

Повторение материала первого семестра

Даниил Меркулов

Методы оптимизации. МФТИ



# Векторы и матрицы

## Векторные нормы:

- $\|x\|_1 = \sum_{i=1}^n |x_i|$  —  $\ell_1$ -норма

# Векторы и матрицы

## Векторные нормы:

- $\|x\|_1 = \sum_{i=1}^n |x_i|$  —  $\ell_1$ -норма
- $\|x\|_2 = \sqrt{\sum_{i=1}^n x_i^2}$  — евклидова норма

# Векторы и матрицы

## Векторные нормы:

- $\|x\|_1 = \sum_{i=1}^n |x_i|$  —  $\ell_1$ -норма
- $\|x\|_2 = \sqrt{\sum_{i=1}^n x_i^2}$  — евклидова норма
- $\|x\|_\infty = \max_i |x_i|$  —  $\ell_\infty$ -норма

# Векторы и матрицы

## Векторные нормы:

- $\|x\|_1 = \sum_{i=1}^n |x_i|$  —  $\ell_1$ -норма
- $\|x\|_2 = \sqrt{\sum_{i=1}^n x_i^2}$  — евклидова норма
- $\|x\|_\infty = \max_i |x_i|$  —  $\ell_\infty$ -норма
- $\|x\|_p = \left(\sum_{i=1}^n |x_i|^p\right)^{1/p}$  —  $\ell_p$ -норма

# Векторы и матрицы

## Векторные нормы:

- $\|x\|_1 = \sum_{i=1}^n |x_i|$  —  $\ell_1$ -норма
- $\|x\|_2 = \sqrt{\sum_{i=1}^n x_i^2}$  — евклидова норма
- $\|x\|_\infty = \max_i |x_i|$  —  $\ell_\infty$ -норма
- $\|x\|_p = \left(\sum_{i=1}^n |x_i|^p\right)^{1/p}$  —  $\ell_p$ -норма

# Векторы и матрицы

## Векторные нормы:

- $\|x\|_1 = \sum_{i=1}^n |x_i|$  —  $\ell_1$ -норма
- $\|x\|_2 = \sqrt{\sum_{i=1}^n x_i^2}$  — евклидова норма
- $\|x\|_\infty = \max_i |x_i|$  —  $\ell_\infty$ -норма
- $\|x\|_p = \left(\sum_{i=1}^n |x_i|^p\right)^{1/p}$  —  $\ell_p$ -норма

**Неравенство Гёльдера:**  $|\langle x, y \rangle| \leq \|x\|_p \|y\|_q$ , где

$$\frac{1}{p} + \frac{1}{q} = 1$$

## Матричные нормы:

- $\|A\|_F = \sqrt{\sum_{i,j} a_{ij}^2} = \sqrt{\text{tr}(A^T A)}$  — норма Фробениуса

# Векторы и матрицы

## Векторные нормы:

- $\|x\|_1 = \sum_{i=1}^n |x_i|$  —  $\ell_1$ -норма
- $\|x\|_2 = \sqrt{\sum_{i=1}^n x_i^2}$  — евклидова норма
- $\|x\|_\infty = \max_i |x_i|$  —  $\ell_\infty$ -норма
- $\|x\|_p = \left(\sum_{i=1}^n |x_i|^p\right)^{1/p}$  —  $\ell_p$ -норма

**Неравенство Гёльдера:**  $|\langle x, y \rangle| \leq \|x\|_p \|y\|_q$ , где

$$\frac{1}{p} + \frac{1}{q} = 1$$

## Матричные нормы:

- $\|A\|_F = \sqrt{\sum_{i,j} a_{ij}^2} = \sqrt{\text{tr}(A^T A)}$  — норма Фробениуса
- $\|A\|_* = \sum_i \sigma_i(A)$  — ядерная норма

# Векторы и матрицы

## Векторные нормы:

- $\|x\|_1 = \sum_{i=1}^n |x_i|$  —  $\ell_1$ -норма
- $\|x\|_2 = \sqrt{\sum_{i=1}^n x_i^2}$  — евклидова норма
- $\|x\|_\infty = \max_i |x_i|$  —  $\ell_\infty$ -норма
- $\|x\|_p = \left(\sum_{i=1}^n |x_i|^p\right)^{1/p}$  —  $\ell_p$ -норма

**Неравенство Гёльдера:**  $|\langle x, y \rangle| \leq \|x\|_p \|y\|_q$ , где

$$\frac{1}{p} + \frac{1}{q} = 1$$

## Матричные нормы:

- $\|A\|_F = \sqrt{\sum_{i,j} a_{ij}^2} = \sqrt{\text{tr}(A^T A)}$  — норма Фробениуса
- $\|A\|_* = \sum_i \sigma_i(A)$  — ядерная норма
- $\|A\|_2 = \sigma_{\max}(A)$  — спектральная норма

# Векторы и матрицы

## Векторные нормы:

- $\|x\|_1 = \sum_{i=1}^n |x_i|$  —  $\ell_1$ -норма
- $\|x\|_2 = \sqrt{\sum_{i=1}^n x_i^2}$  — евклидова норма
- $\|x\|_\infty = \max_i |x_i|$  —  $\ell_\infty$ -норма
- $\|x\|_p = \left(\sum_{i=1}^n |x_i|^p\right)^{1/p}$  —  $\ell_p$ -норма

**Неравенство Гёльдера:**  $|\langle x, y \rangle| \leq \|x\|_p \|y\|_q$ , где

$$\frac{1}{p} + \frac{1}{q} = 1$$

## Матричные нормы:

- $\|A\|_F = \sqrt{\sum_{i,j} a_{ij}^2} = \sqrt{\text{tr}(A^T A)}$  — норма Фробениуса
- $\|A\|_* = \sum_i \sigma_i(A)$  — ядерная норма
- $\|A\|_2 = \sigma_{\max}(A)$  — спектральная норма

# Векторы и матрицы

## Векторные нормы:

- $\|x\|_1 = \sum_{i=1}^n |x_i|$  —  $\ell_1$ -норма
- $\|x\|_2 = \sqrt{\sum_{i=1}^n x_i^2}$  — евклидова норма
- $\|x\|_\infty = \max_i |x_i|$  —  $\ell_\infty$ -норма
- $\|x\|_p = \left(\sum_{i=1}^n |x_i|^p\right)^{1/p}$  —  $\ell_p$ -норма

**Неравенство Гёльдера:**  $|\langle x, y \rangle| \leq \|x\|_p \|y\|_q$ , где

$$\frac{1}{p} + \frac{1}{q} = 1$$

## Матричные нормы:

- $\|A\|_F = \sqrt{\sum_{i,j} a_{ij}^2} = \sqrt{\text{tr}(A^T A)}$  — норма Фробениуса
- $\|A\|_* = \sum_i \sigma_i(A)$  — ядерная норма
- $\|A\|_2 = \sigma_{\max}(A)$  — спектральная норма

**Индукционная норма:**  $\|A\| = \max_{\|x\|=1} \|Ax\|$

## Собственные значения и SVD

**Спектральное разложение** (для симметричных матриц  $A = A^T$ ):

$$A = Q\Lambda Q^T, \quad Q^T Q = I$$

где  $\Lambda = \text{diag}(\lambda_1, \dots, \lambda_n)$  — собственные значения.

## Собственные значения и SVD

**Спектральное разложение** (для симметричных матриц  $A = A^T$ ):

$$A = Q\Lambda Q^T, \quad Q^T Q = I$$

где  $\Lambda = \text{diag}(\lambda_1, \dots, \lambda_n)$  — собственные значения.

**Положительная определённость:**  $A \succ 0 \Leftrightarrow \lambda_i > 0 \ \forall i$

# Собственные значения и SVD

**Спектральное разложение** (для симметричных матриц  $A = A^T$ ):

$$A = Q\Lambda Q^T, \quad Q^T Q = I$$

где  $\Lambda = \text{diag}(\lambda_1, \dots, \lambda_n)$  — собственные значения.

**Положительная определённость:**  $A \succ 0 \Leftrightarrow \lambda_i > 0 \ \forall i$

**Число обусловленности:**  $\kappa(A) = \frac{\lambda_{\max}(A)}{\lambda_{\min}(A)}$

**Сингулярное разложение (SVD):**

$$A = U\Sigma V^T$$

где  $U, V$  — ортогональные,  $\Sigma = \text{diag}(\sigma_1, \dots, \sigma_r)$

# Собственные значения и SVD

**Спектральное разложение** (для симметричных матриц  $A = A^T$ ):

$$A = Q\Lambda Q^T, \quad Q^T Q = I$$

где  $\Lambda = \text{diag}(\lambda_1, \dots, \lambda_n)$  — собственные значения.

**Положительная определённость:**  $A \succ 0 \Leftrightarrow \lambda_i > 0 \ \forall i$

**Число обусловленности:**  $\kappa(A) = \frac{\lambda_{\max}(A)}{\lambda_{\min}(A)}$

**Сингулярное разложение (SVD):**

$$A = U\Sigma V^T$$

где  $U, V$  — ортогональные,  $\Sigma = \text{diag}(\sigma_1, \dots, \sigma_r)$

**Низкоранговое приближение (теорема Эккарта-Янга):**

$$\min_{\text{rank}(B) \leq k} \|A - B\|_F = \sqrt{\sum_{i>k} \sigma_i^2}$$

# Собственные значения и SVD

**Спектральное разложение** (для симметричных матриц  $A = A^T$ ):

$$A = Q\Lambda Q^T, \quad Q^T Q = I$$

где  $\Lambda = \text{diag}(\lambda_1, \dots, \lambda_n)$  — собственные значения.

