Вводная лекция

Федоров Глеб Владимирович
Научно-технологический университет «Сириус»
Направление «Финансовая математика и финансовые технологии»

сентябрь 2023 года

Предмет оптимизации

Постановка задачи

Пусть $f : \mathbb{R}^n \to \mathbb{R}$. Задача оптимизации:

$$\begin{cases} \min f(x), \\ x \in Q \subset \mathbb{R}^n, \\ g_i(x) \leq 0, \ i = 1, \ldots, l, \\ g_i(x) \leq 0, \ i = l+1, \ldots, m. \end{cases} \tag{1}$$

Типы задачи оптимизации

- линейное программирование (LP)
- квадратичное программирование (QP)
- математическое программирование (нелинейное программирование - NLP)
  Задача оптимизации
  - негладкая / гладкая
  - локальная / глобальная
  - невыпуклая / выпуклая / сильно выпуклая
  - условная (равенства, неравенства, на множестве) / безусловная
- целочисленное программирование (IP)

Решение задачи оптимизации

Точка минимума x^*, значение функции $f^* = f(x^*)$.

Решение $\overline{x}$ или $\overline{f} = f(\overline{x})$ задачи оптимизации:

- $\|\overline{x} - x^*\| < \varepsilon$
- $\|\overline{f} - f^*\| < \varepsilon$
- $\nabla f(\overline{x}) < \varepsilon$

Обзор градиентных методов

Постановка задачи

Пусть $f : \mathbb{R}^n \to \mathbb{R}$. Задача оптимизации без ограничений:

$$\min_{x \in \mathbb{R}^n} f(x). \tag{2}$$

Все рассматриваемые далее методы (алгоритмы) *итеративные*.

Необходиые условия

Рассмотри задачу минимизации для одномерной функции

$$f(x) \to \min_{x \in \mathbb{R}}.$$

17 век, Пьер Ферма: необходимое условие $f'(x^*) = 0$.

Это условие основано на линейном приближении

$$f(x) \approx f(x^*) + f'(x^*)(x - x^*).$$

Необходиые условия

В многомерном случае аналогично из линейного приближения

$$f(x) \approx f(x^*) + \nabla f(x^*)^T (x - x^*)$$

получается необходимое условие

$$\nabla f(x^*) = 0.$$

Указанные условия являются необходимыми, но не достаточными, самый простой пример — $x = 0$ для $f(x) = x^2$ и $f(x) = x^3$.

Упражнение: рассмотреть $\min f(x)$ для $f(x) = x^4 - x^2$ и $x_0 = 0$.

Квадратичные функции

$$f(x) = \frac{1}{2}x^T Ax - b^T x + c$$

Упражнение: показать, что без ограничения общности матрицу A можно считать симметричной.

$$\nabla f(x) = Ax - b$$

Получается, что для нахождения минимума нужно решить линейную систему.

Упражнение: показать, что минимум квадратичной функции $f(x)$ существует тогда и только тогда, когда матрица A положительно определена.

Направление убывания функции

Попробуем пока решить задачу: дана точка x, найти точку $\bar{x}$ такую, что $f(\bar{x}) < f(x)$ (*релаксация*).

$$f(\bar{x}) \approx f(x) + \nabla f(x)^T(\bar{x} - x).$$

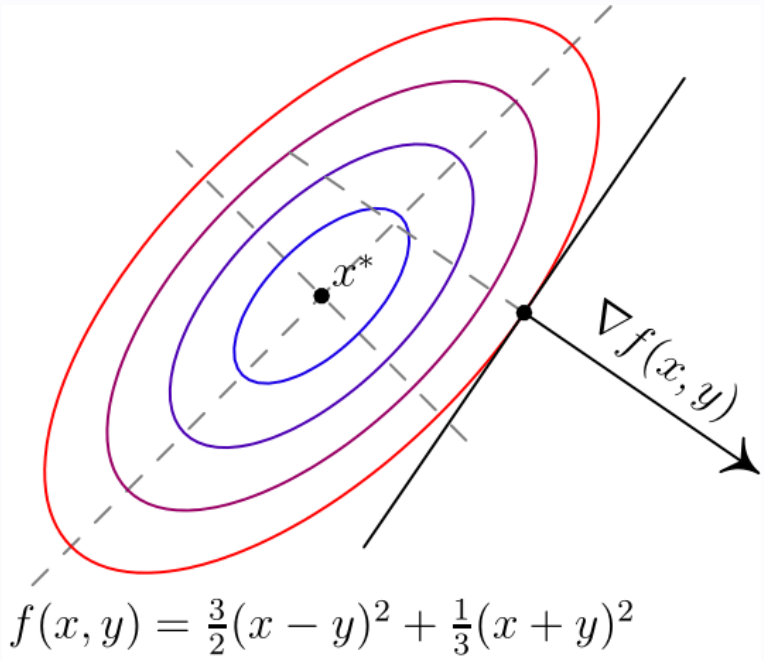
Если взять $\bar{x} = x - \alpha \nabla f(x)$, $\alpha > 0$ то мы получим

$$f(\bar{x}) \approx f(x) - \alpha \|\nabla f(x)\|^2 < f(x).$$

Эти соображения верны только для малых α.

Если $\nabla f(x) \neq 0$, то градиент указывает направление наибольшего локального увеличения функции.

Направление убывания функции



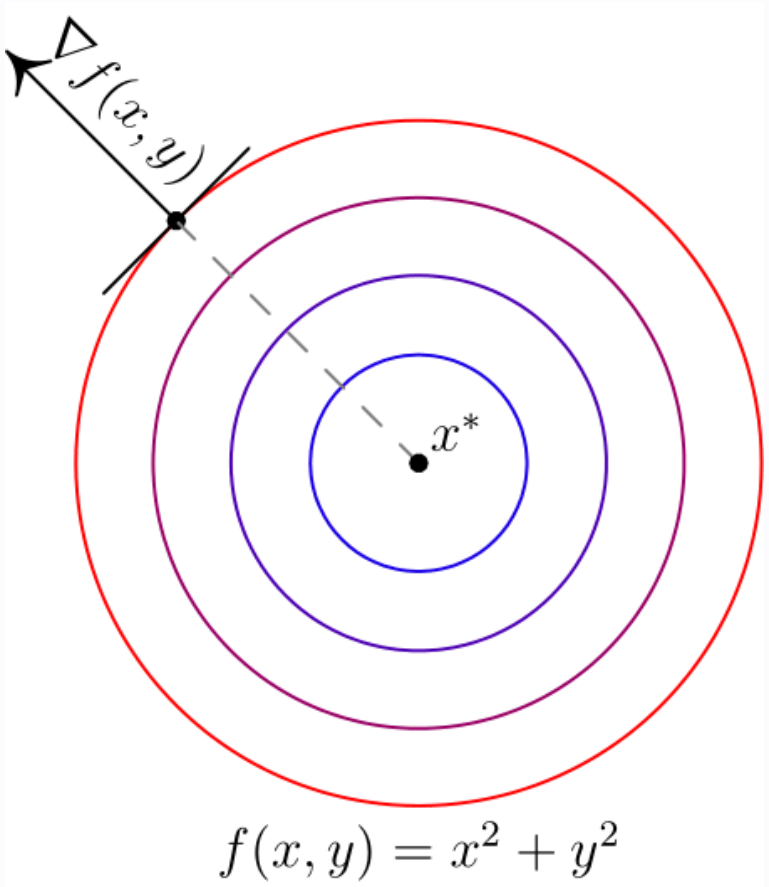


Рис. 1: Направление градиента.

