

МЕТОДЫ ОПТИМИЗАЦИИ

Распределение

Лекция 1	Яровикова А.
Лекция 2	Лозовска К.
Лекция 3	Емельяненко Д.
Лекция 4	Яровикова А.
Лекция 5	Емельяненко Д., Герман В.
Лекция 6	Яровикова А.
Лекция 7	Емельяненко Д., Герман В.
Лекция 8	Яровикова А.
Лекция 9	Емельяненко Д., Герман В.
Лекция 10	Яровикова А.
Лекция 11	Емельяненко Д., Герман В.
Оформление графиков в LaTeX и дизайн лекций	Лозовска К.

ОГЛАВЛЕНИЕ

Лекция 1. Поиск минимума функции одного переменного: метод перебора, метод деления пополам, метод золотого сечения, метод Фибоначчи	5
Постановка оптимизационной задачи в \mathbb{R}^n	5
1.1. Обзор экстремальных задач.	5
1.2. Разрешимость задачи оптимизации (условия, гарантирующие существование \min или \max функции $f(x)$).	10
Численные методы минимизации функции одной переменной.	14
Примеры реализации методов на языке Julia.	22
Лекция 2.....	24
Необходимое условие существования локального экстремума.....	24
Достаточные условия существования локального экстремума.	25
Общая схема отыскания экстремума.....	27
Выпуклые функции.....	28
Оптимизация функций многих переменных.	29
Общая схема отыскания безусловного экстремума функции многих переменных.....	34
Условный экстремум функции нескольких переменных.....	35
Необходимое условие существования локального экстремума.....	36
Условный экстремум функции двух переменных.	38
Лекция 3. Численные методы минимизации функции нескольких переменных (безусловная оптимизация).....	40
Описание метода спуска.....	40
Задачи методов спуска.	40
Графическая интерпретация.....	41
Геометрическая интерпретация.	42
Метод градиентного спуска.....	43
Метод наискорейшего спуска.....	44
Метод сопряженных градиентов.....	46
Метод Ньютона.....	48

Метод Свенна.....	50
Метод Гаусса – Зейделя (модификация метода по координатного спуска).....	51
Лекция 4.....	54
4.1. Три модификации метода Ньютона.....	54
Простейшие поисковые методы.....	56
4.2. Метод обратного переменного шага.	56
4.3. Метод квадратичной аппроксимации.....	56
4.4. Метод Пауэлла.....	57
4.5. Метод Хука-Дживса.....	57
4.6. Симплексные алгоритмы.	58
4.7. Метод Нельдера-Мида (деформируемых многогранников).	59
Примеры реализации методов на языке Julia.	62
Лекция 5. Усовершенствование метода спуска.....	68
Решение методом Флэтчера и Ривса.....	68
Решение методом наискорейшего спуска.	70
Многокритериальный поиск.	74
Методы второго порядка. Метод Ньютона. Квазиньютоновские методы.....	75
Методы переменной метрики или квазиньютоновские методы.	78
Лекция 6. Метод Дэвидона-Флэтчера-Пауэла (Д.Ф.П.)	81
Метод BFGS (Бройдена, Флэтчера, Гольдфарба, Шанно).	82
Примеры реализации методов на языке Julia.	87
Лекция 7. Эффективность численных методов оптимизации.....	89
Алгоритм Пирсона.	89
Алгоритма Ньютона – Рафсона.....	89
Алгоритмы с аппроксимацией матрицы Гессе.....	89
Алгоритм Гольштайна и Прайса.	90
Эффективность численных методов.	92
Лекция 8. Метод многомерной условной оптимизации.....	97
Метод штрафных функций.....	97
Метод барьерных функций.....	99
Линейное программирование.	101

Графический метод решения задач линейного программирования	104
Лекция 9. Алгоритмы глобальной оптимизации.	110
Проблемы глобального поиска минимума целевой функции.	110
Функция Швефеля.....	111
Алгоритм оптимизации роем частиц.....	116
Модификация LBEST.....	117
Модификации метода роя частиц.	119
Генетические алгоритмы.	120
Лекция 10. Генетические алгоритмы – продолжение.....	127
Схема турнирного отбора.	130
Виды и модификации кроссинговера.	134
Лекция 11. Мутации.	139
Мутация для вещественных особей.	140
Двоичная мутация.	140
Отбор особей.	141
Метод Больцмана.....	143
Разнообразие генетических алгоритмов.	144
Канонические ГА.	144
Генитор.....	145
Метод непрерывного равновесия.	146
Гибридный алгоритм.	146
Генетический алгоритм с нефиксированным размером популяции.	147

Лекция 1. Поиск минимума функции одного переменного: метод перебора, метод деления пополам, метод золотого сечения, метод Фибоначчи

Постановка оптимизационной задачи в \mathbb{R}^n

1.1. Обзор экстремальных задач.

Задача: поиск наибольшей и наименьшей величины целевой функции $f(x)$: $\mathbb{R}^n \rightarrow \mathbb{R}$, где $x \in \mathbb{R}^n, f(x) \in \mathbb{R}$.

Определение.

Экстремальная задача или оптимизационная задача – это тройка вида: $f, X, extr$, где

- $f(x): \mathbb{R}^n \rightarrow \mathbb{R}$ – целевая функция;
- $X \in \mathbb{R}^n$ – множество допустимых решений (допустимое множество) для функции $f(x)$;
- $extr \in \{min, max\}$ – критерий оптимизации.

Математическая постановка задачи оптимизации: $f(x) \rightarrow_{x \in X} extr$ (на множестве X целевая функция $f(x)$ достигает экстремального значения).

Формулировка экстремальной задачи.

Необходимо найти $x_0 \in X$ (если x_0 существует), доставляющее экстремальное (минимальное или максимальное) значение целевой функции $f(x)$ на множестве X , при этом для x_0 должно выполняться одно из условий:

$$- \text{либо } f(x_0) \leq f(x), \forall x \in X; \quad (1.1)$$

$$- \text{либо } f(x_0) \geq f(x), \forall x \in X; \quad (1.2)$$

Краткая формулировка экстремальной задачи.

$$\exists x_0: \forall x \in X: f(x_0) \leq f(x) \text{ или } f(x_0) \geq f(x) \quad (1.3)$$

Если точка x_0 на множестве X не существует, то требуется построить последовательность: $\{x_k\}, k = 1, 2, \dots$ где $x_k \in X$, такую что выполняется одно из соотношений:

$$-\lim_{n \rightarrow \infty} f(x_k) = \inf_{x \in X} f(x) \quad (1.4)$$

$$-\lim_{n \rightarrow \infty} f(x_k) = \sup_{x \in X} f(x) \quad (1.5)$$

Определение 1.1.1.

- 1) Точка $x_0 \in X$, удовлетворяющая условию (1.1): $f(x_0) \leq f(x), \forall x \in X$, называется **точкой глобального минимума функции $f(x)$** на множестве X .
- 2) Точка $x_0 \in X$, удовлетворяющая условию (1.2): $f(x_0) \geq f(x), \forall x \in X$, называется **точкой глобального максимума функции $f(x)$** на множестве X .

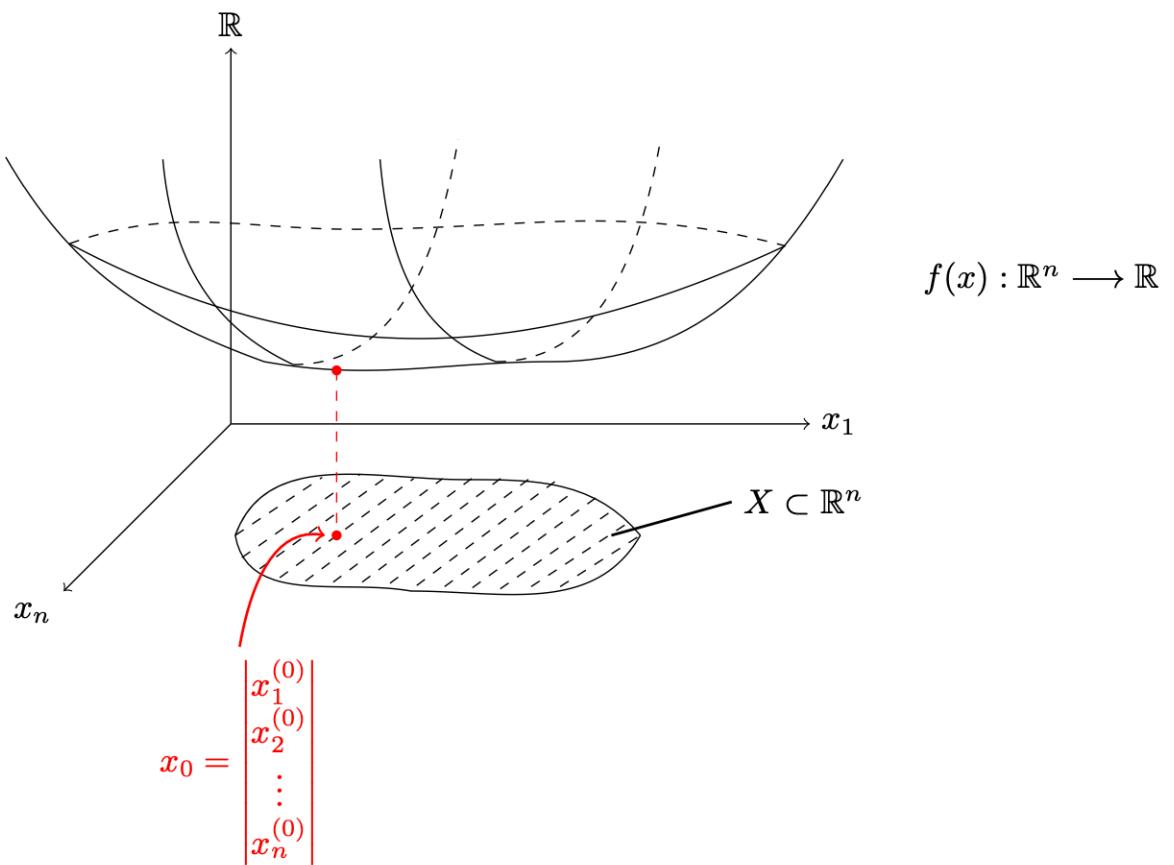


Рисунок 1. Точка глобального минимума

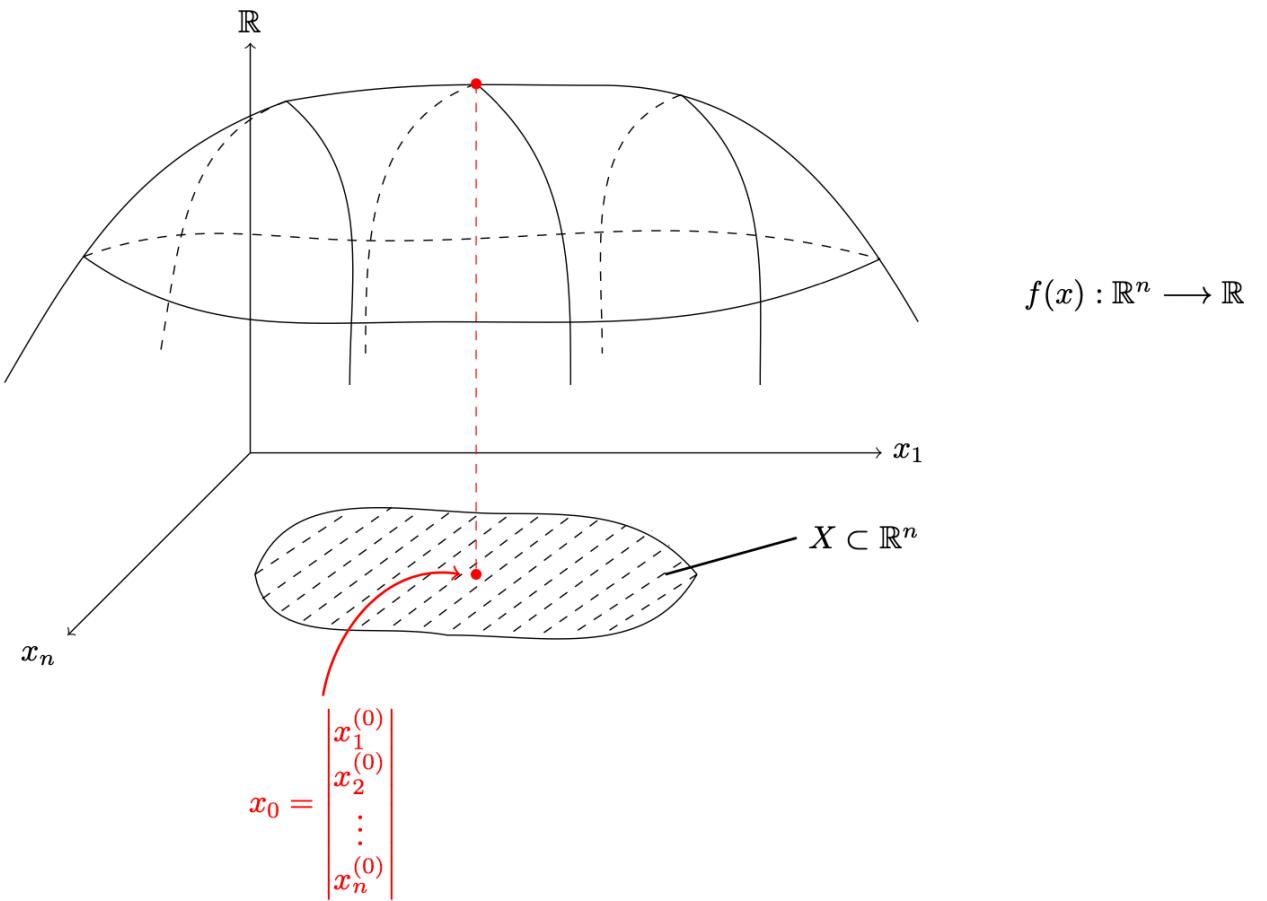


Рисунок 2 - Точка глобального максимума

Определение 1.2.1.

Функция $f(x)$ называется **ограниченной сверху на множестве X** , если $\exists M: f(x) \leq M, \forall x \in X$.

Определение 1.3.1.

Число $m_0 = \inf_{x \in X} f(x)$ называется **нижней гранью функции $f(x)$ на множестве X** :

- если $m_0 \leq f(x), \forall x \in X$;
- для $\forall \varepsilon > 0: \exists x_\varepsilon \in X: f(x_\varepsilon) \leq m_0 + \varepsilon$.

Если $f(x)$ не ограничена снизу на множестве X , то полагают, что $m_0 = \inf_{x \in X} f(x) = -\infty$.

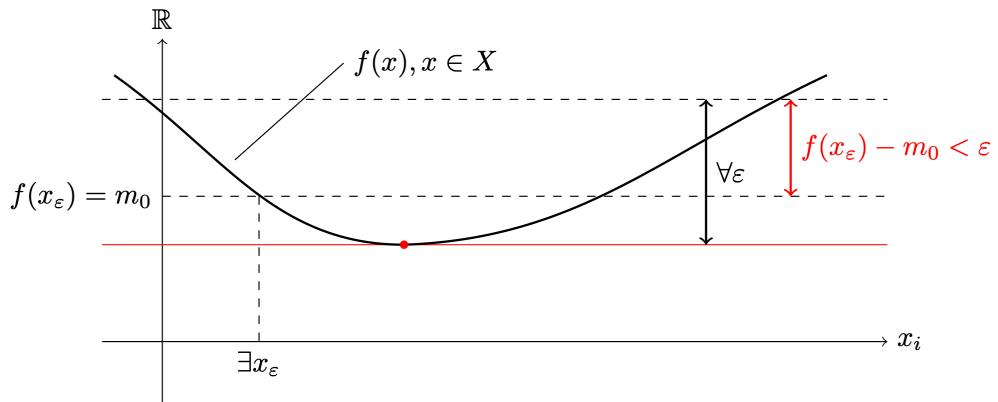


Рисунок 3. Нижняя грань функции

Определение 1.3.2.

Число $M_0 = \sup_{x \in X} f(x)$ называется **верхней гранью функции $f(x)$ на множестве X** :

- если $f(x) \leq M_0, \forall x \in X$;
- для $\forall \varepsilon > 0: \exists x_\varepsilon \in X: f(x_\varepsilon) > M_0 - \varepsilon$.

Если $f(x)$ не ограничена сверху на множестве X , то полагают, что $M_0 = \sup_{x \in X} f(x) = +\infty$.

Пример 1.1.

Рассмотрим функцию $f(x) = \frac{1}{x}$, $X = [1; +\infty)$, задача: показать, что множество точек минимума функции $f(x)$ на X пусто и $m_0 = \inf_{x \in X} f(x) = 0$, а минимум функции $f(x)$ на X существует и равен 1.

▷ Доказательство пустоты множества точек минимума проведем от противного. Пусть множество точек минимума функции $f(x)$ на X не пусто: $\exists x_0: f(x_0) \leq f(x)$ для $\forall x \in X$ (формула 1.1). Проверим для наиб функции $f(x) = \frac{1}{x}$ на $[1; +\infty)$, получим если x_0

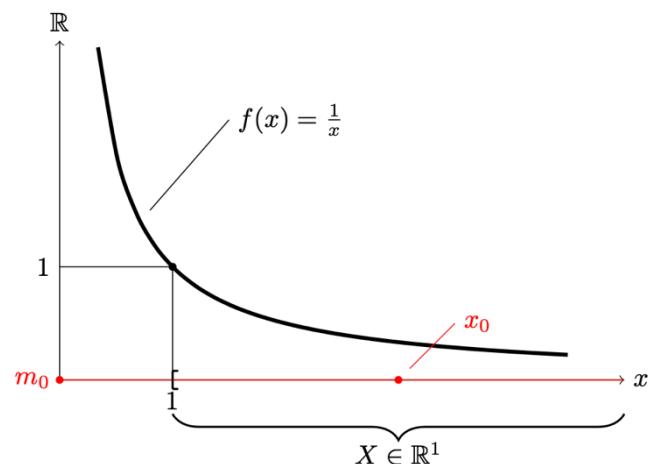


Рисунок 4. Пустота множества точек минимума.

точка минимума, то: $\forall x \in X x > x_0: f(x_0) = \frac{1}{x_0} \leq f(x) = \frac{1}{x} \Rightarrow x \leq x_0$, что и требовалось доказать.

Покажем, что $m_0 = \inf_{x \in X} f(x) = 0$.

Запишем определение нижней грани:

$m_0 = \inf_{x \in X} f(x)$, если:

- 1) $m_0 \leq f(x), \forall x \in X$;
- 2) $\forall \varepsilon > 0: \exists x_\varepsilon \in X: f(x_\varepsilon) - m_0 \leq \varepsilon$.

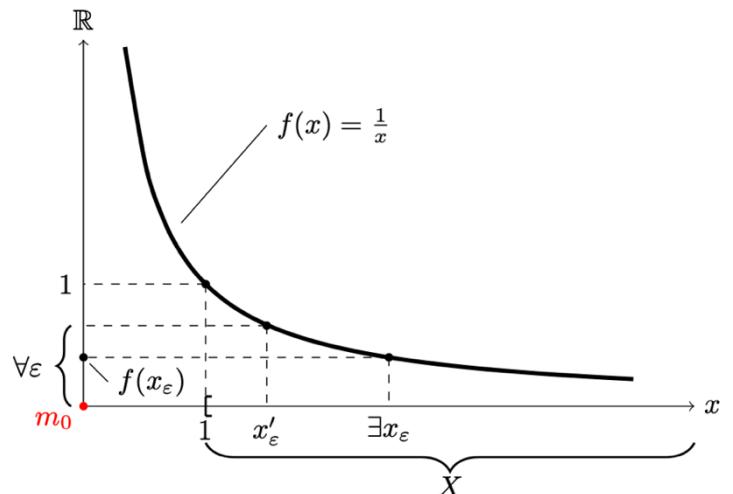


Рисунок 5. $m_0 = 0$.

Для $\forall x \in X = [1; +\infty): f(x) = \frac{1}{x} > 0$, т. к. $x > 0$, т.к. $x \in X = [1; +\infty)$, $\forall \varepsilon > 0: \varepsilon = f(x'_\varepsilon) = \frac{1}{x'_\varepsilon} > 0 \Rightarrow x'_\varepsilon = \frac{1}{\varepsilon} > 0$.

Возьмем $\exists x_\varepsilon > x'_\varepsilon$, тогда: $f(x_\varepsilon) = \frac{1}{x_\varepsilon} < f(x'_\varepsilon) = \frac{1}{x'_\varepsilon} = \varepsilon \Rightarrow f(x_\varepsilon) < \varepsilon \Leftrightarrow f(x_\varepsilon) - 0 < \varepsilon \Leftrightarrow f(x_\varepsilon) - m_0 < \varepsilon$, где $m_0 = 0$. \triangleleft

Пример 1.2.

Пусть целевая функция $f(x) = e^{-|x|}$, $X \subset \mathbb{R}^1$, задача: показать, что множество точек минимума функции $f(x)$ на X пусто и $m_0 = \inf_{x \in X} f(x) = 0$, найти

$$M_0 = \sup_{x \in X} f(x).$$

▷ Доказательство от противного. Пусть множество точек минимума не пусто (*). Пусть $\exists x_0: \forall x \in X: f(x_0) \leq f(x)$, тогда выбираем $\forall x: x > x_0$, при этом $x \in X$, тогда

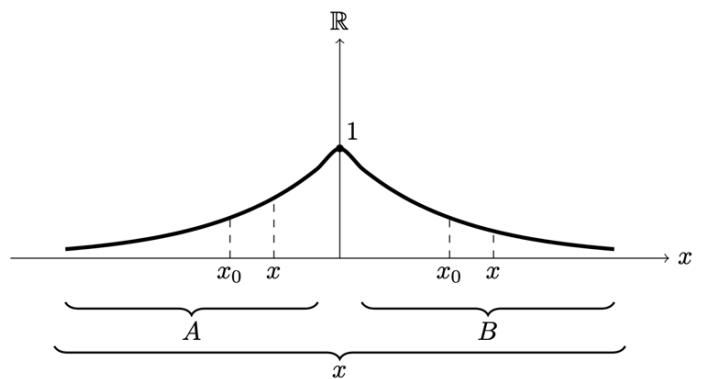


Рисунок 6. Случай A и B.

учитывая, что есть $f(x) = e^{-|x|}$, то при $x > x_0$ $f(x_0) = e^{-1|x_0|}$, $f(x) = e^{-1|x|}$, то при $x > x_0$:

- Случай А: $f(x) > f(x_0)$
- Случай В: $f(x) < f(x_0)$

Видим, что для случая В возникает противоречие с (*).

Покажем, что $m_0 = \inf_{x \in X} f(x) = 0$. Очевидно, что $\forall x \in X: f(x) = e^{-1|x|} > 0$

$\forall \varepsilon > 0 \exists x_\varepsilon > x'_\varepsilon: f(x_\varepsilon) < \varepsilon \Leftrightarrow f(x_\varepsilon) - m_0 < \varepsilon, m_0 = 0$.

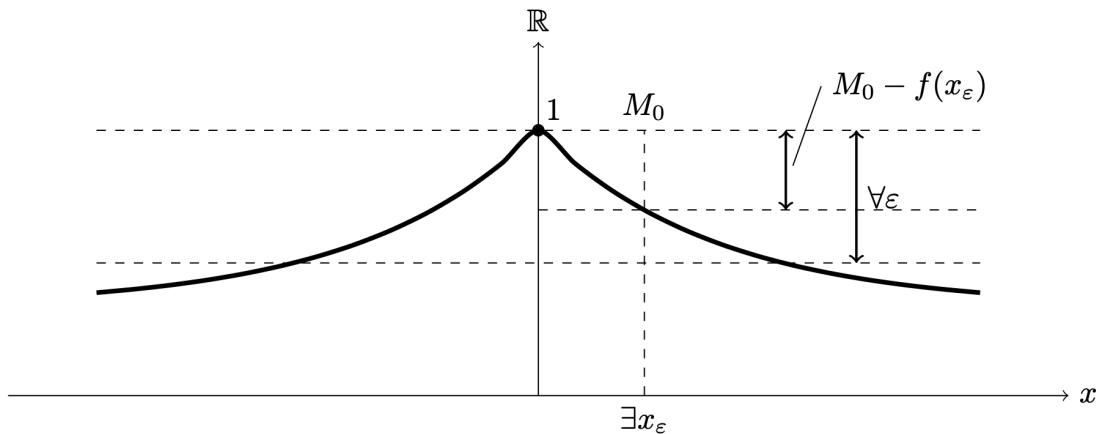


Рисунок 7. $m_0 = \inf_{x \in X} f(x) = 0$

- $f(x) \leq M_0 = 1, \forall x \in X$
- Для $\forall \varepsilon > 0 \exists x_\varepsilon \in X: M_0 - f(x_\varepsilon) < \varepsilon$

$$1 - f(x_\varepsilon) < \varepsilon$$

$$f(x_\varepsilon) > 1 - \varepsilon . \triangleleft$$

1.2. Разрешимость задачи оптимизации (условия, гарантирующие существование \min или \max функции $f(x)$).

Определение 0.1.

Числовое множество E называется **ограниченным сверху или снизу**, если $\exists M_0 \in R: \forall x \in X, x \leq M_0$ либо $x \geq M_0$.

Определение 0.2.

Числовое множество, ограниченное и сверху, и снизу, называется **ограниченным**.

Определение 0.3.

Множество F называется **замкнутым**, если оно содержит все свои предельные точки.

Пример 0.1.

Отрезок $[a, b] \in R$ числовой прямой является замкнутым множеством, а полуинтервал $[a, b)$ не замкнут, т. к. его предельная точка b не принадлежит этому множеству.

Вопрос: $(x - x_0)^2 + (y - y_0)^2 + (z - z_0)^2 < R^2$?

Ответ: не замкнут, т. к. граница не принадлежит этому множеству.

Замечание.

- отрезок – замкнутое множество, но не открытое;
- интервал – открытое множество, но не замкнутое;
- полуинтервал – не открытое, не замкнутое.

Теорема 1.1. (Вейерштрасса)

Если множество $X \in R^n$ не пусто и компактно (ограничено и замкнуто), а функция $f(x)$ непрерывна на X , то множество точек глобального минимума или максимума функции $f(x)$ на нем не пусто и компактно.

Замечание.

В условии теоремы Вейерштрасса любая минимизирующая последовательность $\{x_k\}$ сходится к множеству точек глобального минимума.

Теорема 1.2.

Пусть множество $X \subset R^n$ не пусто и незамкнуто, а функция $f(x)$ непрерывна на нем. Пусть выполнены хотя бы одно из следующих условий:

- 1) $\exists X_* \subset X: X_* = \{x \in X: f(x) \leq f(x_*)\}$ – ограничено (см. рисунок 8);

- 2) Для любой последовательности $\{x_k\}, k = 1, 2, \dots, x_k \in X$, $\lim_{k \rightarrow \infty} \|x_k\| = +\infty$,
 если такая последовательность найдется, то $\lim_{k \rightarrow \infty} f(x_k) = +\infty$.

Тогда множество точек глобального минимума функции $f(x)$ на множестве X не пусто и компактно.

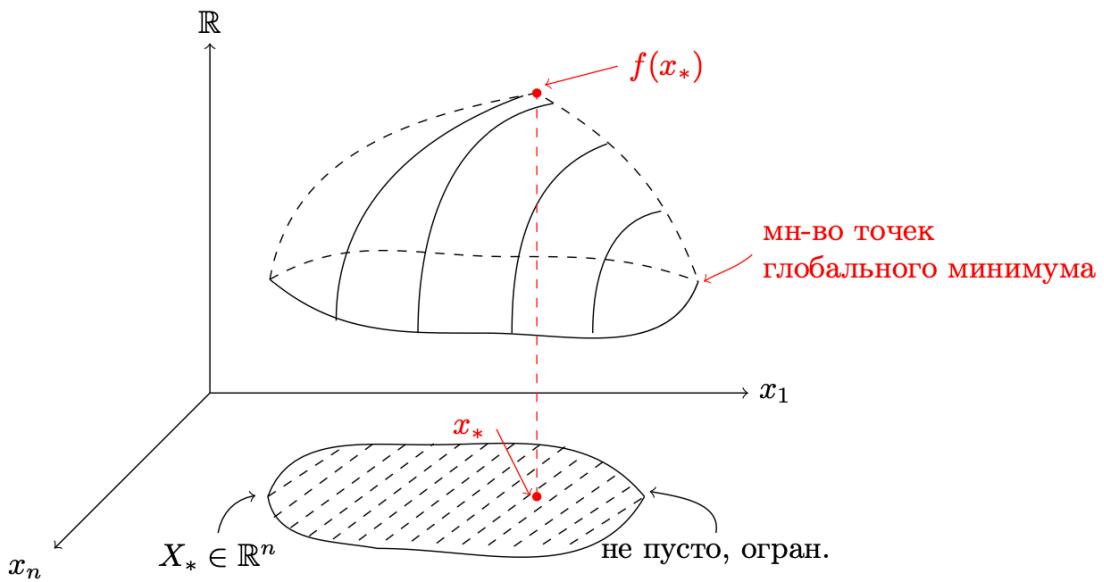


Рисунок 8. Множество точек глобального минимума.

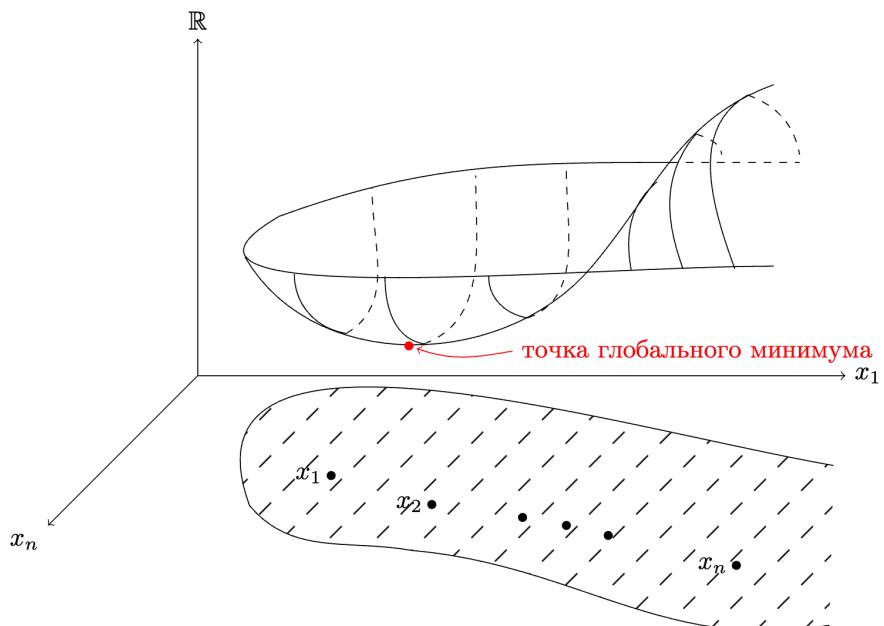


Рисунок 9. Точка глобального минимума.

Определение 1.4.

Функция $f(x) : X \in R^n \rightarrow R$ называется **полунепрерывной снизу в точке $x_0 \in X$** , если для $\forall \varepsilon > 0 \exists \delta > 0 : \forall x \in X : \|x - x_0\| < \delta \Rightarrow f(x) \geq f(x_0) - \varepsilon$ или $f(x_0) - f(x) \leq \varepsilon$.

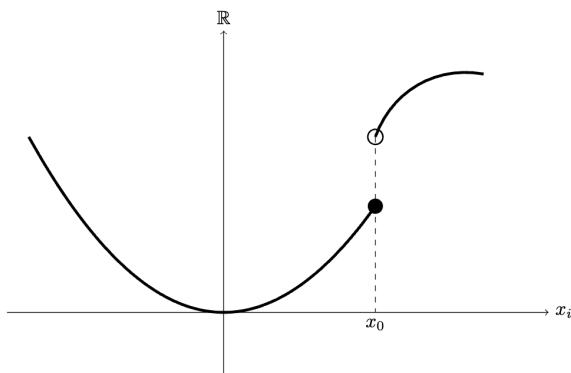


Рисунок 10. Полунепрерывность функции снизу (a).

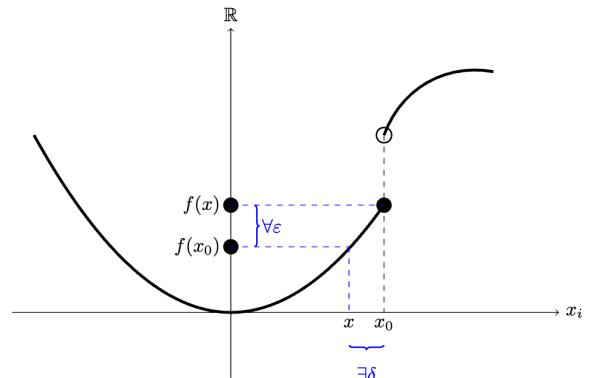


Рисунок 11. Полунепрерывность функции (b).

Функция $f(x)$ **полунепрерывная сверху** (см. рис 4.1) **в точке $x_0 \in X$** , если $\forall \varepsilon > 0 \exists \delta > 0 : \forall x \in X : \|x - x_0\| < \delta \Rightarrow f(x) - f(x_0) \leq \varepsilon$.

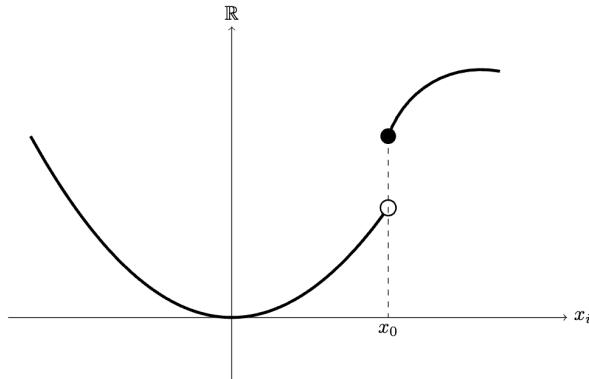


Рисунок 12. Полунепрерывность функции сверху.

Функция $f(x)$ **непрерывна в точке x_0** тогда и только тогда, когда в этой точке функция полунепрерывна и снизу, и сверху.

Теорема 1.3. (Обобщенная теорема Вейерштрасса)

Полунепрерывная снизу (сверху) функция $f(x) : R^n \rightarrow R$ достигает глобального минимума (максимума) на всяком компакте $X \subset R^n$.

Теорема 1.4.

Полунепрерывная снизу (сверху) функция $f(x): R^n \rightarrow R$ достигает глобального минимума (максимума) на всём пространстве R^n , если $\exists C: \{x \in R^n: f(x) \leq C\}$ ($\exists C: \{x \in R^n: f(x) \geq C\}$) не пусто и ограничено.

Определение 1.5.

Функция $f(x)$ имеет в $x_* \in X$ локальный минимум (максимум), если $f(x_* + \Delta x) > f(x_*)$ ($f(x_* + \Delta x) < f(x_*)$) для $\forall x_* + \Delta x$, достаточно малых Δx .

Свойства полунепрерывных функций:

1. Если $f(x)$ - полунепрерывная сверху функция, то $-1 * f(x)$ полунепрерывная снизу.
2. Пусть $f(x)$ и $g(x)$ - две полунепрерывные функции снизу (сверху), тогда $f(x) + g(x)$ тоже полунепрерывная функция снизу (сверху).
3. Если $f(x)$ - полунепрерывная функция снизу на заданном множестве X , то для $\forall c = const$ множество $X_* = \{x \in X, f(x) \leq c\}$ замкнуто, если оно не пусто.
4. Предел монотонно возрастающей (убывающей) последовательности полунепрерывных снизу (сверху) в точке x_0 функций есть полунепрерывная функция снизу (сверху) в x_0 . При этом пусть задана последовательность функций $\{f_k\}, k = 1, 2, \dots, f_k: X \rightarrow R, (X \subset R^n)$, при этом $f_{k+1}(x) \geq f_k(x)$ ($f_{k+1}(x) \leq f_k(x)$) для $\forall x \in X$. Тогда если $\exists \lim_{k \rightarrow \infty} f_k(x) = f(x) \forall x \in X$, то $f(x)$ – полунепрерывна снизу (сверху).

Численные методы минимизации функции одной переменной.

Определение.

Функция $f(x)$ называется **нимодальной** на $[a, b]$, если $\exists! x_*$ и слева от x_* $f(x)$ строго убывает, справа x_* $f(x)$ строго возрастает:

- $f(x_1) > f(x_2), \forall x_1 < x_2 < x_*$;

- $f(x_1) < f(x_2), \forall x_* < x_1 < x_2$.

Пример групп унимодальных функций:

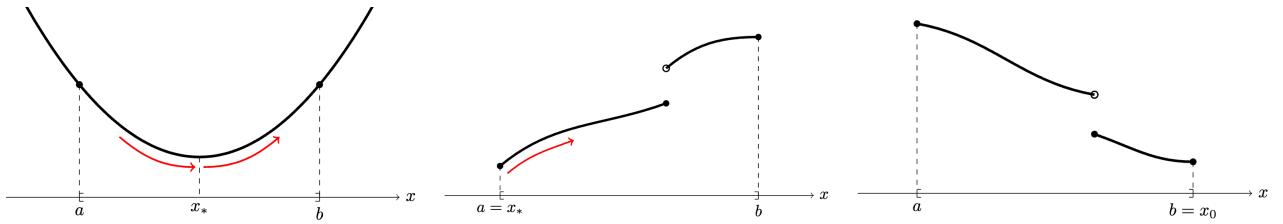


Рисунок 13. Унимодальные функции.

Теорема.

Если функция $f(x)$ дифференцируема на $[a, b]$ и $f'(x)$ не убывает на $[a, b]$, то $f(x)$ унимодальна на $[a, b]$.

Теорема.

Если функция $f(x)$ дважды дифференцируема на $[a, b]$ и $f''(x) \geq 0$ при $x \in [a, b]$, то $f(x)$ унимодальна на $[a, b]$.

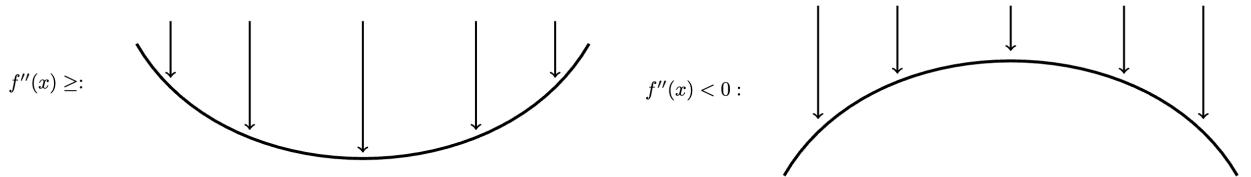


Рисунок 14. Критерий унимодальности функции.

Метод перебора.

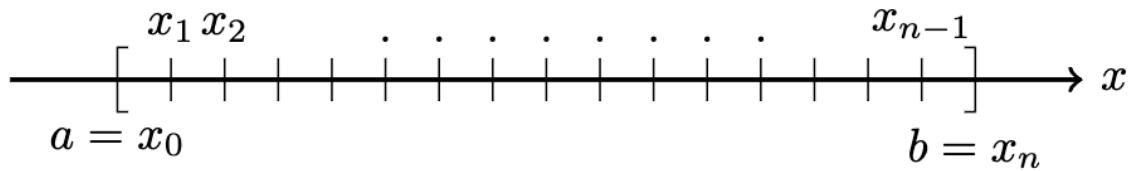


Рисунок 15. Метод перебора.

$$x_i = a + i * \frac{b-a}{n}, i = 0, 1, \dots, n,$$

где $n \geq \frac{b-a}{\varepsilon}$, далее для каждого x_i вычисляем $f(x_i)$ и находим $\tilde{x} = \min_i f(x_i)$,

т.е. $\tilde{x} \approx x_*$ с точностью $\varepsilon_n = \frac{b-a}{n}$, при этом итеративность: $(\tilde{x} - x_*) \leq \varepsilon$.

Метод деления пополам (дихотомия).

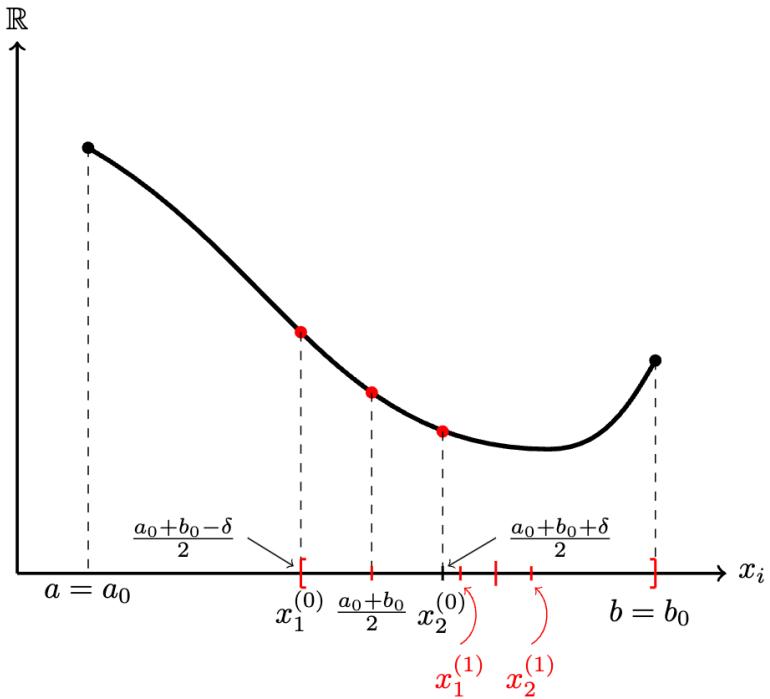


Рисунок 16. Метод деления пополам.

1) $\delta \in (0, 2\varepsilon), a_0 = a, b_0 = b$

a) $x_1^{(0)} = \frac{a_0 + b_0 - \delta}{2} \quad x_2^{(0)} = \frac{a_0 + b_0 + \delta}{2}$

b) $f(x_1^{(0)})$ и $f(x_2^{(0)})$

2) Выбор нового отрезка $[a_1, b_1]$:

– если $f(x_1^{(0)}) \leq f(x_2^{(0)})$, то $a_1 = a_0, b_1 = x_2^{(0)}$;

– если $f(x_1^{(0)}) > f(x_2^{(0)})$, то $a_1 = x_1^{(0)}, b_1 = b_0$.

Далее для $[a_1, b_1]$: $x_1^{(1)} = \frac{a_1 + b_1 - \delta}{2} \quad x_2^{(1)} = \frac{a_1 + b_1 + \delta}{2}$, вычисляем

$f(x_1^{(1)})$ и $f(x_2^{(1)})$ и повторяем алгоритм (см. выше).

Формулы для i -го шага:

для отрезка $[a_i, b_i]$: $x_1^{(i-1)} = \frac{a_{i-1} + b_{i-1} - \delta}{2}$ $x_2^{(i-1)} = \frac{a_{i-1} + b_{i-1} + \delta}{2}$, вычислим $f(x_1^{(i-1)})$ и $f(x_2^{(i-1)})$.

- если $f(x_1^{(i-1)}) \leq f(x_2^{(i-1)})$, то $a_i = a_{i-1}, b_i = x_2^{(i-1)}$;
- если $f(x_1^{(i-1)}) > f(x_2^{(i-1)})$, то $a_i = x_1^{(i-1)}, b_i = b_{i-1}$.

Условие останова: $|b_i - a_i| \leq \varepsilon = \varepsilon_n$ (*)

Пусть для достижения условия (*) потребуется n итераций. Так как условие останова: $|b_i - a_i| \leq \varepsilon, i = 1, 2, \dots, n$ или $|b_n - a_n| \leq \varepsilon$, тогда $\tilde{x} \in [a_n, b_n]$: $\tilde{x} = \frac{a_n + b_n}{2}$, пусть $x_* = \tilde{x} + \delta$, тогда $\varepsilon_n = \frac{b_n - a_n}{2} = \frac{b - a - \delta}{2^{n+1}} + \frac{\delta}{2}$.

Степень близости значения функции в x_* оценивается так:

$$\delta_f = |f_* - \tilde{f}| \equiv |f(x_*) - f(\tilde{x})| \leq L * |x_* - \tilde{x}| \leq L \frac{b_n - a_n}{2} \leq L \frac{\varepsilon}{2}, \text{ где}$$

$L = \max_{x \in [a_n, b_n]} |\rho(x)|$, $\rho(x) = tg(\alpha)$ – коэффициент наклона касательной к графику функции $f(x)$ в точке x .

Метод золотого сечения.

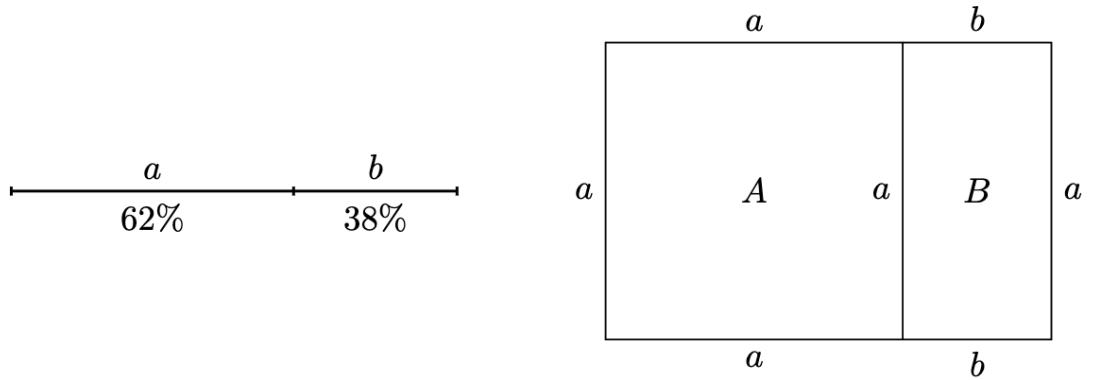


Рисунок 17. Метод золотого сечения.

$$\frac{a}{b} = \frac{a+b}{a} = 1,618 \dots$$

$$\tau = \frac{b}{a} = \frac{-1+\sqrt{5}}{2} \approx 0,618$$

Применение метода золотого сечения для нашей задачи:

Идея:



Рисунок 18. Метод золотого сечения - идея.

$$\frac{b - x_1^{(0)}}{x_1^{(0)} - a} = \frac{b - a}{b - x_1^{(0)}}$$

$$\frac{x_2^{(0)} - a}{b - x_2^{(0)}} = \frac{b - a}{x_2^{(0)} - a}$$

Шаг 0:

Положим $a_0 = a, b_0 = b$, найдем $x_1^{(0)}$ и $x_2^{(0)}$ из золотого сечения $[a_0, b_0]$:

$$\begin{array}{ccccccc} \hline & [& & &] & \rightarrow x \\ a = a_0 & | & x_1^{(0)} & | & x_2^{(0)} & | & b = b_0 \\ \hline \end{array} \quad \begin{aligned} x_1^{(0)} &= a_0 + (1 - \tau)(b_0 - a_0); \\ x_2^{(0)} &= a_0 + \tau(b_0 - a_0); \end{aligned}$$

Рисунок 19. Шаг 0.

