и не растеряется перед непростыми завтрашними.

Предлагаем Вам провести исследование методов безусловной оптимизации, в рамках данного учебно-методологического пособия.

Рассмотренные алгоритмы широко применяются в различных прикладных задачах при нахождении минимумов функций. Например, метод наискорейшего спуска лежит в основе обучения искусственных нейронных сетей.

Постановка задачи

Задача многомерной оптимизации формулируется следующим образом: найти точку локального минимума $\overline{x^*} = (x_1^*, x_2^*, \dots, x_n^*)$ целевой функции $f(\overline{x}) = f(x_1, x_2, \dots, x_n)$ на множестве допустимых значений $\overline{x} = (x_1, x_2, \dots, x_n) \in R^n$. Символически задачу записывают так:

$$f(\overline{x^*}) = \min f(\overline{x}), \quad \overline{x} \in R^n.$$

Все методы нахождения точки локального минимума основаны на *итерационной процедуре*, реализуемой в соответствии с формулой

$$\overline{x^{(k+1)}} = \overline{x^{(k)}} + h_k \overline{p^{(k)}}, \quad (1)$$

где k – номер итерации $k = 0, 1, \dots$;

$\overline{x^{(k)}}$ – текущее приближение;

h_k – величина шага;

$\overline{p^k}$ – вектор, определяющий направление убывания функции $f(\overline{x})$.

Переходы от точек $\overline{x^{(k)}}$ к точкам $\overline{x^{(k+1)}}$ выполняются таким образом, чтобы обеспечивалось убывание значения целевой функции $f(\overline{x^{(k+1)}}) \leq f(\overline{x^{(k)}})$. Такие методы принято называть *методами спуска*.

Название метода спуска определяется способом выбора вектора $\overline{p^{(k)}}$, а его варианты связываются с различными способами выбора величины шага h_k .

Методы решения задач безусловной оптимизации принято разделять на группы: *нулевого, первого и второго порядков*, в зависимости от уровня используемой в методе информации о целевой функции.

Глава 1. Методы нулевого порядка

Методы нулевого порядка используют информацию только о значениях целевой функции $f(\bar{x})$. Направление минимизации в данном случае полностью определяется последовательными вычислениями значений целевой функции. Ниже рассматриваются методы, довольно часто применяемые на практике: метод Хука–Дживса, симплексный метод, метод Нелдера–Мида (деформируемого многогранника). Основное достоинство этих методов состоит в том, что они не требуют непрерывности целевой функции и существования производных.

1.1. Симплексный метод

Регулярным симплексом в n -мерном пространстве называется правильный многогранник с $n+1$ вершиной. При $n = 2$ симплексом является правильный треугольник, при $n = 3$ – тетраэдр и т.д. Отрезок, соединяющий 2 вершины симплекса, называется ребром симплекса.

Поиск симплексным методом ведется по следующей схеме. Устанавливаются координаты вершин симплексов. Определяется вершина с наибольшим значением целевой функции. Вместо нее строится новая вершина отражением старой через центр тяжести остальных вершин симплекса. На рисунке 1.1 представлен процесс построения нового симплекса на плоскости.

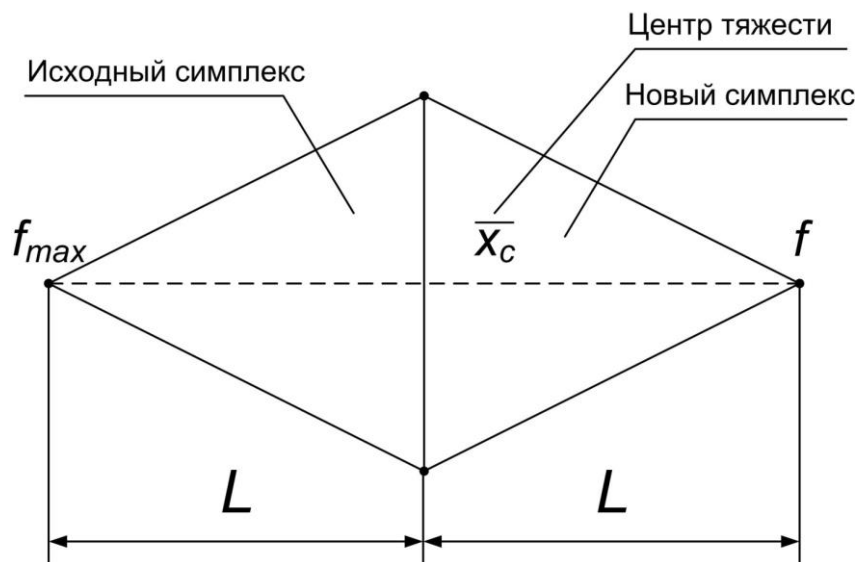


Рис 1.1. Построение нового симплекса

Если попытка отражения не приводит к уменьшению целевой функции, то выполняется операция редукции, в результате которой формируется новый симплекс с уменьшенными вдвое сторонами. При

операции редукции в качестве базовой точки выбирается вершина старого симплекса, в которой функция принимает минимальное значение.

В результате исключения вершин симплексов с наибольшим значением целевой функции процесс поиска сходится к минимальному значению.

Поиск завершается, когда разности между значениями функции в центре тяжести симплекса и вершинах становится достаточно малым.

На рисунке 1.2 представлена иллюстрация построения симплекса для двумерной области.

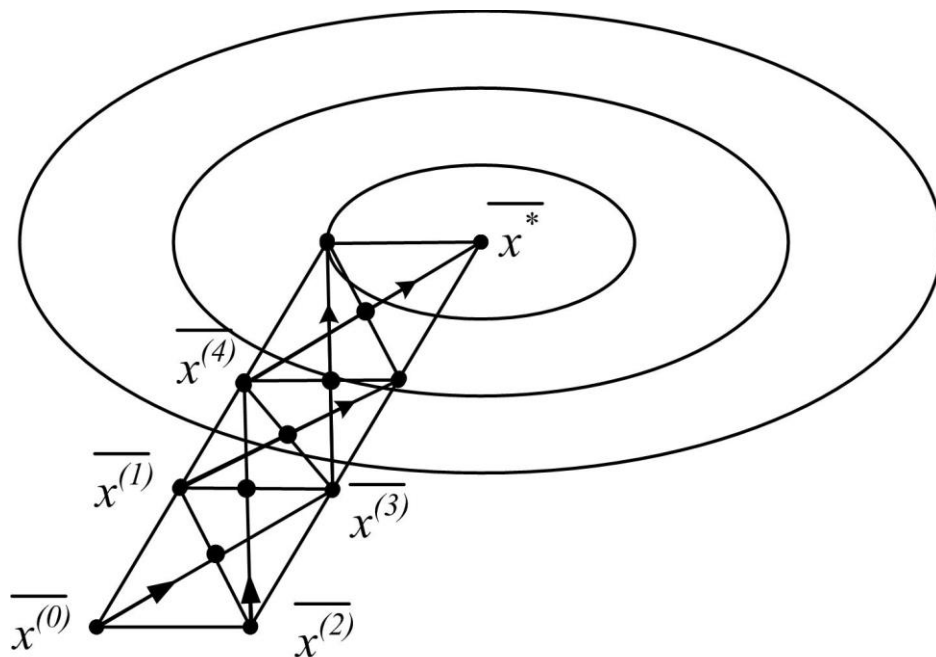


Рис. 1.2. Графическая иллюстрация поиска точки минимума симплексным методом

Объясним рис. 1.2. Пусть требуется решить задачу:

$$f(\bar{x}^*) = \min f(\bar{x}), \quad \bar{x} \in R^n$$

В двумерном пространстве R^2 решению такой задачи можно дать геометрическую иллюстрацию. Пусть точка $x = (x_1, x_2)$ лежит на плоскости Ox_1x_2 . Введем третью координату x_3 так, чтобы ось координат Ox_3 была перпендикулярна плоскости Ox_1x_2 (рис.3). Уравнению $x_3 = f(x_1, x_2)$ соответствует поверхность в трехмерном пространстве.

Если функция $f(x)$ достигает локального минимума в точке $x^* \in R^2$, то поверхность в некоторой окрестности точки x^* имеет форму чаши (рис. 1.3).

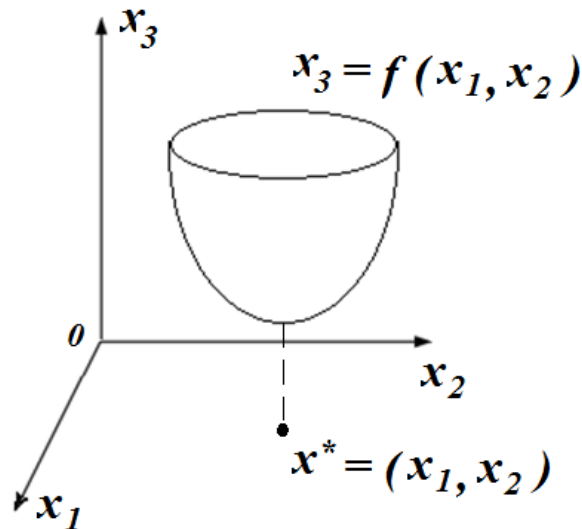


Рис. 1.3. Минимум функции

Линиями уровня функции $f(x_1, x_2)$ называют семейство линий плоскости R^2 , на которых функция принимает постоянное значение. Неявным уравнением линии уровня является уравнение $f(x_1, x_2) = C$. Если функция $f(x)$ имеет в R^2 единственную точку локального минимума $x^* (x_1^*, x_2^*)$, то такая функция называется мономодальной. Взаимное расположение ее линий уровня имеет вид, изображенный на рис. 1.4.

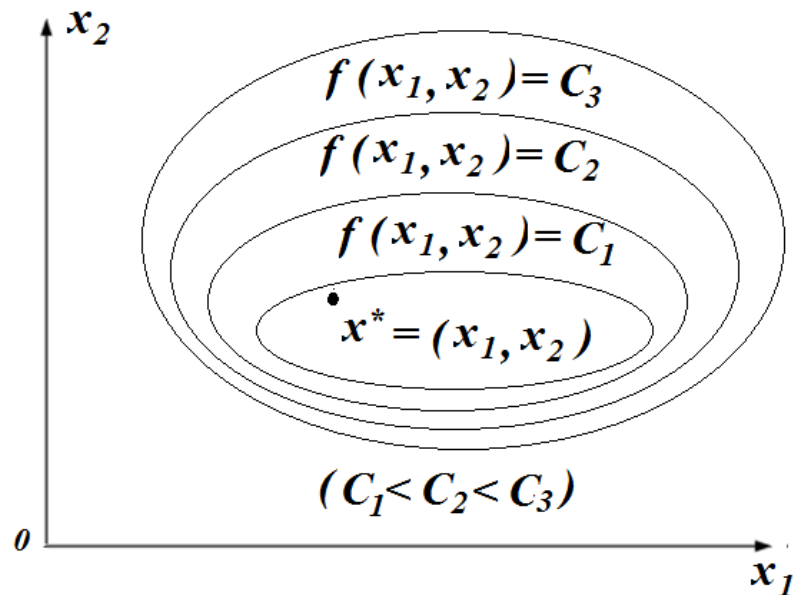
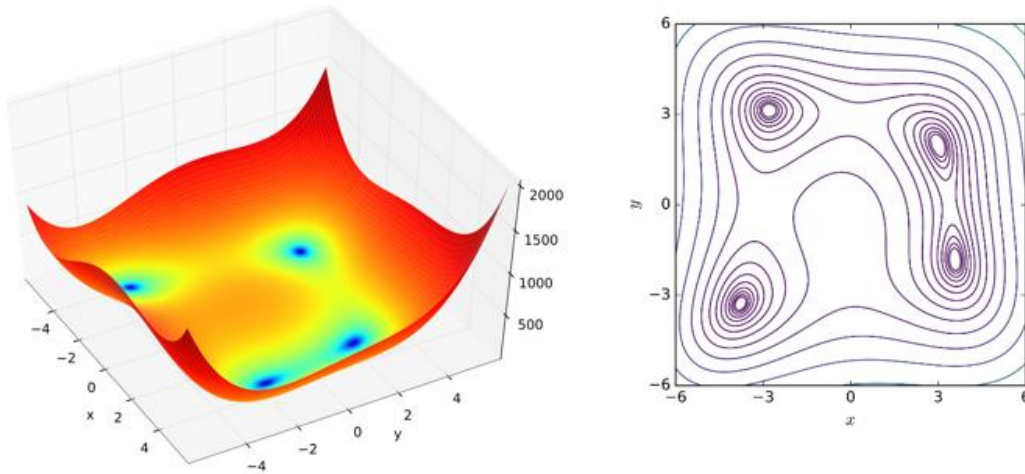


Рис. 1.4. Линии уровня

Мультимодальными называются функции, которые имеют более одного экстремума. Такова, например, функция Химмельблау

$$F(x) = (x_1^2 + x_2 - 11)^2 + (x_1 + x_2^2 - 7)^2,$$

имеющая четыре изолированные точки минимума (рис.1.5).



а) 3-D изображение

б) График линии уровня

Рис. 1.5. Функция Химмельблау

Рассмотрим алгоритм поиска симплексным методом

1. Задать размерность задачи оптимизации n , координаты начальной точки симплекса $\overline{x^{(0)}} = \{x_1^{(0)}, x_2^{(0)}, \dots, x_n^{(0)}\}$, длину ребра симплекса m , точность поиска ε .

2. Вычислить координаты остальных n вершин симплекса $\overline{x^{(1)}}$, $\overline{x^{(2)}}$, ..., $\overline{x^{(n)}}$ по формулам

$$\overline{x^{(i)}} = \begin{cases} x_j^{(0)} + \delta_1, & \text{если } i = j \\ x_j^{(0)} + \delta_2, & \text{если } i \neq j \end{cases} \quad \text{для } i, j = \overline{1, n}$$

где приращения δ_1 и δ_2 определяются по формулам

$$\delta_1 = \left(\frac{\sqrt{n+1}-1}{n\sqrt{2}} \right) * m, \quad \delta_2 = \left(\frac{\sqrt{n+1}+n-1}{n\sqrt{2}} \right) * m.$$

3. Определить номер вершины k с наибольшим значением целевой функции $f_{\max} = f(\overline{x^{(k)}})$.

4. Определить центр тяжести всех вершин многогранника за исключением вершины $\overline{x^{(k)}}$:

$$\overline{x_c} = \frac{1}{n} \sum_{\substack{i=0 \\ i \neq k}}^n \overline{x^{(i)}}.$$

5. Отобразить вершину $\overline{x^{(k)}}$ относительно центра тяжести $\overline{x} = 2\overline{x_c} - \overline{x^{(k)}}$. Вычислить значение целевой функции в отраженной точке $f(\overline{x})$ и перейти к пункту 6.

6. Проверить условие. Если $f(\overline{x}) < f(\overline{x^{(k)}})$, то операция

закончилась успешно. Положить $\overline{x^{(k)}} = \bar{x}$, $f(\overline{x^{(k)}}) = f(\bar{x})$ и перейти к пункту 8. В противном случае перейти к пункту 7.

7. Выполнить операцию редукции. Для этого отделить номер вершины r с минимальным значением целевой функции $f_{min} = f(\overline{x^{(r)}})$. Используя соотношение $\overline{x^{(i)}} = \overline{x^{(r)}} + 0,5(\overline{x^{(i)}} - \overline{x^{(r)}})$, $i = \overline{0, n}$, $i \neq r$, сформировать новый многогранник с уменьшенными вдвое сторонами. Перейти к шагу 8.

8. Определить центр тяжести симплекса $\overline{x_c} = \frac{1}{n+1} \sum_{i=0}^n \overline{x^{(i)}}$ значение функции в этой точке $f(\overline{x_c})$.

9. Проверить условие окончания процесса вычислений. Если $|f(\overline{x^{(i)}}) - f(\overline{x_c})| < \varepsilon$, $i = \overline{1, n}$, то процесс вычислений завершен. В качестве приближенного решения принять вершину с минимальным значением целевой функции. В противном случае перейти к пункту 3.

Информация о симплексе формируется и хранится в двумерном массиве размером $(n+1) \times (n+1)$. При этом каждая i -я строка массива ($i = \overline{0, n}$) содержит информацию о вершине симплекса $\overline{x^{(i)}}$ ($x_1^{(i)}, x_2^{(i)}, \dots, x_n^{(i)}$) и значении целевой функции в этой вершине $f(\overline{x^{(i)}})$ (таблица 1).

Таблица 1

		Координаты вершины				Значение функции
		1	2	...	n	
Номер вершины	0	$x_1^{(0)}$	$x_2^{(0)}$...	$x_n^{(0)}$	$f(\overline{x^{(0)}})$
	1	$x_1^{(1)}$	$x_2^{(1)}$...	$x_n^{(1)}$	$f(\overline{x^{(1)}})$

	n	$x_1^{(2)}$	$x_2^{(2)}$...	$x_n^{(2)}$	$f(\overline{x^{(n)}})$

При успешном отражении информация в массиве о старой вершине симплекса аннулируется, и на ее место записывается информация о новой вершине. Таким образом, размер массива в процессе итераций остается неизменным и характеризует текущий симплекс.

1.1.1. Практический расчет задачи

Постановка задачи. Найти минимум целевой функции.

$$f(\bar{x}) = 2.8 * x_2^2 + 1.9 * x_1 + 2.7 * x_1^2 + 1.6 - 1.9 * x_2$$

Симплексным методом с точностью $\varepsilon = 0,1$, размерность задачи $n = 2$, длина ребра симплекса $m = 0,5$.

Решение. Зададим начальную точку симплекса $\overline{x^{(0)}} = (x_1^{(0)}, x_2^{(0)})^T = (1, 1)^T$, подставим значения в уравнение целевой функции и для понимания отобразим полученное значение таблицу 2.

Таблица 2

		Координаты вершины		Значение функции
		1	2	
Номер вершины	0	$x_1^{(0)} = 1$	$x_2^{(0)} = 1$	$f(\overline{x^{(0)}}) = 7,1$

Вычислим приращения

$$\delta_1 = \left(\frac{\sqrt{n+1}-1}{n\sqrt{2}} \right) * m = \left(\frac{\sqrt{3}-1}{2\sqrt{2}} \right) * 0,5 = 0,129,$$

$$\delta_2 = \left(\frac{\sqrt{n+1}+n-1}{n\sqrt{2}} \right) * m = \left(\frac{\sqrt{3}+2-1}{2\sqrt{2}} \right) * 0,5 = 0,483.$$

Используя δ_1 и δ_2 , вычислим координаты двух остальных вершин симплекса

$$\begin{aligned} \overline{x^{(1)}} &= (x_1^{(0)} + \delta_1, x_2^{(0)} + \delta_2)^T = (1 + 0,129; 1 + 0,483)^T = \\ &= (1,129; 1,483)^T \end{aligned}$$

$$\begin{aligned} \overline{x^{(2)}} &= (x_1^{(0)} + \delta_2, x_2^{(0)} + \delta_1)^T = (1 + 0,483; 1 + 0,129)^T = \\ &= (1,483; 1,129)^T \end{aligned}$$

Итерация $k = 0$. Вычислим значение целевой функции $f(\overline{x})$ в вершинах $\overline{x^{(0)}}$, $\overline{x^{(1)}}$, $\overline{x^{(2)}}$:

$$f(\overline{x^{(0)}}) = 7,1, f(\overline{x^{(1)}}) = 10,53, f(\overline{x^{(2)}}) = 11,781.$$

Отобразим эти значения в таблице 3

Таблица 3

		Координаты вершины		Значение функции
		1	2	
Номер вершины	0	$x_1^{(0)} = 1$	$x_2^{(0)} = 1$	$f(\overline{x^{(0)}}) = 7,1$
	1	$x_1^{(1)} = 1,129$	$x_2^{(1)} = 1,483$	$f(\overline{x^{(1)}}) = 10,53$
	2	$x_1^{(2)} = 1,483$	$x_2^{(2)} = 1,129$	$f(\overline{x^{(2)}}) = 11,781$

Наибольшее значение целевой функции соответствует вершине $\overline{x^{(2)}}$, поэтому необходимо отразить ее относительно центра тяжести остальных

вершин $\overline{x^{(0)}}$ и $\overline{x^{(1)}}$. Тем самым вершина $\overline{x^{(2)}}$ исключается из расчета. Тогда центр тяжести расположен в точке

$$\overline{x_{c2}} = \frac{1}{2}(\overline{x_1^{(0)}} + \overline{x_1^{(1)}}, \overline{x_2^{(0)}} + \overline{x_2^{(1)}})^T = (1,065; 1,241)^T.$$

Используя свойство регулярности, найдем координаты отраженной вершины

$$\overline{x^{(3)}} = 2\overline{x_{c2}} - \overline{x^{(2)}} = (0,646; 1,354)^T.$$

В полученной вершине значение целевой функции $f(\overline{x^{(3)}}) = 6,515$.

Занесем полученные значения в таблицу 4.

Таблица 4

		Координаты вершины		Значение функции
		1	2	
Номер вершины	0	$\overline{x_1^{(0)}} = 1$	$\overline{x_2^{(0)}} = 1$	$f(\overline{x^{(0)}}) = 7,1$
	1	$\overline{x_1^{(1)}} = 1,129$	$\overline{x_2^{(1)}} = 1,483$	$f(\overline{x^{(1)}}) = 10,53$
	2	$\overline{x_1^{(2)}} = 1,483$	$\overline{x_2^{(2)}} = 1,129$	$f(\overline{x^{(2)}}) = 11,781$
	3	$\overline{x_1^{(3)}} = 0,646$	$\overline{x_2^{(3)}} = 1,354$	$f(\overline{x^{(3)}}) = 6,515$

Следовательно, наблюдается уменьшение целевой функции $f(\overline{x^{(3)}}) < f(\overline{x^{(2)}})$. Новый симплекс образован вершинами $\overline{x^{(0)}}$, $\overline{x^{(1)}}$, $\overline{x^{(3)}}$, которым соответствует значение целевой функции $f(\overline{x^{(0)}}) = 7,1$, $f(\overline{x^{(1)}}) = 10,53$, $f(\overline{x^{(3)}}) = 6,515$.

Проверим условие окончания поиска $|f(\overline{x^{(i)}}) - f(\overline{x_c})| < \varepsilon$, $i = 0,1,3$.
Определим центр тяжести симплекса

$$\overline{x_c} = \frac{1}{3}(\overline{x^{(0)}} + \overline{x^{(1)}} + \overline{x^{(3)}})^T = (0,925; 1,279)^T.$$

В полученной точке $f(\overline{x_c}) = 7,819$. Вычислим условие окончания поиска для точек симплекса:

$$|f(\overline{x^{(0)}}) - f(\overline{x_c})| = 0,719 > \varepsilon,$$

$$|f(\overline{x^{(1)}}) - f(\overline{x_c})| = 2,711 > \varepsilon,$$

$$|f(\overline{x^{(3)}}) - f(\overline{x_c})| = 1,304 > \varepsilon.$$

Так как все условия окончания поиска не выполняются, то процесс итераций должен быть продолжен.

Итерация $k = 1$. Наибольшее значение целевой функции соответствует вершине $\overline{x^{(2)}}$, поэтому необходимо отразить ее

относительно центра тяжести остальных вершин $\overline{x^{(0)}}$ и $\overline{x^{(3)}}$. центр тяжести расположен в точке

$$\overline{x_{c1}} = \frac{1}{2}(x_1^{(0)} + x_1^{(3)}, x_2^{(0)} + x_2^{(3)})^T = (0,823; 1,177)^T.$$

Используя свойство регулярности, найдем координаты отраженной вершины

$$\overline{x^{(4)}} = 2\overline{x_{c1}} - \overline{x^{(1)}} = (0,517; 0,871)^T.$$

В полученной вершине значение целевой функции $f(\overline{x^{(4)}}) = 3,772$.

Занесем полученные значения в таблицу 5.

Таблица 5

		Координаты вершины		Значение функции
		1	2	
Номер вершины	0	$x_1^{(0)} = 1$	$x_2^{(0)} = 1$	$f(\overline{x^{(0)}}) = 7,1$
	1	$x_1^{(1)} = 1,129$	$x_2^{(1)} = 1,483$	$f(\overline{x^{(1)}}) = 10,53$
	2	$x_1^{(2)} = 1,483$	$x_2^{(2)} = 1,129$	$f(\overline{x^{(2)}}) = 11,781$
	3	$x_1^{(3)} = 0,646$	$x_2^{(3)} = 1,354$	$f(\overline{x^{(3)}}) = 6,515$
	4	$x_1^{(4)} = 0,517$	$x_2^{(4)} = 0,871$	$f(\overline{x^{(4)}}) = 3,772$

Следовательно, наблюдается уменьшение целевой функции $f(\overline{x^{(4)}}) < f(\overline{x^{(2)}})$. Новый симплекс образован вершинами $\overline{x^{(0)}}$, $\overline{x^{(3)}}$, $\overline{x^{(4)}}$, которым соответствует значение целевой функции $f(\overline{x^{(0)}}) = 7,1$, $f(\overline{x^{(3)}}) = 6,515$, $f(\overline{x^{(4)}}) = 3,772$.

Проверим условие окончания поиска $|f(\overline{x^{(i)}}) - f(\overline{x_c})| < \varepsilon$, $i = 0,3,4$.

Определим центр тяжести симплекса

$$\overline{x_c} = \frac{1}{3}(\overline{x^{(0)}} + \overline{x^{(3)}} + \overline{x^{(4)}})^T = (0,721; 1,075)^T.$$

В полученной точке $f(\overline{x_c}) = 5,566$. Вычислим условие окончания поиска для точек симплекса:

$$|f(\overline{x^{(0)}}) - f(\overline{x_c})| = 1,534 > \varepsilon,$$

$$|f(\overline{x^{(3)}}) - f(\overline{x_c})| = 1,794 > \varepsilon,$$

$$|f(\overline{x^{(4)}}) - f(\overline{x_c})| = 0,948 > \varepsilon.$$

Так как все условия окончания поиска не выполняются, то процесс итераций должен быть продолжен.

Итерация $k = 2$. Наибольшее значение целевой функции соответствует вершине $\overline{x^{(0)}}$, поэтому необходимо отразить ее

относительно центра тяжести остальных вершин $\overline{x^{(3)}}$ и $\overline{x^{(4)}}$. центр тяжести расположен в точке

$$\overline{x_{c0}} = \frac{1}{2}(x_1^{(3)} + x_1^{(4)}, x_2^{(3)} + x_2^{(4)})^T = (0,582; 1,112)^T.$$

Используя свойство регулярности, найдем координаты отраженной вершины

$$\overline{x^{(5)}} = 2\overline{x_{c0}} - \overline{x^{(0)}} = (0,163; 1,224)^T.$$

В полученной вершине значение целевой функции $f(\overline{x^{(5)}}) = 3,853$.

Занесем полученные значения в таблицу 6.

Таблица 6

		Координаты вершины		Значение функции
		1	2	
Номер вершины	0	$x_1^{(0)} = 1$	$x_2^{(0)} = 1$	$f(\overline{x^{(0)}}) = 7,1$
	1	$x_1^{(1)} = 1,129$	$x_2^{(1)} = 1,483$	$f(\overline{x^{(1)}}) = 10,53$
	2	$x_1^{(2)} = 1,483$	$x_2^{(2)} = 1,129$	$f(\overline{x^{(2)}}) = 11,781$
	3	$x_1^{(3)} = 0,646$	$x_2^{(3)} = 1,354$	$f(\overline{x^{(3)}}) = 6,515$
	4	$x_1^{(4)} = 0,517$	$x_2^{(4)} = 0,871$	$f(\overline{x^{(4)}}) = 3,772$
	5	$x_1^{(5)} = 0,163$	$x_2^{(5)} = 1,224$	$f(\overline{x^{(5)}}) = 3,853$

Следовательно, наблюдается уменьшение целевой функции $f(\overline{x^{(5)}}) < f(\overline{x^{(0)}})$. Новый симплекс образован вершинами $\overline{x^{(5)}}$, $\overline{x^{(3)}}$, $\overline{x^{(4)}}$, которым соответствует значение целевой функции $f(\overline{x^{(5)}}) = 3,853$, $f(\overline{x^{(3)}}) = 6,515$, $f(\overline{x^{(4)}}) = 3,772$.

Проверим условие окончания поиска $|f(\overline{x^{(i)}}) - f(\overline{x_c})| < \varepsilon$, $i = 5,3,4$.

Определим центр тяжести симплекса

$$\overline{x_c} = \frac{1}{3}(\overline{x^{(5)}} + \overline{x^{(3)}} + \overline{x^{(4)}})^T = (0,442; 1,149)^T.$$

В полученной точке $f(\overline{x_c}) = 4,484$. Вычислим условие окончания поиска для точек симплекса:

$$|f(\overline{x^{(5)}}) - f(\overline{x_c})| = 0,631 > \varepsilon,$$

$$|f(\overline{x^{(3)}}) - f(\overline{x_c})| = 0,712 > \varepsilon,$$

$$|f(\overline{x^{(4)}}) - f(\overline{x_c})| = 2,031 > \varepsilon.$$

Так как все условия окончания поиска не выполняются, то процесс итераций должен быть продолжен.

Итерация $k = 3$. Наибольшее значение целевой функции соответствует вершине $\overline{x^{(3)}}$, поэтому необходимо отразить ее относительно центра тяжести остальных вершин $\overline{x^{(4)}}$ и $\overline{x^{(5)}}$. центр тяжести расположен в точке

$$\overline{x_{c3}} = \frac{1}{2}(x_1^{(4)} + x_1^{(5)}, x_2^{(4)} + x_2^{(5)})^T = (0,34; 1,047)^T.$$

Используя свойство регулярности, найдем координаты отраженной вершины

$$\overline{x^{(6)}} = 2\overline{x_{c3}} - \overline{x^{(3)}} = (0,034; 0,741)^T.$$

В полученной вершине значение целевой функции $f(\overline{x^{(6)}}) = 1,798$.

Занесем полученные значения в таблицу 7.

Таблица 7

		Координаты вершины		Значение функции
		1	2	
Номер вершины	0	$x_1^{(0)} = 1$	$x_2^{(0)} = 1$	$f(\overline{x^{(0)}}) = 7,1$
	1	$x_1^{(1)} = 1,129$	$x_2^{(1)} = 1,483$	$f(\overline{x^{(1)}}) = 10,53$
	2	$x_1^{(2)} = 1,483$	$x_2^{(2)} = 1,129$	$f(\overline{x^{(2)}}) = 11,781$
	3	$x_1^{(3)} = 0,646$	$x_2^{(3)} = 1,354$	$f(\overline{x^{(3)}}) = 6,515$
	4	$x_1^{(4)} = 0,517$	$x_2^{(4)} = 0,871$	$f(\overline{x^{(4)}}) = 3,772$
	5	$x_1^{(5)} = 0,163$	$x_2^{(5)} = 1,224$	$f(\overline{x^{(5)}}) = 3,853$
	6	$x_1^{(6)} = 0,034$	$x_2^{(6)} = 0,741$	$f(\overline{x^{(6)}}) = 1,798$

Следовательно, наблюдается уменьшение целевой функции $f(\overline{x^{(6)}}) < f(\overline{x^{(3)}})$. Новый симплекс образован вершинами $\overline{x^{(5)}}$, $\overline{x^{(6)}}$, $\overline{x^{(4)}}$, которым соответствует значение целевой функции $f(\overline{x^{(5)}}) = 3,853$, $f(\overline{x^{(6)}}) = 1,798$, $f(\overline{x^{(4)}}) = 3,772$.