**Положительная определённость:**  $A \succ 0 \Leftrightarrow \lambda_i > 0 \forall i$

**Число обусловленности:**  $\kappa(A) = \frac{\lambda_{\max}(A)}{\lambda_{\min}(A)}$

**Сингулярное разложение (SVD):**

$$A = U\Sigma V^T$$

где  $U, V$  — ортогональные,  $\Sigma = \text{diag}(\sigma_1, \dots, \sigma_r)$

**Низкоранговое приближение (теорема Эккарта-Янга):**

$$\min_{\text{rank}(B) \leq k} \|A - B\|_F = \sqrt{\sum_{i>k} \sigma_i^2}$$

**LoRA:**  $W = W_0 + BA$ , где  $B \in \mathbb{R}^{m \times r}$ ,  $A \in \mathbb{R}^{r \times n}$ ,  $r \ll \min(m, n)$

## Задача (след и собственные значения)

**i**

Упростите выражение:

$$\operatorname{tr}((A + \lambda I)^{-1} A), \quad \text{где } A \in \mathbb{S}_{++}^n, \lambda > 0$$

## Задача (след и собственные значения)

**i** Упростите выражение:

$$\operatorname{tr}((A + \lambda I)^{-1} A), \quad \text{где } A \in \mathbb{S}_{++}^n, \lambda > 0$$

**Решение:** Пусть  $A = Q\Lambda Q^T$  — спектральное разложение. Тогда:

$$(A + \lambda I)^{-1} = Q(\Lambda + \lambda I)^{-1}Q^T$$

## Задача (след и собственные значения)

**i** Упростите выражение:

$$\operatorname{tr}((A + \lambda I)^{-1} A), \quad \text{где } A \in \mathbb{S}_{++}^n, \lambda > 0$$

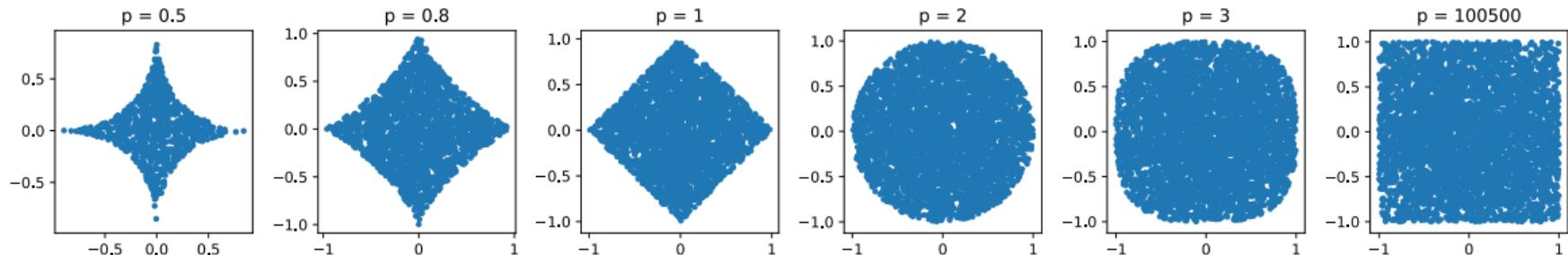
**Решение:** Пусть  $A = Q\Lambda Q^T$  — спектральное разложение. Тогда:

$$(A + \lambda I)^{-1} = Q(\Lambda + \lambda I)^{-1}Q^T$$

$$\operatorname{tr}((A + \lambda I)^{-1} A) = \operatorname{tr}((\Lambda + \lambda I)^{-1} \Lambda) = \sum_{i=1}^n \frac{\lambda_i}{\lambda_i + \lambda}$$

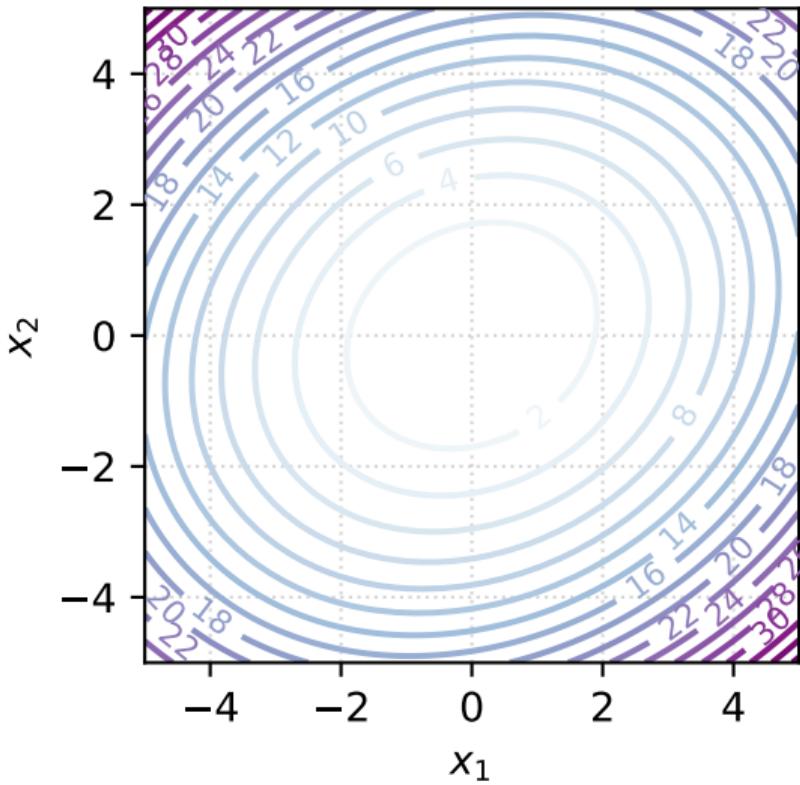
# Шары в разных нормах

Unit disk in the  $p$ -th norm

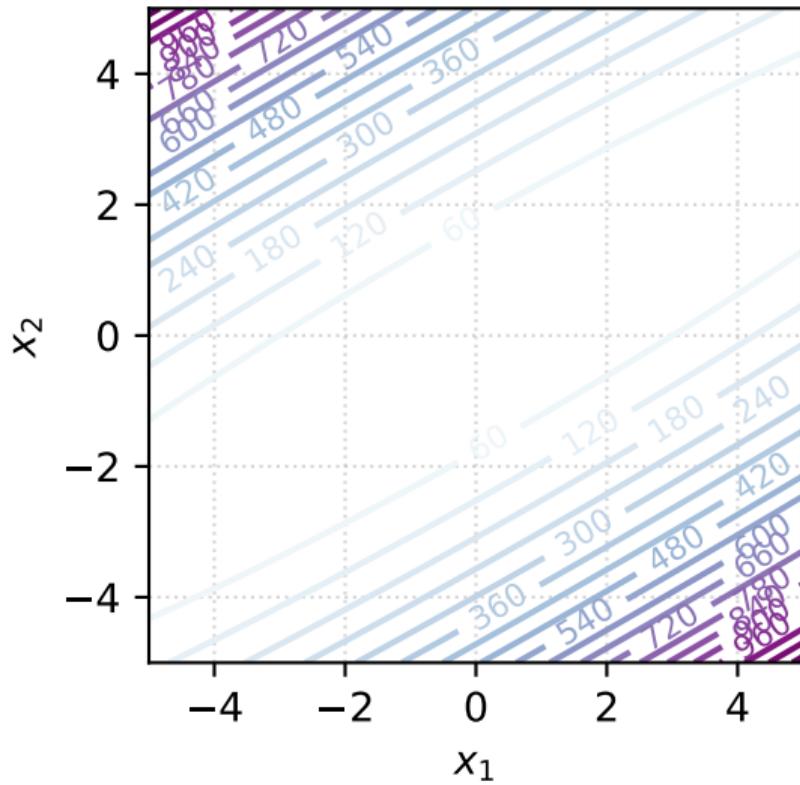


## Число обусловленности (геометрическая интерпретация)

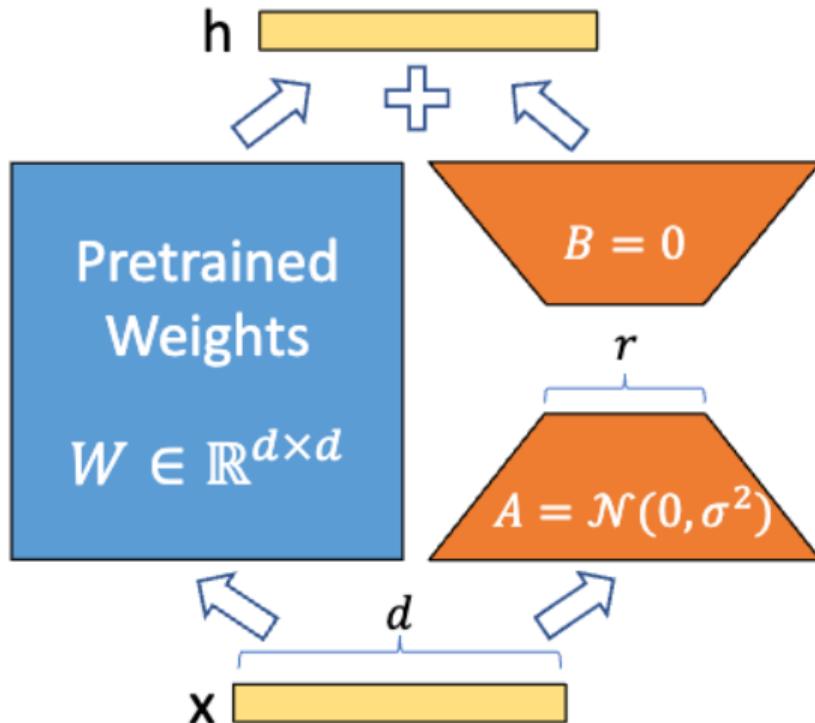
$$\kappa = 1.5$$



$$\kappa = 50$$



## LoRA: Low-Rank Adaptation



## Автоматическое дифференцирование (Лекция 2)

## Градиент, гессиан, якобиан

Градиент  $f : \mathbb{R}^n \rightarrow \mathbb{R}$ :

$$\nabla f(x) = \left( \frac{\partial f}{\partial x_1}, \dots, \frac{\partial f}{\partial x_n} \right)^T$$

## Градиент, гессиан, якобиан

Градиент  $f : \mathbb{R}^n \rightarrow \mathbb{R}$ :

$$\nabla f(x) = \left( \frac{\partial f}{\partial x_1}, \dots, \frac{\partial f}{\partial x_n} \right)^T$$