Градиентный спуск

$$x_{k+1} = x_k - \alpha_k \nabla f(x_k).$$

Пример: целевая функция $f(x, y) = x^2 + 100y^2$, $x_0 = (\sqrt{2}, \sqrt{2}/10)$.

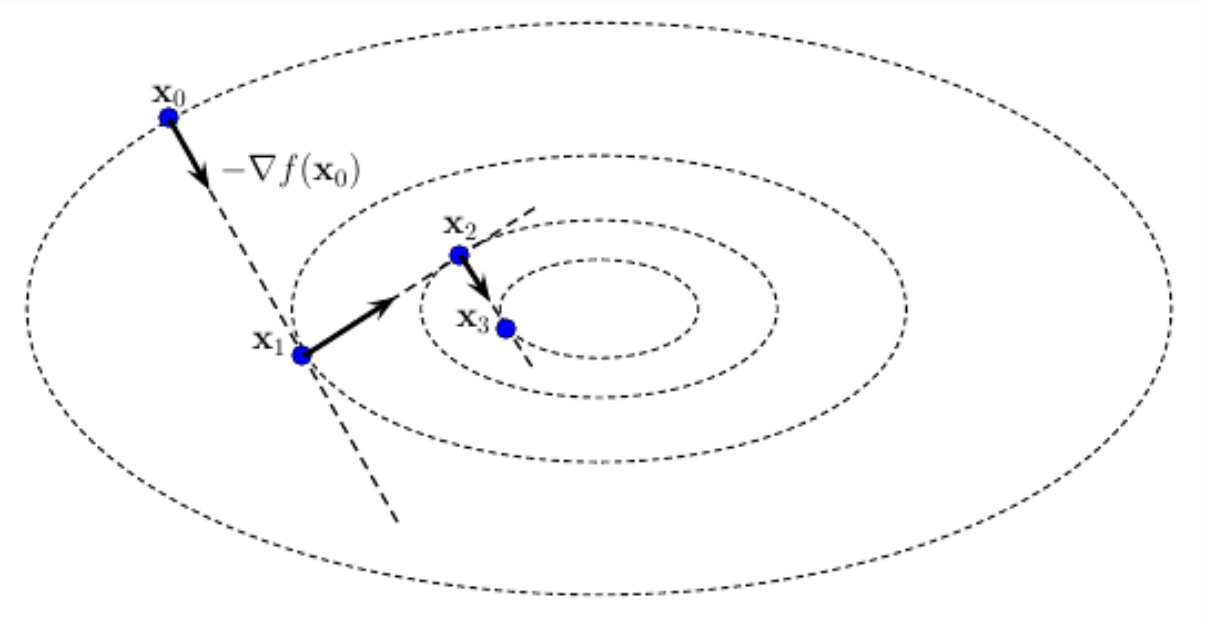


Рис. 2: Градиентный спуск.

Градиентный спуск

Липшицевость градиента:

$$\|\nabla f(x) - \nabla f(y)\| \leq L\|x - y\|$$

для всех x, y.

Выбор $\alpha_k < \frac{2}{L}$ гарантирует убывание $f(x_k)$.

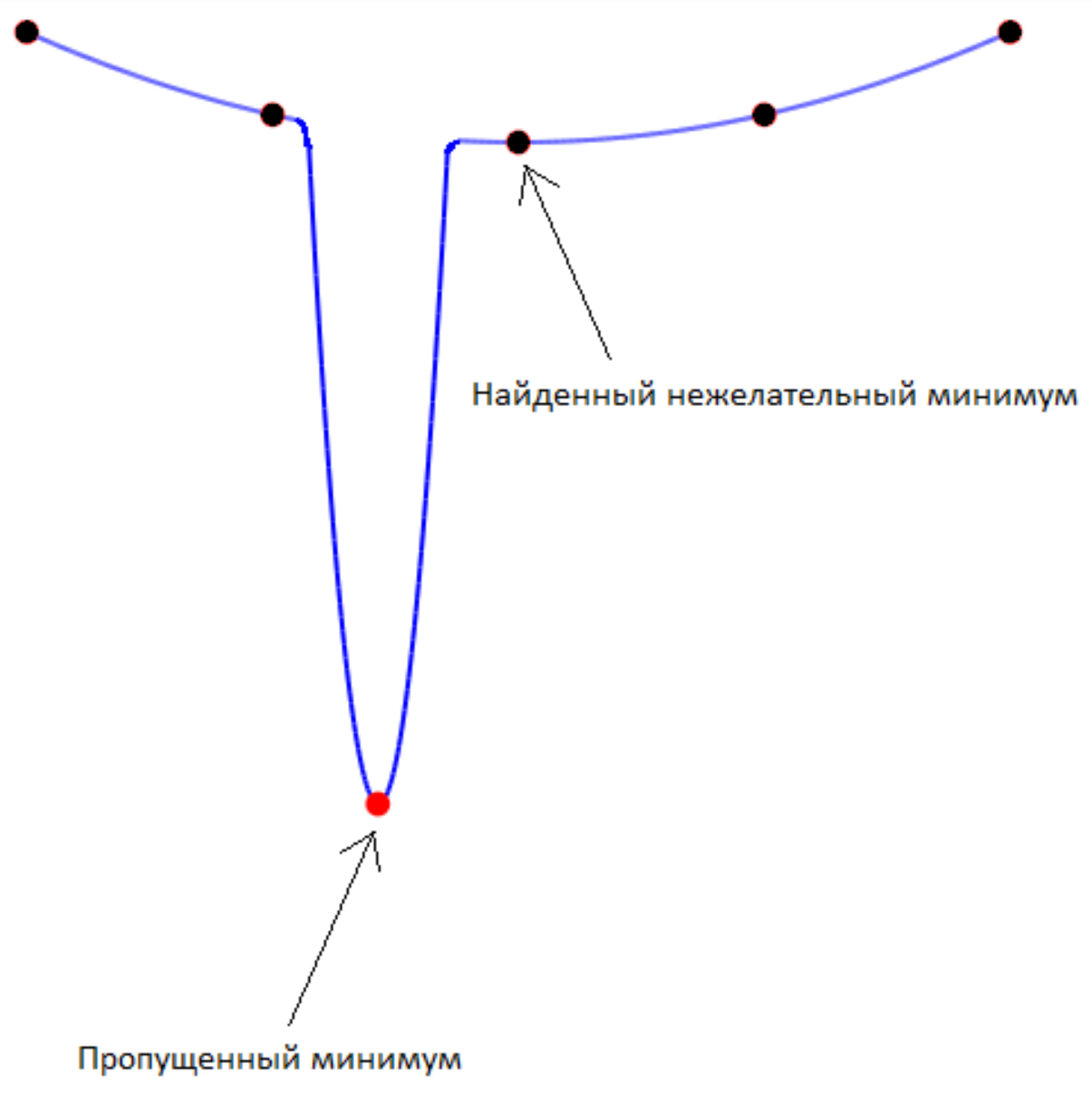
Упражнение: получить оценку скорости сходимости для квадратичной функции $f(x) = \frac{1}{2}x^T Ax - b^T x + c$.

Градиентный спуск

Проблемы

- застревание в локальных минимумах и на плато
- маленькая скорость сходимости
- выбор шага α_k
- задачи с ограничениями
- негладкие задачи

Градиентный спуск



Инерционные градиентные методы

$$x_{k+1} = x_k - \alpha_k \nabla f(x_k) + \beta_k (x_k - x_{k-1}).$$

- не усложняют обычный градиентный спуск в вычислительном плане;
- при аккуратном подборе α_k, β_k на порядок быстрее, чем обычный градиентный спуск даже с оптимально подобранным шагом.