где $\tau = \frac{b}{a} = \frac{a}{a+b}$, тогда

$$x_1^{(0)} = a_0 + \left(1 - \frac{\sqrt{5} - 1}{2}\right)(b_0 - a_0) = a_0 + 0.381966011(b_0 - a_0);$$

$$x_2^{(0)} = a_0 + \frac{\sqrt{5} - 1}{2}(b_0 - a_0) = a_0 + 0.618033969(b_0 - a_0);$$

Вычислим $f(x_1^{(0)})$ и $f(x_2^{(0)})$

Шаг 1:

Найдем $[a_1, b_1]$, тогда сравним $f(x_1^{(1)})$ и $f(x_2^{(1)})$:

- если $f(x_1^{(0)}) \leq f(x_2^{(0)})$ (см. салатовая стрелка), то $a_1 = a_0; b_1 = x_2^{(0)}; x_2^{(1)} = x_1^{(0)}; x_1^{(1)} = a_1 + b_1 - x_1^{(0)}; \tilde{x}_1 = x_1^{(0)}$
- если $f(x_1^{(0)}) > f(x_2^{(0)})$ (см. синяя стрелка), то $a_1 = x_1^{(0)}; b_1 = b_0; x_1^{(1)} = x_2^{(0)}; x_2^{(1)} = a_1 + b_1 - x_2^{(0)}; \tilde{x}_1 = x_2^{(0)}$

$$\text{Получим } x_2^{(1)} = a_1 + \frac{\sqrt{5}-1}{2}(b_1 - a_1) = x_1^{(1)} + \frac{\sqrt{5}-1}{2}(b_0 - x_1^{(0)})$$

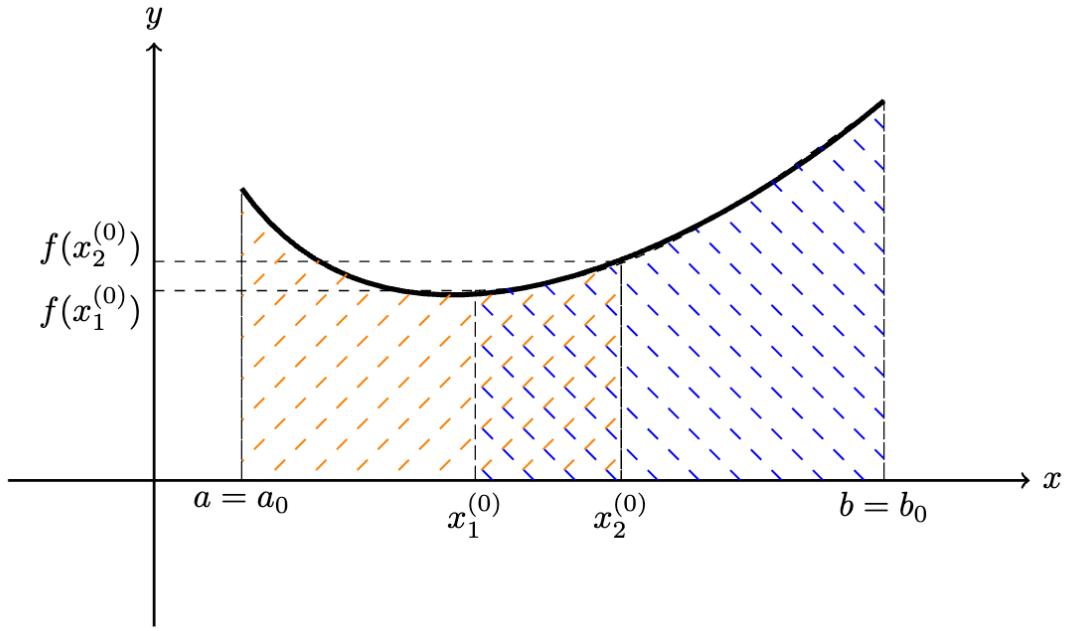


Рисунок 20. Шаг 1.

Шаг *i*:

Определим $[a_i, b_i]$ и точки $x_1^{(i)}$ и $x_2^{(i)}$: для этого сравним $f(x_1^{(i-1)})$ и $f(x_2^{(i-1)})$:

- если $f(x_1^{(i-1)}) \leq f(x_2^{(i-1)})$, то $a_i = a_{i-1}$, $b_i = x_2^{(i-1)}$,
 $x_2^{(i)} = x_1^{(i-1)}$, $x_1^{(i)} = a_i + b_i - x_1^{(i-1)}$, $\tilde{x}_1 = x_1^{(i-1)}$;
- если $f(x_1^{(i-1)}) > f(x_2^{(i-1)})$, то $a_i = x_1^{(i-1)}$, $b_i = b_{i-1}$,
 $x_1^{(i)} = x_2^{(i-1)}$, $x_2^{(i)} = a_i + b_i - x_2^{(i-1)}$; $\tilde{x}_2 = x_2^{(i-1)}$.

Условие останова: $|b_i - a_i| \leq \varepsilon$

Для $i = n$: $|b_n - a_n| \leq \varepsilon$, тогда пусть \tilde{x} – приблизительное решение x_* , т.е.
 $\tilde{x} \approx x_* \pm \varepsilon$, $\tilde{f} = f(\tilde{x})$ – приблизительное значение $f_* = f(x_*)$, тогда
максимальная погрешность ε_n определяется следующим образом:

$$\varepsilon_n = b_n - a_n = \left(\frac{\sqrt{5} - 1}{2} \right)^{n+1} \cdot (b - a)$$

Число шагов, обеспечивающее заданную точность ε нахождения x_*
обеспечивают минимум функции $f(x)$:

$$n \geq \frac{\ln\left(\frac{\varepsilon}{b-a}\right)}{\ln\left(\frac{\sqrt{5}-1}{2}\right)} - 1 \approx -2 \cdot 1 \cdot \ln\left(\frac{\varepsilon}{b-a}\right) - 1$$

Формула степени близости f_* к \tilde{f} :

$$\delta_f = |f_* - \tilde{f}| \equiv |f(x_*) - f(\tilde{x})| \leq L * |x_* - \tilde{x}| \leq L(b_n - a_n) \leq L \frac{\sqrt{5}-1}{2} \varepsilon, \text{ где}$$

$L = \max_{x \in [a_n, b_n]} |\rho(x)|$, $\rho(x) = tg(\alpha)$ – угловой коэффициент касательной к графику функции $f(x)$ в точке x : $\rho(x) = tg(\alpha)$.

Метод Фибоначчи.

Определение.

Последовательность чисел **Фибоначчи** $\{F_n\}, n = 1, 2, \dots$ задается следующим образом:

$$F_{n+2} = F_n + F_{n+1}, \text{ где } F_1 = F_2 = 1$$

и имеет вид: 1, 1, 2, 3, 5, 8,

Замечание.

$$F_k = \frac{1}{\sqrt{5}} \left[\left(\frac{1+\sqrt{5}}{2} \right)^k - \left(\frac{1-\sqrt{5}}{2} \right)^k \right], \text{ где } k = 1, 2, \dots$$

Тогда пусть $f(x)$ унимодальна на $[a, b]$, требуется найти x_* функции $f(x)$ на $[a, b]$ с абсолютной погрешностью $\varepsilon > 0$

Алгоритм метода:

Шаг 1:

Положим $a_1 = a$, $b_1 = b$, найдем $x_1^{(1)}$ и $x_2^{(1)}$ по формулам:

$$x_1^{(1)} = a_1 + \frac{F_n}{F_{n+2}}(b_1 - a_1),$$

$$x_2^{(1)} = a_1 + \frac{F_{n+1}}{F_{n+2}}(b_1 - a_1) = a_1 + b_1 - x_1^{(1)}.$$

Вычислим $f(x_1^{(1)})$ и $f(x_2^{(1)})$:

Шаг 2:

Определим новый отрезок $[a_2, b_2]$ и $x_1^{(2)}$ и $x_2^{(2)}$:

- если $f(x_1^{(1)}) \leq f(x_2^{(1)})$, то $a_2 = a_1; b_2 = x_2^{(1)}; x_2^{(2)} = x_1^{(1)}; x_1^{(2)} = a_2 + b_2 - x_1^{(1)}; \tilde{x}_2 = x_1^{(1)}$
- если $f(x_1^{(1)}) > f(x_2^{(1)})$, то $a_2 = x_1^{(1)}; b_2 = b_1; x_1^{(2)} = x_2^{(1)}; x_2^{(2)} = a_2 + b_2 - x_2^{(1)}; \tilde{x}_2 = x_2^{(1)}$

На новом отрезке $[a_2, b_2]$ вычислим только значения функции $f(x)$ в точке $x_1^{(2)} \rightarrow f(x_1^{(2)})$ или в точке $x_2^{(2)} \rightarrow f(x_2^{(2)})$

Шаг i ($i \geq 3$):

Определим новый отрезок $[a_i, b_i]$ и точки $x_1^{(i)}$ и $x_2^{(i)}$, сравним $f(x_1^{(i-1)})$ и $f(x_2^{(i-1)})$:

- если $f(x_1^{(i-1)}) \leq f(x_2^{(i-1)})$, то $a_i = a_{i-1}; b_i = x_2^{(i-1)}; x_2^{(i)} = x_1^{(i-1)}; x_1^{(i)} = a_i + b_i - x_1^{(i-1)}; \tilde{x}_i = x_1^{(i-1)} = a_i + \frac{F_{n-i+1}}{F_{n-i+3}}(b_i - a_i) = a_i + \frac{F_{n-i+1}}{F_{n+2}}(b_1 - a_1);$
- если $f(x_1^{(i-1)}) > f(x_2^{(i-1)})$, то $a_i = x_1^{(i-1)}; b_i = b_{i-1}; x_1^{(i)} = x_2^{(i-1)}; x_2^{(i)} = a_i + b_i - x_2^{(i-1)}; \tilde{x}_i = x_2^{(i-1)} = a_i + \frac{F_{n-i+2}}{F_{n-i+3}}(b_i - a_i) = a_i + \frac{F_{n-i+2}}{F_{n+2}}(b_1 - a_1).$

Таким образом, приближенное значение $x_* \approx \tilde{x}_i \pm \varepsilon$.

Максимальная погрешность ε_n определения точки минимума x_* :

$$\varepsilon_n = \frac{b_n - a_n}{2} = \frac{b - a}{F_{n-2}} \leq \varepsilon$$

Степень близости f_* и \tilde{f} на $[a, b]$ оценивается следующим образом:

$$\delta_f = |f_* - \tilde{f}| \equiv |f(x_*) - f(\tilde{x})| \leq L * |x_* - \tilde{x}| \leq L \cdot \frac{b_n - a_n}{2} \leq L \cdot \varepsilon, \text{ где}$$

$L = \max_{x \in [a_n, b_n]} |\rho(x)|$, $\rho(x) = tg(\alpha)$ – угловой коэффициент касательной к графику

функции $f(x)$ в точке x : $\rho(x) = tg(\alpha)$.

Примеры реализации методов на языке Julia.

Метод перебора:

```
function perebor(f, a, b, step)
    x_vals = a:step:b
    y_vals = f.(x_vals)
    min_index = argmin(y_vals)
    return x_vals[min_index], y_vals[min_index]
end
```

Метод деления пополам (дихотомия):

```
function bisection(f, a, b, eps)
    a = Float64(a)
    b = Float64(b)
    intervals = [(a,b)]
    while b - a > eps
        m = (a + b) / 2
        if f(m - eps) < f(m + eps)
            b = m
        else
            a = m
        end
        push!(intervals, (a,b))
    end

    min = (a + b) / 2
    return min, f(min), intervals
end
```

Метод Золотого сечения:

```
function golden_section(f, a, b, eps)
    k = (sqrt(5) - 1) / 2
    x1 = a + (1 - k) * (b - a)
    x2 = a + k * (b - a)
    a = Float64(a)
    b = Float64(b)
    intervals = [(a,b)]
    iters = 0
    while abs(x1 - x2) > eps
        iters += 1
        if f(x1) <= f(x2)
            b = x2
            x2 = x1
            x1 = a + b - x1
        else
            a = x1
            x1 = x2
            x2 = a + b - x2
        end
        push!(intervals, (x1,x2))
    end
end
```

```

    min = (a + b) / 2
    return min, f(min), intervals, iters
end

```

Метод Фибоначчи:

```

function fibonacci(n)
    if n == 1 || n == 2
        return 1
    end
    return fibonacci(n - 1) + fibonacci(n - 2)
end

function fibonacci_search(f, a, b, eps)
    n = 10
    fib = [fibonacci(i) for i in 1:n] # Генерация чисел Фибоначчи

    x1 = a + (fib[n-2] / fib[n]) * (b - a)
    x2 = a + (fib[n-1] / fib[n]) * (b - a)

    a = Float64(a)
    b = Float64(b)
    intervals = [(a,b)]
    iters = 0
    for k in 1:(n-3)
        iters += 1
        if f(x1) > f(x2)
            a = x1
            x1 = x2
            x2 = a + (fib[n-k-1] / fib[n-k]) * (b - a)
        else
            b = x2
            x2 = x1
            x1 = a + (fib[n-k-2] / fib[n-k]) * (b - a)
        end
        push!(intervals, (a,b))
    end
    min = (a + b) / 2
    return min, f(min), intervals, iters
end

```

Лекция 2.

Необходимое условие существования локального экстремума.

Теорема 2.1. (Теорема Ферма)

Пусть функция $f(x)$ определена и непрерывна на отрезке $[a, b]$, при этом эта функция дифференцируема в некоторой точке $x_0 \in (a, b)$. Тогда если x_0 является точкой локального экстремума, то $f'(x_0) = 0$.

Заметим, что в обратную сторону утверждение не выполняется, то есть из того, что $f'(x_0) = 0$ не следует, что x_0 является точкой локального экстремума.

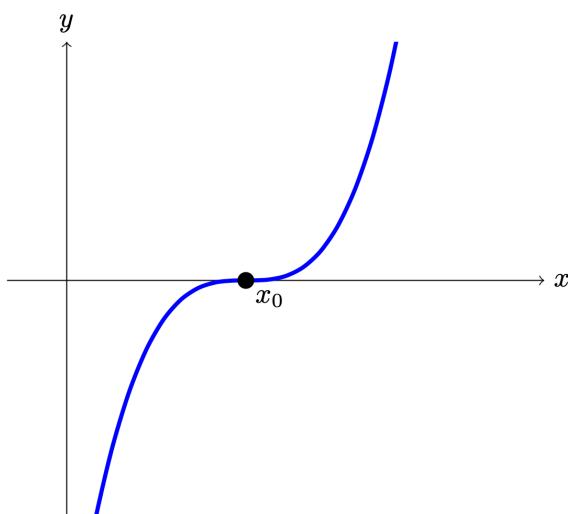


Рисунок 21. Обратное утверждение.

Геометрический смысл теоремы. Производная параллельно оси Ox , то есть функция в этой точке не имеет ускорения.

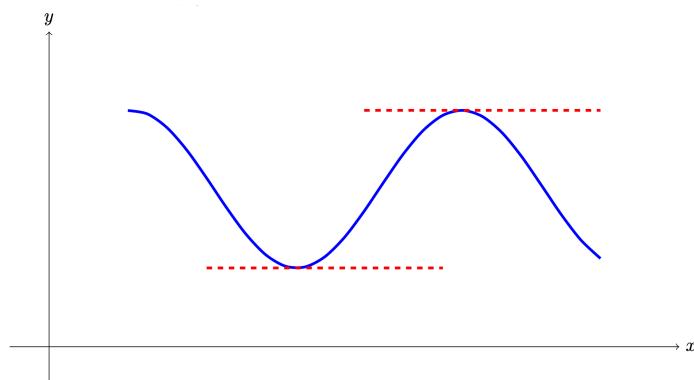


Рисунок 22. Геометрический смысл теоремы Ферма.

Следствие 2.1.1.

Если функция $f(x)$ дифференцируема на отрезке $[a, b]$, то она может иметь экстремумы только в точках, в которых выполняется условие $f'(x) = 0$.

Следствие 2.1.2.

Функция $f(x)$ может иметь экстремум в точках, где производная не существует.

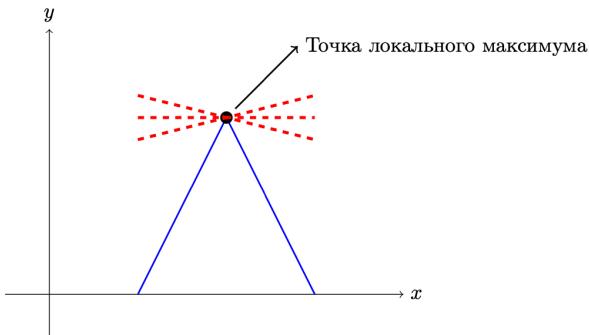


Рисунок 23. Точка локального максимума.

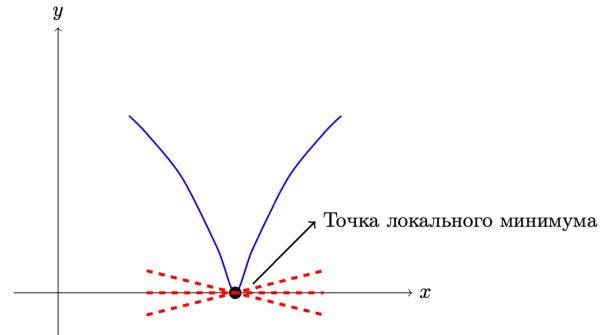


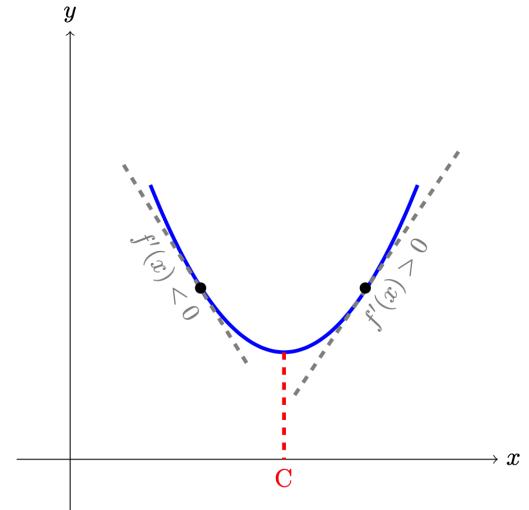
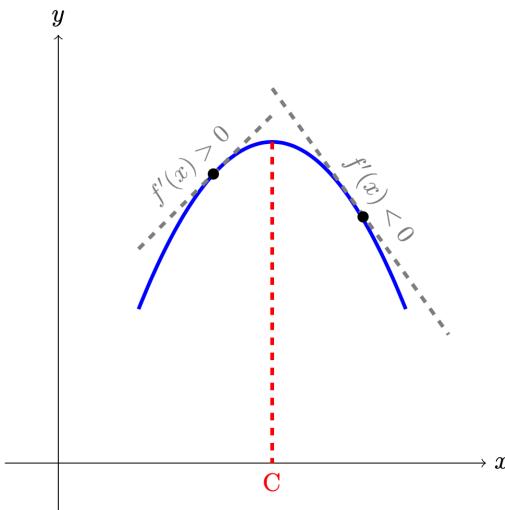
Рисунок 24. Точка локального минимума.

Достаточные условия существования локального экстремума.

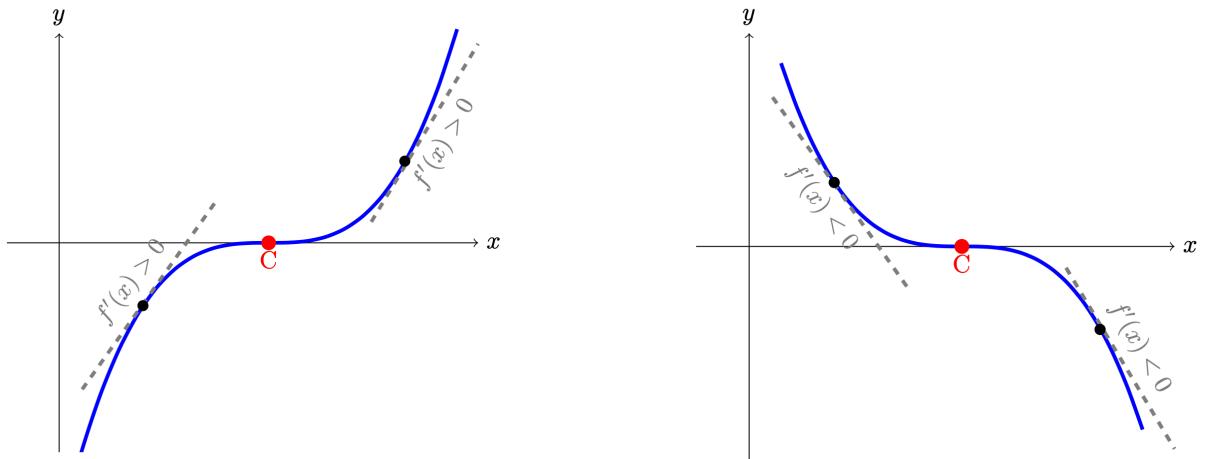
Теорема 2.2. (Для дифференцируемых функций)

Пусть точка c является точкой возможного экстремума функции $f(x)$, такой что функция $f(x)$ дифференцируема всюду в некоторой окрестности точки c . Тогда:

- 1) Если $f'(x) > 0$ (или $f'(x) < 0$) для $x < c$ и $f'(x) < 0$ (или $f'(x) > 0$) для $x > c$, то точка c – точка локального максимума (минимума) функции $f(x)$;



2) Если $f'(x)$ имеет один и тот же знак и слева, и справа, то экстремума в точке c нет.

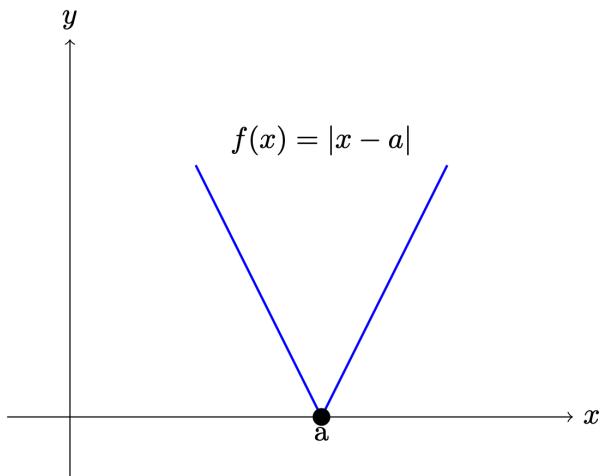


Теорема 2.3. (Для не дифференцируемых функций в точке возможного экстремума)

Пусть точка c является точкой возможного экстремума функции $f(x)$, а функция $f(x)$ является дифференцируемой в некоторой окрестности точки c , кроме самой точки c , и непрерывной в точке c . Тогда точка c является точкой локального максимума (минимума), если:

- 1) $f'(x) > 0$ ($f'(x) < 0$) для $x < c$;
- 2) $f'(x) < 0$ ($f'(x) > 0$) для $x > c$.

Если $f'(x)$ имеет один и тот же знак слева и справа, то точка c не является точкой локального экстремума.



Общая схема отыскания экстремума.

Пусть функция $f(x)$ непрерывна и дифференцируема на (a, b) за исключением конечного количества точек.

Шаг 1. Ищем точки возможного экстремума или критические точки:

- $f'(x) = 0$;
- $f'(x)$ – не существует.

Располагаем предполагаемые точки экстремума в порядке возрастания:
 $a < x_1 < x_2 < \dots < x_n < b$.

Шаг 2. Определяем знак производной функции $f'(x)$ в областях $(a, x_1), (x_1, x_2), \dots, (x_n, b)$.

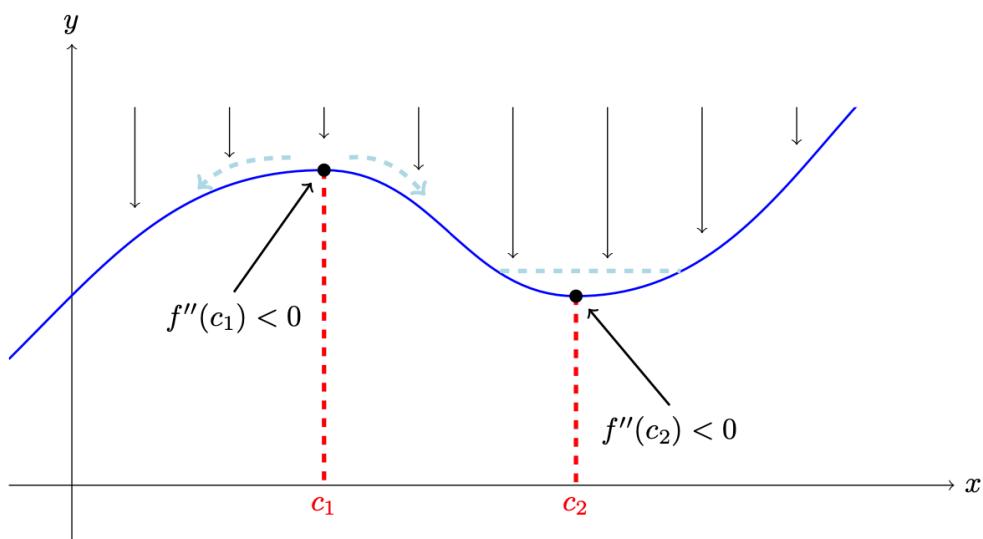
Шаг 3. Вычисляем значения функции $f(x_1), f(x_2), \dots, f(x_n)$.

Шаг 4. Определяем тип экстремума по теореме 2.2. или 2.3.

Теорема 2.4. (Второй достаточный признак экстремума)

Пусть функция $f(x)$ имеет в критической точке c конечную вторую производную $f''(x)$. Тогда:

- 1) Если $f''(c) > 0$, то точка c – точка локального минимума.
- 2) Если $f''(c) < 0$, то точка c – точка локального максимума.



Теорема 2.5. (Третий достаточный признак экстремума)

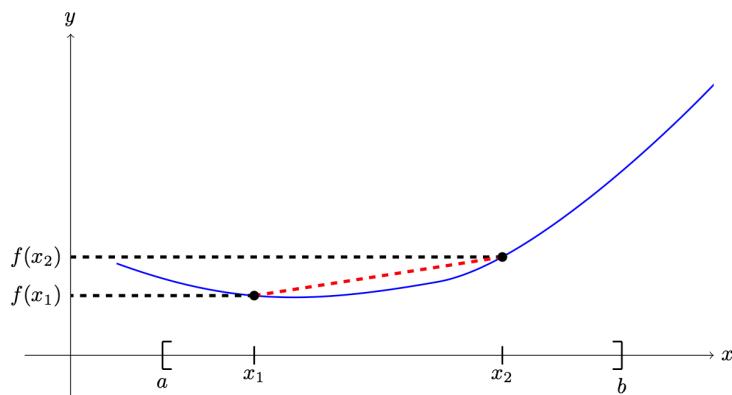
Пусть функция $f(x)$ в некоторой точке c имеет конечную производную порядка $2n$, то есть $\exists f^{(2n)}(c)$ и $f'(c) = f''(c) = \dots = f^{(2n-1)}(c) = 0$. Тогда:

- 1) при $f^{(2n)}(c) > 0$ точка c является точкой локального минимума функции $f(x)$;
- 2) при $f^{(2n)}(c) < 0$ точка c является точкой локального максимума функции $f(x)$.

Выпуклые функции.

Определение.

Функция $f(x)$, заданная на множестве $X \in [a, b]$, представляющем из себя отрезок, называется **выпуклой на X** , если выполняется следующее условие:
 $f(\alpha x_1 + (1 - \alpha)x_2) \leq \alpha f(x_1) + (1 - \alpha)f(x_2), \forall x_1, x_2 \in [a, b], \alpha \in [0, 1]$.

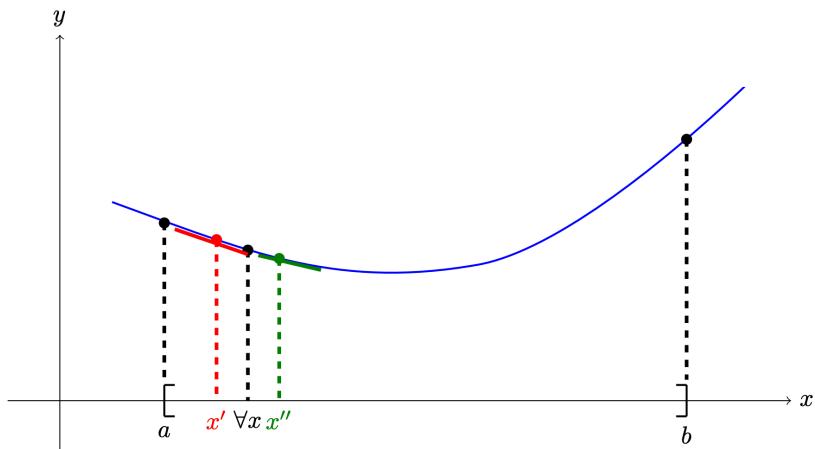


Теорема 2.6.

Пусть функция $f(x)$ выпуклая на отрезке $[a, b]$, непрерывная для любого $x \in (a, b)$ и имеет конечные односторонние производные:

- справа: $f'(x+0) = \lim_{\lambda \rightarrow 0} \frac{f(x+\lambda) - f(x)}{\lambda}$;
- слева: $f'(x-0) = \lim_{\lambda \rightarrow 0} \frac{f(x) - f(x-\lambda)}{\lambda}$.

Тогда $f'(x-0) \leq f'(x+0)$, другими словами, производная слева меньше, чем справа.



Замечание.

На концах отрезка $[a, b]$ выпуклая функция может не иметь односторонних производных, более того может терпеть разрывы.

Теорема 2.7.

График выпуклой всюду дифференцируемой на отрезке $[a, b]$ функции $f(x)$ лежит не ниже любой касательной к нему, причем для любого $x_0 \in [a, b]$:

$$f(x) \geq f(x_0) + f'(x_0)(x - x_0), x \in [a, b].$$

Теорема 2.8.

Для того, чтобы дифференцируемая на отрезке $[a, b]$ функция $f(x)$ была выпуклой, необходимо и достаточно, чтобы её производная $f'(x)$ не убывала на $[a, b]$.

Теорема 2.9.

Для того, чтобы дважды дифференцируемая на отрезке $[a, b]$ функция $f(x)$ была выпуклой, необходимо и достаточно, чтобы вторая производная была неотрицательной на этом отрезке, то есть $f''(x) \geq 0$ для $x \in [a, b]$.

Оптимизация функций многих переменных.

Формулировка задачи безусловной оптимизации целевой функции: для функции $f(x) : \mathbb{R}^n \rightarrow \mathbb{R}, x \in \mathbb{R}^n$ (без ограничений), решаем экстремальную задачу $f(x) \underset{x \in \mathbb{R}^n}{\longrightarrow} \text{extr.}$

Определение.

Градиентом функции $f(x) = f(x_1, x_2, \dots, x_n)$, $x \in \mathbb{R}^n$ в точке x^* называется $\nabla f(x^*) = \left(\frac{\partial f(x^*)}{\partial x_1}, \frac{\partial f(x^*)}{\partial x_2}, \dots, \frac{\partial f(x^*)}{\partial x_n} \right)$.

Определение.

Точка $x^* \in \mathbb{R}$, в которой выполняется условие $\nabla f(x^*) = 0$ или $\frac{\partial f(x^*)}{\partial x_i} = 0 \forall i$, называется **стационарной точкой**.

Теорема 4.1. (Необходимое условие экстремума 1-го порядка)

Пусть функция $f(x) : \mathbb{R}^n \rightarrow \mathbb{R}$ дифференцируема в точке $x^* \in \mathbb{R}$. Если точка x^* является точкой безусловного локального экстремума, то $\nabla f(x^*) = 0$.

Определение 4.3.

Пусть $A = \{a_{ij}\}$, $i, j = 1, 2, \dots, n$, где A – симметричная матрица. Тогда A называется положительно определенной матрицей, если $\forall h \in \mathbb{R}^n, h \neq 0 :$

$$(h, Ah) = \sum_{i=1}^n \sum_{j=1}^n a_{ij} h_i h_j > 0.$$

Если:

- $(h, Ah) < 0$, то A – отрицательно определена;
- $(h, Ah) > 0$, то A – положительно определена;
- $(h, Ah) \leq 0$, то A – неотрицательно определена;
- $(h, Ah) \geq 0$, то A – неположительно определена.

Замечание.

Положительная (отрицательная) определенность A или квадратичной формы (h, Ah) может быть установлена с помощью критерия Сильвестра.

Теорема 4.2. (Критерий Сильвестра)

Симметричная матрица A является положительно определенной тогда и только тогда, когда её главные миноры положительны:

$$\det A_k = \begin{vmatrix} a_{11} & \dots & a_{1k} \\ \dots & \dots & \dots \\ a_{k1} & \dots & a_{kk} \end{vmatrix} > 0, k = 1, 2, \dots, n.$$

Замечание.

Для отрицательной определенности матрицы необходимо и достаточно выполнение следующего условия:

$$(-1)^k \det A_k = (-1)^k \begin{vmatrix} a_{11} & \dots & a_{1k} \\ \dots & \dots & \dots \\ a_{k1} & \dots & a_{kk} \end{vmatrix} > 0, k = 1, 2, \dots, n.$$

Теорема 4.3.

Необходимым и достаточным условием неотрицательной определенности симметричной матрицы A является выполнение следующих ($2^n - 1$) неравенств:

$$\left\{ \begin{array}{l} a_{11} \geq 0, a_{22} \geq 0, \dots, a_{nn} > 0; \\ \left| \begin{matrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{matrix} \right| \geq 0, \left| \begin{matrix} a_{11} & a_{13} \\ a_{31} & a_{33} \end{matrix} \right| \geq 0, \dots, \left| \begin{matrix} a_{n-1, n-1} & a_{n-1, n} \\ a_{n, n-1} & a_{nn} \end{matrix} \right| \geq 0, \left| \begin{matrix} a_{11} & \dots & a_{1n} \\ \dots & \dots & \dots \\ a_{n1} & \dots & a_{nn} \end{matrix} \right| \geq 0 \end{array} \right..$$

Соответственно необходимым и достаточным условием неположительной определенности симметричной матрицы A является выполнения следующих неравенств:

$$\left\{ \begin{array}{l} a_{11} \leq 0, a_{22} \leq 0, \dots, a_{nn} \leq 0; \\ \left| \begin{matrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{matrix} \right| \leq 0, \left| \begin{matrix} a_{11} & a_{13} \\ a_{31} & a_{33} \end{matrix} \right| \leq 0, \dots, \left| \begin{matrix} a_{n-1, n-1} & a_{n-1, n} \\ a_{n, n-1} & a_{nn} \end{matrix} \right| \leq 0, \left| \begin{matrix} a_{11} & \dots & a_{1n} \\ \dots & \dots & \dots \\ a_{n1} & \dots & a_{nn} \end{matrix} \right| \leq 0 \end{array} \right..$$

Теорема 4.4. (Необходимое условие экстремума 2-го порядка)

Пусть функция $f(x) : \mathbb{R}^n \rightarrow \mathbb{R}$ дважды дифференцируема в точке $x^* \in \mathbb{R}^n$. Если точка x^* является точкой безусловного локального минимума функции $f(x)$, то матрица Гессе $H(x^*) = \frac{\partial^2 f(x^*)}{\partial x_i \partial x_j}, i, j = 1, 2, \dots, n$, неотрицательно определена, то есть $\forall h \in \mathbb{R}^n : (h, H(x^*)h) \geq 0$.

Если x^* является точкой безусловного локального максимума функции $f(x)$, то матрица Гессе $H(x^*)$ неположительно определена, то есть $\forall h \in \mathbb{R}^n : (h, H(x^*)h) \leq 0$.

Теорема 4.5. (Достаточное условие экстремума)

Пусть функция $f(x) : \mathbb{R}^n \rightarrow \mathbb{R}$ дважды дифференцируема в точке $x^* \in \mathbb{R}$.

1. Для того, чтобы точка x^* являлась точкой безусловного локального минимума функции $f(x)$, достаточно, чтобы:

- $\nabla f(x^*) = 0$;
- $\forall h \ (h, H(x^*)h) > 0$.

2. Для того, чтобы точка x^* являлась точкой безусловного локального максимума функции $f(x)$, достаточно, чтобы:

- $\nabla f(x^*) = 0$;
- $\forall h \ (h, H(x^*)h) < 0$.

▷ Так как функция $f(x) : \mathbb{R}^n \rightarrow \mathbb{R}$ дважды дифференцируема в точке $x^* \in \mathbb{R}^n$, то для $\forall h = (h_1, h_2, \dots, h_n) \in \mathbb{R}^n : f(x^* + h^*) = f(x^*) + \frac{1}{1!}(\nabla f(x^*), h) + \frac{1}{2!}(h, H(x^*)h) + \alpha(h)$, где $\lim_{\|h\| \rightarrow 0} \frac{\alpha(h)}{\|h\|^2} = 0$. (*)

Учитывая условие Теоремы 4.5.: $(h, H(x^*)h) > 0$ (**), так как матрица Гессе $H(x^*)$ положительно определена при $h \neq 0$. Тогда:

$$\begin{cases} (*) \\ (**) \end{cases} \Rightarrow f(x^* + h) - f(x^*) = \frac{1}{2}(h, H(x^*)h) + \alpha(h) \quad (***)$$

$$> 0, \quad > 0, \quad \|h\| > 0$$

Таким образом равенство (***) выполняется в некоторой окрестности точки x^* :

$\cup_\delta (x^*) = \{x \in \mathbb{R}^n : x = x + h, \|h\| < \delta\}$, таким образом $f(x^* + h) - f(x^*) > 0 \Rightarrow f(x^* + h) > f(x^*) \Rightarrow x^*$ по определению является точкой безусловного локального минимума функции $f(x)$. ◁

Теорема 4.6.

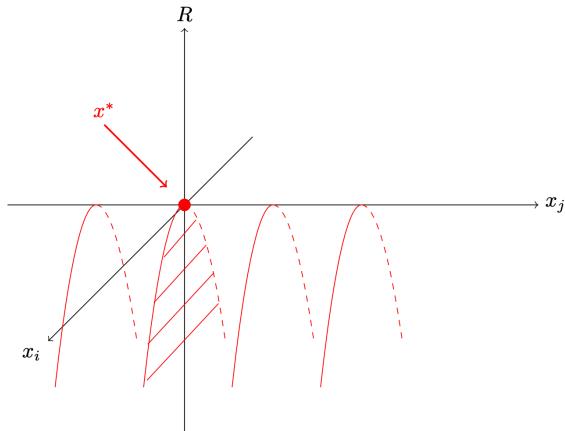
Пусть x^* – стационарная точка функции $f(x) : \mathbb{R}^n \rightarrow \mathbb{R}$, а функция $f(x)$ дважды дифференцируема в точке $x^* \in \mathbb{R}^n$ и все вторые частные производные функции $f(x)$ непрерывны в точке x^* . Тогда:

- Если 2-ой дифференциал функции $d^2f(x^*, \Delta x_1, \Delta x_2, \dots, \Delta x_n) > 0, \forall \Delta x_i, i = 1, 2, \dots, n$, из окрестности точки x^* , то точка x^* является точкой безусловного локального минимума функции $f(x)$.
- Если 2-ой дифференциал функции $d^2f(x^*, \Delta x_1, \Delta x_2, \dots, \Delta x_n) < 0, \forall \Delta x_i, i = 1, 2, \dots, n$, из окрестности точки x^* , то точка x^* является точкой безусловного локального максимума функции $f(x)$.
- Если 2-ой дифференциал функции $d^2f(x^*, \Delta x_1, \Delta x_2, \dots, \Delta x_n)$ – знакопеременная функция, то точка x^* не является точкой экстремума функции $f(x)$.

Замечание. x^* – седловая точка.

- Если 2-ой дифференциал функции $d^2f(x^*, \Delta x_1, \Delta x_2, \dots, \Delta x_n) \geq 0$ или $d^2f(x^*, \Delta x_1, \Delta x_2, \dots, \Delta x_n) \leq 0$, причем существуют такие наборы $\Delta x_1, \Delta x_2, \dots, \Delta x_n$, не равных одновременно нулю, для которых значение второго дифференциала обращается в нуль, то функция $f(x)$ в точке x^* может иметь экстремум, но может не иметь его.

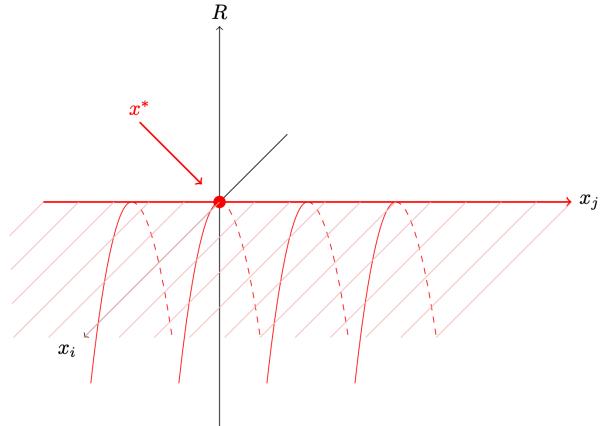
Пример.



В плоскости $x_j OR$

$$d^2f(x^*, \Delta x_1, \Delta x_2, \dots, \Delta x_n) < 0 \Rightarrow$$

экстремум есть.



В плоскости $x_j OR$

$$d^2f(x^*, \Delta x_1, \Delta x_2, \dots, \Delta x_n) = 0 \Rightarrow$$

экстремума нет

Общая схема отыскания безусловного экстремума функции многих переменных.

Шаг 1. Найти точки возможного экстремума функции $f(x)$:

- $\nabla f(x) = 0$ – стационарные точки;
- точки, в которых частные $\frac{\partial f(x)}{\partial x_i}, i = 1, 2, \dots, n$ не существуют.

Шаг 2. Проанализировать выполнение достаточных условий экстремума (Теоремы 4.5. или 4.6.).

Шаг 3. Вычислить $f_{\text{экстр}}(x^*)$.

Пример 4.1.

Исследовать функцию $f(x) = f(x_1, x_2) = x_1 x_2 - \frac{1}{2(x_1+x_2)} \in \mathbb{R}, x \in \mathbb{R}^2$.

$$\triangleright 1. \nabla f(x) = \nabla f(x_1, x_2) = \left(x_2 + \frac{1}{2(x_1+x_2)^2}, x_1 + \frac{1}{2(x_1+x_2)} \right) \in \mathbb{R}^2.$$

$$2. H(x) = H(x_1, x_2) = \begin{pmatrix} -\frac{1}{(x_1+x_2)^3} & 1 - \frac{1}{(x_1+x_2)^3} \\ 1 - \frac{1}{(x_1+x_2)^3} & -\frac{1}{(x_1+x_2)^3} \end{pmatrix}.$$

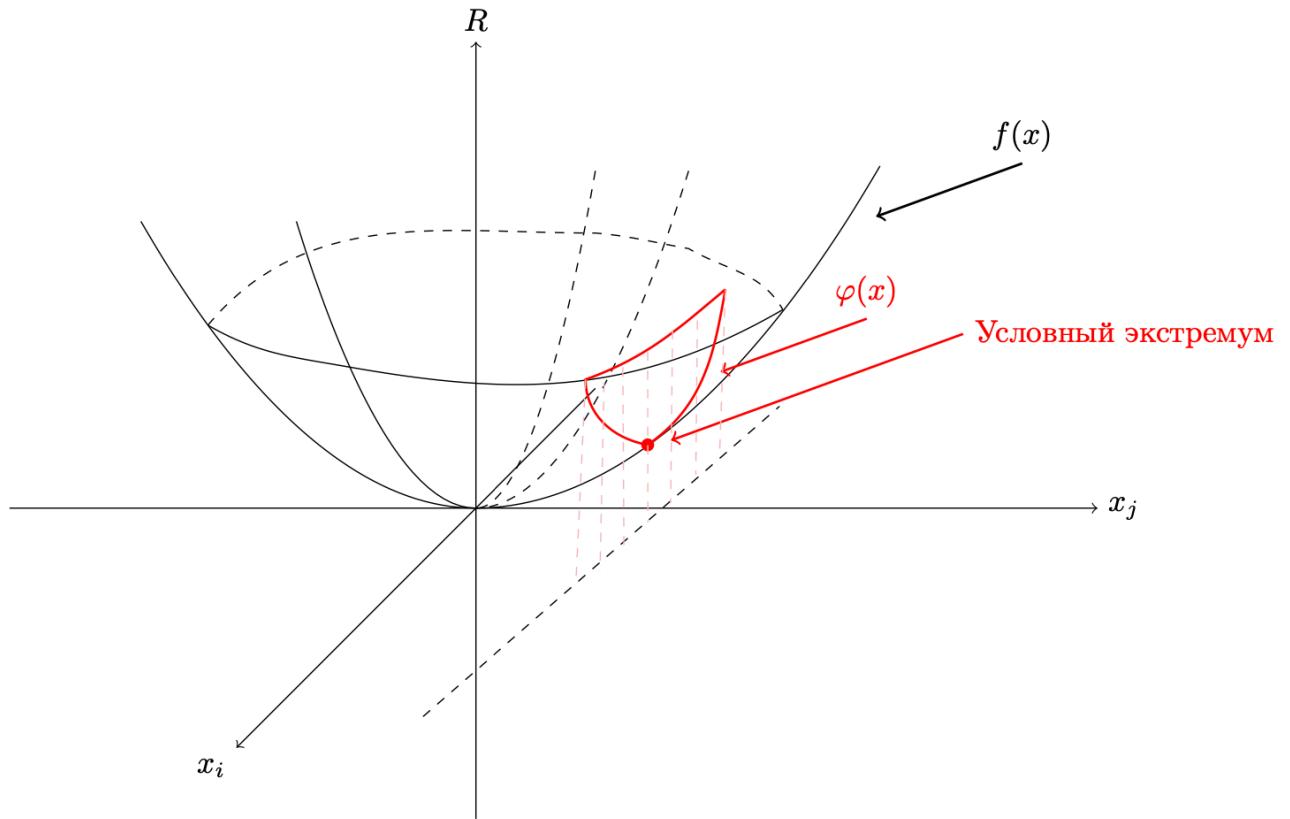
3. Найдем стационарные точки функции $f(x)$ по теореме 4.1. (необходимое условие: $\nabla f(x_1, x_2) = 0$):

$$\nabla f(x) = \nabla f(x_1, x_2) = \begin{pmatrix} x_2 + \frac{1}{2(x_1+x_2)^2} \\ x_1 + \frac{1}{2(x_1+x_2)^2} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix} = 0 \in \mathbb{R}^2 \Rightarrow$$

$$x_1 + \frac{1}{8x_1^2} = 0 \Rightarrow x_1^3 = -\frac{1}{8} \Rightarrow x_1 = -\frac{1}{2}, x_2 = -\frac{1}{2}.$$

4. $H(x^*) = \begin{pmatrix} 1 & 2 \\ 2 & 1 \end{pmatrix}$, тогда $\det H(x^*) = \det H\left(-\frac{1}{2}, -\frac{1}{2}\right) < 0$, то есть матрица не является ни положительно определенной, ни отрицательно определенной $\Rightarrow x^*$ не является экстремальной точкой. \triangleleft

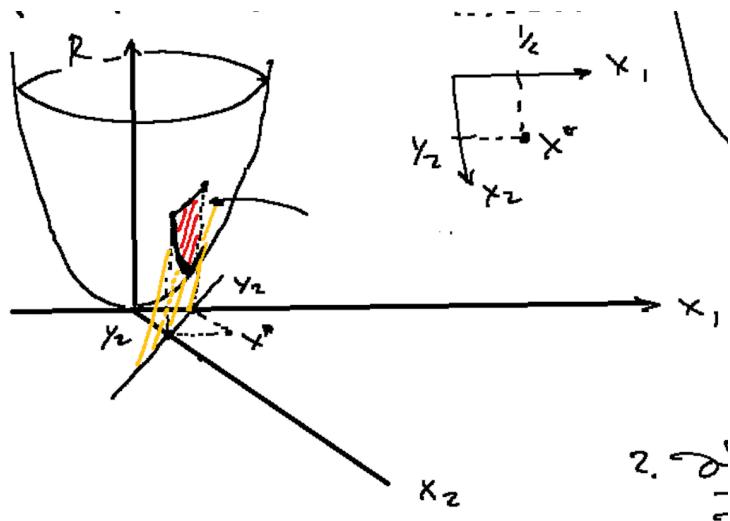
Условный экстремум функции нескольких переменных.