Гессиан:

$$\nabla^2 f(x) = \begin{pmatrix} \frac{\partial^2 f}{\partial x_1^2} & \dots & \frac{\partial^2 f}{\partial x_1 \partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial^2 f}{\partial x_n \partial x_1} & \dots & \frac{\partial^2 f}{\partial x_n^2} \end{pmatrix}$$

Якобиан  $f : \mathbb{R}^n \rightarrow \mathbb{R}^m$ :

$$J_f(x) = \begin{pmatrix} \frac{\partial f_1}{\partial x_1} & \dots & \frac{\partial f_1}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_m}{\partial x_1} & \dots & \frac{\partial f_m}{\partial x_n} \end{pmatrix}$$

# Градиент, гессиан, якобиан

**Градиент**  $f : \mathbb{R}^n \rightarrow \mathbb{R}$ :

$$\nabla f(x) = \left( \frac{\partial f}{\partial x_1}, \dots, \frac{\partial f}{\partial x_n} \right)^T$$

**Гессиан:**

$$\nabla^2 f(x) = \begin{pmatrix} \frac{\partial^2 f}{\partial x_1^2} & \dots & \frac{\partial^2 f}{\partial x_1 \partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial^2 f}{\partial x_n \partial x_1} & \dots & \frac{\partial^2 f}{\partial x_n^2} \end{pmatrix}$$

**Якобиан**  $f : \mathbb{R}^n \rightarrow \mathbb{R}^m$ :

$$J_f(x) = \begin{pmatrix} \frac{\partial f_1}{\partial x_1} & \dots & \frac{\partial f_1}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_m}{\partial x_1} & \dots & \frac{\partial f_m}{\partial x_n} \end{pmatrix}$$

**Цепное правило:**

$$J_{f \circ g}(x) = J_f(g(x)) \cdot J_g(x)$$

## Аппроксимации Тейлора

**Первый порядок** (линейное приближение вблизи  $x_0$ ):

$$f_{x_0}^I(x) = f(x_0) + \nabla f(x_0)^T(x - x_0)$$

# Аппроксимации Тейлора

**Первый порядок** (линейное приближение вблизи  $x_0$ ):

$$f_{x_0}^I(x) = f(x_0) + \nabla f(x_0)^T(x - x_0)$$

**Второй порядок** (квадратичное приближение вблизи  $x_0$ ):

$$f_{x_0}^{II}(x) = f(x_0) + \nabla f(x_0)^T(x - x_0) + \frac{1}{2}(x - x_0)^T \nabla^2 f(x_0)(x - x_0)$$

## Задача (градиент и гессиан)

**i**

Рассмотрим квадратичную функцию  $f(x) = \frac{1}{2}(x_1^2 + 4x_2^2)$ ,  $x \in \mathbb{R}^2$ .

1) Найдите градиент и гессиан  $\nabla^2 f(x)$ .

## Задача (градиент и гессиан)

**i**

Рассмотрим квадратичную функцию  $f(x) = \frac{1}{2}(x_1^2 + 4x_2^2)$ ,  $x \in \mathbb{R}^2$ .

- 1) Найдите градиент и гессиан  $\nabla^2 f(x)$ .
- 2) Найдите константу сильной выпуклости  $\mu$  и константу гладкости  $L$ .

## Задача (градиент и гессиан)

**i**

Рассмотрим квадратичную функцию  $f(x) = \frac{1}{2}(x_1^2 + 4x_2^2)$ ,  $x \in \mathbb{R}^2$ .

- 1) Найдите градиент и гессиан  $\nabla^2 f(x)$ .
- 2) Найдите константу сильной выпуклости  $\mu$  и константу гладкости  $L$ .

## Задача (градиент и гессиан)

**i**

Рассмотрим квадратичную функцию  $f(x) = \frac{1}{2}(x_1^2 + 4x_2^2)$ ,  $x \in \mathbb{R}^2$ .

- 1) Найдите градиент и гессиан  $\nabla^2 f(x)$ .
- 2) Найдите константу сильной выпуклости  $\mu$  и константу гладкости  $L$ .

**Решение:**  $\nabla f(x) = (x_1, 4x_2)^T$ ,  $\nabla^2 f(x) = \begin{pmatrix} 1 & 0 \\ 0 & 4 \end{pmatrix}$

## Задача (градиент и гессиан)

**i**

Рассмотрим квадратичную функцию  $f(x) = \frac{1}{2}(x_1^2 + 4x_2^2)$ ,  $x \in \mathbb{R}^2$ .

- 1) Найдите градиент и гессиан  $\nabla^2 f(x)$ .
- 2) Найдите константу сильной выпуклости  $\mu$  и константу гладкости  $L$ .

**Решение:**  $\nabla f(x) = (x_1, 4x_2)^T$ ,  $\nabla^2 f(x) = \begin{pmatrix} 1 & 0 \\ 0 & 4 \end{pmatrix}$

Собственные числа гессиана: 1 и 4. Следовательно,  $\mu = 1$ ,  $L = 4$ .

# Forward и Backward mode AD

**Forward mode** (прямой проход):

- Вычисляет  $J_f \cdot v$  за один проход

**Backward mode** (обратный проход):

# Forward и Backward mode AD

**Forward mode** (прямой проход):

- Вычисляет  $J_f \cdot v$  за один проход
- Эффективен когда  $n \ll m$  (мало входов)

**Backward mode** (обратный проход):

# Forward и Backward mode AD

**Forward mode** (прямой проход):

- Вычисляет  $J_f \cdot v$  за один проход
- Эффективен когда  $n \ll m$  (мало входов)
- Используется в JVP (Jacobian-vector product)

**Backward mode** (обратный проход):

# Forward и Backward mode AD

**Forward mode** (прямой проход):

- Вычисляет  $J_f \cdot v$  за один проход
- Эффективен когда  $n \ll m$  (мало входов)
- Используется в JVP (Jacobian-vector product)

**Backward mode** (обратный проход):

- Вычисляет  $v^T \cdot J_f$  за один проход

# Forward и Backward mode AD

**Forward mode** (прямой проход):

- Вычисляет  $J_f \cdot v$  за один проход
- Эффективен когда  $n \ll m$  (мало входов)
- Используется в JVP (Jacobian-vector product)

**Backward mode** (обратный проход):

- Вычисляет  $v^T \cdot J_f$  за один проход
- Эффективен когда  $n \gg m$  (много входов)

# Forward и Backward mode AD

**Forward mode** (прямой проход):

- Вычисляет  $J_f \cdot v$  за один проход
- Эффективен когда  $n \ll m$  (мало входов)
- Используется в JVP (Jacobian-vector product)

**Backward mode** (обратный проход):

- Вычисляет  $v^T \cdot J_f$  за один проход
- Эффективен когда  $n \gg m$  (много входов)
- Используется в VJP (vector-Jacobian product)

# Forward и Backward mode AD

**Forward mode** (прямой проход):

- Вычисляет  $J_f \cdot v$  за один проход
- Эффективен когда  $n \ll m$  (мало входов)
- Используется в JVP (Jacobian-vector product)

**Backward mode** (обратный проход):

- Вычисляет  $v^T \cdot J_f$  за один проход
- Эффективен когда  $n \gg m$  (много входов)
- Используется в VJP (vector-Jacobian product)
- **Backpropagation** в нейросетях!

# Forward и Backward mode AD

**Forward mode** (прямой проход):

- Вычисляет  $J_f \cdot v$  за один проход
- Эффективен когда  $n \ll m$  (мало входов)
- Используется в JVP (Jacobian-vector product)

**Backward mode** (обратный проход):

- Вычисляет  $v^T \cdot J_f$  за один проход
- Эффективен когда  $n \gg m$  (много входов)
- Используется в VJP (vector-Jacobian product)
- **Backpropagation** в нейросетях!

# Forward и Backward mode AD

**Forward mode** (прямой проход):

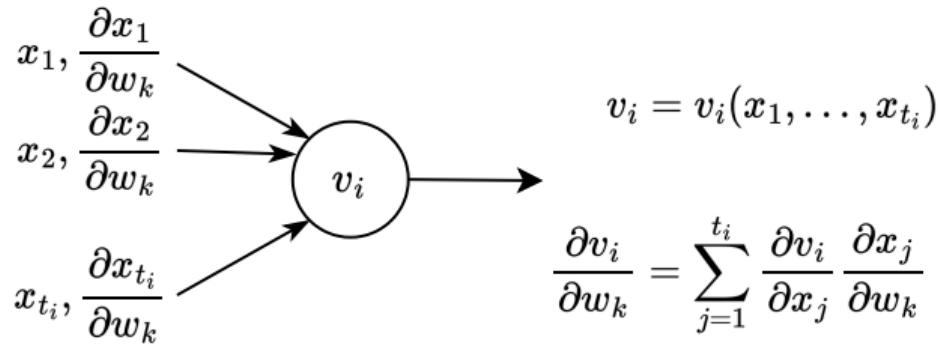
- Вычисляет  $J_f \cdot v$  за один проход
- Эффективен когда  $n \ll m$  (мало входов)
- Используется в JVP (Jacobian-vector product)

**Backward mode** (обратный проход):

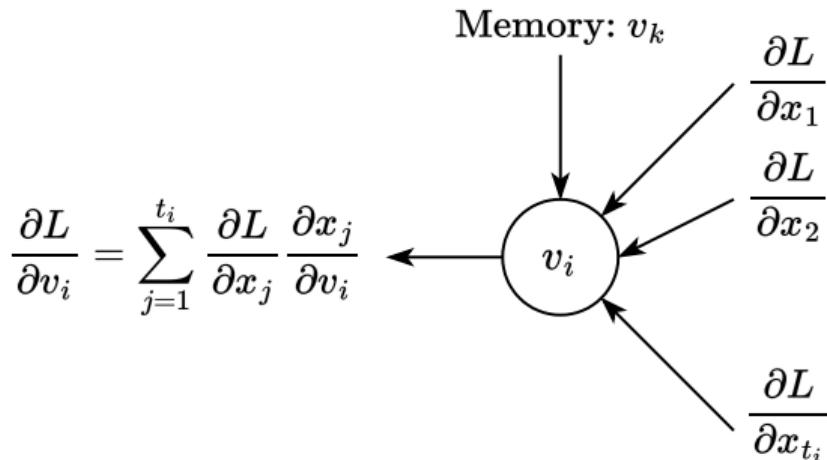
- Вычисляет  $v^T \cdot J_f$  за один проход
- Эффективен когда  $n \gg m$  (много входов)
- Используется в VJP (vector-Jacobian product)
- **Backpropagation** в нейросетях!