Проверим условие окончания поиска $|f(\overline{x^{(i)}}) - f(\overline{x_c})| < \varepsilon$, $i = 5, 6, 4$.
Определим центр тяжести симплекса

$$\overline{x_c} = \frac{1}{3}(\overline{x^{(5)}} + \overline{x^{(6)}} + \overline{x^{(4)}})^T = (0,238; 0,945)^T.$$

В полученной точке $f(\overline{x_c}) = 2,912$. Вычислим условие окончания поиска для точек симплекса:

$$|f(\overline{x^{(5)}}) - f(\overline{x_c})| = 0,941 > \varepsilon,$$

$$|f(\overline{x^{(6)}}) - f(\overline{x_c})| = 0,86 > \varepsilon,$$

$$|f(\overline{x^{(4)}}) - f(\overline{x_c})| = 1,114 > \varepsilon.$$

Так как все условия окончания поиска не выполняются, то процесс итераций должен быть продолжен.

Итерация $k = 4$. Наибольшее значение целевой функции соответствует вершине $\overline{x^{(5)}}$, поэтому необходимо отразить ее относительно центра тяжести остальных вершин $\overline{x^{(4)}}$ и $\overline{x^{(6)}}$. центр тяжести расположен в точке

$$\overline{x_{c5}} = \frac{1}{2}(x_1^{(4)} + x_1^{(6)}, x_2^{(4)} + x_2^{(6)})^T = (0,276; 0,806)^T.$$

Используя свойство регулярности, найдем координаты отраженной вершины

$$\overline{x^{(7)}} = 2\overline{x_{c5}} - \overline{x^{(5)}} = (0,388; 0,388)^T.$$

В полученной вершине значение целевой функции $f(\overline{x^{(7)}}) = 2,426$.

Занесем полученные значения в таблицу 8.

Таблица 8

		Координаты вершины		Значение функции
		1	2	
Номер вершины	0	$x_1^{(0)} = 1$	$x_2^{(0)} = 1$	$f(\overline{x^{(0)}}) = 7,1$
	1	$x_1^{(1)} = 1,129$	$x_2^{(1)} = 1,483$	$f(\overline{x^{(1)}}) = 10,53$
	2	$x_1^{(2)} = 1,483$	$x_2^{(2)} = 1,129$	$f(\overline{x^{(2)}}) = 11,781$
	3	$x_1^{(3)} = 0,646$	$x_2^{(3)} = 1,354$	$f(\overline{x^{(3)}}) = 6,515$
	4	$x_1^{(4)} = 0,517$	$x_2^{(4)} = 0,871$	$f(\overline{x^{(4)}}) = 3,772$
	5	$x_1^{(5)} = 0,163$	$x_2^{(5)} = 1,224$	$f(\overline{x^{(5)}}) = 3,853$
	6	$x_1^{(6)} = 0,034$	$x_2^{(6)} = 0,741$	$f(\overline{x^{(6)}}) = 1,798$
	7	$x_1^{(7)} = 0,388$	$x_2^{(7)} = 0,388$	$f(\overline{x^{(7)}}) = 2,426$

Следовательно, наблюдается уменьшение целевой функции $f(\overline{x^{(7)}}) < f(\overline{x^{(5)}})$. Новый симплекс образован вершинами $\overline{x^{(7)}}$, $\overline{x^{(6)}}$, $\overline{x^{(4)}}$, которым соответствует значение целевой функции $f(\overline{x^{(7)}}) = 2,426$, $f(\overline{x^{(6)}}) = 1,798$, $f(\overline{x^{(4)}}) = 3,772$.

Проверим условие окончания поиска $|f(\overline{x^{(i)}}) - f(\overline{x_c})| < \varepsilon$, $i = 7,6,4$.
Определим центр тяжести симплекса

$$\overline{x_c} = \frac{1}{3}(\overline{x^{(7)}} + \overline{x^{(6)}} + \overline{x^{(4)}})^T = (0,313; 0,666)^T.$$

В полученной точке $f(\overline{x_c}) = 2,436$. Вычислим условие окончания поиска для точек симплекса:

$$|f(\overline{x^{(7)}}) - f(\overline{x_c})| = 0,01 < \varepsilon,$$

$$|f(\overline{x^{(6)}}) - f(\overline{x_c})| = 1,336 > \varepsilon,$$

$$|f(\overline{x^{(4)}}) - f(\overline{x_c})| = 0,639 > \varepsilon.$$

Так как не все условия окончания поиска выполняются, то процесс итераций должен быть продолжен (первое условие выполнено).

Итерация $k = 5$. Наибольшее значение целевой функции соответствует вершине $\overline{x^{(4)}}$, поэтому необходимо отразить ее относительно центра тяжести остальных вершин $\overline{x^{(7)}}$ и $\overline{x^{(6)}}$. центр тяжести расположен в точке

$$\overline{x_{c4}} = \frac{1}{2}(x_1^{(7)} + x_1^{(6)}, x_2^{(7)} + x_2^{(6)})^T = (0,211; 0,564)^T.$$

Используя свойство регулярности, найдем координаты отраженной вершины

$$\overline{x^{(8)}} = 2\overline{x_{c4}} - \overline{x^{(4)}} = (-0,095; 0,258)^T.$$

В полученной вершине значение целевой функции $f(\overline{x^{(8)}}) = 1,139$.

Занесем полученные значения в таблицу 9.

Таблица 9

		Координаты вершины		Значение функции
		1	2	
Номер вершины	0	$x_1^{(0)} = 1$	$x_2^{(0)} = 1$	$f(\overline{x^{(0)}}) = 7,1$
	1	$x_1^{(1)} = 1,129$	$x_2^{(1)} = 1,483$	$f(\overline{x^{(1)}}) = 10,53$
	2	$x_1^{(2)} = 1,483$	$x_2^{(2)} = 1,129$	$f(\overline{x^{(2)}}) = 11,781$
	3	$x_1^{(3)} = 0,646$	$x_2^{(3)} = 1,354$	$f(\overline{x^{(3)}}) = 6,515$
	4	$x_1^{(4)} = 0,517$	$x_2^{(4)} = 0,871$	$f(\overline{x^{(4)}}) = 3,772$
	5	$x_1^{(5)} = 0,163$	$x_2^{(5)} = 1,224$	$f(\overline{x^{(5)}}) = 3,853$
	6	$x_1^{(6)} = 0,034$	$x_2^{(6)} = 0,741$	$f(\overline{x^{(6)}}) = 1,798$
	7	$x_1^{(7)} = 0,388$	$x_2^{(7)} = 0,388$	$f(\overline{x^{(7)}}) = 2,426$
	8	$x_1^{(8)} = -0,095$	$x_2^{(8)} = 0,258$	$f(\overline{x^{(8)}}) = 1,139$

Следовательно, наблюдается уменьшение целевой функции $f(\overline{x^{(8)}}) < f(\overline{x^{(4)}})$. Новый симплекс образован вершинами $\overline{x^{(7)}}$, $\overline{x^{(6)}}$, $\overline{x^{(8)}}$, которым соответствует значение целевой функции $f(\overline{x^{(7)}}) = 2,426$, $f(\overline{x^{(6)}}) = 1,798$, $f(\overline{x^{(8)}}) = 1,139$.

Проверим условие окончания поиска $|f(\overline{x^{(i)}}) - f(\overline{x_c})| < \varepsilon, i = 7, 6, 8$.

Определим центр тяжести симплекса

$$\overline{x_c} = \frac{1}{3}(\overline{x^{(7)}} + \overline{x^{(6)}} + \overline{x^{(8)}})^T = (0,109; 0,462)^T.$$

В полученной точке $f(\overline{x_c}) = 1,559$. Вычислим условие окончания поиска для точек симплекса:

$$|f(\overline{x^{(7)}}) - f(\overline{x_c})| = 0,868 > \varepsilon,$$

$$|f(\overline{x^{(6)}}) - f(\overline{x_c})| = 0,419 > \varepsilon,$$

$$|f(\overline{x^{(8)}}) - f(\overline{x_c})| = 0,239 > \varepsilon.$$

Так как все условия окончания поиска не выполняются, то процесс итераций должен быть продолжен.

Итерация $k = 6$. Наибольшее значение целевой функции соответствует вершине $\overline{x^{(7)}}$, поэтому необходимо отразить ее относительно центра тяжести остальных вершин $\overline{x^{(8)}}$ и $\overline{x^{(6)}}$. центр тяжести расположен в точке

$$\overline{x_{c7}} = \frac{1}{2}(\overline{x_1^{(8)}} + \overline{x_1^{(6)}}, \overline{x_2^{(8)}} + \overline{x_2^{(6)}})^T = (-0,031; 0,5)^T.$$

Используя свойство регулярности, найдем координаты отраженной вершины

$$\overline{x^{(9)}} = 2\overline{x_{c7}} - \overline{x^{(7)}} = (-0,449; 0,612)^T.$$

В полученной вершине значение целевой функции $f(\overline{x^{(9)}}) = 1,177$.

Занесем полученные значения в таблицу 10.

Таблица 10

		Координаты вершины		Значение функции
		1	2	
Номер вершины	0	$x_1^{(0)} = 1$	$x_2^{(0)} = 1$	$f(\overline{x^{(0)}}) = 7,1$
	1	$x_1^{(1)} = 1,129$	$x_2^{(1)} = 1,483$	$f(\overline{x^{(1)}}) = 10,53$
	2	$x_1^{(2)} = 1,483$	$x_2^{(2)} = 1,129$	$f(\overline{x^{(2)}}) = 11,781$
	3	$x_1^{(3)} = 0,646$	$x_2^{(3)} = 1,354$	$f(\overline{x^{(3)}}) = 6,515$
	4	$x_1^{(4)} = 0,517$	$x_2^{(4)} = 0,871$	$f(\overline{x^{(4)}}) = 3,772$
	5	$x_1^{(5)} = 0,163$	$x_2^{(5)} = 1,224$	$f(\overline{x^{(5)}}) = 3,853$
	6	$x_1^{(6)} = 0,034$	$x_2^{(6)} = 0,741$	$f(\overline{x^{(6)}}) = 1,798$
	7	$x_1^{(7)} = 0,388$	$x_2^{(7)} = 0,388$	$f(\overline{x^{(7)}}) = 2,426$
	8	$x_1^{(8)} = -0,095$	$x_2^{(8)} = 0,258$	$f(\overline{x^{(8)}}) = 1,139$

	9	$x_1^{(9)} = -0,449$	$x_2^{(9)} = 0,612$	$f(\overline{x^{(9)}}) = 1,177$
--	---	----------------------	---------------------	---------------------------------

Следовательно, наблюдается уменьшение целевой функции $f(\overline{x^{(9)}}) < f(\overline{x^{(7)}})$. Новый симплекс образован вершинами $\overline{x^{(9)}}$, $\overline{x^{(6)}}$, $\overline{x^{(8)}}$, которым соответствует значение целевой функции $f(\overline{x^{(9)}}) = 1,177$, $f(\overline{x^{(6)}}) = 1,798$, $f(\overline{x^{(8)}}) = 1,139$.

Проверим условие окончания поиска $|f(\overline{x^{(i)}}) - f(\overline{x_c})| < \varepsilon$, $i = 9, 6, 8$.
Определим центр тяжести симплекса

$$\overline{x_c} = \frac{1}{3}(\overline{x^{(9)}} + \overline{x^{(6)}} + \overline{x^{(8)}})^T = (-0,17; 0,537)^T.$$

В полученной точке $f(\overline{x_c}) = 1,142$. Вычислим условие окончания поиска для точек симплекса:

$$|f(\overline{x^{(9)}}) - f(\overline{x_c})| = 0,035 < \varepsilon,$$

$$|f(\overline{x^{(6)}}) - f(\overline{x_c})| = 0,003 < \varepsilon,$$

$$|f(\overline{x^{(8)}}) - f(\overline{x_c})| = 0,656 > \varepsilon.$$

Так как не все условия окончания поиска выполняются, то процесс итераций должен быть продолжен (первое и второе условия выполнены).

Итерация $k = 7$. Наибольшее значение целевой функции соответствует вершине $\overline{x^{(6)}}$, поэтому необходимо отразить ее относительно центра тяжести остальных вершин $\overline{x^{(8)}}$ и $\overline{x^{(9)}}$. центр тяжести расположен в точке

$$\overline{x_{c6}} = \frac{1}{2}(x_1^{(8)} + x_1^{(9)}, x_2^{(8)} + x_2^{(9)})^T = (-0,272; 0,435)^T.$$

Используя свойство регулярности, найдем координаты отраженной вершины

$$\overline{x^{(10)}} = 2\overline{x_{c6}} - \overline{x^{(6)}} = (-0,578; 0,129)^T.$$

В полученной вершине значение целевой функции $f(\overline{x^{(10)}}) = 1,206$.

Занесем полученные значения в таблицу 11.

Таблица 11

		Координаты вершины		Значение функции
		1	2	
Номер вершины	0	$x_1^{(0)} = 1$	$x_2^{(0)} = 1$	$f(\overline{x^{(0)}}) = 7,1$
	1	$x_1^{(1)} = 1,129$	$x_2^{(1)} = 1,483$	$f(\overline{x^{(1)}}) = 10,53$
	2	$x_1^{(2)} = 1,483$	$x_2^{(2)} = 1,129$	$f(\overline{x^{(2)}}) = 11,781$
	3	$x_1^{(3)} = 0,646$	$x_2^{(3)} = 1,354$	$f(\overline{x^{(3)}}) = 6,515$

4	$x_1^{(4)} = 0,517$	$x_2^{(4)} = 0,871$	$f(\overline{x^{(4)}}) = 3,772$
5	$x_1^{(5)} = 0,163$	$x_2^{(5)} = 1,224$	$f(\overline{x^{(5)}}) = 3,853$
6	$x_1^{(6)} = 0,034$	$x_2^{(6)} = 0,741$	$f(\overline{x^{(6)}}) = 1,798$
7	$x_1^{(7)} = 0,388$	$x_2^{(7)} = 0,388$	$f(\overline{x^{(7)}}) = 2,426$
8	$x_1^{(8)} = -0,095$	$x_2^{(8)} = 0,258$	$f(\overline{x^{(8)}}) = 1,139$
9	$x_1^{(9)} = -0,449$	$x_2^{(9)} = 0,612$	$f(\overline{x^{(9)}}) = 1,177$
10	$x_1^{(10)} = -0,578$	$x_2^{(10)} = 0,129$	$f(\overline{x^{(10)}}) = 1,206$

Следовательно, наблюдается уменьшение целевой функции $f(\overline{x^{(10)}}) < f(\overline{x^{(6)}})$. Новый симплекс образован вершинами $\overline{x^{(9)}}$, $\overline{x^{(10)}}$, $\overline{x^{(8)}}$, которым соответствует значение целевой функции $f(\overline{x^{(9)}}) = 1,177$, $f(\overline{x^{(10)}}) = 1,206$, $f(\overline{x^{(8)}}) = 1,139$.

Проверим условие окончания поиска $|f(\overline{x^{(i)}}) - f(\overline{x_c})| < \varepsilon, i = 9, 10, 8$.
Определим центр тяжести симплекса

$$\overline{x_c} = \frac{1}{3}(\overline{x^{(9)}} + \overline{x^{(10)}} + \overline{x^{(8)}})^T = (-0,374; 0,333)^T.$$

В полученной точке $f(\overline{x_c}) = 0,945$. Вычислим условие окончания поиска для точек симплекса:

$$|f(\overline{x^{(9)}}) - f(\overline{x_c})| = 0,232 > \varepsilon,$$

$$|f(\overline{x^{(10)}}) - f(\overline{x_c})| = 0,195 > \varepsilon,$$

$$|f(\overline{x^{(8)}}) - f(\overline{x_c})| = 0,261 > \varepsilon.$$

Так как все условия окончания поиска не выполняются, то процесс итераций должен быть продолжен.

Итерация $k = 8$. Наибольшее значение целевой функции соответствует вершине $\overline{x^{(10)}}$, поэтому необходимо отразить ее относительно центра тяжести остальных вершин $\overline{x^{(8)}}$ и $\overline{x^{(9)}}$. центр тяжести расположен в точке

$$\overline{x_{c10}} = \frac{1}{2}(x_1^{(8)} + x_1^{(9)}, x_2^{(8)} + x_2^{(9)})^T = (-0,272; 0,435)^T.$$

Используя свойство регулярности, найдем координаты отраженной вершины

$$\overline{x^{(11)}} = 2\overline{x_{c10}} - \overline{x^{(10)}} = (0,034; 0,741)^T.$$

В полученной вершине значение целевой функции $f(\overline{x^{(11)}}) = 1,798$.

Учитывая, что уменьшение целевой функции $f(\overline{x^{(10)}}) < f(\overline{x^{(11)}})$ не наблюдается, то проведем редукцию (значения $\overline{x^{(11)}}$ и $f(\overline{x^{(11)}})$ в таблицу 1.12 не записываем)

Сформируем новый многогранник с уменьшенными вдвое сторонами и вершиной $\overline{x^{(8)}}$, которой соответствует наименьшее значение целевой функции $f_l = f(\overline{x^{(8)}}) = 1,139$.

$$\overline{x^{(12)}} = \overline{x^{(8)}} + 0,5 * (\overline{x^{(10)}} - \overline{x^{(8)}}) = (-0,337; 0,194)^T.$$

$$f(\overline{x^{(12)}}) = 1,004.$$

$$\overline{x^{(13)}} = \overline{x^{(8)}} + 0,5 * (\overline{x^{(13)}} - \overline{x^{(8)}}) = (-0,272; 0,435)^T.$$

$$f(\overline{x^{(13)}}) = 0,986.$$

Новый симплекс образован вершинами $\overline{x^{(13)}}$, $\overline{x^{(8)}}$, $\overline{x^{(12)}}$, которым соответствует значение целевой функции $f(\overline{x^{(13)}}) = 0,986$, $f(\overline{x^{(8)}}) = 1,139$, $f(\overline{x^{(12)}}) = 1,004$.

Занесем полученные значения в таблицу 12.

Таблица 12

		Координаты вершины		Значение функции
		1	2	
Номер вершины	0	$x_1^{(0)} = 1$	$x_2^{(0)} = 1$	$f(\overline{x^{(0)}}) = 7,1$
	1	$x_1^{(1)} = 1,129$	$x_2^{(1)} = 1,483$	$f(\overline{x^{(1)}}) = 10,53$
	2	$x_1^{(2)} = 1,483$	$x_2^{(2)} = 1,129$	$f(\overline{x^{(2)}}) = 11,781$
	3	$x_1^{(3)} = 0,646$	$x_2^{(3)} = 1,354$	$f(\overline{x^{(3)}}) = 6,515$
	4	$x_1^{(4)} = 0,517$	$x_2^{(4)} = 0,871$	$f(\overline{x^{(4)}}) = 3,772$
	5	$x_1^{(5)} = 0,163$	$x_2^{(5)} = 1,224$	$f(\overline{x^{(5)}}) = 3,853$
	6	$x_1^{(6)} = 0,034$	$x_2^{(6)} = 0,741$	$f(\overline{x^{(6)}}) = 1,798$
	7	$x_1^{(7)} = 0,388$	$x_2^{(7)} = 0,388$	$f(\overline{x^{(7)}}) = 2,426$
	8	$x_1^{(8)} = -0,095$	$x_2^{(8)} = 0,258$	$f(\overline{x^{(8)}}) = 1,139$
	9	$x_1^{(9)} = -0,449$	$x_2^{(9)} = 0,612$	$f(\overline{x^{(9)}}) = 1,177$
	10	$x_1^{(10)} = -0,578$	$x_2^{(10)} = 0,129$	$f(\overline{x^{(10)}}) = 1,206$
	12	$x_1^{(12)} = -0,337$	$x_2^{(12)} = 0,194$	$f(\overline{x^{(12)}}) = 1,004$
	13	$x_1^{(13)} = -0,272$	$x_2^{(13)} = 0,435$	$f(\overline{x^{(13)}}) = 0,986$

Проверим условие окончания поиска $|f(\overline{x^{(i)}}) - f(\overline{x_c})| < \varepsilon$, $i = 13, 8, 12$. Определим центр тяжести симплекса

$$\overline{x_c} = \frac{1}{3}(\overline{x^{(13)}} + \overline{x^{(8)}} + \overline{x^{(12)}})^T = (-0,235; 0,296)^T.$$

В полученной точке $f(\overline{x_c}) = 0,986$. Вычислим условие окончания поиска для точек симплекса:

$$|f(\overline{x^{(13)}}) - f(\overline{x_c})| = 0 < \varepsilon,$$

$$|f(\overline{x^{(8)}}) - f(\overline{x_c})| = 0,154 > \varepsilon,$$

$$|f(\overline{x^{(12)}}) - f(\overline{x_c})| = 0,018 < \varepsilon.$$

Так как не все условия окончания поиска выполняются, то процесс итераций должен быть продолжен (первое и третье условия выполнены).

Итерация $k = 9$. Наибольшее значение целевой функции соответствует вершине $\overline{x^{(8)}}$, поэтому необходимо отразить ее относительно центра тяжести остальных вершин $\overline{x^{(13)}}$ и $\overline{x^{(12)}}$. центр тяжести расположен в точке

$$\overline{x_{c8}} = \frac{1}{2}(\overline{x_1^{(13)}} + \overline{x_1^{(12)}}, \overline{x_2^{(13)}} + \overline{x_2^{(12)}})^T = (-0,304; 0,314)^T.$$

Используя свойство регулярности, найдем координаты отраженной вершины

$$\overline{x^{(14)}} = 2\overline{x_{c8}} - \overline{x^{(8)}} = (-0,514; 0,37)^T.$$

В полученной вершине значение целевой функции $f(\overline{x^{(14)}}) = 1,017$.

Занесем полученные значения в таблицу 13.

Таблица 13

		Координаты вершины		Значение функции
		1	2	
Номер вершины	0	$x_1^{(0)} = 1$	$x_2^{(0)} = 1$	$f(\overline{x^{(0)}}) = 7,1$
	1	$x_1^{(1)} = 1,129$	$x_2^{(1)} = 1,483$	$f(\overline{x^{(1)}}) = 10,53$
	2	$x_1^{(2)} = 1,483$	$x_2^{(2)} = 1,129$	$f(\overline{x^{(2)}}) = 11,781$
	3	$x_1^{(3)} = 0,646$	$x_2^{(3)} = 1,354$	$f(\overline{x^{(3)}}) = 6,515$
	4	$x_1^{(4)} = 0,517$	$x_2^{(4)} = 0,871$	$f(\overline{x^{(4)}}) = 3,772$
	5	$x_1^{(5)} = 0,163$	$x_2^{(5)} = 1,224$	$f(\overline{x^{(5)}}) = 3,853$
	6	$x_1^{(6)} = 0,034$	$x_2^{(6)} = 0,741$	$f(\overline{x^{(6)}}) = 1,798$
	7	$x_1^{(7)} = 0,388$	$x_2^{(7)} = 0,388$	$f(\overline{x^{(7)}}) = 2,426$
	8	$x_1^{(8)} = -0,095$	$x_2^{(8)} = 0,258$	$f(\overline{x^{(8)}}) = 1,139$

	9	$x_1^{(9)} = -0,449$	$x_2^{(9)} = 0,612$	$f(\overline{x^{(9)}}) = 1,177$
	10	$x_1^{(10)} = -0,578$	$x_2^{(10)} = 0,129$	$f(\overline{x^{(10)}}) = 1,206$
	12	$x_1^{(12)} = -0,337$	$x_2^{(12)} = 0,194$	$f(\overline{x^{(12)}}) = 1,004$
	13	$x_1^{(13)} = -0,272$	$x_2^{(13)} = 0,435$	$f(\overline{x^{(13)}}) = 0,986$
	14	$x_1^{(14)} = -0,514$	$x_2^{(14)} = 0,314$	$f(\overline{x^{(14)}}) = 1,017$

Следовательно, наблюдается уменьшение целевой функции $f(\overline{x^{(14)}}) < f(\overline{x^{(8)}})$. Новый симплекс образован вершинами $\overline{x^{(13)}}$, $\overline{x^{(14)}}$, $\overline{x^{(12)}}$, которым соответствует значение целевой функции $f(\overline{x^{(13)}}) = 0,986$, $f(\overline{x^{(14)}}) = 1,017$, $f(\overline{x^{(12)}}) = 1,004$.

Проверим условие окончания поиска $|f(\overline{x^{(i)}}) - f(\overline{x_c})| < \varepsilon$, $i = 13, 14, 12$. Определим центр тяжести симплекса

$$\overline{x_c} = \frac{1}{3}(\overline{x^{(13)}} + \overline{x^{(14)}} + \overline{x^{(12)}})^T = (-0,374; 0,333)^T.$$

В полученной точке $f(\overline{x_c}) = 0,945$. Вычислим

$$|f(\overline{x^{(13)}}) - f(\overline{x_c})| = 0,041 < \varepsilon,$$

$$|f(\overline{x^{(14)}}) - f(\overline{x_c})| = 0,072 < \varepsilon,$$

$$|f(\overline{x^{(12)}}) - f(\overline{x_c})| = 0,059 < \varepsilon.$$

Так как все условия окончания поиска выполняются, то процесс итерации завершен.

В качестве приближенного решения $\overline{x^*}$ выбирается $\overline{x^{(13)}}$ $= (-0,272; 0,435)^T$, которой соответствует наименьшее значение целевой функции $f(\overline{x^{(13)}}) = 0,986$.

1.1.2. Программная реализация

Программный код выполнен на языке Java.

```
public class CoordRow {
    double[] vector;
    double funcValue;

    public CoordRow(double[] vector, double funcValue) {
        this.vector = vector;
        this.funcValue = funcValue;
    }

    public double[] getVector() {
```



```

        return vector;
    }

    public double getFuncValue() {
        return funcValue;
    }

    @Override
    public String toString() {
        StringBuilder sb = new StringBuilder();
        for (double v : vector) {
            sb.append(v);
            sb.append("| ");
        }
        sb.append(funcValue);
        sb.append("\n");
        return sb.toString();
    }
}

import java.util.ArrayList;

public abstract class SimplexMethods extends Printable {
    //Метод определяет центр тяжести симплекса без вершины index
    protected double[] getCenterVector(int index, int dimension, double[][] basis)
    {
        double[] center = new double[dimension];
        for (int i = 0; i < basis.length; i++) {
            if (i == index) continue;
            for (int j = 0; j < dimension; j++) {
                center[j] += basis[i][j];
            }
        }
        for (int j = 0; j < dimension; j++) {
            center[j] /= dimension;
        }
        return center;
    }
}

```

```
}
```

```
//Метод считает отраженный вектор
```

```
protected double[] getReflectedVector(double[] center, int index, int
dimension, double[][] basis) {
    double[] reflectedVector = new double[dimension];
    for (int j = 0; j < dimension; j++) {
        reflectedVector[j] = 2*center[j] - basis[index][j];
    }
    return reflectedVector;
}
```

```
//Метод считает центр тяжести симплекса
```

```
protected double[] getCenterSimplex(int dimension, double[][] basis) {
    double[] center = new double[dimension];
    for (int i = 0; i < basis.length; i++) {
        for (int j = 0; j < dimension; j++) {
            center[j] += basis[i][j];
        }
    }
    for (int j = 0; j < dimension; j++) {
        center[j] /= dimension + 1;
    }
    return center;
}
```

```
//Метод считает приращение 1
```

```
protected double getDelta1(int dimension, double edgeLength) {
    return (Math.sqrt(dimension + 1) - 1) / (dimension * Math.sqrt(2)) *
edgeLength;
}
```

```
//Метод считает приращение 2
```

```
protected double getDelta2(int dimension, double edgeLength) {
    return (Math.sqrt(dimension + 1) + dimension - 1) / (dimension *
Math.sqrt(2)) * edgeLength;
}
```

```

//Метод выполняет редукцию базиса
protected void reduction(int minIndex, double basis[],
ArrayList<CoordRow> arr) {
    for (int i = 0; i < basis.length; i++) {
        printVector(basis[i], "Начальный вектор: ");
        if (i == minIndex) continue;
        for (int j = 0; j < basis.length - 1; j++) {
            basis[i][j] = basis[minIndex][j] + 0.5d * (basis[i][j] -
basis[minIndex][j]);
        }
        printVector(basis[i], "Редуцированный вектор:");
        printValue(getFuncValue(basis[i]), "Численное значение: ");
        arr.add(new CoordRow(cloneVector(basis[i]), getFuncValue(basis[i])));
    }
}
}
}

```

```
import java.util.ArrayList;
```

```

public abstract class Printable {
    //Проверяет эквивалентность двух векторов
    protected boolean isEqual(double[] vector1, double[] vector2) {
        if (vector1.length != vector2.length) return false;
        for (int i = 0; i < vector1.length; i++) {
            if (vector1[i] != vector2[i]) return false;
        }
        return true;
    }
}

```

```

//Клонирует вектор
protected double[] cloneVector(double[] vector) {
    double[] tmpVector = new double[vector.length];
    for (int i = 0; i < vector.length; i++) {
        tmpVector[i] = vector[i];
    }
    return tmpVector;
}

```

```
}
```

```
//Выводит вектор
```

```
protected void printVector(double[] vector) {
    for (double v : vector) {
        System.out.printf("%-15s", roundDouble(v));
    }
    System.out.println();
}
```

```
//Выводит вектор с сообщением
```

```
protected void printVector(double[] vector, String msg) {
    System.out.println(msg);
    for (double v : vector) {
        System.out.printf("%-15s", roundDouble(v));
    }
    System.out.println();
}
```

```
//Выводит значение с сообщением
```

```
protected void printValue(double value, String msg) {
    System.out.println(msg);
    System.out.println(roundDouble(value));
}
```

```
//Выводит таблицу координат
```

```
protected void printTable(ArrayList<CoordRow> arr) {
    System.out.println();
    System.out.println("Таблица векторов: ");
    for (CoordRow row : arr) {
        double[] vector = row.getVector();
        for (double v : vector) {
            System.out.printf("%-15s", roundDouble(v));
        }
        System.out.printf("%-15s", roundDouble(row.getFuncValue()));
        System.out.println();
    }
}
```

```

}

//Выводит матрицу с сообщением
protected void printMatrix(double[][] matrix, String msg) {
    System.out.println(msg);
    for (int i = 0; i < matrix.length; i++) {
        for (int j = 0; j < matrix[i].length; j++) {
            System.out.printf("%-15s", roundDouble(matrix[i][j]));
        }
        System.out.println();
    }
}

//Округляет вещественное число до третьего знака после запятой
protected double roundDouble(double value) {
    return Math.round(value * 1000d)/1000d;
}

protected abstract double getFuncValue(double[] value);
}

import java.util.ArrayList;

public class Simplex extends SimplexMethods{
    //Длина грани
    double edgeLength = 0.5;
    //Размерность
    int dimension = 2;
    //Точность искомого значения функции
    double eps = 0.1;
    //Стартовый вектор
    double[] startVector = {1, 1};
    //Базис
    double[][] basis;
    //Массив векторов
    ArrayList<CoordRow> arr = new ArrayList<>();