Метод тяжелого шарика

$$v_{t-1} = x_k - x_{k-1}, \quad t = k,$$
$$v_t := \gamma v_{t-1} + \alpha_k \nabla f(x_k),$$
$$x_{k+1} := x_k - v_t,$$
$$\gamma \sim 0.9.$$

Физическая интерпретация: представьте как шарик катится по холмистой поверхности. Если в момент t под шариком был ненулевой уклон $\nabla_\theta J(\theta)$, а затем он попал на плато, он всё равно продолжит катиться по этому плато. Более того, шарик продолжит двигаться пару обновлений в ту же сторону, даже если уклон изменился на противоположный.

Метод сопряженных градиентов

Для решения **системы линейных уравнений** при градиентном спуске

$$x_{k+1} - x^* = (I - \alpha_k A)(x_k - x^*) = \ldots =$$
$$= (I - \alpha_k A)(I - \alpha_{k-1} A) \ldots (I - \alpha_1 A)(x_0 - x^*) = P_k(A)(x_0 - x^*),$$

Метод Чебышева: чтобы подобрать параметры спуска так, чтобы P_k был многочленом Чебышева (наименее отклоняющийся от нуля)

$$T_{n+1}(x) = 2xT_n(x) - T_{n-1}(x).$$

Метод сопряженных градиентов

Следствие теоремы Гамильтона-Кэли: для любой квадратной матрицы A размера $n \times n$ существует многочлен P степени не больше n, для которого $P(A) = 0$.

Идея: на каждой итерации выбирать параметры, дающие наилучший многочлен $P_k(A)$.

Метод сопряженных градиентов (NCG)

В общем случае

$$\mathbf{x}_{k+1} := \mathbf{x}_k + \alpha_k \mathbf{p}_k$$

$$\begin{aligned}
\mathbf{p}_k &= -\nabla f(\mathbf{x}_k) && \text{метод наискорейшего спуска;} \\
\nabla^2 f(\mathbf{x}_k)\mathbf{p}_k &= -\nabla f(\mathbf{x}_k) && \text{метод Ньютона;} \\
\mathbf{p}_k &= -\mathbf{H}_k \nabla f(\mathbf{x}_k) && \text{квазиньютоновские методы;} \\
\mathbf{p}_k &= -\nabla f(\mathbf{x}_k) + \beta_k \mathbf{p}_{k-1} && \text{метод сопряжённых градиентов,}
\end{aligned}$$

Метод сопряженных градиентов

$$FR: \ \beta_k = \frac{|\nabla f(\mathbf{x}_k)|^2}{|\nabla f(\mathbf{x}_{k-1})|^2},$$

$$PR: \ \beta_k = \frac{|\nabla f(\mathbf{x}_k)|^2 - \nabla f(\mathbf{x}_{k-1}) \cdot \nabla f(\mathbf{x}_k)}{|\nabla f(\mathbf{x}_{k-1})|^2},$$

$$HS: \ \beta_k = \frac{|\nabla f(\mathbf{x}_k)|^2 - \nabla f(\mathbf{x}_{k-1}) \cdot \nabla f(\mathbf{x}_k)}{(\nabla f(\mathbf{x}_k) - \nabla f(\mathbf{x}_{k-1})) \cdot \mathbf{p}_{k-1}},$$

$$DY: \ \beta_k = \frac{|\nabla f(\mathbf{x}_k)|^2}{(\nabla f(\mathbf{x}_k) - \nabla f(\mathbf{x}_{k-1})) \cdot \mathbf{p}_{k-1}},$$

$$HZ: \ \rho_k = \frac{1}{\mathbf{y}_k \cdot \mathbf{p}_k}, \quad \beta_k = \rho_{k-1}(\mathbf{y}_{k-1} - 2\rho_{k-1}|\mathbf{y}_{k-1}|^2 \mathbf{p}_{k-1}) \cdot \nabla f(\mathbf{x}_k)$$

$$\mathbf{y}_k = \nabla f(\mathbf{x}_{k+1}) - \nabla f(\mathbf{x}_k).$$

Метод Нестерова

Идея: заглядывание вперед по вектору обновления

$$x_{k+1} = x_k - \alpha_k \nabla f(x_k + \beta_k(x_k - x_{k-1}))$$

Такое изменение позволяет быстрее «катиться», если в стороне, куда мы направляемся, производная увеличивается, и медленнее, если наоборот.

Продвинутый вариант с инерционным членом:

$$x_{k+1} = x_k - \alpha_k \nabla f(x_k + \beta_k(x_k - x_{k-1})) + \beta_k(x_k - x_{k-1})$$

Стохастический градиентный спуск

Идея: использовать вместо градиента $\nabla f(x)$ функцию $g(x, \theta)$ такую, что

$$E_\theta g(x, \theta) = \nabla f(x),$$

где E_θ — математическое ожидание по случайной величине θ.

Стохастический градиентный спуск имеет вид

$$x_{k+1} = x_k - \alpha_k g(x_k, \theta_k).$$

В среднем мы идем против градиента.

Стохастический градиентный спуск

Пример 1.

$$f(x) = \frac{1}{2m}\sum_{j=1}^{m}\|x - y_j\|^2,$$

$$\nabla f(x) = \frac{1}{m}\sum_{j=1}^{m}(x - y_j),$$

$$g(x, i) = x - y_i.$$

Если i принимает значения $1, \ldots, m$ равновероятно, то как раз в среднем g — это градиент f.

Стохастический градиентный спуск

Пример 2. Предположим, что мы хотим приблизить прямую $\hat{y} = w_1 + w_2 x$ тренировочным набором с множеством наблюдений $(x_1, x_2, \ldots, x_n)$ и соответствующих ответов $(\hat{y}_1, \hat{y}_2, \ldots, \hat{y}_n)$ с помощью метода наименьших квадратов. Целевой функцией для минимизации будет

$$Q(w) = \sum_{i=1}^{n} Q_i(w) = \sum_{i=1}^{n} (\hat{y}_i - y_i)^2 = \sum_{i=1}^{n} (w_1 + w_2 x_i - y_i)^2 .$$

Стохастический градиентный спуск

Имеем

$$\begin{bmatrix} w_1 \\ w_2 \end{bmatrix} := \begin{bmatrix} w_1 \\ w_2 \end{bmatrix} - \eta \begin{bmatrix} \frac{\partial}{\partial w_1}(w_1 + w_2 x_i - y_i)^2 \\ \frac{\partial}{\partial w_2}(w_1 + w_2 x_i - y_i)^2 \end{bmatrix} =$$

$$= \begin{bmatrix} w_1 \\ w_2 \end{bmatrix} - \eta \begin{bmatrix} 2(w_1 + w_2 x_i - y_i) \\ 2x_i(w_1 + w_2 x_i - y_i) \end{bmatrix}.$$

Ключевое различие по сравнению со стандартным (пакетным) градиентным спуском в том, что только одна часть данных из всего множества используется на каждом шаге и эта часть выбирается на каждом шаге случайно.

Субградиентный спуск

Дифференцируемая функция f выпукла, когда

$$f(y) \geq f(x) + \nabla f(x)^T (y - x)$$

для всех x, y.