Для  $f : \mathbb{R}^n \rightarrow \mathbb{R}$  (типичная функция потерь): backward mode вычисляет полный градиент за  $O(1)$  проходов, forward mode — за  $O(n)$  проходов.

## Forward mode AD



## Backward mode AD (Backpropagation)



## Выпуклость (Лекция 3)

# Выпуклые множества и функции

**Выпуклое множество:**  $S$  выпукло, если

$$\forall x, y \in S, \alpha \in [0, 1] : \alpha x + (1 - \alpha)y \in S$$

# Выпуклые множества и функции

**Выпуклое множество:**  $S$  выпукло, если

$$\forall x, y \in S, \alpha \in [0, 1] : \alpha x + (1 - \alpha)y \in S$$

**Примеры:**

- Полупространство  $\{x : a^T x \leq b\}$

**Выпуклая функция:**  $f$  выпукла, если  $\text{dom } f$  выпуклая и

$$f(\alpha x + (1 - \alpha)y) \leq \alpha f(x) + (1 - \alpha)f(y)$$

# Выпуклые множества и функции

**Выпуклое множество:**  $S$  выпукло, если

$$\forall x, y \in S, \alpha \in [0, 1] : \alpha x + (1 - \alpha)y \in S$$

**Примеры:**

- Полупространство  $\{x : a^T x \leq b\}$
- Шар  $\{x : \|x - c\| \leq r\}$

**Выпуклая функция:**  $f$  выпукла, если  $\text{dom } f$  выпуклая и

$$f(\alpha x + (1 - \alpha)y) \leq \alpha f(x) + (1 - \alpha)f(y)$$

# Выпуклые множества и функции

**Выпуклое множество:**  $S$  выпукло, если

$$\forall x, y \in S, \alpha \in [0, 1] : \alpha x + (1 - \alpha)y \in S$$

**Примеры:**

- Полупространство  $\{x : a^T x \leq b\}$
- Шар  $\{x : \|x - c\| \leq r\}$
- Конус  $\{x : \|x\| \leq t\}$

**Выпуклая функция:**  $f$  выпукла, если  $\text{dom } f$  выпуклая и

$$f(\alpha x + (1 - \alpha)y) \leq \alpha f(x) + (1 - \alpha)f(y)$$

# Выпуклые множества и функции

**Выпуклое множество:**  $S$  выпукло, если

$$\forall x, y \in S, \alpha \in [0, 1] : \alpha x + (1 - \alpha)y \in S$$

**Примеры:**

- Полупространство  $\{x : a^T x \leq b\}$
- Шар  $\{x : \|x - c\| \leq r\}$
- Конус  $\{x : \|x\| \leq t\}$

**Выпуклая функция:**  $f$  выпукла, если  $\text{dom } f$  выпуклая и

$$f(\alpha x + (1 - \alpha)y) \leq \alpha f(x) + (1 - \alpha)f(y)$$

# Выпуклые множества и функции

**Выпуклое множество:**  $S$  выпукло, если

$$\forall x, y \in S, \alpha \in [0, 1] : \alpha x + (1 - \alpha)y \in S$$

**Примеры:**

- Полупространство  $\{x : a^T x \leq b\}$
- Шар  $\{x : \|x - c\| \leq r\}$
- Конус  $\{x : \|x\| \leq t\}$

**Выпуклая функция:**  $f$  выпукла, если  $\text{dom } f$  выпуклая и

$$f(\alpha x + (1 - \alpha)y) \leq \alpha f(x) + (1 - \alpha)f(y)$$

**Примеры:**

- $f(x) = \|x\|_p$  для  $p \geq 1$

# Выпуклые множества и функции

**Выпуклое множество:**  $S$  выпукло, если

$$\forall x, y \in S, \alpha \in [0, 1] : \alpha x + (1 - \alpha)y \in S$$

**Примеры:**

- Полупространство  $\{x : a^T x \leq b\}$
- Шар  $\{x : \|x - c\| \leq r\}$
- Конус  $\{x : \|x\| \leq t\}$

**Выпуклая функция:**  $f$  выпукла, если дом  $f$  выпуклая и

$$f(\alpha x + (1 - \alpha)y) \leq \alpha f(x) + (1 - \alpha)f(y)$$

**Примеры:**

- $f(x) = \|x\|_p$  для  $p \geq 1$
- $f(x) = e^x, f(x) = -\log x$

# Выпуклые множества и функции

**Выпуклое множество:**  $S$  выпукло, если

$$\forall x, y \in S, \alpha \in [0, 1] : \alpha x + (1 - \alpha)y \in S$$

**Примеры:**

- Полупространство  $\{x : a^T x \leq b\}$
- Шар  $\{x : \|x - c\| \leq r\}$
- Конус  $\{x : \|x\| \leq t\}$

**Выпуклая функция:**  $f$  выпукла, если дом  $f$  выпуклая и

$$f(\alpha x + (1 - \alpha)y) \leq \alpha f(x) + (1 - \alpha)f(y)$$

**Примеры:**

- $f(x) = \|x\|_p$  для  $p \geq 1$
- $f(x) = e^x$ ,  $f(x) = -\log x$
- $f(x) = x^T Ax$  при  $A \succeq 0$

## Критерии выпуклости

**Первый дифференциальный критерий** (для дифференцируемых):

$$f(y) \geq f(x) + \langle \nabla f(x), y - x \rangle \quad \forall x, y$$

## Критерии выпуклости

**Первый дифференциальный критерий** (для дифференцируемых):

$$f(y) \geq f(x) + \langle \nabla f(x), y - x \rangle \quad \forall x, y$$

**Второй дифференциальный критерий** (для дважды дифференцируемых):

$$\nabla^2 f(x) \succeq 0 \quad \forall x$$

## Критерии выпуклости

**Первый дифференциальный критерий** (для дифференцируемых):

$$f(y) \geq f(x) + \langle \nabla f(x), y - x \rangle \quad \forall x, y$$

**Второй дифференциальный критерий** (для дважды дифференцируемых):

$$\nabla^2 f(x) \succeq 0 \quad \forall x$$

**Сильная выпуклость** ( $\mu$ -strong convexity):

$$f(y) \geq f(x) + \langle \nabla f(x), y - x \rangle + \frac{\mu}{2} \|y - x\|^2$$

или  $\nabla^2 f(x) \succeq \mu I$  для всех  $x$ .

## Задача (доказательство выпуклости)

**i** Докажите, что функция LASSO выпуклая:

$$f(x) = \|Ax - b\|_2^2 + \lambda\|x\|_1, \quad \lambda > 0$$

## Задача (доказательство выпуклости)

**i** Докажите, что функция LASSO выпуклая:

$$f(x) = \|Ax - b\|_2^2 + \lambda\|x\|_1, \quad \lambda > 0$$

**Решение:**

- $f_1(x) = \|Ax - b\|_2^2$  — выпуклая, т.к.  $\nabla^2 f_1(x) = 2A^T A \succeq 0$

## Задача (доказательство выпуклости)

**i** Докажите, что функция LASSO выпуклая:

$$f(x) = \|Ax - b\|_2^2 + \lambda\|x\|_1, \quad \lambda > 0$$

**Решение:**

- $f_1(x) = \|Ax - b\|_2^2$  — выпуклая, т.к.  $\nabla^2 f_1(x) = 2A^T A \succeq 0$

## Задача (доказательство выпуклости)

**i** Докажите, что функция LASSO выпуклая:

$$f(x) = \|Ax - b\|_2^2 + \lambda\|x\|_1, \quad \lambda > 0$$

**Решение:**

- $f_1(x) = \|Ax - b\|_2^2$  — выпуклая, т.к.  $\nabla^2 f_1(x) = 2A^T A \succeq 0$
- $f_2(x) = \lambda\|x\|_1$  — выпуклая как норма, умноженная на  $\lambda > 0$

## Задача (доказательство выпуклости)

**i** Докажите, что функция LASSO выпуклая:

$$f(x) = \|Ax - b\|_2^2 + \lambda\|x\|_1, \quad \lambda > 0$$

**Решение:**

- $f_1(x) = \|Ax - b\|_2^2$  — выпуклая, т.к.  $\nabla^2 f_1(x) = 2A^T A \succeq 0$
- $f_2(x) = \lambda\|x\|_1$  — выпуклая как норма, умноженная на  $\lambda > 0$