```

```

public Simplex() {
    basis = new double[dimension + 1][dimension];
    basis[0] = startVector;
    arr.add(new CoordRow(basis[0], getFuncValue(basis[0])));
    initBasis();
}

public static void main(String[] args) {
    Simplex simplex = new Simplex();
    int i = 0;
    while (true) {
        System.out.println("\nИтерация " + i++);
        if (simplex.run()) break;
    }
    simplex.printTable(simplex.getArr());
}

public ArrayList<CoordRow> getArr() {
    return arr;
}

//Метод инициализирует стартовый базис
private void initBasis() {
    for (int i = 1; i < dimension + 1; i++) {
        initBasisVector(basis[i], i);
        arr.add(new CoordRow(cloneVector(basis[i]), getFuncValue(basis[i])));
    }
    System.out.println("Приращения:\nDelta1          =          "          +
roundDouble(getDelta1(dimension, edgeLength))
        + "\nDelta2 = " + roundDouble(getDelta2(dimension, edgeLength)));
    System.out.println("Исходный базис: ");
    for (int i = 0; i < basis.length; i++) {
        printVector(basis[i]);
    }
}

//Инициализирует базисный вектор
private void initBasisVector(double[] x, int index) {
    for (int i = 0; i < x.length; i++) {

```

```

        if (i + 1 == index) x[i] = startVector[i] + getDelta1(dimension,
edgeLength);
        else x[i] = startVector[i] + getDelta2(dimension, edgeLength);
    }
}
@Override
protected double getFuncValue(double[] x) {
    return 2.8d*x[1]*x[1] + 1.9d*x[0] + 2.7d*x[0]*x[0] + 1.6d - 1.9*x[1];
}

public boolean run() {
    //Определяем индекс с наибольшим значением функции
    int maxIndex = getMax(basis);
    //Считаем центр тяжести без вершины с максимальный значением
    double[] center = getCenterVector(maxIndex, dimension, basis);
    printVector(center, "Центр тяжести: ");
    //Считаем отраженный вектор относительно центра тяжести
    double[] reflected = getReflectedVector(center, maxIndex, dimension,
basis);
    printVector(reflected, "Отраженный вектор: ");
    arr.add(new CoordRow(reflected, getFuncValue(reflected)));
    //Проверяем, меньше ли значение отраженного вектора относительно
наибольшего
    if (getFuncValue(basis[maxIndex]) < getFuncValue(reflected))
        //Выполняем редукцию
        reduction(getMin(basis), basis, arr);
    else
        //Заменяем наибольший вектор на отраженный
        basis[maxIndex] = reflected;
    //Считаем центр тяжести симплекса
    double[] centerSimplex = getCenterSimplex(dimension, basis);
    printVector(centerSimplex, "Векторное значение центра тяжести
симплекса: ");
    printValue(getFuncValue(centerSimplex), "Численное значение центра
тяжести симплекса: ");
    System.out.println("Полученный базис: ");
    for (int i = 0; i < basis.length; i++) {

```

```

        printVector(basis[i]);
    }
    return isEndOfSearch(getFuncValue(centerSimplex));
}
//Метод определяет индекс вектора с yfbvtymibv значением функции
private int getMin(double[][] basis) {
    int minIndex = 0;
    double min = getFuncValue(basis[0]);
    for (int i = 0; i < basis.length; i++) {
        if (getFuncValue(basis[i]) < min) {
            min = getFuncValue(basis[i]);
            minIndex = i;
        }
    }
    return minIndex;
}
//Метод определяет индекс вектора с наибольшим значением функции
private int getMax(double[][] basis) {
    int maxIndex = 0;
    double max = getFuncValue(basis[0]);
    for (int i = 0; i < dimension + 1; i++) {
        if (max < getFuncValue(basis[i])) {
            maxIndex = i;
            max = getFuncValue(basis[i]);
        }
    }
    return maxIndex;
}
//Метод проверяет условие окончания поиска
private boolean isEndOfSearch(double centerValue) {
    for (int i = 0; i < basis.length; i++) {
        if (Math.abs(getFuncValue(basis[i]) - centerValue) > eps)
            return false;
    }
    return true;
}
}

```


Результат работы программы:

```

Приращения:
Delta1 = 0.129
Delta2 = 0.483
Исходный базис:
1.0      1.0
1.129    1.483
1.483    1.129

Итерация 0
Центр тяжести:
1.065    1.241
Отраженный вектор:
0.646    1.354
Векторное значение центра тяжести симплекса:
0.925    1.279
Численное значение центра тяжести симплекса:
7.819
Полученный базис:
1.0      1.0
1.129    1.483
0.646    1.354

Итерация 1
Центр тяжести:
0.823    1.177
Отраженный вектор:
0.517    0.871
Векторное значение центра тяжести симплекса:
0.721    1.075
Численное значение центра тяжести симплекса:
5.566
Полученный базис:
1.0      1.0
0.517    0.871
0.646    1.354

...

Итерация 9
Центр тяжести:
-0.304    0.314
Отраженный вектор:
-0.514    0.37
Векторное значение центра тяжести симплекса:
-0.374    0.333
Численное значение центра тяжести симплекса:
0.945
Полученный базис:
-0.272    0.435
-0.514    0.37
-0.337    0.194

Таблица векторов:
1.0      1.0      7.1
1.129    1.483    10.53
1.483    1.129    11.781
0.646    1.354    6.515
0.517    0.871    3.772
0.163    1.224    3.853
0.034    0.741    1.798
0.388    0.388    2.426
-0.095    0.258    1.139
-0.272    0.435    1.177
-0.337    0.194    1.206
0.034    0.741    1.798
-0.272    0.435    0.986
-0.337    0.194    1.004
-0.514    0.37    1.017

```

Рис. 1.6. Результат работы программы

1.1.3. Контрольные вопросы

1. Сформулируйте задачу многомерной оптимизации. Запишите ее символически.
2. Итерационная процедура нахождения точки локального минимума.
3. Смысл методов спуска. Чем обусловлено их название.
4. Сущность методов нулевого порядка.
5. Дайте определение регулярному симплексу. Что является симплексом при размерности равной 2, при размерности равной 3.
6. Схема поиска симплексного метода. Покажите процесс построения нового симплекса на плоскости.
7. Что происходит, если попытка отражения не приводит к уменьшению целевой функции?
8. Дайте определение линиями уровня функции. Поясните рисунком.
9. Проиллюстрируйте поиск точки минимума симплексным методом. Объясните рисунок.
10. Поясните мультимодальность функций. Проиллюстрируйте рисунком.
11. Шаги алгоритма симплексного метода.
12. Какие необходимы начальные параметры для расчета минимального значения целевой функции по симплексному методу.
13. Как вычислить координаты вершин симплекса?
14. По какой формуле определяется центр тяжести вершин многогранника?
15. По какой формуле необходимо отразить вершину относительно центра тяжести?
16. По какому условию проверяется окончание поиска для симплексного метода?
17. По какому соотношению выполняется операция редукции?
18. Как проверить условие окончания процесса вычислений?

1.2. Метод Нелдера-Мида.

В 1964 году Нелдер и Мид предложили модификацию, в которой симплекс может изменять свою форму (растягиваясь и сжимаясь) в зависимости от свойств поверхности целевой функции. Так как в этом случае симплекс не будет уже регулярным, метод назвали **поиском по деформируемому многограннику**.

И так модифицируем рассмотренный на предыдущей теме алгоритм минимизации целевой функции по регулярному симплексу, добавив к процедуре отражения при построении нового симплекса процедуры сжатия и растяжения. Геометрическая иллюстрация этих процедур для случая $n = 2$ представлена на рис. 1.7, 1.8 и 1.9, где введены следующие обозначения:

f_h – наибольшее значение целевой функции;

f_s – следующее по величине за наибольшим значение целевой функции;

f_l – наименьшее значение целевой функции;

f, f' – текущие значения целевой функции

а) Операция отображения

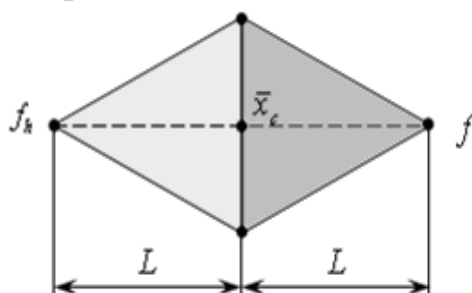


Рис.1.7. Операция отображения

б) Если $f < f_l$, то выполняется операция растяжения $L_H = \beta L$, где β – параметр растяжения.

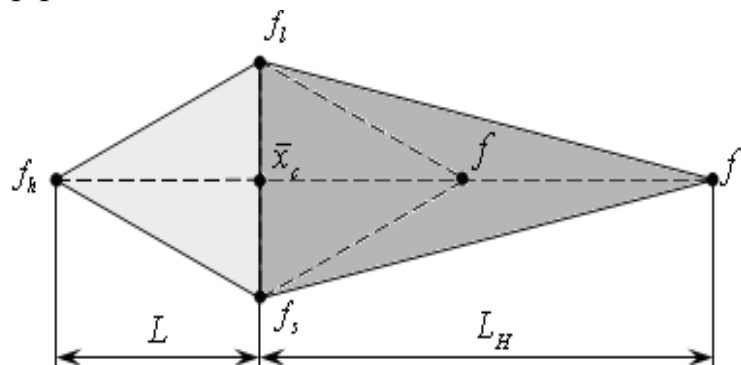


Рис.1.8. Операция растяжения

с) Если $f_s < f < f_h$, то выполняется операция сжатия $L_H = \gamma L$,

где γ – параметр сжатия.

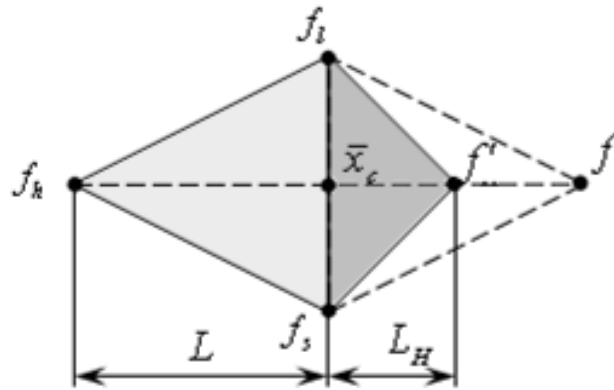


Рис.1.9. Операция сжатия

При решении практически задач минимизации параметры растяжения β и сжатия γ Нелдер и Мид рекомендует брать $\beta = 2$, $\gamma = 0,5$, но Павиани – выбирать эти параметры из интервалов $2,8 \leq \beta \leq 3,0$ и $0,4 \leq \gamma \leq 0,6$

Рассмотрим алгоритм поиска методом Нелдера-Мида

1. Задать размерность задачи оптимизации n , координаты начальной точки многогранника $\overline{x^{(0)}} = \{x_1^{(0)}, x_2^{(0)}, \dots, x_n^{(0)}\}$, длину ребра многогранника m , параметр растяжения β , параметр сжатия γ (гамма), точность поиска ϵ .

2. Построить начальный многогранник в виде регулярного симплекса, вычисляя координаты остальных n вершин $\overline{x^{(1)}}$, $\overline{x^{(2)}}$, ..., $\overline{x^{(n)}}$ по формулам:

$$\overline{x^{(i)}} = \begin{cases} x_j^{(0)} + \delta_1, & \text{если } i = j \\ x_j^{(0)} + \delta_2, & \text{если } i \neq j \end{cases} \quad \text{для } i, j = \overline{1, n}$$

где приращения δ_1 и δ_2 определяются по формулам

$$\delta_1 = \left(\frac{\sqrt{n+1}-1}{n\sqrt{2}} \right) * m, \quad \delta_2 = \left(\frac{\sqrt{n+1}+n-1}{n\sqrt{2}} \right) * m.$$

3. Определить номер вершины k с наибольшим значением целевой функции $f_h = f(\overline{x^{(k)}})$, номер вершины k_1 с наименьшим значением целевой функции $f_l = f(\overline{x^{(k_1)}})$ и номер вершины k_2 – следующим по величине за наибольшим значением целевой функции $f_s = f(\overline{x^{(k_2)}})$.

4. Определить центр тяжести всех вершин многогранника за исключением вершины $\overline{x^{(k)}}$:

$$\bar{x}_c = \frac{1}{n} \sum_{\substack{i=0 \\ i \neq k}}^n \bar{x}^{(i)}$$

10. Отобразить вершину $\bar{x}^{(k)}$ относительно центра тяжести $\bar{x} = 2\bar{x}_c - \bar{x}^{(k)}$. Вычислить значение целевой функции в отраженной точке $f(\bar{x})$ и перейти к пункту 6.

11. Проверить условие. Если $f(\bar{x}) < f(\bar{x}^{(k)})$, то операция отражения закончилась успешно. Положить $\bar{x}^{(k)} = \bar{x}$, $f(\bar{x}^{(k)}) = f(\bar{x})$ и перейти к пункту 7. В противном случае перейти к пункту 9 и выполнить операцию сжатия.

12. Проверить условие. Если $f(\bar{x}^{(k)}) < f_l$, то выполнить операцию растяжения $\bar{x} = \bar{x}_c + \beta(\bar{x}^{(k)} - \bar{x}_c)$, вычислить значение целевой функции $f(\bar{x})$ и перейти к пункту 8, иначе – к пункту 9.

13. Проверить условие. Если $f(\bar{x}) < f(\bar{x}^{(k)})$, то операция растяжения закончилась успешно. Положить $\bar{x}^{(k)} = \bar{x}$, $f(\bar{x}^{(k)}) = f(\bar{x})$ и перейти к пункту 12, иначе – к пункту 9.

14. Проверить условие. Если $f_s < f(\bar{x}) < f_h$, то выполнить операцию сжатия $\bar{x} = \bar{x}_c + \gamma(\bar{x}^{(k)} - \bar{x}_c)$ вычислить значение целевой функции $f(\bar{x})$ и перейти к пункту 10, иначе – к пункту 11.

15. Проверить условие. Если $f(\bar{x}) < f(\bar{x}^{(k)})$, то операция сжатия закончилась успешно. Положить $\bar{x}^{(k)} = \bar{x}$, $f(\bar{x}^{(k)}) = f(\bar{x})$ и перейти к пункту 12, иначе – к пункту 11.

16. Выполнить операцию редукции. Для этого определить номер вершины r с минимальным значением целевой функции $f_{min} = f(\bar{x}^{(r)})$. Используя соотношение

$$\bar{x}^{(i)} = \bar{x}^{(r)} + 0,5 \cdot (\bar{x}^{(i)} - \bar{x}^{(r)}), \quad i = 0, n, \quad i \neq r$$

сформировать новый многогранник с уменьшенными вдвое сторонами. Перейти к шагу 12.

17. Проверить критерий окончания процесса поиска, предложенный Нелдером и Мидом

$$\sigma = \left\{ \frac{1}{n+1} \sum_{i=0}^n [f(\bar{x}^{(i)}) - f(\bar{x}_c)]^2 \right\}^{\frac{1}{2}} < \varepsilon,$$

где $\bar{x}_c = \frac{1}{n+1} \sum_{i=0}^n \bar{x}^{(i)}$ - центр тяжести многогранника на данном шаге

18. Если условие выполнено $\sigma(\text{сигма}) < \varepsilon$, то процесс вычислений завершен. В качестве приближенного решения принять вершину многогранника с минимальным значением целевой функции. В противном случае перейти к шагу 3 и продолжить процесс итераций.

1.2.1. Практический расчет задачи

Постановка задачи. Найти минимум целевой функции.

$$f(\bar{x}) = 2.8 * x_2^2 + 1.9 * x_1 + 2.7 * x_1^2 + 1.6 - 1.9 * x_2$$

методом Нелдера-Мида с точностью $\varepsilon = 0,1$.

Решение. Зададим начальную точку симплекса $\bar{x}^{(0)} = (x_1^{(0)}, x_2^{(0)})^T = (0, 0)^T$ и длину ребра симплекса $m = 0,75$, параметр растяжения $\beta = 1,85$, параметр сжатия $\gamma = 0,1$

Вычислим приращения

$$\delta_1 = \left(\frac{\sqrt{n+1}-1}{n\sqrt{2}} \right) * m = \left(\frac{\sqrt{3}-1}{2\sqrt{2}} \right) * 0.75 = 0,194,$$

$$\delta_2 = \left(\frac{\sqrt{n+1}+n-1}{n\sqrt{2}} \right) * m = \left(\frac{\sqrt{3}+2-1}{2\sqrt{2}} \right) * 0.75 = 0,724.$$

Используя δ_1 и δ_2 , вычислим координаты двух остальных вершин симплекса

$$\bar{x}^{(1)} = (x_1^{(0)} + \delta_1, x_2^{(0)} + \delta_2)^T = (0,194; 0,724)^T$$

$$\bar{x}^{(2)} = (x_1^{(0)} + \delta_2, x_2^{(0)} + \delta_1)^T = (0,724; 0,194)^T$$

Итерация $k = 0$. Вычислим значение целевой функции $f(\bar{x})$ в вершинах $\bar{x}^{(0)}$, $\bar{x}^{(1)}$, $\bar{x}^{(2)}$ и обозначим наибольшее значение функции f_h , следующее за наибольшим значением f_s , наименьшее значение функции f_l

$$f_l = f(\bar{x}^{(0)}) = 1,6, f_s = f(\bar{x}^{(1)}) = 2,164, f_h = f(\bar{x}^{(2)}) = 4,13.$$

Отообразим эти значения в таблице 1.

Таблица 1.

Точка	Координаты точки		Значение функции
	1	2	
$x^{(0)}$	$x_1^{(0)} = 0$	$x_2^{(0)} = 0$	$f(\bar{x}^{(0)}) = 1,6$
$x^{(1)}$	$x_1^{(1)} = 0,194$	$x_2^{(1)} = 0,724$	$f(\bar{x}^{(1)}) = 2,164$
$x^{(2)}$	$x_1^{(2)} = 0,724$	$x_2^{(2)} = 0,194$	$f(\bar{x}^{(2)}) = 4,13$

Наибольшее значение целевой функции соответствует вершине $\bar{x}^{(2)}$, поэтому необходимо отразить ее относительно центра тяжести остальных вершин $\bar{x}^{(0)}$ и $\bar{x}^{(1)}$. центр тяжести расположен в точке

$$\overline{x_{c2}} = \frac{1}{2}(x_1^{(0)} + x_1^{(1)}, x_2^{(0)} + x_2^{(1)})^T = (0,097; 0,362)^T.$$

Используя свойство регулярности, найдем координаты отраженной вершины

$$\overline{x^{(3)}} = 2\overline{x_{c2}} - \overline{x^{(2)}} = (-0,53; 0,53)^T.$$

В полученной вершине значение целевой функции $f(\overline{x^{(3)}}) = 1,132$. Составим таблицу 2.

Таблица 2.

Точка	Координаты точки		Значение функции
	1	2	
$x^{(0)}$	$x_1^{(0)} = 0$	$x_2^{(0)} = 0$	$f(\overline{x^{(0)}}) = 1,6$
$x^{(1)}$	$x_1^{(1)} = 0,194$	$x_2^{(1)} = 0,724$	$f(\overline{x^{(1)}}) = 2,164$
$x^{(2)}$	$x_1^{(2)} = 0,724$	$x_2^{(2)} = 0,194$	$f(\overline{x^{(2)}}) = 4,13$
$x^{(3)}$	$x_1^{(3)} = -0,53$	$x_2^{(3)} = 0,53$	$f(\overline{x^{(3)}}) = 1,132$

Следовательно, наблюдается уменьшение целевой функции $f(\overline{x^{(3)}}) < f(\overline{x^{(1)}})$

Так как не выполняется условие

$$f(\overline{x^{(1)}}) < f(\overline{x^{(3)}}) < f(\overline{x^{(4)}})$$

то выполним операцию растяжения симплекса.

$$\overline{x^{(4)}} = \overline{x_{c2}} + 1.85 * (\overline{x^{(3)}} - \overline{x_{c2}}) = (-1,064; 0,673)^T$$

В полученной вершине значение целевой функции $f(\overline{x^{(4)}}) = 2,623$.

Учитывая, что условие растяжения $f(\overline{x^{(4)}}) < f(\overline{x^{(3)}})$ не выполнено, то проведем редукцию.

Сформируем новый многогранник с уменьшенными вдвое сторонами и вершиной $\overline{x^{(3)}}$, которой соответствует наименьшее значение целевой функции $f_l = f(\overline{x^{(3)}}) = 1,132$. Составим таблицу 3.

Таблица 3.

Точка	Координаты точки		Значение функции
	1	2	
$x^{(0)}$	$x_1^{(0)} = 0$	$x_2^{(0)} = 0$	$f(\overline{x^{(0)}}) = 1,6$
$x^{(1)}$	$x_1^{(1)} = 0,194$	$x_2^{(1)} = 0,724$	$f(\overline{x^{(1)}}) = 2,164$
$x^{(2)}$	$x_1^{(2)} = 0,724$	$x_2^{(2)} = 0,194$	$f(\overline{x^{(2)}}) = 4,13$

$x^{(3)}$	$x_1^{(3)} = -0,53$	$x_2^{(3)} = 0,53$	$f(\overline{x^{(3)}}) = 1,132$
$x^{(4)}$	$x_1^{(4)} = -0,265$	$x_2^{(4)} = 0,265$	$f(\overline{x^{(4)}}) = 0,979$
$x^{(5)}$	$x_1^{(5)} = -0,168$	$x_2^{(5)} = 0,627$	$f(\overline{x^{(5)}}) = 1,267$

$$\overline{x^{(4)}} = \overline{x^{(3)}} + 0,5 * (\overline{x^{(0)}} - \overline{x^{(3)}}) = (-0,265; 0,265)^T.$$

$$f(\overline{x^{(4)}}) = 0,979.$$

$$\overline{x^{(5)}} = \overline{x^{(3)}} + 0,5 * (\overline{x^{(1)}} - \overline{x^{(3)}}) = (-0,168; 0,627)^T.$$

$$f(\overline{x^{(5)}}) = 1,267.$$

После операции редукции текущий многогранник образован вершинами $\overline{x^{(4)}}$, $\overline{x^{(5)}}$, $\overline{x^{(3)}}$ которым соответствует значение целевой функции

$$f(\overline{x^{(4)}}) = 0,979, f(\overline{x^{(5)}}) = 1,267, f(\overline{x^{(3)}}) = 1,132$$

Проверим условие окончания поиска. Определим координаты центра тяжести симплекса

$$\overline{x_c} = \frac{1}{3}(\overline{x^{(3)}} + \overline{x^{(4)}} + \overline{x^{(5)}})^T = (-0,321; 0,474)^T.$$

В полученной точке $f(\overline{x_c}) = 0,997$.

Вычислим σ (сигма)

$$\left\{ \frac{1}{n+1} \cdot \sum_{i=0}^2 [f(\overline{x^{(i)}}) - f(\overline{x_c})]^2 \right\}^{\frac{1}{2}}$$

$$\sigma = 0,174 < \varepsilon$$

Так как условие окончания поиска не выполняется, то процесс итерации должен быть продолжен.

Итерация $k = 1$. Вычислим значение целевой функции $f(\overline{x})$ в вершинах $\overline{x^{(4)}}$, $\overline{x^{(5)}}$, $\overline{x^{(3)}}$ и обозначим наибольшее значение функции f_h , следующее за наибольшим значением f_s , наименьшее значение функции f_l

$$f_l = f(\overline{x^{(4)}}) = 0,979, f_s = f(\overline{x^{(3)}}) = 1,132, f_h = f(\overline{x^{(5)}}) = 1,267.$$

Наибольшее значение целевой функции соответствует вершине $\overline{x^{(5)}}$, поэтому необходимо отразить ее относительно центра тяжести остальных вершин $\overline{x^{(4)}}$ и $\overline{x^{(3)}}$. центр тяжести расположен в точке

$$\overline{x_{c5}} = \frac{1}{2}(\overline{x_1^{(3)}} + \overline{x_1^{(4)}}; \overline{x_2^{(3)}} + \overline{x_2^{(4)}})^T = (-0,398; 0,398)^T.$$

Используя свойство регулярности, найдем координаты отраженной вершины

$$\overline{x^{(6)}} = 2\overline{x_{c5}} - \overline{x^{(5)}} = (-0,627; 0,168)^T.$$

В полученной вершине значение целевой функции $f(\overline{x^{(6)}}) = 1,23$.
Составить таблицу 4.

Таблица 4.

Точка	Координаты точки		Значение функции
	1	2	
$x^{(0)}$	$x_1^{(0)} = 0$	$x_2^{(0)} = 0$	$f(\overline{x^{(0)}}) = 1,6$
$x^{(1)}$	$x_1^{(1)} = 0,194$	$x_2^{(1)} = 0,724$	$f(\overline{x^{(1)}}) = 2,164$
$x^{(2)}$	$x_1^{(2)} = 0,724$	$x_2^{(2)} = 0,194$	$f(\overline{x^{(2)}}) = 4,13$
$x^{(3)}$	$x_1^{(3)} = -0,53$	$x_2^{(3)} = 0,53$	$f(\overline{x^{(3)}}) = 1,132$
$x^{(4)}$	$x_1^{(4)} = -0,265$	$x_2^{(4)} = 0,265$	$f(\overline{x^{(4)}}) = 0,979$
$x^{(5)}$	$x_1^{(5)} = -0,168$	$x_2^{(5)} = 0,627$	$f(\overline{x^{(5)}}) = 1,267$
$x^{(6)}$	$x_1^{(6)} = -0,627$	$x_2^{(6)} = 0,168$	$f(\overline{x^{(6)}}) = 1,23$

Следовательно, наблюдается уменьшение целевой функции $f(\overline{x^{(3)}}) < f(\overline{x^{(5)}})$

Так как выполняется условие

$$f(\overline{x^{(3)}}) < f(\overline{x^{(6)}}) < f(\overline{x^{(5)}})$$

то выполним операцию сжатия симплекса.

$$\overline{x^{(7)}} = \overline{x_{c5}} + 1.85 * (\overline{x^{(6)}} - \overline{x_{c5}}) = (-0,421; 0,365)^T$$

В полученной вершине значение целевой функции $f(\overline{x^{(7)}}) = 0,96$.

Учитывая, что условие сжатия $f(\overline{x^{(7)}}) < f(\overline{x^{(5)}})$ выполнено, то добавим значение в таблицу 5.

Таблица 5.

Точка	Координаты точки		Значение функции
	1	2	
$x^{(0)}$	$x_1^{(0)} = 0$	$x_2^{(0)} = 0$	$f(\overline{x^{(0)}}) = 1,6$
$x^{(1)}$	$x_1^{(1)} = 0,194$	$x_2^{(1)} = 0,724$	$f(\overline{x^{(1)}}) = 2,164$
$x^{(2)}$	$x_1^{(2)} = 0,724$	$x_2^{(2)} = 0,194$	$f(\overline{x^{(2)}}) = 4,13$
$x^{(3)}$	$x_1^{(3)} = -0,53$	$x_2^{(3)} = 0,53$	$f(\overline{x^{(3)}}) = 1,132$
$x^{(4)}$	$x_1^{(4)} = -0,265$	$x_2^{(4)} = 0,265$	$f(\overline{x^{(4)}}) = 0,979$

$x^{(5)}$	$x_1^{(5)} = -0,168$	$x_2^{(5)} = 0,627$	$f(\overline{x^{(5)}}) = 1,267$
$x^{(6)}$	$x_1^{(6)} = -0,627$	$x_2^{(6)} = 0,168$	$f(\overline{x^{(6)}}) = 1,23$
$x^{(7)}$	$x_1^{(7)} = -0,421$	$x_2^{(7)} = 0,365$	$f(\overline{x^{(7)}}) = 0,96$

Проверим условие окончания поиска. Определим координаты центра тяжести симплекса

$$\overline{x_c} = \frac{1}{3}(\overline{x^{(3)}} + \overline{x^{(4)}} + \overline{x^{(7)}})^T = (-0,405; 0,39)^T.$$

В полученной точке $f(\overline{x_c}) = 0,958$.

Вычислим σ (сигма)

$$\left\{ \frac{1}{n+1} \cdot \sum_{i=0}^2 [f(\overline{x^{(i)}}) - f(\overline{x_c})]^2 \right\}^{\frac{1}{2}}$$

$$\sigma = 0,101 < \varepsilon$$

Так как условие окончания поиска не выполняется, то процесс итерации должен быть продолжен.

Итерация $k = 2$. Вычислим значение целевой функции $f(\overline{x})$ в вершинах $\overline{x^{(4)}}$, $\overline{x^{(7)}}$, $\overline{x^{(3)}}$ и обозначим наибольшее значение функции f_h , следующее за наибольшим значением f_s , наименьшее значение функции f_l
 $f_l = f(\overline{x^{(7)}}) = 0,96$, $f_s = f(\overline{x^{(4)}}) = 0,979$, $f_h = f(\overline{x^{(3)}}) = 1,132$.

Наибольшее значение целевой функции соответствует вершине $\overline{x^{(3)}}$, поэтому необходимо отразить ее относительно центра тяжести остальных вершин $\overline{x^{(4)}}$ и $\overline{x^{(7)}}$. центр тяжести расположен в точке

$$\overline{x_{c3}} = \frac{1}{2}(\overline{x_1^{(7)}} + \overline{x_1^{(4)}}; \overline{x_2^{(7)}} + \overline{x_2^{(4)}})^T = (-0,343; 0,32)^T.$$

Используя свойство регулярности, найдем координаты отраженной вершины

$$\overline{x^{(8)}} = 2\overline{x_{c3}} - \overline{x^{(3)}} = (-0,156; 0,11)^T.$$

В полученной вершине значение целевой функции $f(\overline{x^{(8)}}) = 1,195$.

Так как не выполняется условие

$$f(\overline{x^{(4)}}) < f(\overline{x^{(8)}}) < f(\overline{x^{(3)}})$$

И не выполняется условие $f(\overline{x^{(8)}}) < f(\overline{x^{(3)}})$ то проведем редукцию.

Сформируем новый многогранник с уменьшенными вдвое сторонами и вершиной $\overline{x^{(7)}}$, которой соответствует наименьшее значение целевой функции $f_l = f(\overline{x^{(7)}}) = 0,96$. Составить таблицу 6.

Таблица 6.

