Лекция 2
Алгоритмы безусловной нелинейной оптимизации. Прямые методы

Курс “Анализ и разработка алгоритмов”

УНИВЕРСИТЕТ ИТМО

Содержание

Литература

- Сухарев А.Г., Тимохов А.В., Федоров В.В. Курс методов оптимизации;
- Покорный Ю.В. Оптимальные задачи;
- Алексеев В.М., Тихомиров В.М., Фомин С.В. Оптимальное управление;
- Гилл Ф., Мюррей У., Райт М. Практическая оптимизация;
- Штойер Р. Многокритериальная оптимизация: теория, вычисления и приложения;
- Nocedal J., Wright Stephen J. Numerical Optimization;
- Дэннис Дж., Шнабель Р. Численные методы безусловной оптимизации и решения нелинейных уравнений;
- Жиглявский А.А., Жилинскас А.Г. Методы поиска глобального экстремума;
- и многие другие.

Задача оптимизации

Методы оптимизации — методы построения оптимальных (в некотором смысле) решений для математических моделей

Разнообразные приложения, в т.ч. в машинном обучении и анализе данных

Математическую модель объекта зачастую можно представить в виде **целевой функции** $f = f(x)$, где x, вообще говоря, — многомерный вектор, или критерия оптимальности (с ограничениями или без).

Задача оптимизации

Решить задачу оптимизации $f(x) \to \max_{x \in Q}(\min_{x \in Q})$ означает найти $x^* \in Q$, где Q — область допустимых значений, для которых достигается максимум (минимум) функции f. Обозначение: $x^* = \arg\max_{x \in Q}(\min_{x \in Q}) f(x)$.

Если x^* найдено, то можно найти и $f(x^*)$

- **Локальная** (проще) и **глобальная** (сложнее) оптимизация (совпадают для некоторых классов f)
- Задачу max можно свести к задаче min, рассмотрев функцию $-f$

Виды задач. Безусловная нелинейная оптимизация

Q, область допустимых значений, может быть

- не задана (сведена к незаданной) (**задача безусловной оптимизации**); например, минимизировать $f(x) = x^2$ при $x \in [-1, 1]$ (Q задано) = безусловно минимизировать функцию

$$\tilde{f}(x) = \begin{cases} 1, & x \notin [-1, 1], \\ x^2, & x \in [-1, 1]. \end{cases}$$

- задана с помощью условий — системы S линейных или нелинейных уравнений или неравенств (**задача условной оптимизации**)

$$f \text{ или } S \text{ (не)линейны} \rightarrow \begin{array}{l} \textbf{(не)линейное программирование} \\ \textbf{(не)линейная оптимизация} \end{array}$$

Для нужд машинного обучения и анализа данных нас интересует

безусловная нелинейная оптимизация

Линейное программирование и условная нелинейная оптимизация — в рамках проектов

Типы методов поиска минимума

Инфо об $f \rightarrow$ **прямые методы**, **методы первого** или **второго порядка**

Прямые методы или методы нулевого порядка

используют только значения f, но не ее производных.
☺ оптимизация широкого класса функций; ☹ медленная сходимость

Методы первого порядка

используют значения f и f' (градиентные методы).
☺ относительно быстрая сходимость; ☹ необходимо знать аналитические выражения для функции и ее первой производной

Методы второго порядка

используют значения f, f' и f'' (метод Ньютона и его модификации).
☺ быстрая сходимость; ☹ необходимо знать аналитические выражения для функции и ее первой и второй производной

Одномерные прямые методы

Метод перебора=метод равномерного поиска

Пусть $f(x): [a, b] \to \mathbb{R}$. Решим приближенно задачу оптимизации $f(x) \to \min_{x \in [a, b]}$, найдя x^* с погрешностью $\varepsilon > 0$.

