

## **СОДЕРЖАНИЕ**

## 1 Методы локальной оптимизации

Оптимизация – это задача нахождения экстремума (минимума или максимума) целевой функции в некоторой области конечномерного векторного пространства, ограниченной набором линейных и/или нелинейных равенств и/или неравенств.

Во многих практически важных случаях для целевой функции многих переменных  $f(\mathbf{x})$  задача оптимизации может быть сформулирована в виде:

$$f(\mathbf{x}) \rightarrow \min,$$

где  $\mathbf{x} = (x_1, x_2, \dots, x_n)$  – вектор неизвестных (управляющих параметров);  $\min$  – минимальное значение функции в ограниченной или неограниченной области изменения неизвестных.

Для нахождения абсолютного минимума целевой функции  $f(\mathbf{x})$  существует только один способ: найти все локальные минимумы этой функции, сравнить их и выбрать из них тот, в котором функция принимает наименьшее значение.

### 1.1 Минимум функции одного переменного

Для функции одной переменной  $f(x)$ , задача нахождения минимума эквивалента задачи нахождения корней уравнения:

$$\frac{df(x)}{dx} = 0 \quad (1)$$

Эта одномерная задача нередко возникает в практических приложениях. Кроме того, большинство методов решения многомерных задач сводится к поиску одномерного минимума.

Предположим, что  $f(x)$  задана и кусочно-непрерывна на отрезке  $x \in [a, b]$ , и имеет на этом отрезке (включая его концы) только один локальный минимум. Построим итерационный процесс, сходящийся к этому минимуму.

Вычислим значение функции на концах отрезка  $x = a$  и  $x = b$ , а также в двух внутренних точках  $x_1 < x_2$ . Так как функция  $f(x)$  имеет минимум на отрезке  $x \in [a, b]$ , то справедливо утверждение:

$$f(a) \geq f(x_1), \quad f(x_2) \leq f(b)$$

Сравним все четыре значения функции между собой  $f(a)$ ,  $f(x_1)$ ,  $f(x_2)$  и  $f(b)$  и выберем среди них наименьшее.

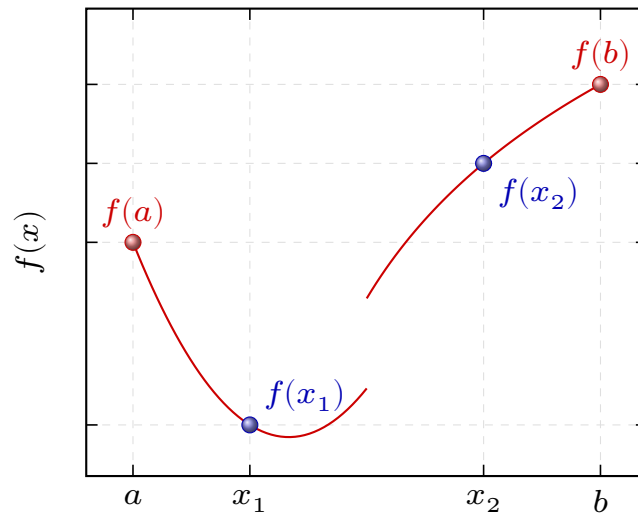


Рисунок 1 – График кусочно-непрерывной функции  $y = f(x)$ , имеющей минимум на отрезке  $x \in [a, b]$

Из рисунка ?? видно, что наименьшее значение функция достигает в точке  $x = x_1$ :

$$f(x_1) < f(a) < f(x_2) < f(b)$$

Очевидно, что минимум функции  $f(x)$  расположен в одном из прилегающих к точке  $x = x_1$  отрезков, то есть минимум находится либо в пределах отрезка  $[a, x_1]$ , либо в  $[x_1, x_2]$ .

Поэтому на первом шаге итерационного процесса отбрасывается отрезок  $[x_2, b]$ , и для поиска минимума функции  $f(x)$  рассматривается отрезок  $[a, x_2]$ , то есть область поиска минимума функции сужается:

$$|a - x_2| < |a - b|, \quad \text{так как} \quad x_2 < b.$$

На отрезке  $[a, x_2]$  вновь необходимо выбрать две внутренние точки, вычислить в них и на концах отрезка значения функции, и сделать следующий шаг итерационного процесса.