Точка	Координаты точки		Значение функции
	1	2	
$x^{(0)}$	$x_1^{(0)} = 0$	$x_2^{(0)} = 0$	$f(\overline{x^{(0)}}) = 1,6$
$x^{(1)}$	$x_1^{(1)} = 0,194$	$x_2^{(1)} = 0,724$	$f(\overline{x^{(1)}}) = 2,164$
$x^{(2)}$	$x_1^{(2)} = 0,724$	$x_2^{(2)} = 0,194$	$f(\overline{x^{(2)}}) = 4,13$
$x^{(3)}$	$x_1^{(3)} = -0,53$	$x_2^{(3)} = 0,53$	$f(\overline{x^{(3)}}) = 1,132$
$x^{(4)}$	$x_1^{(4)} = -0,265$	$x_2^{(4)} = 0,265$	$f(\overline{x^{(4)}}) = 0,979$
$x^{(5)}$	$x_1^{(5)} = -0,168$	$x_2^{(5)} = 0,627$	$f(\overline{x^{(5)}}) = 1,267$
$x^{(6)}$	$x_1^{(6)} = -0,627$	$x_2^{(6)} = 0,168$	$f(\overline{x^{(6)}}) = 1,23$
$x^{(7)}$	$x_1^{(7)} = -0,421$	$x_2^{(7)} = 0,365$	$f(\overline{x^{(7)}}) = 0,96$
$x^{(8)}$	$x_1^{(8)} = -0,343$	$x_2^{(8)} = 0,32$	$f(\overline{x^{(8)}}) = 0,945$
$x^{(9)}$	$x_1^{(9)} = -0,476$	$x_2^{(9)} = 0,453$	$f(\overline{x^{(9)}}) = 1,021$

$$\overline{x^{(8)}} = \overline{x^{(7)}} + 0,5 * (\overline{x^{(4)}} - \overline{x^{(7)}}) = (-0,343; 0,32)^T.$$

$$f(\overline{x^{(8)}}) = 0,945.$$

$$\overline{x^{(9)}} = \overline{x^{(7)}} + 0,5 * (\overline{x^{(3)}} - \overline{x^{(7)}}) = (-0,476; 0,453)^T.$$

$$f(\overline{x^{(9)}}) = 1,021.$$

После операции редукции текущий многогранник образован вершинами $\overline{x^{(9)}}$, $\overline{x^{(8)}}$, $\overline{x^{(7)}}$ которым соответствует значение целевой функции

$$f(\overline{x^{(9)}}) = 1,021, f(\overline{x^{(8)}}) = 0,945, f(\overline{x^{(7)}}) = 0,96$$

Проверим условие окончания поиска. Определим координаты центра тяжести симплекса

$$\overline{x_c} = \frac{1}{3}(\overline{x^{(9)}} + \overline{x^{(8)}} + \overline{x^{(7)}})^T = (-0,413; 0,382)^T.$$

В полученной точке $f(\overline{x_c}) = 0,959$.

Вычислим σ (сигма)

$$\left\{ \frac{1}{n+1} \cdot \sum_{i=0}^2 [f(\overline{x^{(i)}}) - f(\overline{x_c})]^2 \right\}^{\frac{1}{2}}$$

$$\sigma = 0,037 < \varepsilon$$

Так как условие окончания поиска выполняется, то процесс итераций завершен.

В качестве приближенного решения $\overline{x^*}$ выбирается $\overline{x^{(8)}}$ $= (-0,343; 0,32)^T$, которой соответствует наименьшее значение целевой функции $f(\overline{x^{(8)}}) = 0,945$ текущего симплекса $\overline{x^{(9)}}$, $\overline{x^{(8)}}$, $\overline{x^{(7)}}$.

1.2.2. Программная реализация

Программный код выполнен на языке Java.

```
import java.util.ArrayList;
```

```
public class SimplexDeformation extends SimplexMethods {
```

```
    //Длина грани
```

```
    double edgeLength = 0.75;
```

```
    //Размерность
```

```
    int dimension = 2;
```

```
    //Точность искомого значения функции
```

```
    double eps = 0.1;
```

```
    //Стартовый вектор
```

```
    double[] startVector = {0, 0};
```

```
    //Базис
```

```
    double[][] basis;
```

```
    //Параметр растяжения
```

```
    double beta = 1.85;
```

```
    //Параметр сжатия
```

```
    double gamma = 0.1;
```

```
    //Массив векторов
```

```
    ArrayList<CoordRow> arr = new ArrayList<>();
```

```
    public SimplexDeformation() {
```

```
        basis = new double[dimension + 1][dimension];
```

```
        basis[0] = startVector;
```

```
        arr.add(new CoordRow(cloneVector(basis[0]), getFuncValue(basis[0])));
```

```
        initBasis();
```

```
    }
```

```
    public static void main(String[] args) {
```

```

SimplexDeformation simplex = new SimplexDeformation();
int i = 0;
while (true) {
    System.out.println("\nИтерация " + i++);
    if (simplex.run()) break;
}
simplex.printTable(simplex.getArr());
}

public ArrayList<CoordRow> getArr() {
    return arr;
}

@Override
protected double getFuncValue(double[] x) {
    return 2.8d*x[1]*x[1] + 1.9d*x[0] + 2.7d*x[0]*x[0] + 1.6d - 1.9*x[1];
}

//Метод инициализирует стартовый базис
private void initBasis() {
    for (int i = 1; i < dimension + 1; i++) {
        initBasisVector(basis[i], i);
        arr.add(new CoordRow(cloneVector(basis[i]), getFuncValue(basis[i])));
    }
    System.out.println("Приращения:\nDelta1          =          "          +
roundDouble(getDelta1(dimension, edgeLength))
        + "\nDelta2 = " + roundDouble(getDelta2(dimension, edgeLength)));
    System.out.println("Исходный базис: ");
    for (int i = 0; i < basis.length; i++) {
        printVector(basis[i]);
    }
}

//Инициализирует базисный вектор
private void initBasisVector(double[] x, int index) {
    for (int i = 0; i < x.length; i++) {

```

```

    if (i + 1 == index) x[i] = startVector[i] + getDelta1(dimension, edgeLength);
    else x[i] = startVector[i] + getDelta2(dimension, edgeLength);
  }
}

```

```

private boolean run() {
    //Определяем индекс с наибольшим значением функции
    int maxIndex = getMax(basis);
    //Определяем индекс с наименьшим значением функции
    int minIndex = getMin(basis);
    //Определяем индекс с преднаибольшим значением функции
    int maxBeforeIndex = getBeforeMax(basis, maxIndex);
    //Считаем центр тяжести без вершины с максимальный значением
    double[] center = getCenterVector(maxIndex, dimension, basis);
    printVector(center, "Центр тяжести: ");
    //Считаем отраженный вектор относительно центра тяжести
    double[] reflected = getReflectedVector(center, maxIndex, dimension, basis);
    printVector(reflected, "Векторное значение отраженного вектора: ");
    printValue(getFuncValue(reflected), "Численное значение отраженного вектора: ");
    ");
    arr.add(new CoordRow(cloneVector(reflected), getFuncValue(reflected)));
    //Проверяем, меньше ли значение отраженного вектора относительно
    наибольшего
    if (getFuncValue(reflected) < getFuncValue(basis[maxIndex])) {
        basis[maxIndex] = cloneVector(reflected);
        //Проверяем, меньше или равно значение отраженного вектора
        относительно преднаибольше
        if (getFuncValue(basis[maxBeforeIndex]) >= getFuncValue(reflected)) {
            //Проверяем, меньше ли значение отраженного вектора относительно
            наименьшего
            if (getFuncValue(reflected) < getFuncValue(basis[minIndex])) {
                //Растягиваем вектор
                double[] tmpVector = changeVector(center, reflected, beta);
                printVector(tmpVector, "Векторное значение растяженного вектора: ");
                printValue(getFuncValue(tmpVector), "Численное значение растяженного
                вектора: ");
            }
        }
    }
}

```

```

        //Проверяем, меньше ли значение растяженного вектора относительно
наименьшего
        if (getFuncValue(tmpVector) < getFuncValue(reflected)) {
            basis[maxIndex] = tmpVector;
            arr.add(new CoordRow(cloneVector(tmpVector),
getFuncValue(tmpVector)));
            //Проверяем условие окончания поиска
            return isEndOfSearch();
        }
    }
} else {
    //Сжимаем вектор
    double[] tmpVector = changeVector(center, reflected, gamma);
    printVector(tmpVector, "Векторное значение сжатого вектора: ");
    printValue(getFuncValue(tmpVector), "Численное значение сжатого
вектора: ");
    arr.add(new CoordRow(cloneVector(tmpVector), getFuncValue(tmpVector)));
    //Проверяем, меньше ли значение сжатого вектора относительно
наименьшего
    if (getFuncValue(tmpVector) < getFuncValue(reflected)) {
        basis[maxIndex] = tmpVector;
        //Проверяем окончание поиска
        return isEndOfSearch();
    }
}
}
//Выполняем редукцию
reduction(getMin(basis), basis, arr);
//Проверяем окончание поиска
return isEndOfSearch();
}

//Метод проверяющий условие окончания поиска
private boolean isEndOfSearch() {
    double sigma = 0;
    double[] center = getCenterSimplex(dimension, basis);

```

```

printVector(center, "Векторное значение центра тяжести симплекса: ");
printValue(getFuncValue(center), "Численное значение центра тяжести
симплекса: ");
for (int i = 0; i <= dimension; i++) {
    sigma += Math.pow(getFuncValue(basis[i]) - getFuncValue(center), 2);
}
sigma = Math.sqrt(sigma/(dimension+1));
System.out.println("Сигма:\n" + roundDouble(sigma));
if (sigma < eps) return true;
else return false;
}

```

//Метод растягивающий или сжимающий вектор

```

private double[] changeVector(double[] center, double[] reflected, double coefficient)
{
    double[] stretchVector = new double[dimension];
    for (int i = 0; i < dimension; i++) {
        stretchVector[i] = center[i] + coefficient * (reflected[i] - center[i]);
    }
    return stretchVector;
}

```

//Метод определяет индекс вектора с наибольшим значением функции

```

private int getMax(double[][] basis) {
    int maxIndex = 0;
    double max = getFuncValue(basis[0]);
    for (int i = 0; i < dimension + 1; i++) {
        if (max < getFuncValue(basis[i])) {
            maxIndex = i;
            max = getFuncValue(basis[i]);
        }
    }
    return maxIndex;
}

```

//Метод определяет индекс вектора с наименьшим значением функции


```
private int getMin(double[][] basis) {
    int minIndex = 0;
    double min = getFuncValue(basis[0]);
    for (int i = 0; i < dimension + 1; i++) {
        if (min > getFuncValue(basis[i])) {
            minIndex = i;
            min = getFuncValue(basis[i]);
        }
    }
    return minIndex;
}
```

//Метод определяет индекс вектора с преднаибольшим значением функции

```
private int getBeforeMax(double[][] basis, int maxIndex) {
    int maxBeforeIndex = maxIndex == 0 ? 1 : 0;
    double maxBefore = getFuncValue(basis[maxBeforeIndex]);
    for (int i = 0; i < dimension + 1; i++) {
        if (maxBefore < getFuncValue(basis[i]) && maxIndex != i) {
            maxBeforeIndex = i;
            maxBefore = getFuncValue(basis[i]);
        }
    }
    return maxBeforeIndex;
}
```

Результат работы программы (рис. 1.10):

```

Итерация 2
Центр тяжести:
-0.343      0.32
Векторное значение отраженного вектора:
-0.156      0.11
Численное значение отраженного вектора:
1.195
Начальный вектор:
-0.265      0.265
Редуцированный вектор:
-0.343      0.32
Численное значение:
0.945
Начальный вектор:
-0.421      0.375
Начальный вектор:
-0.53       0.53
Редуцированный вектор:
-0.476      0.453
Численное значение:
1.021
Векторное значение центра тяжести симплекса:
-0.413      0.382
Численное значение центра тяжести симплекса:
0.959
Сигма:
0.037

Таблица векторов:
0.0      0.0      1.6
0.194    0.724    2.164
0.724    0.194    4.13
-0.53    0.53     1.132
-0.265   0.265    0.979
-0.168   0.627    1.267
-0.627   0.168    1.23
-0.421   0.375    0.96
-0.156   0.11     1.195
-0.343   0.32     0.945
-0.476   0.453    1.021

```

Рис. 1.10. Результат работы программы

1.2.3. Контрольные вопросы

1. Сущность метода Нелдера-Мида, как поиск по деформируемому многограннику.
2. Проиллюстрируйте процедуру сжатия при построении нового симплекса.
3. Проиллюстрируйте процедуру растяжения при построении нового симплекса.
4. Шаги алгоритма метода Нелдера-Мида.
5. Какие необходимы начальные параметры для расчета минимального значения целевой функции по методу Нелдера-Мида.
6. Проверка условия операции сжатия и растяжения.
7. Критерий окончания процесса поиска.

1.3. Метод Хука-Дживса

Метод Хука – Дживса (метод конфигураций, метод пробных шагов) относится, с одной стороны, к классу прямых методов оптимизации, а с другой стороны – к классу детерминированных методов оптимизации. Метод предназначен для решения многомерных задач локальной безусловной оптимизации.

В методе Хука-Дживса поиск минимума состоит из последовательности шагов исследующего поиска относительно базисной точки и поиска по образцу.

Рассмотрим для примера первую итерацию (последующие итерации выполняются по такой же схеме)

Исследующий поиск состоит в следующем. Задаются некоторой начальной (базисной) точкой $\overline{x^{(0)}}$ и величиной шага h для каждого координатного направления ($i = \overline{1, n}$). Обследуют окрестность данной точки, изменяя по очереди компоненты вектора $\overline{x^{(0)}}$ вдоль каждого координатного направления. Для этого вычисляется значение целевой функции $f(\overline{x^{(0)}} + h_i \overline{e}_i)$ в пробной точке $\overline{x^{(0)}} + h_i \overline{e}_i$, где \overline{e}_i – единичный вектор в направлении оси x_i . Если значение целевой функции в пробной точке меньше значения целевой функции в базисной точке $\overline{x^{(0)}}$, то выбранный шаг h_i считается удачным, и точка $\overline{x^{(0)}}$ заменяется на $\overline{x^{(0)}} + h_i \overline{e}_i$. В противном случае вычисляется величина $\overline{x^{(0)}} - h_i \overline{e}_i$, и если значение целевой функции уменьшается, то $\overline{x^{(0)}}$ заменяется на $\overline{x^{(0)}} - h_i \overline{e}_i$. После перебора всех координатных направлений ($i = \overline{1, n}$) исследующий поиск завершается. Полученную в результате точку $\overline{x^{(1)}}$ называют новым базисом. Если в точке нового базиса $\overline{x^{(1)}}$ уменьшение значения целевой функции не было достигнуто, то исследующий поиск повторяется вокруг той же базисной точки $\overline{x^{(0)}}$ но с меньшей величиной шага. Поиск завершается, когда все текущие величины шага будут меньше заданной точности. Если исследующий поиск был удачен, т.е. $f(\overline{x^{(1)}}) < f(\overline{x^{(0)}})$, то производится поиск по образцу.

Поиск по образцу производится из точки $\overline{x^{(1)}}$ в направлении вектора $(\overline{x^{(1)}} - \overline{x^{(0)}})$, поскольку это направление привело к уменьшению значения целевой функции. Координаты новой точки определяются в соответствии с формулой

$$\overline{x^{(2)}} = \overline{x^{(1)}} + \alpha(\overline{x^{(1)}} - \overline{x^{(0)}})$$

где $\alpha > 0$ – ускоряющий множитель

Если в результате получена точка с меньшим значением целевой функции, то она рассматривается как новая базисная точка. Если поиск по образцу был неудачен, то происходит возврат в новый базис $\overline{x^{(1)}}$, где продолжается исследующий поиск с целью выявления нового направления минимизации.

Изобразим схематично шаги метода Хука-Дживса для функций двух переменных (рис.1). Пунктирными линиями схематично отображаются шаги исследующего поиска вокруг базисной точки, сплошными – шаги удачного поиска

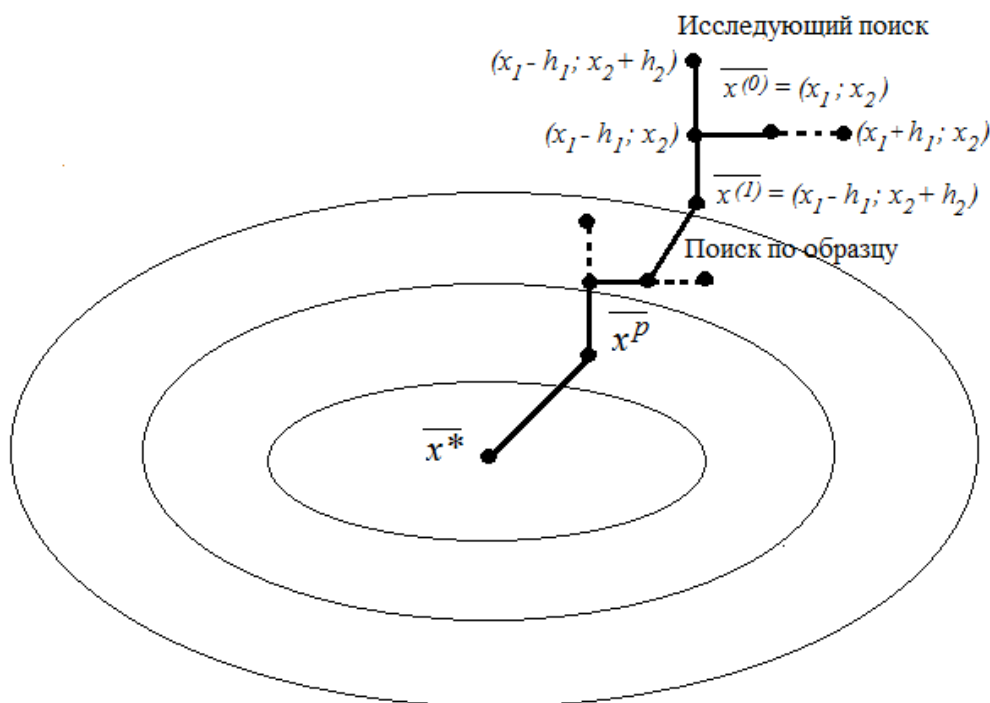


Рис.1.11. Графическая иллюстрация поиска точки минимума методом Хука-Дживса

Рассмотрим алгоритм метода Хука-Дживса

1. Задать размерность задачи оптимизации n , координаты начальной базисной точки $\overline{x^{(0)}} = \{x_1^{(0)}, x_2^{(0)}, \dots, x_n^{(0)}\}$, шаг h , для каждой переменной ($i = \overline{1, n}$), коэффициент уменьшения шага d ($d > 1$), ускоряющий множитель m ($m > 0$), точность поиска ε .

2. Ввести в рассмотрение текущую точку $\overline{x^{(1)}} = \{x_1^{(1)},$

$$\{x_2^{(1)}, \dots, x_n^{(1)}\}$$

3. Положить $\overline{x^{(1)}} = \overline{x^{(0)}}$. Вычислить значение функции $f(\overline{x^{(1)}})$ в точке $\overline{x^{(1)}}$.

4. Зафиксировать первое координатное направление $i = 1$.

5. Провести исследующий поиск вдоль оси x_i . Сделать шаг h_i в положительном направлении координатной оси $\overline{x_i^{(1)}} = \overline{x_i^{(1)}} + h_i$ и вычислить значение функции $f(\overline{x^{(1)}})$ в полученной точке $\overline{x^{(1)}}$.

6. Если $f(\overline{x^{(1)}}) < f(\overline{x^{(0)}})$, то шаг в положительном направлении оси x_i считается удачным, и осуществляется переход к пункту 8. В противном случае происходит возврат в исходную точку и делается шаг в отрицательном направлении координатной оси $\overline{x_i^{(1)}} = \overline{x_i^{(1)}} - 2h_i$.

7. Если $f(\overline{x^{(1)}}) < f(\overline{x^{(0)}})$, то шаг в положительном направлении оси x_i считается удачным. Если условие не выполнено, то происходит возврат в исходную точку $\overline{x_i^{(1)}} = \overline{x_i^{(1)}} + h_i$. И в том и другом случае осуществляется переход к следующему пункту 8.

8. Проверить условие окончания исследующего поиска. Если $i < n$, то положить $i = i + 1$ и перейти к пункту 5 для продолжения исследующего поиска по оставшимся координатным направлениям. Если $i = n$ перейти к пункту 9.

9. Проверить успешность исследующего поиска. Если $\overline{x^{(1)}} = \overline{x^{(0)}}$, т.е. исследующий поиск был неудачным, то необходимо уменьшить величину шага $h_i = h_i/d$ ($i = \overline{1, n}$) и проверить поиск с пункта 4.

10. Если $\overline{x^{(1)}} \neq \overline{x^{(0)}}$, то проводится поиск по образцу

$$\overline{x^{(p)}} = \overline{x^{(1)}} + m(\overline{x^{(1)}} - \overline{x^{(0)}})$$

и вычисляется значение функции в точке образца $f(\overline{x^{(p)}})$

11. Проверить удачность поиска по образцу. Если $f(\overline{x^{(p)}}) < f(\overline{x^{(1)}})$, то поиск по образцу удачен, и точка $\overline{x^{(0)}} = \overline{x^{(p)}}$ становится новой базисной точкой. В противном случае за базис применяется точка $\overline{x^{(0)}} = \overline{x^{(1)}}$.

12. Проверить условие окончания поиска. Если длины шагов $h_i \leq \varepsilon$ ($i = \overline{1, n}$), то поиск завершен $\overline{x^*} = \overline{x^{(1)}}$, $f_{min} = f(\overline{x^*})$. В противном случае осуществляется переход к пункту 3 и проводится исследующий поиск

вокруг новой базисной точки $\overline{x^{(0)}}$.

1.3.1. Практический расчет задачи

Постановка задачи. Найти минимум целевой функции.

$$f(\overline{x}) = 2.8 * x_2^2 + 1.9 * x_1 + 2.7 * x_1^2 + 1.6 - 1.9 * x_2$$

методом Хука-Дживса с точностью $\varepsilon = 0,1$.

Решение. Зададим начальную (базисную) точку $\overline{x^{(0)}} = (x_1^{(0)}; x_2^{(0)}) = (1, 1)$, шаг по координатным направлениям $h = 0,2$ (принят постоянным), коэффициент уменьшения шага $d = 2$.

Итерация 1. Проверим исследующий поиск вокруг базисной точки $\overline{x^{(0)}}$, которой соответствует значение целевой функции $f(\overline{x^{(0)}}) = 7.1$. Фиксируя координату $x_2^{(0)}$ и делая шаг в положительном направлении координатной оси x_1 , получим пробную точку $(\overline{x_1^{(0)}} + h, \overline{x_2^{(0)}}) = (1 + 0,2; 1) = (1,2; 1)$. Так как $f(1,2; 1) = 8.668 > f(\overline{x^{(0)}}) = 7.1$, то шаг в этом направлении считается неудачным. Возвращаемся на исходную точку и делаем шаг в отрицательном направлении оси x_1 : $(1 - h; 1) = (0,8; 1)$. Так как $f(0,8; 1) = 5.748 < f(\overline{x^{(0)}}) = 7.1$, то шаг в этом направлении считается удачным.

Фиксируем пробную точку $x^{(1)} = (0,8; 1)$ и принимаем ее (табл.1).

Таблица 1.

Точка	Координаты точки		Значение функции
	1	2	
$x^{(0)}$	$x_1^{(0)} = 1$	$x_2^{(0)} = 1$	$f(\overline{x^{(0)}}) = 7.1$
$x^{(1)}$	$x_1^{(1)} = 0,8$	$x_2^{(1)} = 1$	$f(\overline{x^{(1)}}) = 5.748$

Далее делаем пробный шаг в направлении оси x_2 , получим пробную точку $(0,8; 1 + h) = (0,8; 1,2)$. Так как $f(0,8; 1,2) = 6.6 > f(\overline{x^{(1)}}) = 5.748$, то шаг считается неудачным. Возвращаемся на исходную точку и делаем шаг в отрицательном направлении оси x_2 : $(0,8; 1 - h) = (0,8; 0,8)$. Так как $f(0,8; 0,8) = 5.12 > f(\overline{x^{(1)}}) = 5.748$, то шаг в этом направлении считается удачным.

Таким образом, рассмотрены все координатные направления и найдена новая базисная точка $\overline{x^{(1)}} = (0,8; 0,8)$, которой соответствует значение целевой функции $f(\overline{x^{(1)}}) = 5.12$. Учитывая, что исследующий

поиск

был

удачен

$x^{(0)} \neq x^{(1)}$, то произведем поиск по образцу

$$\overline{x^{(p)}} = \overline{x^{(1)}} + m(\overline{x^{(1)}} - \overline{x^{(0)}}) = (0,8; 0,8) + 2[(0,8; 0,8) - (1; 1)] = (0,4; 0,4)$$

$$\text{Тогда } f(\overline{x^{(p)}}) = 2,48$$

Таблица 2.

Точка	Координаты точки		Значение функции
	1	2	
$x^{(0)}$	$x_1^{(0)} = 1$	$x_2^{(0)} = 1$	$f(\overline{x^{(0)}}) = 7.1$
$x^{(1)}$	$x_1^{(1)} = 0,8$	$x_2^{(1)} = 0.8$	$f(\overline{x^{(1)}}) = 5.12$
$x^{(2)}$	$x_1^{(2)} = 0,4$	$x_2^{(2)} = 0.4$	$f(\overline{x^{(2)}}) = 2.48$

Так как $f(\overline{x^{(p)}}) < f(\overline{x^{(1)}})$, то шаг по образцу считается удачным и точка $\overline{x^{(p)}} = \overline{x^{(2)}}$ становится новой базисной точкой (табл. 2).

Итерация 2. Проведем исследующий поиск вокруг базисной точки $\overline{x^{(2)}} = (0,4; 0,4)$, которой соответствует значение целевой функции $f(\overline{x^{(2)}}) = 2,48$.

Из точки $\overline{x^{(2)}}$ сделаем шаг в положительном направлении оси x_1 и рассмотрим пробную точку $(0,4 + h; 0,4) = (0,4 + 0,2; 0,4) = (0,6; 0,4)$. Так как $f(0,6; 0,4) = 3,4 > f(\overline{x^{(2)}}) = 2,48$, то шаг в этом направлении считается неудачным. Возвращаемся на исходную точку и делаем шаг в отрицательном направлении оси x_1 : $(0,4 - h; 0,4) = (0,2; 0,4)$. Так как $f(0,2; 0,4) = 1,776 < f(\overline{x^{(0)}}) = 2,48$, то шаг в этом направлении считается удачным.

Фиксируем пробную точку $x^{(3)} = (0,2; 0,4)$ и принимаем ее (табл.3).

Таблица 3.

Точка	Координаты точки		Значение функции
	1	2	
$x^{(0)}$	$x_1^{(0)} = 1$	$x_2^{(0)} = 1$	$f(\overline{x^{(0)}}) = 7.1$
$x^{(1)}$	$x_1^{(1)} = 0,8$	$x_2^{(1)} = 0.8$	$f(\overline{x^{(1)}}) = 5.12$
$x^{(2)}$	$x_1^{(2)} = 0,4$	$x_2^{(2)} = 0.4$	$f(\overline{x^{(2)}}) = 2.48$
$x^{(3)}$	$x_1^{(3)} = 0,2$	$x_2^{(3)} = 0.4$	$f(\overline{x^{(3)}}) = 1,776$

Повторим те же действия для переменной x_2 . Рассмотрим пробную точку $(0,2; 0,4 + h) = (0,2; 0,4 + 0,2) = (0,2; 0,6)$. Шаг в этом направлении неудачный, так как $f(0,2; 0,6) = 1,956 > f(\overline{x^{(3)}}) = 1,776$. Возвращаемся в исходную точку $(0,2; 0,4)$ и делаем шаг в отрицательном направлении оси x_2 : $(0,2; 0,4 - h) = (0,2; 0,4 - 0,2) = (0,2; 0,2)$. Так как $f(0,2; 0,2) = 1,82 > f(\overline{x^{(3)}}) = 1,776$, то шаг также является неудачным.

Таким образом, рассмотрены все координатные направления и найдена новая базисная точка $\overline{x^{(3)}} = (0,2; 0,4)$, которой соответствует значение целевой функции $f(\overline{x^{(3)}}) = 1,776$

Учитывая, что исследующий поиск был удачен $x^{(2)} \neq x^{(3)}$, то проводим поиск по образцу

$$\begin{aligned}\overline{x^{(p)}} &= \overline{x^{(3)}} + m(\overline{x^{(3)}} - \overline{x^{(2)}}) = (0,2; 0,4) + 2[(0,2; 0,4) - (0,4; 0,4)] \\ &= (-0,2; 0,4)\end{aligned}$$

$$\text{Тогда } f(\overline{x^{(p)}}) = 1,016$$

Таблица 4.

Точка	Координаты точки		Значение функции
	1	2	
$x^{(0)}$	$x_1^{(0)} = 1$	$x_2^{(0)} = 1$	$f(\overline{x^{(0)}}) = 7.1$
$x^{(1)}$	$x_1^{(1)} = 0,8$	$x_2^{(1)} = 0,8$	$f(\overline{x^{(1)}}) = 5.12$
$x^{(2)}$	$x_1^{(2)} = 0,4$	$x_2^{(2)} = 0,4$	$f(\overline{x^{(2)}}) = 2.48$
$x^{(3)}$	$x_1^{(3)} = 0,2$	$x_2^{(3)} = 0,4$	$f(\overline{x^{(3)}}) = 1,776$
$x^{(4)}$	$x_1^{(4)} = -0,2$	$x_2^{(4)} = 0,4$	$f(\overline{x^{(4)}}) = 1,016$

Так как $f(\overline{x^{(p)}}) < f(\overline{x^{(3)}})$, то шаг по образцу считается удачным и точка $\overline{x^{(p)}} = \overline{x^{(4)}}$ становится новой базисной точкой (табл. 4).

Итерация 3. Проведем исследующий поиск вокруг базисной точки $\overline{x^{(4)}} = (-0,2; 0,4)$, которой соответствует значение целевой функции $f(\overline{x^{(4)}}) = 1,016$.

Из точки $\overline{x^{(4)}}$ сделаем шаг в положительном направлении оси x_1 и рассмотрим пробную точку $(-0,2 + h; 0,4) = (-0,2 + 0,2; 0,4) = (0; 0,4)$. Так как $f(0; 0,4) = 1,288 > f(\overline{x^{(4)}}) = 1,016$, то шаг в этом направлении считается неудачным. Возвращаемся на исходную точку и делаем шаг в

отрицательном направлении оси x_1 : $(-0,2 - h; 0,4) = (-0,4; 0,4)$. Так как $f(-0,4; 0,4) = 0,96 < f(\overline{x^{(4)}}) = 1,016$, то шаг в этом направлении считается удачным.

Фиксируем пробную точку $x^{(5)} = (-0,4; 0,4)$ и принимаем ее (табл.5).

Таблица 5.

