Векторно-матричное дифференцирование

Семинар 3

Агенда

- Зачем это нужно?
- Дифференцирование
- Gradient descent
- SGD

Зачем это нужно?

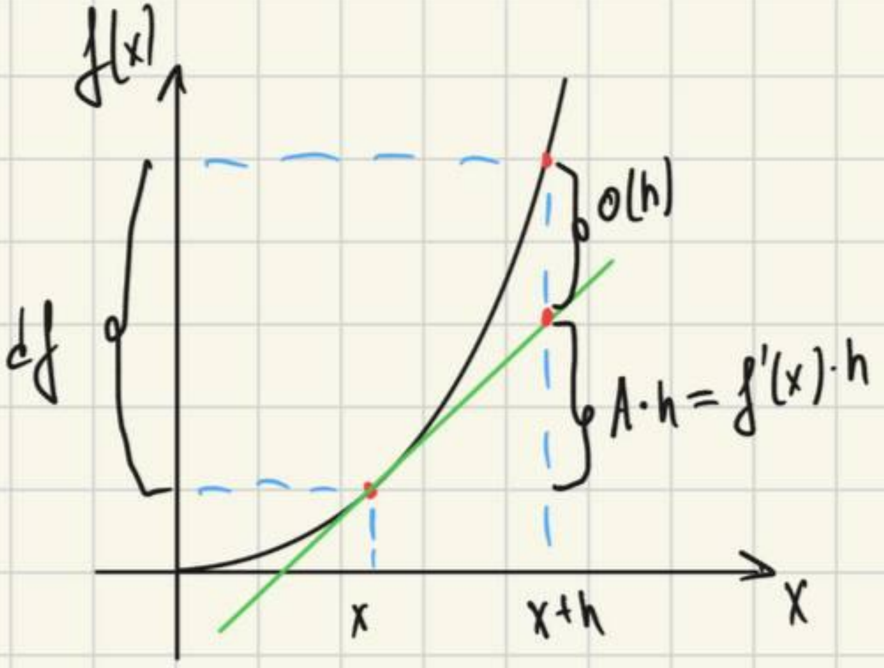
В ML/DL большинство моделей обучаются с помощью методов оптимизации, основанных на **градиентном спуске** или его модификациях.

Для этого необходимо уметь вычислять **градиенты функции потерь** относительно **параметров модели**.

Однако в задачах, где параметры представлены **векторами или матрицами**, покоординатное вычисление производных становится **громоздким и непрактичным**.

Одномерный случай

$$df = f(x+h) - f(x) = \underset{h \to 0}{A \cdot h} + o(h) = f'(x) \cdot h + o(h)$$