Алгоритм

Рассмотрим на отрезке $[a, b]$ следующие точки:

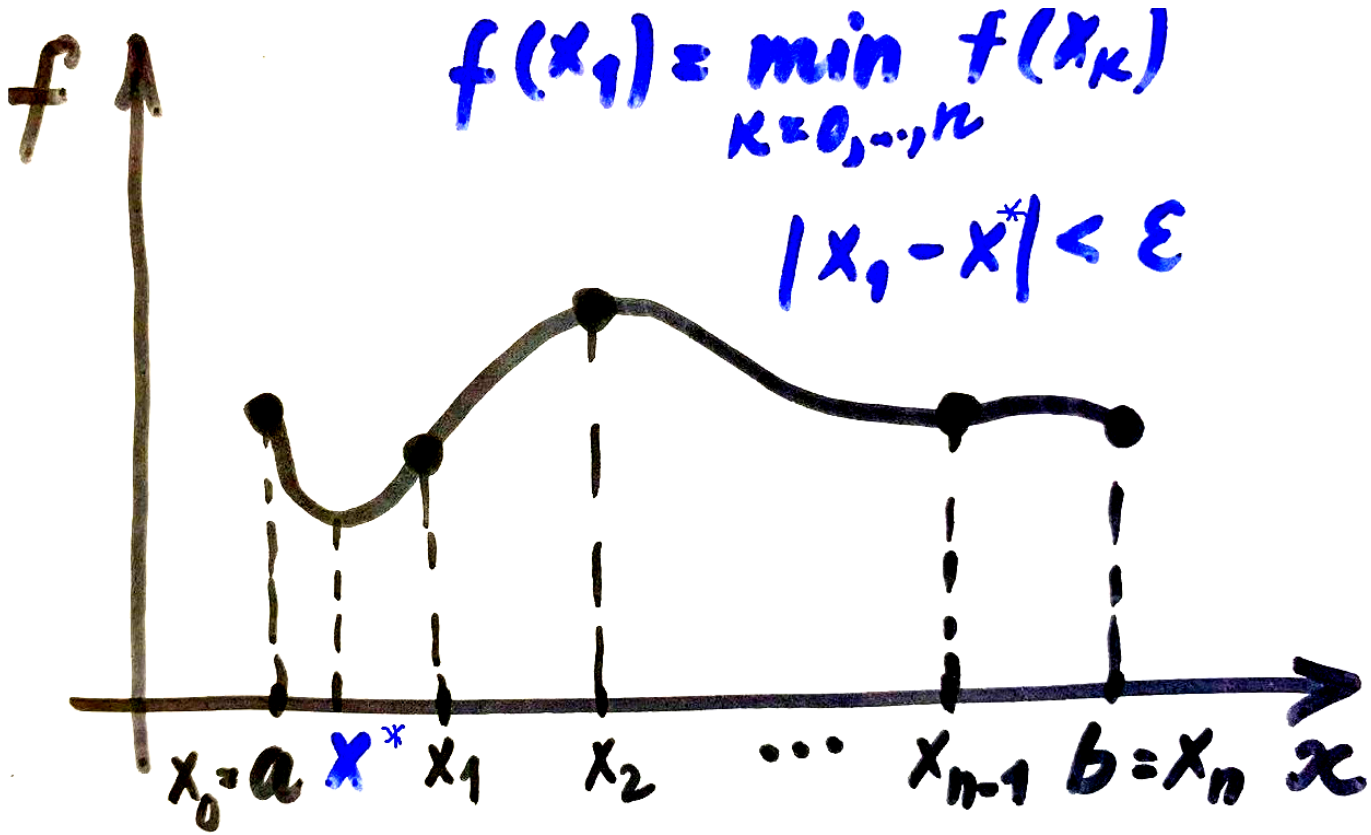
$$x_k = a + k\frac{b-a}{n}, \quad k = 0, \ldots, n,$$

где n выбрано из условия $\frac{b-a}{n} \leqslant \varepsilon$.
Вычисляем значения $f(x_k)$ и находим точку x_m, $m \in \{1, \ldots, n\}$, такую, что

$$f(x_m) = \min_{k=0,\ldots,n} f(x_k).$$

Тогда $|x_m - x| < \varepsilon$. Используем полученное x_m в качестве приближения x^*.

Этот метод можно иногда применять для поиска начальных приближений.


$f(x_1) = \min_{k=0,\dots,n} f(x_k)$
$|x_1 - x^*| < \varepsilon$
f
$x_0 = a$
x^*
x_1
x_2
$\dots$
x_{n-1}
$b = x_n$
x

Метод дихотомии

Пусть $f(x): [a_0, b_0] \to \mathbb{R}$ — выпукла. Решим приближенно задачу $f(x) \to \min_{x \in [a_0, b_0]}$, найдя x^* с погрешностью $\varepsilon > 0$.