Точка	Координаты точки		Значение функции
	1	2	
$x^{(0)}$	$x_1^{(0)} = 1$	$x_2^{(0)} = 1$	$f(\overline{x^{(0)}}) = 7,1$
$x^{(1)}$	$x_1^{(1)} = 0,8$	$x_2^{(1)} = 0,8$	$f(\overline{x^{(1)}}) = 5,12$
$x^{(2)}$	$x_1^{(2)} = 0,4$	$x_2^{(2)} = 0,4$	$f(\overline{x^{(2)}}) = 2,48$
$x^{(3)}$	$x_1^{(3)} = 0,2$	$x_2^{(3)} = 0,4$	$f(\overline{x^{(3)}}) = 1,776$
$x^{(4)}$	$x_1^{(4)} = -0,2$	$x_2^{(4)} = 0,4$	$f(\overline{x^{(4)}}) = 1,016$
$x^{(5)}$	$x_1^{(5)} = -0,4$	$x_2^{(5)} = 0,4$	$f(\overline{x^{(5)}}) = 0,96$

Повторим те же действия для переменной x_2 . Рассмотрим пробную точку $(-0,4; 0,4 + h) = (-0,4; 0,4 + 0,2) = (-0,4; 0,6)$. Шаг в этом направлении неудачный, так как $f(-0,4; 0,6) = 1,14 > f(\overline{x^{(5)}}) = 0,96$. Возвращаемся в исходную точку $(-0,4; 0,4)$ и делаем шаг в отрицательном направлении оси x_2 : $(-0,4; 0,4 - h) = (-0,4; 0,4 - 0,2) = (-0,4; 0,2)$. Так как $(-0,4; 0,2) = 1,004 > f(\overline{x^{(5)}}) = 0,96$, то шаг также является неудачным.

Таким образом, рассмотрены все координатные направления и найдена новая базисная точка $\overline{x^{(5)}} = (-0,4; 0,4)$, которой соответствует значение целевой функции $f(\overline{x^{(5)}}) = 0,96$

Учитывая, что исследующий поиск был удачен $x^{(4)} \neq x^{(5)}$, то проводим поиск по образцу

$$\begin{aligned} \overline{x^{(p)}} &= \overline{x^{(5)}} + m(\overline{x^{(5)}} - \overline{x^{(4)}}) = (-0,4; 0,4) + 2[(-0,4; 0,4) - (-0,2; 0,4)] \\ &= (-0,8; 0,4) \end{aligned}$$

$$\text{Тогда } f(\overline{x^{(p)}}) = 1,496$$

Так как $f(\overline{x^{(p)}}) > f(\overline{x^{(5)}})$, то шаг по образцу считается неудачным и точка $\overline{x^{(5)}}$ становится новой базисной точкой (табл. 5).

Итерация 4. Проведем исследующий поиск вокруг базисной точки $\overline{x^{(5)}} = (-0,4; 0,4)$, которой соответствует значение целевой функции $f(\overline{x^{(5)}}) = 0,96$.

Из точки $\overline{x^{(5)}}$ сделаем шаг в положительном направлении оси x_1 и рассмотрим пробную точку $(-0,4 + h; 0,4) = (-0,4 + 0,2; 0,4) = (-0,2; 0,4)$. Так как $f(-0,2; 0,4) = 1,016 > f(\overline{x^{(5)}}) = 0,96$, то шаг в этом направлении считается неудачным. Возвращаемся на исходную точку и делаем шаг в отрицательном направлении оси x_1 : $(-0,4 - h; 0,4) = (-0,6; 0,4)$. Так как $f(-0,6; 0,4) = 1,12 > f(\overline{x^{(5)}}) = 0,96$, то шаг в этом направлении считается неудачным. Возвращаемся в исходную точку.

Повторим те же действия для переменной x_2 . Рассмотрим пробную точку $(-0,4; 0,4 + h) = (-0,4; 0,4 + 0,2) = (-0,4; 0,6)$. Шаг в этом направлении неудачный, так как $f(-0,4; 0,6) = 1,14 > f(\overline{x^{(5)}}) = 0,96$. Возвращаемся в исходную точку $(-0,4; 0,4)$ и делаем шаг в отрицательном направлении оси x_2 : $(-0,4; 0,4 - h) = (-0,4; 0,4 - 0,2) = (-0,4; 0,2)$. Так как $f(-0,4; 0,2) = 1,004 > f(\overline{x^{(5)}}) = 0,96$, то шаг также является неудачным.

Таким образом, исследующий поиск с величиной шага, равной $h = 0,2$, был неудачен. Тогда уменьшим величину шага на $d = 2$. Получим $h = h/d = 0,1$. Возобновим исследующий поиск относительно точки $\overline{x^{(5)}}$.

Итерация 5. Проведем исследующий поиск вокруг базисной точки $\overline{x^{(5)}} = (-0,4; 0,4)$, которой соответствует значение целевой функции $f(\overline{x^{(5)}}) = 0,96$.

Из точки $\overline{x^{(45)}}$ сделаем шаг в положительном направлении оси x_1 и рассмотрим пробную точку $(-0,4 + h; 0,4) = (-0,4 + 0,1; 0,4) = (-0,3; 0,4)$. Так как $f(-0,3; 0,4) = 0,961 > f(\overline{x^{(5)}}) = 0,96$, то шаг в этом направлении считается неудачным. Возвращаемся на исходную точку и делаем шаг в отрицательном направлении оси x_1 : $(-0,4 - h; 0,4) = (-0,5; 0,4)$. Так как $f(-0,5; 0,4) = 1,013 < f(\overline{x^{(5)}}) = 0,96$, то шаг в этом направлении считается неудачным. Возвращаемся в исходную точку.

Повторим те же действия для переменной x_2 . Рассмотрим пробную точку $(-0,4; 0,4 + h) = (-0,4; 0,4 + 0,1) = (-0,4; 0,5)$. Шаг в этом направлении неудачный, так как $f(-0,4; 0,5) = 1,022 > f(\overline{x^{(5)}}) = 0,96$.

Возвращаемся в исходную точку $(-0,4; 0,4)$ и делаем шаг в отрицательном направлении оси x_2 : $(-0,4; 0,4 - h) = (-0,4; 0,4 - 0,1) = (-0,4; 0,3)$. Так как $(-0,4; 0,3) = 0,954 < f(\overline{x^{(5)}}) = 0,96$, то шаг является удачным.

Таким образом, рассмотрены все координатные направления и найдена новая базисная точка $\overline{x^{(6)}} = (-0,4; 0,3)$, которой соответствует значение целевой функции $f(\overline{x^{(6)}}) = 0,954$ (табл.6)

Таблица 6.

Точка	Координаты точки		Значение функции
	1	2	
$x^{(0)}$	$x_1^{(0)} = 1$	$x_2^{(0)} = 1$	$f(\overline{x^{(0)}}) = 7.1$
$x^{(1)}$	$x_1^{(1)} = 0,8$	$x_2^{(1)} = 0.8$	$f(\overline{x^{(1)}}) = 5.12$
$x^{(2)}$	$x_1^{(2)} = 0,4$	$x_2^{(2)} = 0.4$	$f(\overline{x^{(2)}}) = 2.48$
$x^{(3)}$	$x_1^{(3)} = 0,2$	$x_2^{(3)} = 0.4$	$f(\overline{x^{(3)}}) = 1,776$
$x^{(4)}$	$x_1^{(4)} = -0,2$	$x_2^{(4)} = 0.4$	$f(\overline{x^{(4)}}) = 1,016$
$x^{(5)}$	$x_1^{(5)} = -0,4$	$x_2^{(5)} = 0.4$	$f(\overline{x^{(5)}}) = 0,96$
$x^{(6)}$	$x_1^{(6)} = -0,4$	$x_2^{(6)} = 0.3$	$f(\overline{x^{(6)}}) = 0,954$

Проверяем условие окончания поиска $h = 0,1 \leq \varepsilon$, то поиск завершен и требуемая точность достигнута $\overline{x^*} = \overline{x^{(6)}} = (-0,4; 0,3)$, $f(\overline{x^*}) = 0,954$

1.3.2. Программная реализация

Программный код выполнен на языке Java.

```
import java.util.ArrayList;
```

```
public class HookJeeves extends Printable {
```

```
    //Размерность
```

```
    private int dimension = 2;
```

```
    //Точность искомого значения функции
```

```
    private double eps = 0.1;
```

```
    //Вектор шагов по координатам
```

```
    private double[] step = {0.2, 0.2};
```

```
    //Коэффициент уменьшения шага
```

```
    private double decStep = 2;
```

```

//Ускоряющий множитель
private double incMult = 2;
//Стартовый вектор
private double[] basisVector = { 1, 1 };
//Массив векторов
private ArrayList<CoordRow> arr = new ArrayList<>();

public HookJeeves() {
    arr.add(new CoordRow(cloneVector(basisVector),
        getFuncValue(basisVector)));
}

public static void main(String[] args) {
    HookJeeves hookJeeves = new HookJeeves();
    int i = 0;
    while (true) {
        System.out.println("\nИтерация " + i++);
        if (hookJeeves.run()) break;
    }
    hookJeeves.printTable(hookJeeves.getArr());
}

public ArrayList<CoordRow> getArr() {
    return arr;
}

public boolean run() {
    double[] vector = cloneVector(basisVector);
    double[] tmpVector = cloneVector(basisVector);
    printVector(basisVector, "Базисный вектор: ");
    printValue(getFuncValue(basisVector), "Численное значение базисного
вектора: ");
    //Проходим по всем координатам
    for (int i = 0; i < dimension; i++) {
        //Меняем значение координаты на +шаг
        tmpVector[i] += step[i];
    }
}

```

```

    printVector(tmpVector, "Вектор при положительном шаге:");
    printValue(getFuncValue(tmpVector), "Значение вектора при
положительном шаге: ");
    //Проверяем, стало ли значение функции меньше
    if (getFuncValue(tmpVector) < getFuncValue(vector)) {
        //Клонируем вектор
        vector = cloneVector(tmpVector);
        continue;
    }
    //Меняем значение координаты на -шаг
    tmpVector[i] -= 2 * step[i];
    printVector(tmpVector, "Вектор при отрицательном шаге:");
    printValue(getFuncValue(tmpVector), "Значение вектора при
отрицательном шаге: ");
    //Проверяем, стало ли значение функции меньше
    if (getFuncValue(tmpVector) < getFuncValue(vector)) {
        //Клонируем вектор
        vector = cloneVector(tmpVector);
        continue;
    }
    //Возвращаем значение координаты в исходное значение
    tmpVector[i] += step[i];
}
printVector(vector, "Вектор X1: ");
printValue(getFuncValue(vector), "Численное значение вектора X1: ");
//Проверяем, равен ли полученный вектор при изменении координат
начальному
if (isEqual(vector, basisVector)) {
    //Меняем шаг координат
    for (int i = 0; i < dimension; i++) {
        step[i] /= decStep;
    }
    printVector(step, "Уменьшаем шаг: ");
    return false;
} else {
    arr.add(new CoordRow(cloneVector(vector), getFuncValue(vector)));
}

```

```

//Проводим поиск по образцу
double[] pVector = searchVector(vector);
printVector(pVector, "Вектор Xp: ");
printValue(getFuncValue(pVector), "Численное значение вектора Xp:
");

//Проверяем удачность поиска по образцу
if (getFuncValue(pVector) < getFuncValue(vector)) {
    arr.add(new CoordRow(cloneVector(pVector),
getFuncValue(pVector)));
    basisVector = pVector;
} else {
    basisVector = vector;
}
}
printVector(step, "Шар: ");
//Проверяем условие окончания поиска
for (int i = 0; i < dimension; i++) {
    if (step[i] > eps) return false;
}
return true;
}

//Метод осуществляет поиск по образцу
private double[] searchVector(double[] vector) {
    double[] tmpVector = new double[dimension];
    for (int i = 0; i < dimension; i++) {
        tmpVector[i] = vector[i] + incMult * (vector[i] - basisVector[i]);
    }
    return tmpVector;
}

@Override
protected double getFuncValue(double[] x) {
    return 14d/5d*x[1]*x[1] + 1.9d*x[0] + 2.7d*x[0]*x[0] + 8d/5d - 1.9*x[1];
}
}

```

Результат работы программы (рис. 1.12):

```

Итерация 4
Базисный вектор:
-0.4      0.4
Численное значение базисного вектора:
0.96
Вектор при положительном шаге:
-0.3      0.4
Значение вектора при положительном шаге:
0.961
Вектор при отрицательном шаге:
-0.5      0.4
Значение вектора при отрицательном шаге:
1.013
Вектор при положительном шаге:
-0.4      0.5
Значение вектора при положительном шаге:
1.022
Вектор при отрицательном шаге:
-0.4      0.3
Значение вектора при отрицательном шаге:
0.954
Вектор X1:
-0.4      0.3
Численное значение вектора X1:
0.954
Вектор Xp:
-0.4      0.1
Численное значение вектора Xp:
1.11
Шаг:
0.1      0.1

Таблица векторов:
1.0      1.0      7.1
0.8      0.8      5.12
0.4      0.4      2.48
0.2      0.4      1.776
-0.2     0.4      1.016
-0.4     0.4      0.96
-0.4     0.3      0.954

```

Рис.1.12. Результат работы программы

1.3.3. Контрольные вопросы

1. Сущность метода Хука-Дживса, как класса детерминированной оптимизации.
2. Сущность и алгоритм исследующего поиска.
3. Сущность и алгоритм поиска по образцу.
4. Изобразите схематично шаги метода Хука-Дживса для функций двух переменных
5. Шаги алгоритма метода Хука-Дживса.
6. Какие необходимы начальные параметры для расчета минимального значения целевой функции по методу Хука-Дживса.
7. Проверка условия окончания исследующего поиска.
8. Проверка удачности поиска по образцу.

Глава 2. Методы первого порядка.

Методы первого порядка используют информацию о значениях целевой функции $f(\bar{x})$ и её первых производных. Предполагается, что функция $f(\bar{x})$ и её первые производные существуют и непрерывны. Направление смещения от точки $\bar{x}^{(k)}$ к точке $\bar{x}^{(k+1)}$ описывается представленной на первой лекции интеграционной процедурой (1)

$$\bar{x}^{(k+1)} = \bar{x}^{(k)} + h_k \bar{p}^{(k)},$$

где k – номер итерации $k = 0, 1, \dots$;

$\bar{x}^{(k)}$ – текущее приближение;

h_k – величина шага;

\bar{p}^k – вектор, определяющий направление убывания функции $f(\bar{x})$.

и совпадает с направлением вектора антиградиента целевой функции $\bar{p}^{(k)} = -\nabla f(\bar{x}^{(k)})$. Все итерационные процессы, в которых направление движения на каждом шаге совпадает с антиградиентом функции, называются градиентными методами. Существует несколько модификаций градиентных методов, различающихся правилом выбора длины шага в направлении антиградиента функции.

Мы рассмотрим наиболее распространённые на практике следующие методы: метод градиентного спуска с постоянным шагом, метод наискорейшего градиентного спуска, метод покоординатного спуска, метод сопряжённых направлений.

2.1. Метод градиентного спуска с постоянным шагом.

Сущность метода градиентного спуска с постоянным шагом заключается в следующем. Выбирается начальная точка $\bar{x}^{(0)}$ из области определения функции $f(\bar{x})$. Координаты новой точки вычисляются по формуле

$$\bar{x}^{(k+1)} = \bar{x}^{(k)} - h_k \nabla f(\bar{x}^{(k)}),$$

где k – номер итерации $k = 0, 1, \dots$,

h_k – величина шага,

$\nabla f(\bar{x}^{(k)})$ – градиент функции $f(\bar{x})$ в точке $\bar{x}^{(k)}$,

$$\nabla f(\bar{x}^{(k)}) = \left(\frac{\partial f(\bar{x}^{(k)})}{\partial x_1}, \frac{\partial f(\bar{x}^{(k)})}{\partial x_2}, \dots, \frac{\partial f(\bar{x}^{(k)})}{\partial x_n} \right).$$

Начальная величина шага h_0 задаётся пользователем. В каждой новой точке поиска $\overline{x^{(k+1)}}$ проверяется условие убывания функции $f(\overline{x^{(k+1)}}) < f(\overline{x^{(k)}})$. Если условие нарушается, то постепенно уменьшается величина шага h_k , т.е. точка $\overline{x^{(k+1)}}$ приближается к точке $\overline{x^{(k)}}$ до тех пор, пока условие не выполнится. В полученной точке $\overline{x^{(k+1)}}$ определяется новое направление градиента и осуществляется новый спуск. Процесс продолжается пока не будет выполнено условие окончания поиска. В качестве условия окончания поиска используется близость к нулю нормы градиента $\|\nabla f(\overline{x^{(k+1)}})\| \leq \varepsilon$.

Геометрическая иллюстрация поиска минимума целевой функции методом градиентного спуска с постоянным шагом для случая $n = 2$ представлена на рис. 2.1.

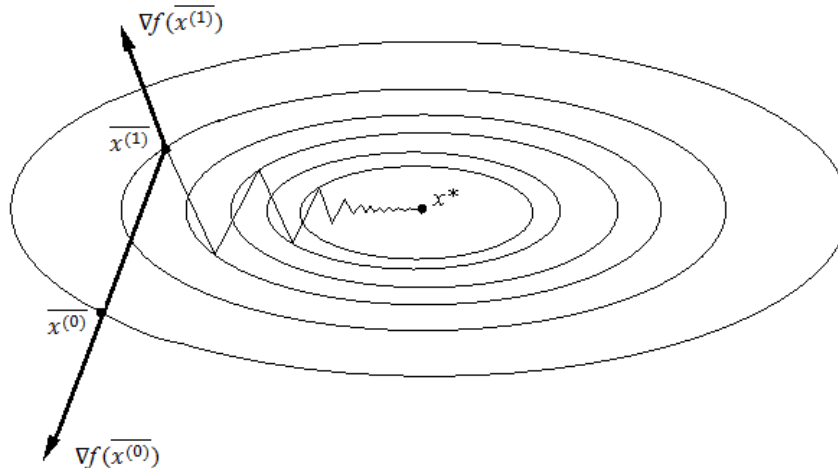


Рис. 2.1. Графическая иллюстрация поиска точки минимума методом градиентного спуска с постоянным шагом

Алгоритм метода минимизации целевой функции $f(\bar{x})$ методом градиентного спуска с постоянным шагом заключается в следующем:

1. Задать размерность задачи оптимизации n , координаты начальной точки $\overline{x^{(0)}} = (x_1^{(0)}, x_2^{(0)}, \dots, x_n^{(0)})$, начальную величину шага h_0 , точность поиска ε .
2. Положить счётчик числа итерация $k = 0$
3. Вычислить значение функции $f(\overline{x^{(k)}})$ в точке $\overline{x^{(k)}}$.
4. Определить координаты вектора градиента функции $f(\bar{x})$ в точке

$$\overline{x^{(k)}} = (x_1^{(k)}, x_2^{(k)}, \dots, x_n^{(k)}).$$

5. Проверить условие окончания поиска

$$\|\nabla f(\overline{x^{(k)}})\| = \sqrt{\sum_{i=1}^n \left(\frac{\partial f(\overline{x^{(k)}})}{\partial x_i}\right)^2} \leq \varepsilon.$$

Если условие выполнено, то перейти к пункту 8, иначе – к пункту 6.

6. Определить координаты точки $\overline{x^{(k+1)}} = \overline{x^{(k)}} - h_k \nabla f(\overline{x^{(k)}})$ и значение целевой функции $f(\overline{x^{(k+1)}})$.

7. Проверить условие убывания функции $f(\overline{x^{(k+1)}}) < f(\overline{x^{(k)}})$.

Если условие выполнено, то положить $k = k + 1$, $f(\overline{x^{(k)}}) = f(\overline{x^{(k+1)}})$ и перейти к пункту 4, иначе – положить $h_k = \frac{h_k}{2}$ и перейти к пункту 6.

8. Расчёт окончен. Полагаем $\overline{x^*} = \overline{x^{(k)}}$.

2.1.1. Практический расчет задачи

Постановка задачи: Найти минимум целевой функции

$$f(\overline{x}) = 2.8 * x_2^2 + 1.9 * x_1 + 2.7 * x_1^2 + 1.6 - 1.9 * x_2$$

методом градиентного спуска с постоянным шагом с точностью $\varepsilon = 0,1$.

Решение. Зададим начальную точку $\overline{x^{(0)}} = (x_1^{(0)}, x_2^{(0)})^T = (1; 1)^T$ и начальную величину шага $h = 0,4$.

Найдем градиент функции в произвольной точке $\overline{x^{(k)}} = (x_1^{(k)}, x_2^{(k)})^T$

$$\nabla f(\overline{x^{(k)}}) = \left(\frac{\partial f(\overline{x^{(k)}})}{\partial x_1}; \frac{\partial f(\overline{x^{(k)}})}{\partial x_2} \right)^T = (5.4 * x_1^{(k)} + 1.9; 5.6 * x_2^{(k)} - 1.9)^T.$$

Итерация $k = 0$. Вычислим значение целевой функции $f(\overline{x^{(0)}})$ и градиент $\nabla f(\overline{x^{(0)}})$ в начальной точке $\overline{x^{(0)}}$: $f(\overline{x^{(0)}}) = 7,1$; $\nabla f(\overline{x^{(0)}}) = (7,3; 3,7)^T$. Занесем значения в таблицу 1.

Таблица 1.

Точка	Координаты точки		Значение функции
	1	2	
$\overline{x^{(0)}}$	$x_1^{(0)} = 1$	$x_2^{(0)} = 1$	$f(\overline{x^{(0)}}) = 7,1$

Определим координаты точки $\overline{x^{(1)}}$

$$\overline{x^{(1)}} = \overline{x^{(0)}} - h \nabla f(\overline{x^{(0)}}) = (1; 1)^T - 0,4 * (-7,3; 3,7)^T = (-1,92; -0,48)^T$$

и значение целевой функции в этой точке

$$f(\overline{x^{(1)}}) = 9,462.$$

Сравним $f(\overline{x^{(0)}})$ и $f(\overline{x^{(1)}})$. Поскольку $f(\overline{x^{(1)}}) > f(\overline{x^{(0)}})$, то условие убывания функции не выполнено. Уменьшим величину шага $h = \frac{h}{2} = 0,2$ и повторим вычисления координат точки $\overline{x^{(3)}}$

$$\overline{x^{(1)}} = \overline{x^{(0)}} - h \nabla f(\overline{x^{(0)}}) = (1; 1)^T - 0,2 * (-7,3; 3,7)^T = (-0,46; 0,26)^T$$

Значение целевой функции $f(\overline{x^{(1)}}) = 0,993$.

Сравним $f(\overline{x^{(1)}})$ и $f(\overline{x^{(0)}})$. Поскольку $f(\overline{x^{(1)}}) < f(\overline{x^{(0)}})$, то условие убывания функции выполнено. Составим таблицу 2.

Таблица 2.

Точка	Координаты точки		Значение функции
	1	2	
$\overline{x^{(0)}}$	$\overline{x_1^{(0)}} = 1$	$\overline{x_2^{(0)}} = 1$	$f(\overline{x^{(0)}}) = 7,1$
$\overline{x^{(1)}}$	$\overline{x_1^{(1)}} = -0,46$	$\overline{x_2^{(1)}} = 0,26$	$f(\overline{x^{(1)}}) = 0,993$

Проверим условие окончания поиска. Для этого вычислим вектор градиента $\nabla f(\overline{x^{(1)}})$ в точке $\overline{x^{(1)}}$:

$$\nabla f(\overline{x^{(1)}}) = (-0,584; -0,444)^T.$$

Найдем норму вектора градиента $\overline{x^{(1)}} = (-0,46; 0,26)^T$

$$\nabla f(\overline{x^{(1)}}) = (2\overline{x_1^{(1)}} - \overline{x_2^{(1)}} - 1; -\overline{x_1^{(1)}} + 6\overline{x_2^{(1)}})^T = (-0,584; -0,444)^T$$

$$\|\nabla f(\overline{x^{(1)}})\| = \sqrt{(-0,584)^2 + (-0,444)^2} = 0,734 > \varepsilon.$$

Итерации продолжаются.

Итерация $k = 1$. Определим координаты точки $\overline{x^{(2)}}$

$$\overline{x^{(2)}} = \overline{x^{(1)}} - h \nabla f(\overline{x^{(1)}}) = (-0,46; 0,26)^T - 0,2 * (-0,584; -0,444)^T = (-0,343; 0,349)^T$$

и значение целевой функции в этой точке

$$f(\overline{x^{(2)}}) = 0,944.$$

Запишем значения в таблицу 3.

Таблица 3.

Точка	Координаты точки		Значение функции
	1	2	
$x^{(0)}$	$x_1^{(0)} = 1$	$x_2^{(0)} = 1$	$f(\overline{x^{(0)}}) = 7,1$
$x^{(1)}$	$x_1^{(1)} = -0,46$	$x_2^{(1)} = 0,26$	$f(\overline{x^{(1)}}) = 0,993$
$x^{(2)}$	$x_1^{(2)} = -0,343$	$x_2^{(2)} = 0,349$	$f(\overline{x^{(2)}}) = 0,944$

Сравним $f(\overline{x^{(1)}})$ и $f(\overline{x^{(2)}})$. Поскольку $f(\overline{x^{(2)}}) < f(\overline{x^{(1)}})$, то условие убывания функции выполнено.

Проверим условие окончания поиска. Для этого вычислим вектор градиента $\nabla f(\overline{x^{(2)}})$ в точке $\overline{x^{(2)}}$:

$$\nabla f(\overline{x^{(2)}}) = (0,047; 0,053)^T.$$

Найдем норму вектора градиента

$$\|\nabla f(\overline{x^{(2)}})\| = \sqrt{(0,043)^2 + (0,053)^2} = 0,071 < \varepsilon.$$

Расчет окончен. Найдена точка

$$x^{(*)} = x^{(2)} = (-0,343; 0,349)^T, f(\overline{x^{(*)}}) = 0,944.$$

2.1.2. Программная реализация

Программный код выполнен на языке Java.

```
public abstract class Differential extends Printable {
    //Численный метод подсчета первой частной производной
    protected double getFirstPartialDerivative(double[] values, double accuracy,
int numArg) {
        double funcValue2 = getFuncValue(values);
        values[numArg] += accuracy;
        double funcValue1 = getFuncValue(values);
        values[numArg] -= accuracy;
        return (funcValue1 - funcValue2) / accuracy;
    }

    //Численный метод подсчета второй частной производной
    protected double getSecondPartialDerivative(double[] values, double
accuracy, int numArg) {
        double funcValue2 = getFuncValue(values);
        values[numArg] += accuracy;
        double funcValue1 = getFuncValue(values);
```

```

    values[numArg] -= 2 * accuracy;
    double funcValue3 = getFuncValue(values);
    values[numArg] += accuracy;
    return (funcValue1 - 2 * funcValue2 + funcValue3) / (accuracy *
accuracy);
}

```

```

//Численный метод подсчета смешанной частной производной
protected double getMixedDerivative(double[] values, double accuracy1, int
numArg1, double accuracy2, int numArg2) {
    double funcValue1 = getFuncValue(values);
    values[numArg1] -= accuracy1;
    double funcValue2 = getFuncValue(values);
    values[numArg2] -= accuracy2;
    double funcValue4 = getFuncValue(values);
    values[numArg1] += accuracy1;
    double funcValue3 = getFuncValue(values);
    values[numArg2] += accuracy2;
    return (funcValue1 - funcValue2 - funcValue3 + funcValue4) / (accuracy1
* accuracy2);
}

```

```

//Численный метод подсчета градиента
protected double[] getGradientValue(double[] values, double accuracy) {
    double[] tmp = new double[values.length];
    for (int i = 0; i < values.length; i++) {
        tmp[i] = getFirstPartialDerivative(values, accuracy, i);
    }
    return tmp;
}

```

```

//Численный метод подсчета антиградиента
protected double[] getAntiGradientValue(double[] values, double accuracy) {
    double[] tmp = new double[values.length];
    for (int i = 0; i < values.length; i++) {
        tmp[i] = -getFirstPartialDerivative(values, accuracy, i);
    }
}

```

```

    }
    return tmp;
}

//Метод вычисляющий новый вектор
protected double[] countNewVector(double[] basisVector, double[]
gradientVector, double step) {
    double[] tmpVector = new double[basisVector.length];
    for (int i = 0; i < tmpVector.length; i++) {
        tmpVector[i] = basisVector[i] - step * gradientVector[i];
    }
    return tmpVector;
}

//Метод подсчета нормы вектора
protected double getNorma(double[] vector) {
    double sum = 0d;
    for (int i = 0; i < vector.length; i++) {
        sum += vector[i] * vector[i];
    }
    return Math.sqrt(sum);
}

//Метод считающий матрицу Гессе
protected double[][] getGesseMatrix(double[] vector, double accuracy, int
dimension) {
    double[][] gesseMatrix = new double[dimension][dimension];
    for (int i = 0; i < dimension; i++) {
        for (int j = 0; j < dimension; j++) {
            if (j == i) {
                gesseMatrix[i][j] = getSecondPartialDerivative(vector, accuracy, i);
            } else {
                gesseMatrix[i][j] = getMixedDerivative(vector, accuracy, i,
accuracy, j);
            }
        }
    }
}

```

```

    }
    return gesseMatrix;
}

//Численный метод получения шага
protected double getStep(double[] vector, double accuracy, int dimension) {
    double[][] gesseMatrix = getGesseMatrix(vector, accuracy, dimension);
    double[][] gradientMatrixHorizontal = new double[1][dimension];
    gradientMatrixHorizontal[0] = getGradientValue(vector, accuracy);
    double[][] gradientMatrixVertical =
MatrixOperations.transportMatrix(gradientMatrixHorizontal);
    double[][] tmp = MatrixOperations.multiplyByMatrix(gesseMatrix,
gradientMatrixVertical);
    tmp = MatrixOperations.multiplyByMatrix(gradientMatrixHorizontal,
tmp);
    double value = tmp[0][0];
    tmp = MatrixOperations.multiplyByMatrix(gradientMatrixHorizontal,
gradientMatrixVertical);
    return tmp[0][0] / value;
}

//Абстрактный метод, который переопределяется конкретной функцией
protected abstract double getFuncValue(double[] values);

}

import java.util.ArrayList;

public class ConstGradient extends Differential {
    //Массив векторов
    private static ArrayList<CoordRow> arr = new ArrayList<>();
    //Размерность
    private int dimension = 2;
    //Точность численных вычислений
    private double accuracy = 0.00000001d;
    //Точность искомого значения функции

```

```

private double eps = 0.1;
//Стартовый вектор
private double[] basisVector = { 1, 1 };
//Значение шага
private double step = 0.4;

private ConstGradient() {
    arr.add(new CoordRow(cloneVector(basisVector),
getFuncValue(basisVector)));
}

public static void main(String[] args) {
    ConstGradient constGradient = new ConstGradient();
    int i = 0;
    while (true) {
        System.out.println("\nИтерация " + i++);
        if (constGradient.run()) break;
    }
    constGradient.printTable(constGradient.getArr());
}

public ArrayList<CoordRow> getArr() {
    return arr;
}

@Override
protected double getFuncValue(double[] x) {
    return 2.8d*x[1]*x[1] + 1.9d*x[0] + 2.7d*x[0]*x[0] + 1.6d - 1.9*x[1];
}

private boolean run() {
    printVector(basisVector, "Координаты базисного вектора: ");
    printValue(getFuncValue(basisVector), "Скалярное значение базисного
вектора: ");
    //Считаем значение вектора градиента
    double[] gradientVector = getGradientValue(basisVector, accuracy);

```



```

printVector(gradientVector, "Координаты вектора градиента: ");
//Считаем новый вектор
double[] vector = countNewVector(basisVector, gradientVector, step);
printVector(vector, "Координаты нового вектора: ");
printValue(getFuncValue(vector), "Скалярное значение нового вектора:
");
//Проверяем меньше ли значение нового вектора
if (getFuncValue(vector) < getFuncValue(basisVector)) {
    //Считаем значение вектора градиента от нового вектора
    gradientVector = getGradientValue(vector, accuracy);
    printVector(gradientVector, "Координаты вектора нового градиента:
");

    //Считаем норму
    double norma = getNorma(gradientVector);
    printValue(norma, "Значение норма вектора: ");
    arr.add(new CoordRow(cloneVector(vector), getFuncValue(vector)));
    //Меняем текущий вектор на новый
    basisVector = vector;
    //Проверяем условие выхода
    if (norma < eps)
        return true;
    else
        return false;
} else {
    //Уменьшаем шаг в два разара
    step /= 2;
    System.out.println("Новый шаг:\n" + step);
}
return false;
}
}

```

Результат работы программы (рис. 2.2):

```

Итерация 0
Координаты базисного вектора:
1.0          1.0
Скалярное значение базисного вектора:
7.1
Координаты вектора градиента:
7.3          3.7
Координаты нового вектора:
-1.92        -0.48
Скалярное значение нового вектора:
9.462
Новый шаг:
0.2

Итерация 1
Координаты базисного вектора:
1.0          1.0
Скалярное значение базисного вектора:
7.1
Координаты вектора градиента:
7.3          3.7
Координаты нового вектора:
-0.46        0.26
Скалярное значение нового вектора:
0.993
Координаты вектора нового градиента:
-0.584       -0.444
Значение норма вектора:
0.734

Итерация 2
Координаты базисного вектора:
-0.46        0.26
Скалярное значение базисного вектора:
0.993
Координаты вектора градиента:
-0.584       -0.444
Координаты нового вектора:
-0.343       0.349
Скалярное значение нового вектора:
0.944
Координаты вектора нового градиента:
0.047        0.053
Значение норма вектора:
0.071

Таблица векторов:
1.0          1.0          7.1
-0.46        0.26        0.993
-0.343       0.349       0.944

```

Рис. 2.2. Результат работы программы

2.1.3. Контрольные вопросы

1. Сущность методов первого порядка, интеграционная процедура.
2. Сущность метода градиентного спуска с постоянным шагом.
3. Алгоритм метода градиентного спуска с постоянным шагом.
4. Определение вектора градиента целевой функции.
5. Проверка условия окончания поиска.