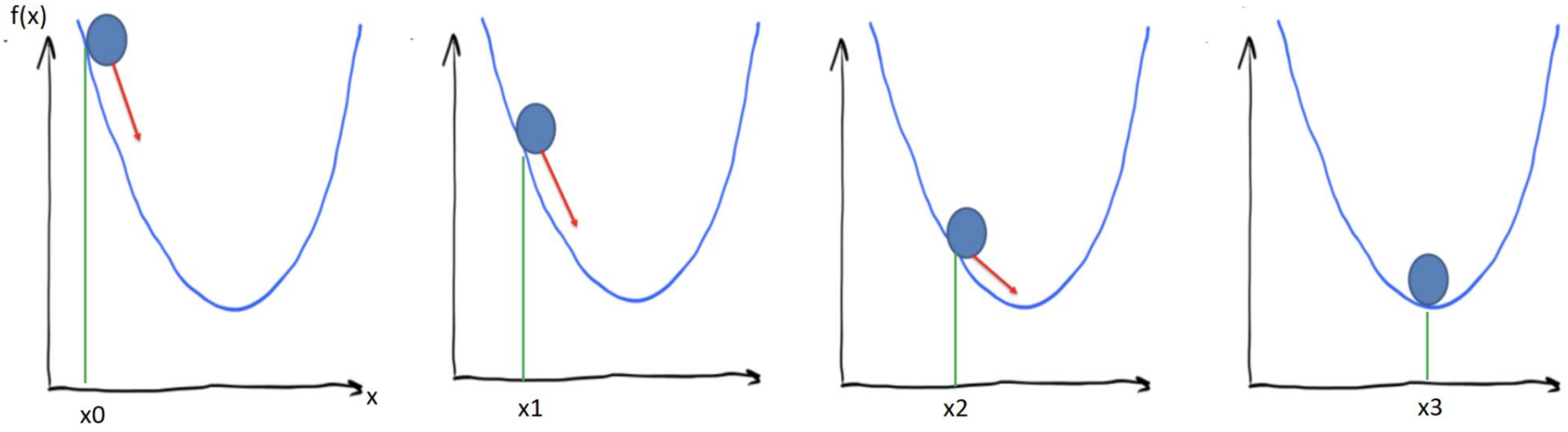


$$\boxed{df = A \cdot h = f'(x) \cdot h}$$

Дифференциал

Одномерный случай

Геометрический смысл производной



Многомерная функция

$$f(x_1, \dots, x_n)$$

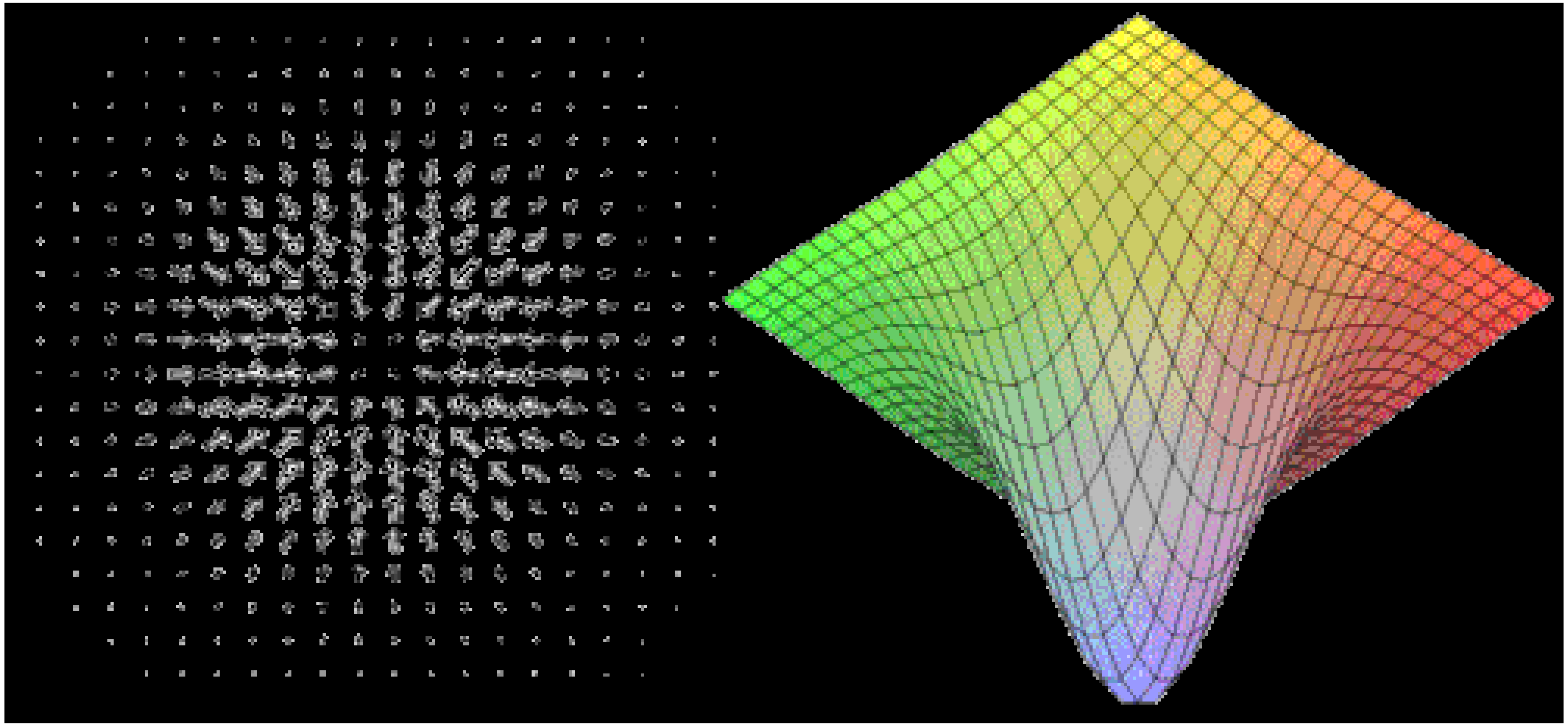
$$\nabla f = \begin{pmatrix} \frac{\partial f}{\partial x_1} \\ \dots \\ \frac{\partial f}{\partial x_n} \end{pmatrix}$$