$f(x) : \mathbb{R}^n \rightarrow \mathbb{R}, x \in \mathbb{R}^n : f(x) \xrightarrow[x \in X]{} extr,$ где $X = \{x \in \mathbb{R}^n : \varphi_j(x) = 0, j = 1, 2, \dots, m\}$

Пример.

$$f(x) = f(x_1, x_2) = x_1^2 + x_2^2, \text{ условие } x_1 + x_2 = 1.$$



1. $x_1 + x_2 = 1 \Leftrightarrow x_1 + x_2 - 1 = 0 = \varphi(x)$ – уравнение связи. $\Rightarrow x_2 = 1 - x_1$, получим $f(x) = f(x_1, x_2) = x_1^2 + (1 - x_1)^2 = 2x_1^2 - 2x_1 + 1$.
2. Необходимое условие: $\frac{\partial f(x_i)}{\partial x_i} = 4x_1 - 2 = 0 \Rightarrow x_1^* = \frac{1}{2}$.
3. $\frac{\partial^2 f(x_1^*)}{\partial x_1^2} = 4 > 0$ по правилу «дождя-улыбки» $\Rightarrow x_1^*$ - точка минимума.

Необходимое условие существования локального экстремума.

Определение 4.4.

- a) Точка $x^* \in X \subset \mathbb{R}$ - **точка условного минимума** функции $f(x)$ при условной связи $\varphi_j(x) = 0, j = 1, \dots, m$, если $\exists U(x^k): \forall x \in U(x^k): x \neq x^k \Rightarrow f(x^k) < f(x)$.
- б) Точка $x^* \in X \subset \mathbb{R}$ - **точка условного максимума** функции $f(x)$ при условной связи $\varphi_j(x) = 0, j = 1, \dots, m$, если $\exists U(x^k): \forall x \in U(x^k): x \neq x^k \Rightarrow f(x) < f(x^k)$.

Определение 4.5.

Функцией Лагранжа, соответствующей функции $f(x)$ и условной связи $\varphi_j(x) = 0, j = 1, \dots, m$, где m – количество условных связей, называется функция:

$$\begin{aligned} L(x, \lambda) &= L(x_1, x_2, \dots, x_n, \lambda_1, \lambda_2, \dots, \lambda_m) \\ &= f(x_1, x_2, \dots, x_n) + \sum_{j=1}^m \lambda_j \varphi_j(x_1, x_2, \dots, x_n). \end{aligned}$$

Теорема 4.7. (о соответствии точек экстремума)

Точке безусловного экстремума функции Лагранжа $L(x, \lambda)$ соответствует точка x^* - точка условного экстремума соответствующей функции $f(x)$ при условной связи $\varphi_j(x) = 0, j = 1, \dots, m$.

Общая схема поиска условного экстремума:

Шаг 1. Построим функцию Лагранжа: $L(x, \lambda) = L(x_1, x_2, \dots, x_n, \lambda_1, \lambda_2, \dots, \lambda_m)$.

Шаг 2. Используя необходимое условие существования экстремума функции Лагранжа ищем (аналитически) возможные точки безусловных экстремумов $L(x, \lambda)$:

а) найдем стационарные точки $L(x, \lambda)$:

$$\begin{cases} \frac{\partial L(x, \lambda)}{\partial x_1} = 0, & \frac{\partial L(x, \lambda)}{\partial \lambda_1} = 0 \\ \frac{\partial L(x, \lambda)}{\partial x_2} = 0, & \frac{\partial L(x, \lambda)}{\partial \lambda_2} = 0 \\ \dots & \dots \\ \frac{\partial L(x, \lambda)}{\partial x_n} = 0, & \frac{\partial L(x, \lambda)}{\partial \lambda_m} = 0 \\ \varphi_1(x) = 0 \\ \varphi_2(x) = 0 \\ \dots \\ \varphi_m(x) = 0 \end{cases}$$

б) найдем точки, в которых частная производная не существует.

Шаг 3. Проверить выполнение достаточного условия безусловного экстремума (Теорема 4.6.) для $L(x, \lambda)$, то есть исследовать знак $\partial^2 L(x, \lambda)$.

Замечание.

$$dx_1^2 + dx_2^2 + \dots + dx_n^2 \neq 0, d\varphi_j = \sum_{i=1}^n \frac{\partial \varphi(x^*)}{\partial x_i} dx_i = 0.$$

Пример.

Пусть есть некоторая фигура F , которая ограничена линиями $x_1 = 0$, $x_2 = 0$, $x_2 + x_1^2 - 6 = 0$.

Задача: вписать внутрь F прямоугольник наибольшей площади.

$\triangleright S(x) = S(x_1, x_2) = f(x) = x_1 x_2$ – целевая функция, $\varphi_j(x) = x_2 + x_1^2 - 6 = 0, j = 1, m = 1$.

Шаг 1. $L(x, \lambda) = L(x_1, x_2, \lambda_1) = S(x) + \sum_{j=1}^{m=1} \lambda_j \varphi_j(x) = x_1 x_2 + \lambda_1 \varphi_1 = x_1 x_2 + (x_2 + x_1^2 - 6) \lambda_1$.

Шаг 2. (необходимое условие экстремума)

$$\begin{cases} \frac{\partial L(x_1, x_2, \lambda_1)}{\partial x_1} = x_2 + 2\lambda_1 x_1 = 0 \\ \frac{\partial L(x_1, x_2, \lambda_1)}{\partial x_2} = x_1 + \lambda_1 = 0 \\ \frac{\partial L(x_1, x_2, \lambda_1)}{\partial \lambda_1} = x_2 + x_1^2 - 6 = 0 \end{cases} \Rightarrow \begin{cases} x_2 + 2\lambda_1 x_1 = 0 \\ x_1 + \lambda_1 = 0 \\ x_2 + x_1^2 - 6 = 0 \end{cases} \Rightarrow x^* = \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} \sqrt{2} \\ 4 \\ -\sqrt{2} \end{pmatrix} -$$

стационарная точка $L(x, \lambda)$.

Шаг 3. (достаточное условие)

$$d^2L(x_1, x_2, \lambda_1) \stackrel{\text{def}}{=} \frac{\partial^2 L(x_1, x_2, \lambda_1)}{\partial x_1^2} dx_1^2 + \frac{\partial^2 L(x_1, x_2, \lambda_1)}{\partial x_2^2} dx_2^2 + \frac{\partial^2 L(x_1, x_2, \lambda_1)}{\partial \lambda_1} d\lambda_1 + 2 \left(\frac{\partial^2 L(x_1, x_2, \lambda_1)}{\partial x_1 \partial x_2} dx_1 dx_2 + \frac{\partial^2 L(x_1, x_2, \lambda_1)}{\partial x_1 \partial \lambda_1} dx_1 d\lambda_1 + \frac{\partial^2 L(x_1, x_2, \lambda_1)}{\partial x_2 \partial \lambda_1} dx_2 d\lambda_1 \right).$$

Вычисляем вторые производные:

$$\frac{\partial^2 L(x, \lambda)}{\partial x_1^2} = 2\lambda_1; \quad \frac{\partial^2 L(x, \lambda)}{\partial x_1 \partial \lambda_1} = 2x_1; \quad \frac{\partial^2 L(x, \lambda)}{\partial x_1 \partial x_2} = 1; \quad \frac{\partial^2 L(x, \lambda)}{\partial \lambda_1^2} = 0; \quad \frac{\partial^2 L(x, \lambda)}{\partial x_2^2} = 0; \quad \frac{\partial^2 L(x, \lambda)}{\partial x_2 \partial \lambda_1} = 1,$$

$$\text{тогда } d^2L(\sqrt{2}, 4, -\sqrt{2}) = -2\sqrt{2}dx_1^2 + 2(dx_1 dx_2 + 2\sqrt{2}dx_1 d\lambda_1 + dx_2 dx_1).$$

Учтем связь: $\varphi(x) = x_2 + x_1^2 - 6 = 0$:

$$d\varphi_1 = \sum_{i=1}^2 \frac{\partial \varphi_1(x)}{\partial x_i} = 0 \Rightarrow d\varphi_1(x) = 0 = \frac{\partial \varphi_1(x)}{\partial x_1} dx_1 + \frac{\partial \varphi_1(x)}{\partial x_2} dx_2 = \frac{\partial}{\partial x_1} (x_2 + x_1^2 - 6) dx_1 + \frac{\partial}{\partial x_2} (x_2 + x_1^2 - 6) dx_2 = 2x_1 dx_1 + dx_2 \xrightarrow{x_1=\sqrt{2}} dx_2 = -2\sqrt{2}dx, \quad \text{тогда}$$

$$d^2L(\sqrt{2}, 4, -\sqrt{2}) = -2\sqrt{2}dx_1^2 - 4\sqrt{2}dx_1^2 + 4\sqrt{2}dx_1 d\lambda_1 - 2\sqrt{2}dx_1 d\lambda_1 = -6\sqrt{2}dx_1^2 < 0 \Rightarrow x^* = (\sqrt{2}, 4, -\sqrt{2})^T - \text{точка максимума функции Лагранжа} \Rightarrow$$

$$x^* - \text{точка условного максимума функции } f(x) \equiv S(x) = S(x_1, x_2). \triangleleft$$

Условный экстремум функции двух переменных.

Пусть:

- $f(x) = f(x_1, x_2), x \in \mathbb{R}^2,$
- $\varphi_1 = \varphi(x_1, x_2) = 0,$
- $L(x_1, x_2, \lambda) = f(x_1, x_2) + \lambda\varphi(x_1, x_2),$

$$- \begin{cases} \frac{\partial L(x_1, x_2, \lambda)}{\partial x_1} = 0 \\ \frac{\partial L(x_1, x_2, \lambda)}{\partial x_2} = 0, x^* = \begin{pmatrix} x_1^* \\ x_2^* \\ x_3^* \end{pmatrix} \\ \varphi(x_1, x_2) = 0 \end{cases} - \text{ некоторое решение этой системы уравнений,}$$

$$-\Delta = -1 \times \begin{vmatrix} 0 & \varphi'_{x_1} & \varphi'_{x_2} \\ \varphi'_{x_1} & L''_{x_1 x_1} & L''_{x_1 x_2} \\ \varphi'_{x_2} & L''_{x_2 x_1} & L''_{x_2 x_2} \end{vmatrix}.$$

Тогда:

- $\Delta < 0$, то $f(x)$ в точке x^* имеет условный максимум;
- $\Delta > 0$, то $f(x)$ в точке x^* имеет условный минимум;
- $\Delta = 0$, то $f(x)$ в точке x^* не имеет условного экстремума.

Пример.

$$f(x) = f(x_1, x_2) = -2x_1 x_2, -x_1 + 2x_2 = 1$$

▷ Шаг 1. $\varphi(x_1, x_2) = -x_1 + 2x_2 - 1$, тогда $L(x_1, x_2, \lambda) = -2x_1 x_2 + \lambda(-x_1 + 2x_2 - 1)$.

Шаг 2. (необходимое условие)

$$\begin{cases} \frac{\partial L(x_1, x_2, \lambda)}{\partial x_1} = -2x_2 - \lambda = 0 \\ \frac{\partial L(x_1, x_2, \lambda)}{\partial x_2} = -2x_1 + 2\lambda = 0 \\ \frac{\partial L(x_1, x_2, \lambda)}{\partial \lambda} = -x_1 + 2x_2 - 1 = 0 \end{cases} \Rightarrow \begin{cases} -2x_2 - \lambda = 0 \\ -2x_1 + 2\lambda = 0 \\ -x_1 + 2x_2 - 1 = 0 \end{cases} \Rightarrow x_2 = -\frac{\lambda}{2}, x_1 = \lambda, \\ -(-\lambda) + 2\left(-\frac{\lambda}{2}\right) - 1 = 0 \Rightarrow \lambda = -\frac{1}{2}, x_2 = \frac{1}{4}, x_1 = -\frac{1}{2} \Rightarrow x^* = \left(-\frac{1}{2}, \frac{1}{4}\right), \lambda^* = -\frac{1}{2}$$

Шаг 3. Вычисляем вторые производные функции Лагранжа и первые производные условия связи:

$$\frac{\partial^2 L(x, \lambda)}{\partial x_1^2} = 0; \frac{\partial^2 L(x, \lambda)}{\partial x_1 \partial x_2} = -2; \frac{\partial^2 L(x, \lambda)}{\partial x_2^2} = 0.$$

$$\varphi'_{x_1} = \frac{\partial}{\partial x_1}(-x_1 + 2x_2 - 1) = -1, \varphi'_{x_2} = \frac{\partial}{\partial x_2}(-x_1 + 2x_2 - 1) = 2$$

Шаг 4. Составим детерминант:

$$\Delta = -1 \times \begin{vmatrix} 0 & \varphi'_{x_1} & \varphi'_{x_2} \\ \varphi'_{x_1} & L''_{x_1 x_1} & L''_{x_1 x_2} \\ \varphi'_{x_2} & L''_{x_2 x_1} & L''_{x_2 x_2} \end{vmatrix} = \begin{vmatrix} 0 & -1 & 2 \\ -1 & 0 & -2 \\ 2 & -2 & 0 \end{vmatrix} = -(4 + 4) = -8 < 0$$

$\Rightarrow x^* = \left(-\frac{1}{2}, \frac{1}{4}\right)$ – точка условного максимума. \triangleleft

Лекция 3. Численные методы минимизации функции нескольких переменных (безусловная оптимизация)

Задача.

$f(x): R^n \rightarrow R$ (без ограничений), $x \in R^n, f(x) \rightarrow \min_{x \in R^n}$

Метод.

Необходимо найти $\{x^i\}: x^{(0)}, x^{(1)}, x^{(2)}, \dots, x^{(k)}, \dots, k \rightarrow \infty : f(x^{(0)}) \geq f(x^{(1)}) \geq \dots \geq f(x^{(k)}) \geq \dots$

Определение.

Последовательность, приведенная выше, называется **релаксационной**, а метод поиска такой последовательности – метод спуска.

Описание метода спуска.

Пусть $x^{(0)}$ – начальная точка, далее $\{x^{(i)}\}: x^{(1)}, x^{(2)}, \dots, x^{(k)}, \dots, :$

- в точке $x^{(k)}$ необходимо выбрать вектор $y^{(k)} \in R^n$ – вектор непрерывного спуска;
- после нахождения $y^{(k)}$, находят $x^{(k+1)}$ – следующее приближение: $x^{(k+1)} = x^{(k)} + \alpha^{(k)}y^{(k)}, k = 0, 1, 2, \dots, \alpha^{(k)} \in R$, где $x^{(k+1)}$ – точка на $(k+1)$ -м шаге $x^{(k)}$ – точка на (k) шаге, $\alpha^{(k)}$ – значение шага спуска, $y^{(k)}$ – вектор направления спуска.

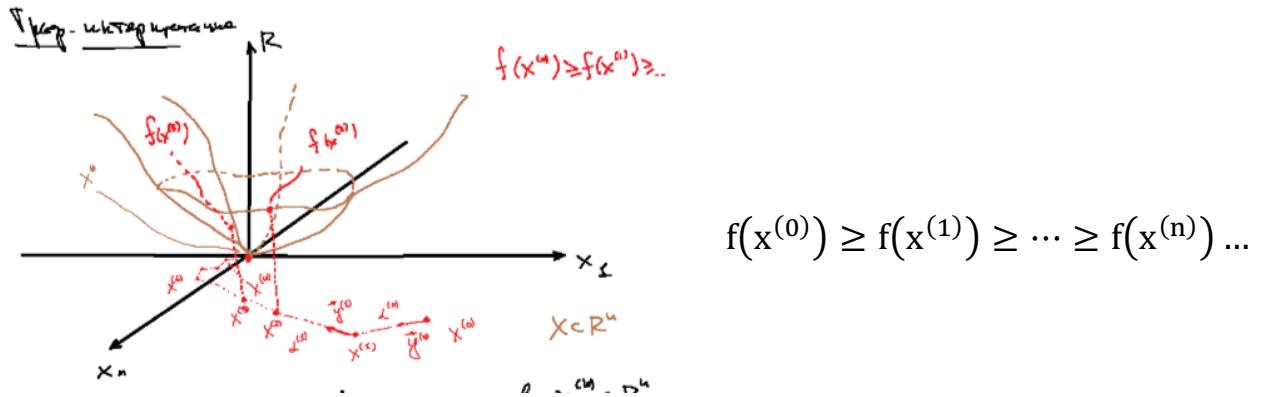
Этапы метода спуска:

1. Выбор направления спуска $y^{(k)}$;
2. Способ движения вдоль направления спуска – $\alpha^{(k)}$.

Задачи методов спуска.

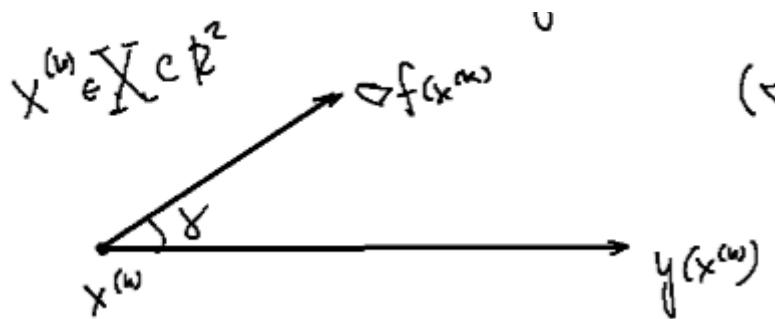
Параметры $\alpha^{(k)} \in R$ и $y^{(k)}$ выбирают таким образом, чтобы на (k) шаге итерационного процесса происходило убывание значения условной функции $f(x)$ в точках $x^{(k)}, k = 0, 1, 2, \dots, k, \dots, k \rightarrow \infty$.

Графическая интерпретация.



Теорема 5.1.

Пусть функция $f(x)$ дифференцируема в $x^{(k)} \in R^n, \forall y \in R^n$ удовлетворяет $(\nabla f(x), y) < 0$, тогда y — определяет направление убывания функции $f(x)$ в $x^{(k)}$, то есть $\exists \alpha > 0, \alpha \in R, f(x^{(k)} + \alpha y) < f(x^{(k)})$

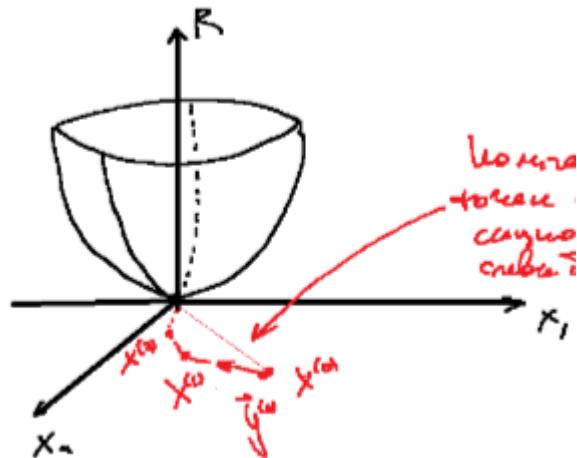


$$(\nabla f(x^{(k)}), y^{(k)}) = \left| \left| \nabla f(x^{(k)}) \right| \right|_2 \|y^{(k)}\|_2 \cos \gamma > 0.$$

Таким образом y — это вектор направления:

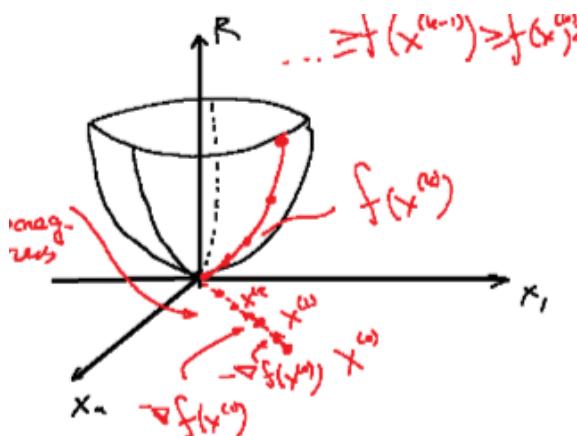
- в сторону роста условной функции: $(\nabla f(x^{(k)}), y^{(k)}) > 0$;
- убывания условной функции: $(\nabla f(x^{(k)}), y^{(k)}) < 0$.

Геометрическая интерпретация.



Количество точек в релаксационной последовательности слева больше, чем справа.

Не оптимальное движение.



$$\dots \geq f(x^{(n-1)}) \geq f(x^{(n)}) \geq f(x^{(n+1)}) \geq \dots$$

Оптимальное движение.

Замечание.

При одинаковом условии остановки $|f(x^{(k)}) - f(x_{min})| \leq \text{eps}$ последовательность минимизации, генерируемая не по направлению антиградиента, будет содержать больше точек, чем та последовательность, которая будет генерироваться по направлению антиградиента при одинаковом выборе $\alpha^{(k)}$.

Метод градиентного спуска.

Алгоритм генерации последовательности $\{x^{(k)}\}: x^{(k+1)} = x^{(k)} + \alpha^{(k)}y^{(k)}, k = 0,1,2, \dots$ $y^{(k)} = -\nabla f(x^{(k)})$, где $\alpha^{(k)} > 0$ – величина шага спуска, значение α – мало для выполнения условия $f(x^{(k)}) \geq f(x^{(k+1)}), k = 0,1,2, \dots$

Теорема 5.2.

Функция $f(x)$, ограниченная снизу, дифференцируема на R^n и её градиент удовлетворяет условию Липшица, $L = \text{const} > 0 : ||\nabla f(x') - \nabla f(x'')|| \leq L||x' - x''||$ для $\forall x', x'' \in R^n$, тогда $\forall x^{(0)} \in R^n$ (начальной точки) итерационного процесса $x^{(k+1)} = x^{(k)} + \alpha y^{(k)}, y^{(k)} = -\nabla f(x^{(k)}), k = 0,1, \dots \exists \alpha > 0, \forall k: \alpha^{(k)} = \alpha = \text{const}$, что $\lim_{k \rightarrow \infty} |\nabla f(x^{(k)})| = 0$, при этом последовательность $\{x^{(k)}\}$ будет являться релаксационной.

Теорема 5.3.

Функция $f(x)$ удовлетворяет условию Липшица и $\exists c \in R: L(c) = \{x \in R^n: f(x) \leq c\}$ не пусто и компактно, тогда $\forall x^{(0)} \in L(c)$ последовательность $\{x^{(k)}\}$, которая определяется по $x^{(k+1)} = x^{(k)} + \alpha^{(k)}y^{(k)}, y^{(k)} = -\nabla f(x^{(k)}), k = 0,1,2, \dots$ будет релаксационной, при этом $\forall x^k$, которая является предельной точкой этой релаксационной последовательности, удовлетворяет условию $\nabla f(x^k) = 0$.

Следствие.

Если дополнительно $f(x)$ – выпуклая, то $\forall x^*$ являющаяся предельной точкой $\{x^{(k)}\}$ будет точкой минимума $f(x)$ на R^n

Замечания по методу градиентного спуска.

1. Итерационный процесс $x^{(k+1)} = x^{(k)} + \alpha y^{(k)}$ может привести ($\alpha^k = \alpha = \text{const}$) к нарушению $f(x^{(k+1)}) = f(x^{(k)} + \alpha y^{(k)}) < f(x^{(k)})$. (**)

Для выполнения (**) необходимо сделать откат, например со значением $\alpha := \frac{\alpha}{2}$ и далее продолжаем процесс: $x^{(k+1)} = x^{(k)} + \frac{\alpha}{2}y^{(k)}$ и так далее

2. Условие остановки:

- $\|x^{(k+1)} - x^{(k)}\| \leq \epsilon_1$
- $\|f(x^{(k+1)}) - f(x^{(k)})\| \leq \epsilon_2$, где $\epsilon_1 > 0, \epsilon_2 > 0$ – заранее заданные числа

Тогда $x^{(k+1)} = x^{(k)} + \alpha y^{(k)} \Rightarrow x^{(k+1)} - x^{(k)} = \alpha y^{(k)} \Rightarrow \|x^{(k+1)} - x^{(k)}\| \leq \alpha \|y^{(k)}\| = \alpha \|\nabla f(x^{(k)})\|$. Тогда получили неравенство условия окончания градиентного спуска: $|\frac{df(x^{(k)})}{dx_i}| \leq \epsilon_3, i = 1, 2, \dots, n$ или $\|\nabla f(x^{(k)})\| \leq \epsilon_3, \epsilon_3 > 0$

3. Выбор решения.

В качестве приближенного решения x^* принимаются последние вычисления $x^{(k)}$, тогда $f^* = f(x^*)$

Метод наискорейшего спуска.

Теорема 5.4

Пусть функция $f(x)$ удовлетворяет условию теоремы 5.2, тогда для любой начальной точки $x^{(0)}$ метод наискорейшего спуска приведет к построению последовательности $\{x^{(k)}\}$ удовлетворяющей $\lim_{k \rightarrow \infty} \|\nabla f(x^{(k)})\| = 0$.

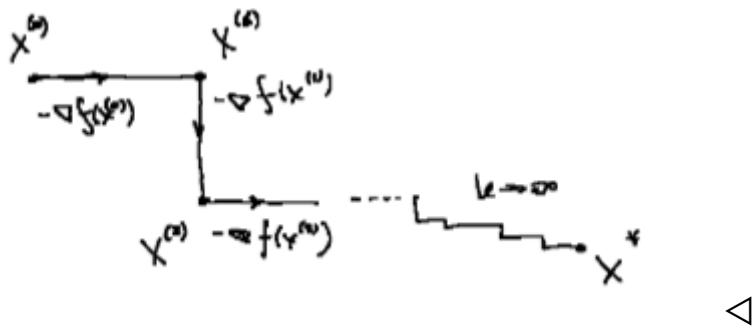
▷ Пусть $y^{(k)} \neq 0$, тогда задача $f(x^{(k+1)}) \leq f(x^{(k)})$, &&& при $\alpha^{(k)} > 0$, так как

$$\begin{aligned} \frac{dg^{(k)}(\alpha)}{d\alpha} &= \frac{d(f(x^{(k)} + \alpha y^{(k)}))}{d\alpha} = |\nabla f(x^{(k)} + \alpha y^{(k)})| = (\nabla f(x^{(k)} + \alpha y^{(k)}), \frac{d}{d\alpha}(x^{(k)} + \alpha y^{(k)})) = \\ &= (\nabla f(x^{(k)} + \alpha y^{(k)}), y^{(k)}) = |\alpha| = (\nabla f(x^{(k)}, y^{(k)})) = -(\nabla f(x^{(k)}, y^{(k)}), y^{(k)}) = -\|y^{(k)}\|_k < 0. \end{aligned}$$

Далее пусть на k -м шаге $\alpha^{(k)} > 0$, тогда:

$$\frac{dy^{(k)}(\alpha)}{d\alpha} \Big|_{\alpha=\alpha^{(k)}} = (\nabla f(x^{(k+1)}), y^{(k)}) = 0 \Rightarrow (y^{(k+1)}, y^{(k)}) = 0$$

То есть направление спуска на $(k+1)$ -й итерации ортогонально направлению спуска на k -й итерации.



Алгоритмы метода наискорейшего спуска.

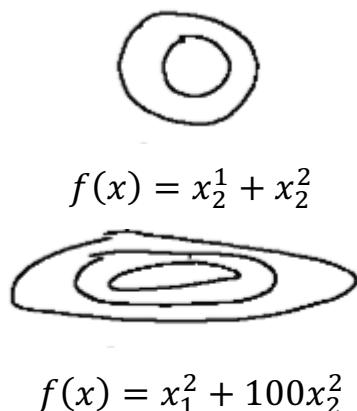
Формулы генерации последовательности $x^{(k+1)} = x^{(k)} + \alpha^{(k)}y^{(k)}, k = 0, 1, 2, \dots, y^{(k)} = -\nabla f(x^{(k)})$, тогда $\alpha^{(k)} > 0$ берем из задачи одномерной оптимизации: $g^{(k)}(\alpha) = f(x^{(k)} + \alpha y^{(k)}) \equiv f(x^{(k)} - \alpha \nabla f(x^{(k)}))$ α находится путем одномерной оптимизации $g^{(k)}(\alpha)$: $g^{(k)}(\alpha) = \min_{\alpha > 0} g^{(k)}(\alpha)$. Применяем к $g^{(k)}(\alpha)$ метод Фибоначчи, золотое сечение или метод дихотомии.

Пример 1.

Для $f(x) = f(x_1, x_2) = x_1^2 + x_2^2$ метод наискорейшего спуска сходится за 1 итерацию.

Пример 2.

Для функции вида $f(x) = x_1^2 + 100x_2^2$ сходимость будет очень медленная.



Пример 3.

Минимизируем целевую функцию: $f(x) = x_1^2 + 2x_2^2 + e^{x_1+x_2}, x \in R^2$.

Критерий остановки: $\left| \frac{df(x^{(k)})}{dx_i} \right| \leq 0.05, i = 1, 2$.

▷Шаг 0: Пусть $x^{(0)} = (0,0)^T$, тогда $y^{(0)} = -\nabla f(x^{(0)}) = \begin{pmatrix} -1 \\ -1 \end{pmatrix}$, тогда

$$g^{(0)}(\alpha) = f(0 - 1^\alpha, 0 - 1^\alpha) = 3\alpha^2 + e^{-2\alpha}, \quad \frac{dg^{(0)}(\alpha)}{d\alpha} = \frac{d}{d\alpha}(3\tilde{\alpha} + e^{-2\alpha}) = 0,$$

применим метод золотого сечения, дихотомии, метод Фибоначчи.

$$\alpha^{(0)} \approx 0.22, \text{ тогда } x^{(1)} = x^{(0)} + \alpha y^{(0)} = \begin{pmatrix} 0 \\ 0 \end{pmatrix} + 0.22 \begin{pmatrix} -1 \\ -1 \end{pmatrix} = \begin{pmatrix} -0.22 \\ -0.22 \end{pmatrix}$$

$$\text{Шаг 1: } \nabla f(x^{(1)}) = \nabla f(-0.22, -0.22) = \begin{pmatrix} 0.204 \\ -0.236 \end{pmatrix}, \text{ тогда } g^\alpha(\alpha) =$$

$$f\left(\begin{pmatrix} x_1^{(1)} \\ x_2^{(1)} \end{pmatrix} - \alpha \begin{pmatrix} y_1^{(1)} \\ y_2^{(1)} \end{pmatrix}\right) = f\left(\begin{pmatrix} -0.22 \\ -0.22 \end{pmatrix} - \alpha \begin{pmatrix} 0.204 \\ -0.236 \end{pmatrix}\right) = f(-0.22 - \alpha * 0.204, -0.22 + 0.236\alpha) = (-0.22 - 0.204\alpha)^2 + 2(-0.22 + 0.236\alpha)^2 + e^{-0.22 - \alpha * 0.204 - 0.22 + 0.236\alpha}, \text{ далее используем дихотомию, Фибоначчи, золотое сечение и находим } \alpha^{(1)}: \alpha^{(1)} = 0.32, \text{ тогда } x^{(2)} = x^{(1)} + \alpha y^{(1)} = \begin{pmatrix} -0.22 \\ -0.22 \end{pmatrix} - 0.32 * \begin{pmatrix} 0.204 \\ -0.236 \end{pmatrix} = \begin{pmatrix} -0.2853 \\ -0.1445 \end{pmatrix}.$$

Приведем условие ортогональности $(y^{(k+1)}, y^{(k)}) = (y^{(1)}, y^{(0)}) = (\begin{pmatrix} -0.22 \\ -0.236 \end{pmatrix}, \begin{pmatrix} -1 \\ -1 \end{pmatrix}) = -0.204 + 0.23 \approx 0$. \diamond

Метод сопряженных градиентов.



Метод наискорейшего спуска



Метод сопряженных градиентов

Алгоритм.

$$x^{(k+1)} = x^{(k)} - \alpha^{(k)} p^{(k)}, k = 0, 1, 2, \dots$$

$$p^{(0)} = \nabla f(x^{(0)}), p^{(k)} = \nabla f(x^{(k)}) + \beta^{(k)} p^{(k-1)}, k = 1, 2, \dots$$

a) $\alpha^{(k)} > 0$ определяется при решении задачи:

$$g^{(k)}(\alpha^{(k)}) = \min_{\alpha > 0} g^{(k)}(\alpha), g^{(k)}(\alpha) = f(x^{(k)} - \alpha p^{(k)})$$

$$6) \beta^{(k)} = \frac{\|\nabla f(x^{(k)})\|_2^2}{\|\nabla f(x^{(k-1)})\|_2^2} = \frac{\sum_{i=1}^n \frac{df(x^{(k)})}{dx_i}^2}{\sum_{i=1}^n \frac{df(x^{(k-1)})}{dx_i}^2}, k = 1, 2, \dots$$

Замечание.

Выбор вектора спуска $p^{(k)}$ в методе сопряженных градиентов зависит от значения этого вектора на предыдущем шаге.

Теорема 5.5

Пусть функция $f(x)$ ограничена снизу и её градиент $\nabla f(x)$ удовлетворяет условию Липшица с постоянной $L > 0$: $\|\nabla f(x') - \nabla f(x'')\| \leq L\|x' - x''\|$, для $\forall x', x'' \in R^n$

Тогда для последовательности $\{x^{(k)}\}$, которые получаются по $(*), (**), (***)$ справедливо равенство $\lim_{k \rightarrow \infty} \|\nabla f(x^{(k)})\| = 0$

Критерий остановки.

$$\left| \frac{df(x^{(k)})}{dx_i} \right| \leq \varepsilon_3, i = 1, 2, \dots, n; \varepsilon_3 > 0$$

или

$$\|\nabla f(x^{(k)})\| \leq \varepsilon_3, \varepsilon_3 > 0$$

Замечание.

$$p^{(0)} = \nabla f(x^{(0)});$$

$$p^{(1)} = \nabla f(x^{(0)}) + \frac{\|\nabla f(x^{(1)})\|_2^2}{\|\nabla f(x^{(0)})\|_2^2} * p^{(0)} = \nabla f(x^{(1)}) + \frac{\|\nabla f(x^{(1)})\|_2^2}{\|\nabla f(x^{(0)})\|_2^2} \nabla f(x^{(0)})$$

Вывод.

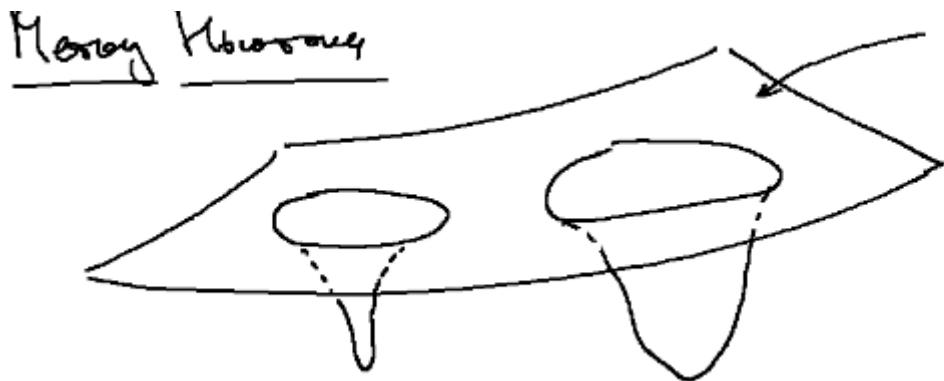
Качественно траектория получается методом сопряженных градиентов более вытянутой по сравнению с методом наискорейшего спуска.

Замечание.

Для минимизации накопления погрешности каждые N итераций получают $\beta_{mn} = 0, m = 0, 1, 2, \dots$

Производится обновление метода, N – говорят, что он называется параметром метода сопряженных градиентов.

Метод Ньютона.



Овражная версия.

Пусть $f(x)$ дважды непрерывно дифференцируема на R^n , то в окрестности $U = U(h)$ точки $x^{(k)}$: $f(x) = f^{(k)}(x) + r(x^{(k)}, h)$, где $h = x - x^{(k)}$, $\lim_{\alpha \rightarrow 0} \frac{r(x^{(k)}, h)}{\|h\|^2} = 0$, $\alpha > 0$ – шаг итерационного процесса.

$f^k(x) = f(x^{(k)}) + (\nabla f(x^{(k)}), h) + \frac{1}{2}(h, H(x^{(k)}h))$, где $H(x^{(k)})$ – матрица Гессе. Если матрица Гессе $H(x^{(k)})$ положительно определена, в качестве следующей точки $x^{(k+1)}$ выберем точку глобального минимума $f^k(x)$:

$$x^{(k+1)} = x^{(k)} - H^{-1}(x^{(k)}) * \nabla f(x^{(k)})$$

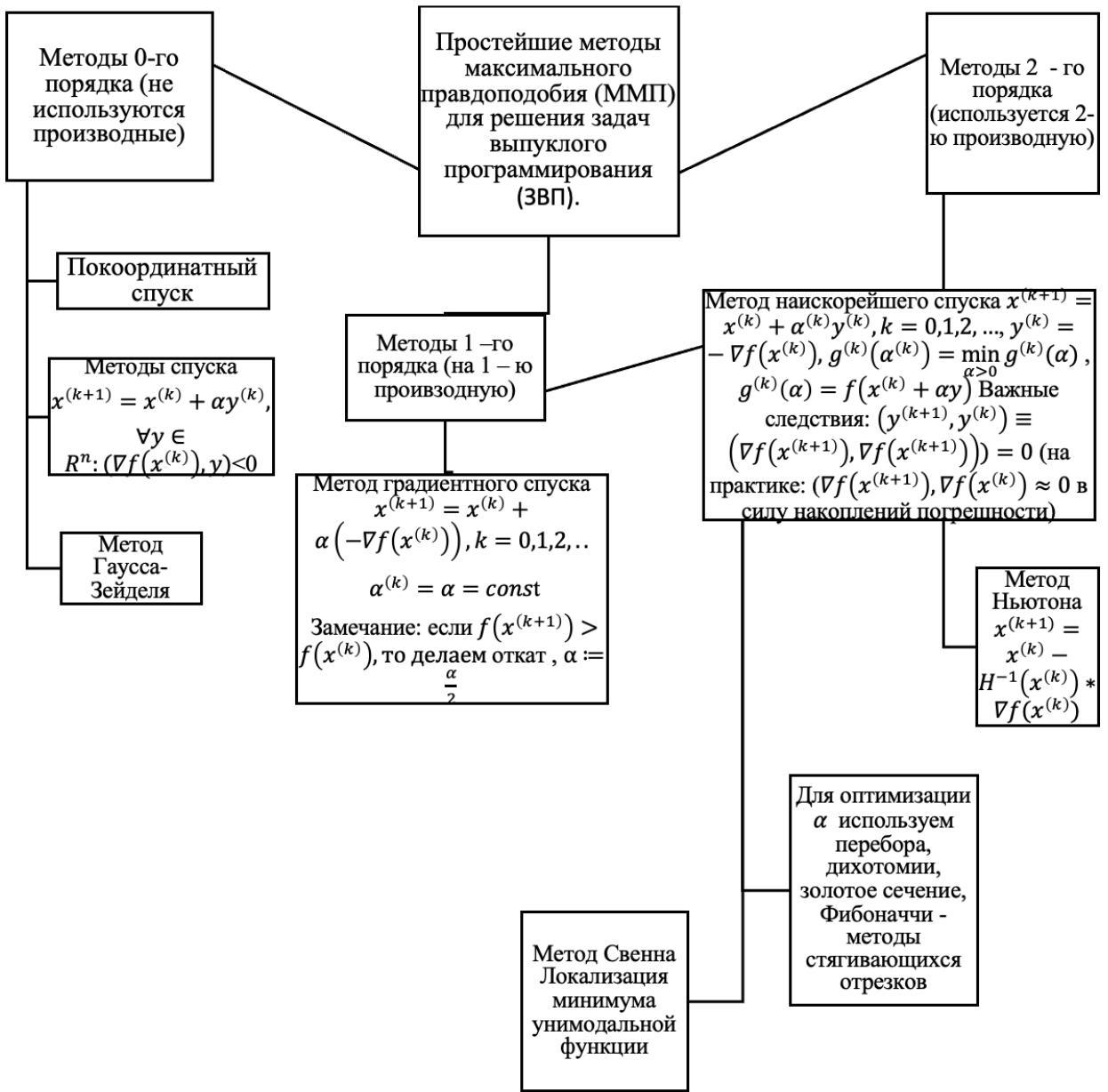
Тогда подставим $(*****)$ в $(****)$:

$$f^{(k)}(x^{(k+1)}) = f(x^{(k)}) - \frac{1}{2} \left(H^{-1}(x^{(k)}) \nabla f(x^{(k)}) \right)^2$$

Так как для $\forall y \in R^n, y \neq 0 : (y, H^{-1}y) > 0$, тогда:

$f^k(x^{(k+1)}) < f(x^{(k)})$, $h = x^{(k+1)} - x^{(k)} = -H^{-1}(x^{(k)}) * \nabla f(x^{(k)})$ – определяет направление убывания целевой функции $f(x)$, то есть :

$$f(x^{(k+1)}) - f(x^{(k)}) < 0.$$



Определение (ЗВП).

Задачи выпуклого программирования – это задачи, которые определяются следующими ограничениями:

Пусть $f(x): R^n \rightarrow R, x \in R^n, f(x) \in R, g(x): R^n \rightarrow R^m, g(x) = \begin{pmatrix} g_1(x) \\ g_2(x) \\ \dots \\ g_m(x) \end{pmatrix}$.

Решается экстремальная задача: $f(x) \rightarrow extr$, где $x = \begin{pmatrix} x_1 \\ x_2 \\ \dots \\ x_n \end{pmatrix} \in R^n$.

При этом выполняются ограничения:

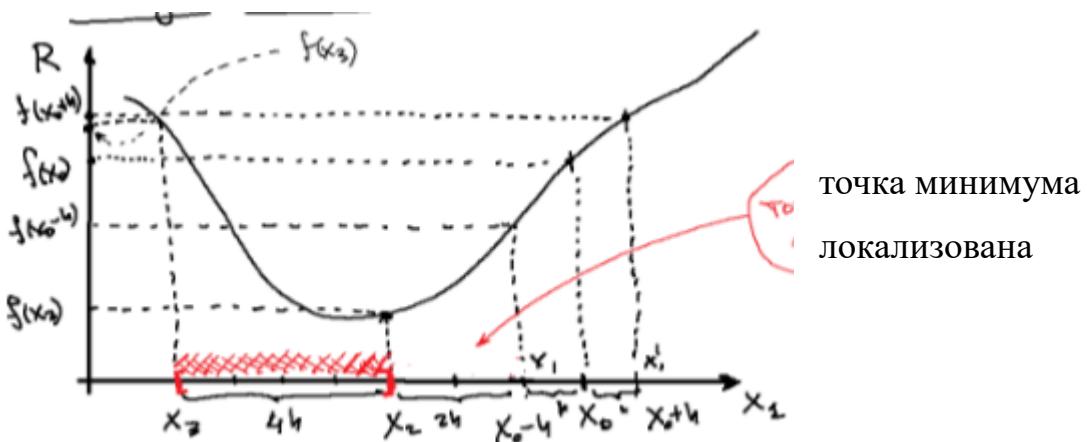
- $g_j(x) = 0, j = 1, \dots, l$
- $g_j(x) \leq 0, k = (l + 1), \dots, m,$

$$- g(x) = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ g_{l+1} \\ \vdots \\ g_m \end{pmatrix}$$

Определение (ММП).

ММП или метод наибольшего правдоподобия – это метод оценивания неизвестного параметра путем нахождения функции правдоподобия.

Метод Свенна.