Алгоритм

Вычисляем значения $f(x_1)$ и $f(x_2)$ в точках

$$x_1 = \frac{a_0 + b_0 - \delta}{2}, \qquad x_2 = \frac{a_0 + b_0 + \delta}{2}, \qquad 0 < \delta < \varepsilon.$$

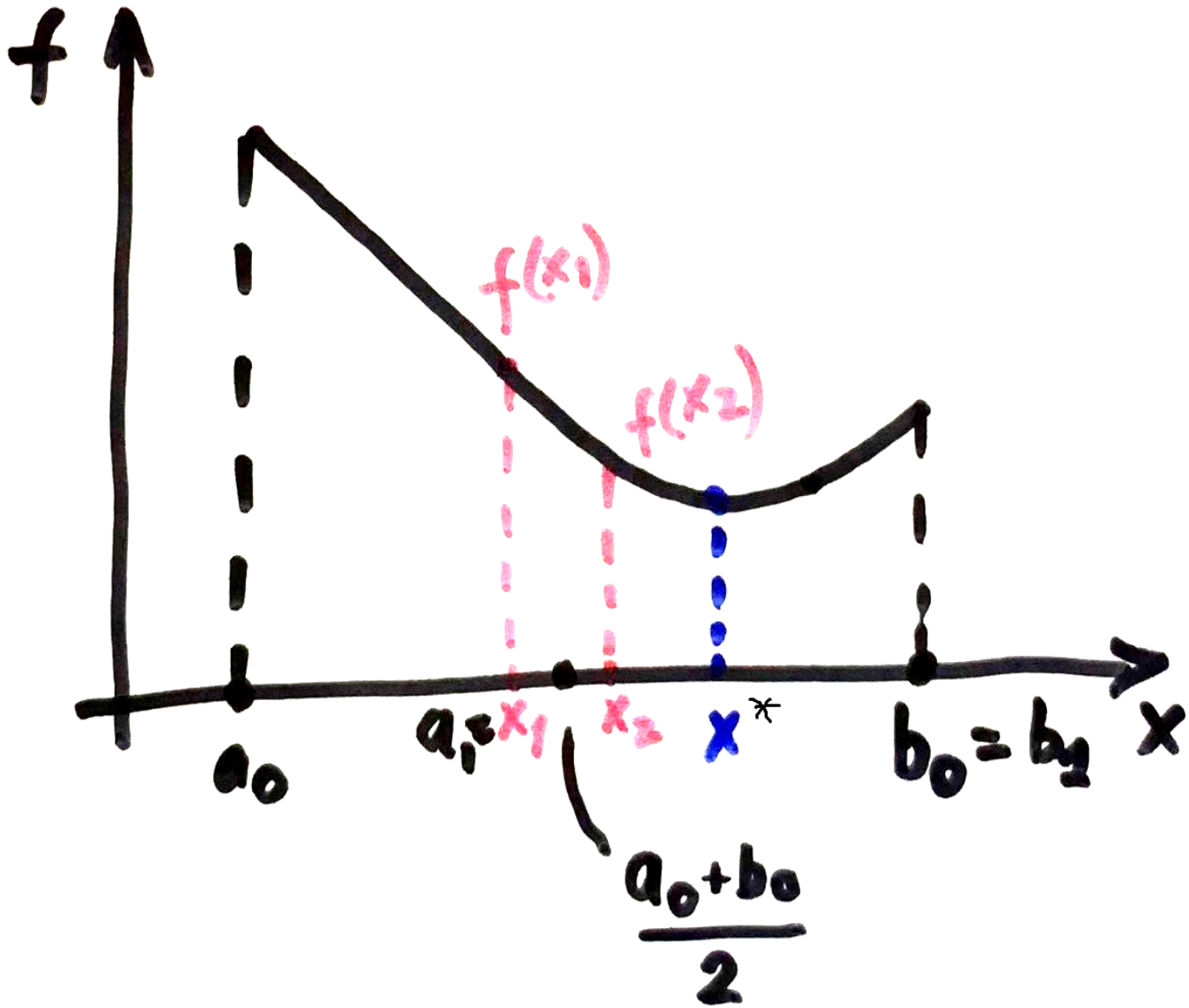
Далее сокращаем интервал неопределенности и получаем интервал $[a_1, b_1]$:

- если $f(x_1) \leqslant f(x_2)$, то $a_1 = a_0$ и $b_1 = x_2$;
- в противном случае $a_1 = x_1$ и $b_1 = b_0$.