градиент

$$df = \frac{\partial f}{\partial x_1} \cdot dx_1 + \dots + \frac{\partial f}{\partial x_n} dx_n = \langle \nabla f; dx \rangle = \underset{[1 \times n]}{\nabla^T f} \; \underset{[n \times 1]}{dx}$$

Многомерная функция

Геометрический сымсл градиента



Дифференциал

Правила преобразования
$dA = 0$
$d(\alpha X) = \alpha(dX)$
$d(AXB) = A(dX)B$
$d(X+Y) = dX + dY$
$d(X^T) = (dX)^T$
$d(XY) = (dX)Y + X(dY)$
$d\langle X, Y\rangle = \langle dX, Y\rangle + \langle X, dY\rangle$
$d\left(\frac{X}{\phi}\right) = \frac{\phi dX - (d\phi)X}{\phi^2}$

Таблица стандартных производных
$d\langle A, X\rangle = \langle A, dX\rangle$
$d\langle Ax, x\rangle = \langle (A + A^T)x, dx\rangle$
$d\langle Ax, x\rangle = 2\langle Ax, dx\rangle \qquad (\textbf{если } A = A^T)$
$d(\mathrm{Det}(X)) = \mathrm{Det}(X)\langle X^{-T}, dX\rangle$
$d(X^{-1}) = -X^{-1}(dX)X^{-1}$

Матричнозначная функция

$$f\begin{pmatrix} X_{11} & \dots & X_{1n} \\ & \cdots & \\ X_{m1} & \dots & X_{mn} \end{pmatrix}$$

$$df = \sum_{i=1}^{m} \sum_{j=1}^{n} A_{i,j}\, dX_{i,j} + o(\|H\|_F) = \langle A, dX \rangle_F + o(\|H\|_F)$$

$$\boxed{A = \nabla f(X) = \left(\frac{\partial f}{\partial X_{i,j}}\right)_{i,j=1}^{m,n}}$$

$$\langle X, Y \rangle_F = \sum_{i=1}^{m} \sum_{j=1}^{n} X_{i,j} Y_{i,j} = T_r(X^T Y)$$

$$\|X\|_F = \sqrt{\langle X, X \rangle_F}$$

Линейная регрессия в матричном виде

$$L(w) = \frac{1}{N}\|Xw - y\|^2 = \frac{1}{N}\sum_{i=1}^{N}\left(\langle x_i, w\rangle - y\right)^2$$

$$dL(w) = d\left(\frac{1}{N}\|Xw - y\|^2\right) = \frac{1}{N}\, d\left((Xw - y)^T (Xw - y)\right) =$$

$$= \frac{1}{N}\, d\langle (Xw - y), (Xw - y)\rangle = \frac{1}{N}\left(\langle (Xw - y), d(Xw - y)\rangle + \langle (Xw - y), d(Xw - y)\rangle\right) =$$

$$= \frac{2}{N}\langle (Xw - y), d(Xw - y)\rangle$$

Линейная регрессия в матричном виде

$$d(Xw - y) = d(Xw) - 0 = X\,dw$$

$$\Downarrow$$

$$dL(w) = \frac{2}{N}\langle (Xw - y), d(Xw - y)\rangle = \frac{2}{N}\langle (Xw - y), X\,dw\rangle =$$