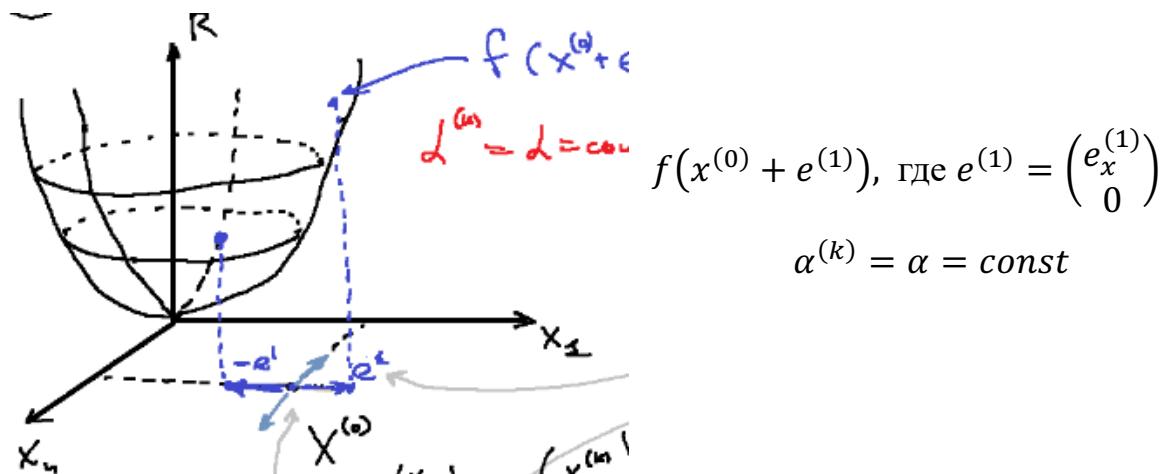
1. Выберем $\forall x_0$, зададим h , $x_1 = x_0 + h$, если $f(x_1) > f(x_0)$, меняем направление : $x_1 = x_0 - h$.
2. $x_1 = x_0 - h$, проверяем выполнение $f(x_1) < f(x_0)$ – выполнено (смотреть рисунок выше).
3. Если $f(x_{k+1}) < f(x_k)$: удваиваем шаг $h := 2h$, тогда $x_2 = x_1 + h$, если $f(x_{k+1}) < f(x_k)$, то далее генерируем $\{x_k\}$: $x_{k+1} = x_k + h, h := h * 2$.
4. В нашем примере смотреть выше $f(x_3) > f(x_2)$ ($f(x_{k+1}) > f(x_k)$), то для унимодальной $f(x)$ локализован \min на отрезке $[x_3, x_2]$.
5. Далее используем метод: Фибоначчи, золотое сечение, дихотомия.

Замечание.

Метод сопряженных градиентов не относится к простейшим методам.

Метод Гаусса – Зейделя (модификация метода покоординатного спуска).

Метод покоординатного спуска.



Алгоритм генерации релаксационной последовательности:

$$\{x^{(k)}\}: x^{(k+1)} = x^{(k)} + \alpha y^{(k)}, x^{(k)} \in R^n, y^{(k)} = \frac{x_k}{\|x^{(k)}\|} = \frac{x_k^{(k)}}{\|x^{(k)}\|} = e_k^{(k)} \quad x_k =$$

$$\begin{pmatrix} x_1 \\ \dots \\ x_k \\ \dots \\ x_n \end{pmatrix} = \begin{pmatrix} x_1^{(k)} \\ \dots \\ x^{(k)} \\ \dots \\ x_n^{(k)} \end{pmatrix}, \quad k = 1, 2, \dots, n; \quad e_k^{(k)} - \text{направление } y^{(k)}, \text{ которое выбирается}$$

вдоль координаты $x^k \equiv x_k^{(k)}$ от точки $x^{(k)}$, при этом $\alpha^{(k)} = \alpha = const, \forall k$.

Другими словами: из начальной точки делается шаг по первой переменной $x_1 \equiv x_1^{(0)}$, если шаг “удачный” и $f(x^{(k+1)}) < f(x^{(k)})$, то переходим к следующей переменной $x_2 \equiv x_2^{(0)}$, то есть: $x^{(k+1)} = x^{(k)} + \alpha * e_k^{(k)} = x^{(k)} + \alpha e_2^{(0)}, e_2^{(0)} = \frac{x_2^{(0)}}{\|x^{(0)}\|}$.

Если шаг оказался “неудачным”, то есть $f(x^{(k+1)}) > f(x^{(k)})$, то делается шаг в направлении $-e^{(k)}$, данная процедура повторяется до тех пор, пока во всех направлениях не будут получаться одни “неудачные шаги”. В этом случае

(все шаги неудачные), делаем “откат” и задаем $\alpha := \frac{\alpha}{2}$ и продолжаем алгоритм снова.

Метод Гаусса-Зейделя.

Идея! Оптимизация α на каждом шаге.

При выполнении шага по каждой переменной ищут \min целевой функции в её направлении по α , то есть алгоритм генерации релаксационной последовательности: $\{x^{(k)}\}: x^{(k+1)} = x^{(k)} + \alpha^{(k)} y^{(k)}, x^{(k)} \in R^n, y^{(k)} = \frac{x_k^{(k)}}{\|x^{(k)}\|} = e^{(k)}, k = 1, 2, \dots, n$, где $e^{(k)}$ направление $y^{(k)}$, которое выбирается вдоль

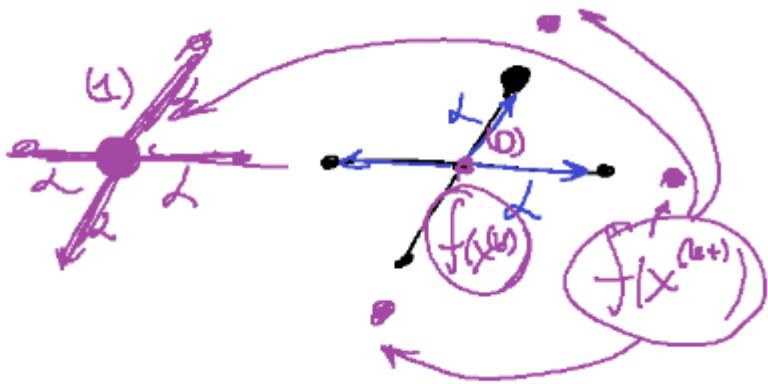
координаты x_k вектора $x^{(k)} = \begin{pmatrix} x_1 \\ \vdots \\ x_k \\ \vdots \\ x_n \end{pmatrix} \equiv \begin{pmatrix} x_1^{(k)} \\ \vdots \\ x_k^{(k)} \\ \vdots \\ x_n^{(k)} \end{pmatrix}$. При этом $\alpha^{(k)}$ выбирается

путем:

$$g^{(k)}(\alpha^{(k)}) = \min_{\alpha > 0} g^{(k)}(\alpha), g^{(k)}(\alpha) = f(x^{(k+1)}) = (x^{(k)} + \alpha e^{(k)})$$

Алгоритм метода Гаусса-Зейделя:

- Для $x^{(0)}$ фиксируют все координаты кроме одной (для определенности $x_1 \equiv x_1^{(0)}$), проводят поиски минимума: $g^{(0)}(\alpha) = f(x^{(0)} + \alpha e^{(0)}), e_1^{(0)} = \frac{x_1}{\|x^{(0)}\|} = \frac{x_1^{(0)}}{\|x^{(0)}\|}$, минимизируя $g^{(0)}(\alpha) \rightarrow \min_{\alpha > 0} \square$, получаем $\alpha^{(0)}$: $g^{(0)}(\alpha) \rightarrow \min_{\alpha > 0}$, тогда $x^{(1)} = x^{(0)} + \alpha^{(0)} * e_1^{(0)}$.
- Далее повторяем пункт 1 в новой точке, проверяя условие $f(x^k) > f(x^{k+1})$.



Замечание.

Критерий остановки $\|x^{(k)} - x^{(k-1)}\| < \varepsilon$.

Замечание.

Недостаток метода Гаусса – Зейделя – неоправданная остановка на овражных функциях.

Пример 1.

Функция Розенброка: $f(x_1, x_2) = (1 - x_1)^2 + 100(x_2 - x_1^2)^2$,
глобальный минимум $(1,1)$

Пример 2.

Функция Растигина: $f(x) = An + \sum_{i=1}^n (x_i^2 - A * \cos(2\pi x_i))$, $A = 10$, $x_i \in [-5.12, 5.12]$, глобальный минимум $x = 0, f(0) = 0$.

Лекция 4.

4.1. Три модификации метода Ньютона.

Теорема 5.6. (о методе Ньютона)

Пусть $f(x)$ дважды дифференцируема на R^n , x^* – стационарная точка $f(x)$, $H(x)$ – матрица Гессе, тогда существует окрестность x^* , что для $\forall x^{(0)}$ – начального приближения этой окрестности. Последовательность $\{x^{(k)}\}$:

$$x^{(k+1)} = x^{(k)} - H^{-1}(x^{(k)}) \cdot \nabla f(x^{(k)}), \quad k = 1, 2, \dots \quad (*)$$

сходится к x^* .

Замечание.

В условии теоремы 5.6. для $\forall x^{(0)} \in U(x^*)$ получим $\{x^{(k)}\}$ согласно формуле (*), для которой каждая предыдущая точка x^* будет удовлетворять $\nabla f(x^{(*)}) = 0$.

Замечание.

Если $x^{(0)}$ достаточно близко к точке минимума $f(x)$, то последовательность, построенная с помощью (*) будет сходиться к этой точке минимума. НО! Не зная положения точки минимума, такой выбор $x^{(0)}$ практически осуществить невозможно!

Выводы по методу Ньютона:

1. " – " – вычисление $H^{-1}(x^{(k)})$ – затратная операция, которая снижает эффективность метода Ньютона.
2. Сходимость гарантируется если $x^{(0)} \in U(x^*)$ – начальная точка достаточно близко к точке экстремума.
3. Условие остановки метода Ньютона:

$$\left| \frac{\partial f(x^{(k)})}{\partial x_i} \right| \leq \varepsilon, i = 1, 2, \dots, n \quad \text{или} \quad \|\nabla f(x^{(k)})\| \leq \varepsilon, \varepsilon > 0.$$

Метод Ньютона-Рафсона.

$$x^{(k+1)} = x^{(k)} - \alpha^{(k)} \cdot H^{-1}(x^{(k)}) \cdot \nabla f(x^{(k)}), k = 0, 1, 2, \dots$$

В качестве величины шага спуска $\alpha^{(k)} > 0$ используется решение задачи одномерной оптимизации: $g^{(k)}(\alpha^{(k)}) = \min_{\alpha>0} g^{(k)}(\alpha)$, где $g^{(k)}(\alpha) = f(x^{(k)} - \alpha \cdot H^{-1}(x^{(k)}) \cdot \nabla f(x^{(k)}))$

Теорема 5.7. (условие, гарантирующее сходимость метода Ньютона)

Пусть $f(x)$ дважды дифференцируемая функция на R^n , $\exists M$ и $\exists m : M \geq m \geq 0$, $m\|s\|^2 \leq (s, H(x) \cdot s) \leq M\|s\|^2 \quad \forall x, s \in R^n$, тогда $\{x^{(k)}\}$ полученная из (*) сходится к точке x^* независимо от выбора начального приближения $x^{(0)}$.

Метод Ньютона с возможностью не вычислять H^{-1} .

$$x^{(k+1)} = x^{(k)} - \alpha^{(k)} \cdot H^{-1}(x^{(0)}) \cdot \nabla f(x^{(k)}), k = 0, 1, 2, \dots \quad (!)$$

$H^{-1}(x^{(0)})$ – вычисляем один раз и не пересчитываем для $\forall k > 0$. Это значение будет постоянным.

В качестве величины шага $\alpha^{(k)} > 0$ используется решение задачи одномерной оптимизации: $g^{(k)}(\alpha^{(k)}) = \min_{\alpha>0} g^{(k)}(\alpha)$, где $g^{(k)}(\alpha) = f(x^{(k)} - \alpha \cdot H^{-1}(x^{(0)}) \cdot \nabla f(x^{(k)}))$.

Основная идея: нет необходимости пересчитывать $H^{-1}(x^{(k)})$ на каждом $\forall k > 0$.

Простейшие поисковые методы.

4.2. Метод обратного переменного шага.

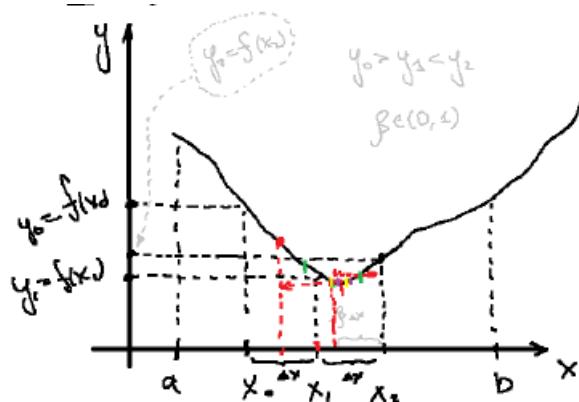


Рисунок 25 - Метод обратного переменной шага

Алгоритм.

- 1) $y_0 > y_1, x_1^{(1)} = x_1^{(0)} + \Delta_0$
- 2) $y_0 < y_1, \Delta_1 = -\beta\Delta_0$, где $\beta \in (0, 1)$
- 3) $y_0 = y_1, x^* = \frac{(x_0+x_1)}{2}$

Замечание.

Иногда $\Delta_1 = \alpha\Delta_0, \alpha > 1$ в случае, когда $y_0 > y_1$.

4.3. Метод квадратичной аппроксимации.

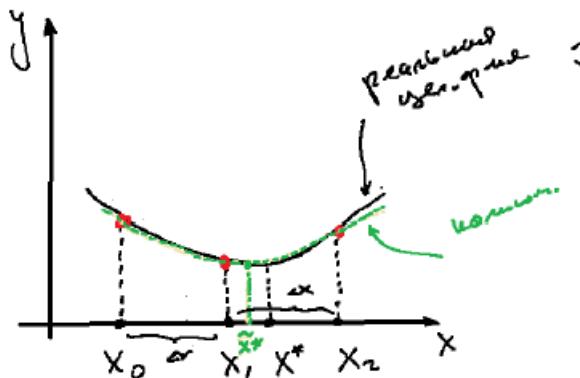


Рисунок 26 - Метод квадратичной аппроксимации

Пусть x^* — реальное значение $x \rightarrow \min f(x)$. \tilde{x}^* — приближенное значение точки минимума, полученное путем аппроксимации полиномом:

$$f(x) \approx g(x) = a_0 + (x - x_0)a_1 + (x - x_0)(x - x_1)a_2, \text{ где}$$

$$a_0 = y_0 ; \quad a_1 = \frac{y_1 - y_0}{x_1 - x_0} ; \quad a_2 = \frac{1}{x_2 - x_1} \left(\frac{y_2 - y_0}{x_2 - x_0} - \frac{y_1 - y_0}{x_1 - x_0} \right)$$

$$x^* = \frac{x_1 + x_2}{2} - \frac{a_1}{2a_2}.$$

4.4. Метод Пауэлла.

Основан на последовательном применении квадратичной аппроксимации:

1. $x_1 = x_0 + \Delta x;$
2. $y_0 = f(x_0), y_1 = f(x_1);$
3. Если $y_0 > y_1, x_2 = x_0 + 2\Delta x$, иначе $y_0 \leq y_1, x_2 = x_0 - \Delta x;$
4. $y_2 = f(x_2).$
5. Зная x_0, x_1, x_2 и y_0, y_1, y_2 , вычисляем x^* с помощью квадратичной аппроксимации.
6. Находим $y_{min} = \min(y_0, y_1, y_2)$ и x_{min} , соответствующий $y_{min}.$
7. Проверяем условие $|x_{min} - x^*| \leq \varepsilon$, ε – точность поиска. Если условие выполняется, то нашли, в противном случае – 8 шаг.
8. Выбираем «наилучшую» точку x_{min} или x^* и две точки по обе стороны и выполняем шаг 5.

4.5. Метод Хука-Дживса.

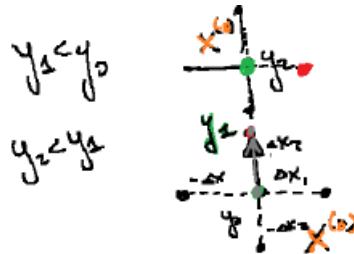


Рисунок 27 - Метод Хука-Дживса

1. выбор направления (исследующий поиск)
 - $x^{(0)}$ И Δ_i – приращение по каждой координате.

- Рассчитываем значение целевой функции при приращении по каждой координате $x_i^{(1)} = x_i^{(0)} + \Delta_i$.
- Если приращение улучшает значение целевой функции, то шаг считается удачным и дается приращение по другой координате, в противном случае – неудачным, тогда движение делается в $-\Delta_i$ Направлении.
- Если и этот шаг неудачный, то $x_i^{(0)}$ Остается неудачным и делается приращение по следующей координате, пока все координаты не будут перебраны. Таким образом, получим точку $x^{(1)}$.
- Все шаги стали «неудачными», $\Delta x_i := \frac{\Delta x_i}{2}$.

2. Движение по образцу:

$$x^{(2)} = x^{(1)} + (x^{(1)} - x^{(0)}) = 2x^{(1)} - x^{(0)}.$$

И т.д. см. пункт 1. Проверяем условие остановки.

4.6. Симплексные алгоритмы.

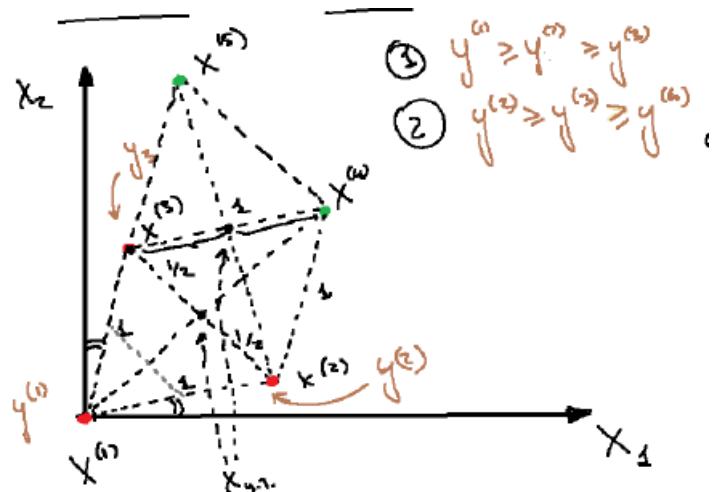


Рисунок 28 - Симплексный алгоритм

Формула:

- для текущего примера (см. рисунок 4)

$$x^{(1)} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}, x^{(2)} = \begin{pmatrix} \frac{\sqrt{3}+1}{2\sqrt{2}} \\ \frac{\sqrt{3}-1}{2\sqrt{2}} \end{pmatrix}, x^{(3)} = \begin{pmatrix} \frac{\sqrt{3}-1}{2\sqrt{2}} \\ \frac{\sqrt{3}+1}{2\sqrt{2}} \end{pmatrix}$$

$$x^{(4)} = x_{\text{центр.тяж.}} + (x_{\text{центр.тяж.}} - x^{(1)}) = x^{(2)} + x^{(3)} - x^{(1)}$$

– общая

$$x^{(1)} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix}, x^{(2)} = \begin{pmatrix} P \\ Q \\ Q \\ \vdots \\ Q \end{pmatrix}, x^{(3)} = \begin{pmatrix} Q \\ P \\ Q \\ \vdots \\ Q \end{pmatrix}, x^{(4)} = \begin{pmatrix} Q \\ Q \\ P \\ \vdots \\ Q \end{pmatrix}, \dots, x^{(n+1)} = \begin{pmatrix} Q \\ Q \\ Q \\ \vdots \\ P \end{pmatrix}$$

$$P = \frac{1}{n\sqrt{2}}(\sqrt{n+1} + n - 1)$$

$$Q = \frac{1}{n\sqrt{2}}(\sqrt{n+1} - 1)$$

4.7. Метод Нельдера-Мида (деформируемых многогранников).

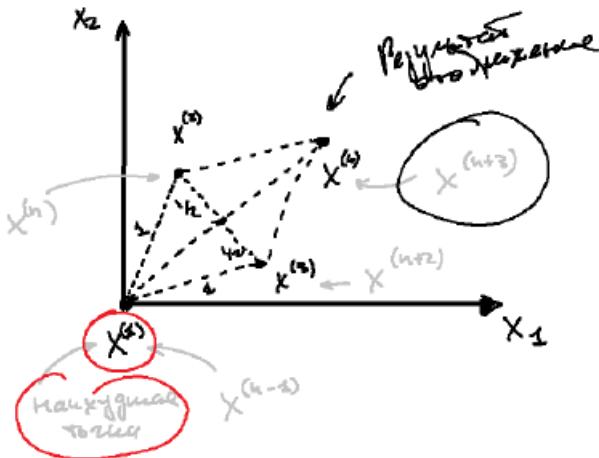


Рисунок 29

Алгоритм

1. Начальный симплекс

a. $(x^{(1)}, x^{(2)}, x^{(3)})$, где $x^{(1)}, x^{(2)}, x^{(3)} \in R^2$

2. Получим $\{x^{(1)}, x^{(2)}, x^{(3)}, x^{(4)}\}$ и $\{y^{(1)}, y^{(2)}, y^{(3)}, y^{(4)}\}$, где $y^{(i)} = f(x^{(i)})$, $i = 1, 2, \dots$. Находим $y_{min} = f(x_{min})$, $y_{max} = f(x_{max})$
3. Производим отображение «наихудшей» точки (вершины) относительно центра тяжести $x_{\text{центр.тяж.}}$:

$$x_{\text{центр.тяж.}} = \frac{1}{n} \sum_{i=2}^{n+1} x^{(i)} \stackrel{n=2}{=} \frac{1}{2} \sum_{i=2}^3 x^{(i)} = \frac{1}{2} x^{(2)} + \frac{1}{2} x^{(3)}$$

$$x^{(n+2)} = x_{\text{центр.тяж.}} + \alpha \cdot (x_{\text{центр.тяж.}} - x^{(1)}) \stackrel{n=2}{\Rightarrow} R^2$$

$$x^{(4)} = x_{\text{центр.тяж.}} + 1 \cdot (x_{\text{центр.тяж.}} - x^{(1)}) =$$

$$= \frac{1}{2} x^{(2)} + \frac{1}{2} x^{(3)} + \frac{1}{2} x^{(2)} + \frac{1}{2} x^{(3)} - x^{(1)}$$

4. Анализ результатов отображения:

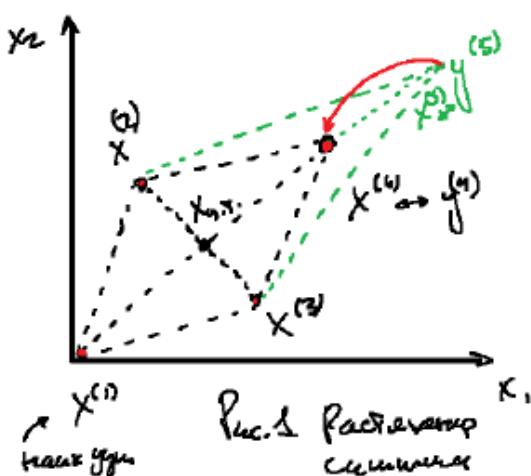


Рисунок 30 - Расжение симплекса

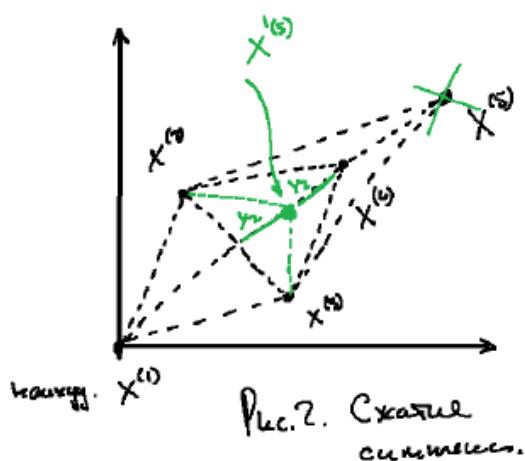


Рисунок 31 - Сжатие симплекса

a) если $y^{(n)} \equiv y^{(n+2)} < y_{min}$, растягиваем симплекс с $\beta = 2$.

a.1) если $y^{(5)} < y^{(4)} < y^{(3)} < y^{(2)} < y^{(1)}$

(или $y^{(5)} < y^{(4)} < y^{(2)} < y^{(3)} < y^{(1)}$), то $y^{(5)}$ – вершина нового симплекса. ($y^{(n+3)} < y^{(n+2)} < y^{(n+1)} < y^{(n)} < y^{(n-1)}$)

a.2) если $y^{(5)} > y^{(4)}$, $y^{(4)}$ – вершина нового симплекса.

b) если $y^{(5)} < y^{(1)}, y^{(5)} > y^{(2)}, y^{(5)} > y^{(3)}, y^{(5)} > y^{(4)}$, то производим поиск $y^{(5)}$ заново с $\beta = \frac{1}{2}$.

c) если $y^{(4)} > y^{(1)}$, т.е. $y^{(n+2)} > y_{max}$, тогда выполняем редукцию (уменьшение симплекса в 2 раза), т.е. координаты всех вершин симплекса сдвигаются на половину расстояния до «наилучшей» точки по формуле:

$$x_{\text{редукц}}^{(k)} = x_{min} + 0.5 \cdot (x^{(k)} - x_{min})$$

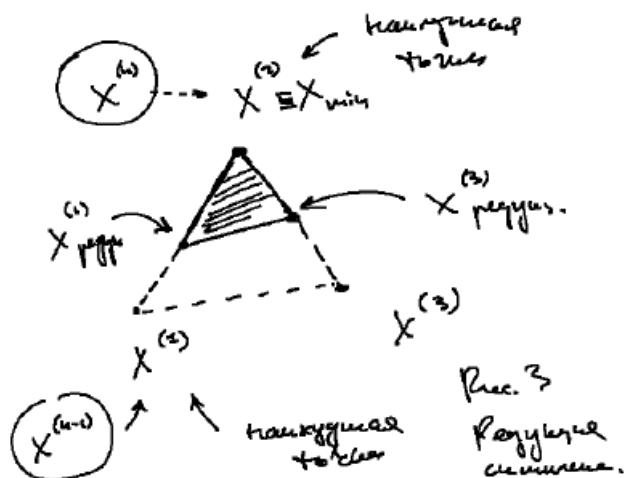


Рисунок 32 - Редукция симплекса

$$x_{\text{редукц}}^{(k)} = x_{min} + \frac{1}{2}(x^{(k)} - x_{min}) - \text{наилучшая точка}$$

Критерий остановки:

$$\sigma_f \underset{n=2}{=} \sqrt{\frac{1}{n+1} \sum_{i=1}^{n+1} [f(x^{(i)}) - f(x_{min})]^2} \leq \varepsilon, \varepsilon - \text{точность}$$

Ссылка на пошаговый алгоритм:

https://ru.wikipedia.org/wiki/Метод_Нелдера_—_Мида

Примеры реализации методов на языке Julia.

Метод Ньютона:

```
function newton(x0)
    xs = []
    x = x0
    iters = 1
    push!(xs, x)
    while true
        g = anti_gradient(f, x)
        h = inv_hessian(f, x)
        x_new = x + h * g
        push!(xs, x_new)
        if norm(g) < eps
            return x_new, iters
        end

        x = x_new
        iters += 1
    end
end
```

Метод Ньютона-Рафсона:

```
function newton_rafson(x0)
    xs = []
    x = x0
    iters = 1
    alpha = 0.0
    push!(xs, x)

    while true
        g = anti_gradient(f, x)
        h = inv_hessian(f, x)

        gfun(alp) = f(x + alpha*h * g)
        a, b = swann(gfun, 1, 0.0001)
        alpha = golden_section(gfun, a, b, 0.0001)

        x_new = x + alpha*h * g
        push!(xs, x_new)

        if norm(g) < eps
            return x_new, iters
        end

        x = x_new
        iters += 1
    end
end
```

Метод Ньютона с фиксированной H^{-1} :

```
function newton_with_fixH(x0)
    xs = []
    x = x0
    iters = 1
    alpha = 0.0
    push!(xs, x)
    h = inv_hessian(f, x)

    while true
        g = anti_gradient(f, x)
        gfun(alp) = f(x + alpha*h * g)
        a, b = swann(gfun, 1, 0.0001)
        alpha = golden_section(gfun, a, b, 0.0001)

        x_new = x + alpha*h * g
        push!(xs, x_new)

        if norm(g) < eps
            return x_new, iters
        end

        x = x_new
        iters += 1
    end
end
```

Метод квадратичной аппроксимации:

```
function square_approximation(x0, x1, x2, y0, y1, y2)
    a0 = y0
    a1 = (y1-y0) / (x1-x0)
    a2 = (1.0 / (x2-x1)) * ((y2-y0) / (x2-x0) - (y1-y0) / (x1-x0))
    x_star = (x1+x0) / 2.0 - a1 / (2.0*a2)
    return x_star
end
```

Метод Пауэлла:

```
function pauell_method(delta_x, delta_y)
    x_best = copy(x0)
    iters = 0
    xs = [x0]

    while true
        # по x
        x0 = copy(x_best)
        # x1 x2 - точки по обе стороны от "наилучшей"
        x1 = [x0[1] + delta_x, x0[2]]
        x2 = []
        y0 = f(x0)
        y1 = f(x1)
```

```

if(y0 > y1) # т.е. функция минимизируется
    x2 = x0 + [2.0 * delta_x, 0.0]
else # т.е. функция наоборот растет (неубывает)
    x2 = x0 - [delta_x, 0.0]
end

y2 = f(x2)
# вычисляем x* с помощью формулы кв. аппроксимации
x_star = [square_approximation(x0[1], x1[1], x2[1], y0,
y1, y2), x0[2]]

# находим y_min и соответствующий ему x_min
y_min = min(y0, y1, y2)
if (y_min == y0)
    x_min = copy(x0)
elseif(y_min == y1)
    x_min = copy(x1)
else
    x_min = copy(x2)
end

# выбираем наилучшую точку между x* и x_min
x_best = []
if(y_min > f(x_star))
    x_best = copy(x_star)
else
    x_best = copy(x_min)
end

# аналогично по y, тк у нас не одномерный случай
x0 = copy(x_best)
x1 = x0 + [0.0, delta_y]
x2 = []
y0 = f(x0)
y1 = f(x1)

if(y0 > y1)
    x2 = x0 + [0.0, 2.0 * delta_y]
else
    x2 = x0 - [0.0, delta_y]
end

y2 = f(x2)
x_star = [x0[1], square_approximation(x0[2], x1[2], x2[2],
y0, y1, y2), x0[2]]

y_min = min(y0, y1, y2)
if (y_min == y0)
    x_min = copy(x0)
elseif(y_min == y1)
    x_min = copy(x1)
else

```

```

        x_min = copy(x2)
    end
    if(y_min > f(x_star))
        x_best = copy(x_star)
    else
        x_best = copy(x_min)
    end

    iters += 1
    push!(xs, x_best)

    # проверяем условие
    if(norm(x_min - x_star) <= eps)
        return x_best, iters
    end
end

```

Симплексный алгоритм:

```

function simplex_method()
    x1 = [0,0]
    x2 = [(sqrt(3)+1)/(2*sqrt(2)), (sqrt(3)-1)/(2*sqrt(2))]
    x3 = [(sqrt(3)-1)/(2*sqrt(2)), (sqrt(3)+1)/(2*sqrt(2))]
    points = [x1,x2,x3]
    push!(xs, x1)
    push!(xs, x2)
    push!(xs, x3)
    while true
        if(norm(points[1]-points[2])) < 0.001
            return points[1]
        end

        if(f(points[2]) >= f(points[1]) && f(points[2]) >=
f(points[3]))
            temp = points[1]
            points[1] = points[2]
            points[2] = temp
        elseif (f(points[3]) >= f(points[1]) && f(points[3]) >=
f(points[2]))
            temp = points[1]
            points[1] = points[3]
            points[3] = temp
        end

        x4 = points[2] + points[3] - points[1]
        if(f(x4) >= f(points[2]) && f(x4) >= f(points[3]))
            points[1] = x4
            points[2] = x4+(points[2] - x4)/2
            points[3] = x4+(points[3] - x4)/2
            push!(xs, x4)
            push!(xs, points[2])
        end
    end

```

```

        push! (xs, points[3])
    else
        points[1] = x4
        push! (xs, x4)
        push! (xs, points[2])
        push! (xs, points[3])
    end
end

```

Метод Нельдера-Мида:

```

function nelder_meed()
x1 = [0.0,0.0]
x2 = [(sqrt(3)+1)/(2*sqrt(2)), (sqrt(3)-1)/(2*sqrt(2))]
x3 = [(sqrt(3)-1)/(2*sqrt(2)), (sqrt(3)+1)/(2*sqrt(2))]
points = [x1,x2,x3]
xs = []
center = [0.0,0.0]
beta = 2.0
push! (xs, x1)
push! (xs, x2)
push! (xs, x3)
while true
    points = sort(points, by=x -> f(x), rev=true)
    push! (xs, points[3])
    push! (xs, points[1])
    push! (xs, points[2])
    push! (xs, points[3])
    center = (points[2]+points[3])/2.0

    if (sqrt(((f(points[1]) - f(center))^2 + ((f(points[2]) - f(center))^2 + ((f(points[2]) - f(center))^2)) / (3.0)) < 0.001)
        return points[3],xs
    end

    x4 = points[2]+points[3]-points[1]
    beta = 2.0
    y_min = f(points[3])

    if (f(x4) < y_min)
        beta = 2.0
        x5 = beta*x4 + (1-beta)*center

        if (f(x5) < f(x4) && f(x5) < f(points[3]) && f(x5) < f(points[2]))
            points[1] = x5
        else
            if (f(x5) > f(x4))
                points[1] = x4
            end
        end
    end
else
    if f(points[3]) < f(x4) < f(points[2])

```

```

    points[1] = x4
else
    if f(points[2]) < f(x4) < f(points[1])
        points[1] = x4
    end

    points = sort(points, by=x -> f(x), rev=true)
    beta = 0.5
    x5 = beta * points[1] + (1 - beta) * center

    if f(x5) < f(points[1])
        points[1] = x5
    else
        points[1] = points[3] + 0.5 * (points[1] -
points[3])
        points[2] = points[3] + 0.5 * (points[2] -
points[3])
    end
    end
end
end

```

Лекция 5. Усовершенствование метода спуска.

Метод градиентного спуска $\alpha^{(k)} = \text{const} \rightarrow$ Метод наискорейшего спуска $\alpha^{(k)} \neq \text{const}, g^{(k)}(\alpha^{(k)}) = \min_{\alpha > 0} g^{(k)}(\alpha) \rightarrow$ Метод Флэтчера-Ривса

$$x^{(k+1)} = x^{(k)} - \alpha^{(k)} p^{(k)}, p^{(0)} = \nabla f(x^{(0)}), p^{(k)} = \nabla f(x^{(k)}) + \beta^{(k)} *$$

$$p^{(k-1)}, g^{(k)}(\alpha^{(k)}) = \min_{\alpha > 0} g^{(k)}(\alpha), \beta^{(k)} = \frac{\|\nabla f(x^{(k)})\|^2}{\|\nabla f(x^{(k-1)})\|^2} \rightarrow \text{Метод Полака} -$$

$$\text{Рибьера те же формулы, но } \beta^{(k)} = \frac{1}{\|\nabla f(x^{(k-1)})\|^2} * \left(\nabla f(x^{(k)}), (\nabla f(x^{(k)}) - \nabla f(x^{(k-1)})) \right).$$

Замечание.

Методы сопряженных градиентов сходятся в 4 раза быстрее, чем наискорейший спуск.

Пример.

$$f(x) = x_1^2 + 2x_2^2, \left| \frac{df(x^{(k)})}{dx_i} \right| \leq 0.03, i = 1, 2, \dots$$

Решение методом Флэтчера и Ривса.

Шаг 0:

$$1. \quad x^{(0)} = \begin{pmatrix} 1 \\ 1 \end{pmatrix} = \begin{pmatrix} x_1^{(0)} \\ x_2^{(0)} \end{pmatrix}$$

$$2. \quad \nabla f(x^{(0)}) = \begin{pmatrix} \frac{df(x^{(0)})}{dx_1} \\ \frac{df(x^{(0)})}{dx_2} \end{pmatrix} = \begin{pmatrix} 2x_1^{(0)} \\ 4x_2^{(0)} \end{pmatrix} = \begin{matrix} x^{(0)} = \begin{pmatrix} 1 \\ 1 \end{pmatrix} \\ \begin{pmatrix} 2 \\ 4 \end{pmatrix} \end{matrix} = p^{(0)}$$

$$x^{(1)} = x^{(0)} - \alpha p^{(0)}, g^{(0)}(\alpha) = f(x^{(0)} - \alpha p^{(0)}) = f(1 - \alpha * 2, 1 - 4 * \alpha) = (x_1^{(0)})^2 + 2(x_2^{(0)})^2 = (1 - 2\alpha)^2 + (1 - 4\alpha)^2 = 36\alpha^2 - 20\alpha + 3,$$

$$\text{оптимизируем по } \alpha: \frac{dg^{(0)}(\alpha)}{d\alpha} = 0 \Rightarrow \alpha^{(0)} = \frac{5}{18}.$$

$$x^{(1)} = x^{(0)} - \alpha^{(0)} p^{(0)} = \begin{pmatrix} 1 \\ 1 \end{pmatrix} - \frac{5}{18} * \begin{pmatrix} 2 \\ 4 \end{pmatrix} = \begin{pmatrix} \frac{4}{9} \\ -\frac{1}{9} \end{pmatrix} = \begin{pmatrix} 0.44 \\ 0.11 \end{pmatrix};$$

Шаг 1:

$$1. \quad x^{(1)} = \begin{pmatrix} 0.44 \\ 0.11 \end{pmatrix}$$

$$2. \quad p^{(1)} = \nabla f(x^{(1)}) + \beta^{(1)} p^{(0)}, \quad \beta^{(1)} = \frac{\|\nabla f(x^{(1)})\|^2}{\|\nabla f(x^{(0)})\|^2}$$

$$\nabla f(x^{(1)}) = \begin{pmatrix} \frac{df(x)}{dx_1} \\ \frac{df(x)}{dx_2} \end{pmatrix} = \begin{matrix} \text{at } x^{(1)} = \begin{pmatrix} \frac{4}{9} \\ -\frac{1}{9} \end{pmatrix} \end{matrix} \begin{pmatrix} \frac{8}{9} \\ -\frac{4}{9} \end{pmatrix},$$

$$\beta^{(1)} = \frac{\left(\frac{df(x^{(1)})}{dx_1}\right)^2 + \left(\frac{df(x^{(1)})}{dx_2}\right)^2}{\left(\frac{df(x^{(0)})}{dx_1}\right)^2 + \left(\frac{df(x^{(0)})}{dx_2}\right)^2} = \frac{\left(\frac{8}{9}\right)^2 + \left(-\frac{4}{9}\right)^2}{2^2 + 4^2} = \frac{4}{81},$$

$$p^{(1)} = \nabla f(x^{(1)}) + \beta^{(1)} * p^{(0)} = \begin{pmatrix} \frac{8}{9} \\ -\frac{4}{9} \end{pmatrix} + \frac{4}{81} * \begin{pmatrix} 2 \\ 4 \end{pmatrix} = \begin{pmatrix} \frac{80}{81} \\ -\frac{20}{81} \end{pmatrix} = \begin{pmatrix} 0.9 \\ -0.2 \end{pmatrix}$$

$$x^{(2)} = x^{(1)} - \alpha p^{(1)},$$

$$g^{(1)}(\alpha) = f(x^{(1)} - \alpha p^{(1)}) = f\left(x_1^{(1)} - \alpha p_1^{(1)}, x_2^{(1)} - \alpha p_2^{(1)}\right) = \left(\left(x_1^{(2)}\right)^2 + 2\left(x_2^{(2)}\right)\right) = \left(\frac{4}{9} - \alpha \frac{80}{81}\right)^2 + 2\left(-\frac{1}{9} - \alpha \left(-\frac{80}{81}\right)\right)^2 = \frac{800}{729}\alpha^2 - \frac{80}{81}\alpha + \frac{2}{9},$$

$$\text{Из } \frac{dg^{(1)}(\alpha)}{d\alpha} \mid \alpha = \alpha^{(0)} = 0 \Rightarrow \alpha^{(1)} = \alpha = \frac{9}{20}, \text{ тогда } x^{(2)} = x^{(1)} + \alpha^{(1)} * p^{(1)} = \begin{pmatrix} \frac{4}{9} \\ -\frac{1}{9} \end{pmatrix} - \frac{9}{20} \begin{pmatrix} \frac{80}{81} \\ -\frac{20}{81} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

$$1. \quad \nabla f(x^{(2)}) = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

Вывод:

Для сходимости метода Флэтчера-Пауэлла потребовалось 2 итерации.

Решение методом наискорейшего спуска.

$$y^{(1)} = -\nabla f(x^{(1)}) = \begin{pmatrix} -\frac{8}{9} \\ \frac{4}{9} \end{pmatrix} \approx \begin{pmatrix} -0.88 \\ 0.44 \end{pmatrix}, \text{ далее } x^{(2)} = x^{(1)} + \alpha y^{(1)}, g^{(1)}(\alpha) =$$

$$f(x^{(1)} + \alpha y^{(1)}) = f\left(\begin{pmatrix} \frac{4}{9} \\ -\frac{1}{9} \end{pmatrix} + \alpha \begin{pmatrix} -\frac{8}{9} \\ \frac{4}{9} \end{pmatrix}\right) = (x_1^{(2)})^2 + 2(x_2^{(2)})^2 = (x_1^{(1)} +$$

$$\alpha y_1^{(1)})^2 + 2(x_2^{(1)} + \alpha y_2^{(1)})^2 = \left(\frac{4}{9} + \alpha \left(-\frac{8}{9}\right)\right)^2 + 2\left(-\frac{1}{9} + \alpha \left(\frac{4}{9}\right)\right)^2 = \frac{2}{9} - \frac{80}{81}\alpha + \frac{32}{27}\alpha^2$$

$$\frac{dg^{(1)}(\alpha)}{d\alpha} = \frac{d}{d\alpha} \left(\frac{32}{27}\alpha^2 - \frac{80}{81}\alpha + \frac{2}{9} \right) = 0 \Rightarrow \alpha = \frac{5}{12} \approx 0.4166(6)$$

$$x^{(2)} = x^{(1)} + \alpha y^{(1)} = \begin{pmatrix} \frac{4}{9} \\ -\frac{1}{9} \end{pmatrix} + \frac{5}{12} \begin{pmatrix} -\frac{8}{9} \\ \frac{4}{9} \end{pmatrix} = \begin{pmatrix} \frac{2}{27} \\ \frac{2}{27} \end{pmatrix} \approx \begin{pmatrix} 0.1 \\ 0.1 \end{pmatrix}$$

Таким образом, $x_{\text{наискорейший спуск}}^{(2)} = \begin{pmatrix} 0.1 \\ 0.1 \end{pmatrix}; x_{\text{Флэтчера-Ривса}}^{(2)} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$

$$\nabla f(x^{(2)}) = \begin{pmatrix} \frac{df(x)}{dx_1} \\ \frac{df(x)}{dx_2} \end{pmatrix} = \begin{pmatrix} 2x_1 \\ 4x_2 \end{pmatrix} = \begin{matrix} x_1 = x_1^{(2)} = \frac{3}{27} \\ x_2 = x_2^{(2)} = \frac{8}{27} \end{matrix} \begin{pmatrix} \frac{4}{27} \\ \frac{8}{27} \end{pmatrix} = \begin{pmatrix} 0.15 \\ 0.3 \end{pmatrix}, \text{ но условие остановки}$$

$\left| \frac{df(x)}{dx_i} \right| \leq 0.03 \forall i = 1, 2, \dots$, таким образом итерационный процесс будет продолжаться.

Геометрическая интерпретация метода сопряженных градиентов.

Определение.

Два вектора $x^{(i)}$ и $x^{(j)}$ называются A-сопряженными, если скалярное произведение $(x^{(i)}, Ax^{(j)}) = 0$

Замечание.

Сопряженность можно рассматривать, как обобщение понятия ортогональности.

Замечание.

Для метода наискорейшего спуска $A=E$

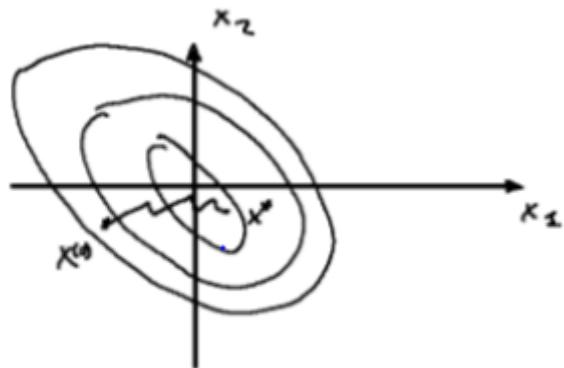


Рисунок 1. Траектория движения в точку минимума методом наискорейшего спуска

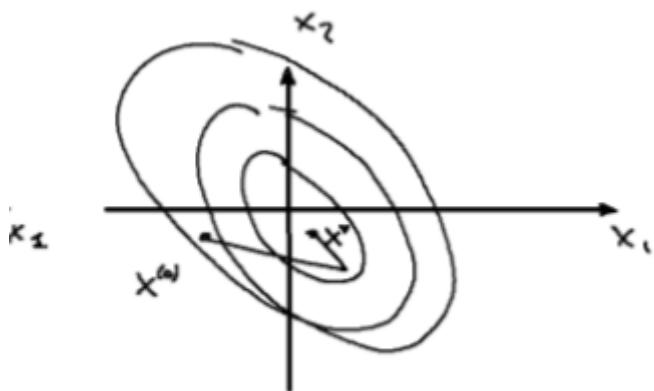


Рисунок 2. Траектория движения в точку минимума методом сопряженных градиентов

Замечание.

Можно по-другому продемонстрировать взаимосвязь понятий ортогональности и сопряженности: мысленно растянем рисунок 2 таким образом, чтобы линии уровня из эллипсов превратились в окружность, при этом сопряженные направления станут ортогональными.

Определение.

Квадратичная функция некоторого вектора: $f(x) = \frac{1}{2}(x, Ax) - (b, x) + c$, где $x \in R^n, A \in R^{n \times n}, b \in R^n, c \in R$.

Замечание.

Для квадратичных функций метод сопряженных градиентов сходится за n шагов, где n -размерность пространства.

Замечание: Для функций общего вида алгоритм не является конечным.

Формула Полака-Рибьера:

$$\beta^{(n)} = \frac{(\nabla f(x^{(n)}), (\nabla f(x^{(n)}) - \nabla f(x^{(n-1)})))}{\|\nabla f(x^{(n-1)})\|^2}$$

Замечание.