Далее по аналогичным формулам на интервале $[a_1, b_1]$ вычисляем следующую пару точек x_1 и x_2. С помощью найденных точек определяем новый интервал неопределенности $[a_2, b_2]$, и так далее.
Поиск заканчивается, если на текущей итерации k имеем

$$|a_k - b_k| < \varepsilon \qquad (x^* \in [a_k, b_k]).$$

При $\delta = 0$ вырождается в метод бисекции.


f
f(x1)
f(x2)
a0
a1=x1
x2
x*
b0=b1
x
a0+b0
2

Метод золотого сечения

Позволяет найти точку минимума на $[a_0, b_0]$ с меньшими вычислительными затратами. По сути, особый выбор δ в методе дихотомии.

Алгоритм

Вычисляем значения $f(x_1)$ и $f(x_2)$ в точках

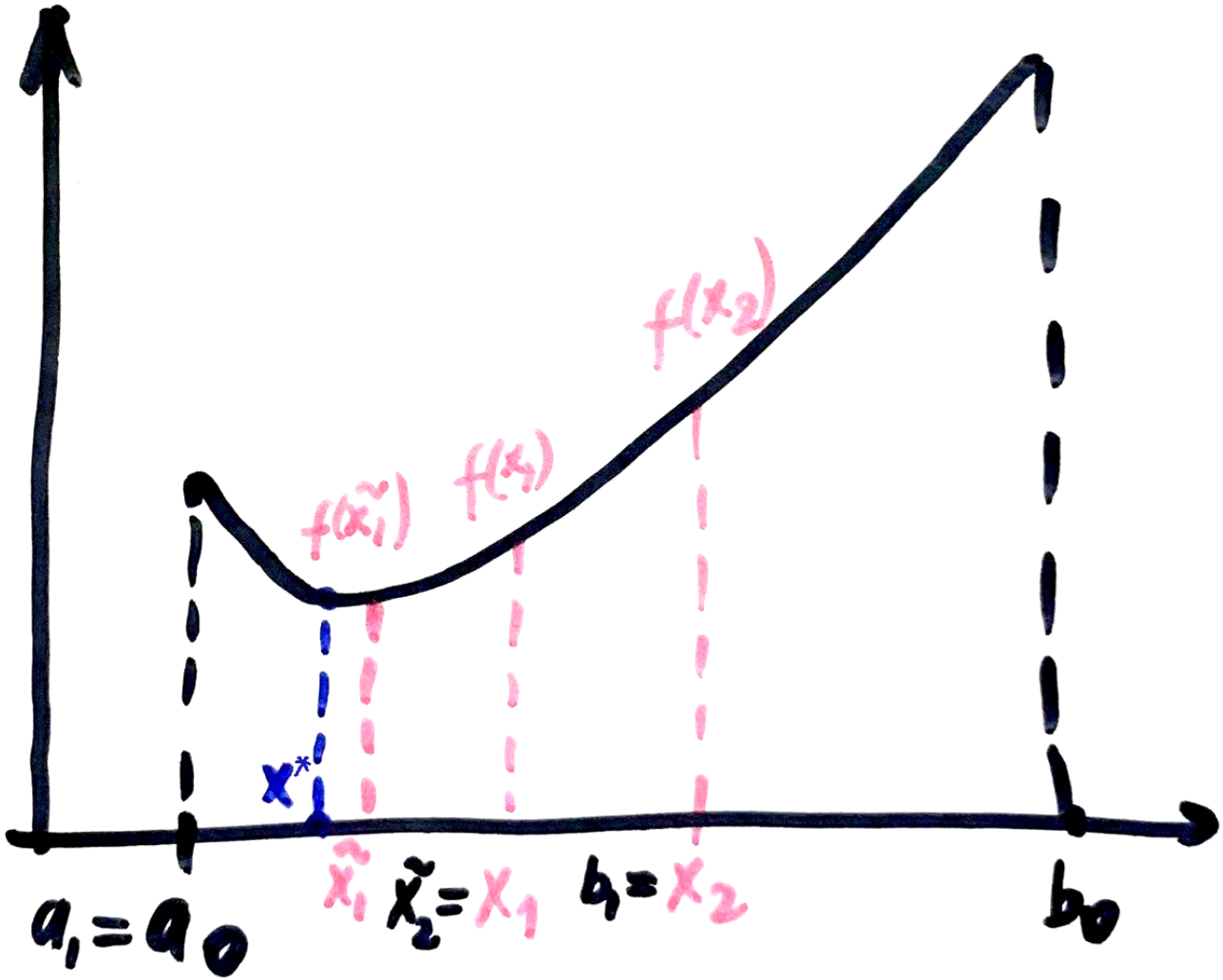
$$x_1 = a_0 + \frac{3-\sqrt{5}}{2}(b_0 - a_0), \quad x_2 = b_0 + \frac{\sqrt{5}-3}{2}(b_0 - a_0), \quad \text{причем } \frac{x_1+x_2}{2} = \frac{a_0+b_0}{2}.$$