## Задача (доказательство выпуклости)

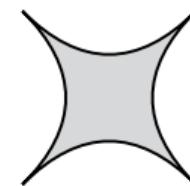
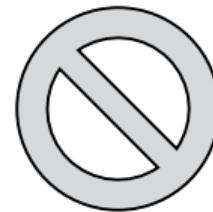
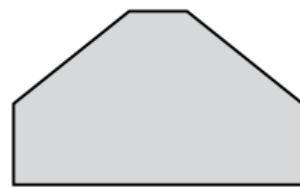
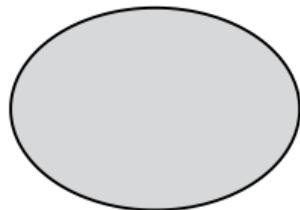
**i** Докажите, что функция LASSO выпуклая:

$$f(x) = \|Ax - b\|_2^2 + \lambda\|x\|_1, \quad \lambda > 0$$

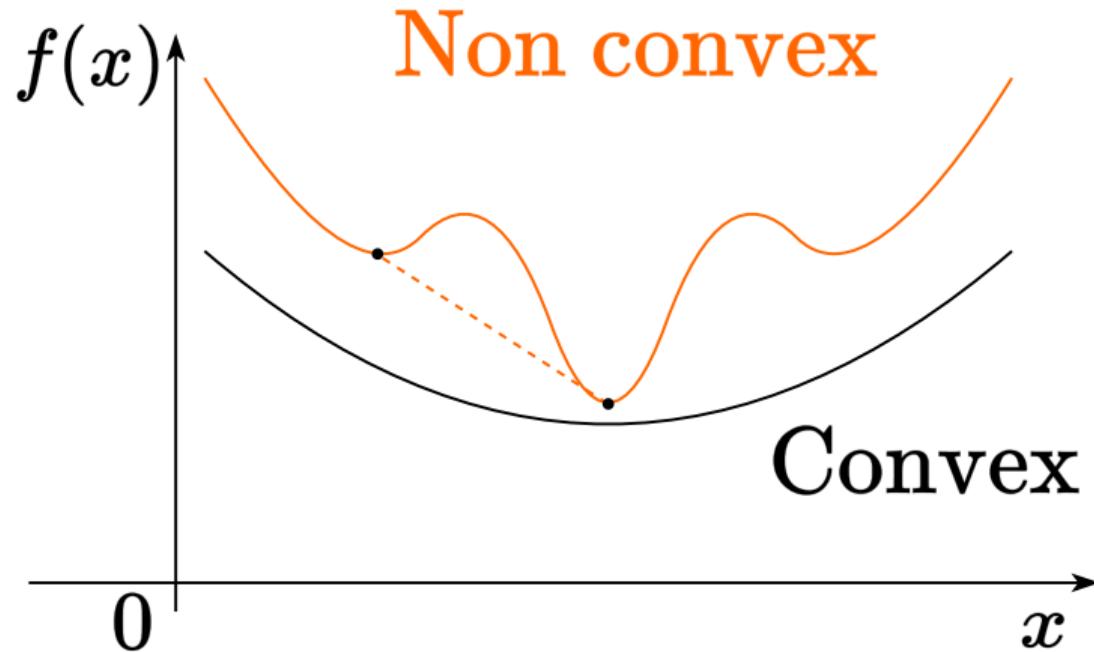
**Решение:**

- $f_1(x) = \|Ax - b\|_2^2$  — выпуклая, т.к.  $\nabla^2 f_1(x) = 2A^T A \succeq 0$
- $f_2(x) = \lambda\|x\|_1$  — выпуклая как норма, умноженная на  $\lambda > 0$
- $f(x) = f_1(x) + f_2(x)$  — выпукла как сумма выпуклых функций.

## Выпуклые и невыпуклые множества



## Выпуклая и невыпуклая функции



## Сопряжённые множества и функции (Лекция 4)

## Двойственный конус

Двойственный конус:

$$K^* = \{y : \langle x, y \rangle \geq 0 \ \forall x \in K\}$$

## Двойственный конус

Двойственный конус:

$$K^* = \{y : \langle x, y \rangle \geq 0 \ \forall x \in K\}$$

Нормальный конус:

$$N_S(x) = \{g : \langle g, y - x \rangle \leq 0 \ \forall y \in S\}$$

Примеры:

- $(K^*)^* = K$  для замкнутого выпуклого  $K$

# Двойственный конус

Двойственный конус:

$$K^* = \{y : \langle x, y \rangle \geq 0 \ \forall x \in K\}$$

Нормальный конус:

$$N_S(x) = \{g : \langle g, y - x \rangle \leq 0 \ \forall y \in S\}$$

Примеры:

- $(K^*)^* = K$  для замкнутого выпуклого  $K$
- $(\mathbb{R}_+^n)^* = \mathbb{R}_+^n$  (самодвойственный)

# Двойственный конус

Двойственный конус:

$$K^* = \{y : \langle x, y \rangle \geq 0 \ \forall x \in K\}$$

Примеры:

- $(K^*)^* = K$  для замкнутого выпуклого  $K$
- $(\mathbb{R}_+^n)^* = \mathbb{R}_+^n$  (самодвойственный)
- Конус Лоренца самодвойственный

Нормальный конус:

$$N_S(x) = \{g : \langle g, y - x \rangle \leq 0 \ \forall y \in S\}$$

# Двойственный конус

Двойственный конус:

$$K^* = \{y : \langle x, y \rangle \geq 0 \ \forall x \in K\}$$

Примеры:

- $(K^*)^* = K$  для замкнутого выпуклого  $K$
- $(\mathbb{R}_+^n)^* = \mathbb{R}_+^n$  (самодвойственный)
- Конус Лоренца самодвойственный

Нормальный конус:

$$N_S(x) = \{g : \langle g, y - x \rangle \leq 0 \ \forall y \in S\}$$

# Двойственный конус

Двойственный конус:

$$K^* = \{y : \langle x, y \rangle \geq 0 \ \forall x \in K\}$$

Примеры:

- $(K^*)^* = K$  для замкнутого выпуклого  $K$
- $(\mathbb{R}_+^n)^* = \mathbb{R}_+^n$  (самодвойственный)
- Конус Лоренца самодвойственный

Нормальный конус:

$$N_S(x) = \{g : \langle g, y - x \rangle \leq 0 \ \forall y \in S\}$$

Для выпуклого  $S$  и  $x^* \in S$ :

$x^*$  — минимум  $f$  на  $S \Leftrightarrow -\nabla f(x^*) \in N_S(x^*)$

## Сопряжённая функция (преобразование Лежандра)

$$f^*(y) = \sup_x (\langle y, x \rangle - f(x))$$

## Сопряжённая функция (преобразование Лежандра)

$$f^*(y) = \sup_x (\langle y, x \rangle - f(x))$$

**Свойства:**

- $f^*$  всегда выпуклая (даже если  $f$  не выпуклая)

## Сопряжённая функция (преобразование Лежандра)

$$f^*(y) = \sup_x (\langle y, x \rangle - f(x))$$

Свойства:

- $f^*$  всегда выпуклая (даже если  $f$  не выпуклая)
- $(f^*)^* = f$  для выпуклых замкнутых  $f$

## Сопряжённая функция (преобразование Лежандра)

$$f^*(y) = \sup_x (\langle y, x \rangle - f(x))$$

## Свойства:

- $f^*$  всегда выпуклая (даже если  $f$  не выпуклая)
  - $(f^*)^* = f$  для выпуклых замкнутых  $f$
  - Неравенство Юнга-Фенхеля:  $f(x) + f^*(y) \geq \langle x, y \rangle$

## Сопряжённая функция (преобразование Лежандра)

$$f^*(y) = \sup_x (\langle y, x \rangle - f(x))$$

#### **Свойства:**

- $f^*$  всегда выпуклая (даже если  $f$  не выпуклая)
  - $(f^*)^* = f$  для выпуклых замкнутых  $f$
  - Неравенство Юнга-Фенхеля:  $f(x) + f^*(y) \geq \langle x, y \rangle$

# Сопряжённая функция (преобразование Лежандра)

$$f^*(y) = \sup_x (\langle y, x \rangle - f(x))$$

**Свойства:**

- $f^*$  всегда выпуклая (даже если  $f$  не выпуклая)
- $(f^*)^* = f$  для выпуклых замкнутых  $f$
- Неравенство Юнга-Фенхеля:  $f(x) + f^*(y) \geq \langle x, y \rangle$

**Примеры:**

- $f(x) = \frac{1}{2}\|x\|_2^2 \Rightarrow f^*(y) = \frac{1}{2}\|y\|_2^2$

# Сопряжённая функция (преобразование Лежандра)

$$f^*(y) = \sup_x (\langle y, x \rangle - f(x))$$

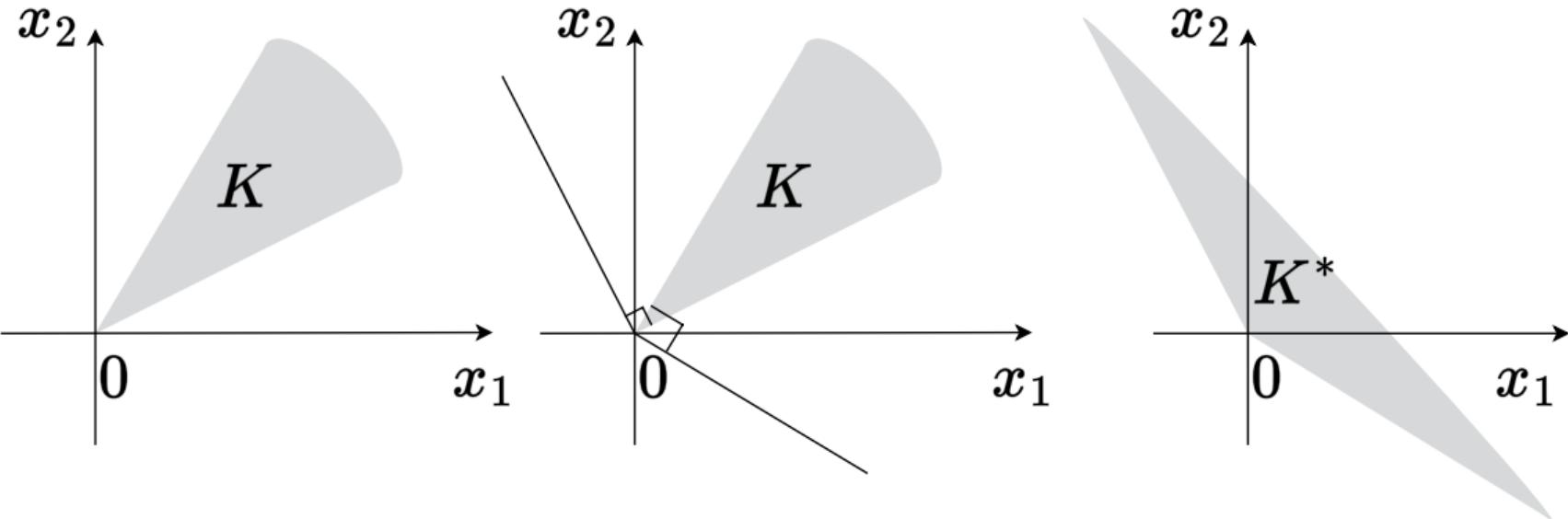
**Свойства:**

- $f^*$  всегда выпуклая (даже если  $f$  не выпуклая)
- $(f^*)^* = f$  для выпуклых замкнутых  $f$
- Неравенство Юнга-Фенхеля:  $f(x) + f^*(y) \geq \langle x, y \rangle$