Идея: найти такой вектор g, что

$$f(y) \geq f(x) + g^T (y - x)$$

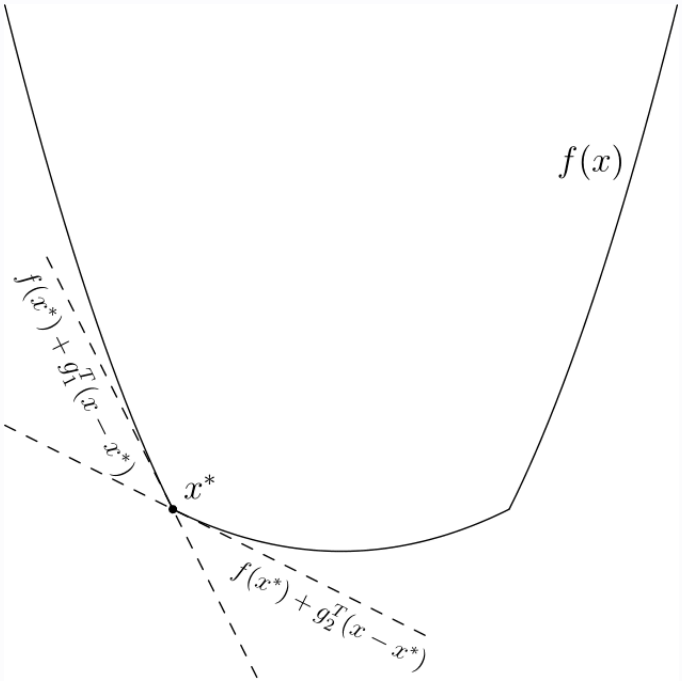
для всех y.

Для выпуклых функций вектор g существует и называется *субградиентом*.

Множество всех субградиентов в точке x называют *субдифференциалом* x и обозначают $\partial f(x)$.

Субградиентный спуск

В одномерном случае g — это число, и график f лежит выше прямой, проходящей через $(x, f(x))$ и имеющей тангенс угла наклона g



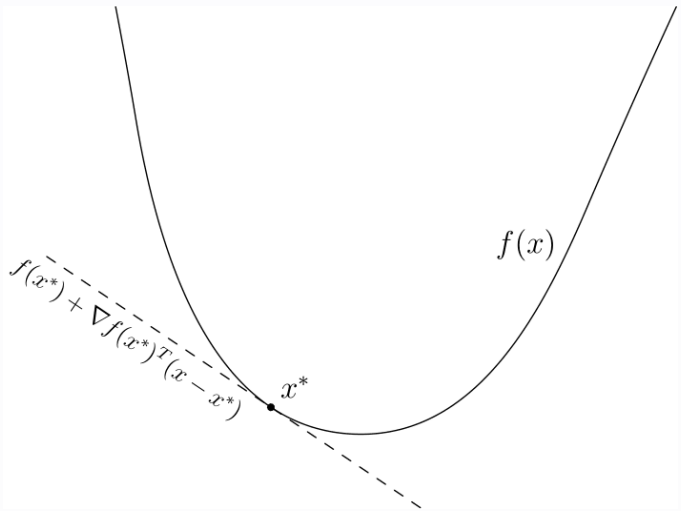


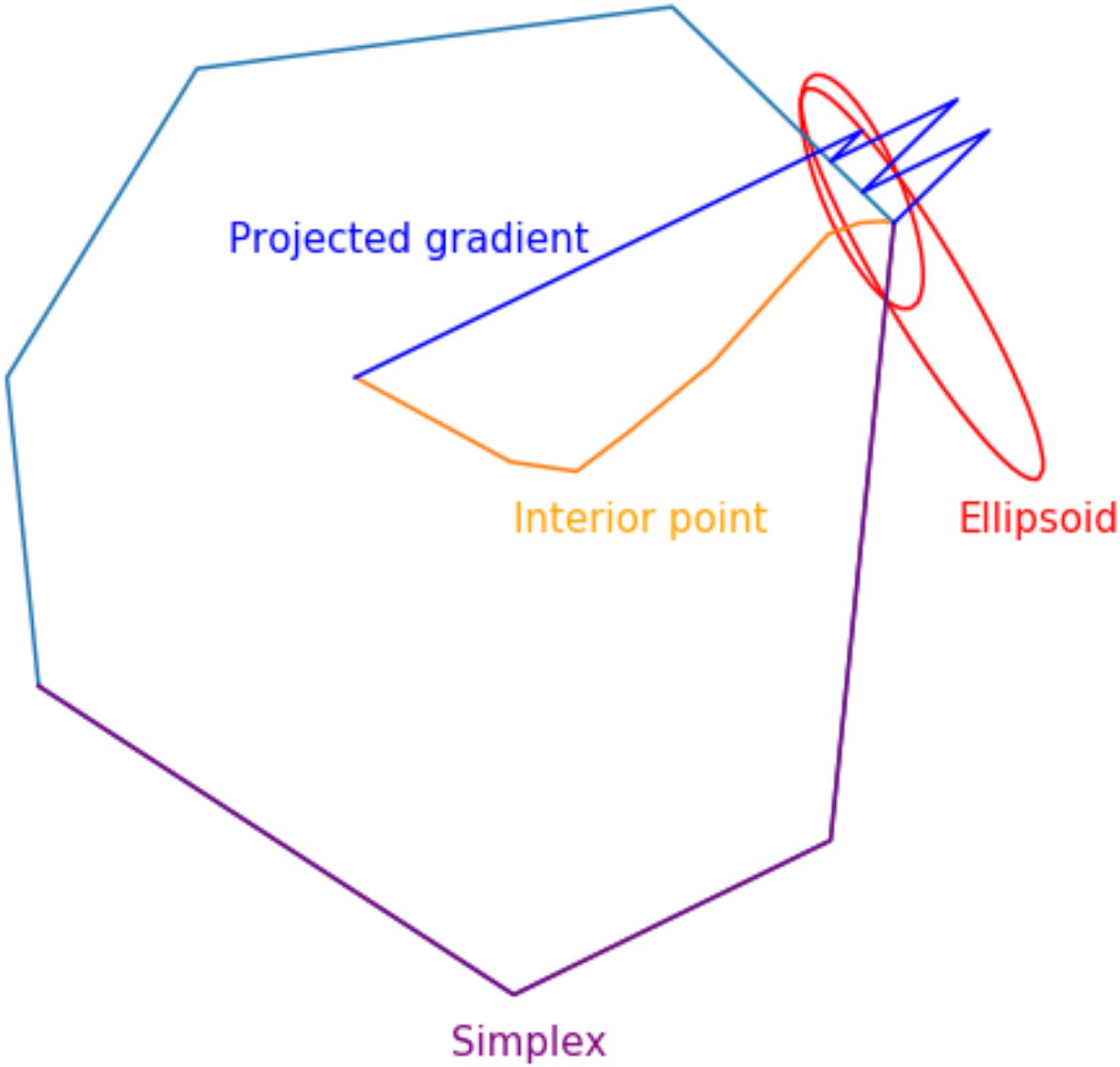
Рис. 4: Субградиентный спуск.

Субградиентный спуск

Упражнение: вычислить $\partial f(x)$ для $f(x) = |x|$.

Упражнение: показать, что для $f(x) = |x|$ при постоянном шаге субградиентный метод не сходится.