(На первой итерации находим 2 точки и дважды вычисляем значения f.)
Далее сокращаем интервал неопределенности и получаем интервал $[a_1, b_1]$:

- если $f(x_1) \leqslant f(x_2)$, то $a_1 = a_0$, $b_1 = x_2$, $x_2 = x_1$;
- в противном случае $a_1 = x_1$ и $b_1 = b_0$, $x_1 = x_2$.

На последующих итерациях производим расчет только той точки и значение функции в ней, которые необходимо обновить: в первом случае вычисляем новое значение x_1 и $f(x_1)$; во втором — вычисляем x_2 и $f(x_2)$.
Поиск заканчивается, если на текущей итерации k имеем

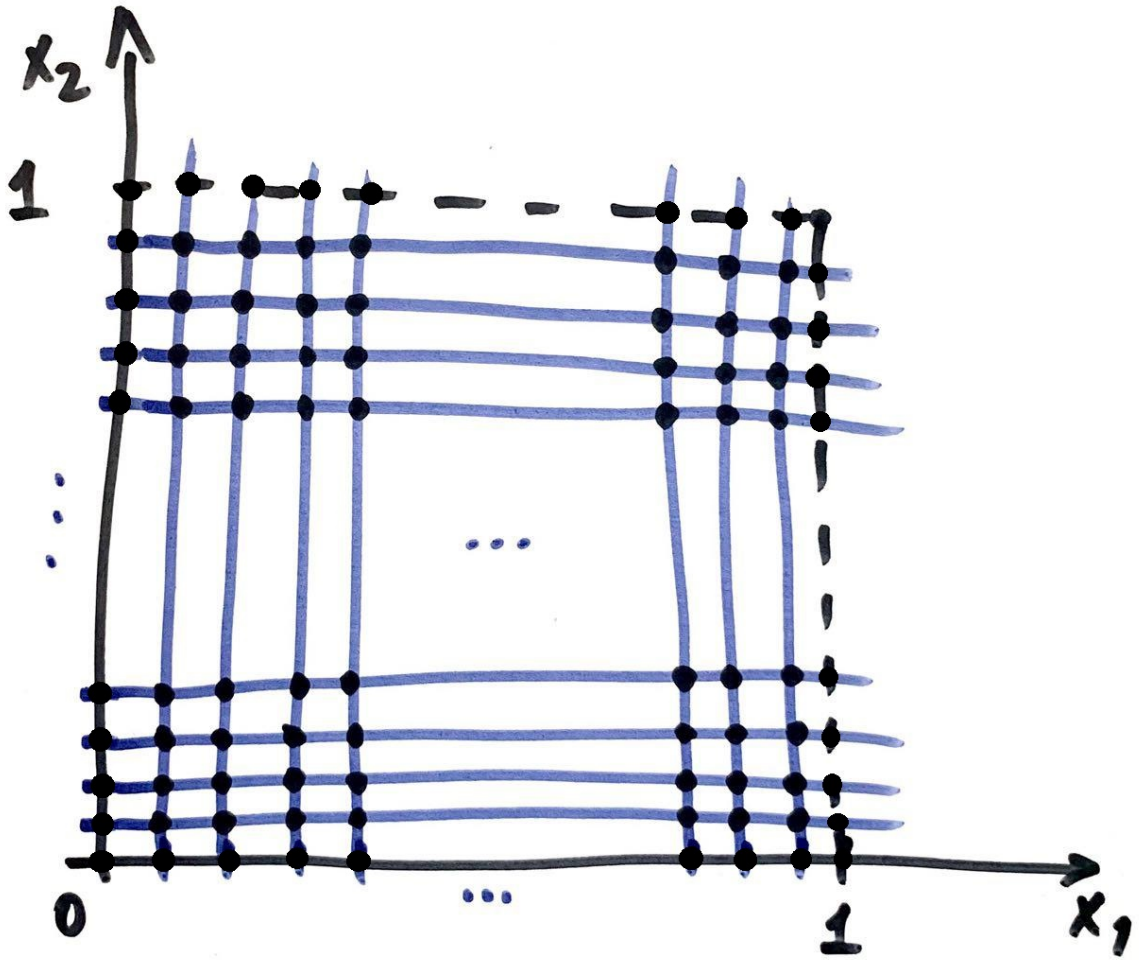
$$|a_k - b_k| < \varepsilon \qquad (x^* \in [a_k, b_k]).$$