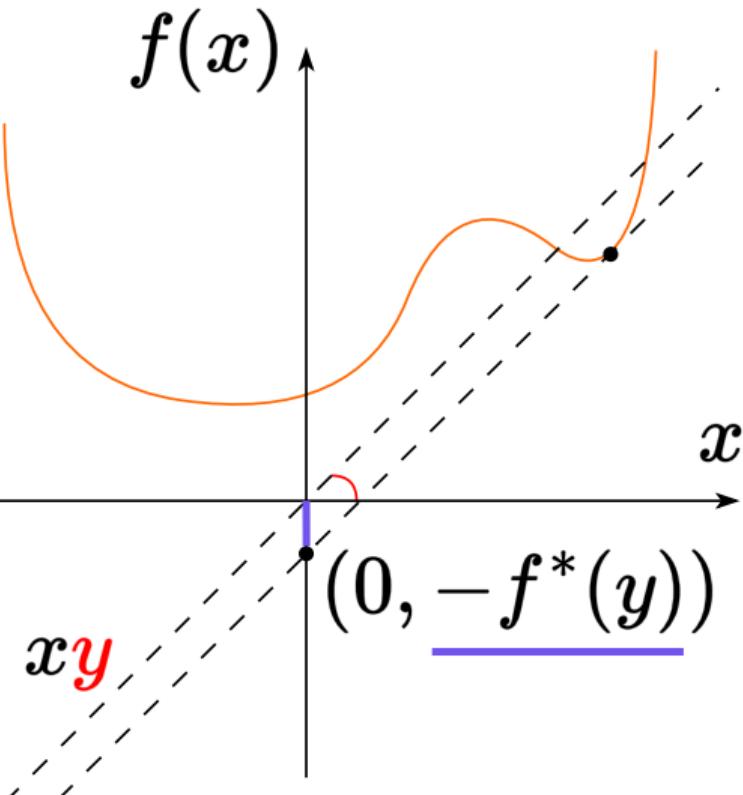
**Примеры:**

- $f(x) = \frac{1}{2}\|x\|_2^2 \Rightarrow f^*(y) = \frac{1}{2}\|y\|_2^2$
- $f(x) = \|x\|_p \Rightarrow f^*(y) = \mathbb{I}_{\|y\|_q \leq 1}$ , где  $\frac{1}{p} + \frac{1}{q} = 1$

## Двойственные конусы



## Сопряжённая функция (геометрическая интерпретация)



## Субградиент (Лекция 5)

## Субдифференциал

Для выпуклой  $f$ , **субградиент** в точке  $x$ :

$$g \in \partial f(x) \Leftrightarrow f(y) \geq f(x) + \langle g, y - x \rangle \quad \forall y$$

## Субдифференциал

Для выпуклой  $f$ , **субградиент** в точке  $x$ :

$$g \in \partial f(x) \Leftrightarrow f(y) \geq f(x) + \langle g, y - x \rangle \quad \forall y$$

Субдифференциал  $\partial f(x)$  — множество всех субградиентов.

## Субдифференциал

Для выпуклой  $f$ , **субградиент** в точке  $x$ :

$$g \in \partial f(x) \Leftrightarrow f(y) \geq f(x) + \langle g, y - x \rangle \quad \forall y$$

**Субдифференциал**  $\partial f(x)$  — множество всех субградиентов.

**Свойства:**

- Если  $f$  дифференцируема в  $x$ , то  $\partial f(x) = \{\nabla f(x)\}$

# Субдифференциал

Для выпуклой  $f$ , **субградиент** в точке  $x$ :

$$g \in \partial f(x) \Leftrightarrow f(y) \geq f(x) + \langle g, y - x \rangle \quad \forall y$$

**Субдифференциал**  $\partial f(x)$  — множество всех субградиентов.

**Свойства:**

- Если  $f$  дифференцируема в  $x$ , то  $\partial f(x) = \{\nabla f(x)\}$
- $\partial f(x)$  — выпуклое замкнутое множество

# Субдифференциал

Для выпуклой  $f$ , **субградиент** в точке  $x$ :

$$g \in \partial f(x) \Leftrightarrow f(y) \geq f(x) + \langle g, y - x \rangle \quad \forall y$$

**Субдифференциал**  $\partial f(x)$  — множество всех субградиентов.

**Свойства:**

- Если  $f$  дифференцируема в  $x$ , то  $\partial f(x) = \{\nabla f(x)\}$
- $\partial f(x)$  — выпуклое замкнутое множество
- $0 \in \partial f(x^*) \Leftrightarrow x^*$  — глобальный минимум

## Правила вычисления субдифференциала

**Сумма:**  $\partial(f + g)(x) = \partial f(x) + \partial g(x)$

## Правила вычисления субдифференциала

**Сумма:**  $\partial(f + g)(x) = \partial f(x) + \partial g(x)$

**Скалярное умножение:**  $\partial(\alpha f)(x) = \alpha \cdot \partial f(x)$  при  $\alpha > 0$

## Правила вычисления субдифференциала

**Сумма:**  $\partial(f + g)(x) = \partial f(x) + \partial g(x)$

**Скалярное умножение:**  $\partial(\alpha f)(x) = \alpha \cdot \partial f(x)$  при  $\alpha > 0$

**Цепное правило (аффинная композиция):**

$$\partial(f(Ax + b))(x) = A^T \partial f(Ax + b)$$

## Правила вычисления субдифференциала

**Сумма:**  $\partial(f + g)(x) = \partial f(x) + \partial g(x)$

**Скалярное умножение:**  $\partial(\alpha f)(x) = \alpha \cdot \partial f(x)$  при  $\alpha > 0$

**Цепное правило (аффинная композиция):**

$$\partial(f(Ax + b))(x) = A^T \partial f(Ax + b)$$

**Пример:**  $f(x) = |x|$

$$\partial f(x) = \begin{cases} \{1\}, & x > 0 \\ [-1, 1], & x = 0 \\ \{-1\}, & x < 0 \end{cases}$$

## Задача (субдифференциал)

- i** Рассмотрим функцию  $f(x) = \max(e^{x_1}, 1) + 2|x_2|$ ,  $x \in \mathbb{R}^2$ .
- 1) Найдите  $\partial f(x)$  в точке  $x = (1, -2)$ .

## Задача (субдифференциал)

**i** Рассмотрим функцию  $f(x) = \max(e^{x_1}, 1) + 2|x_2|$ ,  $x \in \mathbb{R}^2$ .

- 1) Найдите  $\partial f(x)$  в точке  $x = (1, -2)$ .
- 2) Найдите  $\partial f(x)$  в точке  $x = (0, 0)$ .

## Задача (субдифференциал)

**i** Рассмотрим функцию  $f(x) = \max(e^{x_1}, 1) + 2|x_2|$ ,  $x \in \mathbb{R}^2$ .

- 1) Найдите  $\partial f(x)$  в точке  $x = (1, -2)$ .
- 2) Найдите  $\partial f(x)$  в точке  $x = (0, 0)$ .

## Задача (субдифференциал)

**i** Рассмотрим функцию  $f(x) = \max(e^{x_1}, 1) + 2|x_2|$ ,  $x \in \mathbb{R}^2$ .

- 1) Найдите  $\partial f(x)$  в точке  $x = (1, -2)$ .
- 2) Найдите  $\partial f(x)$  в точке  $x = (0, 0)$ .

**Решение:** 1) В точке  $(1, -2)$ :  $e^1 > 1$ , поэтому  $\partial_1 f = \{e\}$ ;  $x_2 < 0$ , поэтому  $\partial_2 f = \{-2\}$ .

$$\partial f(1, -2) = \{(e, -2)^T\}$$

## Задача (субдифференциал)

**i** Рассмотрим функцию  $f(x) = \max(e^{x_1}, 1) + 2|x_2|$ ,  $x \in \mathbb{R}^2$ .

- 1) Найдите  $\partial f(x)$  в точке  $x = (1, -2)$ .
- 2) Найдите  $\partial f(x)$  в точке  $x = (0, 0)$ .