2.2. Метод наискорейшего градиентного спуска.

Метод наискорейшего спуска отличается от метода градиентного спуска способом определения величины шага h_k . Величина шага задается не произвольно, а выбирается так, чтобы на каждой итерации достигалось максимально возможное уменьшение целевой функции $f(\bar{x})$ вдоль направления ее антиградиента $-\nabla f(\bar{x}^{(k)})$, вычисленного в точке $\bar{x}^{(k)}$. Величина шага h_k определяется из решения вспомогательной одномерной задачи минимизации

$$\varphi(h_k) = f\left(\bar{x}^{(k)} - h_k \nabla f\left(\bar{x}^{(k)}\right)\right) \rightarrow \min_{h_k > 0},$$

которая может быть решена аналитически или численно. При квадратичной интерполяции целевой функции величину шага можно определить по формуле

$$h_k = \frac{\left(\nabla f\left(\bar{x}^{(k)}\right), \nabla f\left(\bar{x}^{(k)}\right)\right)}{\left(H\left(\bar{x}^{(k)}\right) \nabla f\left(\bar{x}^{(k)}\right), \nabla f\left(\bar{x}^{(k)}\right)\right)}$$

где $H\left(\bar{x}^{(k)}\right)$ – матрица Гессе, вычисленная в точке $\bar{x}^{(k)}$.

$$H\left(\bar{x}^{(k)}\right) = \begin{bmatrix} \frac{\partial^2 f\left(\bar{x}^{(k)}\right)}{\partial x_1^2} & \frac{\partial^2 f\left(\bar{x}^{(k)}\right)}{\partial x_1 \partial x_2} & \cdots & \frac{\partial^2 f\left(\bar{x}^{(k)}\right)}{\partial x_1 \partial x_n} \\ \frac{\partial^2 f\left(\bar{x}^{(k)}\right)}{\partial x_2 \partial x_1} & \frac{\partial^2 f\left(\bar{x}^{(k)}\right)}{\partial x_2^2} & \cdots & \frac{\partial^2 f\left(\bar{x}^{(k)}\right)}{\partial x_2 \partial x_n} \\ \cdots & \cdots & \cdots & \cdots \\ \frac{\partial^2 f\left(\bar{x}^{(k)}\right)}{\partial x_n \partial x_1} & \frac{\partial^2 f\left(\bar{x}^{(k)}\right)}{\partial x_n \partial x_2} & \cdots & \frac{\partial^2 f\left(\bar{x}^{(k)}\right)}{\partial x_n^2} \end{bmatrix}$$

На рис. 2.3 представлена траектория приближения к точке минимума \bar{x}^* методом наискорейшего спуска для случая $n = 2$. Здесь каждая последующая точка находится как точка касания антиградиента целевой функции и линии уровня.

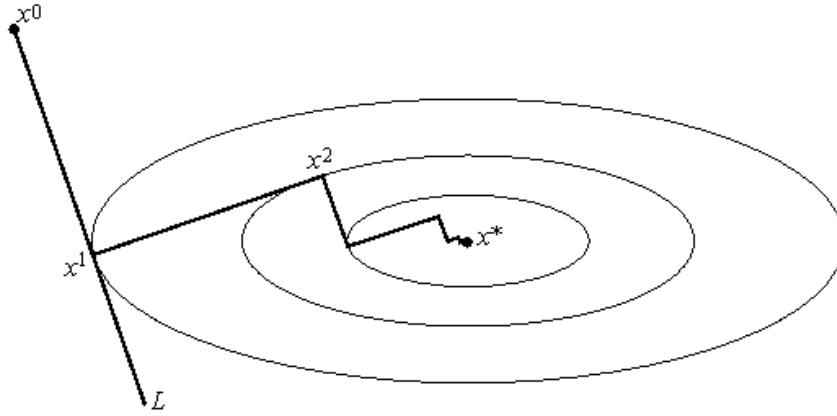


Рис. 2.3. Траектория движения к точке минимума в методе наискорейшего спуска

Разработанные ниже алгоритм метода наискорейшего спуска и программы на используют формулу

$$h_k = \frac{\left(\nabla f \left(\overline{x^{(k)}} \right), \nabla f \left(\overline{x^{(k)}} \right) \right)}{\left(H \left(\overline{x^{(k)}} \right) \nabla f \left(\overline{x^{(k)}} \right), \nabla f \left(\overline{x^{(k)}} \right) \right)}$$

для вычисления наилучшей величины шага и каждой итерации.

Процесс вычисления частных производных формализован и выполняется по разностным формулам:

- Первые частные производные вычисляются по формуле.

$$\frac{\partial f \left(\overline{x^{(k)}} \right)}{\partial x_i} \approx \frac{f(x_1^{(k)}, x_2^{(k)}, \dots, x_i^{(k)} + \Delta x_i^{(k)}, \dots, x_n^{(k)}) - f(x_1^{(k)}, x_2^{(k)}, \dots, x_i^{(k)}, \dots, x_n^{(k)})}{\Delta x_i^{(k)}}$$

- Вторые частные производные по формуле

$$\frac{\partial^2 f \left(\overline{x^{(k)}} \right)}{\partial x_i^2} \approx \frac{u_1 - 2u_2 + u_3}{(\Delta x_i^{(k)})^2}$$

где

$$u_1 = f \left(x_1^{(k)}, x_2^{(k)}, \dots, x_i^{(k)} + \Delta x_i^{(k)}, \dots, x_n^{(k)} \right),$$

$$u_2 = f \left(x_1^{(k)}, x_2^{(k)}, \dots, x_i^{(k)}, \dots, x_n^{(k)} \right),$$

$$u_3 = f \left(x_1^{(k)}, x_2^{(k)}, \dots, x_i^{(k)} - \Delta x_i^{(k)}, \dots, x_n^{(k)} \right),$$

- Смешанные производные по формуле

$$\frac{\partial f(\bar{x}^{(k)})}{\partial x_i \partial x_j} \approx \frac{u_1 - u_2 - u_3 + u_4}{\Delta x_i^{(k)} \Delta x_j^{(k)}}$$

где

$$\begin{aligned} u_1 &= f(x_1^{(k)}, x_2^{(k)}, \dots, x_i^{(k)}, \dots, x_n^{(k)}) \\ u_2 &= f(x_1^{(k)}, x_2^{(k)}, \dots, x_i^{(k)} - \Delta x_i^{(k)}, \dots, x_j^{(k)}, \dots, x_n^{(k)}), \\ u_3 &= f(x_1^{(k)}, x_2^{(k)}, \dots, x_i^{(k)}, \dots, x_j^{(k)} - \Delta x_j^{(k)}, \dots, x_n^{(k)}), \\ u_4 &= f(x_1^{(k)}, x_2^{(k)}, \dots, x_i^{(k)} - \Delta x_i^{(k)}, \dots, x_j^{(k)} - \Delta x_j^{(k)}, \dots, x_n^{(k)}). \end{aligned}$$

Алгоритм метода минимизации целевой функции $f(\bar{x})$ методом наискорейшего спуска заключается в следующем:

1. Задать размерность задачи оптимизации n , координаты начальной точки $\bar{x}^{(0)} = (x_1^{(0)}, x_2^{(0)}, \dots, x_n^{(0)})$, точка поиска ε .
2. Положить счетчик числа $k = 0$.
3. Определить направление вектора градиента целевой функции $f(\bar{x})$ $\nabla f(\bar{x}^{(k)}) = \left(\frac{\partial f(\bar{x}^{(k)})}{\partial x_1}, \frac{\partial f(\bar{x}^{(k)})}{\partial x_2}, \dots, \frac{\partial f(\bar{x}^{(k)})}{\partial x_n} \right)$ в точке $\bar{x}^{(k)} = (x_1^{(k)}, x_2^{(k)}, \dots, x_n^{(k)})$. Для вычисления координат вектора градиента использовать разностную формулу первых частных производных.
4. Проверить условие окончания поиска

$$\|\nabla f(\bar{x}^{(k)})\| = \sqrt{\sum_{i=1}^n \left(\frac{\partial f(\bar{x}^{(k)})}{\partial x_i} \right)^2} \leq \varepsilon.$$

Если условие выполнено, то расчет окончен $\bar{x}^* = \bar{x}^{(k)}$, иначе перейти к пункту 5.

5. Вычислить шаг h_k по формуле

$$h_k = \frac{(\nabla f(\bar{x}^{(k)}), \nabla f(\bar{x}^{(k)}))}{(H(\bar{x}^{(k)}) \nabla f(\bar{x}^{(k)}), \nabla f(\bar{x}^{(k)}))}$$

используя результаты вычислений пункта 3 и разностные формулы вторых и смешанных производных.

6. Определить координаты точки

$$\overline{x^{(k+1)}} = \overline{x^{(k)}} - h_k \nabla f(\overline{x^{(k)}})$$

положить $k = k + 1$ и перейти к пункту 3.

2.2.1. Практический расчет задачи

Найти минимум целевой функции

$$f(\bar{x}) = 2.8 * x_2^2 + 1.9 * x_1 + 2.7 * x_1^2 + 1.6 - 1.9 * x_2$$

методом наискорейшего градиентного спуска с точностью $\varepsilon = 0,1$.

Решение. За начальную точку примем

$$\overline{x^{(0)}} = (x_1^{(0)}, x_2^{(0)})^T = (0.5, 0.25)^T.$$

Найдем градиент функции в произвольной точке $\overline{x^{(k)}} = (x_1^{(k)}, x_2^{(k)})^T$.

$$\nabla f(\overline{x^{(k)}}) = \left(\frac{\partial f(\overline{x^{(k)}})}{\partial x_1}; \frac{\partial f(\overline{x^{(k)}})}{\partial x_2} \right)^T = (5.4 * x_1^{(k)} + 1.9; 5.6 * x_2^{(k)} - 1.9)^T.$$

Итерация $k=0$. Вычислим градиент функции $\nabla f(\overline{x^{(0)}})$ в начальной точке $\overline{x^{(0)}}$:

$$\nabla f(\overline{x^{(0)}}) = (4,6; -0,5)^T$$

Определим координаты точки $\overline{x^{(1)}}$ по формуле

$$\overline{x^{(1)}} = \overline{x^{(0)}} - h_0 \nabla f(\overline{x^{(0)}}) = (0,5; 0,25)^T - h_0(4,6; -0,5)^T$$

Численным методом определим значения шага $h_0 = 0,182$.

Найденное значение величины шага h_0 обеспечивает минимум функции $\varphi(h_0)$. По данной величине шага находим координаты точки $\overline{x^{(1)}}$:

$$\overline{x^{(1)}} = \overline{x^{(0)}} - h_0 \nabla f(\overline{x^{(0)}}) = (0,5; 0,25)^T - h_0(4,6; -0,5)^T = (-0,337; 0,341)^T.$$

Составим таблицу 1.

Таблица 1.

Точка	Координаты точки		Значение функции
	1	2	
$\overline{x^{(0)}}$	$x_1^{(0)} = 0,5$	$x_2^{(0)} = 0,25$	$f(\overline{x^{(0)}}) = 2,925$
$\overline{x^{(1)}}$	$x_1^{(1)} = -0,337$	$x_2^{(1)} = 0,341$	$f(\overline{x^{(1)}}) = 0,944$

Проверим условие окончания процесса поиска. Для этого вычислим градиент целевой функции $\nabla f(\overline{x^{(1)}})$ в точке $\overline{x^{(1)}}$: $\nabla f(\overline{x^{(1)}}) = (0,083; 0,009)^T$.

Так как норма вектора градиента $\|\nabla f(\overline{x^{(1)}})\| = 0,083 < \varepsilon$, то требуемая точность достигнута и точка $\overline{x^{(1)}}$ есть найденное приближение точки минимума $\overline{x^*} = (-0,337; 0,341)^T$, $f(\overline{x^*}) = 0,944$.

2.2.2. Программная реализация

Программный код выполнен на языке Java.

```
import java.util.ArrayList;
```

```
public class FastGradient extends Differential {
    //Массив векторов
    private static ArrayList<CoordRow> arr = new ArrayList<>();
    //Размерность
    private int dimension = 2;
    //Точность численных вычислений
    private double accuracy = 0.00000001d;
    //Точность искомого значения функции
    private double eps = 0.1;
    //Стартовый вектор
    private double[] basisVector = {0.5, 0.25};

    private FastGradient() {
        arr.add(new CoordRow(cloneVector(basisVector),
getFuncValue(basisVector)));
    }
    public static void main(String[] args) {
        FastGradient gradient = new FastGradient();
        int i = 0;
        while (true) {
            System.out.println("\nИтерация " + i++);
            if (gradient.run()) break;
        }
        gradient.printTable(gradient.getArr());
    }
}
```

```

    }
    public ArrayList<CoordRow> getArr() {
        return arr;
    }
    @Override
    protected double getFuncValue(double[] x) {
        return 2.8d*x[1]*x[1] + 1.9d*x[0] + 2.7d*x[0]*x[0] + 1.6d - 1.9*x[1];
    }
    private boolean run() {
        printVector(basisVector, "Координаты базисного вектора: ");
        printValue(getFuncValue(basisVector), "Скалярное значение базисного
вектора: ");
        //Считаем значение градиента
        double[] gradientVector = getGradientValue(basisVector, accuracy);
        printVector(gradientVector, "Координаты вектора градиента: ");
        //Считаем значение нормы
        double norma = getNorma(gradientVector);
        printValue(norma, "Значение норма вектора: ");
        //Проверяем условие окончания поиска
        if (norma <= eps) {
            return true;
        } else {
            //Считаем значение шага
            double step = getStep(basisVector, accuracy, dimension);
            printValue(step, "Новый шаг: ");
            //Считаем новый вектор
            basisVector = countNewVector(basisVector, gradientVector, step);
            printVector(basisVector, "Координаты нового вектора: ");
            printValue(getFuncValue(basisVector), "Скалярное значение нового
вектора: ");
            arr.add(new CoordRow(cloneVector(basisVector),
getFuncValue(basisVector)));
        }
        return false;
    }
}

```


Результат работы программы (рис. 2.4):

```

Итерация 0
Координаты базисного вектора:
0.5      0.25
Скалярное значение базисного вектора:
2.925
Координаты вектора градиента:
4.6      -0.5
Значение норма вектора:
4.627
Новый шаг:
0.182
Координаты нового вектора:
-0.337    0.341
Скалярное значение нового вектора:
0.944

Итерация 1
Координаты базисного вектора:
-0.337    0.341
Скалярное значение базисного вектора:
0.944
Координаты вектора градиента:
0.083     0.009
Значение норма вектора:
0.083

Таблица векторов:
0.5      0.25      2.925
-0.337    0.341    0.944

```

Рис.2.4. Результат работы программы

2.2.3. Контрольные вопросы

1. Отличие метода наискорейшего спуска от метода градиентного спуска.
2. Сущность метода наискорейшего спуска.
3. Формула определения величины шага.
4. Сформулируйте матрицу Гессе.
5. Проиллюстрируйте траекторию движения к точке минимума в методе наискорейшего спуска
6. Алгоритм метода наискорейшего спуска
7. Формула определения первых частных производных.
8. Формула определения вторых частных производных.
9. Формула определения смешанных частных производных.
10. Определение вектора градиента целевой функции.
11. Проверка условия окончания поиска.

2.3. Метод покоординатного спуска.

Суть данного метода заключается в следующем. Выбирается произвольная начальная точка

$$\overline{x^{(0)}} = (x_1^{(0)}, x_2^{(0)}, \dots, x_n^{(0)})$$

Из области определения целевой функции $f(\bar{x})$.

В первой итерации фиксируются все координаты целевой функции, кроме первой x_1 . Находится минимум функции одной переменной $f(x_1, x_2^{(0)}, \dots, x_n^{(0)})$ в направлении проекций вектора антиградиента $-f(\overline{x^{(0)}}) \cdot \bar{e}_1$ на ось x_1 , где \bar{e}_1 — единичный вектор этой оси. В результате решения одномерной задачи оптимизации находится точка $\overline{x^{(1)}} = (x_1^{(1)}, x_2^{(0)}, \dots, x_n^{(0)})$, в которой целевая функция $f(x_1, x_2^{(0)}, \dots, x_n^{(0)})$ принимает минимальное значение по координате x_1 .

В следующей итерации фиксируют все координаты целевой функции, кроме координаты

$$x_2, f(x_1^{(1)}, x_2, \dots, x_n^{(0)}).$$

Снова решается одномерная задача оптимизации и находится точка

$\overline{x^{(2)}} = (x_1^{(1)}, x_2^{(1)}, x_3^{(0)}, \dots, x_n^{(0)})$, в которой целевая функция $f(x_1^{(1)}, x_2, \dots, x_n^{(0)})$ принимает минимум вдоль направления проекции вектора антиградиента $-f(\overline{x^{(1)}}) \cdot \bar{e}_2$ на ось x_2 , здесь \bar{e}_2 — единичный вектор этой оси.

Аналогично находятся точки минимума по координатам x_3, x_4, \dots, x_n .

Одномерный поиск на каждой итерации проводится в соответствии с формулой

$$\overline{x^{(k+1)}} = \overline{x^{(k)}} - h_k \nabla f(\overline{x^{(k)}}) \cdot \bar{e}_{k+1},$$

где h_k — величина шага, обеспечивающая максимально возможное уменьшение целевой функции $f(\bar{x})$,

k — номер итерации $k = \overline{0, (n-1)}$. После завершения n итераций точка $\overline{x^{(n)}}$ берется за начальную точку $\overline{x^{(0)}} = \overline{x^{(n)}}$ и итерации повторяются снова.

В качестве условия окончания поиска используется близость к нулю нормы градиента $\|\nabla f(\overline{x^{(k+1)}})\| \leq \varepsilon$.

Величина шага h_k для каждого значения k определяется из решения вспомогательной одномерной задачи минимизации

$f(h_k) = f(\bar{x}^{(k)} - h_k \nabla f(\bar{x}^{(k)})) \rightarrow \min_{h_k > 0}$, которая может быть решена аналитически или численно. При квадратичной интерполяции целевой функции величину шага h_k можно определить по формуле

$$h_k = \frac{\left(\nabla f(\bar{x}^{(k)}), \nabla f(\bar{x}^{(k)}) \right)}{\left(H(\bar{x}^{(k)}) \nabla f(\bar{x}^{(k)}), \nabla f(\bar{x}^{(k)}) \right)}$$

При ее использовании необходимо учитывать текущие значения переменных на каждой итерации.

На рисунке 2.5 представлена иллюстрация последовательных приближений к точке минимума \bar{x}^* методом покоординатного спуска для случая двух переменных. Траектория спуска производится по ломаной линии, состоящей из отрезков прямых, параллельных осям координат.

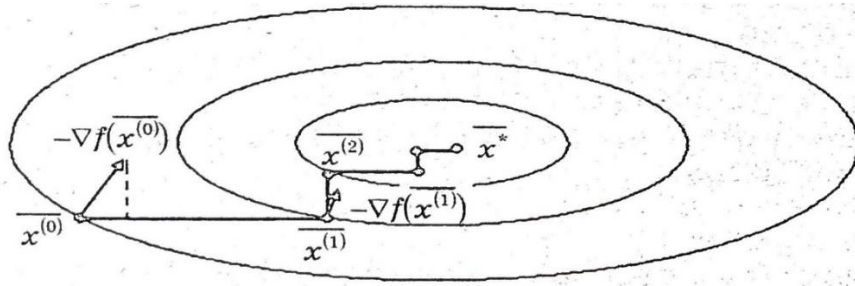


Рис. 2.5. Графическая иллюстрация поиска точки минимума методом покоординатного спуска

2.3.1. Практический расчет задачи

Найти минимум целевой функции

$$f(\bar{x}) = 2.8 * x_2^2 + 1.9 * x_1 + 2.7 * x_1^2 + 1.6 - 1.9 * x_2$$

методом покоординатного спуска с точностью $\varepsilon = 0,1$.

Решение. За начальную точку примем:

$$\bar{x}^0 = (x_1^{(0)}, x_2^{(0)})^T = (-0,25, 0,25)^T.$$

Найдем градиент функции в произвольной точке $\bar{x}^{(k)} = (x_1^{(k)}, x_2^{(k)})^T$

$$\nabla f(\bar{x}^{(k)}) = \left(\frac{\partial f(\bar{x}^{(k)})}{\partial x_1}; \frac{\partial f(\bar{x}^{(k)})}{\partial x_2} \right)^T = (5.4 * x_1^{(k)} + 1.9; 5.6 * x_2^{(k)} - 1.9)^T.$$

Итерация $k = 0$. Спуск по координате x_1 . Фиксируем координату $x_2 = x_2^{(0)} = 0,25$ в целевой функции $f(\bar{x})$. Тогда

$f(x_1, x_2^{(0)}) = 2.8 * (x_2^{(0)})^2 + 1.9 * x_1 + 2.7 * x_1^2 + 1.6 - 1.9 * x_2^{(0)} =$
 $\varphi(x_1)$ – функция одной переменной. Вычислим градиент функции $\nabla f(\bar{x}^{(0)})$
 в начальной точке $\bar{x}^{(0)}: \nabla f(\bar{x}^{(0)}) = (0,55; -0,5)^T$.

Определим координату x_1 по формуле:

$$x_1 = x_1^{(0)} - h_0 \nabla f(\bar{x}^{(0)}) \cdot \bar{e}_1 = -0,25 - h_0(0,55; -0,5)^T \begin{pmatrix} 1 \\ 0 \end{pmatrix} \\ = -0,25 - 0,55h_0$$

Численным методом определим значения шага $h_0 = 0,099$.

Найденное значение величины шага h_0 обеспечивает минимум функции $\varphi(h_0)$. По данной величине шага находим координату точки $x_1^{(1)}$:

$$x_1^{(1)} = x_1^{(0)} - h_0 \nabla f(\bar{x}^{(0)}) \cdot \bar{e}_1 = -0,25 - 0,099 * (0,55; -0,5)^T \begin{pmatrix} 1 \\ 0 \end{pmatrix} \\ = -0,305.$$

Таким образом, переходим от точки $\bar{x}^{(0)} = (x_1^{(0)}, x_2^{(0)})$ к точке

$\bar{x}^{(1)} = (x_1^{(1)}, x_2^{(0)})^T = (-0,305, 0,25)$, в которой целевая функция $f(\bar{x})$ принимает наименьшее значение по координате x_1 .

Составим таблицу 1.

Таблица 1.

Точка	Координаты точки		Значение функции
	1	2	
$x^{(0)}$	$x_1^{(0)} = -0,25$	$x_2^{(0)} = 0,25$	$f(\bar{x}^{(0)}) = 0,994$
$x^{(1)}$	$x_1^{(1)} = -0,305$	$x_2^{(0)} = 0,25$	$f(\bar{x}^{(1)}) = 0,972$

Проверим условие окончания процесса поиска для этого вычислим градиент целевой функции $\nabla f(\bar{x}^{(1)})$ в точке $\bar{x}^{(1)}: \nabla f(\bar{x}^{(1)}) = (0,256; -0,5)^T$. Так как норма вектора градиента $\|\nabla f(\bar{x}^{(1)})\| = 0,562 > \varepsilon$, то переходим к следующей итерации.

Итерация $k = 1$. Спуск по координате x_1 . Фиксируем координату $x_1 = x_1^{(1)} = -0,305$ в целевой функции $f(\bar{x})$. Тогда

$f(x_1^{(1)}, x_2) = 2.8 * x_2^2 + 1.9 * x_1^{(1)} + 2.7 * (x_1^{(1)})^2 + 1.6 - 1.9 * x_2 =$
 $\varphi(x_2)$ – функция одной переменной. Вычислим градиент функции $\nabla f(\bar{x}^{(1)})$
 в начальной точке $\bar{x}^{(1)}: \nabla f(\bar{x}^{(1)}) = (0,256; -0,5)^T$.

Определим координату x_2 по формуле:

$$x_2 = x_2^{(0)} - h_0 \nabla f(\bar{x}^{(1)}) \cdot \bar{e}_1 = 0,25 - h_0(0,256; -0,5)^T \begin{pmatrix} 0 \\ 1 \end{pmatrix} = 0,25 - 0,5h_0$$

Численным методом определим значения шага $h_0 = 0,161$.

Найденное значение величины шага h_0 обеспечивает минимум функции $\varphi(h_0)$. По данной величине шага находим координату точки $x_2^{(1)}$:

$$x_2^{(1)} = x_2^{(0)} - h_0 \nabla f(\bar{x}^{(1)}) \cdot \bar{e}_1 = 0,25 - h_0(0,256; -0,5)^T \begin{pmatrix} 0 \\ 1 \end{pmatrix} = 0,331.$$

Таким образом, переходим от точки $\bar{x}^{(1)} = (x_1^{(1)}, x_2^{(0)})$ к точке

$\bar{x}^{(2)} = (x_1^{(1)}, x_2^{(1)})^T = (-0,305, 0,331)$, в которой целевая функция $f(\bar{x})$ принимает наименьшее значение по координате x_1 .

Составим таблицу 2.

Таблица 2.

Точка	Координаты точки		Значение функции
	1	2	
$x^{(0)}$	$x_1^{(0)} = -0,25$	$x_2^{(0)} = 0,25$	$f(\bar{x}^{(0)}) = 0,994$
$x^{(1)}$	$x_1^{(1)} = -0,305$	$x_2^{(0)} = 0,25$	$f(\bar{x}^{(0)}) = 0,972$
$x^{(2)}$	$x_1^{(1)} = -0,305$	$x_2^{(1)} = 0,331$	$f(\bar{x}^{(0)}) = 0,95$

Проверим условие окончания процесса поиска для этого вычислим градиент целевой функции $\nabla f(\bar{x}^{(2)})$ в точке $\bar{x}^{(2)}$: $\nabla f(\bar{x}^{(2)}) = (0,256; -0,048)^T$. Так как норма вектора градиента $\|\nabla f(\bar{x}^{(2)})\| = 0,26 > \varepsilon$, то переходим к следующей итерации.

Итерация $k = 2$. Спуск по координате x_1 . Фиксируем координату $x_2 = x_2^{(1)} = 0,331$ в целевой функции $f(\bar{x})$. Тогда

$f(x_1, x_2^{(1)}) = 2,8 * (x_2^{(1)})^2 + 1,9 * x_1 + 2,7 * x_1^2 + 1,6 - 1,9 * x_2^{(1)} = \varphi(x_1)$ – функция одной переменной. Вычислим градиент функции $\nabla f(\bar{x}^{(2)})$ в начальной точке $\bar{x}^{(0)}$: $\nabla f(\bar{x}^{(0)}) = (0,256; -0,048)^T$.

Определим координату x_1 по формуле:

$$\begin{aligned} x_1 &= x_1^{(1)} - h_0 \nabla f(\bar{x}^{(2)}) \cdot \bar{e}_1 = -0,305 - h_0(0,256; -0,048)^T \begin{pmatrix} 1 \\ 0 \end{pmatrix} \\ &= -0,305 - 0,256h_0 \end{aligned}$$

Численным методом определим значения шага $h_0 = 0,221$.

Найденное значение величины шага h_0 обеспечивает минимум функции $\varphi(h_0)$. По данной величине шага находим координату точки $x_1^{(2)}$:

$$x_1^{(2)} = x_1^{(1)} - h_0 \nabla f(\bar{x}^{(2)}) \cdot \bar{e}_1 = -0,305 - 0,221 * (0,256; -0,048)^T \begin{pmatrix} 1 \\ 0 \end{pmatrix} = -0,361.$$

Таким образом, переходим от точки $\bar{x}^{(2)} = (x_1^{(1)}, x_2^{(1)})$ к точке $\bar{x}^{(3)} = (x_1^{(2)}, x_2^{(1)})^T = (-0,361, 0,331)$, в которой целевая функция $f(\bar{x})$ принимает наименьшее значение по координате x_1 .

Составим таблицу 3.

Таблица 3.

Точка	Координаты точки		Значение функции
	1	2	
$x^{(0)}$	$x_1^{(0)} = -0,25$	$x_2^{(0)} = 0,25$	$f(\bar{x}^{(0)}) = 0,994$
$x^{(1)}$	$x_1^{(1)} = -0,305$	$x_2^{(0)} = 0,25$	$f(\bar{x}^{(0)}) = 0,972$
$x^{(2)}$	$x_1^{(1)} = -0,305$	$x_2^{(1)} = 0,331$	$f(\bar{x}^{(0)}) = 0,95$
$x^{(3)}$	$x_1^{(2)} = -0,361$	$x_2^{(1)} = 0,331$	$f(\bar{x}^{(0)}) = 0,944$

Проверим условие окончания процесса поиска для этого вычислим градиент целевой функции $\nabla f(\bar{x}^{(3)})$ в точке $\bar{x}^{(3)}$: $\nabla f(\bar{x}^{(3)}) = (-0,05; -0,048)^T$. Так как норма вектора градиента $\|\nabla f(\bar{x}^{(3)})\| = 0,07 < \varepsilon$, то требуемая точность достигнута и точка $\bar{x}^{(3)}$ есть найденное приближение точки минимума $\bar{x}^* = (-0,361, 0,331)^T, f(\bar{x}^*) = 0,944$.