Метод Флэтчера-Ривса сходится, если $x^{(0)}$ достаточно близко к x^* (требуемый минимум), при этом метод Полака-Рибьера может в различных случаях бесконечно зацикливаться. Для избавления зацикливания необходимо “перезагружать” данный метод каждые $n+1$ итераций, $\beta^{(n)} = 0$, $k = n + 1$.

Замечание.

Метод Полака-Рибьера сходится, как правило, быстрее, чем метод Флэтчера – Ривса.

Замечание.

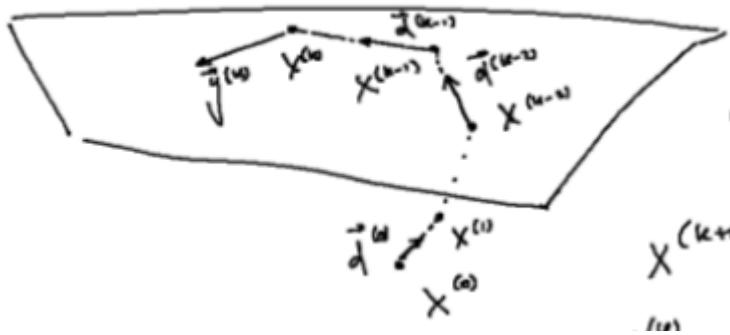
Если $\beta^{(n)} \leq 0$ на k – й итерации метода Полака-Рибьера, то метод необходимо “перезагрузить”.

Замечание.

Для обоих методов на каждом шаге необходимо проводить одномерную оптимизации $g^{(k)}(\alpha^{(k)}) = f(x^{(k)} - \alpha p^{(k)})$

Различные виды методов сопряженных градиентов.

Общий вид метода сопряженных градиентов:



Должно выполняться условие A-сопряжения: $(d^{(k+1)}, Ad^{(k)}) = 0$

$$x^{(k+1)} = x^{(k)} + \alpha^{(k)} d^{(k)}$$

$$d^{(k)} = -y^{(k)} + \sum_{i=0}^{k-1} \beta_i d^{(i)}$$

$$y^{(k)} = \nabla f(x^{(k)}), d^{(0)} = -\nabla f(x^{(0)})$$

Метод Хестенса-Стифеля:

$$\begin{aligned} \beta^{(k)} &= -\frac{(-\nabla f(x^{(k)}), (\nabla f(x^{(k)}) - \nabla f(x^{(k-1)})))}{(d^{(k-1)}, (\nabla f(x^{(k)}) - \nabla f(x^{(k-1)})))} \\ d^{(k)} &= -\nabla f(x^{(k)}) + \beta^{(k)} * d^{(k-1)} \end{aligned}$$

Метод Диксона:

$$\beta^{(k)} = -\frac{(\nabla f(x^{(k)}), \nabla f(x^{(k)}))}{(d^{(k-1)}, \nabla f(x^{(k-1)}))}$$

Метод Дайяна:

$$\beta^{(k)} = -\frac{(\nabla f(x^{(k)}), (\nabla f(x^{(k)}) - \nabla f(x^{(k-1)})))}{(d^{(k-1)}, (\nabla f(x^{(k)}) - \nabla f(x^{(k-1)})))}$$

Методы сопряженных градиентов (простейшие методы 1-го порядка).

1. Флэтчера-Ривса:

$$x^{(k+1)} = x^{(k)} - \alpha^{(k)} p^{(k)}, k = 0, 1, 2, \dots$$

$$p^{(0)} = \nabla f(x^{(0)}), p^{(k)} = \nabla f(x^{(k)}) + \beta^{(k)} * p^{(k-1)}, k = 1, 2, \dots$$

2. Полака-Рибьера:

$$x^{(k+1)} = x^{(k)} - \alpha^{(k)} p^{(k)}, k = 0, 1, 2, \dots$$

$$p^{(0)} = \nabla f(x^{(0)}), p^{(k)} = \nabla f(x^{(k)}) + \beta^{(k)} * p^{(k-1)}, k = 1, 2, \dots$$

$$\beta^{(k)} = \frac{(\nabla f(x^{(k)}), (\nabla f(x^{(k)}) - \nabla f(x^{(k-1)})))}{\|\nabla f(x^{(k-1)})\|^2}$$

3. Диксона:

$$d^{(k)} = -y^{(k)} + \sum_{i=0}^{n-1} \beta^{(i)} d^{(i)}, y^{(k)} = \nabla f(x^{(k)}), d^{(0)} = \nabla f(x^{(0)})$$

$$x^{(k+1)} = x^{(k)} + \alpha^{(k)} * d^{(k)}, \beta^{(i)} = -\frac{(\nabla f(x^{(i)}), \nabla f(x^{(i)}))}{(d^{(i-1)}, \nabla f(x^{(i-1)}))}$$

4. Дайяна:

$$d^{(k)} = -y^{(k)} + \sum_{i=0}^{n-1} \beta^{(i)} d^{(i)}, y^{(k)} = \nabla f(x^{(k)}), d^{(0)} = \nabla f(x^{(0)})$$

$$x^{(k+1)} = x^{(k)} + \alpha^{(k)} * d^{(k)}, \beta^{(i)} = \frac{(\nabla f(x^{(i)}), (\nabla f(x^{(i)}) - \nabla f(x^{(i-1)})))}{(d^{(i-1)}, (\nabla f(x^{(i)}) - \nabla f(x^{(i-1)})))}$$

5. Хестенса – Стифеля:

$$d^{(k)} = -\nabla f(x^{(k)}) + \beta^{(k)} * d^{(k-1)}, y^{(k)} = \nabla f(x^{(k)}), d^{(0)} = \nabla f(x^{(0)})$$

$$x^{(k+1)} = x^{(k)} + \alpha^{(k)} * d^{(k)}, \beta^{(i)} = -\frac{(\nabla f(x^{(k)}), (\nabla f(x^{(k)}) - \nabla f(x^{(k-1)})))}{(d^{(k-1)}, (\nabla f(x^{(k)}) - \nabla f(x^{(k-1)})))}$$

Многокритериальный поиск.

Метод предложен Милем и Контреллом. (*)

$$x^{(k+1)} = x^{(k)} - \lambda_0^{(k)} + \nabla f(x^{(k)}) + \lambda_1^{(k)} \Delta x^{(k-1)}, \Delta x^{(k-1)} = x^{(k)} - x^{(k-1)}$$

На каждом шаге:

$$g^{(k)}(\lambda_0^{(k)}, \lambda_1^{(k)}) = \min_{\lambda_0, \lambda_1} f(x^{(k)} - \lambda_0 \nabla f(x^{(k)}) + \lambda_1 * \Delta x^{(k-1)}),$$

Далее находим очередное приближение по (*).

Замечание.

$$(\nabla f(x^{(k)}), \nabla f(x^{(k+1)})) = 0, (\nabla f(x^{(k+1)}), \Delta x^{(k+1)}) = 0 \text{ и } (\nabla f(x^{(k+1)}), \Delta x^{(k)}) = 0$$

При реализации данного алгоритма помогают: $\Delta x^{(k-1)} =_{k=0} 0$, а $x^{(0)}$ – должно быть задано, тогда на k – м шаге:

1. Вычисляем $x^{(k)}, \nabla f(x^{(k)})$ и $\Delta x^{(k-1)} = x^{(k)} - x^{(k-1)}$.
2. Минимизируем $g^{(k)}(\lambda_0, \lambda_1)$ одним из методов, например, Ньютона с точностью e_{λ_0} и e_{λ_1} , получаем $x^{(k+1)}$ и переходим к пункту 1.
3. Каждый $n+1$ шаг начинаем $\Delta x^{(k-1)} = 0$, где n – это число линейно – независимое направлении поиска.
4. Критерий остановки итерационного процесса $|\nabla f(x^{(k)})| < \varepsilon$.

Замечание.

На квадратичных функциях алгоритм по эффективности близок к методу сопряженных градиентов.

Метод Крэчти и Лева.

$$x^{(k+1)} = x^{(k)} - \lambda_0 \nabla f(x^{(k)}) + \sum_{i=1}^m \lambda_i * \nabla x^{(i-1)},$$

$$m \leq n - 1, \text{ таким образом на каждом шаге: } g^{(k)}(\lambda_0^{(k)}, \lambda_1^{(k)}, \dots, \lambda_m^{(k)}) = \min_{\lambda_0, \lambda_1, \dots, \lambda_m} f(x^{(k)} - \lambda_0 * \nabla f(x^{(k)}) + \sum_{i=1}^m \lambda_i * \Delta x^{(i-1)}).$$

Методы второго порядка. Метод Ньютона. Квазиньютоновские методы.

Идея: При поиске минимума используют информацию о функции и её производных до второго порядка включительно.

Разложим $f(x)$ в ряд Тейлора: $f(x) \approx f(x^k) + (\nabla f(x^k), (x - x^k)) + \frac{1}{2} ((x - x^k), \nabla^2 f(x^k)(x - x^k)), (1)$ где $\nabla^2 f(x^k) = H(x^k)$ – матрица Гессе,

$$H(x^k) = \begin{vmatrix} \frac{d^2 f(x^k)}{dx_1^2} & \frac{d^2 f(x^k)}{dx_1 dx_2} & \dots \\ \frac{d^2 f(x^k)}{dx_2 dx_1} & \ddots & \vdots \\ \ddots & \vdots & \ddots \\ \frac{d^2 f(x^k)}{dx_n dx_1} & \vdots & \ddots \end{vmatrix}$$

Направление поиска s выберем:

В (1) положим следующее $x = x^{(k+1)}, \Delta x^{(k)} = x^{(k+1)} - x^{(k)}$, тогда (1) имеет вид: $f(x^{(k+1)}) \approx f(x^k)(\nabla f(x^k), \Delta x^{(k)}) + \frac{1}{2}(\Delta x^{(k)}, \nabla^2 f(x^k) \Delta x^{(k)})$ (2).

Минимум $f(x)$ по направлению $\Delta x^{(k)}$ определяется дифференцированием по каждой компоненте вектора Δx и приравниванием к 0:

$\nabla f(x^k) + (\nabla^2 f(x^k), \Delta x^{(k)}) = \vec{0} \Leftrightarrow \nabla f(x^k) + H(x^k) \Delta x^{(k)} = 0$, умножим уравнение на H^{-1} :

$H^{-1}(x^k) \nabla f(x^k) + \Delta x^{(k)} = 0 \Rightarrow \Delta x^{(k)} \equiv S^{(k)} = H^{-1}(x^k) \nabla f(x^k) = -[\nabla^2 f(x^k)]^{-1} \nabla f(x^k)$, где $[\nabla^2 f(x^k)]^{-1}$ – обратная матрица Гессе.

$x^{(k+1)} = x^{(k)} - H^{-1}(x^k) \nabla f(x^k)$ (5).

Замечание 1.

Если $f(x)$ – квадратичная функция (выпуклая вниз), то для достижения минимума достаточно одного шага.

Тут Линии уровня квадратичной функции.

$$\Delta x_2^{(0)} = -[\nabla^2 f(x_2^{(0)})]^{-1} \nabla f(x_2^{(0)}),$$

$$\Delta x_3^{(0)} = -[\nabla^2 f(x_3^{(0)})]^{-1} \nabla f(x_3^{(0)})$$

$$\nabla f(x^k) = f(x_1, x_2) = x_1^2 + x_2^2$$

$$\nabla f(x) = \begin{pmatrix} \frac{df(x)}{dx_1} \\ \frac{df(x)}{dx_2} \end{pmatrix} = \begin{pmatrix} 2x_1 \\ 2x_2 \end{pmatrix},$$

$$H = \begin{vmatrix} \frac{df^2(x)}{dx_1^2} & \frac{df^2(x)}{dx_1 dx_2} \\ \frac{df^2(x)}{dx_2 dx_1} & \frac{df^2(x)}{dx_2^2} \end{vmatrix} = \begin{vmatrix} 2 & \square \\ \square & 2 \end{vmatrix}$$

Замечание.

В общем случае для нелинейной целевой функции $f(x)$ минимум за один шаг не находится и (5) имеет вид:

$$x^{(k+1)} = x^{(k)} - \lambda^{(k)} \frac{H^{-1}(x^k) \nabla f(x^k)}{\|H^{-1}(x^k) \nabla f(x^k)\|}, \quad \lambda^{(k)} - \text{пример длины шага.}$$

Замечание.

В одномерном случае это был бы метод Ньютона-Рафсона.

Замечание.

Другой вид итерационной формулы: $x^{(k+1)} = x^{(k)} - \lambda^{(k)} [\nabla^2 f(x^k)]^{-1} \nabla f(x^k) = x^{(k)} - \lambda^{(k)} H^{-1}(x^k) \nabla f(x^k)$ (7).

Тогда направление спуска имеет вид: $S^{(k)} = -H^{-1}(x^k) \nabla f(x^k)$.

Критерий остановки метода Ньютона: $\|\Delta x^{(k)}\| < \epsilon_1$ или $\|\nabla f(x^k)\| < \epsilon_2$.

Замечание.

Условиями, гарантирующими сходимость метода Ньютона, являются:

1. $f(x)$ – дважды непрерывно дифференцируема.
2. $H^{-1}(x)$ – должна быть положительно определена.

Замечание.

Операция выполнения $H^{-1}(x)$ на каждом k-ом шаге бывает затратной операцией. Иногда используется $H^{-1}(x^{(0)})$ на всех дальнейших интервалах $k = 1, 2, \dots$, $x^{(k+1)} = x^{(k)} - \lambda^{(k)} H^{-1}(x^0) \nabla f(x^k)$.

В отличие от метода наискорейшего спуска с линейной скоростью сходимости, метод Ньютона обладает квадратичной скоростью сходимости.

Применение метода Ньютона очень эффективно в случае выполнения необходимых и достаточных условий его сходимости.

Исследование необходимых и достаточных условий метода Ньютона в конкретных практических случаях может быть сложной задачей.

Методы переменной метрики или квазиньютоновские методы.

Идея:

- A) аппроксимация матрицы Гессе.
- Б) Для аппроксимации используется только первые производные.
- В) Очередное приближение находится по формуле:

$$x^{(k+1)} = x^{(k)} - \lambda^{(k)} S^{(k)} = x^{(k)} - \lambda^{(k)} \eta(x^{(k)}) \nabla f(x^{(k)}) \quad (1)$$

где $\eta(x^{(k)})$ – матрица направлений, представляющая из себя аппроксимацию обратной матрицы Гессе.

Для квадратичной целевой функции:

$$f(x) \approx f(x^k) + (\nabla f(x^k), (x - x^k)) + \frac{1}{2} ((x - x^k), \nabla^2 f(x^k) (x - x^k)),$$

$\nabla^2 f(x^k) = H(x^k)$, положим $x = x^{(k+1)}$, продифференцируем эту целевую функцию:

$$\begin{aligned} \nabla f(x^{(k+1)}) &= \nabla f(x^k) + H(x^k)(x^{(k+1)} - x^{(k)}) \\ \nabla f(x^{(k+1)}) - \nabla f(x^k) &= H(x^k)(x^{(k+1)} - x^{(k)}) \end{aligned}$$

Домножим на $H^{-1}(x^k)$:

$$H^{-1}(x^k)(\nabla f(x^{(k+1)}) - \nabla f(x^k)) = E(x^{(k+1)} - x^{(k)}) \Rightarrow$$

$$x^{(k+1)} - x^{(k)} = H^{-1}(x^k)(\nabla f(x^{(k+1)}) - \nabla f(x^k)) \quad (2)$$

Если $f(x)$ квадратичная целевая функция, то $H(x^k) = const.$

В основном $H^{-1}(x^{(k+1)}) \approx \omega \eta(x^{(k)}) = \omega(\eta^{(k)} + \Delta \eta^{(k)})$ (3), где $\eta^{(k)}$ – матрица приближения Гессиана, $\Delta \eta^{(k)}$ – прирост приближения Гессиана.

$\eta^{(k)} \equiv \eta(x^{(k)})$ – матрица аппроксимации $H^{-1}(x^{(k)})$ на предыдущем k-ом шаге.

Замечание.

$\Delta\eta^{(k)}$ – некоторая вычисляемая на каждом шаге матрица. ω – множитель, как правило равен 1.

Замечание.

Выбор метода вычисления $\Delta\eta^{(k)}$ на k-ом шаге и определяет соответствующий метод переменной метрики.

Замечание.

А) Для обеспечения сходимости метода $\omega\eta^{(k+1)}$ должна быть такой положительно определенной.

Б) На k+1-ом шаге знаем: $x^{(k)}, \nabla f(x^k), (\nabla f(x^{(k+1)})$ и $\eta^{(k)}$, тогда необходимо вычислить $\eta^{(k+1)}$ (должна удовлетворять (2) с учетом (3)): $\Delta x^{(k)} = \omega\eta^{(k+1)}(\nabla f(x^{(k+1)}) - \nabla f(x^k)) = \omega\eta^{(k+1)}\Delta g^{(k)}$, тогда $\eta^{(k+1)}\Delta g^{(k)} = \frac{1}{\omega}\Delta x^{(k)}$ (4), так как $\eta^{(k)} = \eta^{(k)} + \Delta\eta^{(k)}$, то $\Delta\eta^{(k+1)}\Delta g^{(k)} = \frac{1}{\omega}\Delta x^{(k)} - \eta^{(k)}\Delta g^{(k)}$ (5).

Тогда разрешим (5) относительно $\Delta\eta^{(k)}$:

$$\Delta\eta^{(k)} = \frac{\frac{1}{\omega}(\Delta x^{(k)}y^T)}{(y, \Delta g^{(k)})} - \frac{\eta^{(k)}\Delta g^{(k)}z^T}{(z, \Delta g^{(k)})},$$

где z и y – произвольные векторы размерности n.

Метод Бройдена.

Бройден показал, что если $\Delta\eta^{(k)}$ является симметричной матрицей с рангом 1 и выполняется соотношение $\eta^{(k+1)}\Delta g^{(k)} = \Delta x^{(k)}$, где $\Delta x^{(k)} = x^{(k+1)} - x^{(k)}$, $\Delta g^{(k)} = \nabla f(x^{(k+1)}) - \nabla f(x^k)$, единственным возможным выбором $\Delta\eta^{(k)}$:

$$\Delta\eta^{(k)} = \frac{[\Delta x^{(k)} - \eta^{(k)}\Delta g^{(k)}][\Delta x^{(k)} - \eta^{(k)}\Delta g^{(k)}]^T}{[\Delta x^{(k)} - \eta^{(k)}\Delta g^{(k)}]^T\Delta g^{(k)}}$$

В знаменателе скалярное произведение.

Реализация алгоритма:

1. Задается начальное приближение $x^{(0)}$ и положительно определённая матрица $\eta^{(0)} = E$.
2. Вычислим:
 - $x^{(k+1)} = x^{(k)} - \lambda^{(k)}\eta(x^{(k)})\nabla f(x^{(k)})$,
 - $\lambda^{(k)} = \arg \min_{\lambda} f(x^{(k)} - \lambda\eta(x^{(k)}))\nabla f(x^{(k)})$
3. Находим очередное приближение матрицы: $\eta^{(k+1)} = \eta^{(k)} + \Delta\eta^{(k)}$, где
4. $\Delta\eta^{(k)} = \frac{[\Delta x^{(k)} - \eta^{(k)}\Delta g^{(k)}][\Delta x^{(k)} - \eta^{(k)}\Delta g^{(k)}]^T}{(\Delta x^{(k)} - \eta^{(k)}\Delta g^{(k)}, \Delta g^{(k)})}$.
5. Критерий остановки: $\|\Delta x^{(k)}\| < \epsilon_2$ или $\|\nabla f(x^k)\| < \epsilon_1$, если критерий остановки не выполнен, то переходим к 2.

Замечание.

Если целевая функция является квадратичной, то направление поиска $S^{(k)} = -\eta^{(k)}\nabla f(x^k)$ на каждой итерации оказываются сопряженными и для определения минимума оказывается достаточным сделать n шагов.

В случае оптимизации неквадратичной функции возможны нежелательные явления:

1. $\eta^{(k+1)}$ – может перестать быть положительно определенной, в этом случае её необходимо каким-то образом сделать положительно определенной;
2. $\Delta\eta^{(k)}$ – в силу ошибок округления может стать неограниченной величиной;
3. $\Delta x^{(k)} = \lambda^{(k)}\eta^{(k)}\nabla f(x^{(k)})$ может случайно на текущем шаге совпасть с направлением на предыдущем, то $\eta^{(k+1)}$ может стать сингулярным или неопределенной.

Замечание.

Чтобы избежать этих явлений необходимо «обнулять» алгоритм после каждой n итераций, считая n+1 итерацию начальной.

Лекция 6. Метод Дэвидона-Флэтчера-Пауэла (Д.Ф.П.)

Замечание.

Метод Брайдена: $\nabla\eta^{(*)} \rightarrow$ ранг 1.

Метод Д.Ф.П.: $\nabla\eta^{(*)} \rightarrow$ ранг 2.

Замечание.

$\eta^{(0)} = E$ – начальная матрица.

Формула метода Д.Ф.П.

$$\eta^{(k)} = \frac{1}{\omega} \cdot \frac{\Delta x^{(k)} \otimes y}{(y, \Delta g^{(k)})} - \frac{\eta^{(k)} (\Delta g^{(k)} \otimes z)}{(z, \Delta g^{(k)})}, \text{ где}$$

в методе Д.Ф.П.: $\omega = 1$, $y = \Delta x$, $z = \eta^{(k)} \cdot \Delta g^{(k)}$, \otimes – внешнее произведение, $\eta^{(k)}, \Delta x^{(k)}, (\Delta g^{(k)} \otimes z)$ – матрицы, тогда

$$\eta^{(k+1)} = \eta^{(k)} + A^{(k)} - B^{(k)} = \eta^{(k)} + \frac{(\Delta x^{(k)} \otimes \Delta x^{(k)})}{(\Delta x^{(k)}, \Delta g^{(k)})} - \frac{\eta^{(k)} \cdot (\Delta g^{(k)} \otimes \Delta g^{(k)}) \cdot (\eta^{(k)})^T}{(\Delta g^{(k)}, \eta^{(k)} \cdot \Delta g^{(k)})}$$

Замечание.

$A^{(k)}$ и $B^{(k)}$ – симметричные, так что $\eta^{(k)}$ симметричная и $\eta^{(k+1)}$ – симметричная.

Замечание.

Метод Д.Ф.П. является одним из наиболее эффективных алгоритмов переменной метрики.

Замечание.

Условия эффективности метода Д.Ф.П.:

- ошибки округления $\nabla f(x^{(*)})$ – невелики;
- $\eta^{(k)}$ в процессе вычислений не «портится» (положительно определена $\forall k$).

Замечание.

В методе Д.Ф.П. в ходе оптимизации происходит переход от градиентного направления спуска к ньютоновскому.

Замечание.

- a) Роль $A^{(k)}: \eta^{(k)} \rightarrow H^{-1}$
- б) Роль $B^{(k)}$: обеспечение положительной определенности матрицы $\eta^{(k+1)}, \forall k$
- в) В пределе сумма $A^{(k)}$ и $B^{(k)}$ исключает начальную матрицу $\eta^{(0)}$

Таким образом, процесс выглядит следующим образом:

$$\begin{aligned} \eta^{(0)} &= E; \\ \eta^{(1)} &= E + A^{(0)} - B^{(0)}; \\ \eta^{(2)} &= \eta^{(1)} + A^{(1)} - B^{(1)} = E + (A^{(0)} + A^{(1)}) - (B^{(0)} + B^{(1)}); \\ &\dots \\ \eta^{(k+1)} &= E + \sum_{i=0}^k A^{(i)} - \sum_{i=0}^k B^{(i)}. \end{aligned}$$

Замечание.

В случае квадратичной целевой функции при $k = n - 1$, должно выполняться: $H^{-1} = \sum_{i=0}^{n-1} A^{(i)}$, а сумма матриц $\sum_{i=0}^{k=n-1} B^{(i)}$ строится таким образом, что она сокращается с начальным значением $\eta^{(0)}$.

Замечание.

Для квадратичной целевой функции направления поиска являются сопряженными.

Метод BFGS (Бройдена, Флэтчера, Гольдфарба, Шанно).

B – Broyden, F – Fletcher, G – Goldfarb, S – Shanno

Считается самым оптимальным и эффективным методом:

SciPy: `scipy.optimize.BFGS`



Описание алгоритма.

Пусть задана целевая функция $f(x): x \rightarrow R, x \in R^n$, решаем задачу многомерной оптимизации.

1. Выбираем $x^{(0)}$ – начальную точку приближения, задаем точность $\varepsilon > 0$, вычисляем $H^{-1}(x^{(0)}) = [\nabla^2 f(x)]^{-1}$ либо $\eta^{(0)} = E$ (единичная матрица, которая хорошо обусловлена и невырождена)
2. Вектор направления поиска: $p^{(k)} = -H^{-1}(x^{(k)}) \cdot \nabla f(x^{(k)})$, где $H^{-1}(x^{(k)}) = E$
3. Релаксационная последовательность:

$$x^{(k+1)} = x^{(k)} + \alpha^{(k)} p^{(k)}, \quad \alpha^{(k)} \subset R, p^{(k)} \in R^n, x^{(k)} \in R^n$$

$$g^{(k)}(\alpha^{(k)}) = \min_{\alpha > 0} f(x^{(k)} + \alpha p^{(k)}), \text{ важно}$$

$\alpha^{(k)}$ должно удовлетворять условию Вольфе:

$$f(x^{(k)} + \alpha^{(k)} p^{(k)}) \leq f(x^{(k)}) + c_1 \cdot \alpha^{(k)} (\nabla f(x^{(k)}), p^{(k)})$$

$$\text{и } (\nabla f(x^{(k)} + \alpha^{(k)} p^{(k)}), p^{(k)}) \geq c_2 \cdot (\nabla f(x^{(k)}), p^{(k)}), \text{ где}$$

$$0 \leq c_1 \leq c_2 \leq 1, \text{ как правило } c_1 = 0.0001, c_2 = 0.9$$

4. Вычисляем $\Delta x^{(k)} = x^{(k+1)} - x^{(k)}$, $\Delta g^{(k)} = \nabla f(x^{(k+1)}) - \nabla f(x^{(k)})$, где $\Delta x^{(k)}$ – шаг алгоритма на k -й итерации,

$\Delta g^{(k)}$ – изменение градиента на k -й итерации.

5. Пересчитываем $\eta^{(k+1)}$:

$$\begin{aligned}\eta^{(k+1)} = & \left(E - \rho^{(k)} \cdot (\Delta x^{(k)} \otimes \Delta g^{(k)}) \right) \cdot \eta^{(k)} \cdot \left(E - \rho^{(k)} \cdot (\Delta g^{(k)} \otimes \Delta x^{(k)}) \right) + \\ & + \rho^{(k)} \cdot (\Delta x^{(k)} \otimes \Delta x^{(k)}), \text{ где } \rho^{(k)} = \frac{1}{(\Delta g^{(k)}, \Delta x^{(k)})}.\end{aligned}$$

Замечание.

Пусть $\tilde{H}(x^{(k)})$ это приближение матрицы Гессе $H(x^{(k)})$, тогда имеет место формула вычисления приближенного значения матрицы Гессе на $(k+1)$ -й шаге:

$$\tilde{H}^{(k+1)} = \tilde{H}^{(k)} - \frac{\tilde{H}^{(k)} \cdot (\Delta x^{(k)} \otimes \Delta x^{(k)}) \cdot \tilde{H}^{(k)}}{(\Delta x^{(k)}, \tilde{H}^{(k)} \Delta x^{(k)})} + \frac{(\Delta g^{(k)} \otimes \Delta g^{(k)})}{(\Delta g^{(k)}, \Delta x^{(k)})}$$

6. Критерий остановки алгоритма: $\|\nabla f(x^{(k)})\| < \varepsilon$

Пример решения методом BFGS.

Задача: найти экстремум функции $f(x) = f(x_1, x_2) = x_1^2 - x_1 \cdot x_2 + x_2^2 + 9x_1 - 6x_2 + 20$.

▷ Положим $x^{(0)} = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$, пусть $\varepsilon = 10^{-3} > 0$

Шаг ($k = 0$):

$$\nabla f(x_1, x_2) = \nabla f(x) = \begin{pmatrix} 2x_1 - x_2 + 9 \\ -x_1 + 2x_2 - 6 \end{pmatrix},$$

$$x^{(0)} = \begin{pmatrix} 1 \\ 1 \end{pmatrix}, \text{ тогда } \nabla f(x^{(0)}) = \begin{pmatrix} 10 \\ -5 \end{pmatrix},$$

Проверяем критерий остановки:

$$\|\nabla f(x^{(0)})\|_{k=0} = \|\nabla f(x^{(0)})\| \leq \varepsilon:$$

$$\|\nabla f(x^{(0)})\|_2 = (10^2 + (-5)^2)^{\frac{1}{2}} = 11.18 > \varepsilon = 10^{-3}$$

Т.к. критерий остановки не выполнился, то

$$p^{(k)} = -H^{-1}(x^{(k)}) \cdot \nabla f(x^{(k)}), H(x^{(0)}) = E, \text{ тогда}$$

$$p^{(0)} = -E \cdot \nabla f(x^{(0)}) = -\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} 10 \\ -5 \end{pmatrix} = \begin{pmatrix} -10 \\ 5 \end{pmatrix}.$$

Решаем оптимизационную задачу по α :

$$g^{(0)}(\alpha^{(0)}) = \min_{\alpha > 0} f(x^{(0)} + \alpha p^{(0)}), \quad x^{(0)} + \alpha p^{(0)} = \begin{pmatrix} 1 \\ 1 \end{pmatrix} + \alpha \begin{pmatrix} -10 \\ 5 \end{pmatrix} =$$

$$= \begin{pmatrix} 1 - 10\alpha \\ 1 + 5\alpha \end{pmatrix},$$

$$g^{(0)}(\alpha) = f(x^{(0)} + \alpha p^{(0)}) = (1 - 10\alpha)^2 - \frac{(1 - 10\alpha)(1 + 5\alpha)}{x_1} +$$

$$+ \frac{(1 + 5\alpha)^2}{x_2} + 9(1 - 10\alpha) - \frac{6(1 + 5\alpha)}{x_2} + 20,$$

$$\text{Далее } \frac{dg^{(0)}(\alpha)}{d\alpha} = 350\alpha - 125 = 0 \Rightarrow \alpha^{(0)} = 0.357$$

$$x^{(k+1)} = x^{(k)} + \alpha^{(k)} p^{(k)}, \text{ при } k = 0: x^{(1)} = x^{(0)} + \alpha^{(0)} p^{(0)} =$$

$$= \begin{pmatrix} 1 \\ 1 \end{pmatrix} + 0.357 \cdot \begin{pmatrix} -10 \\ 5 \end{pmatrix} = \begin{pmatrix} -2.571 \\ 2.786 \end{pmatrix}.$$

$$\text{Тогда } \Delta x^{(k)} = \Delta x^{(k+1)} - \Delta x^{(k)} \underset{k=0}{=} \Delta x^{(0)} = \Delta x^{(1)} - \Delta x^{(0)} =$$

$$= \begin{pmatrix} -2.571 \\ 2.786 \end{pmatrix} - \begin{pmatrix} 1 \\ 1 \end{pmatrix} = \begin{pmatrix} -3.571 \\ 1.786 \end{pmatrix}.$$

$$\text{Вычислим: } \nabla f(x^{(1)}) = \begin{pmatrix} 2x_1 - x_2 + 9 \\ -x_1 + 2x_2 - 6 \end{pmatrix} = \left| x^{(1)} = \begin{pmatrix} -2.571 \\ 2.786 \end{pmatrix} \right| =$$

$$\begin{pmatrix} 1.071 \\ 2.143 \end{pmatrix}$$

$$\text{Разность градиента: } \Delta g^{(k)} = \nabla f(x^{(1)}) - \nabla f(x^{(0)}) = \begin{pmatrix} 1.071 \\ 2.143 \end{pmatrix} - \begin{pmatrix} 10 \\ -5 \end{pmatrix} =$$

$$= \begin{pmatrix} -8.929 \\ 7.143 \end{pmatrix}.$$

По формуле:

$$\eta^{(k+1)} = \left(E - \rho^{(k)} \cdot (\Delta x^{(k)} \otimes \Delta g^{(k)}) \right) \cdot \eta^{(k)} \cdot \left(E - \rho^{(k)} \cdot (\Delta g^{(k)} \otimes \Delta x^{(k)}) \right) +$$

$$+ \rho^{(k)} \cdot (\Delta x^{(k)} \otimes \Delta x^{(k)}), \quad \text{где } \rho^{(k)} = \frac{1}{(\Delta g^{(k)}, \Delta x^{(k)})}$$

$$\eta^{(k+1)} \underset{k=0}{=} \eta^{(1)} = \begin{pmatrix} 0.694 & 0.367 \\ 0.367 & 0.709 \end{pmatrix}.$$

Шаг ($k = 1$):

$$x^{(1)} = \begin{pmatrix} -2.571 \\ 2.786 \end{pmatrix}, \text{ градиент } \nabla f(x^{(1)}) = \{2x_1 - x_2 + 9, -x_1 + 2x_2 - 6\} =$$

$$= \begin{pmatrix} 2x_1 - x_2 + 9 \\ -x_1 + 2x_2 - 6 \end{pmatrix} = \left| x^{(1)} = \begin{pmatrix} -2.571 \\ 2.786 \end{pmatrix} \right| = \begin{pmatrix} 1.071 \\ 2.143 \end{pmatrix}$$

Проверим условие остановки:

$$\|\nabla f(\mathbf{x}^{(1)})\|_2 = (1.071^2 + 2.143^2)^{\frac{1}{2}} = 2.396 > \varepsilon = 10^{-3}$$

критерий остановки не выполнился

$$\eta^{(1)} = \begin{pmatrix} 0.694 & 0.367 \\ 0.367 & 0.709 \end{pmatrix}, \text{ тогда}$$

$$p^{(1)} = -\eta^{(1)} \cdot \nabla f(\mathbf{x}^{(1)}) = -\begin{pmatrix} 0.694 & 0.367 \\ 0.367 & 0.709 \end{pmatrix} \cdot \begin{pmatrix} 1.071 \\ 2.143 \end{pmatrix} = \begin{pmatrix} -1.531 \\ -1.913 \end{pmatrix}$$

Решаем оптимизационную задачу по α , находим $\alpha^{(1)}$:

$$\begin{aligned} g^{(1)}(\alpha^{(1)}) &= \min_{\alpha>0} f(\mathbf{x}^{(1)} + \alpha p^{(1)}), \quad \mathbf{x}^{(1)} + \alpha p^{(1)} = \\ &= \begin{pmatrix} -2.571 \\ 2.786 \end{pmatrix} + \alpha \begin{pmatrix} -1.531 \\ -1.913 \end{pmatrix} = \begin{pmatrix} -2.571 - 1.531 \cdot \alpha \\ 2.786 - 1.913 \cdot \alpha \end{pmatrix}, \\ g^{(1)}(\alpha) &= f(\mathbf{x}^{(1)} + \alpha p^{(1)}) = (-2.571 - 1.531 \cdot \alpha)^2 - \\ &\quad - \underbrace{(-2.571 - 1.531 \cdot \alpha)(2.786 - 1.913 \cdot \alpha)}_{x_1} + \underbrace{(2.786 - 1.913 \cdot \alpha)^2}_{x_2} + \\ &\quad + 9 \underbrace{(-2.571 - 1.531 \cdot \alpha)}_{x_1} - 6 \underbrace{(2.786 - 1.913 \cdot \alpha)}_{x_2} + 20, \end{aligned}$$

$$\text{Тогда } \frac{dg^{(1)}(\alpha)}{d\alpha} = 6.15\alpha - 5.74 = 0 \Rightarrow \alpha^{(1)} = 0.933$$

$$\begin{aligned} \mathbf{x}^{(k+1)} &= \mathbf{x}^{(k)} + \alpha^{(k)} p^{(k)}, \text{ при } k = 1: \quad \mathbf{x}^{(2)} = \mathbf{x}^{(1)} + \alpha^{(1)} p^{(1)} = \\ &= \begin{pmatrix} -2.571 \\ 2.786 \end{pmatrix} + 0.933 \cdot \begin{pmatrix} -1.531 \\ -1.913 \end{pmatrix} = \begin{pmatrix} -3.999 \\ 1 \end{pmatrix}. \end{aligned}$$

$$\begin{aligned} \text{Тогда } \Delta \mathbf{x}^{(k)} &= \Delta \mathbf{x}^{(k+1)} - \Delta \mathbf{x}^{(k)} \underset{k=1}{=} \Delta \mathbf{x}^{(1)} = \Delta \mathbf{x}^{(2)} - \Delta \mathbf{x}^{(1)} = \\ &= \begin{pmatrix} -3.999 \\ 1 \end{pmatrix} - \begin{pmatrix} -2.571 \\ 2.786 \end{pmatrix} = \begin{pmatrix} -1.429 \\ -1.786 \end{pmatrix}. \end{aligned}$$

$$\text{Вычислим: } \nabla f(\mathbf{x}^{(2)}) = \begin{pmatrix} 2x_1 - x_2 + 9 \\ -x_1 + 2x_2 - 6 \end{pmatrix} = \left| \mathbf{x}^{(2)} = \begin{pmatrix} -3.999 \\ 1 \end{pmatrix} \right| = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

$$\begin{aligned} \text{Разность градиента: } \Delta \mathbf{g}^{(1)} &= \nabla f(\mathbf{x}^{(2)}) - \nabla f(\mathbf{x}^{(1)}) = \begin{pmatrix} 0 \\ 0 \end{pmatrix} - \begin{pmatrix} 1.071 \\ 2.143 \end{pmatrix} = \\ &= \begin{pmatrix} -1.071 \\ -2.143 \end{pmatrix}. \end{aligned}$$

$$\eta^{(k+1)} \underset{k=1}{=} \eta^{(2)} = \begin{pmatrix} 0.667 & 0.333 \\ 0.333 & 0.667 \end{pmatrix}.$$

Шаг ($k = 2$):

Проверим выполнение критерия остановки:

$$\left\| \nabla f(\mathbf{x}^{(k)}) \right\|_{k=2} = \left\| \nabla f(\mathbf{x}^{(2)}) \right\| \leq \varepsilon:$$

$$\left\| \nabla f(\mathbf{x}^{(2)}) \right\|_2 = (0^2 + 0^2)^{\frac{1}{2}} = 0 \leq \varepsilon = 10^{-3}$$

Алгоритм останавливаем. Таким образом, $\mathbf{x}^{(2)} = \begin{pmatrix} -3.571 \\ 0.786 \end{pmatrix}$. \triangleleft

Выводы:

1. Асимптотика каждой итерации BFGS: $O(n^2)$.
2. Алгоритм устойчив и имеет сверхлинейную сходимость.
3. Преимущество алгоритма – нет необходимости вычислять вторые производные и обращать матрицу Гессе.

Примеры реализации методов на языке Julia.

Метод BFGS:

```
max_iters = 100
d(x) = (1 - x[1])^2 + 100 * (x[2] - x[1]^2)^2
x0 = [-1.2, 2.0]
epsilon = 1e-3

function bfgs(f, x0, epsilon)
    x = copy(x0)
    trajectory = [copy(x)]
    n = length(x)
    B = I(n)
    grad = [df(f, x, j) for j in 1:n]
    i = 0

    while norm(grad) > epsilon && i < max_iters
        s = -B * grad

        g(alpha) = f(x + alpha * s)
        a, b = swann_method(g, 0.0)
        alpha = golden_section_search(g, a, b)

        x_new = x + alpha * s
        grad_new = [df(f, x_new, j) for j in 1:n]

        y = grad_new - grad
        s = alpha * s

        rho = 1 / dot(y, s)
        B = (I - rho * s * y') * B * (I - rho * y * s') + rho * s * s'

        x = x_new
        grad = grad_new
        push!(trajectory, copy(x))
    end
end
```

```

        i += 1
    end
    return trajectory, i
end

```

Метод L-BFGS:

```

function lbfgs(f, x0, epsilon, m=10)
    x = copy(x0)
    trajectory = [copy(x)]
    s_memory = []
    y_memory = []
    rho_memory = []
    grad = [df(f, x, i) for i in 1:length(x)]
    i = 0

    while norm(gradient) > epsilon && i < max_iters
        s = -approx_inv_hessian(s_memory, y_memory, rho_memory, grad,
m)
        g(alpha) = f(x + alpha * s)
        a, b = swann_method(g, 0.0)
        alpha = golden_section_search(g, a, b)
        x_new = x + alpha * s
        grad_new = [df(f, x_new, j) for j in 1:length(x)]
        y = grad_new - grad
        rho = 1 / dot(y, s)

        if length(s_memory) == m
            popfirst!(s_memory)
            popfirst!(y_memory)
            popfirst!(rho_memory)
        end

        push!(s_memory, s)
        push!(y_memory, y)
        push!(rho_memory, rho)
        x = x_new
        grad = grad_new
        push!(trajectory, copy(x))
        i += 1
    end
    return trajectory, i
end

function approx_inv_hessian(s_memory, y_memory, rho_memory, grad, m)
    g = copy(grad)
    alpha = Vector{Float64}(undef, length(s_memory))
    beta = Vector{Float64}(undef, length(s_memory))
    for i in length(s_memory):-1:1
        alpha[i] = rho_memory[i] * dot(s_memory[i], g)
        g -= alpha[i] * y_memory[i]
    end

    z = m * g
    for i in 1:length(s_memory)
        beta[i] = rho_memory[i] * dot(y_memory[i], z)
        z += (alpha[i] - beta[i]) * s_memory[i]
    end
end

```

```

    end
    return z
end

```

Лекция 7. Эффективность численных методов оптимизации.

Алгоритм Пирсона.

$$(6): \Delta\eta^{(k)} = \frac{1}{w} * \frac{(\Delta x^{(k)} \otimes y^{(k)})}{(y^{(k)}, \Delta y^{(k)})} - \frac{\eta^{(k)} * (\Delta g^{(k)} \otimes z)}{(z, \Delta g^{(k)})}$$

$$1. (6) \Rightarrow y = z = \Delta x^{(k)}, w = 1, \eta^{(k+1)} = \eta^{(k)} + \frac{((\Delta x^{(k)} - \eta^{(k)} * \Delta g^{(k)}) \otimes \Delta x^{(k)})}{(\Delta x^{(k)}, \Delta g^{(k)})}, \eta^{(0)} =$$

R положительно определенная матрица

2. 1. + переменная метрика

$$3. (6) \Rightarrow y * z_{w=1} = \eta^{(k)} * \Delta g^{(k)}, \eta^{(k+1)} = \eta^{(k)} + \frac{((\Delta x^{(k)} - \eta^{(k)} * \Delta g^{(k)}) \otimes (\eta^{(k)} * g^{(k)}))}{(\Delta g^{(k)}, \eta^{(k)} * \Delta g^{(k)})}$$

Алгоритма Ньютона – Рафсона.

$$(6) \Rightarrow w \rightarrow \infty, z = \eta^{(k)} * \Delta g^{(k)};$$

$$\eta^{(k+1)} = \eta^{(k)} - \frac{((\eta^{(k)} * \Delta y^{(k)}) \otimes (\eta^{(k)}, \Delta g^{(k)}))}{(\Delta g^{(k)}, \eta^{(k)} * \Delta g^{(k)})}$$

Алгоритмы с аппроксимацией матрицы Гёссе.

Идея: аппроксимируем не $H^{-1}(x^{(k)}) \sim \eta^{(k)}$, а матрицу $H(x^{(k)})$, далее ищем обратную матрицу к $H(x^{(k)})$, тогда на k -ой итерации ищем:

$$H(x^{(k+1)}) \approx \Gamma^{(k+1)} = \Gamma^k + \Delta\Gamma^k,$$

где $\Gamma^k \sim H(x^{(k)})$, при этом $\Delta\Gamma^k$ – симметричная матрица ранга 1, при этом $\Gamma^{(k+1)} \Delta x^{(k)} = \Delta g^{(k)}$,

$$\Delta\Gamma^k = \frac{(\Delta g^{(k)} - \Gamma^{(k)} \Delta x^{(k)}) \otimes (\Delta g^{(k)} - \Gamma^{(k)} \Delta x^{(k)})}{((\Delta g^{(k)} - \Gamma^{(k)} \Delta x^{(k)}), \Delta x^{(k)})}$$

Замечание.

$[\Gamma^{(k+1)}]^{-1}$ – может стать не положительно определенной соответственно необходимо контролировать это условие и принимать меры для того, чтобы $[\Gamma^{(k+1)}]^{-1}$ сохранила положительную определенность.

Замечание.

$$\Gamma^{(0)} = [\eta^{(0)}]^{-1}.$$

Алгоритм Гольштайна и Прайса.

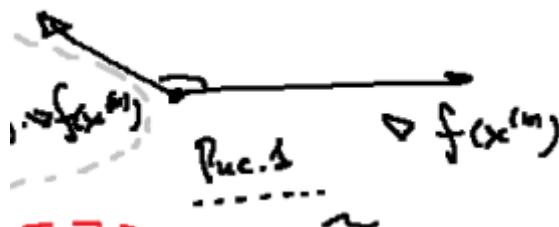
На k-ом шаге $0 < \delta < \frac{1}{2}, r > 0$.

1. Вычислим $\tilde{H}(x^{(k)})$

$\tilde{H}(x^{(k)}) = [h_1, h_2, \dots, h_n]$ (столбцы или векторы), где $h_j \in R^{nx1} : h_j = \nabla f(x^{(k)} + \theta^{(k)} E_j) - \nabla f(x^{(k)})$, где $E_j \in R^{nxn}$ – j – й столбец.

$\theta^{(k)} = r * \|\phi(x^{(k-1)})\|$ (например, Евклидова единичной матрицы или равномерная норма) для $k > 0, \theta^{(0)} = r, \phi^{(k)} \equiv \phi(x^{(k)})$ – вектор, который равен:

«Плохой» случай:



$$\Phi(x^{(k)}) = \tilde{H}^{-1}(x^{(k)}) * \nabla f(x^{(k)})$$

$$\Phi(x^{(k)}) = -\nabla f(x^{(k)}), \text{ если:}$$

- $k=0$;
- $\tilde{H}(x^{(k)})$ – сингулярна или
- $(\nabla f(x^{(k)}), \tilde{H}^{-1}(x^{(k)}) * \nabla f(x^{(k)}) (\Phi(x^{(k)}))) \leq 0$, то есть $\tilde{H}(x^{(k)})$ не положительна определена.