f(x2)
f(x̃1)
f(x1)
x*
a1 = a0
x̃1
x̃2 = x1
a = x2
b0

Многомерные прямые методы
(на примере функций двух переменных)

Метод перебора

Пусть $f(x): \ D \to \mathbb{R}$, где $x = (x_1, x_2)$, $D = \{[0,1] \times [0,1]\}$. Решим приближенно задачу оптимизации $f(x) \to \min_{(x,y) \in D}$ с погрешностью $\varepsilon > 0$.



Метод Гаусса

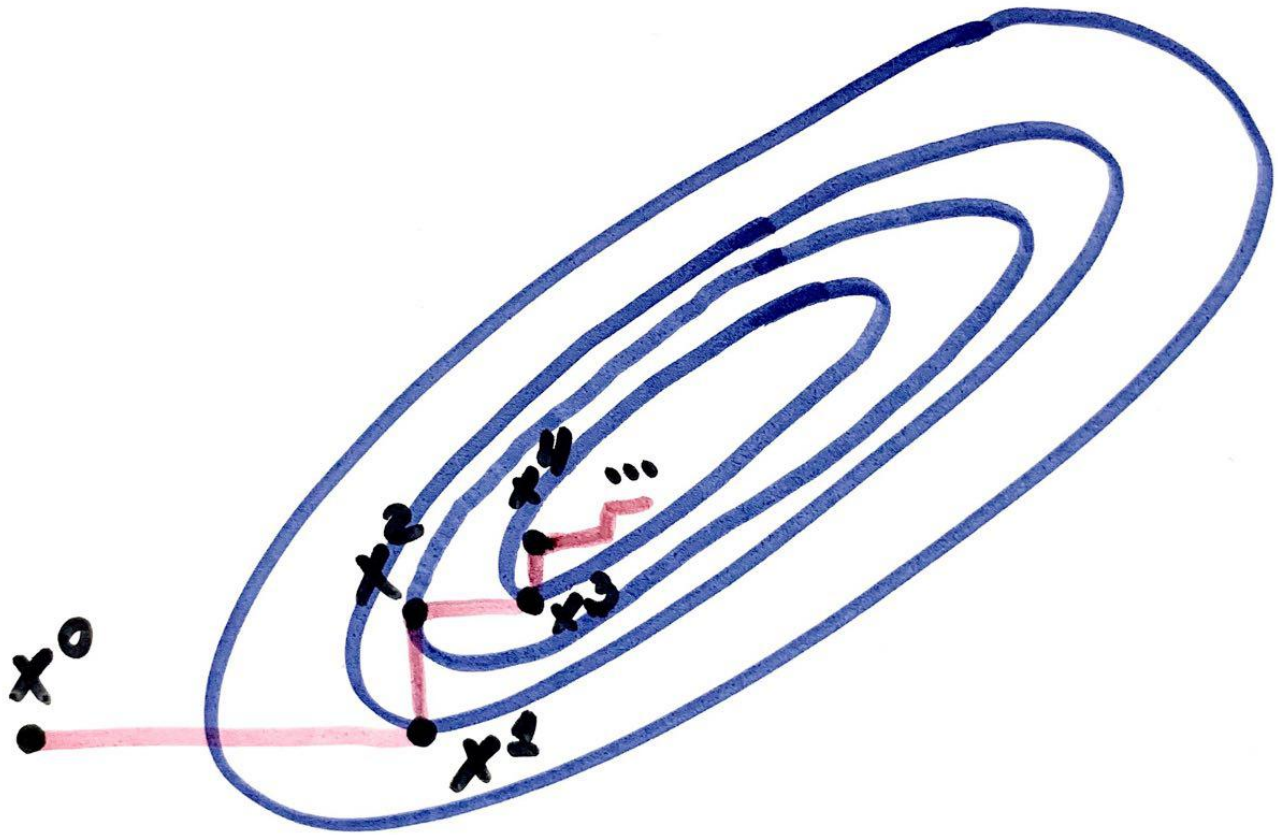
На каждой итерации минимизация осуществляется только по одной компоненте вектора переменных x. Мы рассматриваем $x = (x_1, x_2)$.