Практически на каждом шаге выбирается две точки  $x_1$  и  $x_2$  внутри отрезка  $[a, b]$ , равноудаленные от концов этого отрезка. Например, если точки  $x_1$  и

$x_2$  делят отрезок  $[a, b]$  на три равные части (рисунок ??), то координаты этих точек могут быть определены из соотношений:

$$x_1 = a + \frac{b-a}{3} = \frac{2a+b}{3}, \quad x_2 = b - \frac{b-a}{3} = \frac{a+2b}{3}.$$



Рисунок 2 – Схематическое изображение точек деления отрезка  $[a, b]$

Оценка длины отрезка после первого итерационного шага составит:

$$\ell_1 = (b-a) - \frac{b-a}{3} = \frac{2}{3} \cdot (b-a),$$

после второго шага:

$$\ell_2 = \ell_1 - \frac{\ell_1}{3} = \frac{2}{3} \cdot \ell_1 = \left(\frac{2}{3}\right)^2 \cdot (b-a),$$

а после  $k$ -ого итерационного шага:

$$\ell_k = \left(\frac{2}{3}\right)^k \cdot (b-a).$$

Таким образом, чтобы погрешность вычисления  $|x_k - x_{k-1}|$  была менее  $\epsilon$ , должна выполняться оценка числа итераций  $k$ :

$$\ell_k = \left(\frac{2}{3}\right)^k \cdot (b-a) \leq \epsilon$$

## 1.2 Метод градиентного спуска

Градиентный спуск – метод нахождения локального экстремума (минимума или максимума) функции многих переменных  $f(x_1, x_2, \dots, x_n)$  с помощью движения вдоль градиента этой функции. Это наиболее простой в реализации из всех методов локальной оптимизации, но имеет относительно малую (линейную) скорость сходимости.

Градиент  $\nabla$  это вектор, указывающий направление наибольшего возрас-

тания некоторой функции  $f$ , значение которой меняется от одной точки пространства к другой (скалярного поля), а по величине (модулю) равный скорости роста этой величины в этом направлении. Компонентами вектора градиента являются частные производные  $f$  по всем её аргументам:

$$\nabla f = \left( \frac{\partial f}{\partial x_1}, \frac{\partial f}{\partial x_2}, \dots, \frac{\partial f}{\partial x_n} \right) \quad (2)$$

Для случая трёхмерного пространства градиентом скалярной функции  $f(x, y, z)$  называется векторная функция:

$$\text{grad } f = \nabla f,$$

где  $\nabla$  – векторный дифференциальный оператор набла, компоненты которого являются частными производными по координатам:

$$\nabla = \left( \frac{\partial}{\partial x}, \frac{\partial}{\partial y}, \frac{\partial}{\partial z} \right)$$

Следует отметить, что оператор набла не принадлежит тому же пространству, что и обычные векторы, а говоря точнее, скалярное и векторное произведение для него определено с некоторыми отличиями. Оператор  $\nabla$  действует на те скалярные поля, что стоят от него справа, и не действует на стоящие от него слева. Поэтому скалярное и векторное произведение с участием  $\nabla$  *не коммутативны* и не антикоммутативны, как это свойственно для таких произведений обычных векторов.

Минимизация целевой функции  $f(\mathbf{x})$  сводится к итерационному процессу последовательного выбора нового вектора неизвестных  $\mathbf{x}_{k+1}$ , такого чтобы значение функции в новой точке было меньше чем в предыдущих:

$$f(\mathbf{x}_0) > f(\mathbf{x}_1) > \dots > f(\mathbf{x}_k) > f(\mathbf{x}_{k+1}) > \dots$$

Предполагая, что новый вектор неизвестных мало отличается от предыдущего ( $\mathbf{x}_{k+1} - \mathbf{x}_k \approx 0$ ), можно воспользоваться линейным приближением для разложения в ряд Тейлора целевой функции:

$$f(\mathbf{x}_{k+1}) = f(\mathbf{x}_k) + (\nabla f(\mathbf{x}_k), \mathbf{x}_{k+1} - \mathbf{x}_k), \quad (3)$$

где  $k$  – номер итерационного шага процесса;  $\mathbf{x}_k$  – значение неизвестных на  $k$ -ой

итерации.