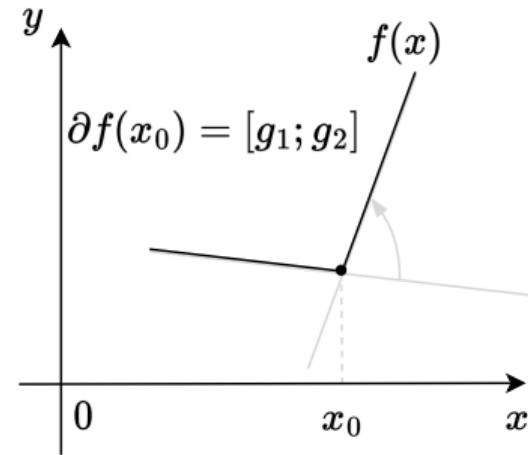
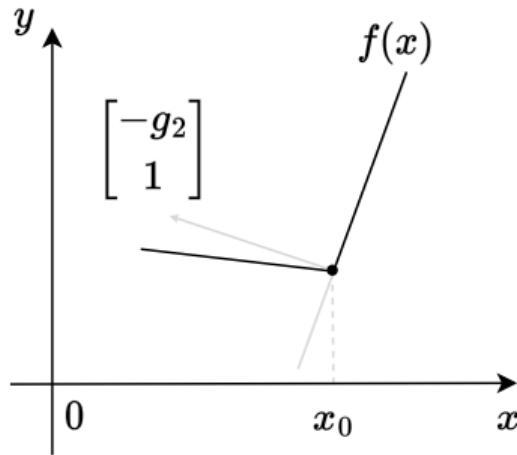
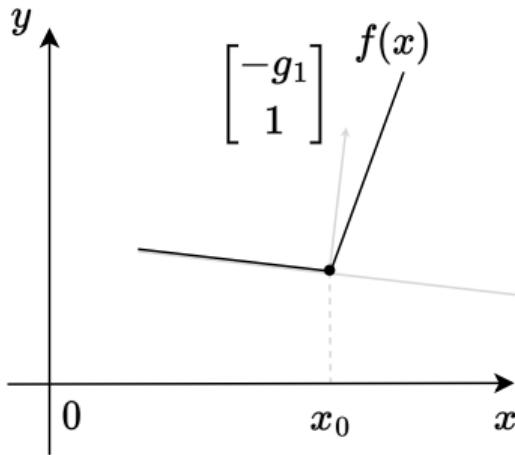
**Решение:** 1) В точке  $(1, -2)$ :  $e^1 > 1$ , поэтому  $\partial_1 f = \{e\}$ ;  $x_2 < 0$ , поэтому  $\partial_2 f = \{-2\}$ .

$$\partial f(1, -2) = \{(e, -2)^T\}$$

2) В точке  $(0, 0)$ :  $e^0 = 1$ , обе функции активны в max, поэтому  $\partial_1 f = [0, 1]$ ;  $x_2 = 0$ , поэтому  $\partial_2 f = [-2, 2]$ .

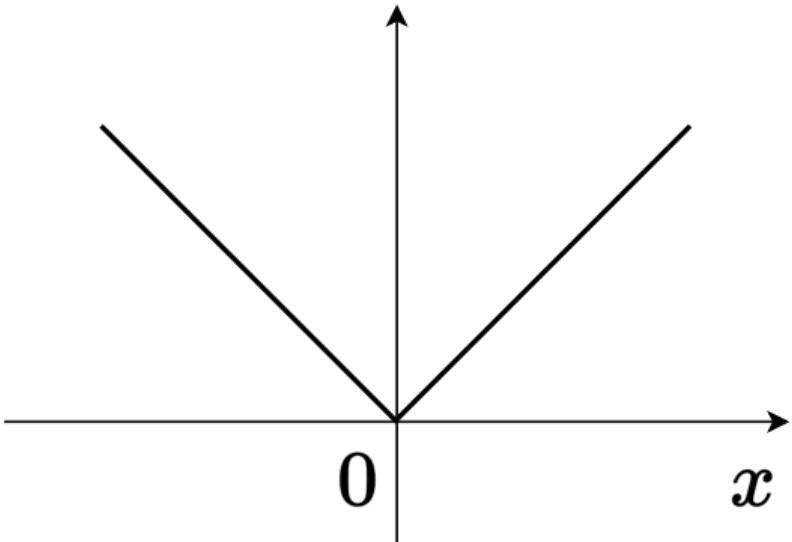
$$\partial f(0, 0) = [0, 1] \times [-2, 2]$$

## Субдифференциал — множество всех субградиентов

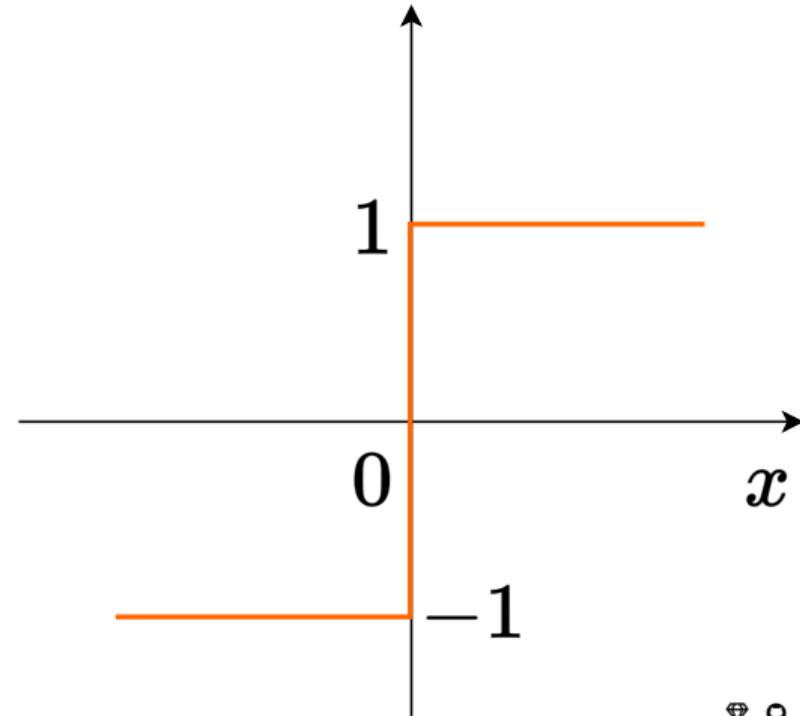


## Субдифференциал $|x|$

$$f(x) = |x|$$



$$\partial f(x)$$





## Лагранжиан и условия ККТ

Рассмотрим задачу:

$$\min_x f(x)$$

$$\text{s.t. } g_i(x) \leq 0, i = 1, \dots, m$$

$$h_j(x) = 0, j = 1, \dots, p$$

# Лагранжиан и условия ККТ

Рассмотрим задачу:

$$\begin{aligned} & \min_x f(x) \\ \text{s.t. } & g_i(x) \leq 0, \quad i = 1, \dots, m \\ & h_j(x) = 0, \quad j = 1, \dots, p \end{aligned}$$

Лагранжиан:

$$L(x, \lambda, \nu) = f(x) + \sum_{i=1}^m \lambda_i g_i(x) + \sum_{j=1}^p \nu_j h_j(x)$$

## Условия Каруша-Куна-Таккера (ККТ)

Необходимые условия оптимальности для  $x^*$ :

## Условия Каруша-Куна-Таккера (ККТ)

Необходимые условия оптимальности для  $x^*$ :

1. **Стационарность:**  $\nabla_x L(x^*, \lambda^*, \nu^*) = 0$

## Условия Каруша-Куна-Таккера (ККТ)

Необходимые условия оптимальности для  $x^*$ :

1. **Стационарность:**  $\nabla_x L(x^*, \lambda^*, \nu^*) = 0$

## Условия Каруша-Куна-Таккера (ККТ)

Необходимые условия оптимальности для  $x^*$ :

1. **Стационарность:**  $\nabla_x L(x^*, \lambda^*, \nu^*) = 0$
2. **Прямая допустимость:**  $g_i(x^*) \leq 0, h_j(x^*) = 0$

## Условия Каруша-Куна-Таккера (ККТ)

Необходимые условия оптимальности для  $x^*$ :

1. **Стационарность:**  $\nabla_x L(x^*, \lambda^*, \nu^*) = 0$
2. **Прямая допустимость:**  $g_i(x^*) \leq 0, h_j(x^*) = 0$

## Условия Каруша-Куна-Таккера (ККТ)

Необходимые условия оптимальности для  $x^*$ :

1. **Стационарность:**  $\nabla_x L(x^*, \lambda^*, \nu^*) = 0$
2. **Прямая допустимость:**  $g_i(x^*) \leq 0, h_j(x^*) = 0$
3. **Двойственная допустимость:**  $\lambda_i^* \geq 0$

## Условия Каруша-Куна-Таккера (ККТ)

Необходимые условия оптимальности для  $x^*$ :

1. **Стационарность:**  $\nabla_x L(x^*, \lambda^*, \nu^*) = 0$
2. **Прямая допустимость:**  $g_i(x^*) \leq 0, h_j(x^*) = 0$
3. **Двойственная допустимость:**  $\lambda_i^* \geq 0$

## Условия Каруша-Куна-Таккера (ККТ)

Необходимые условия оптимальности для  $x^*$ :

1. **Стационарность:**  $\nabla_x L(x^*, \lambda^*, \nu^*) = 0$
2. **Прямая допустимость:**  $g_i(x^*) \leq 0, h_j(x^*) = 0$
3. **Двойственная допустимость:**  $\lambda_i^* \geq 0$
4. **Дополняющая нежёсткость:**  $\lambda_i^* g_i(x^*) = 0$

## Условия Каруша-Куна-Таккера (ККТ)

Необходимые условия оптимальности для  $x^*$ :

1. **Стационарность:**  $\nabla_x L(x^*, \lambda^*, \nu^*) = 0$
2. **Прямая допустимость:**  $g_i(x^*) \leq 0, h_j(x^*) = 0$
3. **Двойственная допустимость:**  $\lambda_i^* \geq 0$
4. **Дополняющая нежёсткость:**  $\lambda_i^* g_i(x^*) = 0$

## Условия Каруша-Куна-Таккера (ККТ)

Необходимые условия оптимальности для  $x^*$ :

1. **Стационарность:**  $\nabla_x L(x^*, \lambda^*, \nu^*) = 0$
2. **Прямая допустимость:**  $g_i(x^*) \leq 0, h_j(x^*) = 0$
3. **Двойственная допустимость:**  $\lambda_i^* \geq 0$
4. **Дополняющая нежёсткость:**  $\lambda_i^* g_i(x^*) = 0$

Для выпуклых задач ККТ условия **достаточны** при выполнении условия регулярности (например, условие Слейтера).

## Задача (условия ККТ)

**i** Найдите минимум функции:

$$f(x, y) = \frac{1}{2}x^T Ax + b^T x + \frac{1}{2}\|y\|_2^2 \rightarrow \min_{x+y=c}$$

где  $A \succ 0$ .

## Задача (условия ККТ)

**i** Найдите минимум функции:

$$f(x, y) = \frac{1}{2}x^T Ax + b^T x + \frac{1}{2}\|y\|_2^2 \rightarrow \min_{x+y=c}$$

где  $A \succ 0$ .