«Хороший» случай:

$$\Phi(x^{(k)}) = -\tilde{H}^{-1}(x^{(k)}) * \nabla f(x^{(k)})$$

Замечание.

Матрица $\tilde{H}(x^{(k)})$ вообще говоря может оказаться не симметричной, тогда если $(\nabla f(x^{(k)}), \tilde{H}^{-1}(x^{(k)}) * \nabla f(x^{(k)})) \leq 0$, то предполагаемо направление $\Phi(x^{(k)}) = \tilde{H}^{-1}(x^{(k)}) * \nabla f(x^{(k)})$ и направление $\nabla f(x^{(k)})$ не будут совпадать (смотреть рисунок 1), а следовательно $\Phi(x^{(k)})$ будет направлять в сторону возрастания целевой функции.

Вычисляется в качестве аппроксимации $H(x^{(k)})$ матрица $\tilde{H}(x^{(k)})$, j-й столбец которой определяется по формуле $\nabla f(x^{(k)} + \theta^k E_i) - \nabla f(x^{(k)})$ где $\theta^k = r \|\phi(x^{k-1})\|$ для $k > 0$, $\theta^0 = r$, E_j – j-й столбец единичной матрицы Е размерности $n \times n$. $\phi(x^k)$ – вектор-столбец, определяемый в соответствии с условиями:

- $\phi(x^k) = -\nabla f(x^k)$, если $k = 0$ или $\tilde{H}(x^{(k)})$ сингулярна, или $\nabla^T f(x^k) \tilde{H}^{-1}(x^{(k)}) \nabla f(x^{(k)}) \leq 0$, так что матрица $\tilde{H}^{-1}(x^{(k)})$ – не является положительно определенной;
- $\phi(x^k) = \tilde{H}^{-1}(x^{(k)}) \nabla f(x^{(k)})$ – в противном случае.

Заметим, что матрица $\tilde{H}(x^{(k)})$ не обязательно симметрическая матрица, и если $\nabla^T f(x^k) \tilde{H}^{-1}(x^{(k)}) \nabla f(x^{(k)}) \leq 0$, то предлагаемое направление поиска $\phi(x^k)$ и направление градиента $\nabla f(x^{(k)})$ отличаются более чем на 90° . Следовательно, $\phi(x^k)$ может быть направлено в сторону, в которой $f(x)$ увеличивается.

2. $F(x^k, \alpha) = \frac{f(x^k) - f(x^k + \alpha \phi(x^k))}{\alpha (\nabla f(x^k), \phi(x^k))}$, вычислим $\alpha^{(k)}$ таким образом: $\delta \leq F(x^k, \alpha) \leq 1 - \delta$, где $\delta \in (0, \frac{1}{2})$ $\alpha^k \neq 1$.
3. $x^{k+1} = x^k + \alpha \phi(x^k)$.
4. Критерий остановки $\|\phi(x^{k-1})\| < \varepsilon$.

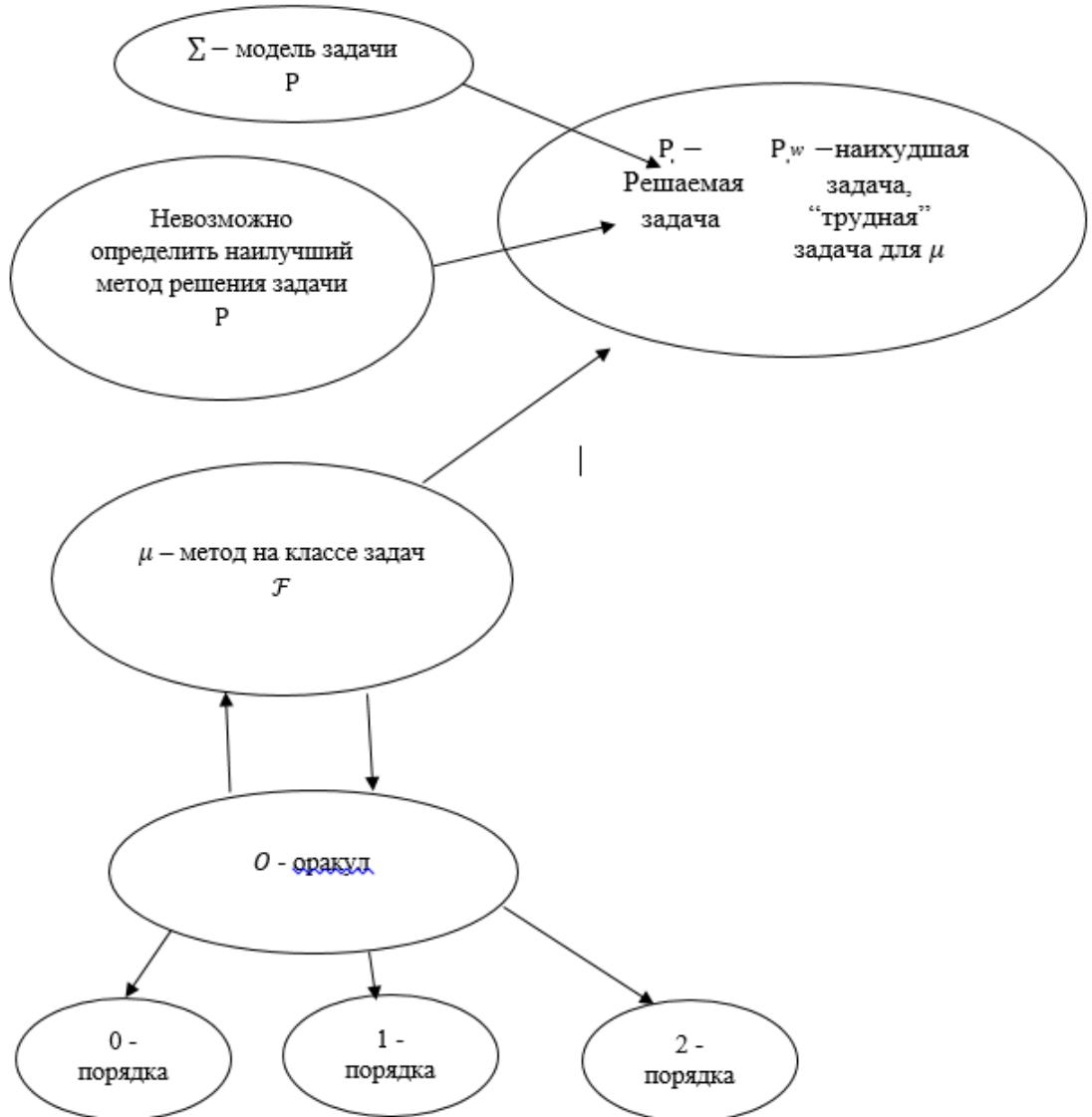
Замечание.

А) г выбираем так, чтобы $\tilde{H}(x^{(k)})$ приближала $H(x^{(k)})$ как можно лучше.

Б) δ выбирается так, чтобы последовательность $f(x^{(k)})$, $k = 1, 2, \dots$ являлась монотонно убывающей.

Эффективность численных методов.

\mathcal{F} – класс задач = $(\Sigma, O, \mathcal{F}_\varepsilon)$



Утверждение.

Не существует универсальных методов решения задач оптимизации, другими словами, невозможно определить наилучший метод решения задач P .

Возможно, определить класс методов для заданного класса задач. Класс задач – F , $P \subset \mathcal{F}$, где \mathcal{F} – выделенный класс задач.

Замечание.

Как правило используемые методы разбиваются для решения многих однотипных задач с близкими характеристиками.

Утверждение.

В силу разбиения эффективности метода M на всем классе F можно считать естественной характеристикой его качества.

Утверждение.

Метод μ не имеет полной информации о решаемой задаче P .

Теорема о бесплатных завтраках. (Дэвид Волперт и Уильям Макриди, NFL, 1997)

Если алгоритм хорошо работает над определенном классом задач, то он обязательно платит за это снижением производительности на множестве всех оставшихся задач.

Определение.

Σ – модель решений задачи – это заранее известная численному методу «часть» задачи P . Как правило в Σ включается формулировка задачи, описание свойств компонент и так далее.

Для того чтобы распознать задачу P среди P_i (других) задач класса F и соответственно решить её, метод должен накапливать информацию об этой задаче P .

Определение.

О – оракул – это процесс наполнения информации о задаче P , другими словами О-это устройство или алгоритм, которое отвечает на последовательные вопросы численного метода.

Определим пару (Σ, O) , которые определяют эффективность μ , как эффективность на наихудшем представителе P_w из (Σ, O) .

Замечание.

Задачи P_w может быть трудной только для M .

Определение.

Эффективность метода μ на задаче P определяется через общие вычислительные затраты метода μ , необходимые для того, чтобы решить задачу P .

Определение.

Решить задачу P – означает найти её приближение с точностью $\varepsilon > 0$, которая заведомо задана.

Определение.

T_ε обозначение некоторого критерия остановки способного оценить качество решения.

Определение.

Введем класс решаемых задач: $\mathcal{F} = (\Sigma, O, F_\varepsilon)$.

Замечание.

Для решения конкретной задачи $P \in \mathcal{F}$ применяем итеративную процедуру, таким образом мы формализуем или записываем $\forall \mu$, работающей с O .

Общая итеративная схема.

Исходные данные: $x^{(0)}, \varepsilon > 0$

Настройки: $k=0, I^{(-1)} = \emptyset$, k -шаг, $I^{(k)}$ –накапливающаяся информационная модель решаемой задачи.

Цикл:

Шаг 1. Задаем вопрос оракулу O в $x^{(k)}$

Шаг 2. Пересчитываем информационную модель:

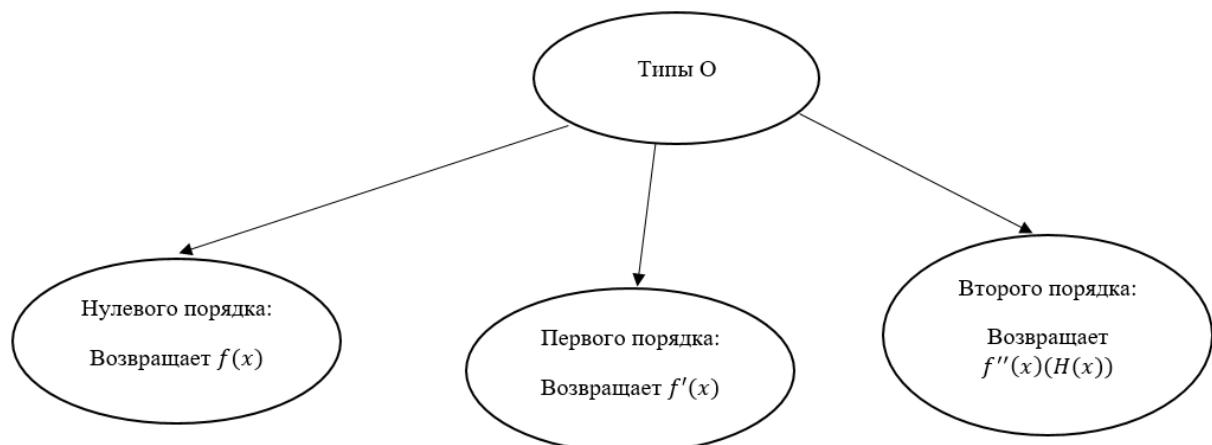
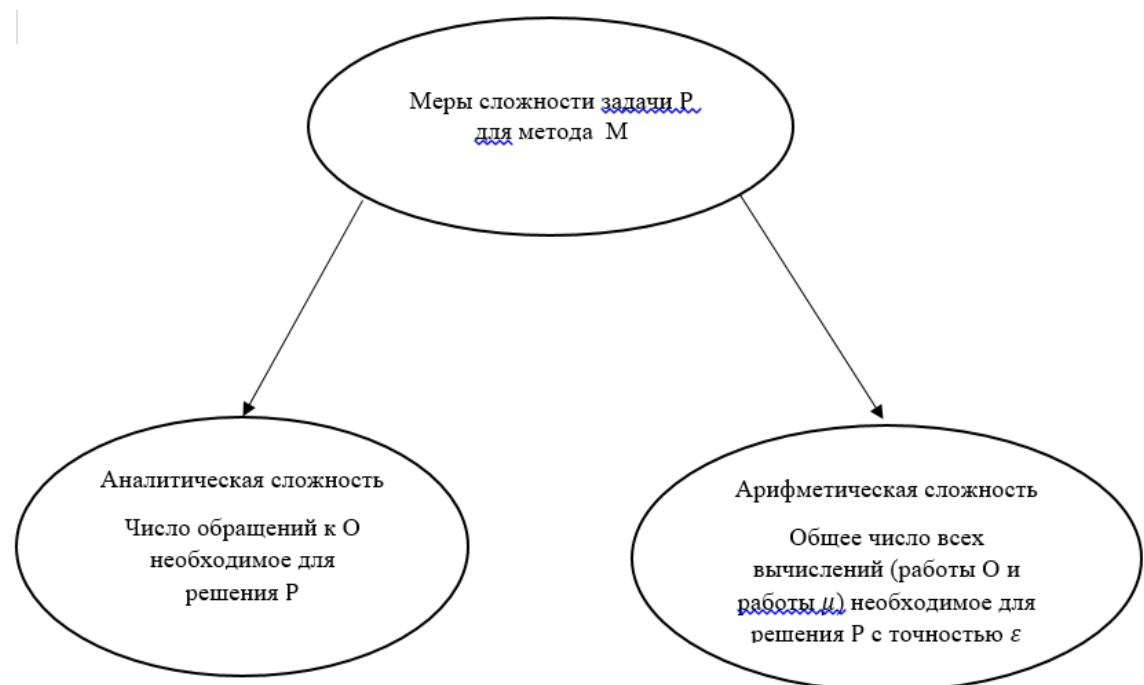
$$I^{(k)} = I^{(k-1)} \cup (x^{(k)}, O(x^{(k)}))$$

Шаг 3. Применяем условия метода М для анализа $I^{(k)}$ и получаем $x^{(k+1)}$

Шаг 4. Проверяем критерий остановки J_ε , если ответ положительный, то генерируем ответ \tilde{x} , иначе $k:=k+1$ и goto Шаг 1.

Замечание.

Наиболее трудоемкие как правило Шаг 1 и Шаг 3



O – 0-порядка

1. Метод Хука-Дживса

2. Метод Розенброка
3. Метод Нельдера-Мида

O – 1-порядка

Методы сопряженных градиентов:

1. Флэтчера – Ривза
2. Полака – Рибьера

O – 2-порядка

Методы переменной метрики:

1. Бройдена
2. Дэвидона-Флэтчера-Пауэлла
3. Бройдена-Флэтчера-Гольдфарба-Шанно

Лекция 8. Метод многомерной условной оптимизации.

Рассмотрим оптимационные задачи при наличии ограничений, т.е. задачи поиска оптимального решения, удовлетворяющего некоторой системе ограничений.

Идея: преобразовать задачу условной оптимизации к задаче безусловной оптимизации.

Формальная формула задачи условной оптимизации.

$$\min \begin{cases} f(x), x \in R^n: h_j(x) = 0, & j = 1, 2, \dots, m \\ g_j(x) \leq 0, & j = m + 1, m + 2, \dots, k \end{cases} \quad (1)$$

где $f(x)$ – целевая функция, $x \in R^n$, $h_j(x)$, $j = 1, 2, \dots, m$ и $g_j(x)$, $j = m + 1, m + 2, \dots, k$ – это функции системы ограничений, которые желательно должны быть выпуклыми.

Метод штрафных функций

Вводим вспомогательную функцию

$$Q(x, r) = f(x) + \sum_{j=1}^m r_j \cdot H_j \cdot (h_j(\bar{x})) + \sum_{j=m+1}^k r_j \cdot G_j \cdot (g_j(x)), \quad (2)$$

где $x \in R^n$, $r = \left(\begin{array}{ll} r_1, r_2, \dots, r_m, & r_{m+1}, r_{m+2}, \dots, r_k \\ \text{для } h_j(x) & \text{для } g_j(x) \end{array} \right)^T \in R^k$

Замечание.

Тогда приближенное решение (1) находится в результате решения последовательности задач безусловной оптимизации функции $Q(x, r)$ из уравнения (2).

Утверждение.

Метод (внешних) штрафных функций заключается в том, что выбираются такие функции $H_j(h_j(x))$ и $G_j(g_j(x))$, чтобы они становились отличными от нуля (положительными) при нарушении соответственных ограничений $h_j(x) = 0, j = 1, 2, \dots, m$ и $g_j(x) \leq 0, j = m + 1, \dots, k$.

Замечание.

Так как при нарушении ограничений $h_j(x) = 0, j = 1, 2, \dots, m$ и $g_j(x) \leq 0, j = m + 1, \dots, k$ вспомогательная функция $Q(x, r)$ возрастает, то это является отрицательным фактом при безусловной оптимизации.

Замечание.

Функции $H_j(h_j(x))$ и $G_j(g_j(x))$ генерируют штрафы, когда становятся больше нуля.

Замечание.

В данном методе функции $H_j(h_j(x))$ и $G_j(g_j(x))$ должны быть равны нулю внутри допустимой области!

Замечание.

Для $g_j(x) \leq 0, j = m + 1, \dots, k$ должно выполняться условие $G_j(g_j(x)) \rightarrow 0$ при $g_j(x) \rightarrow 0$.

Утверждение.

Приближенное решение (1) получается в результате решения последовательности задач (3) при $r_j \rightarrow \infty, j = 1, 2, \dots, m, m + 1, m + 2, \dots, k = 1, 2, \dots, k$.

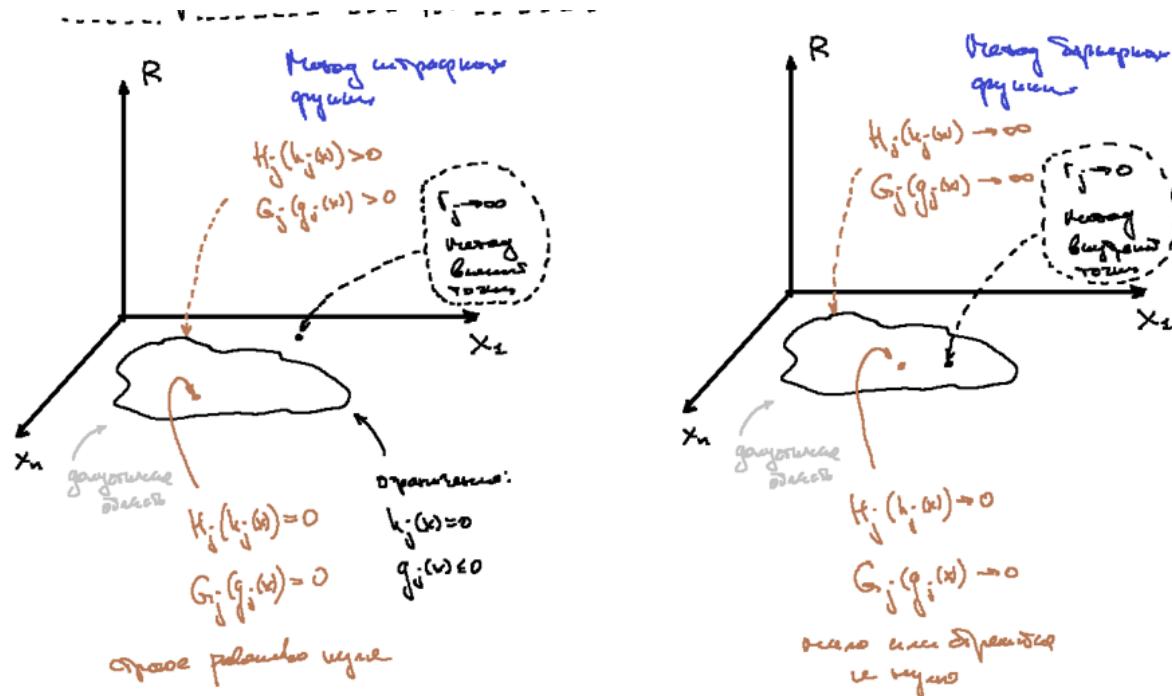
Замечание.

Соответствующие методы называют методами внешней точки $(r_j \rightarrow \infty, j = \overline{1, k})$.

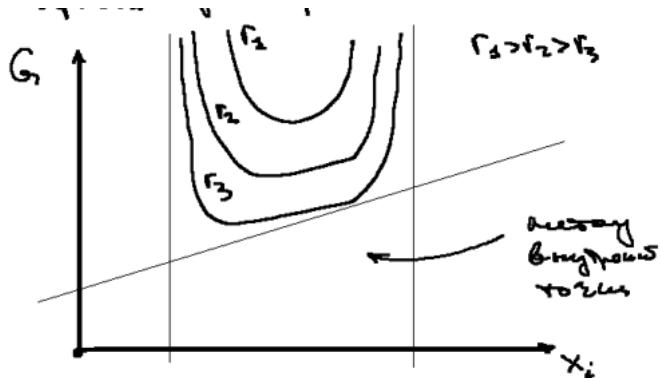
Метод барьерных функций

В методе барьерных функций $H_j(h_j(x))$ и $G_j(g_j(x))$ в допустимой области выбранные отличными от нуля, при этом при приближении к границе $H_j(h_j(x))$ и $G_j(g_j(x))$ должны возвращать, тем самым препятствуя выходу за границу области при поиске минимума целевой функции $f(x)$. Другими словами, $H_j(h_j(x))$ и $G_j(g_j(x))$ должны принимать малые (положительные и отрицательные) значения внутри допустимой области и большие (положительные и отрицательные) значения вблизи границы.

Геометрическая интерпретация.



Пример для ограничения неравенств.



$$G_j(g_j(x)) \rightarrow 0 \text{ при } g_j(x) \rightarrow 0$$

Замечание.

Показано формирование барьера при приближении к границам изнутри области ограничения.

Замечание.

Такие методы называются методами «внутренней» точки $r_j \rightarrow 0, j = \overline{1, k}$.

Замечание.

В методах «внутренней» точки барьерные функции требуют, чтобы в процессе поиска точек $x^{(k)}$ всегда оставалась внутренней точкой допустимой области.

Замечание.

Приближенное решение (1) получается в результате решения задачи (3) при $r_j \rightarrow \infty, j = 1, 2, \dots, k$.

О выборе функций штрафов для ограничений равенств $H_j(h_j(x))$:

Требование при выборе функции $H_j(h_j(x))$: $H_j(h_j(x)) \rightarrow 0$ при $h_j(x) \rightarrow 0$.

Примеры функций $H_j(h_j(x))$:

1. $H_j(h_j(x)) = |h_j(x)|$

2. $H_j(h_j(x)) = |h_j(x)|^2$
3. $H_j(h_j(x)) = |h_j(x)|^\alpha, \alpha - \text{четное}$

Примеры функций $G_j(g_j(x))$:

Требования при выборе функции $G_j(g_j(x))$:

- $G_j(g_j(x)) = 0$ при $g_j(x) \leq 0$
- $G_j(g_j(x)) > 0$ при $g_j(x) > 0$

Таким требованиям, например, удовлетворяют следующие функции:

1. $G_j(g_j(x)) = \frac{1}{2}(g_j(x) + |g_j(x)|)$
2. $G_j(g_j(x)) = \left(\frac{1}{2}(g_j(x) + |g_j(x)|)\right)^2$
3. $G_j(g_j(x)) = \left(\frac{1}{2}(g_j(x) + |g_j(x)|)\right)^\alpha, \alpha - \text{четное}$

Дополнительно:

1. $G_j(g_j(x)) = -\frac{1}{g_j(x)}$
2. $G_j(g_j(x)) = -\ln(g_j(x))$

Линейное программирование.

Рассмотрим задачу оптимизации целевой функции $f(x), x \in R^n$ на допустимом множестве $X \subset R^n$, при этом целевая функция $f(x)$ и система ограничений, определяющих множество X линейно относительно переменных $x = (x_1, x_2, \dots, x_n)^T$.

$f(x) = c_1x_1 + c_2x_2 + \dots + c_nx_n \rightarrow \min$ – задача оптимизации целевой функции линейной относительно x_1, x_2, \dots, x_n .

При этом определены ограничения:

- условия равенства

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + \dots + a_{1n}x_n = b_1 \\ a_{21}x_1 + a_{22}x_2 + \dots + a_{2n}x_n = b_2 \\ \dots \\ a_{l1}x_1 + a_{l2}x_2 + \dots + a_{ln}x_n = b_l \end{cases} \quad (1)$$

– условия неравенства

$$\begin{cases} a_{(l+1)1}x_1 + a_{(l+1)2}x_2 + \dots + a_{(l+1)n}x_n \leq b_{(l+1)} \\ \dots \\ a_{m1}x_1 + a_{m2}x_2 + \dots + a_{mn}x_n \leq b_m \end{cases} \quad (2)$$

Замечание.

В исходной задаче линейного программирования (ЗЛП) неравенства могут быть другого знака, при необходимости умноженные на -1 эти неравенства приводятся к виду (2).

Определение.

Если в условии ЗЛП не содержатся неравенства (2), то есть $l \neq m$, то такая задача ЛП называется **задачей, записанной в каноническом виде**.

Тогда введя дополнительные переменные $x_{n+i} \geq 0, i = l + 1, \dots, m$ ограничения неравенств (2) можно записать в виде равенств:

$$\begin{cases} a_{(l+1)1}x_1 + a_{(l+1)2}x_2 + \dots + a_{(l+1)n}x_n = b_{(l+1)} \\ \dots \\ a_{m1}x_1 + a_{m2}x_2 + \dots + a_{mn}x_n = b_m \end{cases}$$

Таким образом, любая задача линейного программирования может быть записана в каноническом виде.

$$f(x) = c_1x_1 + c_2x_2 + \dots + c_nx_n \rightarrow \min \quad (3)$$

при следующих ограничениях:

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + \dots + a_{1n}x_n = b_1 \\ a_{21}x_1 + a_{22}x_2 + \dots + a_{2n}x_n = b_2 \\ \dots \\ a_{m1}x_1 + a_{m2}x_2 + \dots + a_{mn}x_n = b_m \end{cases} \quad (4)$$

где $x_1 \geq 0, x_2 \geq 0, \dots, x_m \geq 0$

Замечание.

ЗЛП в векторной форме можно записать следующим образом:

$$f(x) = (c, x) \rightarrow \min, \quad (6)$$

$$Ax = b,$$

$x \geq 0$, где $x = (x_1, x_2, \dots, x_n)^T \in R^n$ – вектор неизвестных,

$c = (c_1, c_2, \dots, c_n)^T \in R^n$ – вектор коэффициентов целевой функции из (3),

$A = \begin{pmatrix} a_{11} & \dots & a_{1n} \\ \dots & \dots & \dots \\ a_{m1} & \dots & a_{mn} \end{pmatrix}$ – матрица коэффициентов при неизвестных системы

уравнений (4),

$b = (b_1, b_2, \dots, b_n)^T \in R^n$ – вектор правых частей системы уравнений (4)

Утверждение.

Задача $f(x) = (c, x) \rightarrow \max$ на множестве X может быть сведена к минимизации функции $-f(x) = (-c, x)$ на том же множестве X .

Тогда $\nabla f(x) = c$ указывает на направление возрастания целевой функции $f(x) = (c, x)$, тогда антиградиент $-\nabla f(x) = -c$ на направление убывания целевой функции $f(x)$.

Пример 1.

Составим математическую модель об оптимизации производства изделия.

Задача.

Пусть изготавливаются детали двух видов 1 и 2 на трех станках. Время обработки одной детали типа 1 на первом станке 11 минут, на втором станке 7 минут, на третьем 6 минут. Время обработки одной детали типа 2 на первом станке 9 минут, на втором станке 12 минут, на третьем 16 минут. В течение месяца первый станок работает 9850 минут, второй – 8150 минут, третий – 9600 минут. Детали типа 1 приносят доход 900 рублей, детали типа 2 приносят доход 1000 рублей. Сколько необходимо ежемесячно производить деталей каждого типа, чтобы максимизировать общую прибыль?

▷ Пусть $x = (x_1, x_2)^T \in R^2$ – количество произведенных деталей типа 1 и 2 соответственно. Очевидно, что $x_1 \geq 0, x_2 \geq 0$, тогда пусть t_1, t_2 и t_3 –

время обработки деталей типа 1 и 2 на станке 1, станке 2 и станке 3 соответственно. Значит $t_1 = 9850$, $t_2 = 8150$, $t_3 = 9600$, тогда получим:

$$\begin{cases} t_1 = 9850 = 11x_1 + 9x_2 \\ t_2 = 8150 = 7x_1 + 12x_2 \\ t_3 = 9600 = 6x_1 + 16x_2 \end{cases}$$

Обозначим через $f(x)$ прибыль от производства деталей двух типов при изготовлении x_1 и x_2 единиц продукции, тогда получим:

$$f(x) = f(x_1, x_2) = 900x_1 + 1000x_2$$

Таким образом, условие извлечения максимальной прибыли:

$$f(x) = f(x_1, x_2) = 900x_1 + 1000x_2 \rightarrow \max$$

при ограничениях:

$$\begin{cases} 11x_1 + 9x_2 = 9850 \\ 7x_1 + 12x_2 = 8150 \\ 6x_1 + 16x_2 = 9600 \end{cases} \quad (7)$$

при $x_1 \geq 0, x_2 \geq 0$

◇

Графический метод решения задач линейного программирования

Пусть ЗЛП содержит только две переменные и в условии ЗЛП нет ограничений равенств, тогда такую задачу можно решить графически.

Пусть решается следующая ЗЛП:

$$f(x) = f(x_1, x_2) \rightarrow \min, \quad x = (x_1, x_2)^T \in R^2 \quad (8)$$

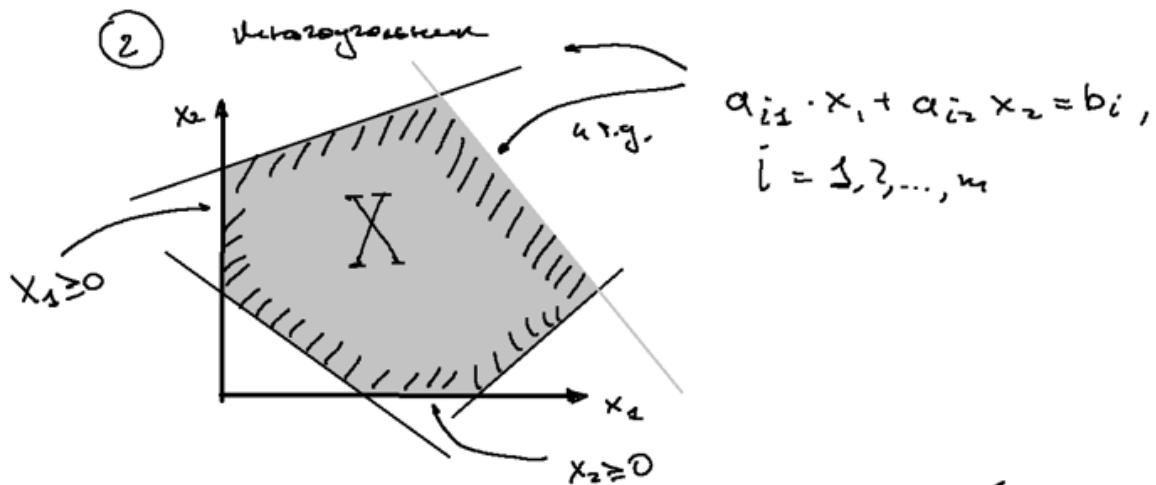
$$\begin{cases} a_{11}x_1 + a_{12}x_2 \leq b_1 \\ a_{21}x_1 + a_{22}x_2 \leq b_2 \\ \dots \\ a_{m1}x_1 + a_{m2}x_2 \leq b_m \end{cases} \quad (9)$$

$$\text{при этом } x_1 \geq 0, x_2 \geq 0 \quad (10)$$

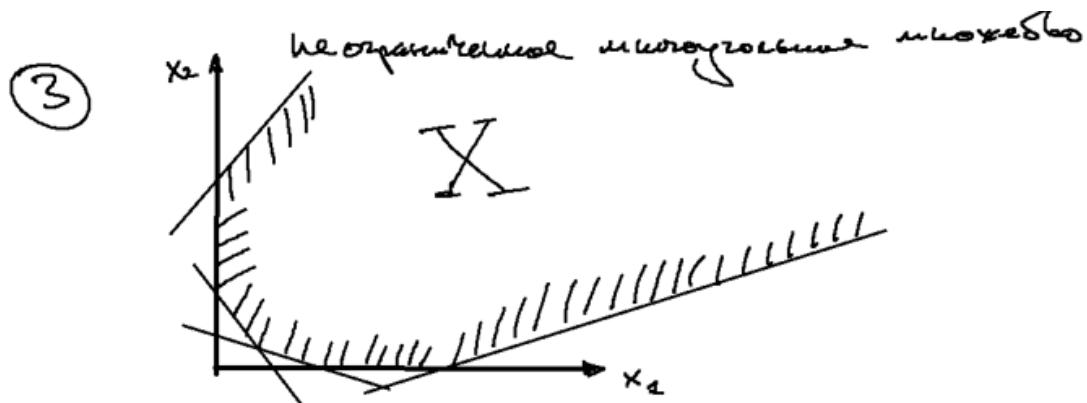
На плоскости (x_1, x_2) любое из неравенств (9) определяет полуплоскость, лежащую по одну сторону от прямой $a_{i1}x_1 + a_{i2}x_2 = b_i$, $i = 1, 2, \dots, m$.

Тогда допустимое множество X , заданное (8) – (10) располагается в первой четверти (x_1, x_2) и внутри области, ограниченной системой неравенств (9), поэтому множество X представляет собой:

1. Пустое множество, когда задача (8) – (10) не имеет решения из-за несовместности ограничений (9) – (10).
2. Многоугольник



3. Неограниченное многоугольное множество



Для решения задачи (8) – (10) в случае $X \neq \emptyset$ рассмотрим семейство линий уровня функции $f(x)$ из (8):

$$f_k(x) = f_k(x_1, x_2) = c_{k1}x_1 + c_{k2}x_2 = C = \text{const} \quad (11)$$

Замечание.

Антиградиент $e_k = -\nabla f(x) = (-c_{k1}, -c_{k2})$ перпендикулярен прямой (11) и указывает на направление убывания функции $f(x)$.

Замечание.

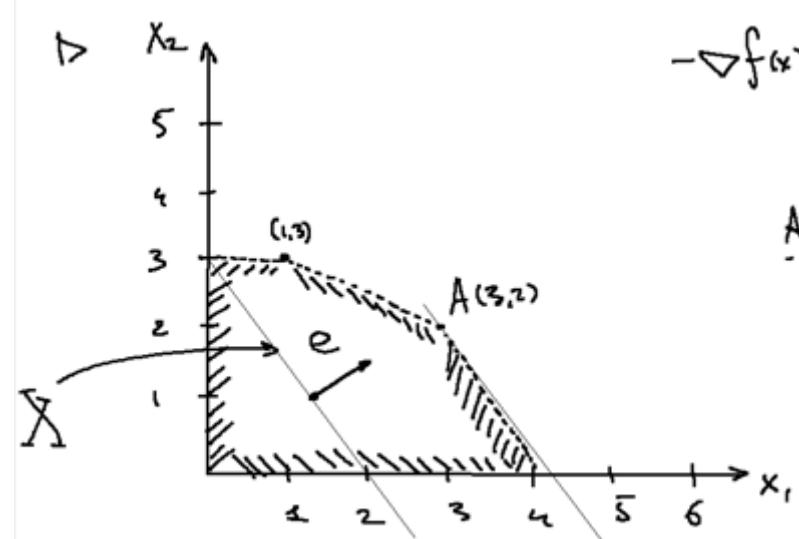
Перемещение прямых (11) по направлению $e_k, k = 1, \dots, m$ в какой-то момент приведет к пересечению в точке $f(x^*) \rightarrow \min$ при ограничениях (9) – (10).

Пример 1.

$$f(x) = f(x_1, x_2) = -3x_1 - 2x_2 \rightarrow \min$$

$$\begin{cases} x_1 + 2x_2 \leq 7 \\ 2x_1 + x_2 \leq 8 \\ x_1 \geq 0, x_2 \geq 0 \\ x_2 \leq 3 \end{cases}$$

▷



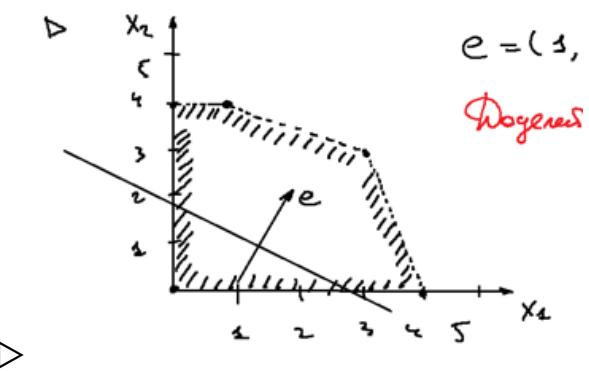
$-\nabla f(x) = (3, 2)$ – указывает на направление функции $f(x) = (c, x)$.

Алгоритм: совершаем параллельный перенос линии уровня вдоль направления антиградиента e , находим ее крайнее положение. В этом положении прямая $-3x_1 - 2x_2 = C$ проходит через точку $A(3, 2)$ множества X , поэтому целевая функция $f(x)$ принимает минимальное значение $f^* = f(x^*)$, $x^* = (3, 2)$. При этом $f^* = f(x^*) = f(3, 2) = -13$. ◁

Пример 2.

$$f(x) = f(x_1, x_2) = -x_1 - 2x_2 \rightarrow \min$$

$$\begin{cases} x_1 + 2x_2 \leq 7 \\ 2x_1 + x_2 \leq 8 \\ x_1 \geq 0, x_2 \geq 0 \\ x_2 \leq 3 \end{cases}$$



$e = -\nabla f(x) = (1, 2)$ – указывает на направление функции $f(x) = (c, x)$.

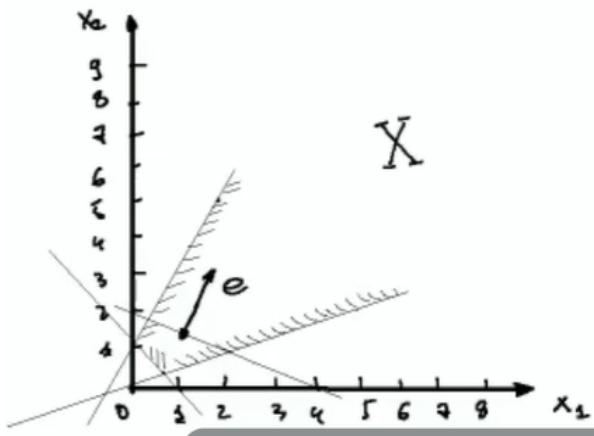
Совершаем параллельный перенос линии уровня вдоль направления антиградиента e , находим ее крайнее положение. В этом положении прямая $-x_1 - 2x_2 = C$ проходит через точки $A(3, 2)$ и $B(1, 3)$ множества X . Поэтому сравним значение целевой функции $f(x)$ в этих точках: $f(3, 2) = -7$, $f(1, 3) = -7$. Целевая функция принимает минимальное значение $f^* = f(x^*)$, $x^* = (3, 2)$ при $x_1 > 1$ и принимает минимальное значение $f^* = f(x^*)$, $x^* = (1, 2)$ при $x_1 \leq 1$. ◁

Пример 3.

$$f(x) = f(x_1, x_2) = -x_1 - 2x_2 \rightarrow \min$$

$$\begin{cases} x_1 + x_2 \geq 1 \\ 2x_1 - x_2 \geq -1 \\ x_1 - 2x_2 \leq 0 \\ x_1 \geq 0, x_2 \geq 0 \end{cases}$$

▷



$$f(x) = -x_1 - 2x_2 = (c, x), \quad -\nabla f(x) = (1, 2) = e$$

При параллельном переносе линии уровня $-x_1 - 2x_2 = C$ вдоль направления антиградиента $-\nabla f(x) = e = (1, 2)$ линия уровня всегда пересекает допустимое множество X , при этом целевая функция $f(x) = -x_1 - 2x_2$ неограниченно убывает. Следовательно, данная задача линейного программирования не имеет решения. \triangleleft

Пример 4.

Используя графический метод решить ЗЛП:

$$f(x) = x_1 + 9x_2 + 5x_3 + 3x_4 + 4x_5 + 14x_6 \rightarrow \min \quad (*)$$

$$\begin{cases} x_1 + x_4 = 20 \\ x_2 + x_5 = 50 \\ x_3 + x_6 = 30 \\ x_4 + x_5 + x_6 = 60 \end{cases} \quad x_j \geq 0, j = \overline{1, 6}$$

\triangleright

Матрица А ограничений-равенств имеет вид:

$$\begin{pmatrix} 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 \end{pmatrix}$$

Ранг матрицы $r = 4$, т.к. число свободных переменных равно 2, то можно воспользоваться графическим методом решения ЗЛП.

Будем реализовывать графическое решение ЗЛП относительно переменных x_5 и x_6 . Выражаем из начальной системы уравнений переменные x_1, x_2, x_3, x_4 :

$$\begin{cases} x_1 = -40 + x_5 + x_6 \\ x_2 = 50 - x_5 \\ x_3 = 30 - x_6 \\ x_4 = 60 - x_5 - x_6 \end{cases} \quad (**)$$

Исключаем из (*) переменные x_1, x_2, x_3, x_4 , тогда функция будет иметь вид:

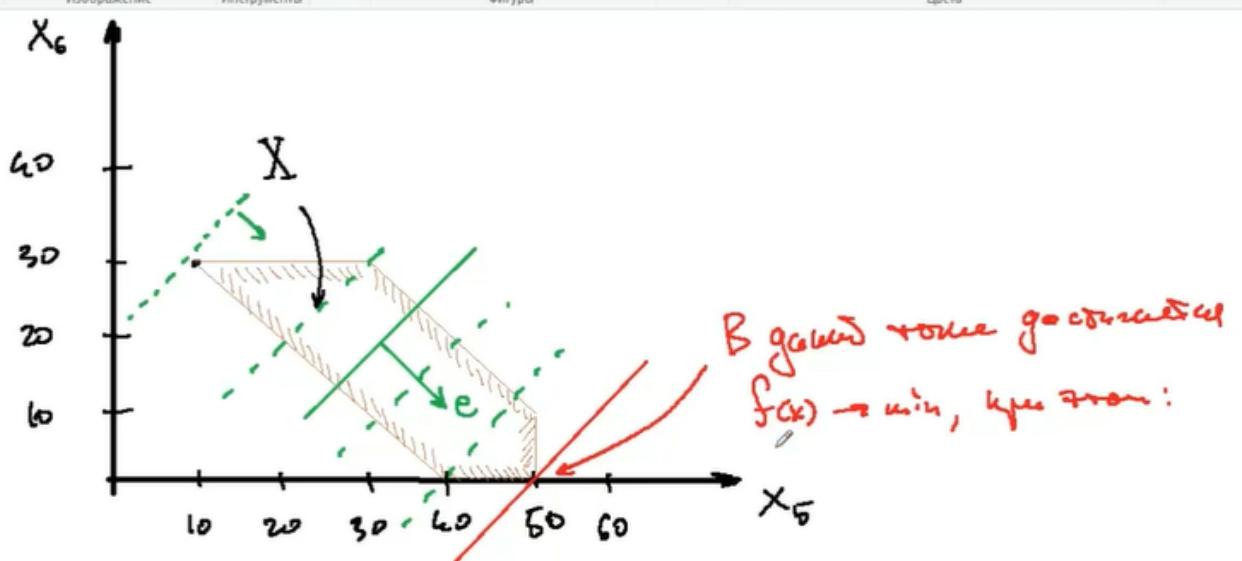
$$f(x) = 740 - 7x_5 + 7x_6 \rightarrow \min$$

Преобразованная ЗЛП выглядит следующим образом:

$$f(x) = 740 - 7x_5 + 7x_6 \rightarrow \min$$

$$\begin{cases} x_5 + x_6 \geq 40 \\ x_5 \leq 50 \\ x_6 \leq 30 \\ x_5 + x_6 \leq 60 \end{cases} \quad x_5 \geq 0, x_6 \geq 0$$

Антиградиент целевой функции $f(x)$: $-\nabla f = (7, 7) = e$.



Совершаем параллельный перенос линии уровня вдоль направления антиградиента e , находим ее крайнее положение. В этом положении прямая $740 - 7x_5 + 7x_6 = C$ проходит через точку $A(50, 0)$ множества X . Таким образом, при $x_5 = 50, x_6 = 0$ достигается минимум функции. Подставим значения $x_5 = 50, x_6 = 0$ в $(**)$ и найдем x_1, x_2, x_3, x_4 :

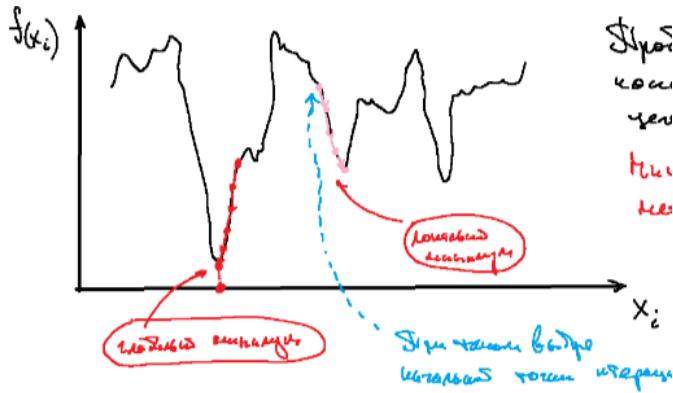
$$x = (x_1, x_2, x_3, x_4, x_5, x_6)^T = (10, 0, 30, 10, 50, 0)^T$$

Вывод: целевая функция $f(x)$ принимает минимальное значение $f^* = f(x^*)$, $x^* = (10, 0, 30, 10, 50, 0)$. При этом $f^* = f(x^*) = f(10, 0, 30, 10, 50, 0) = -390$.

□

Лекция 9. Алгоритмы глобальной оптимизации.