Если в качестве нового вектора неизвестных выбрать:

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \lambda \cdot \nabla f(\mathbf{x}_k), \quad (4)$$

то из (??) получим:

$$f(\mathbf{x}_{k+1}) = f(\mathbf{x}_k) - \lambda \cdot \|\nabla f(\mathbf{x}_k)\|^2 \rightarrow f(\mathbf{x}_{k+1}) \leq f(\mathbf{x}_k) \quad (5)$$

где  $\lambda > 0$  – малое положительное число (параметр метода), имеющий смысл скорости градиентного спуска;  $\|\nabla f(\mathbf{x}_k)\| \geq 0$  – норма вектора градиента (неотрицательное число):

$$\|\nabla f\| = \sqrt{(\nabla f, \nabla f)}$$

Таким образом, выбор нового вектора неизвестных  $\mathbf{x}_{k+1}$  в соответствии с выражением (??), гарантирует монотонное убывание целевой функции  $f(\mathbf{x})$  в каждой итерации. Поэтому основная идея метода градиентного спуска заключается в том, чтобы последовательно идти в направлении наибольшего уменьшения целевой функции, которое задаётся антиградиентом  $-\nabla f(\mathbf{x})$ .

Практически можно задать некоторое число  $\varepsilon > 0$ , связанное с выбранной точностью вычислений, и проводить итерации до тех пор, пока на  $k$ -ой итерации не будут выполнены одно или несколько неравенств вида:

$$\|\mathbf{x}_k - \mathbf{x}_{k-1}\| < \varepsilon_1, \quad \|f(\mathbf{x}_k) - f(\mathbf{x}_{k-1})\| < \varepsilon_2 \quad (6)$$

### Алгоритм метода градиентного спуска

- 1) Задают начальное приближение  $(x_0, y_0)$ , скорость градиентного спуска  $\lambda$ , а также точность расчёта  $\varepsilon$ .
- 2) Рассчитывают градиент целевой функции в текущей точке  $\nabla_0 = \nabla f(x_0, y_0)$ .
- 3) Определяют новый вектор неизвестных в соответствии с соотношением (??):

$$\begin{cases} x_1 = x_0 - \lambda \cdot \nabla_{0x} \\ y_1 = y_0 - \lambda \cdot \nabla_{0y} \end{cases},$$

где  $\nabla_{0x}$  и  $\nabla_{0y}$  – компоненты вектора градиента в выбранной системе координат.

4) Рассчитывают величину расстояния между двумя точками:

$$r = \sqrt{(x_0 - x_1)^2 + (y_0 - y_1)^2}$$

5) Проверяют условие остановки итерационного процесса: если  $r < \varepsilon$ , то итерационный процесс останавливается; иначе текущую точку считают начальной  $x_0 = x_1$  и  $y_0 = y_1$  и переходят к шагу (2) итерационного процесса.

### 1.3 Метод тяжелого шара

Поиск минимума функции многих переменных  $f(x)$  методом “тяжелого шара” основан на аналогии движения материальной частицы массой  $m$  в консервативном силовом поле  $F(x)$  в вязкой среде.

В соответствии с принципом минимальной энергии тело смещается в положение, которое минимизирует общую потенциальную энергию системы  $f(x) \rightarrow \min$ . Поэтому если предположить, что функция  $f(x)$  является потенциальной энергией частицы в консервативном силовом поле  $F(x) = -\nabla f(x)$ , и частица перемещается в пространстве  $x$  минимизируя свою энергию, то уравнение движения этой частицы можно записать в виде:

$$\begin{cases} \frac{dx}{dt} = v \\ m \frac{dv}{dt} = F - \alpha \cdot v \end{cases} \quad (7)$$

где  $x$  – положение частицы в выбранной системе координат;  $v$  и  $\alpha$  – скорость и коэффициент вязкого трения частицы в среде, соответственно.

Этот метод используется в методе стохастического градиентного спуска и в качестве расширения алгоритмов обратного распространения ошибок для обучения искусственных нейронных сетей.

Поиск минимума данным методом начинается с задания начальных условий, которые, как правило, формулируются в виде:

$$\begin{cases} x(0) = x_0 \\ v(0) = v_0 \end{cases}, \quad (8)$$

где  $x_0$  – начальное приближения для поиска минимума функции;  $v_0$  – “начальная скорость” в пространстве неизвестных.

Масса частицы  $m$  и коэффициент вязкого трения  $\alpha$  являются эвристическими параметрами метода и выбираются произвольным образом, отражающим специфику решаемой задачи.