2.3.2. Программная реализация

Программный код выполнен на языке Java.

```
import java.util.ArrayList;

public class CoordGradient extends Differential{
    //Массив векторов
    private static ArrayList<CoordRow> arr = new ArrayList<>();
    //Размерность
    private int dimension = 2;
    //Точность численных вычислений
    private double accuracy = 0.00000001d;
    //Точность искомого значения функции
    private double eps = 0.1;
```

```

//Стартовый вектор
private double[] basisVector = {-0.25, 0.25};
//Индекс текущей координаты
private int numCoord = -1;

private CoordGradient() {
    arr.add(new CoordRow(cloneVector(basisVector),
getFuncValue(basisVector)));
}

public static void main(String[] args) {
    CoordGradient gradient = new CoordGradient();
    int i = 0;
    while (true) {
        System.out.println("\nИтерация " + i++);
        if (gradient.run()) break;
    }
    gradient.printTable(gradient.getArr());
}
public ArrayList<CoordRow> getArr() {
    return arr;
}
@Override
protected double getFuncValue(double[] x) {
    return 2.8d*x[1]*x[1] + 1.9d*x[0] + 2.7d*x[0]*x[0] + 1.6d - 1.9*x[1];
}
private boolean run() {
    //Определяем, какую координату будем менять
    numCoord = numCoord == (dimension - 1) ? 0 : (numCoord + 1);
    printVector(basisVector, "Координаты базисного вектора: ");
    printValue(getFuncValue(basisVector), "Скалярное значение базисного
вектора: ");
    //Считаем значение веткора градиента
    double[] gradientVector = getGradientValue(basisVector, accuracy);
    printVector(gradientVector, "Координаты вектора градиента: ");
    //Считаем норму
    double norma = getNorma(gradientVector);
    printValue(norma, "Значение норма вектора: ");
    //Проверяем условие окончания
    if (norma <= eps) {
        return true;
    } else {
        //Считаем шаг

```

```

double step = getStep(basisVector, accuracy, dimension);
printValue(step, "Новый шаг: ");
//Считаем временный вектор
double[] tmpVector = countNewVector(basisVector, gradientVector,
step);
//Меняем значение координаты
basisVector[numCoord] = tmpVector[numCoord];
printVector(basisVector, "Координаты нового вектора: ");
printValue(getFuncValue(basisVector), "Скалярное значение нового
вектора: ");
arr.add(new CoordRow(cloneVector(basisVector),
getFuncValue(basisVector)));
}
return false;
}
}

```

Результат работы программы (рис.2.6):

```

Итерация 2
Координаты базисного вектора:
-0.305      0.331
Скалярное значение базисного вектора:
0.95
Координаты вектора градиента:
0.256      -0.048
Значение норма вектора:
0.26
Новый шаг:
0.221
Координаты нового вектора:
-0.361      0.331
Скалярное значение нового вектора:
0.944

Итерация 3
Координаты базисного вектора:
-0.361      0.331
Скалярное значение базисного вектора:
0.944
Координаты вектора градиента:
-0.05      -0.048
Значение норма вектора:
0.07

Таблица векторов:
-0.25      0.25      0.994
-0.305     0.25      0.972
-0.305     0.331     0.95
-0.361     0.331     0.944

```

Рис. 2.6. Результат работы программы

2.3.3. Контрольные вопросы

1. Сущность метода покоординатного спуска.
2. Формула определения величины шага.
3. Проиллюстрируйте траекторию движения к точке минимума в методе покоординатного спуска.

2.4. Метод Флетчера-Ривса.

Идея метода Флетчера—Ривса состоит в том, что на каждом шаге в качестве направления спуска используется линейная комбинация вектора градиента с прежним направлением спуска. Последовательность приближений к точке минимума целевой функции $f(x)$ определяется по формуле

$$\overline{x^{(k+1)}} = \overline{x^{(k)}} + h_k \overline{p^{(k)}}, \quad k = 0, 1, \dots,$$

где k — номер итерации ($k=0, 1, \dots$),

$\overline{x^{(0)}}$ — начальное приближение,

h_k — величина шага,

$\overline{p^{(k)}}$ — направление спуска.

Направление спуска $\overline{p^{(k)}}$ определяется в зависимости от номера текущей итерации k :

- Если значение $k = 0$ направление спуска $\overline{p^{(0)}}$ совпадает с направлением вектора антиградиента целевой функции — $\nabla f(\overline{x^{(0)}})$ в точке $\overline{x^{(0)}}$.
- Если значение $k = 1, 2, \dots$ направление спуска определяется по формуле

$$\overline{p^{(k)}} = \nabla f(\overline{x^{(k)}}) + \beta_{k-1} \overline{p^{(k-1)}}$$

где

$$\beta_{k-1} = \frac{\|\nabla f(\overline{x^{(k)}})\|^2}{\|\nabla f(\overline{x^{(k-1)}})\|^2} = \frac{\sum_{i=1}^n \left(\frac{\partial f(\overline{x^{(k)}})}{\partial x_i} \right)^2}{\sum_{i=1}^n \left(\frac{\partial f(\overline{x^{(k-1)}})}{\partial x_i} \right)^2}$$

и называется сопряженным.

В качестве условия окончания поиска используется близость к нулю нормы градиента $\|\nabla f(\overline{x^{(k+1)}})\| \leq \varepsilon$.

Для минимизации квадратичной целевой функции n независимых переменных методом Флетчера—Ривса требуется не более n шагов. Для уменьшения влияния накапливающихся погрешностей вычислений через каждые n итераций проводят обновление метода, полагая $\overline{x^{(0)}} = \overline{x^{(n)}}$.

Иллюстрация последовательных приближений к точке минимума представлена на рис. 2.7.

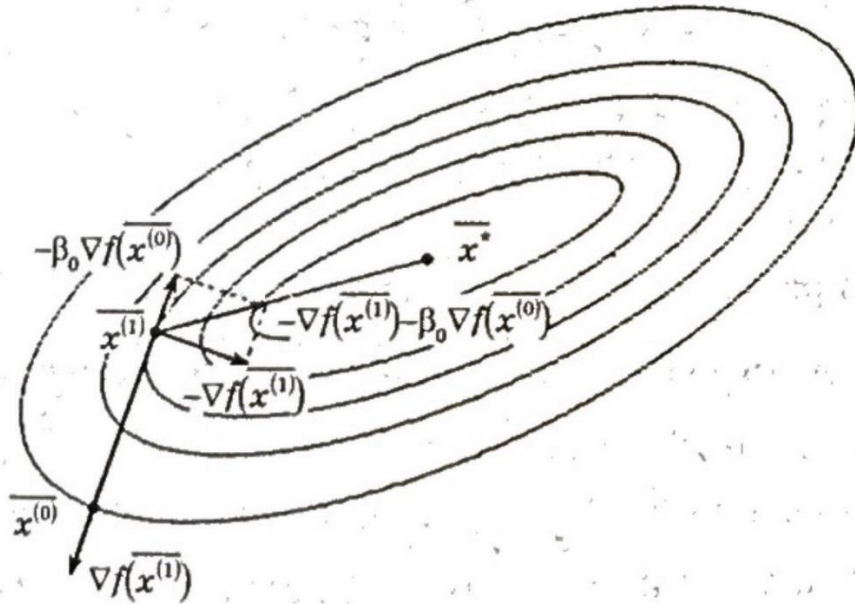


Рис. 2.7. Траектория поиска минимума методом Флетчера—Ривса
 Величина шага h_k , определяется из решения вспомогательной одномерной задачи минимизации $\varphi(h_k) = f(\bar{x}^{(k)} - h_k \nabla f(\bar{x}^{(k)})) \rightarrow \min$ которая может быть решена аналитически или численно. При квадратичной интерполяции целевой функции величину шага можно определить по формуле, аналогичной формуле метода наискорейшего градиентного спуска.

$$h_k = \frac{(\nabla f(\bar{x}^{(k)}), \bar{p}^{(k)})}{(H(\bar{x}^{(k)})\bar{p}^{(k)}, \bar{p}^{(k)})}$$

где $H(\bar{x}^{(k)})$ — матрица Гессе, вычисленная в точке $\bar{x}^{(k)}$.

Алгоритм метода Флетчера—Ривса

1. Задать размерность задачи оптимизации n , координаты начальной точки $\bar{x}^{(0)} = (x_1^{(0)}, x_2^{(0)}, \dots, x_n^{(0)})$, точность поиска ε .

2. Положить счетчик числа итераций $k = 0$.

3. Определить направление вектора градиента

$$\nabla f(\bar{x}^{(k)}) = \left(\frac{\partial f(\bar{x}^{(k)})}{\partial x_1}, \frac{\partial f(\bar{x}^{(k)})}{\partial x_2}, \dots, \frac{\partial f(\bar{x}^{(k)})}{\partial x_n} \right)$$

целевой функции $f(\bar{x})$ в точке

$\bar{x}^{(k)} = (x_1^{(k)}, x_2^{(k)}, \dots, x_n^{(k)})$. Для вычисления координат вектора градиента использовать расчетную формулу первых частных производных.

4. Проверить условие окончания поиска

$$\|\nabla f(\overline{x^{(k)}})\| = \sqrt{\sum_{i=1}^n \left(\frac{\partial f(\overline{x^{(k)}})}{\partial x_i}\right)^2} \leq \varepsilon$$

Если условие выполнено то расчет окончен

$\overline{x^*} = \overline{x^{(k)}}$, иначе перейти к пункту 5.

5. Если $k = 0$, то вычислить $\overline{p^{(0)}} = -\nabla f(\overline{x^{(0)}})$ и перейти к шагу 8, иначе – перейти к шагу 6.

7. Вычислить

$$\beta_{k-1} = \frac{\|\nabla f(\overline{x^{(k)}})\|^2}{\|\nabla f(\overline{x^{(k-1)}})\|^2} = \frac{\sum_{i=1}^n \left(\frac{\partial f(\overline{x^{(k)}})}{\partial x_i}\right)^2}{\sum_{i=1}^n \left(\frac{\partial f(\overline{x^{(k-1)}})}{\partial x_i}\right)^2}$$

7. Вычислить $\overline{p^{(k)}} = -\nabla f(\overline{x^{(k)}}) + \beta_{k-1} \overline{p^{(k-1)}}$.

8. Вычислить шаг h_k формуле,

$$h_k = \frac{(\nabla f(\overline{x^{(k)}}), \overline{p^{(k)}})}{(H(\overline{x^{(k)}}) \overline{p^{(k)}}), \overline{p^{(k)}})}$$

используя результаты пункта 3 и разностные формулы вторых и смешанных производных.

9. Определить координаты точки $\overline{x^{(k+1)}} = \overline{x^{(k)}} + h_k \overline{p^{(k)}}$,

Положить $k = k + 1$ и перейти к пункту 3.

2.4.1. Практический расчет задачи

Найти минимум целевой функции

$$f(\overline{x}) = 2.8 * x_2^2 + 1.9 * x_1 + 2.7 * x_1^2 + 1.6 - 1.9 * x_2$$

методом Флетчера – Ривса с точностью $\varepsilon = 0.1$.

Решение. За начальную точку примем

$$\overline{x^{(0)}} = (x_1^{(0)}, x_2^{(0)})^T = (-0.25, 0.25)^T$$

Найдем градиент функции в произвольной точке $\overline{x^{(k)}} = (x_1^{(k)}, x_2^{(k)})^T$

$$\nabla f(\overline{x^{(k)}}) = \left(\frac{\partial f(\overline{x^{(k)}})}{\partial x_1}; \frac{\partial f(\overline{x^{(k)}})}{\partial x_2} \right)^T = (5.4 * x_1^{(k)} + 1.9; 5.6 * x_2^{(k)} - 1.9)^T.$$

Итерация $k = 0$. Определим координаты точки $\overline{x^{(1)}} = \overline{x^{(0)}} + h_0 \overline{p^{(0)}}$.

Для этого вычислим градиент функции $\nabla f(\overline{x^{(0)}})$ в начальной точке $\overline{x^{(0)}}$: $\nabla f(\overline{x^{(0)}}) = (0,55; -0,5)^T$. Положим $\overline{p^{(0)}} = -\nabla f(\overline{x^{(0)}}) = (-0,55; 0,5)^T$. Вычислим координаты точки $\overline{x^{(1)}}$

$$\overline{x^{(1)}} = \overline{x^{(0)}} - h_0 \overline{p^{(0)}} = (-0,25; 0,25)^T - h_0(-0,55; 0,5)^T$$

Численным методом определим значения шага $h_0 = -0,099$.

Найденное значение величины шага h_0 обеспечивает минимум функции $\varphi(h_0)$. По данной величине шага находим координату точки $\overline{x^{(1)}}$:

$$\overline{x^{(1)}} = \overline{x^{(0)}} - h_0 \overline{p^{(0)}} = (-0,25; 0,25)^T + 0,099 * (-0,55; 0,5)^T = (-0,305; 0,3)^T.$$

Составим таблицу 1.

Таблица 1.

Точка	Координаты точки		Значение функции
	1	2	
$\overline{x^{(0)}}$	$\overline{x_1^{(0)}} = -0,25$	$\overline{x_2^{(0)}} = 0,25$	$f(\overline{x^{(0)}}) = 0,994$
$\overline{x^{(1)}}$	$\overline{x_1^{(1)}} = -0,305$	$\overline{x_2^{(1)}} = 0,3$	$f(\overline{x^{(1)}}) = 0,954$

Проверим условие окончания процесса поиска для этого вычислим градиент целевой функции $\nabla f(\overline{x^{(1)}})$ в точке $\overline{x^{(1)}}$: $\nabla f(\overline{x^{(1)}}) = (0,256; -0,222)^T$. Так как норма вектора градиента $\|\nabla f(\overline{x^{(1)}})\| = 0,339 > \varepsilon$, то переходим к следующей итерации.

Итерация $k = 1$.

Определим координаты точки по формуле:

$$\overline{x^{(2)}} = \overline{x^{(1)}} + h_1 \overline{p^{(1)}}$$

где $\overline{p^{(1)}} = -\nabla f(\overline{x^{(1)}}) + \beta_0 \overline{p^{(0)}}$.

Определим составляющие вектора $\overline{p^{(1)}}$:

Вычислим градиент целевой функции $\nabla f(\overline{x^{(1)}})$ в точке

$$\overline{x^{(1)}}: \nabla f(\overline{x^{(1)}}) = (0,256; -0,222)^T.$$

Найдем координаты вектора антиградиента

$$-\nabla f(\overline{x^{(1)}}) = (-0,256; 0,222)^T; \beta_0 = \frac{\|\nabla f(\overline{x^{(1)}})\|^2}{\|\nabla f(\overline{x^{(0)}})\|^2} = 0,208.$$

$$\overline{x^{(2)}} = \overline{x^{(1)}} + h_1 \overline{p^{(1)}} = (-0,305; 0,3)^T + h_1 [(-0,256; 0,222)^T + 0,208 * (-0,55; 0,5)^T]$$

Численным методом определим значения шага $h_1 = -0,099$.

Найденное значение величины шага h_1 обеспечивает минимум функции $\varphi(h_1)$. По данной величине шага находим координату точки $\overline{x^{(2)}}$:

$$\overline{x^{(2)}} = \overline{x^{(1)}} + h_1 \overline{p^{(1)}} = (-0,305, 0,3)^T + h_1 [(-0,256; 0,222)^T + 0,208 * (-0,55; 0,5)^T] = (-0,356; 0,345)^T.$$

Составим таблицу 2.

Таблица 2.

Точка	Координаты точки		Значение функции
	1	2	
$x^{(0)}$	$x_1^{(0)} = -0,25$	$x_2^{(0)} = 0,25$	$f(\overline{x^{(0)}}) = 0,994$
$x^{(1)}$	$x_1^{(1)} = -0,305$	$x_2^{(1)} = 0,3$	$f(\overline{x^{(1)}}) = 0,954$
$x^{(2)}$	$x_1^{(2)} = -0,356$	$x_2^{(2)} = 0,345$	$f(\overline{x^{(2)}}) = 0,944$

Проверим условие окончания процесса поиска для этого вычислим градиент целевой функции $\nabla f(\overline{x^{(2)}})$ в точке $\overline{x^{(2)}}$: $\nabla f(\overline{x^{(2)}}) = (-0,023; 0,032)^T$. Так как норма вектора градиента $\|\nabla f(\overline{x^{(2)}})\| = 0,04 < \varepsilon$, то требуемая точность достигнута и точка $\overline{x^{(2)}}$ есть найденное приближение точки минимума $\overline{x^*} = (-0,356; 0,345)^T, f(\overline{x^*}) = 0,944$.

2.4.2. Программная реализация

Программный код выполнен на языке Java.

```
import java.util.ArrayList;

public class FletcherRives extends Differential{
    //Массив векторов
    private static ArrayList<CoordRow> arr = new ArrayList<>();
    //Размерность
    private int dimension = 2;
    //Точность численных вычислений
    private double accuracy = 0.00000001d;
    //Точность искомого значения функции
    private double eps = 0.1;
    //Количества итераций до обновления
    private int numIterToUpdate = 5;
    //Стартовый вектор
    private double[] basisVector = {-0.25, 0.25};
```

```

//Предыдущий вектор
private double[] prevVector;
//Вектор спуска
private double[] descentVector;

private FletcherRives() {
    arr.add(new CoordRow(cloneVector(basisVector),
getFuncValue(basisVector)));
}

public static void main(String[] args) {
    FletcherRives gradient = new FletcherRives();
    int i = 0;
    while (true) {
        System.out.println("\nИтерация " + i);
        if (gradient.run(i++)) break;
    }
    gradient.printTable(gradient.getArr());
}

public ArrayList<CoordRow> getArr() {
    return arr;
}

@Override
protected double getFuncValue(double[] x) {
    return 2.8d*x[1]*x[1] + 1.9d*x[0] + 2.7d*x[0]*x[0] + 1.6d - 1.9*x[1];
}

//Переопределяем метод получения шага
@Override
protected double getStep(double[] vector, double accuracy, int dimension) {
    double[][] gesseMatrix = getGesseMatrix(vector, accuracy, dimension);
    double[][] descentVectorHorizontal = new double[1][dimension];
    descentVectorHorizontal[0] = descentVector;
}

```

```

        double[][] descentVectorVertical =
MatrixOperations.transportMatrix(descentVectorHorizontal);
        double[][] tmp = MatrixOperations.multiplyByMatrix(gesseMatrix,
descentVectorVertical);
        tmp = MatrixOperations.multiplyByMatrix(descentVectorHorizontal, tmp);
        double value = tmp[0][0];
        descentVectorHorizontal[0] = getGradientValue(vector, accuracy);
        tmp = MatrixOperations.multiplyByMatrix(descentVectorHorizontal,
descentVectorVertical);
        return tmp[0][0] / value;
    }

    //Считаем вектор спуска
    private double[] countDescentVector(double[] gradient, double beta, double[]
prevDescentVector) {
        double[] descentVector = new double[gradient.length];
        for (int i = 0; i < gradient.length; i++) {
            descentVector[i] = beta * prevDescentVector[i] - gradient[i];
        }
        return descentVector;
    }

    private boolean run(int iter) {
        printVector(basisVector, "Координаты базисного вектора: ");
        printValue(getFuncValue(basisVector), "Скалярное значение базисного
вектора: ");
        //Считаем значение градиента
        double[] gradientVector = getGradientValue(basisVector, accuracy);
        printVector(gradientVector, "Координаты вектора градиента: ");
        //Считаем значение нормы
        double norma = getNorma(gradientVector);
        printValue(norma, "Значение норма вектора: ");
        double beta;
        //Проверяем условие окончания поиска
        if (norma <= eps) {
            return true;

```

```

    } else {
        //Проверяем необходимость обновления
        if (iter % numIterToUpdate != 0) {
            //Считаем значение сопряженного
            beta = Math.pow(getNorma(getGradientValue(basisVector, accuracy))
                / getNorma(getGradientValue(prevVector, accuracy)), 2);
            printValue(beta, "Значение коэффициента бета: ");
            //Считаем вектор спуска
            descentVector = countDescentVector(getGradientValue(basisVector,
accuracy), beta, descentVector);
        } else {
            //Считаем вектор спуска
            descentVector = getAntiGradientValue(basisVector, accuracy);
        }
        //Считаем значение шага
        double step = getStep(basisVector, accuracy, dimension);
        printValue(step, "Новый шаг: ");
        //Присваиваем предыдущему вектору текущий
        prevVector = basisVector;
        //Считаем новый вектор
        basisVector = countNewVector(basisVector, descentVector, step);
        printVector(basisVector, "Координаты нового вектора: ");
        printValue(getFuncValue(basisVector), "Скалярное значение нового
вектора: ");
        arr.add(new CoordRow(cloneVector(basisVector),
getFuncValue(basisVector)));
    }
    return false;
}
}

```

Результат работы программы (рис.2.8):


```

Итерация 1
Координаты базисного вектора:
-0.305      0.3
Скалярное значение базисного вектора:
0.954
Координаты вектора градиента:
0.256      -0.222
Значение норма вектора:
0.339
Значение коэффициента бета:
0.208
Новый шаг:
-0.139
Координаты нового вектора:
-0.356      0.345
Скалярное значение нового вектора:
0.944

Итерация 2
Координаты базисного вектора:
-0.356      0.345
Скалярное значение базисного вектора:
0.944
Координаты вектора градиента:
-0.023      0.032
Значение норма вектора:
0.04

Таблица векторов:
-0.25      0.25      0.994
-0.305     0.3      0.954
-0.356     0.345    0.944

```

Рис. 2.8. Результат работы программы

2.4.3. Контрольные вопросы

1. Идея метода Флетчера—Ривса.
2. Определение последовательности приближений к точке минимума целевой функции.
3. Определение направления спуска при номере итераций равно нулю.
4. Определение направления спуска при номере итераций больше нуля.
5. Проиллюстрируйте траекторию движения к точке минимума в методе Флетчера—Ривса.
6. Алгоритм метода Флетчера-Ривса.
7. Определение вектора градиента целевой функции.
8. Проверка условия окончания поиска.

Глава 3. Методы второго порядка.

В методах второго порядка при поиске минимума функции многих переменных используют информацию о частных производных целевой функции первого и второго порядка. К этой группе относят метод Ньютона и его модификации.

3.1. Метод Ньютона.

В основе метода Ньютона лежит квадратичная аппроксимация целевой функции. Последовательность итераций строится таким образом, чтобы во вновь получаемой точке градиент аппроксимирующей функции обращался в нуль.

Последовательность приближений строится в соответствии с формулой

$$\overline{x^{(k+1)}} = \overline{x^{(k)}} + \overline{p^{(k)}},$$

где k — номер итерации ($k = 0, 1, \dots$),

$\overline{x^{(0)}}$ — начальное приближение,

$\overline{p^{(k)}} = -H^{-1}(\overline{x^{(k)}})\nabla f(\overline{x^{(k)}})$ — вектор направления спуска.

Здесь $H(\overline{x^{(k)}})$ — матрица Гессе.

Направление спуска $\overline{p^{(k)}}$ ведет к убыванию целевой функции только при положительной определенности матрицы Гессе $H(\overline{x^{(k)}}) > 0$. В тех итерациях, в которых матрица Гессе отрицательно определена $H(\overline{x^{(k)}}) < 0$, последовательность приближений к точке минимума строится по методу наискорейшего градиентного спуска. С этой целью проводится замена вектора направления спуска на антиградиентное $\overline{p^{(k)}} = -h_k \nabla f(\overline{x^{(k)}})$.

Алгоритм метода Ньютона.

1. Задать размерность задачи оптимизации n , координаты начальной точки $\overline{x^{(0)}} = (x_1^{(0)}, x_2^{(0)}, \dots, x_n^{(0)})$, точность поиска ε .

2. Положить счетчик итераций $k = 0$.

3. Определить направление вектора градиента целевой функции

$$\nabla f(\overline{x^{(k)}}) = \left(\frac{\partial f(\overline{x^{(k)}})}{\partial x_1}, \frac{\partial f(\overline{x^{(k)}})}{\partial x_2}, \dots, \frac{\partial f(\overline{x^{(k)}})}{\partial x_n} \right) \text{ в точке}$$

$$\overline{x^{(k)}} = (x_1^{(k)}, x_2^{(k)}, \dots, x_n^{(k)}) .$$

Для вычисления координат вектора градиента использовать разностную формулу первой частной производной.

4. Проверить условие окончания поиска

$$\|\nabla f(\overline{x^{(k)}})\| = \sqrt{\sum_{i=1}^n \left(\frac{\partial f(\overline{x^{(k)}})}{\partial x_i}\right)^2} \leq \varepsilon.$$

Если условие выполнено, то расчет окончен $\overline{x^*} = \overline{x^{(k)}}$, иначе перейти к пункту 5.

5. Сформировать матрицу Гессе $H(\overline{x^{(k)}})$, используя разностные формулы вычисления вторых и смешанных производных.

6. Проверить положительную определенность матрицы Гессе $H(\overline{x^{(k)}})$. Если матрица положительно определена $H(\overline{x^{(k)}}) > 0$, то перейти к пункту 7, иначе — к пункту 8.

7. Определить координаты точки $\overline{x^{(k+1)}} = \overline{x^{(k)}} - H^{-1}(\overline{x^{(k)}})\nabla f(\overline{x^{(k)}})$ и перейти к пункту 10.

8. Вычислить шаг h_k по формуле

$$h_k = \frac{(\nabla f(\overline{x^{(k)}}), \overline{p^{(k)}})}{(H(\overline{x^{(k)}})\overline{p^{(k)}}), \overline{p^{(k)}})}$$

используя результаты вычислений пункта 3 и разностные формулы вторых и смешанных производных.

9. Определить координаты точки $\overline{x^{(k+1)}} = \overline{x^{(k)}} - h_k \nabla f(\overline{x^{(k)}})$ по методу наискорейшего градиентного спуска.

10. Положить $k = k + 1$ и перейти к пункту 3.

3.1.1. Практический расчет задачи

Найти минимум целевой функции

$$f(\overline{x}) = 2.8 * x_2^2 + 1.9 * x_1 + 2.7 * x_1^2 + 1.6 - 1.9 * x_2$$

методом Ньютона с точностью $\varepsilon = 0.1$.

Решение. В качестве начального приближения возьмем точку $\overline{x^{(0)}} = (x_1^{(0)}, x_2^{(0)})^T = (-0.25, 0.5)^T$.

Найдем градиент функции и матрицу Гессе в произвольной точке $\overline{x^{(k)}} = (x_1^{(k)}, x_2^{(k)})^T$:

$$\nabla f(\overline{x^{(k)}}) = \left(\frac{\partial f(\overline{x^{(k)}})}{\partial x_1}; \frac{\partial f(\overline{x^{(k)}})}{\partial x_2}\right)^T = (5.4 * x_1^{(k)} + 1.9; 5.6 * x_2^{(k)} - 1.9)^T.$$

Итерация $k = 0$. Определим в формуле $\overline{x^{(1)}} = \overline{x^{(0)}} + \overline{p^{(0)}}$ направление спуска $\overline{p^{(0)}}$. Для этого проведем анализ матрицы Гессе в точке $\overline{x^{(0)}}$: $H(\overline{x^{(0)}}) = \begin{bmatrix} 4,441 & 0 \\ 0 & 4,441 \end{bmatrix}$. Вычислим угловые миноры матрицы Гессе:

$$\Delta_1 = 4,441 > 0, \quad \Delta_2 = \begin{vmatrix} 4,441 & 0 \\ 0 & 4,441 \end{vmatrix} = 19,723 > 0.$$

Так как знаки угловых миноров строго положительны, то согласно критерию Сильвестра, матрица Гессе положительно определена.

Критерием Сильвестра и состоит в следующем:

1) для того чтобы квадратичная форма была положительно определённой необходимо и достаточно, чтобы все главные диагональные миноры матрицы этой квадратичной формы были положительны.

2) для того чтобы квадратичная форма была отрицательно определённой, необходимо и достаточно, чтобы знаки главных диагональных миноров матрицы этой квадратичной формы чередовались, начиная со знака «-» для Δ_1 .

Следовательно, направление спуска определяем по формуле Ньютона $\overline{p^{(0)}} = -H^{-1}(\overline{x^{(0)}})\nabla f(\overline{x^{(0)}})$.

Вычислим составляющие вектора $\overline{p^{(0)}}$: $\nabla f(\overline{x^{(0)}}) = (0,55; 0,9)^T$,

$$H^{-1}(\overline{x^{(0)}}) = 0,225 \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}.$$

Вычислим координаты точки $\overline{x^{(1)}}$

$$\begin{aligned} \overline{x^{(1)}} &= \overline{x^{(0)}} - H^{-1}(\overline{x^{(0)}})\nabla f(\overline{x^{(0)}}) = \begin{pmatrix} -0,25 \\ 0,5 \end{pmatrix} - 0,225 \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{pmatrix} 0,55 \\ 0,9 \end{pmatrix} = \\ &= (-0,374 \quad 0,297)^T. \end{aligned}$$

Составим таблицу 1.

Таблица 1.

Точка	Координаты точки		Значение функции
	1	2	
$x^{(0)}$	$x_1^{(0)} = -0,25$	$x_2^{(0)} = 0,5$	$f(\overline{x^{(0)}}) = 1,044$
$x^{(1)}$	$x_1^{(1)} = -0,374$	$x_2^{(1)} = 0,297$	$f(\overline{x^{(1)}}) = 0,95$

Проверим условие окончания процесса поиска для этого вычислим градиент целевой функции $\nabla f(\overline{x^{(1)}})$ в точке $\overline{x^{(1)}}$: $\nabla f(\overline{x^{(1)}}) =$

$(-0,119; -0,235)^T$. Так как норма вектора градиента $\|\nabla f(\bar{x}^{(1)})\| = 0,263 > \varepsilon$, то переходим к следующей итерации.

Итерация $k = 1$. Определим в формуле $\bar{x}^{(2)} = \bar{x}^{(1)} + \bar{p}^{(1)}$ направление спуска $\bar{p}^{(1)}$. Для этого проведем анализ матрицы Гессе в точке $\bar{x}^{(1)}$: $H(\bar{x}^{(1)}) = \begin{bmatrix} 6,661 & 0 \\ 0 & 4,441 \end{bmatrix}$. Вычислим угловые миноры матрицы Гессе:

$$\Delta_1 = 6,661 > 0, \quad \Delta_2 = \begin{vmatrix} 6,661 & 0 \\ 0 & 4,441 \end{vmatrix} = 29,58 > 0.$$

Так как знаки угловых миноров строго положительны, то согласно критерию Сильвестра, матрица Гессе положительна определена. Следовательно, направление спуска определяем по формуле Ньютона

$$\bar{p}^{(1)} = -H^{-1}(\bar{x}^{(1)})\nabla f(\bar{x}^{(1)}).$$

Вычислим составляющие вектора $\bar{p}^{(1)}$: $\nabla f(\bar{x}^{(1)}) = (-0,119; -0,235)^T$,

$$H^{-1}(\bar{x}^{(0)}) = \begin{bmatrix} 0,15 & 0 \\ 0 & 0,225 \end{bmatrix}.$$

Вычислим координаты точки $\bar{x}^{(2)}$

$$\bar{x}^{(2)} = \bar{x}^{(1)} - H^{-1}(\bar{x}^{(1)})\nabla f(\bar{x}^{(1)}) = \begin{pmatrix} -0,374 \\ 0,297 \end{pmatrix} - \begin{bmatrix} 0,15 & 0 \\ 0 & 0,225 \end{bmatrix} \begin{pmatrix} -0,119 \\ -0,235 \end{pmatrix} = (-0,356 \quad 0,35)^T.$$

Составим таблицу 2.