Обзор методов математической оптимизации для задач с ограничениями


Projected gradient
Interior point
Ellipsoid
Simplex

Симплекс метод

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

Симплекс метод

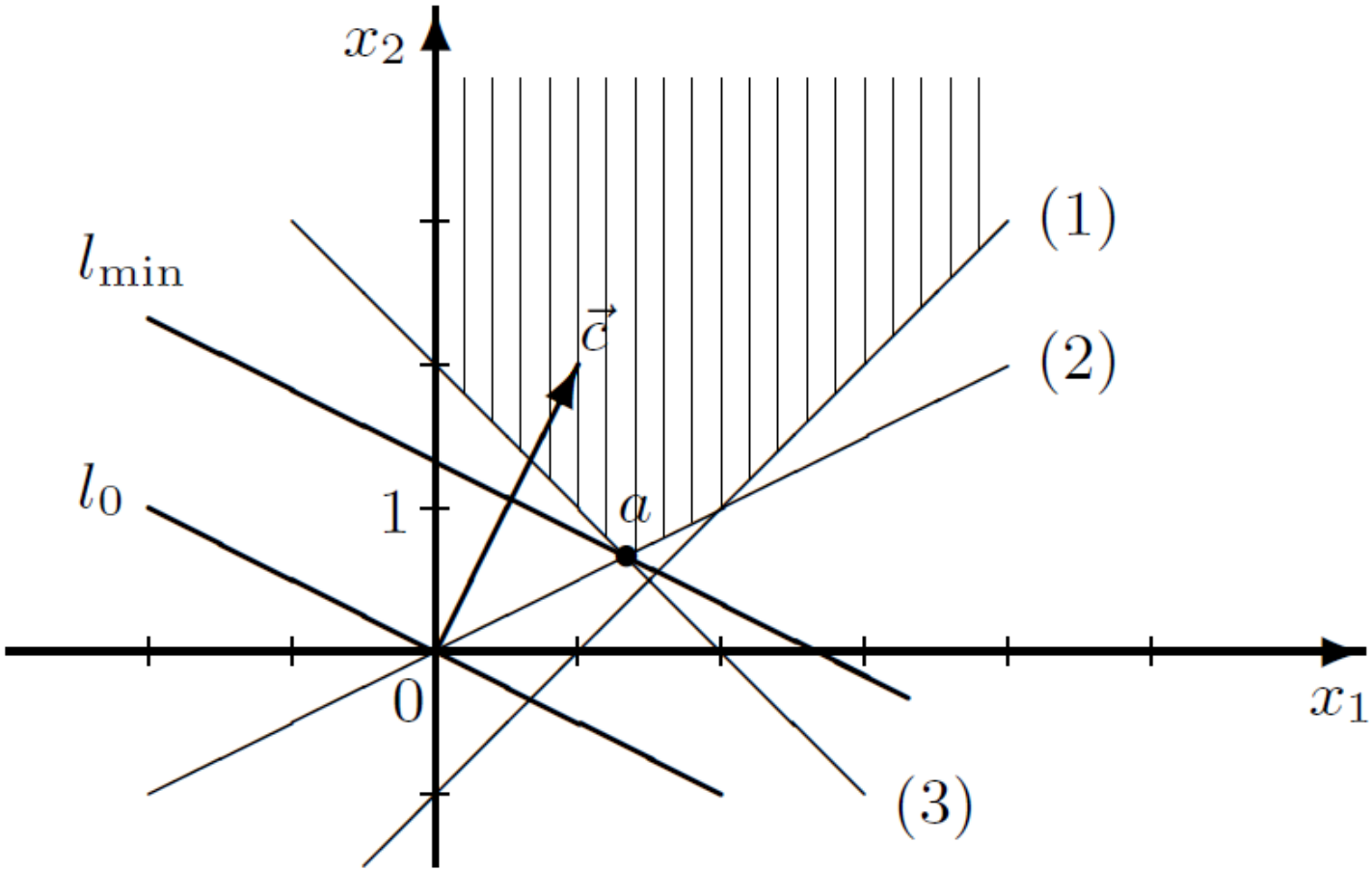


Рис. 6: Графическое решение задачи ЛП.

Симплекс метод

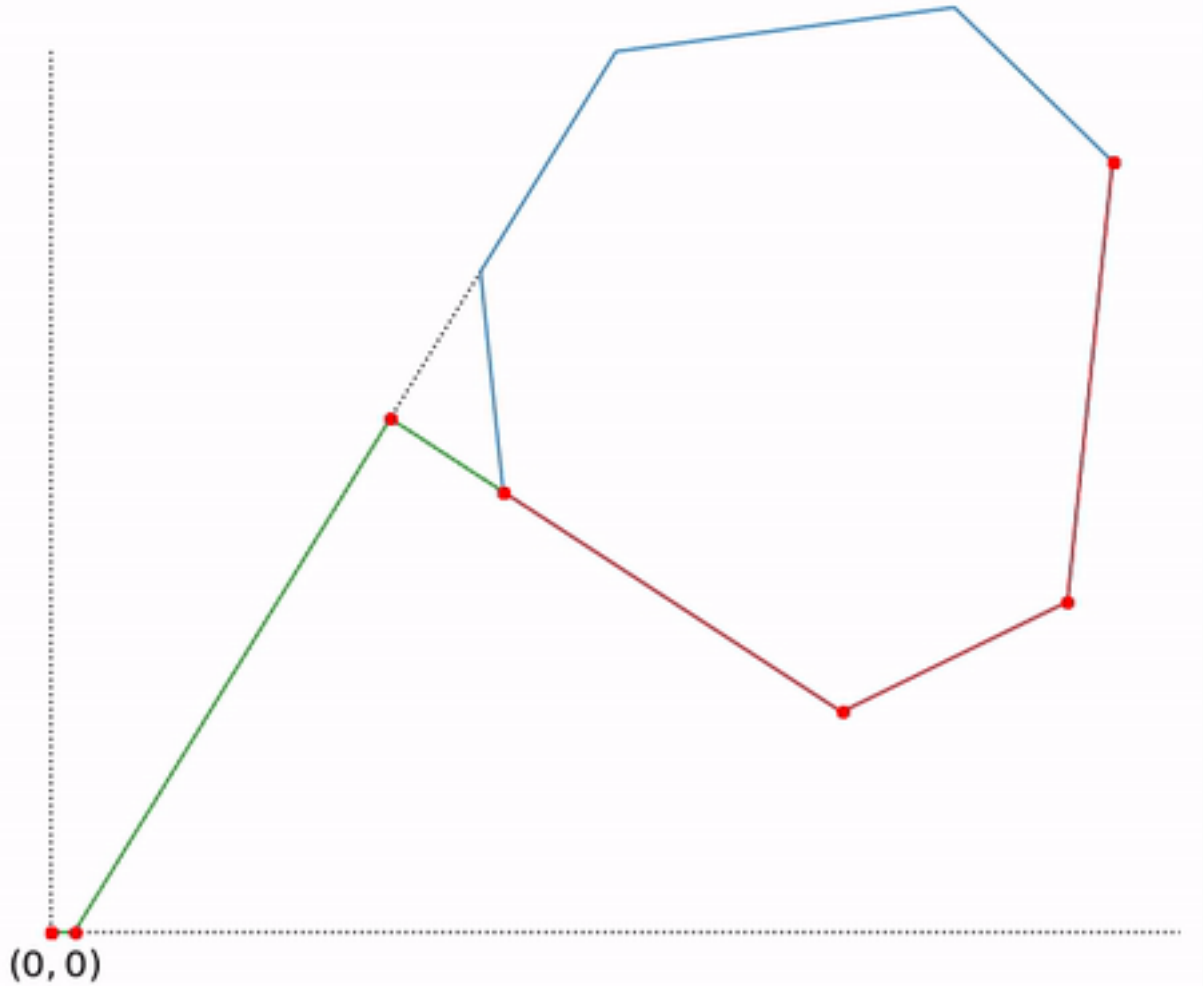


Рис. 7: Траектория двухфазового симплекс метода.