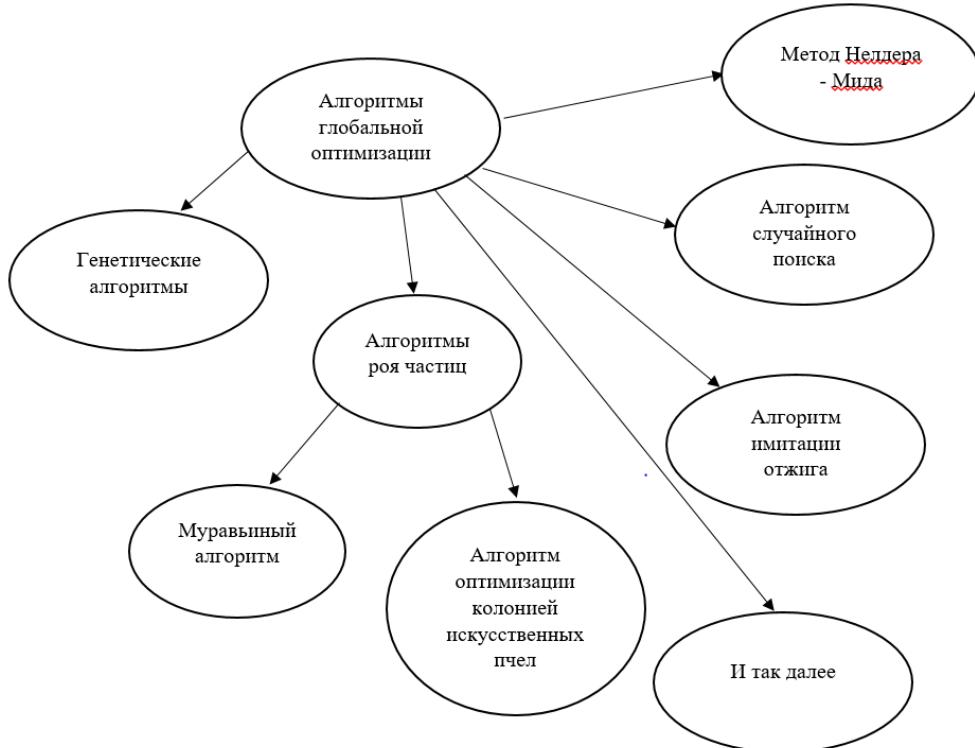
- оптимизация роем частиц;
- генетические алгоритмы.



Проблемы глобального поиска минимума целевой функции.

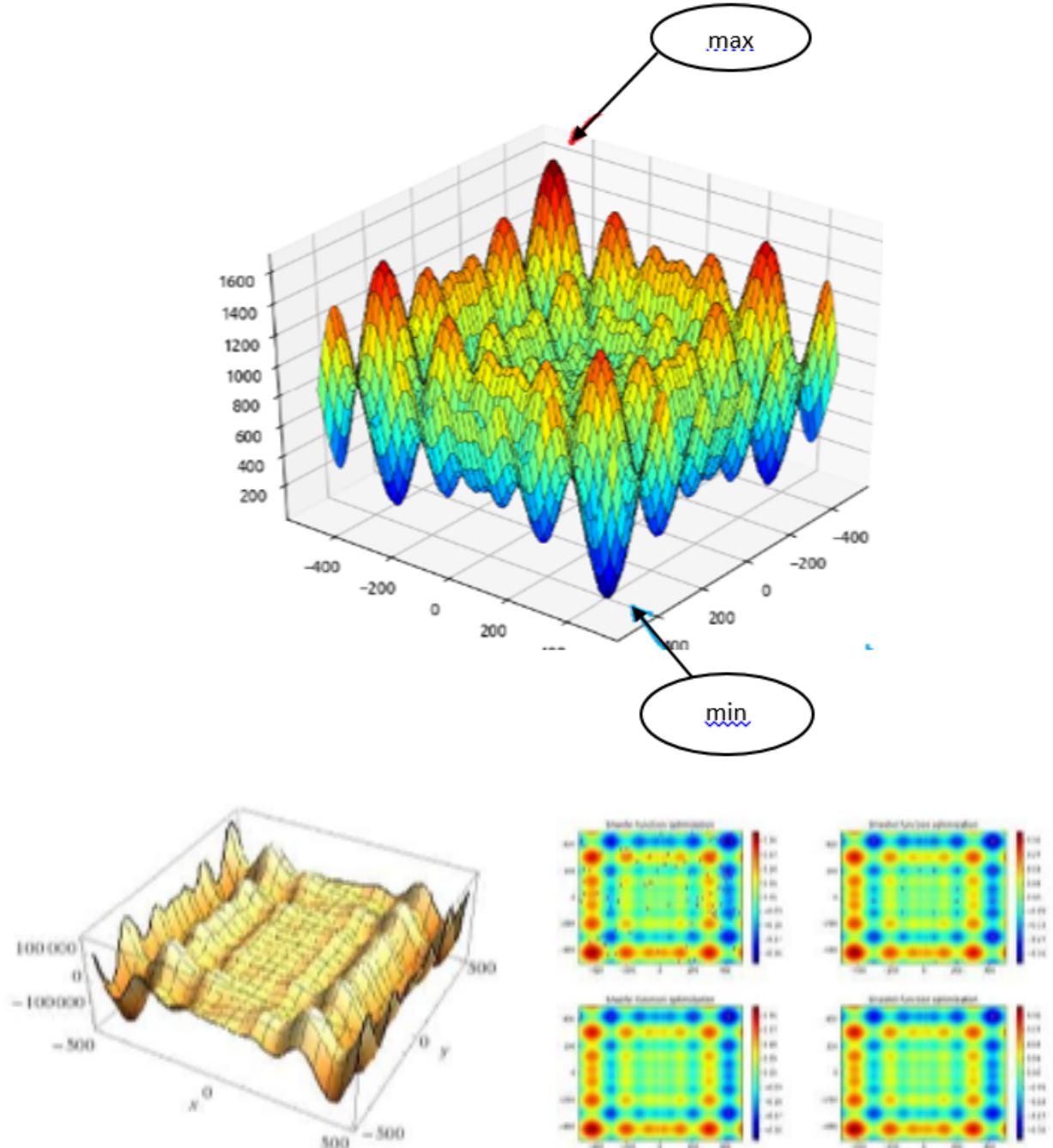
Минус градиентных методов – хорошо работают для отыскания локальных минимумов целевой функции.

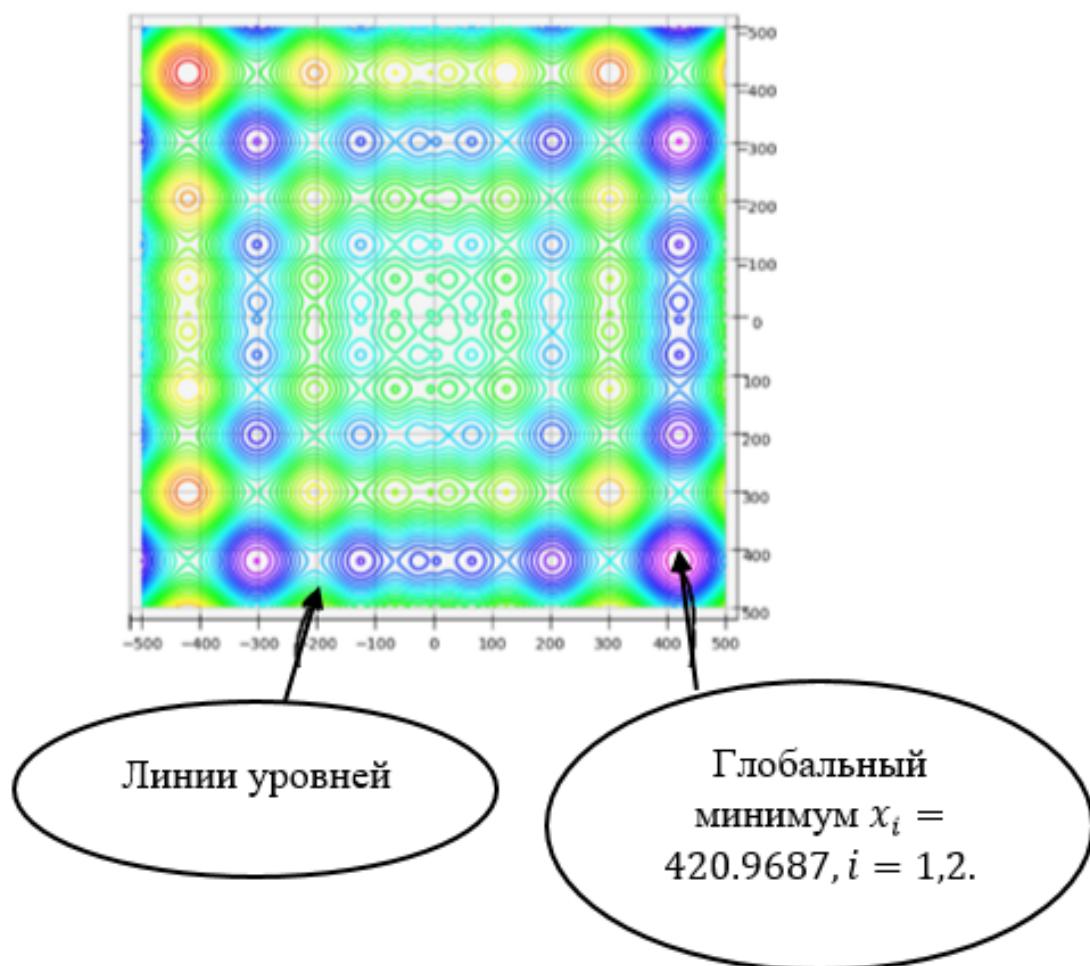
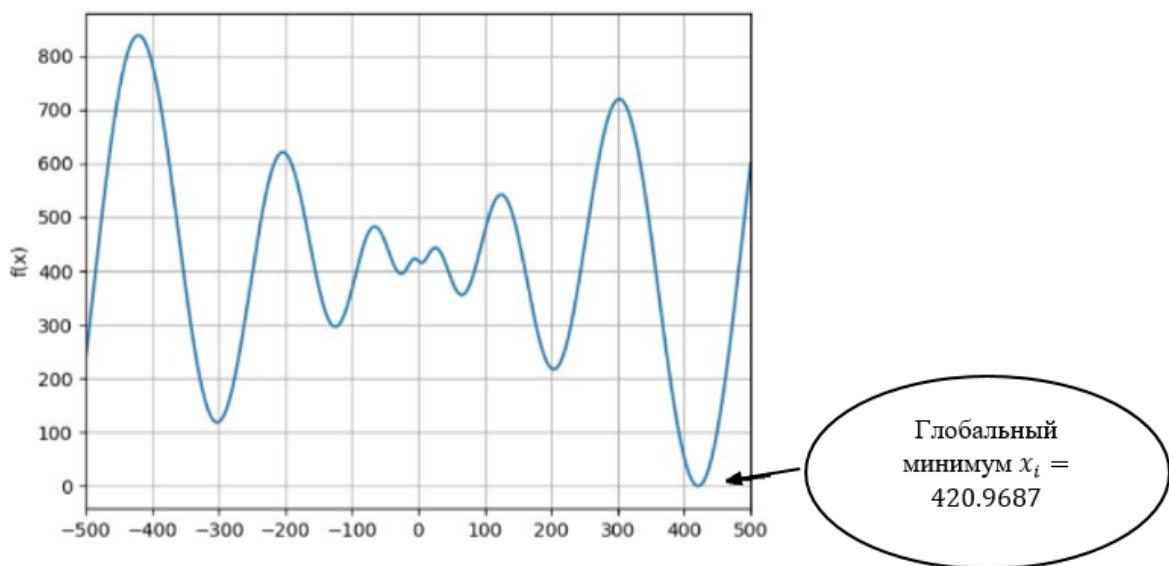
При таком выборе начальной точки итерационный процесс пойдет в локальный минимум, но не найдется глобальный минимум.



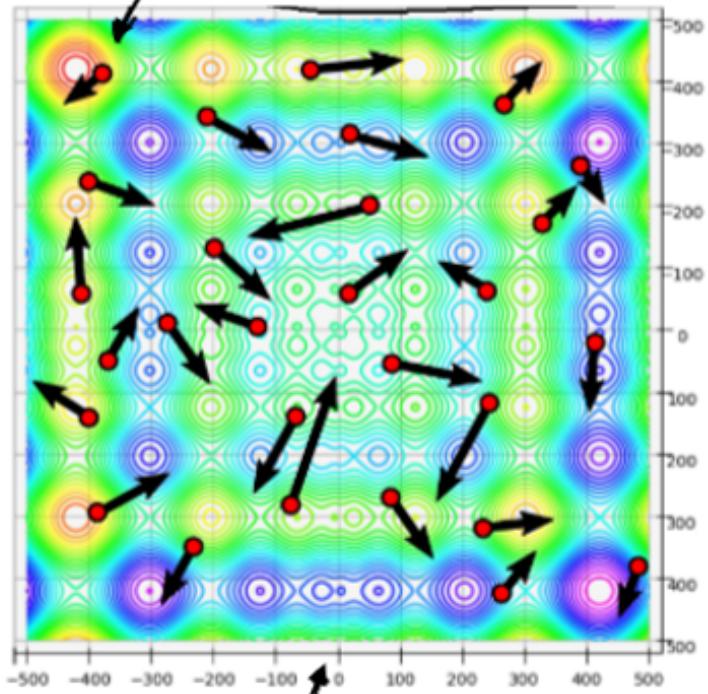
Функция Швефеля.

$f(x) = 418.9829n + \sum_{i=1}^n (-x_i * \sin(\sqrt{|x_i|}))$, глобальный минимум
 $f(x) = 0$, $x \in R^n$, при $x_i = 420.9687$, $i = 1, 2, \dots, n$; $-500 \leq x_i \leq 500$.

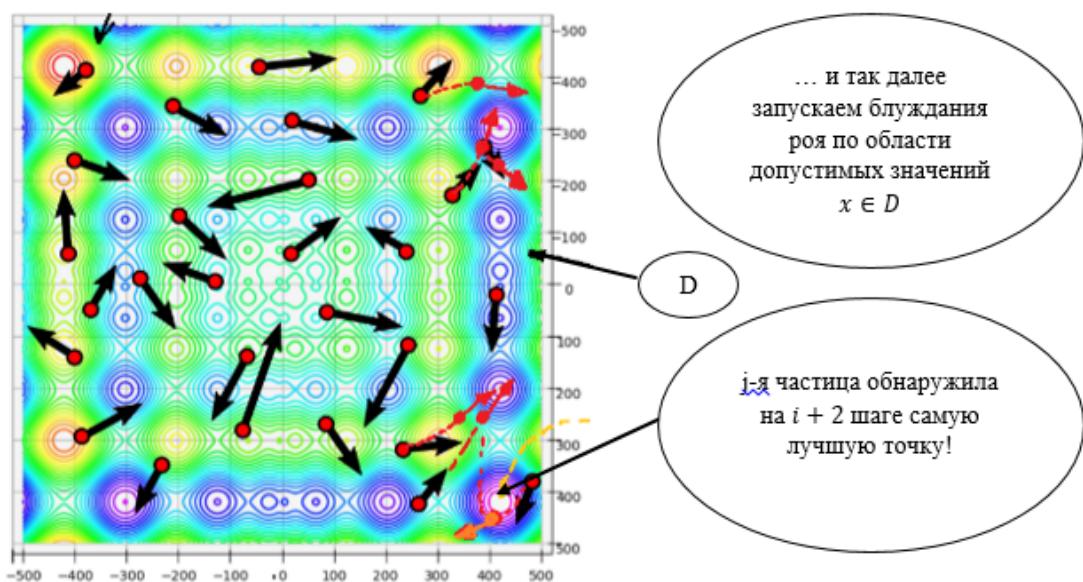
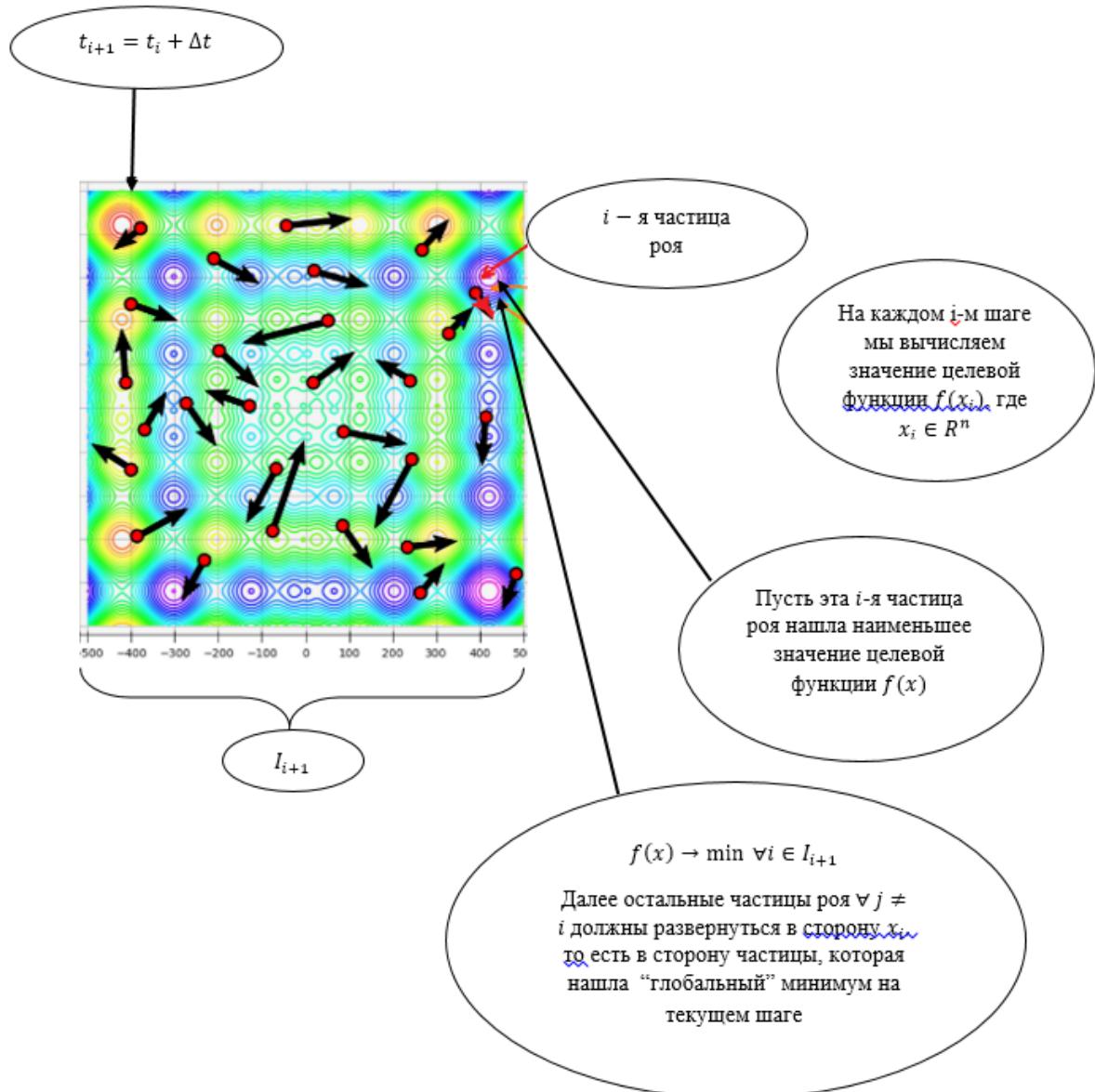




0-й шаг итерационного
процесса по t_0

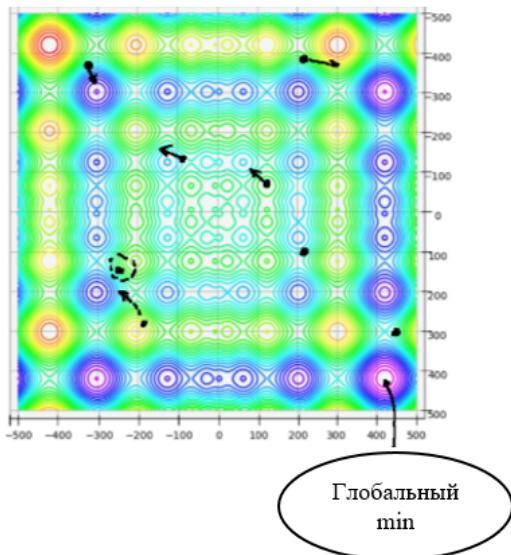


Разбрасываем случайно
рой частиц со случайными
направлениями движения
и случайными скоростями

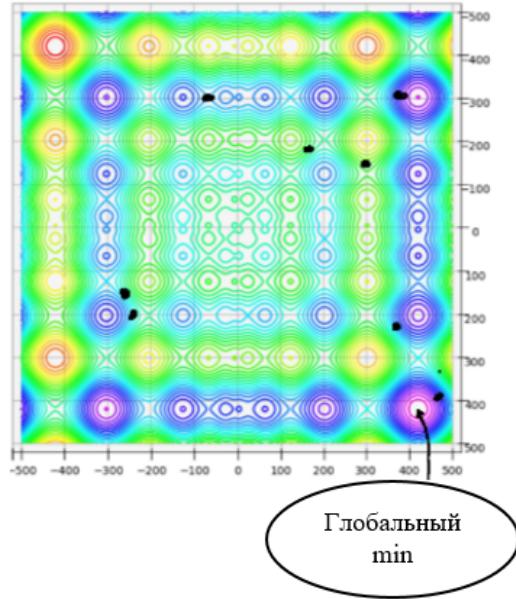


Примерная картинка роения частиц:

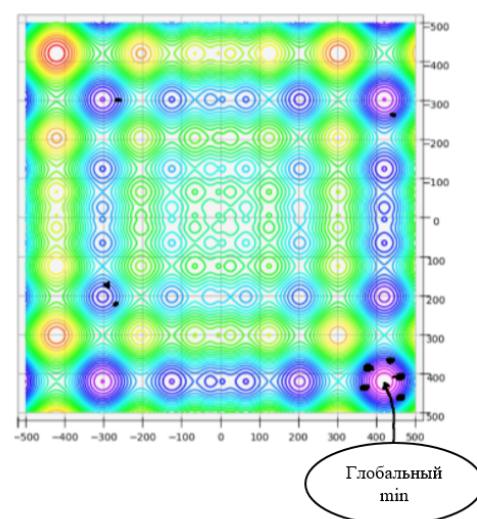
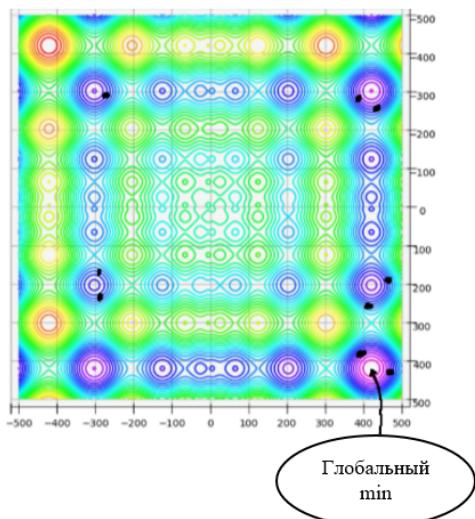
1 итерация:



2 итерация:



i – я итерация



Алгоритм оптимизации роем частиц.

1. Генерируем случайные распределения роя частиц.
 - 2.1. Поиск лучшего значения $f(x_i), \forall i = 1, \dots, N$, где N -количество частиц.
 - 2.2. Выбор $f_i \rightarrow_{\forall i} \min$.
 - 2.3 Коррекция направления движения и скорость частиц.
 - 2.3. Очередное перемещения частиц.
3. Если критерий остановки выполняется, то идем в пункт 4; если не выполняется, то идем в пункт 2.1.
4. Вывод результата оптимизации.

Критерии остановки итерационного процесса по t_i :

1. Неизменность целевой функции.
2. Роение частиц в некоторой области (в идеале в одну точку).
3. Достижение заданного значения целевой функции.
4. Достижение заданного номера итерации.

Расчет скорости и направления движения частиц роя:

$$\text{Классические формулы: } V_{i,t+1} = V_{i,t} + \underbrace{\varphi_p r_p (p_{i,t} - x_{i,t})}_{\text{“помочь” } i\text{-й частице}} + \underbrace{\varphi_g r_g (g_t - x_{i,t})}_{\text{“глобальная” память всего роя}}$$

$V_{i,t+1}$ – вектор скорости i -й частицы в момент времени $t + 1 = t + \Delta t, \Delta t = 1$;

$V_{i,t}$ – вектор скорости i -й частицы в момент времени t (t -я итерации);

$p_{i,t}$ – вектор координат лучшего решения, найденного i -й частицей;

g_t – вектор координат лучшего решения, найденного всеми частицами на t -ом шаге алгоритма;

r_p, r_g – случайные числа в интервале $(0,1)$;

φ_p, φ_g – весовые коэффициенты.

Модификация LBEST.

$V_{i,t+1} = V_{i,t} + \varphi_p r_p(p_{i,t} - x_{i,t}) + \varphi_l r_l(l_{i,t} - x_{i,t})$, где $l_{i,t}$ – вектор координат лучшего решения, найденного среди соседних частиц.

Учёт инерции (понижение скорости частиц).

$V_{i,t+1} = \omega(t) * V_{i,t} + \varphi_p r_p(p_{i,t} - x_{i,t}) + \varphi_g r_g(g_t - x_{i,t})$, где $\omega(t)$ – коэффициент инерции.

Примеры выбора $\omega(t)$:

1. $\omega(t) = c$ – постоянное значение торможения частиц.
2. Линейное убывание $\omega(t) = \omega_{max} - \frac{\omega_{max}-\omega_{min}}{t_{max}} * t$.
3. Сигмоид $\omega(t) = \frac{\omega_{нач}-\omega_{кон}}{1+e^{-n(t-n*t_{max})}} + w_{кон}$, $n = 10^{\log(t_{max})}$.
4. И другие ...

Ограничение скорости частиц с помощью коэффициента сужения.

$V_{i,t+1} = K \left(v_{i,t} + \varphi_p r_p(p_{i,t} - x_{i,t}) + \varphi_g r_g(g_t - x_{i,t}) \right)$, где $K = \frac{2\alpha}{\varphi_p + \varphi_g - 2}$, $\varphi_p + \varphi_g > 4$, $\alpha \in (0,1)$.

Замечание.

Обычно $\alpha \approx 0.9$

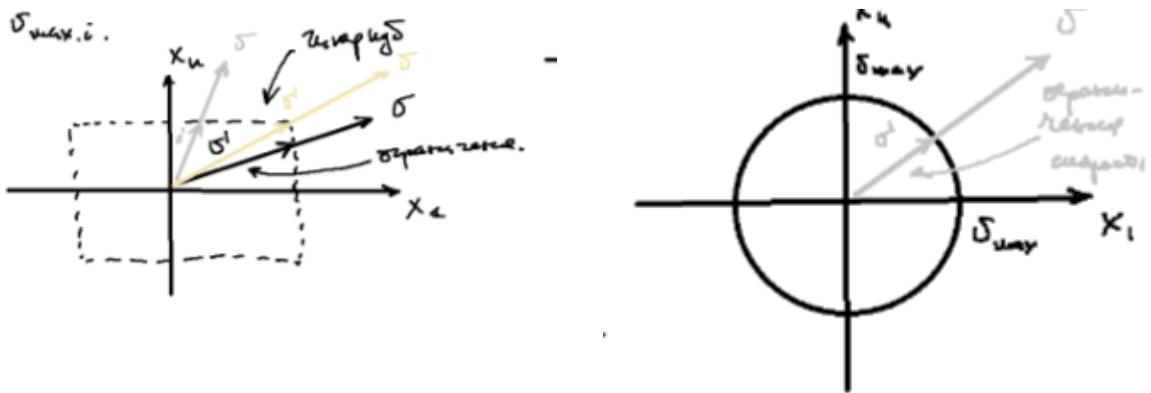
Другие способы ограничения скорости:

По i-ой координате:

Задается максимальные (V_{max}, i) значения скорости по каждой i-ой координате и в случае превышения скорости частицы по соответствующей координате, то значение скорости по этой координате задается равным $V_{max,i}$

По модулю скорости:

Задается V_{max} – модуль скорости, если V превышает V_{max} , то скорость занижается до V_{max}

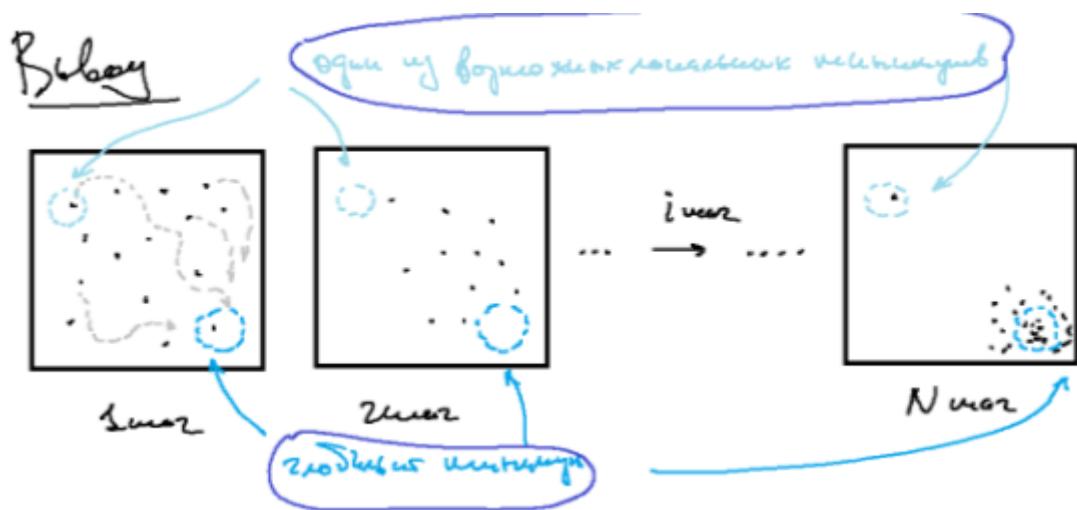


Расчет координат частиц.

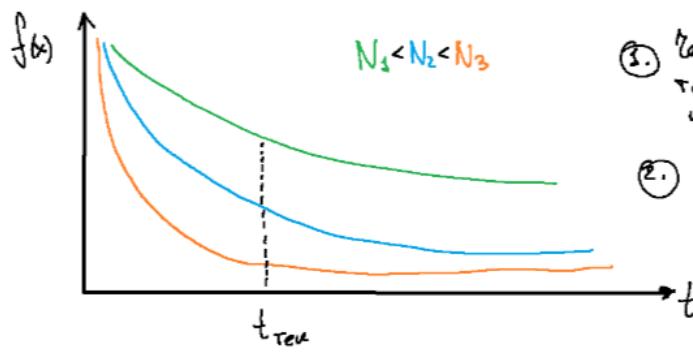
$$x_{i,t+1} = x_{i,t} + V_{i,t+1} * \Delta t, \text{ положим } \Delta t = 1 \text{ ед. времени ; } \Delta t \equiv 1 \text{ сек.}$$

$$x_{i,t+1} = x_{i,t} + V_{i,t+1} * 1 \text{ сек.}$$

Вывод.



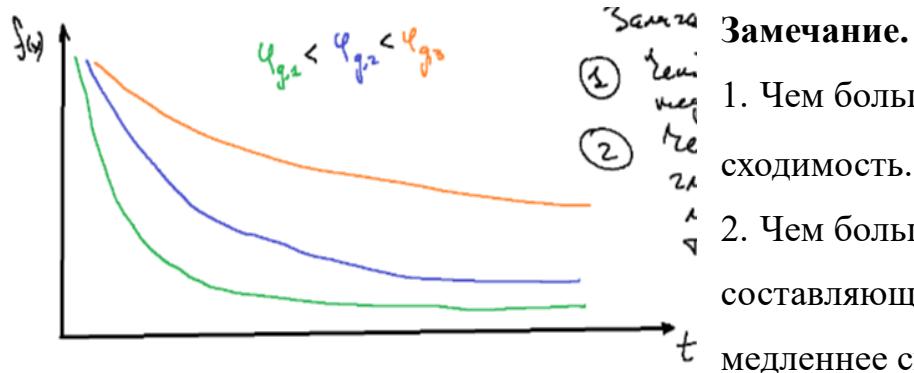
Влияние количества частиц на сходимость.



Замечание.

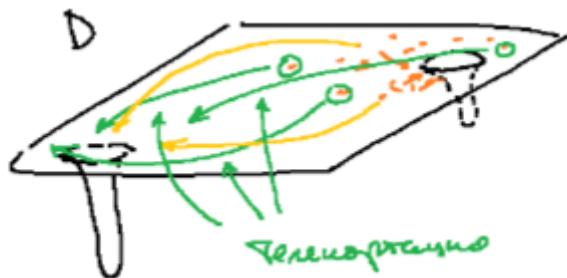
- ① 1. Чем больше частиц, тем меньше итераций.
2. Чем больше частиц, тем больше “трудозатрат” требуется для вычисления целевой функции.

Влияние параметра φ_g на сходимость.

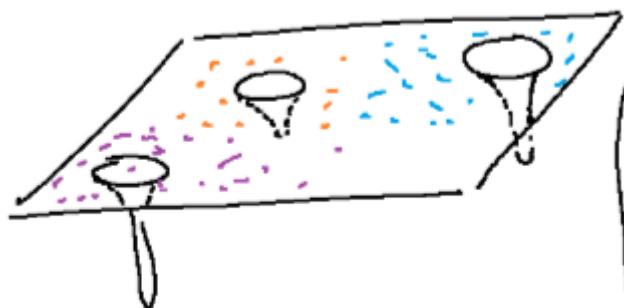


Модификации метода роя частиц.

1. Случайные перемещения частиц (телеportация)



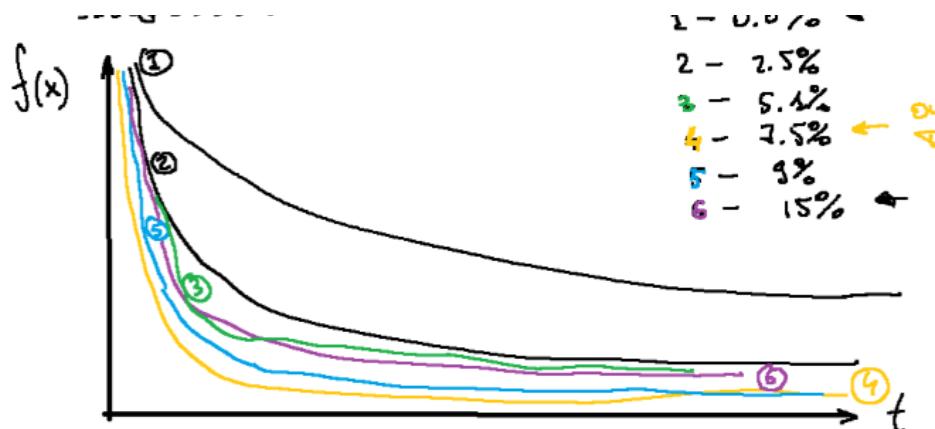
2. Алгоритм с отрицательным подкреплением
3. Совместное использование с другими алгоритмами оптимизации:
 - Рой частиц + какой-то градиентный метод;
 - Рой и дополнительные частицы;
 - Рой + генетический алгоритм.
4. Разделение роя на группы



Замечание.

Популяционные алгоритмы на гладких и унимодальных функциях проигрывают классическим градиентным методам.

Влияние вероятности случайной величины телепортации на сходимость.



1. 0.0% – нет телепортации
2. 2.5%
3. 5.1%
4. 7.5% – оптимальное значение телепортации
5. 9%
6. 15% – самая интенсивная степень телепортации частиц

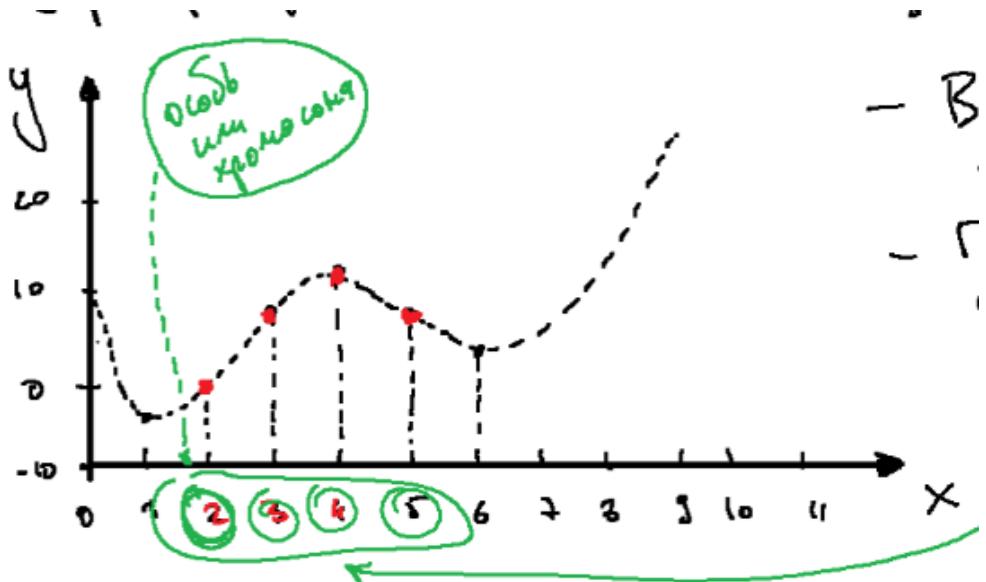
Преимущества и недостатки метода роя частиц:

- + : простота реализации.
- : зависимость от φ_p и φ_g .

Генетические алгоритмы.

Пример.

$$f(x) = 5 - 24x + 17x^2 - \frac{4}{3}x^3 + \frac{1}{4}x^4.$$



- Выберем случайные числа на отрезке $[0, 7]$: $\{2,3,5,4\}$
- Переводим в двоичные числа: $\{010, 011, 101, 100\}$, (каждое число это хромосома) (исходная популяция (пробное решение)).

Замечание.

Процесс размножения происходит, например, следующим образом.

Пусть есть $\{111|111\}$ $\{000|000\}$, где $|$ – точка кроссинговера (разрыва хромосомы) равна 3.

После обмена хромосомами имеем следующий результат: $\{111|000\}$, $\{000|111\}$.

Вычислим приспособленность начальной популяции:

Значения	Двоичные значения	Приспособленность
2	010	-0.35
3	011	7.3
5	101	7.9
4	100	10.33

Основная идея генетических алгоритмов – борьба за выживание, борьба за минимизацию целевой функции.

Определение.

Одно пробное решение в двоичной форме называется особью или хромосомой.

Определение.

Набор всех пробных решений называется популяцией.

Процесс размножения.

1. Необходимо на основе исходной популяции создать новую, более приспособленную к целевой функции.
2. Необходимо на начальном этапе сформулировать брачные пары для скрещивания.
3. Ставим в соответствие каждой особи случайное число из диапазона [1, 4].
4. Какие-то из членов исходной популяции будут участвовать в размножении, какие-то нет, кто-то будет участвовать неоднократно.

Процедура одноточечного кроссинговера.

id	Исходная популяция	Случайный id партнера	Сгенерированный партнер	Точка кроссинговера или разрыва хромосомы
1	010	1	010	1
2	011	4	100	1
3	101	3	101	2
4	100	1	010	2

Процесс скрещивания.

1. Точка разрыва хромосомы: 1

$$0|10 \rightarrow 0|00$$

$$1|00 \quad 1|10$$

2. Точка разрыва хромосомы: 2

$$10|1 \rightarrow 10|0$$

01|0 01|1

Определение.

Мутации – случайные изменения, полученные в результате скрещивания хромосом.

Алгоритм мутирования хромосом.

1. Задаем вероятность мутации, пусть вероятность мутации равна 0.3.
2. Для каждого потомка (ребенка) возьмем случайное число на отрезке $[0, 1]$:
 - a. Если это число меньше 0.3, то инвертируем случайно выбранный ген. (замена 0 на 1 или наоборот).
 - b. В противном случае ничего не делаем.

Тогда применим алгоритм мутации к новой популяции. $\{000, 110, 100, 011\}$ (потомки).

Результат мутации потомков:

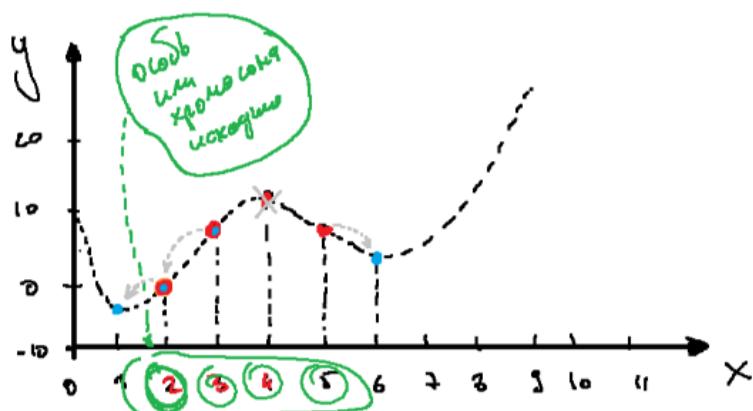
Особи потомки (Хдо)	Случайное число из $[0, 1]$	Случайный выбранный ген для мутации	Потомок (ребенок) после мутации (Хпсле)	Приспособленность потомка до мутации f (Хдо)	Приспособленность потомка после мутации (Хпсле)
000	01	3	001	5	-5.42
110	0.7	-	110	5	5
100	0.9	-	100	10.33	10.33
011	0.25	1	111	7.25	12.58

Из таблицы выше видно, что мутации могут улучшить (см. первый потомок в таблице) или ухудшить (см 4 потомок в таблице) приспособленность особи (хромосомы) потомка (ребенка).

В результате скрещивания хромосом происходит обмен частями, то есть младшими разрядами в двоичном представлении. При том возможен обмен и старшими разрядами.

- Скрещивание приводит к относительно небольшим изменениям пробных решений.
 - Мутации метода приводят к существенным изменениям пробных решений.
- Генерацию новую популяцию: отберем 4 наиболее приспособленных особи из числа «старых» и «новых» особей и соответственно особей потомков.

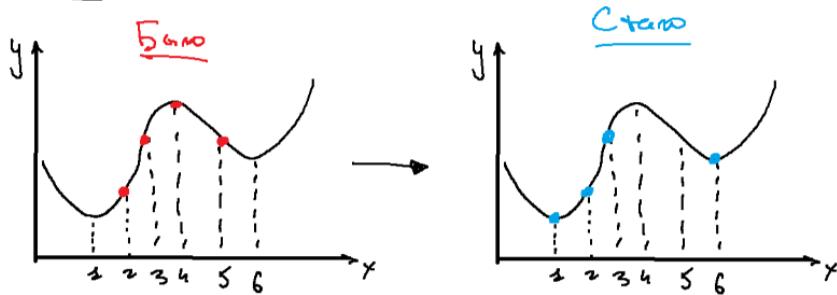
Получается, что новая популяция состоит из набора: {000, 110, 011, 001}.



Формирование новой популяции из особей-родителей и особей-потомков:

id	Особи	Приспособленность	Новая популяция	Приспособленность особей новой популяции
1	010	-0.33	001	-8.42
2	011	7.25	010	-0.33
3	101	7.92	110	5
4	100	10.33	011	7.25
5	001	-9.42		
6	110	5		
7	100	10.33		
8	111	12.58		

Время:



Основные понятия.

1. **Хеммингово расстояние** – используется для булевых векторов и равно числу различающих в обоих векторах компонент.
2. **Хеммингово пространство** – пространство булевых векторов с введенной метрикой Хемминга.
3. **Хромосома** – вектор или строка из каких-либо чисел. Если вектор представлен бинарной строкой, то он получен либо с использованием двоичного кодирования, либо кода Грея.
4. **Индивидуум** – набор хромосом, решения задачи.
5. **Кроссинговер** – операция разрыва хромосомы и обмена частями.
6. **Мутация** – случайное изменение одной или нескольких позиций в хромосоме.
7. **Инверсия** – изменение порядка следования битов хромосомы или её фрагмента.
8. **Популяция** – набор индивидуумов.
9. **Приспособленность** – функция, для которой необходимо найти экстремум.
10. **Локус** – позиция гена в хромосоме.
11. **Аллель** – совокупность подряд идущих генов.
12. **Эпистаз** – влияние гена на пригодность индивидуума в зависимости от значения гена, присутствующего в другом месте.

Основные этапы генетического алгоритма:

1. Генерируем начальную популяцию из n хромосом.
2. Вычислим для каждой хромосомы приспособленность ($f(x)$).

3. Выберем пару хромосом родителей с помощью одного из способов отбора.
4. Проводим кроссинговер двух родителей с некоторой заданной вероятностью, производим двух потомков.
5. Проводим мутацию потомков с заданной вероятностью.
6. Повторим 3-5 пока не будет сгенерировано новое поколение популяции содержащее n индивидуумов.
7. Повторяем 2-6 пока не будет достигнут критерий остановки.

Сопоставление терминологии.

Биологические	Искусственная терминология
Ген	– значение функции
Хромосома	– строка или x
Аллель	– возможное значение генов
Локус	– позиция в строке или координата x
Генотип	– фактическая структура
Фенотип	– множество параметров, альтернативное решение, декодирование структуры.
Эпистаз	– взаимодействие генов

Критерий остановки алгоритма.

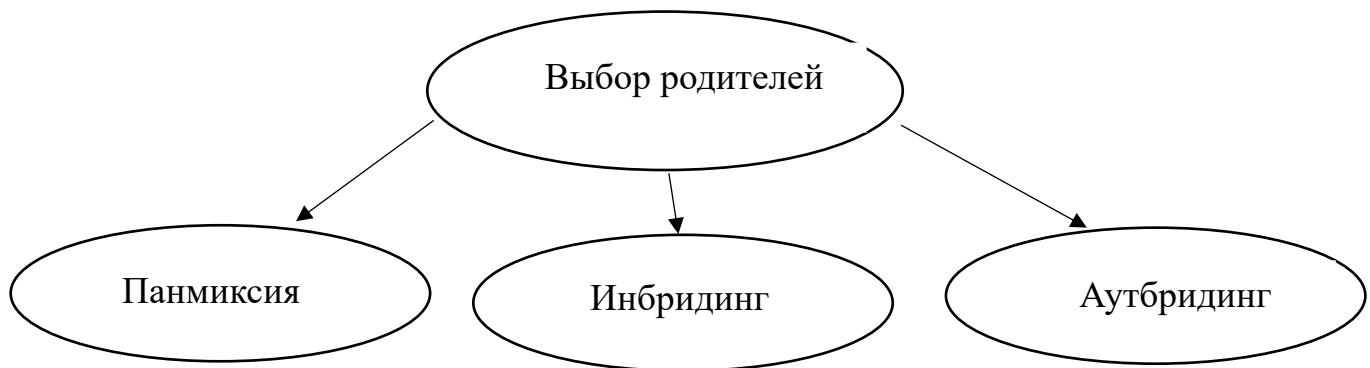
1. Количество популяций (итераций)
2. Сходимость популяции (неизменность целевой функции)

Под сходимостью понимается состояние, когда все строки или хромосомы не меняются.