Алгоритм

Пусть нам дано начальное приближение $x^0 = (x_1^0, x_2^0)$. На первой итерации находим значение минимума функции при изменяющейся первой координате и фиксированных остальных, т.е. $x_1^1 = \arg\min_{x_1} f(x_1, x_2^0)$. Получаем новую точку $x^1 = (x_1^1, x_2^0)$.
Далее из точки x^1 ищем минимум функции, изменяя только вторую координату. В результате получаем значение $x_2^1 = \arg\min_{x_2} f(x_1^1, x_2)$ и новую точку $x^2 = (x_1^1, x_2^2)$.
Процесс поиска возобновляется по первой переменной. В качестве условий прекращения поиска можно использовать следующие критерии:

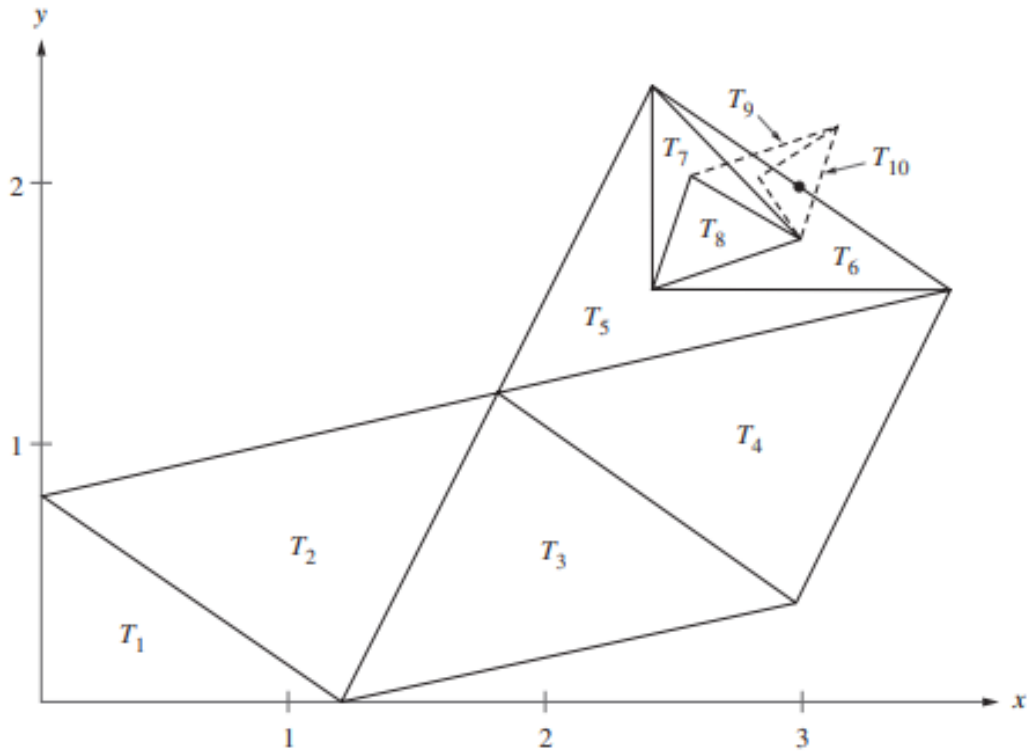
$$1) \quad |x_i^{k+1} - x_i^k| < \varepsilon, \quad i = 1, 2, \quad \text{или} \quad 2) \quad |f(x^{k+1}) - f(x^k)| < \varepsilon.$$