Проективный градиентный спуск

$$x_{k+1} = y_k - \alpha_k \nabla f(y_k),$$
$$y_{k+1} = P_{\mathcal{K}}(x_{k+1}),$$
$$P_{\mathcal{K}}(x) = \operatorname{argmin}_{y \in \mathcal{K}} \|x - y\|.$$

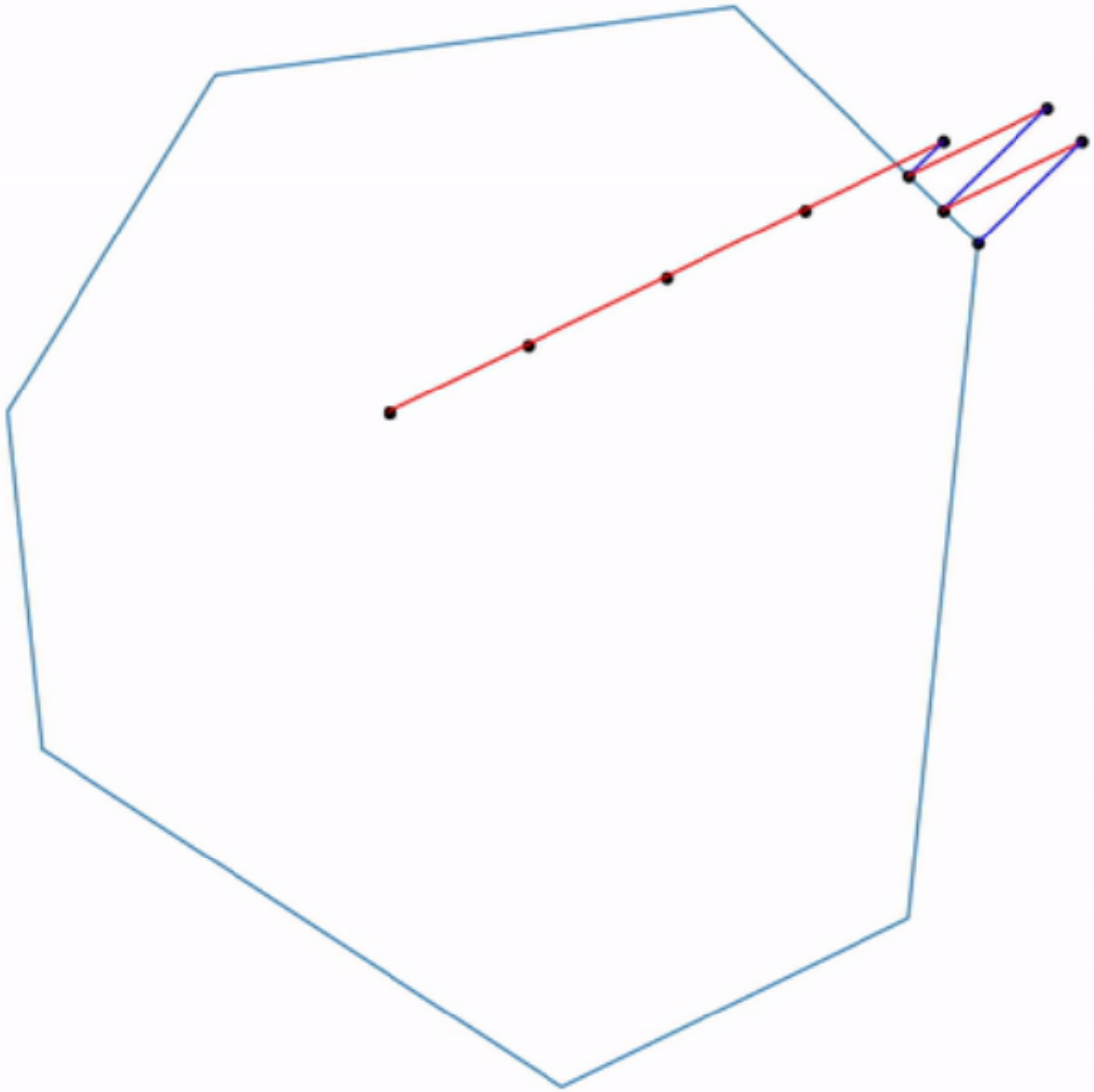
"Коробочные ограничения" для поиска псевдопроекции

$$\ell_i \leq x_i \leq r_i, \quad 1 \leq i \leq n,$$

В этом случае проекция вычисляется очень просто

$$[P_{\mathcal{K}}(x)]_i = \begin{cases} r_i, & x_i > r_i \\ \ell_i, & x_i < \ell_i \\ x_i, & x_i \in [\ell_i, r_i]. \end{cases}$$

Проективный градиентный спуск



Метод разделяющей гиперплоскости

Идея: использовать неравенство для выпуклых функций

$$f(y) \geq f(x) + \nabla f(x)^T (y - x).$$

Если зафиксировать x, то для выпуклой функции f полупространство $\nabla f(x)^T (y - x) \geq 0$ содержит только точки со значением не меньше, чем в точке x, а значит их можно отсечь.

Метод эллипсоидов

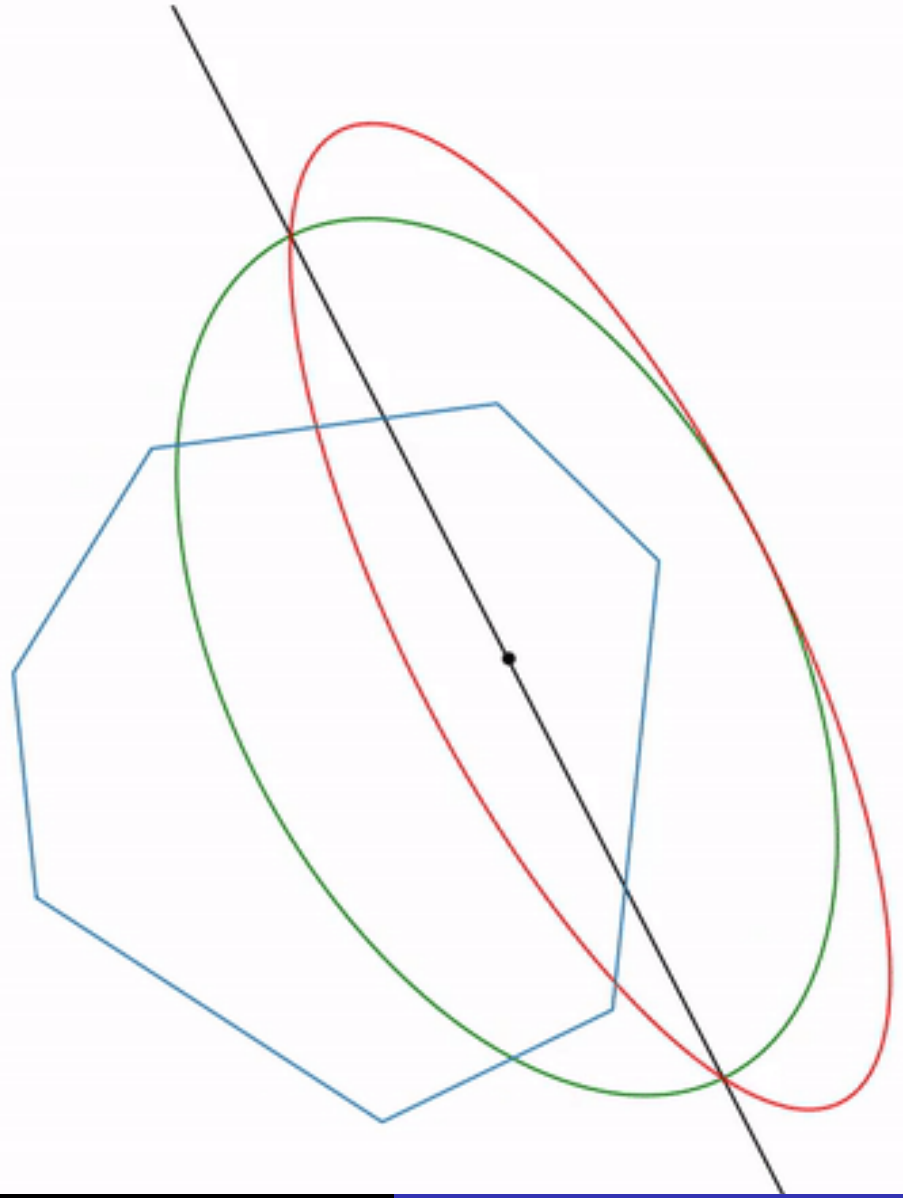
Первый полиномиальный алгоритм для задач линейного программирования.