$$\langle a, Xb\rangle = \langle X^T a, b\rangle$$

$$= \frac{2}{N}\langle X^T(Xw - y), dw\rangle$$

$$\Downarrow$$

$$\nabla L(w) = \frac{2}{N} X^T (Xw - y)$$

Аналитическое решение линейной регрессии

$$\nabla L(w) = \frac{2}{N} X^T (Xw - y) = 0$$

$$X^T X w = X^T y$$

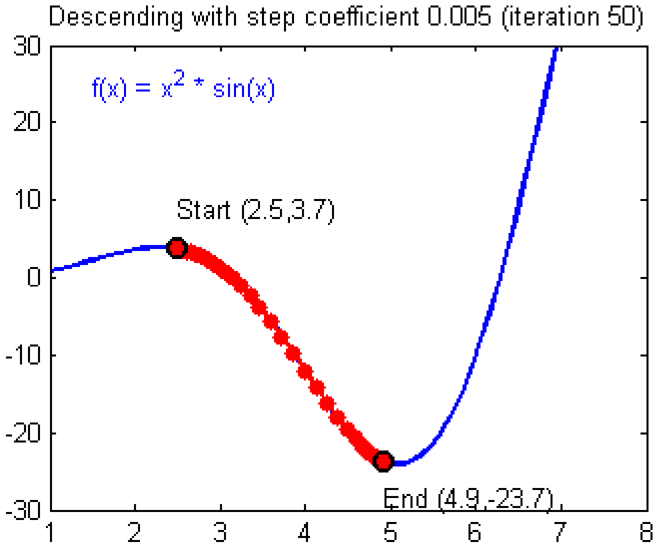
$$\underbrace{(X^T X)^{-1} (X^T X)}_{I} w = (X^T X)^{-1} X^T y$$

$$w = (X^T X)^{-1} X^T y$$

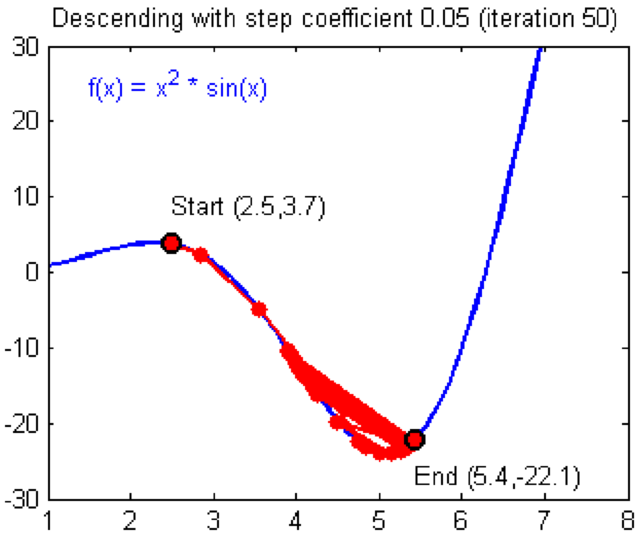
- Обращение матрицы O(d^3)
- Если матрица $X^T X$ необратима, то решений нет

Градиентный спуск

$$w_{new} = w_{old} - \eta \nabla Q(w_{old})$$



Маленький learning rate



Большой learning rate

Градиентный спуск

Критерии останова

Когда останавливать градиентный спуск? Разумно завершить поиск минимума, если выполнено одно или несколько из следующих *критериев останова*:

- $||w_{new} - w_{old}|| < \varepsilon$: вектор весов практически не изменился на соседних итерациях..
- $|Q(w_{new}) - Q(w_{old})| < \varepsilon$: значения функции потерь почти не изменились на соседних итерациях.
- $||\nabla Q(w_{new})|| < \varepsilon$: величина градиента близка к нулю, что указывает на то, что мы находимся около минимума.

SGD

Алгоритм:

- Выбираем случайным образом стартовые параметры метода w_0.
- На каждой итерации:
    - случайным образом выбираем один объект из выборки (пусть ind - его индекс);
    - обновляем веса по формуле:
      $w_{new} = w_{old} - \eta \nabla q_{ind}(w_{old})$, где $\nabla q_{ind}(w_{old})$ - градиент функции потерь, посчитанный на объекте с индексом ind и вычисленный в точке w_{old}.

Mini-Batch Gradient Descent

$$w_{new} = w_{old} - \frac{\eta}{N} \sum_{i=1}^{N} \nabla q_i(w_{old}).$$


Нет, спасибо, я использую ИИ