Метод прост, но не очень эффективен. Проблемы возникают, когда линии уровня сильно вытянуты вдоль прямых $x_1 = x_2$. Если начальное приближение оказывается на $x_1 = x_2$, то процесс так и останется в этой точке.


x0
x1
x2
x3
x4

Метод Нелдера-Мида (Nelder-Mead)

В процессе поиска используются симплексы в пространстве $\mathbb{R}^n$.
В нашем случае ($n = 2$) — треугольник.
Эвристический подход – может застревать в локальных минимумах или сходиться не к точкам минимума.



Numerical Methods Using Matlab, 4th Edition, 2004
John H. Mathews and Kurtis K. Fink

Алгоритм (нахождения минимума функции $f(x^{(1)}, x^{(2)})$)

Параметры: коэф. отражения $\alpha > 0$ (обычно $\alpha = 1$), коэф. сжатия $\beta > 0$ (обычно $\beta = 0.5$), коэф. растяжения $\gamma > 0$ (обычно $\gamma = 2$).

Шаг 1 (подготовка). Вначале выбираются три точки $x_i = (x_i^{(1)}, x_i^{(2)})$, $i = 1, 2, 3$, образующие симплекс (в нашем случае — треугольник). В этих точках вычисляются значения функции: $f_1 = f(x_1)$, $f_2 = f(x_2)$, $f_3 = f(x_3)$.

Шаг 2 (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.

Шаг 3 (центр тяжести). Найдем центр тяжести всех точек, за исключением x_h: $x_c = \frac{1}{2} \sum_{i \neq h} x_i$.

Шаг 4 (отражение). Отразим точку x_h относительно x_c с коэффициентом α (при $\alpha = 1$ это будет центральная симметрия), получим точку $x_r = (1 + \alpha)x_c - \alpha x_h$ и вычислим $f_r = f(x_r)$.

Алгоритм (продолжение)

Шаг 5. Далее смотрим, насколько нам удалось уменьшить функцию, ищем место f_r в ряду f_h, f_g, f_l.

- Если $f_r < f_l$, то направление выбрано удачное и можно попробовать увеличить шаг. Производим растяжение. Новая точка $x_e = (1-\gamma)x_c + \gamma x_r$ и значение функции $f_e = f(x_e)$.
- Если $f_e < f_r$, то можно расширить симплекс до этой точки: присваиваем точке x_h значение x_e и заканчиваем итерацию (на Шаг 7).
- Если $f_r < f_e$, то переместились слишком далеко: присваиваем точке x_h значение x_r и заканчиваем итерацию (на Шаг 7).
- Если $f_l < f_r < f_g$, то выбор точки неплохой (новая лучше двух прежних). Присваиваем точке x_h значение x_r и переходим на Шаг 7.
- 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 идем на Шаг 6.
- Если $f_h < f_r$, то идем на Шаг 6.

В результате $f_l < f_g < f_h < f_r$.

Алгоритм (продолжение)

Шаг 6 (сжатие). Строим точку $x_s = \beta x_h + (1 - \beta)x_c$ и вычисляем $f_s = f(x_s)$.

- Если $f_s < f_h$, то присваиваем точке x_h значение x_s и идем на Шаг 7.
- Если $f_s > f_h$, то первоначальные точки оказались самыми удачными.

Делаем «глобальное сжатие» симплекса — гомотетию к точке с наименьшим значением $x_l : x_i := x_l + (x_i - x_l)/2$, $i \neq l$.

Шаг 7 (проверка сходимости). Проверяем взаимную близость полученных вершин симплекса, что предполагает и близость их к искомому минимуму (например, оценкой дисперсии набора точек).
Если требуемая точность не достигнута, переходим к Шагу 2.

Демонстрация 1
Демонстрация 2

Спасибо за внимание!