Эллипсоид всегда можно задать положительно определенной матрицей P и вектором x (центром эллипсоида) следующим образом

$$\mathcal{E}(P, x) = \{z : \ (z - x)^T P(z - x) \leq 1\}.$$

Идея: на очередном шаге строить минимальный по объему эллипсоид, содержащий пересечение полупространства и эллипсоида предыдущего шага.

Метод эллипсоидов



Метод внутренней точки

Идея: заменить ограничения на *штраф* в виде так называемой *барьерной функции*.

Функция $F : \mathrm{Int}\ Q \to \mathbb{R}$ называется *барьерной функцией* для множества Q, если

$$F(x) \to +\infty \quad \text{при } x \to \partial Q.$$

Новая задача:

$$\min_x \varphi(x, t), \quad \varphi(x, t) = tf(x) + F(x), \quad t \to +\infty.$$

Метод внутренней точки

Если множество Q задано в виде набора неравенств $g_i(x) \leq 0$, $1 \leq i \leq m$, то стандартным выбором барьерной функции является *логарифмический барьер*

$$F(x) = -\sum_{i=1}^{m} \ln(-g_i(x)).$$

Точки минимума $x^*(t)$ функции $\varphi(x, t)$ для разных t образует кривую, которую обычно называют *центральный путь*.

$$x_{k+1} = x_k - [\nabla_x^2 \varphi(x_k, t_k)]^{-1} \nabla_x \varphi(x_k, t_k),$$
$$t_{k+1} = \alpha t_k, \quad \alpha > 1.$$

Метод внутренней точки

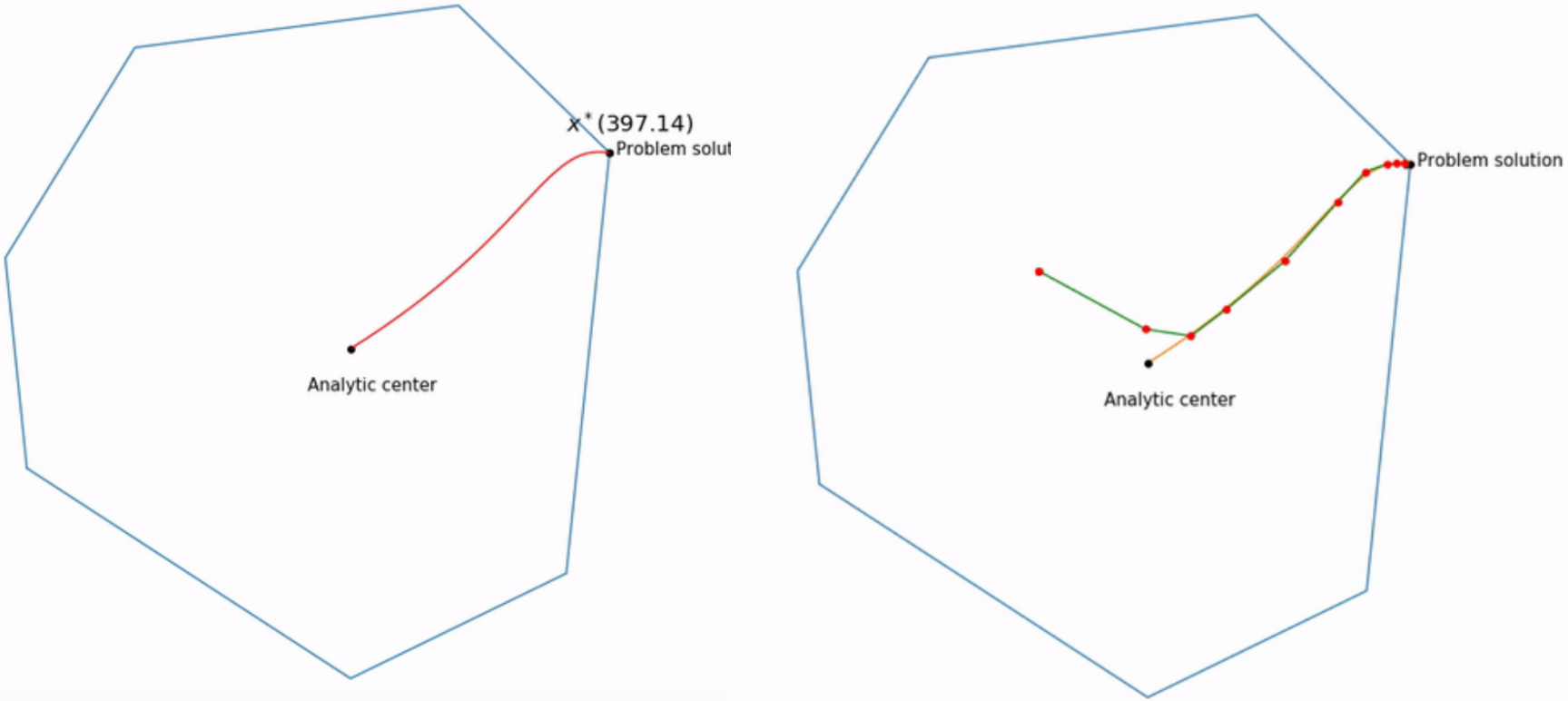
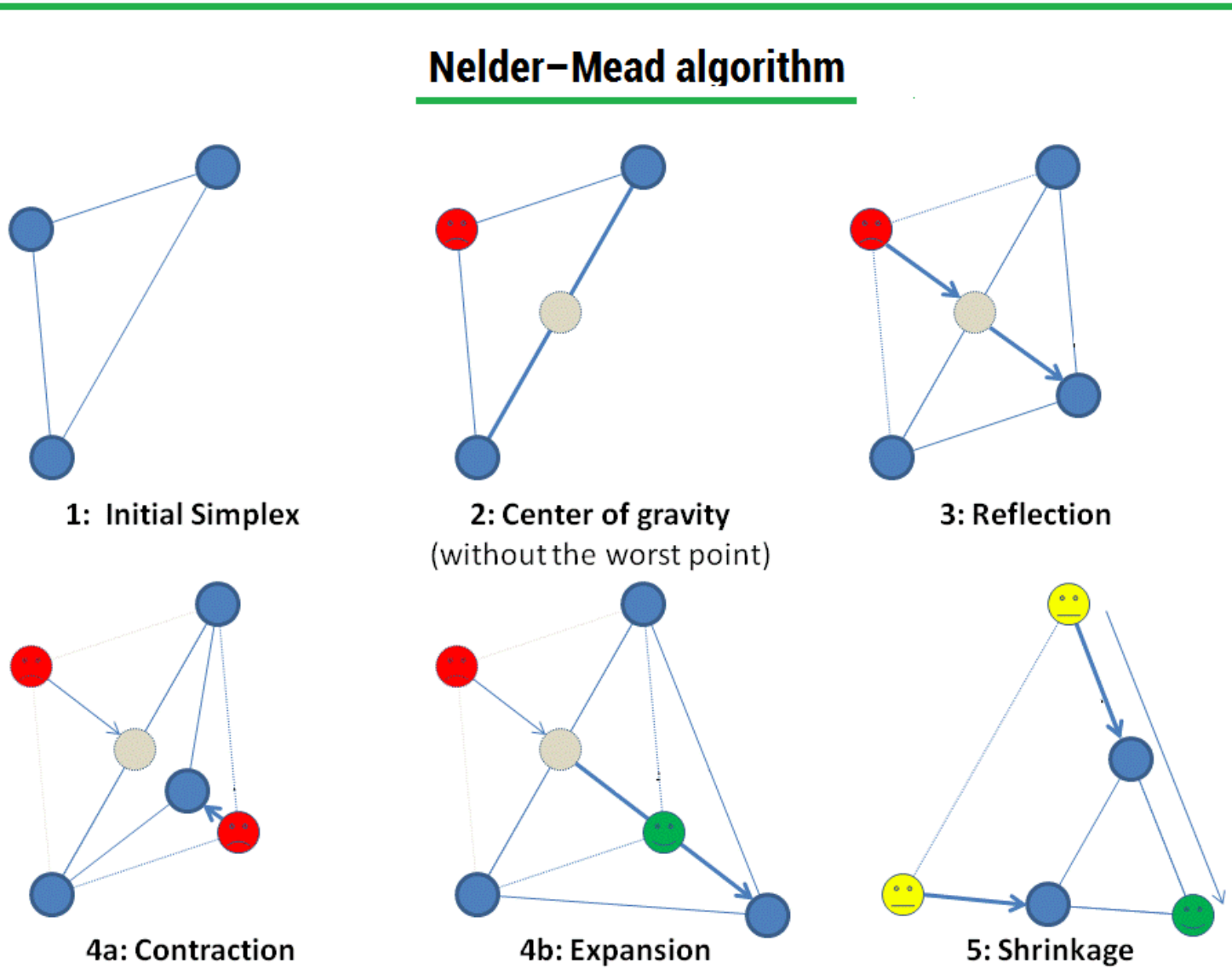