Вывод.

Основными операторами генетических алгоритмов является: кроссинговер, мутация, выбор родителей, селекция.

Лекция 10. Генетические алгоритмы – продолжение.



1. Для \forall членов популяции сопоставлены случайные целые числа из интервала $[1, n]$, где n – количество особей в популяции.

2. Рассмотрим эти числа, как номера особей, участвующих в скрещивании.

3. Кто-то из членов популяции будет участвовать, кто-то нет.

Замечание: Какие-то члены популяции могут неоднократно

1. Первого родителя выбираем случайным образом, второй родитель выбирается ближайший к первому.

2. Расстояния до родителя выбираем в соответствии с метрикой данного пр-ва:

- расстояние Хемминга (для бинарных строк);
- Евклидово расстояние для вещественных векторов.

1. Первого родителя выбираем случайно, второй родитель выбирается максимально удаленный от первого.

2. Расстояния до родителя выбираем в соответствии с метрикой данного пр-ва:

- расстояние Хемминга (для бинарных строк);
- Евклидово расстояние для вещественных векторов.

скрещиваться с другими членами популяции.

Замечание 1:

Универсальный, простой способ, но есть недостаток: эффективность алгоритма снижается с ростом численности популяции.

Замечание 2:

а) Характеризуется характерной концентрацией результатов поиска критических точек в локальных узлах, это приводит к разбиению решения на множества вокруг подозрительных точек, или вокруг локальных экстремумов.
б) Используется для предупреждения ситуаций сходимости алгоритма около уже найденных решений и позволяет локализовать новые области с новыми подозрительными точками или множествами

Замечание 3.

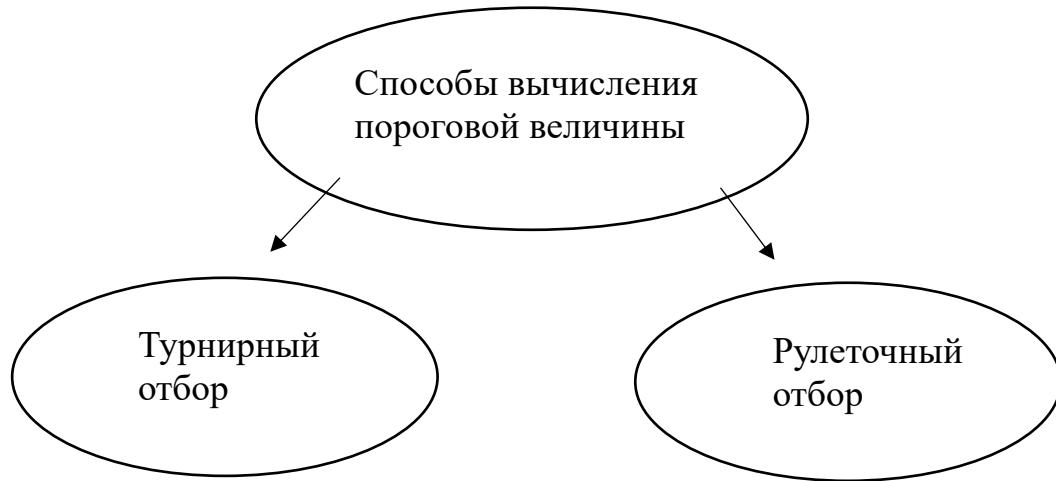
Идея селекции заключается в том, что родителем могут быть только те особи, значение приспособленности (целевой функции, фитнес функции) которых не меньше пороговой величины. (Недостатки: быстрая сходимость).

Замечание 4.

Из-за быстрой сходимости выбор родительской пары не подходит тогда, когда ставится задача определения нескольких экстремумов, т.к., как правило, алгоритм быстро сходится к одному решению.

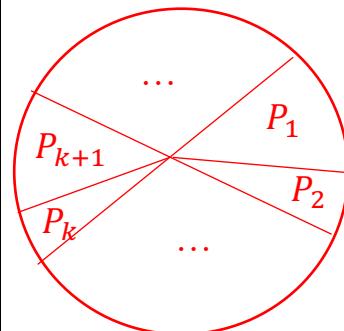
Замечание 5.

Данный недостаток компенсируется различными механизмами торможения алгоритма:



1. Из популяции, состоящей из N особей выбираем t особей случайно, где t – численность турнира. Далее лучшая из особей записывается в стек.
2. Операция пункта 1 повторяется N раз.
3. Особи в стеке используются для скрещивания (как правило следующим образом – см. схема турнирного отбора)

1. Особи выбираются при помощи N запусков «рулетки», при этом N – это размер популяции. Колесо рулетки содержит по одному члену популяции, размер i -го сектора пропорционален вероятности попадания в новую популяцию.



$$P_k = \frac{f(x_k)}{\sum_{i=1}^N f(x_i)}$$

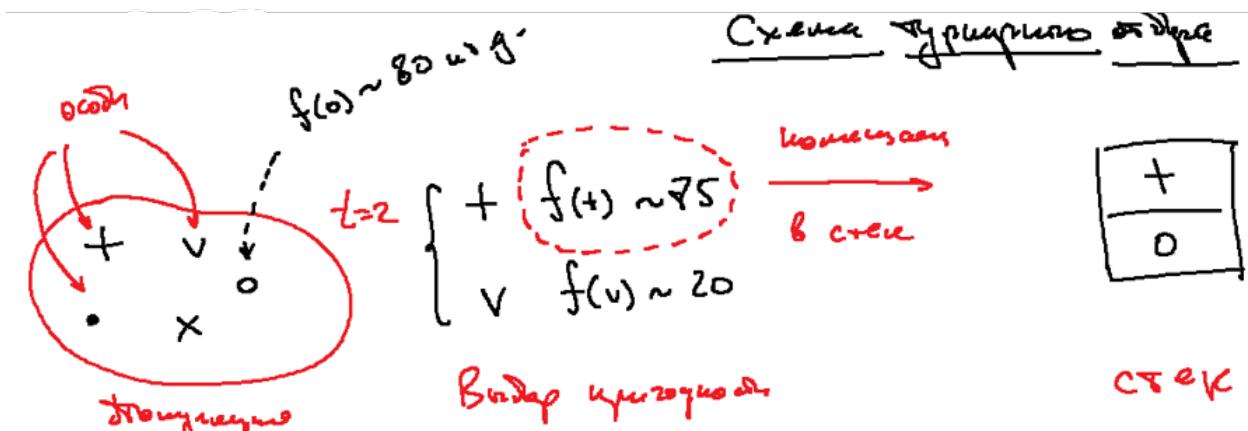
где $f(x_i)$ – пригодность x_i особи,

$$N_i = P_i \cdot N$$

Далее см.

пример расчета рулетки.

Схема турнирного отбора.



Пример расчета рулетки.

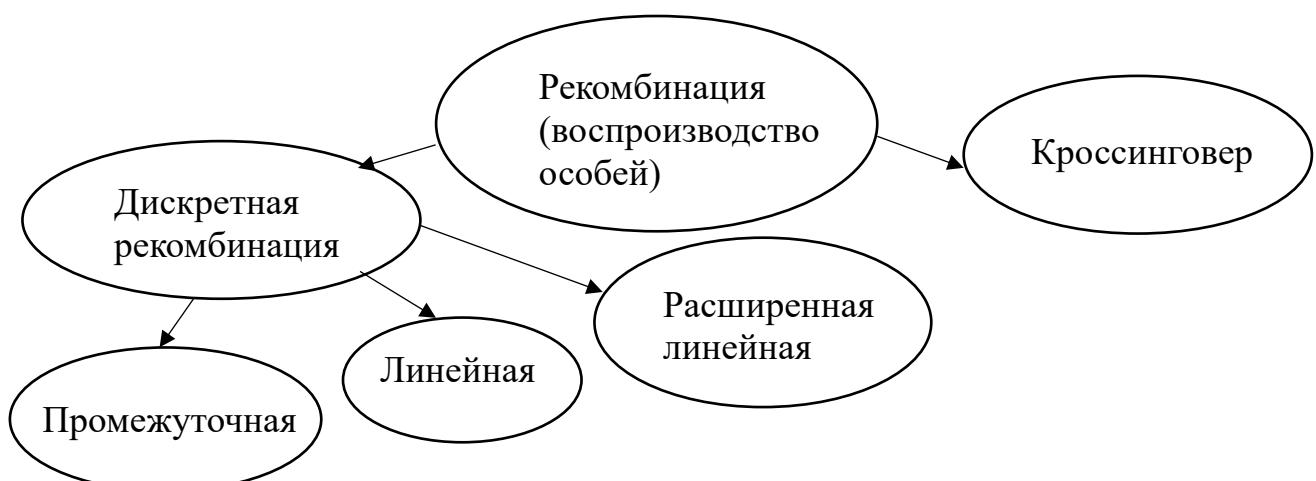
Популяция из 5 особей:

Популяция	Пригодность особи	Вероятность выбора
-----------	-------------------	--------------------

Особь $x_1 \rightarrow$	$f(x_1) \approx 52$	$P_1 = \frac{52}{200} \sim 36\%$
Особь $x_2 \rightarrow$	$f(x_2) \approx 85$	$P_2 = \frac{85}{200} \sim 43\%$
Особь $x_3 \rightarrow$	$f(x_3) \approx 37$	$P_3 = \frac{37}{200} \sim 19\%$
Особь $x_4 \rightarrow$	$f(x_4) \approx 4$	$P_4 = \frac{4}{200} \sim 1\%$
Особь $x_5 \rightarrow$	$f(x_5) \approx 22$	$P_5 = \frac{22}{200} \sim 11\%$

Определение.

Рекомбинация (воспроизведение особей) – процедура, применяемая после отбора родителей для получения новых особей потомков.



Замечание.

Дискретная рекомбинация применяется к хромосомам с вещественными генами.

Принцип дискретной рекомбинации.

	ген1	ген2	ген3
Две особи	$x_1:$	10	30
	$x_2:$	100	2

Генерация перемешивания генов:

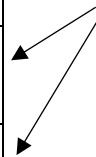
особь	ген 1	ген 2	ген 3
x_3	2	2	1
x_4	1	2	1

Новые особи:

Новый потомок 1:

$$x_3: 100 \quad 2 \quad 5$$

Генерируем случайное число в диапазоне $[1, 2]$, соответствующее номеру особи



Новый потомок 2:

$$x_4: 10 \quad 2 \quad 5$$

Промежуточная рекомбинация.

Замечание.

Используется для вещественных переменных (генов), НЕ битовых!!!

Идея:

$$x_3 = x_1 + \alpha \cdot (x_2 - x_1) \quad (*)$$

где

- x_3 – потомок;
- x_1 – родитель 1, x_2 – родитель 2;
- α – случайное число, $\alpha \in [-d; 1 + d]$, $d \geq 0$; из классических реализаций следует, что наиболее лучшее воспроизведение достигается на $\alpha \in [0.20; 0.35]$.

Важно: α выбирается для каждого гена отдельно.

Пример.

Родители	ген1 ген2 ген3			выбор α $\xrightarrow{\alpha \in [-0.25; 1.25]}$	x_1	α_1	α_2	α_3
	x_1	12	25		x_1	0.5	1.1	-0.1
	x_2	116	4	34	x_2	0.1	0.8	0.5

Получим потомков по формуле (*):

особь	ген 1	ген 2	ген 3
x_3	$12 + 0.5 \cdot (116 - 12)$ = 64	$25 + 1.1 \cdot (4 - 25)$ = 1.9	$7 - 0.1 \cdot (34 - 7)$ = 4.3
x_4	$12 + 0.1 \cdot (116 - 12)$ = 22.4	$25 + 0.8 \cdot (4 - 25)$ = 8.2	$7 + 0.5 \cdot (34 - 7)$ = 20.5

Данный способ иногда называют **дифференциальным скрещиванием**.

Линейная рекомбинация.

Отличается от промежуточной тем, что α выбирается для каждого потомка один раз.

$x_1 \rightarrow \alpha = 0.5$
 $x_2 \rightarrow \alpha = 0.1$ Тогда гены созданных потомков имеют вид:

особь	ген 1	ген 2	ген 3
x_3	$12 + 0.5 \cdot (116 - 12)$ = 64	$25 + 0.5 \cdot (4 - 25)$ = 14.5	$7 + 0.5 \cdot (34 - 7)$ = 20.5
x_4	$12 + 0.1 \cdot (116 - 12)$ = 22.4	$25 + 0.1 \cdot (4 - 25)$ = 22.9	$7 + 0.1 \cdot (34 - 7)$ = 9.7



Одноточечный кроссинговер.

- Пусть заданы две родительские особи:

$$X = \{x_i : i \in [0 ; l]\}, \quad Y = \{y_i : i \in [0 ; l]\}$$

X – первая особь, Y – вторая особь

x_i, y_i – i -ая хромосома первой и второй особи соответственно

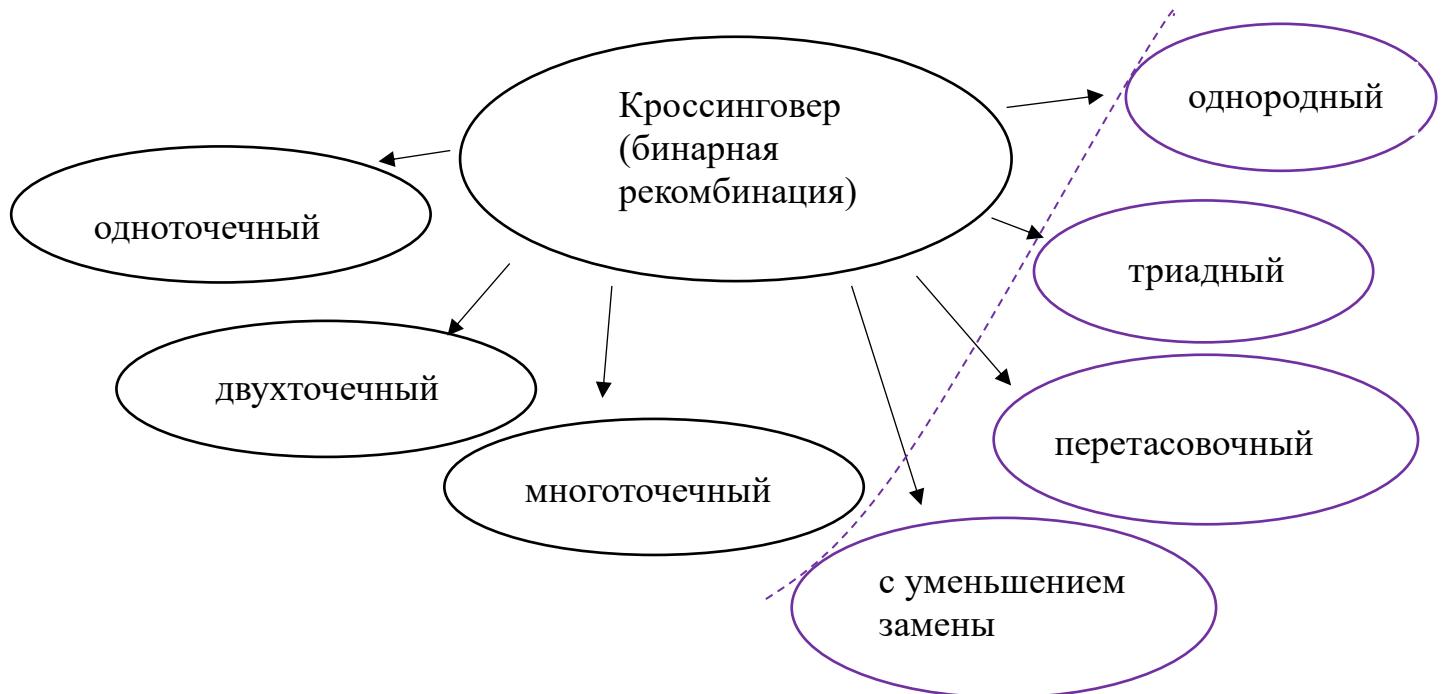
- Случайным образом выделяется точка внутри x_i хромосомы – точка разрыва, далее хромосомы делятся пополам и обмениваются частями.

X, Y – особи-родители, \tilde{X}, \tilde{Y} – потомки

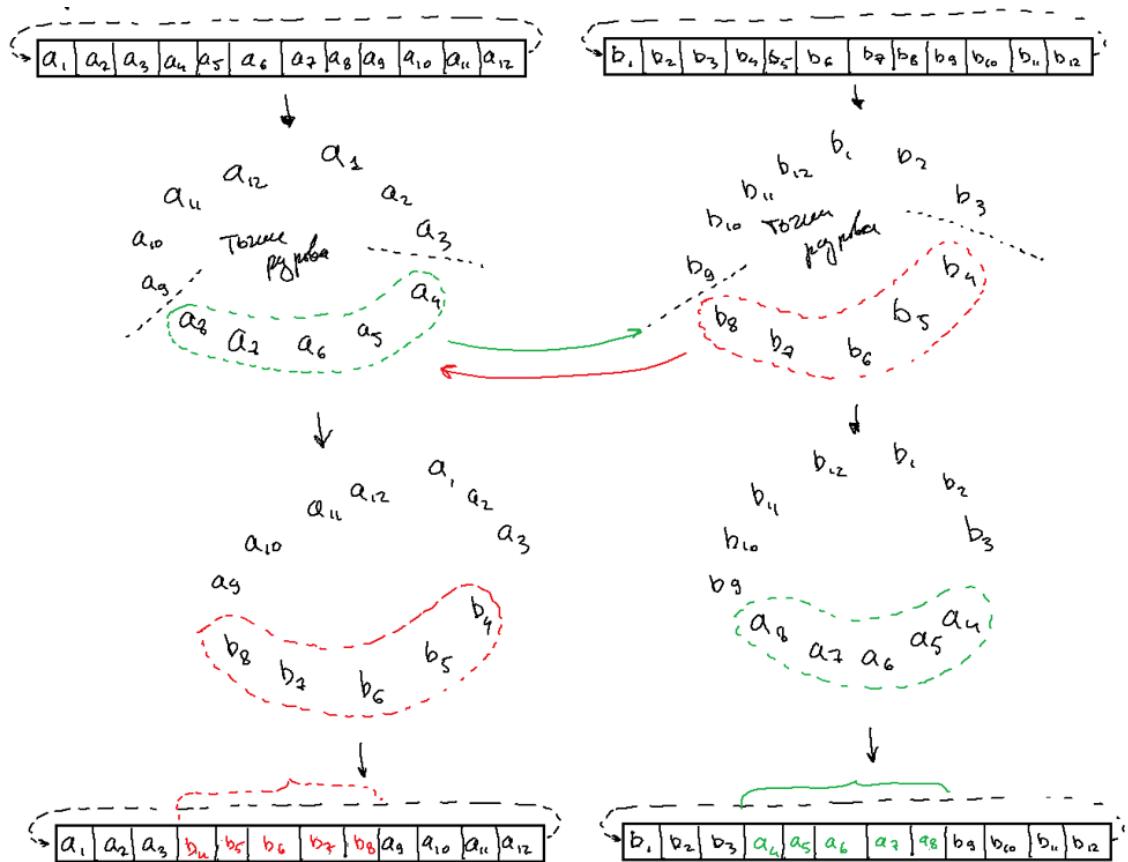
$$\begin{aligned} X &= \{x_1, x_2, x_3, \dots, x_{n-1}, x_n, \dots, x_m\} \rightarrow \tilde{X} = \{x_1, x_2, x_3, \dots, x_{n-1}, y_n, \dots, y_m\} \\ Y &= \{y_1, y_2, y_3, \dots, y_{n-1}, y_n, \dots, y_m\} \rightarrow \tilde{Y} = \{y_1, y_2, y_3, \dots, y_{n-1}, x_n, \dots, x_m\} \end{aligned}$$

↑
↑
Точка кроссинговера

Виды и модификации кроссинговера.



Двухточечный кроссинговер.



Хромосомы представляются в виде циклов или колец, которые формируются путем соединения концов линейной хромосомы.

Замечание.

На данный момент известно, что двухточечный кроссинговер работает лучше одноточечного.

Многоточечный кроссинговер. (Multipoint crossover)

1. Выбираем m точек разреза гена $K_i \in \{1, 2, \dots, N\}$, $i = 1 \div m$, N – количество генов (переменных) в особи.
2. Точки K_i выбираются случайно, без повторений и сортируются по возрастанию.
3. При кроссинговере происходит обмен участками хромосом, которые ограничены точками разреза, таким образом получают двух потомков.

Замечание.

Участок особи с первым геном до первой точки разреза в обмене не участвует.

Пример.

Особь 1	0	1	1	1	0	0	1	1	0	1	0
Особь 2	1	0	1	0	1	1	0	0	1	0	1

Точки разреза кроссинговера, где $m = 3$ выбраны случайным образом

Перемешиваем гены, получаем двух потомков:

Потомок 1	0	1	1	0	1	1	1	1	0	1	1
Потомок 2	1	0	1	1	0	0	0	0	1	0	0

Замечание.

- Применение многоточечного кроссинговера требует введения нескольких переменных (точек разреза).

- Для воспроизведения выбираются особи, которые наиболее приспособлены.

Однородный кроссинговер. (Uniform crossover)

1. Создается маска, или правило, или схема особи, по которой в соответствии с правилом генерации генерируется потомок. Правило генерации потомка см. ниже.
2. Маска (правило, схема) имеет ту же длину, что и скрещивающиеся особи.
3. Правило генерации особи на основе маски:

если в маске 1 → в потомке выбирается ген особи 1.

если в маске 0 → в потомке выбирается ген особи 2.

Пример.

Родители

Особь 1	1	0	1	1	0	0	0	1	1	0	1
Особь 2	0	0	0	0	0	1	1	1	1	0	0

Маска со случайными битами

Маска для потомка 1	0	1	0	1	1	0	0	1	1	0	0
Маска для потомка 2	0	1	0	1	1	0	1	1	0	0	0

Потомки

Потомок 1	0	0	0	1	0	1	1	1	1	0	0
Потомок 2	0	0	0	1	0	1	0	1	1	0	0

Замечание.

Однородный кроссинговер похож на многоточечный, но строка случайных битовых значений в нем длиннее. Это гарантирует, что в потомках будут чередоваться короткие строки особей-родителей.

Замечание.

Иногда такой вид кроссинговера называют **унифицированным**.

Триадный кроссинговер (модификация однородного).

Основное отличие от однородного кроссинговера следующее:

1. Выбираем пару родителей из остальных членов популяции случайным образом.
2. Случайным образом выбираем третью особь из оставшихся членов популяции.
3. Третья особь далее используется в качестве маски.
4. 10% – 20% генов маски мутируют.
5. Гены особи 1 сравниваются с генами маски. Если гены одинаковы, то они передаются первому потомку, иначе на соответствующие позиции хромосомы потомка переходят гены второго родителя.

Замечание.

Генотип второго потомка отличается от генотипа первого тем, что на тех позициях, где у первого потомка стоят гены первого родителя, у второго потомка стоят гены второго родителя, и наоборот.

Перетасовочный кроссинговер.

1. Особи для кроссинговера случайным образом обмениваются генами.
2. Выбор точки одноточечного кроссинговера и обмен хромосомами.
3. После скрещивания созданные потомки опять тасуются.

Замечание.

При каждом шаге кроссинговера создаются не только новые потомки, но и модифицируются (путем перетасовки) родители. При этом старые удаляются.

Замечание.

Этот метод позволяет сократить число операций по сравнению с однородным кроссинговером.

Кроссинговер с уменьшением замены.

1. Оператор уменьшения замены ограничивает кроссинговер, чтобы всегда по возможности создавать новые особи.
2. Это достигается за счет ограничения на выбор точки разреза: точки разреза должны появляться там, где гены различны.

Ссылка на доп. литературу:

<https://mathmod.asu.edu.ru/images/File/ebooks/GAfinal.pdf>

Лекция 11. Мутации.

Замечание.

Мутации необходимы для “выведения” популяции из локального экстремума, а также препятствия быстрой сходимости к точкам локального минимума.

Вид мутации: $x_1 x_2 \dots x_{n-1} x_{n+1} \dots x_m \xrightarrow{\text{мутация}} x_1 x_2 \dots x_{n-1} x'_n x_{n+1} \dots x_m$

Замечание.

p_m – вероятность мутации: $p_m \ll 1$ ($p_m \in [0,1]$), два варианта:

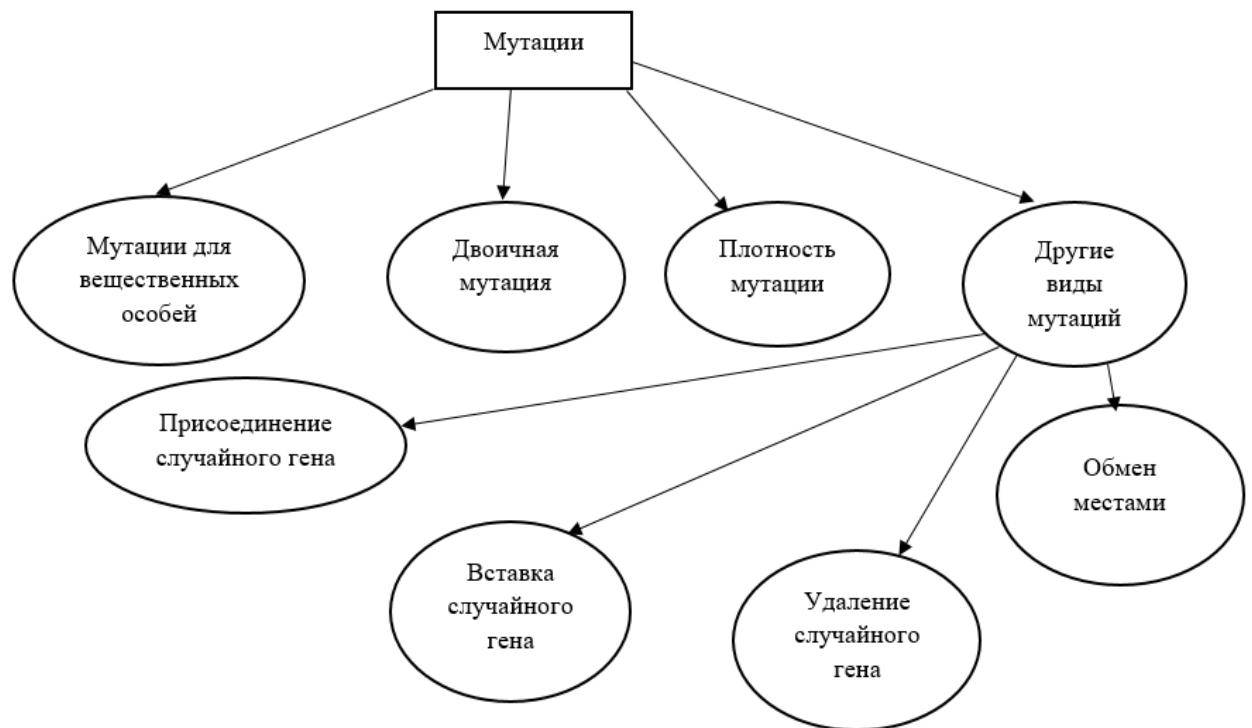
- $p_m = \text{const}$,
- $p_m \neq \text{const}$, p_m зависит от количества генов.

Оптимальное значение p_m :

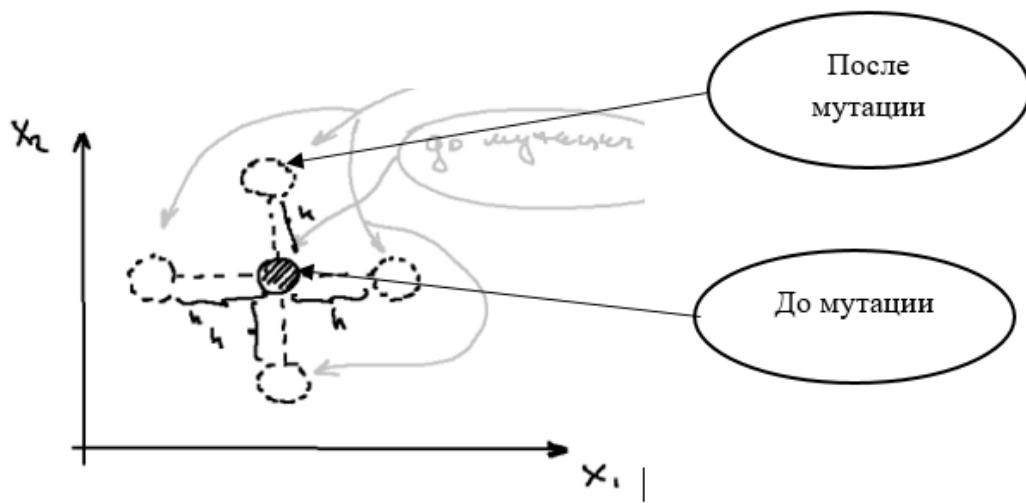
- p_m – фиксировано

Хороший результат для задач с унимодальными функциями.

- p_m – адаптируется
- Подходит для мультимодальных функций.



Мутация для вещественных особей.



1. Определение h – шага мутации, где под мутацией мы предполагаем некоторое число, на которое изменяем значение гена.
2. Выбрав оптимальное размер h (шага мутации), при малых h – алгоритм будет сходится медленно, в противном случае возможны потери решений.
3. $x_i^{\text{нов}} = x_i^{\text{стар}} \pm (\alpha * \delta)(h)$ (\pm – случайным образом),

$$\alpha = \frac{1}{2}x \text{ [интервал изменения переменной]},$$

$$\delta = \sum_{i=1}^m a(i) * 2^{-i}, a(i) = \begin{cases} 1 & \text{с вероятностью } \frac{1}{m} \\ 0 & \end{cases}, \text{ где } m \text{ – параметр}$$

Двоичная мутация.

Условие.

Особи должны быть кодированы двоичным кодом или методом Грея.

1. Проводим случайную инверсию гена: $0 \leftrightarrow 1$;
2. Эффективность мутации зависит от способа
 - Двоичного кодирования
 - Код Грея

Замечание.

В различных случаях двоичный код или метод Грея ведут себя по-разному.

Плотность мутации.

1. Проводится мутация каждого гена потомка с заданной вероятностью.
2. Кроме вероятности мутации самого потомка используется вероятность мутации каждого его гена.

Замечание.

Как правило выбирают, чтобы в среднем мутировало 1-10% генов.

Другие виды мутаций.

Пусть $t_i: t = t_1, \dots, t_n$; данный вид мутаций рассматривается в задачах с разным количеством хромосом.

1. Присоединение случайного гена: $t \rightarrow t_1, \dots, t_k, s$.
2. Вставка случайного гена: $t \rightarrow t_1, \dots, t_{i-1}, s, t_{i+1}, \dots, t_k$.
3. Удаление случайного гена: $t \rightarrow t_1, \dots, t_{i-1}, t_{i+1}, \dots, t_k$.
4. Обмен местами в последовательности двух соседей одного случайного выбранного гена: $t \rightarrow t_1, \dots, t_{i+1}, s, t_{i-1}, \dots, t_k$.

Вывод.

Мутация 1 – есть частичный случай мутации 2.

Для особей с фиксированным размером возможно применение в чистом виде только 4-й мутации, а 1 и 2 мутации мутации должны применяться в сог. 3.

Отбор особей.

Операторы отбора особей в новую популяцию.

1. Отбор удалением.
2. Элитарный отбор.
3. Отбор вытеснением.
4. Метод Больцмана.

Отбор усечением.

1. Используется для отбора особи-родителя особи-потомка.

2. Сортируем их по возрастанию функции пригодности.
3. Выбор количества особей для скрещивания выбираем $T \in [0; 1]$, пара Т может быть больше 1, тогда Т – количество особей для скрещивания.
4. Среди особей, попавших в «пары» Т выбирается один и записывается в новую популяцию.
5. Процесс продолжается N раз.

Замечание.

Новая популяция будет состоять из особей с высокой степенью пригодности, при этом одна и та же особь может встречаться несколько раз, некоторые особи могут не попасть в новую популяцию.

Замечание.

Так как в данной стратегии необходимо сортировать популяцию, то время то время сортировки может существенно повлиять на скорость поиска экстремума.

Элитарный отбор.

Создается промежуточная популяция: состоит из родителей и потомков. Каждый член данной популяции оценивается, далее выбираем N самых лучших, наиболее пригодные записываются в новое поколение.

Замечание.

Использование стратегии отбора лучшей особи не допускает потерю лучших решений.

Пример.

Пусть есть популяция, которая сошлась в локальном максимуме, применим мутацию, мутация вывела одну из точек в область глобального экстремума, то если использовать предыдущую стратегию, то есть вероятность, что в результате скрещивания лучшее решение будет потеряно.

Замечание.

Как правило, элитарный отбор комбинируют с другими способами. Некоторый процент особей отбирают элитарным отбором, а какой-то классическим способом.

Отбор вытеснением.

1. Выбор особи в новую популяцию зависит не только от функции пригодности, но и от формирующей новую популяцию особи, сравнивают хромосомы новой особи с хромосомами родителей, это позволяет получать более разнообразные экземпляры особей.
2. Предпочтение отдается особям с разными генотипами.

Замечание.

Нет потери найденных лучших решений.

Метод Больцмана.

Вероятность отбора зависит от параметра Т – температуры.

Вероятность попадания в новую популяцию:

$$p = \frac{1}{1 + e^{\frac{f(i) - f(j)}{T}}}, \text{ где } f(i) \text{ и } f(j) – \text{ значения целевой функции } i \text{ и } j \text{ особей, } i \text{ и } j – \text{ случайные номера особей.}$$

Если Р становится больше случайного числа из (0; 1), то в новую популяцию попадает особь $f(i)$, в противоположном случае $f(j)$.

$$p = \frac{e^{\frac{f(i)}{T}}}{\langle e^{\frac{f(i)}{T}} \rangle}, \text{ где } \langle \cdot \rangle – \text{ среднее значение на итерации с номером } i. \text{ Если } P$$

больше случайного числа на (0; 1), то особь $f(i)$ попадает в новую популяцию.

Замечание 1.

В методе Больцмана частицы с большими температурами соответствуют частицам первых поколений.

Замечание 2.

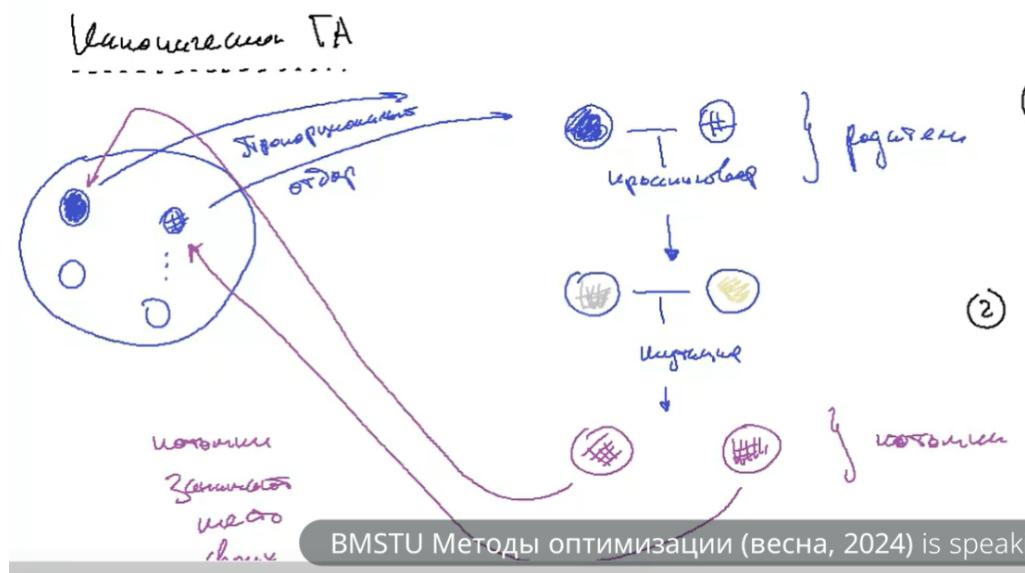
С ростом поколений температура снижается, вероятность отбора уменьшается и в новую популяцию попадает частица с меньшим значением функции приспособленности.

Разнообразие генетических алгоритмов.

Генетические алгоритмы:

- Канонически ГА;
- Генитор;
- Метод прерывистого равновесия;
- Гибридный алгоритм ;
- СНС;
- Генетический алгоритм с нефиксированным размером популяции.

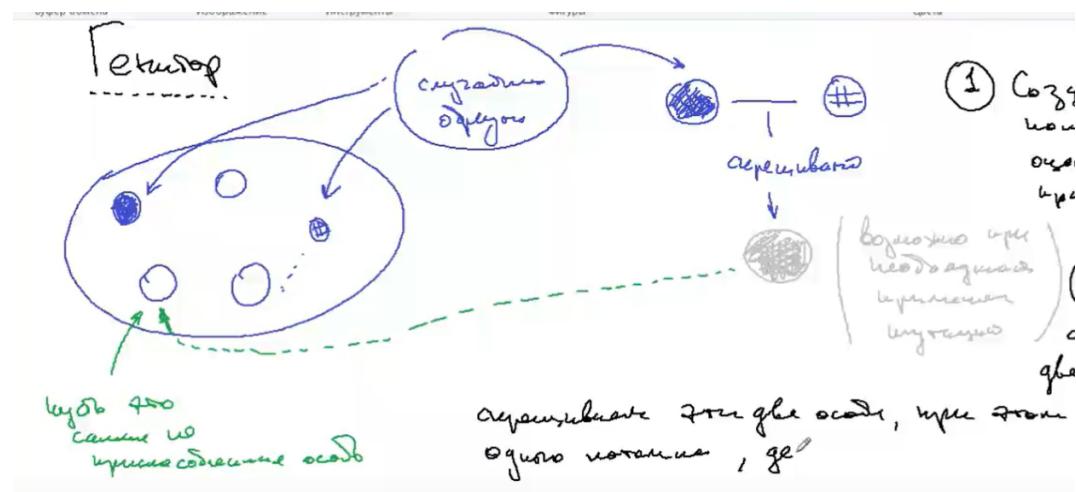
Канонические ГА.



1. Пусть популяция состоит из N хромосом с фиксированной разрядностью генов.
2. С помощью пропорционального отбора формируется промежуточный массив, далее из этого массива случайным образом выбирается два родителя.

3. Выполняем одноточечный кроссинговер, получаем два потомка, применяем оператор мутации с заданной вероятностью.
 4. Полученные потомки занимают место родителя.
 5. Процесс продолжается до достижения критерия остановки.

Генитор.

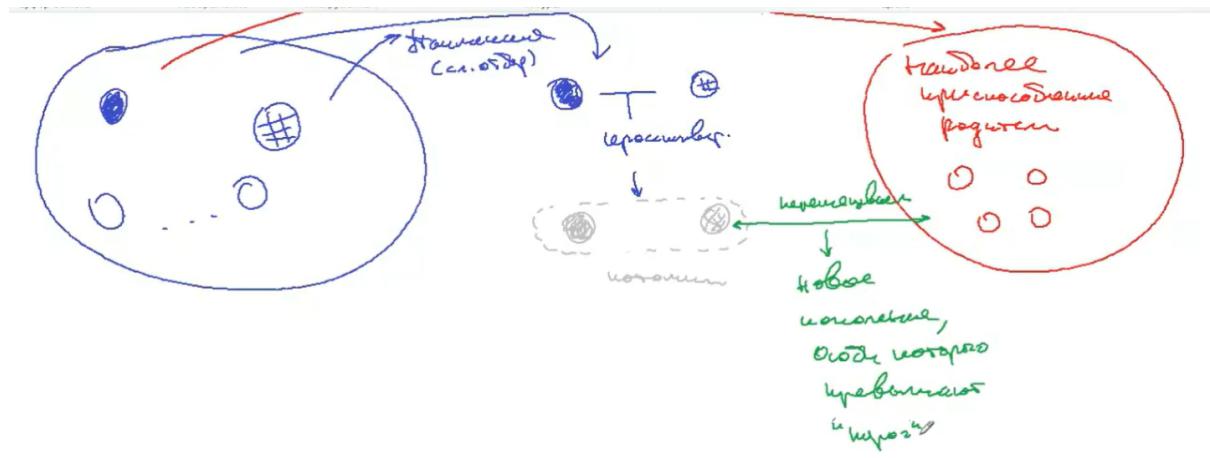


1. Создается популяция, оценивается приспособленность особей ($f(x), \forall i$).
 2. Выбираем случайным образом две особи, при этом получаем одного потомка, делаем оценку приспособленности новой особи.
 3. Новая особь занимает место наименее приспособленной особи в предыдущей популяции (не родителей новая особь заменяет!)
 4. На каждом шаге получаем одну особь до тех пор, пока не достигнем критерия остановки.

Замечание.

Возможно применение мутаций.

Метод непрерывного равновесия.



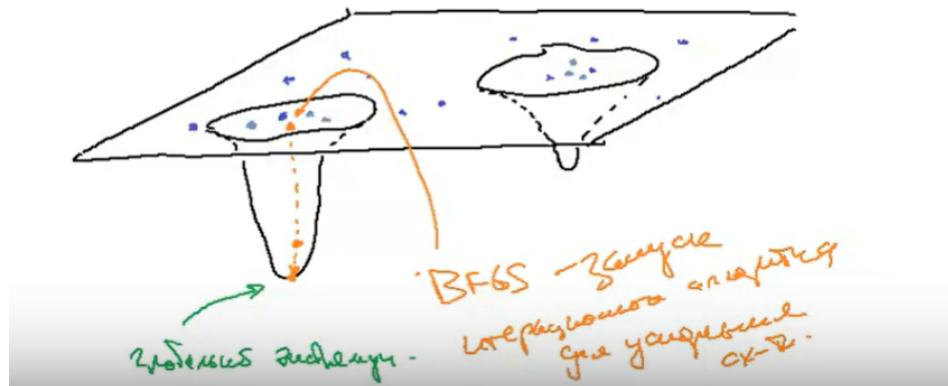
В данном методе после каждой генерации промежуточного поколения случайным образом перемешиваются особи популяций.

1. Используем случайный отбор, далее используем кроссинговер, перемешиваем получившихся потомков с наиболее пригодными родителями.
2. Из общей массы в новое поколение попадут только те особи, которые превышают порог пригодности.

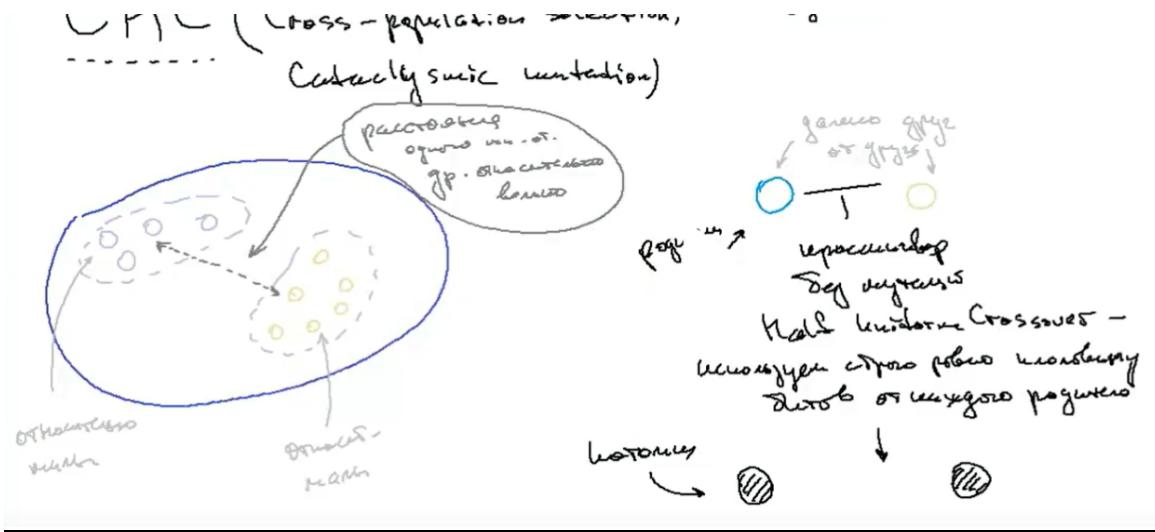
Гибридный алгоритм.

Под гибридным алгоритмом понимают некоторый ГА и классический алгоритм.

Например: Классический ГА + метод градиентного спуска; Классический ГА + BFGS.



CHC (Cross-population selection, Heterogenous recombination and Cataclysmic mutation)



Текст к рисунку: Расстояние одного множества относительно другого относительно велико; группы относительно малы.

Half uniform crossover – используем ровно половину битов от каждого родителя.

Замечание.

Данный алгоритм относительно быстро сходится из-за отсутствия мутаций.

1. Для размножения выбирается случайная пара, но не допускается чтобы между родителями было маленькое хеммингово расстояние.
2. Используем HUX – потому что переходит ровно половина битов каждого родителя.
3. Не допускается дублирование строк.
4. Замечание: в СНС размер популяции достаточно мал – порядка 100 особей.

Генетический алгоритм с нефиксированным размером популяции.

Genetic Algorithm with Varying Population Size – GAVaPS



Замечание.

Каждой особи сопоставляется максимальный возраст, который равен количеству поколений, в течение которых эта особь живет. После прошествия времени t – особь умрет.

Замечание.

Внедрение времени жизни исключает оператор отбора.

Размер дополнительной популяции $\text{AuxPopSize}(t)$ пропорционален размеру основной популяции $\text{PopSize}(t)$:

$$\text{AuxPopSize}(t) = \text{PopSize}(t) * p_c, \text{ где } p_c – \text{вероятность воспроизведения.}$$

1. Для воспроизведения особи выбираются из основной популяции в равной вероятности вне зависимости от их приспособленности.
2. Применяем кроссинговер, мутацию и присваиваем новым особям время жизни t в соответствии с функцией приспособленности.
3. Возраст особи $t = \text{const}$ на протяжении всей эволюции.
4. Из основной популяции удаляются все особи, чей срок жизни истек и добавляются потомки из промежуточной популяции.

Формула размера популяции после одного шага алгоритма:

$$\text{PopSize}(t + 1) = \text{PopSize}(t) + \text{AuxPopSize}(t) - D(t), \text{ где } D(t) – \text{число умерших особей в этом поколении.}$$