Таблица 2.

Точка	Координаты точки		Значение функции
	1	2	
$\bar{x}^{(0)}$	$x_1^{(0)} = -0,25$	$x_2^{(0)} = 0,5$	$f(\bar{x}^{(0)}) = 1,044$
$\bar{x}^{(1)}$	$x_1^{(1)} = -0,374$	$x_2^{(1)} = 0,297$	$f(\bar{x}^{(1)}) = 0,95$
$\bar{x}^{(2)}$	$x_1^{(2)} = -0,356$	$x_2^{(2)} = 0,35$	$f(\bar{x}^{(2)}) = 0,944$

Проверим условие окончания процесса поиска для этого вычислим градиент целевой функции $\nabla f(\bar{x}^{(2)})$ в точке $\bar{x}^{(2)}$: $\nabla f(\bar{x}^{(2)}) = (-0,022; 0,061)^T$. Так как норма вектора градиента $\|\nabla f(\bar{x}^{(2)})\| = 0,065 < \varepsilon$, то требуемая точность достигнута и точка $\bar{x}^{(2)}$ есть найденное приближение точки минимума $\bar{x}^* = (-0,356; 0,35)^T$, $f(\bar{x}^*) = 0,944$.

3.1.2. Программная реализация

Программный код выполнен на языке Java.

```
import java.util.ArrayList;
```

```
public class Newton extends Differential {
    //Массив векторов
    private static ArrayList<CoordRow> arr = new ArrayList<>();
    //Размерность
    private int dimension = 2;
    //Точность численных вычислений
    private double accuracy = 0.00000001d;
    //Точность искомого значения функции
    private double eps = 0.1;
    //Стартовый вектор
    private double[] basisVector = {-0.25, 0.5};

    private Newton() {
        arr.add(new CoordRow(cloneVector(basisVector),
getFuncValue(basisVector)));
    }

    public static void main(String[] args) {
        Newton newton = new Newton();
        int i = 0;
        while (true) {
            System.out.println("\nИтерация " + i++);
            if (newton.run()) break;
        }
        newton.printTable(newton.getArr());
    }

    public ArrayList<CoordRow> getArr() {
        return arr;
    }

    @Override
```

```

protected double getFuncValue(double[] x) {
    return 2.8d*x[1]*x[1] + 1.9d*x[0] + 2.7d*x[0]*x[0] + 1.6d - 1.9*x[1];
}

private boolean run() {
    printVector(basisVector, "Координаты базисного вектора: ");
    printValue(getFuncValue(basisVector), "Скалярное значение базисного
вектора: ");
    //Считаем значение градиента
    double[] gradientVector = getGradientValue(basisVector, accuracy);
    printVector(gradientVector, "Координаты вектора градиента: ");
    //Считаем значение нормы
    double norma = getNorma(gradientVector);
    printValue(norma, "Значение норма вектора: ");
    //Проверяем условие окончания поиска
    if (norma <= eps) {
        return true;
    } else {
        //Считаем матрицу Гессе
        double[][] gesseMatrix = getGesseMatrix(basisVector, accuracy,
dimension);
        printMatrix(gesseMatrix, "Матрица Гессе: ");
        //Проверяем положительную определенность матрицы Гессе
        if (MatrixOperations.isMatrixPositivelyDefined(gesseMatrix)) {
            //Считаем обратную матрицу Гессе
            double[][] invertGesse = MatrixOperations.inversion(gesseMatrix);
            printMatrix(invertGesse, "Обратная матрица Гессе:");
            double[][] gradientMatrixHorizontal = new double[1][dimension];
            gradientMatrixHorizontal[0] = gradientVector;
            //Перемножаем обратную матрицу на вектор градиента
            double[][] tmp = MatrixOperations.multiplyByMatrix( invertGesse
                , MatrixOperations.transportMatrix(gradientMatrixHorizontal));
            double[] vector = MatrixOperations.transportMatrix(tmp)[0];
            //Считаем новый вектор
            basisVector = countNewVector(basisVector, vector, 1);
        } else {

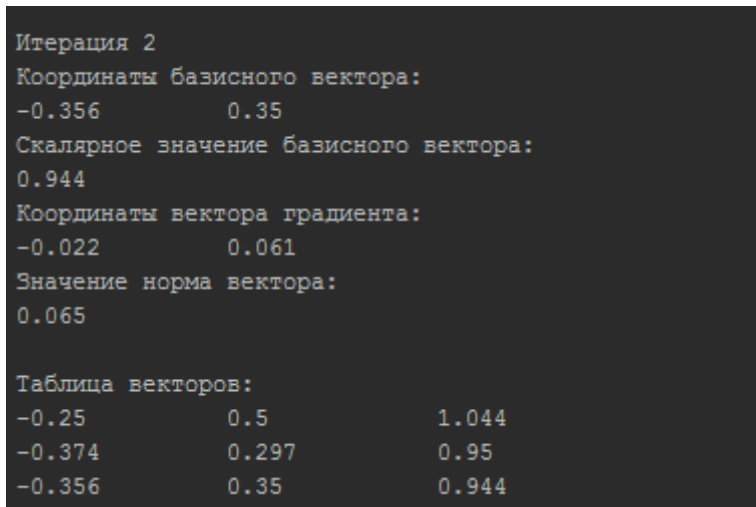
```

```

//Считаем новый шаг
double step = getStep(basisVector, accuracy, dimension);
printValue(step, "Новый шаг: ");
//Считаем новый вектор
basisVector = countNewVector(basisVector, gradientVector, step);
}
printVector(basisVector, "Координаты нового вектора: ");
printValue(getFuncValue(basisVector), "Скалярное значение нового
вектора: ");
arr.add(new CoordRow(cloneVector(basisVector),
getFuncValue(basisVector)));
}
return false;
}
}

```

Результат работы программы (рис. 2.9):



```

Итерация 2
Координаты базисного вектора:
-0.356      0.35
Скалярное значение базисного вектора:
0.944
Координаты вектора градиента:
-0.022      0.061
Значение норма вектора:
0.065

Таблица векторов:
-0.25      0.5      1.044
-0.374     0.297     0.95
-0.356     0.35     0.944

```

Рис. 2.8. Результат работы программы

3.1.3. Контрольные вопросы

1. Идея методов второго порядка.
2. Определение последовательности приближений к точке минимума целевой функции.
3. Раскройте смысл критериев Сильвестра.
4. Алгоритм метода Ньютона.
8. Проверка условия окончания поиска.

3.2. Метод Ньютона -Рафсона.

Метод Ньютона-Рафсона обеспечивает более устойчивую сходимость последовательности приближений, чем рассмотренный выше метод Ньютона. Последовательность приближений строится по модифицированной формуле

$$\overline{x^{(k+1)}} = \overline{x^{(k)}} + h_k \overline{p^{(k)}} ,$$

где k — номер итерации ($k = 0, 1, \dots$),

$\overline{x^{(0)}}$ — начальное приближение,

h_k — величина шага,

$\overline{p^{(k)}} = -H^{-1}(\overline{x^{(k)}}) \nabla f(\overline{x^{(k)}})$ — вектор направления спуска.

Здесь $H^{-1}(\overline{x^{(k)}})$ — обратная матрица для матрицы Гессе.

Направление спуска $\overline{p^{(k)}}$ ведет к убыванию целевой функции только при положительной определенности матрицы Гессе $H(\overline{x^{(k)}}) > 0$. В тех итерациях, в которых матрица Гессе отрицательно определена, последовательность приближений к точке минимума строится по методу наискорейшего градиентного спуска. С этой целью проводится замена вектора направления спуска на анти-градиентное $\overline{p^{(k)}} = -h_k \nabla f(\overline{x^{(k)}})$.

Выбор величины шага h_k осуществляется из решения одномерной задачи оптимизации

$$\varphi(h_k) = f(\overline{x^{(k)}} + h_k \overline{p^{(k)}}) \rightarrow \min. h_k > 0$$

Величина шага h_k может быть найдена в явном виде, аналогично формуле метода наискорейшего градиентного спуска

$$h_k = \frac{\nabla f(\overline{x^{(k)}}), \overline{p^{(k)}}}{H(\overline{x^{(k)}}) \overline{p^{(k)}}, \overline{p^{(k)}}}$$

где $H(\overline{x^{(k)}})$ — матрица Гессе, вычисленная в точке $\overline{x^{(k)}}$.

Рассмотрим алгоритм метода Ньютона—Рафсона:

1. Задать размерность задачи оптимизации n , координаты начальной точки $\overline{x^{(0)}} = (x_1^{(0)}, x_2^{(0)}, \dots, x_n^{(0)})$, точность поиска ε .
2. Положить счетчик числа итераций $k = 0$.
3. Определить направление вектора градиента целевой функции

$$f(\bar{x}) \nabla f(\bar{x}^{(k)}) = \left(\frac{\partial f(\bar{x}^{(k)})}{\partial x_1}, \frac{\partial f(\bar{x}^{(k)})}{\partial x_2}, \dots, \frac{\partial f(\bar{x}^{(k)})}{\partial x_n} \right)$$

в точке $\bar{x}^{(k)} = (x_1^{(k)}, x_2^{(k)}, \dots, x_n^{(k)})$. Для вычисления координат вектора градиента использовать разностную формулу первой производной.

4. Проверить условие окончания поиска

$$\|\nabla f(\bar{x}^{(k)})\| = \sqrt{\sum_{i=1}^n \left(\frac{\partial f(\bar{x}^{(k)})}{\partial x_i} \right)^2} \leq \varepsilon.$$

Если условие выполнено, то расчет окончен $\bar{x}^* = \bar{x}^{(k)}$, иначе перейти к пункту 5.

5. Сформировать матрицу Гессе $H(\bar{x}^{(k)})$, используя разностные формулы вычисления вторых и смешанных производных.

6. Проверить положительную определенность матрицы Гессе $H(\bar{x}^{(k)})$. Если матрица положительно определена $H(\bar{x}^{(k)}) > 0$, то перейти к пункту 7, иначе — положить $H^{-1}(\bar{x}^{(k)}) = E$ и выполнить пункт 7.

7. Вычислить шаг h_k по формуле

$$h_k = \frac{\nabla f(\bar{x}^{(k)}), \bar{p}^{(k)}}{H(\bar{x}^{(k)}) \bar{p}^{(k)}, \bar{p}^{(k)}}$$

используя результаты вычислений пункта 3 и разностные формулы вторых и смешанных производных.

8. Определить координаты точки

$\bar{x}^{(k+1)} = \bar{x}^{(k)} - h_k H^{-1}(\bar{x}^{(k)}) \nabla f(\bar{x}^{(k)})$, положить $k = k + 1$ и перейти к пункту 3.

3.2.1. Практический расчет задачи

Найти минимум целевой функции

$$f(\bar{x}) = 2.8 * x_2^2 + 1.9 * x_1 + 2.7 * x_1^2 + 1.6 - 1.9 * x_2$$

методом Ньютона—Рафсона с точностью $\varepsilon = 0,1$.

Решение. В качестве начального приближения возьмем точку $\bar{x}^{(0)} = (x_1^{(0)}, x_2^{(0)})^T = (-0.5, 0.5)^T$.

Найдем градиент функции и матрицу Гессе в произвольной точке

$$\overline{x^{(k)}} = (x_1^{(k)}, x_2^{(k)})^T :$$

$$\nabla f(\overline{x^{(k)}}) = \left(\frac{\partial f(\overline{x^{(k)}})}{\partial x_1}, \frac{\partial f(\overline{x^{(k)}})}{\partial x_2} \right)^T = (5.4 * x_1^{(k)} + 1.9; 5.6 * x_2^{(k)} - 1.9)^T.$$

Итерация $k = 0$. Определим в формуле $\overline{x^{(1)}} = \overline{x^{(0)}} + h_0 \overline{p^{(0)}}$ направление спуска $\overline{p^{(0)}}$. Для этого проведем анализ матрицы Гессе в точке $\overline{x^{(0)}}$:

$H(\overline{x^{(0)}}) = \begin{bmatrix} 4,441 & 0 \\ 0 & 4,441 \end{bmatrix}$. Вычислим угловые миноры матрицы Гессе:

$$\Delta_1 = 4,441 > 0, \quad \Delta_2 = \begin{vmatrix} 4,441 & 0 \\ 0 & 4,441 \end{vmatrix} = 19,723 > 0.$$

Так как знаки угловых миноров строго положительны, то согласно критерию Сильвестра матрица Гессе положительно определена. Следовательно, направление спуска определяем по формуле Ньютона

$$\overline{p^{(0)}} = -H^{-1}(\overline{x^{(0)}}) \nabla f(\overline{x^{(0)}}).$$

Вычислим составляющие вектора $\overline{p^{(0)}}$: $\nabla f(\overline{x^{(0)}}) = (-0,8; 0,9)^T$,

$$H^{-1}(\overline{x^{(0)}}) = 0,225 \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}.$$

Вычислим координаты точки $\overline{x^{(1)}}$

$$\begin{aligned} \overline{x^{(1)}} &= \overline{x^{(0)}} - h_0 H^{-1}(\overline{x^{(0)}}) \nabla f(\overline{x^{(0)}}) \\ &= \begin{pmatrix} -0,5 \\ 0,5 \end{pmatrix} * h_0 * 0,225 \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} * \begin{pmatrix} -0,8 \\ 0,9 \end{pmatrix} \end{aligned}$$

Численным методом определим значения шага $h_0 = 0,225$.

Найденное значение величины шага h_0 обеспечивает минимум функции $\varphi(h_0)$. По данной величине шага находим координату точки $\overline{x^{(1)}}$:

$$\overline{x^{(1)}} = \begin{pmatrix} -0,5 \\ 0,5 \end{pmatrix} * 0,225 * 0,225 \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} * \begin{pmatrix} -0,8 \\ 0,9 \end{pmatrix} = (-0,459 \quad 0,454)^T$$

Составим таблицу 1.

Таблица 1.

Точка	Координаты точки		Значение функции
	1	2	
$\overline{x^{(0)}}$	$x_1^{(0)} = -0,5$	$x_2^{(0)} = 0,5$	$f(\overline{x^{(0)}}) = 1,075$
$\overline{x^{(1)}}$	$x_1^{(1)} = -0,459$	$x_2^{(1)} = 0,454$	$f(\overline{x^{(1)}}) = 1,012$

Проверим условие окончания поиска. Для этого вычислим вектор градиента $\nabla f(\overline{x^{(1)}}) = (-0,581, 0,644)^T$ в точке $\overline{x^{(1)}} = (x_1^{(1)}, x_2^{(1)})^T = (-0,459 \ 0,454)^T$. Так как норма вектора градиента $\|\nabla f(\overline{x^{(1)}})\| = 0,868 > \varepsilon$, то переходим к следующей итерации.

Итерация $k = 1$. Определим в формуле $\overline{x^{(2)}} = \overline{x^{(1)}} + h_1 \overline{p^{(1)}}$ направление спуска $\overline{p^{(1)}}$. Для этого проведем анализ матрицы Гессе в точке $\overline{x^{(1)}}$:

$H(\overline{x^{(1)}}) = \begin{bmatrix} 6,661 & -2,22 \\ -2,22 & 8,882 \end{bmatrix}$. Вычислим угловые миноры матрицы Гессе:

$$\Delta_1 = 6,661 > 0, \quad \Delta_2 = \begin{vmatrix} 6,661 & -2,22 \\ -2,22 & 8,882 \end{vmatrix} = 54,23 > 0.$$

Так как знаки угловых миноров строго положительны, то согласно критерию Сильвестра матрица Гессе положительно определена. Следовательно, направление спуска определяем по формуле Ньютона

$$\overline{p^{(1)}} = -H^{-1}(\overline{x^{(1)}}) \nabla f(\overline{x^{(1)}}).$$

Вычислим составляющие вектора $\overline{p^{(1)}}$: $\nabla f(\overline{x^{(1)}}) = (-0,581; 0,644)^T$,

$$H^{-1}(\overline{x^{(1)}}) = \begin{bmatrix} 0,164 & 0,041 \\ 0,041 & 1,123 \end{bmatrix}.$$

Вычислим координаты точки $\overline{x^{(2)}}$

$$\begin{aligned} \overline{x^{(2)}} &= \overline{x^{(1)}} - h_1 H^{-1}(\overline{x^{(1)}}) \nabla f(\overline{x^{(1)}}) \\ &= \begin{pmatrix} -0,459 \\ 0,454 \end{pmatrix} * h_1 * \begin{bmatrix} 0,164 & 0,041 \\ 0,041 & 1,123 \end{bmatrix} * \begin{pmatrix} -0,581 \\ 0,644 \end{pmatrix} \end{aligned}$$

Численным методом определим значения шага $h_0 = 0,099$.

Найденное значение величины шага h_0 обеспечивает минимум функции $\varphi(h_0)$. По данной величине шага находим координату точки $\overline{x^{(2)}}$:

$$\overline{x^{(2)}} = \begin{pmatrix} -0,459 \\ 0,454 \end{pmatrix} * 0,099 * \begin{bmatrix} 0,164 & 0,041 \\ 0,041 & 1,123 \end{bmatrix} * \begin{pmatrix} -0,581 \\ 0,644 \end{pmatrix} = (-0,453; 0,449)^T$$

Составим таблицу 2.

Таблица 2.

Точка	Координаты точки		Значение функции
	1	2	
$x^{(0)}$	$x_1^{(0)} = -0,5$	$x_2^{(0)} = 0,5$	$f(\overline{x^{(0)}}) = 1,075$

$x^{(1)}$	$x_1^{(1)} = -0,459$	$x_2^{(1)} = 0,454$	$f(\overline{x^{(1)}}) = 1,012$
$x^{(2)}$	$x_1^{(2)} = -0,453$	$x_2^{(2)} = 0,449$	$f(\overline{x^{(2)}}) = 1,004$

Проверим условие окончания поиска. Для этого вычислим вектор градиента $\nabla f(\overline{x^{(2)}}) = (-0,544, 0,614)^T$ в точке $\overline{x^{(2)}} = (x_1^{(2)}, x_2^{(2)})^T = (-0,453; 0,449)^T$. Так как норма вектора градиента $\|\nabla f(\overline{x^{(2)}})\| = 0,82 > \varepsilon$, то переходим к следующей итерации.

Итерация $k = 2$. Определим в формуле $\overline{x^{(3)}} = \overline{x^{(2)}} + h_2 \overline{p^{(2)}}$ направление спуска $\overline{p^{(2)}}$. Для этого проведем анализ матрицы Гессе в точке $\overline{x^{(2)}}$:

$H(\overline{x^{(2)}}) = \begin{bmatrix} 4,441 & 2,22 \\ 2,22 & 4,441 \end{bmatrix}$. Вычислим угловые миноры матрицы Гессе:

$$\Delta_1 = 4,441 > 0, \quad \Delta_2 = \begin{bmatrix} 4,441 & 2,22 \\ 2,22 & 4,441 \end{bmatrix} = 14,79 > 0.$$

Так как знаки угловых миноров строго положительны, то согласно критерию Сильвестра матрица Гессе положительно определена. Следовательно, направление спуска определяем по формуле Ньютона

$$\overline{p^{(2)}} = -H^{-1}(\overline{x^{(2)}}) \nabla f(\overline{x^{(2)}}).$$

Вычислим составляющие вектора $\overline{p^{(2)}}$: $\nabla f(\overline{x^{(2)}}) = (-0,544; 0,614)^T$,

$$H^{-1}(\overline{x^{(1)}}) = \begin{bmatrix} 0,3 & -0,15 \\ -0,15 & 0,3 \end{bmatrix}.$$

Вычислим координаты точки $\overline{x^{(3)}}$

$$\begin{aligned} \overline{x^{(3)}} &= \overline{x^{(2)}} - h_2 H^{-1}(\overline{x^{(2)}}) \nabla f(\overline{x^{(2)}}) \\ &= \begin{pmatrix} -0,453 \\ 0,449 \end{pmatrix} * h_2 * \begin{bmatrix} 0,3 & -0,15 \\ -0,15 & 0,3 \end{bmatrix} * \begin{pmatrix} -0,544 \\ 0,614 \end{pmatrix} \end{aligned}$$

Численным методом определим значения шага $h_0 = 0,447$.

Найденное значение величины шага h_0 обеспечивает минимум функции $\varphi(h_0)$. По данной величине шага находим координату точки $\overline{x^{(3)}}$:

$$\overline{x^{(3)}} = \begin{pmatrix} -0,453 \\ 0,449 \end{pmatrix} * 0,447 * \begin{bmatrix} 0,3 & -0,15 \\ -0,15 & 0,3 \end{bmatrix} * \begin{pmatrix} -0,544 \\ 0,614 \end{pmatrix} = (-0,338; 0,33)^T$$

Составим таблицу 3.

Таблица 3.

Точка	Координаты точки		Значение функции
	1	2	
$x^{(0)}$	$x_1^{(0)} = -0,5$	$x_2^{(0)} = 0,5$	$f(\overline{x^{(0)}}) = 1,075$
$x^{(1)}$	$x_1^{(1)} = -0,459$	$x_2^{(1)} = 0,454$	$f(\overline{x^{(1)}}) = 1,012$
$x^{(2)}$	$x_1^{(2)} = -0,453$	$x_2^{(2)} = 0,449$	$f(\overline{x^{(2)}}) = 1,004$
$x^{(3)}$	$x_1^{(3)} = -0,338$	$x_2^{(3)} = 0,33$	$f(\overline{x^{(3)}}) = 0,944$

Проверим условие окончания поиска. Для этого вычислим вектор градиента $\nabla f(\overline{x^{(3)}}) = (0,073, -0,052)^T$ в точке $\overline{x^{(3)}} = (x_1^{(3)}, x_2^{(3)})^T = (-0,338; 0,33)^T$. Так как норма вектора градиента $\|\nabla f(\overline{x^{(3)}})\| = 0,09 < \varepsilon$, то требуемая точность достигнута и точка $\overline{x^{(3)}}$ есть найденное приближение точки минимума $\overline{x^*} = (-0,338; 0,33)^T, f(\overline{x^*}) = 0,944$.

3.2.2. Программная реализация

Программный код выполнен на языке Java.

```
import java.util.ArrayList;

public class NewtonRaffson extends Differential {
    //Массив векторов
    private static ArrayList<CoordRow> arr = new ArrayList<>();
    //Размерность
    private int dimension = 2;
    //Точность численных вычислений
    private double accuracy = 0.00000001d;
    //Точность искомого значения функции
    private double eps = 0.1;
    //Стартовый вектор
    private double[] basisVector = {-0.5, 0.5};

    private NewtonRaffson() {
        arr.add(new CoordRow(cloneVector(basisVector),
            getFuncValue(basisVector)));
    }
```

```

public static void main(String[] args) {
    NewtonRaffson newton = new NewtonRaffson();
    int i = 0;
    while (true) {
        System.out.println("\nИтерация " + i++);
        if (newton.run()) break;
    }
    newton.printTable(newton.getArr());
}

public ArrayList<CoordRow> getArr() {
    return arr;
}

@Override
protected double getFuncValue(double[] x) {
    return 2.8d*x[1]*x[1] + 1.9d*x[0] + 2.7d*x[0]*x[0] + 1.6d - 1.9*x[1];
}

private boolean run() {
    printVector(basisVector, "Координаты базисного вектора: ");
    printValue(getFuncValue(basisVector), "Скалярное значение базисного вектора: ");
    //Считаем значение градиента
    double[] gradientVector = getGradientValue(basisVector, accuracy);
    printVector(gradientVector, "Координаты вектора градиента: ");
    //Считаем значение нормы
    double norma = getNorma(gradientVector);
    printValue(norma, "Значение норма вектора: ");
    //Проверяем условие окончания поиска
    if (norma <= eps) {
        return true;
    } else {
        //Считаем значение матрицы Гессе

```

```

        double[][] gesseMatrix = getGesseMatrix(basisVector, accuracy,
dimension);
        printMatrix(gesseMatrix, "Матрица Гессе: ");
        double[][] invertGesse;
        //Проверяем положительную определенность матрицы Гессе
        if (MatrixOperations.isMatrixPositivelyDefined(gesseMatrix)) {
            //Считаем обратную матрицу Гессе
            invertGesse = MatrixOperations.inversion(gesseMatrix);
            printMatrix(invertGesse, "Обратная матрица Гессе:");
        } else {
            //Считаем единичную матрицу размера dimension
            invertGesse = MatrixOperations.getUnitMatrix(dimension);
            printMatrix(invertGesse, "Единичная матрица:");
        }
        double[][] gradientMatrixHorizontal = new double[1][dimension];
        gradientMatrixHorizontal[0] = gradientVector;
        //Перемножаем обрутную или единичную матрицу на вектор
градиента
        double[][] tmp = MatrixOperations.multiplyByMatrix( invertGesse
            , MatrixOperations.transportMatrix(gradientMatrixHorizontal));
        double[] vector = MatrixOperations.transportMatrix(tmp)[0];
        //Считаем шаг
        double step = getStep(basisVector, accuracy, dimension);
        printValue(step, "Новый шаг: ");
        //Считаем новый вектор
        basisVector = countNewVector(basisVector, vector, step);
        printVector(basisVector, "Координаты нового вектора: ");
        printValue(getFuncValue(basisVector), "Скалярное значение нового
вектора: ");
        arr.add(new
                                CoordRow(cloneVector(basisVector),
getFuncValue(basisVector)));
    }
    return false;
}
}

```


Результаты работы программы (рис. 2.9):

```
Итерация 2
Координаты базисного вектора:
-0.453      0.449
Скалярное значение базисного вектора:
1.004
Координаты вектора градиента:
-0.544      0.614
Значение норма вектора:
0.82
Матрица Гессе:
4.441      2.22
2.22      4.441
Обратная матрица Гессе:
0.3      -0.15
-0.15     0.3
Новый шаг:
0.447
Координаты нового вектора:
-0.338      0.33
Скалярное значение нового вектора:
0.944

Итерация 3
Координаты базисного вектора:
-0.338      0.33
Скалярное значение базисного вектора:
0.944
Координаты вектора градиента:
0.073      -0.052
Значение норма вектора:
0.09

Таблица векторов:
-0.5      0.5      1.075
-0.459     0.454     1.012
-0.453     0.449     1.004
-0.338     0.33      0.944
```

Рис. 2.9. Результат работы программы

3.2.3. Контрольные вопросы

1. Отличие метода Ньютона от метода Ньютона-Рафсона.
2. Формула последовательности приближений.
3. Определение вектор направления спуска.
4. Расчет величины шага.
5. Как определена, последовательность приближений к точке минимума при отрицательности матрицы Гессе.
6. Алгоритм метода Ньютона-Рафсона.
7. Проверка условия окончания поиска.
8. Определение координат новой точки точки

Практические задания

Минимизировать целевую функцию методами нулевого, первого и второго порядков с точностью $\varepsilon = 0,0001$. Оценить эффективность методов по числу итераций.

Задания приведены в таблице.

Таблица.

№	Вид функции $f(x_1x_2)$
1	$7x_1^2 + 2x_1x_2 + 5x_2^2 + x_1 - 10x_2$
2	$3x_1^2 - 3x_1x_2 + 4x_2^2 - 2x_1 + x_2$
3	$x_1^2 + 4x_1x_2 + 17x_2^2 + 5x_2$
4	$5x_1^2 - 4x_1x_2 + 5x_2^2 - x_1 - x_2$
5	$4x_1^2 + 4x_1x_2 + 6x_2^2 - 17x_1$
6	$2x_1^2 - 2x_1x_2 + 3x_2^2 + x_1 - 3x_2$
7	$10x_1^2 + 3x_1x_2 + x_2^2 + 10x_2$
8	$x_1^2 - 2x_1x_2 + 6x_2^2 + x_1 - x_2$
9	$x_1^2 + 2x_2^2 + e^{x_1+x_2}$
10	$2x_1^2 + x_1x_2 + x_2^2 + x_1 + x_2$
11	$x_1^2 + 2x_2^2 + e^{x_1^2+x_2^2} - x_1 + 2x_2$
12	$x_1^3 + x_2^2 - 3x_1 - 2x_2 - 2$
13	$x_1^2 + e^{x_1^2+x_2^2} + 4x_1 + 3x_2$
14	$x_1^2 + x_1x_2 + 2x_2^2 - 7x_1 - 7x_2$
15	$x_1^2 - x_1x_2 + x_2^2 + 2x_1 + 3x_2$
16	$10x_1^2 + 3x_1x_2 + x_2^2 + 10x_2$
17	$7x_1^2 + 2x_1x_2 + 5x_2^2 + x_1 + 10x_2$
18	$4(x_1 - 5)^2 + (x_2 - 6)^2$
20	$7x_1^2 + 2x_1x_2 + 5x_2^2 + x_1 - 10x_2$
21	$3x_1^2 - 3x_1x_2 + 4x_2^2 - 2x_1 + x_2$
22	$x_1^2 + 4x_1x_2 + 17x_2^2 + 5x_2$
23	$5x_1^2 - 4x_1x_2 + 5x_2^2 - x_1 - x_2$
24	$4x_1^2 + 4x_1x_2 + 6x_2^2 - 17x_1$
25	$2x_1^2 - 2x_1x_2 + 3x_2^2 + x_1 - 3x_2$
26	$10x_1^2 + 3x_1x_2 + x_2^2 + 10x_2$
27	$x_1^2 - 2x_1x_2 + 6x_2^2 + x_1 - x_2$
28	$x_1^2 + x_1x_2 + x_2^2 + x_1 + x_2$

Список литературы

1. Гончаров, В.А. Методы оптимизации: Учебное пособие для ВУЗов / В.А.Гончаров. - Люберцы: Юрайт, 2016. - 191 с.
2. Горелик, В.А. Исследование операций и методы оптимизации: Учебник / В.А. Горелик. - М.: Academia, 2018. - 384 с.
3. Келлер, И.Э. Методы оптимизации в примерах и задачах: Учебное пособие / И.Э. Келлер. - СПб.: Лань, 2015. - 512 с.
4. Ширяев, В.И. Исследование операций и численные методы оптимизации: Учебное пособие / В.И. Ширяев. - М.: Ленанд, 2015. - 216 с.
5. Зайцев, М.Г. Методы оптимизации управления и принятия решений: примеры, задачи, кейсы / М.Г. Зайцев, С.Е. Варюхин. - М.: Дело АНХ, 2015. - 640 с.
6. Кочегурова, Е.А. Теория и методы оптимизации.: Учебное пособие для академического бакалавриата / Е.А. Кочегурова. - Люберцы: Юрайт, 2016. - 133 с.