Рис. 10: Метод внутренней точки.

Эвристические методы

- Генетические методы (Genetic algorithm)
- Метод отжига (имитация отжига - Simulated Annealing)
- Алгоритм поведения роя пчёл (Bees algorithm)
- Алгоритм поведения колонии муравьёв (Ant colony optimization algorithms)
- Гармонический поиск (Harmony Search)
- Искусственные иммунные системы (Artificial Immune Systems)
- Гравитационный поиск (Gravitational Search)
- Разбросанный поиск (Scatter Search)
- Метод перекрестной энтропии (Cross-Entropy Method)
-

Метод Нелдера-Мида (Nelder-Mead)



Метод Нелдера-Мида (Nelder-Mead)

- не использует градиенты
- достаточно эффективен
- достаточно простой
- не доказана сходимость

Реализован в библиотеке scipy.optimize.

Метод Нелдера-Мида (Nelder-Mead)

Идея: сформировать симплекс и трансформировать его в направлении миниума

- Отражение (reflection) с коэффициентом $\alpha = 1$
- Растяжение (expansion) с коэффициентом $\beta = 0,5$
- Сжатие (contract) с коэффициентом $\gamma = 2$

Метод Нелдера-Мида (Nelder-Mead)

Подготовка. Вначале выбирается $n+1$ точка $x_i = \left(x_i^{(1)}, x_i^{(2)}, \ldots, x_i^{(n)}\right)$, $i = 1, \ldots, n+1$, образующие симплекс n-мерного пространства. В этих точках вычисляются значения функции:
$f_1 = f(x_1), f_2 = f(x_2), \ldots, f_{n+1} = f(x_{n+1})$.

Метод Нелдера-Мида (Nelder-Mead)

1. *Сортировка*. Из вершин симплекса выбираем три точки: x_h с наибольшим (из выбранных) значением функции f_h, x_g со следующим по величине значением f_g и x_l с наименьшим значением функции f_l. Целью дальнейших манипуляций будет уменьшение по крайней мере f_h.
2. Найдём центр тяжести всех точек, за исключением x_h: $x_c = \frac{1}{n} \sum_{i \neq h} x_i$. Вычислять $f_c = f(x_c)$ не обязательно.
3. *Отражение*. Отразим точку x_h относительно x_c с коэффициентом α, получим точку x_r и вычислим в ней функцию: $f_r = f(x_r)$. Координаты новой точки вычисляются по формуле: $x_r = (1 + \alpha)x_c - \alpha x_h$.

Метод Нелдера-Мида (Nelder-Mead)

4. Ищем место f_r в ряду f_h, f_g, f_l.
   1. Если $f_r < f_l$, то направление выбрано удачное. Производим «растяжение». Новая точка $x_e = (1-\gamma)x_c + \gamma x_r$ и значение функции $f_e = f(x_e)$.
      1. Если $f_e < f_r$, то можно расширить симплекс: присваиваем x_h значение x_e и идем на шаг 6.
      2. Если $f_r < f_e$, то переместились слишком далеко: присваиваем x_h значение x_r и идем на шаг 6.
   2. Если $f_l < f_r < f_g$, то выбор точки неплохой. Присваиваем x_h значение x_r и переходим на шаг 6.
   3. Если $f_g < f_r < f_h$, то меняем местами значения x_r и x_h. Также нужно поменять местами значения f_r и f_h. После этого идём на шаг 6.
   4. Если $f_h < f_r$, то просто идём на следующий шаг 5. В результате (возможно, после переобозначения) $f_l < f_g < f_h < f_r$.

Метод Нелдера-Мида (Nelder-Mead)

5. *Сжатие*. Строим точку $x_s = \beta x_h + (1 - \beta)x_c$ и вычисляем в ней значение $f_s = f(x_s)$.
   1. Если $f_s < f_h$, то присваиваем точке x_h значение x_s и идём на шаг 6.
   2. Если $f_s > f_h$, то первоначальные точки оказались самыми удачными. Делаем *глобальное сжатие* симплекса — гомотетию к точке с наименьшим значением x_l: $x_i \leftarrow x_l + (x_i - x_l)/2$, $i \neq l$.
6. Последний шаг — проверка сходимости. Может выполняться по-разному, например, оценкой дисперсии набора точек. Суть проверки заключается в том, чтобы проверить взаимную близость полученных вершин симплекса, что предполагает и близость их к искомому минимуму. Если требуемая точность ещё не достигнута, можно продолжить итерации с шага 1.

Литература

1. Нестеров Ю. Е. *Методы выпуклой оптимизации*
2. Поляк Б. Т. *Введение в оптимизацию*
3. Глебов Н.И., Кочетов Ю.А., Плясунов А.В. *Методы оптимизации*
4. Гасников А. В. *Универсальный градиентный спуск*
5. Boyd. S, Vandenberghe L. *Convex Optimization*
6. Bertsekas D. P. *Convex Optimization Theory*
7. Jorge Nocedal , Stephen J. Wright *Numerical Optimization*

Литература

1. *Обзор методов численной оптимизации. Безусловная оптимизация: метод линий*
   https://habr.com/ru/articles/561128/
2. *Обзор градиентных методов в задачах математической оптимизации*
   https://habr.com/ru/articles/413853/
3. *Обзор основных методов математической оптимизации для задач с ограничениями*
   https://habr.com/ru/articles/428794/
4. *Методы оптимизации нейронных сетей*
   https://habr.com/ru/articles/318970/
5. *Метод оптимизации Нелдера — Мида. Пример реализации на Python*
   https://habr.com/ru/articles/332092/
6. *Введение в оптимизацию. Имитация отжига*
   https://habr.com/ru/articles/209610/