**Решение:** Лагранжиан:  $L = \frac{1}{2}x^T Ax + b^T x + \frac{1}{2}y^T y + \nu^T(x + y - c)$

## Задача (условия ККТ)

**i** Найдите минимум функции:

$$f(x, y) = \frac{1}{2}x^T Ax + b^T x + \frac{1}{2}\|y\|_2^2 \rightarrow \min_{x+y=c}$$

где  $A \succ 0$ .

**Решение:** Лагранжиан:  $L = \frac{1}{2}x^T Ax + b^T x + \frac{1}{2}y^T y + \nu^T(x + y - c)$

Условия оптимальности:  $Ax + b + \nu = 0, y + \nu = 0, x + y = c$

## Задача (условия ККТ)

**i** Найдите минимум функции:

$$f(x, y) = \frac{1}{2}x^T Ax + b^T x + \frac{1}{2}\|y\|_2^2 \rightarrow \min_{x+y=c}$$

где  $A \succ 0$ .

**Решение:** Лагранжиан:  $L = \frac{1}{2}x^T Ax + b^T x + \frac{1}{2}y^T y + \nu^T(x + y - c)$

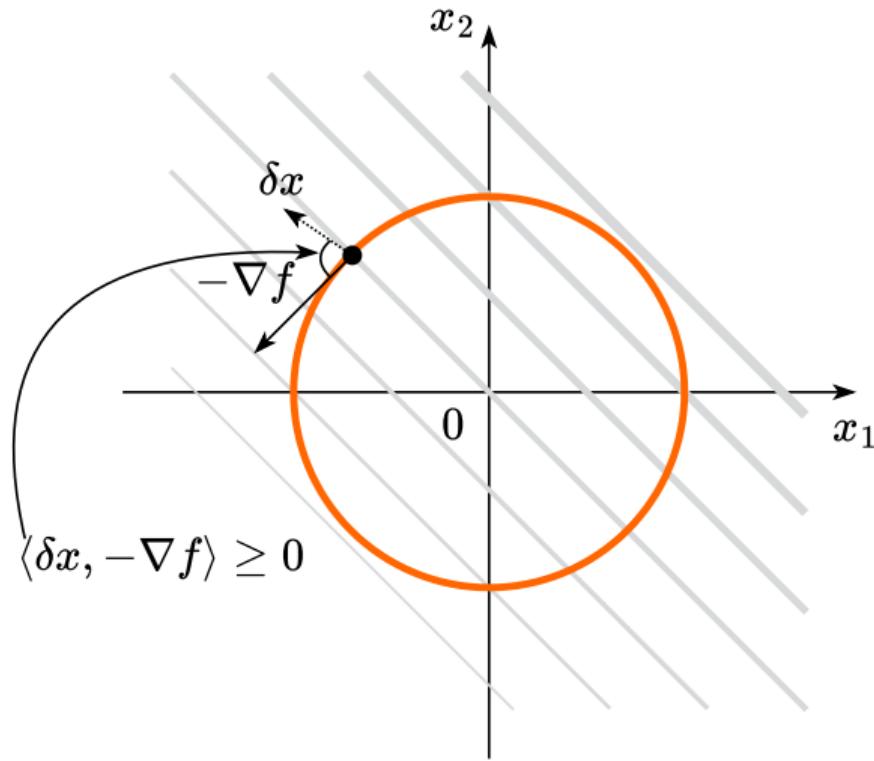
Условия оптимальности:  $Ax + b + \nu = 0, y + \nu = 0, x + y = c$

Из  $y = -\nu$  и  $x = -A^{-1}(b + \nu)$ , подставляя в ограничение:

$$\nu = -(A^{-1} + I)^{-1}(c + A^{-1}b)$$

## ККТ для задачи с ограничением-равенством

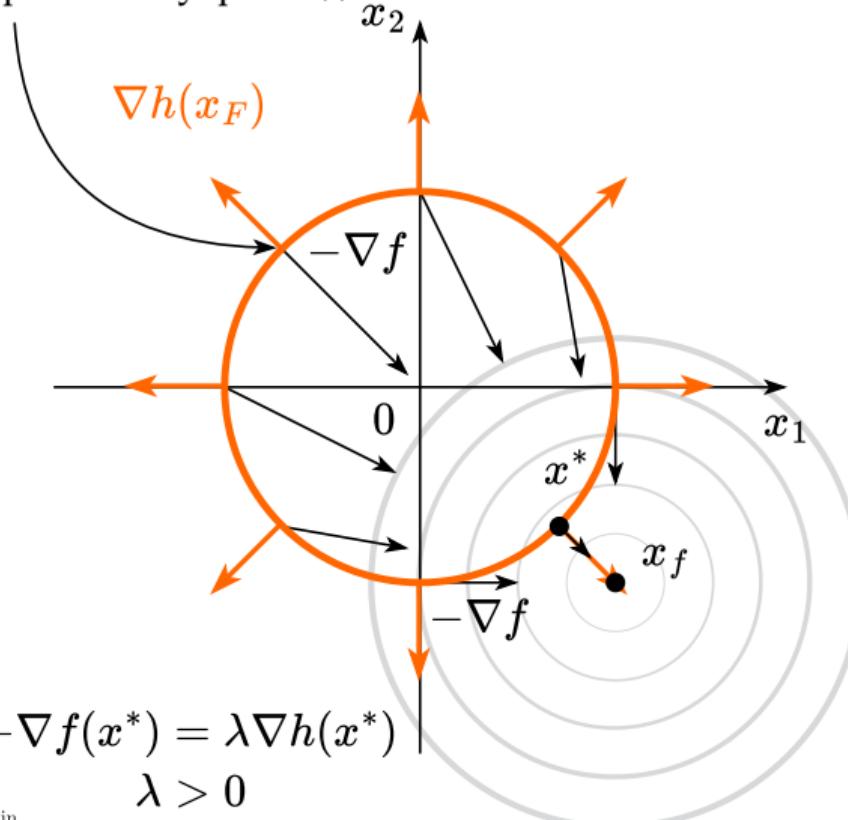
Мы хотим:  $f(x_F + \delta x) \leq f(x_F)$



## ККТ для задачи с ограничением-неравенством

Не является локальным минимумом, т.к.  $-\nabla f(x)$

направлен внутрь бюджетного множества



## Двойственность (Лекция 7)

## Двойственная задача Лагранжа

Двойственная функция:

$$g(\lambda, \nu) = \inf_x L(x, \lambda, \nu)$$

## Двойственная задача Лагранжа

**Двойственная функция:**

$$g(\lambda, \nu) = \inf_x L(x, \lambda, \nu)$$

**Двойственная задача:**

$$\max_{\lambda, \nu} g(\lambda, \nu) \quad \text{s.t.} \quad \lambda \geq 0$$

## Двойственная задача Лагранжа

Двойственная функция:

$$g(\lambda, \nu) = \inf_x L(x, \lambda, \nu)$$

Двойственная задача:

$$\max_{\lambda, \nu} g(\lambda, \nu) \quad \text{s.t.} \quad \lambda \geq 0$$

Слабая двойственность:  $d^* \leq p^*$  (всегда)

## Двойственная задача Лагранжа

Двойственная функция:

$$g(\lambda, \nu) = \inf_x L(x, \lambda, \nu)$$

Двойственная задача:

$$\max_{\lambda, \nu} g(\lambda, \nu) \quad \text{s.t.} \quad \lambda \geq 0$$

Слабая двойственность:  $d^* \leq p^*$  (всегда)

Сильная двойственность:  $d^* = p^*$  (при условиях регулярности)

## Условие Слейтера

**Условие Слейтера:** Существует  $\tilde{x}$  такое, что:

- $g_i(\tilde{x}) < 0$  для всех  $i$  (строгое неравенство)

## Условие Слейтера

**Условие Слейтера:** Существует  $\tilde{x}$  такое, что:

- $g_i(\tilde{x}) < 0$  для всех  $i$  (строгое неравенство)
- $h_j(\tilde{x}) = 0$  для всех  $j$

## Условие Слейтера

**Условие Слейтера:** Существует  $\tilde{x}$  такое, что:

- $g_i(\tilde{x}) < 0$  для всех  $i$  (строгое неравенство)
- $h_j(\tilde{x}) = 0$  для всех  $j$

## Условие Слейтера

**Условие Слейтера:** Существует  $\tilde{x}$  такое, что:

- $g_i(\tilde{x}) < 0$  для всех  $i$  (строгое неравенство)
- $h_j(\tilde{x}) = 0$  для всех  $j$

**Теорема:** Для выпуклой задачи при выполнении условия Слейтера выполняется сильная двойственность.

## Условие Слейтера

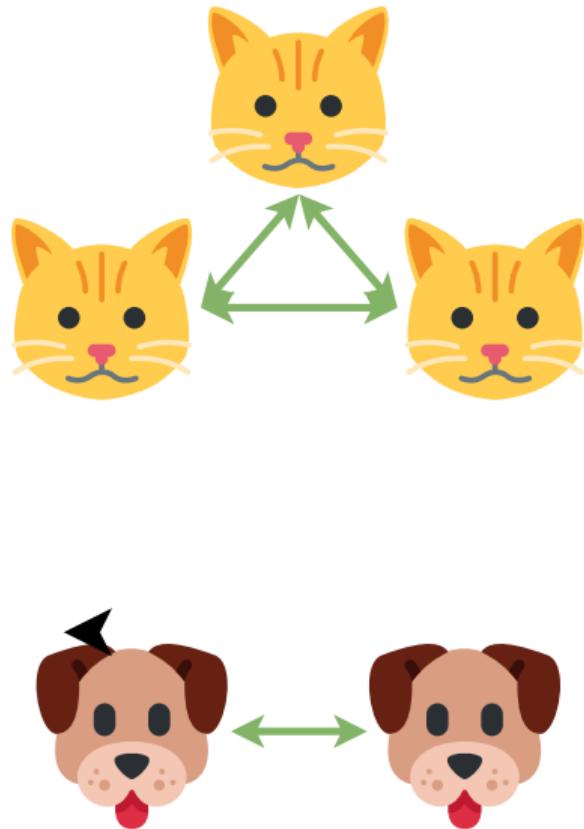
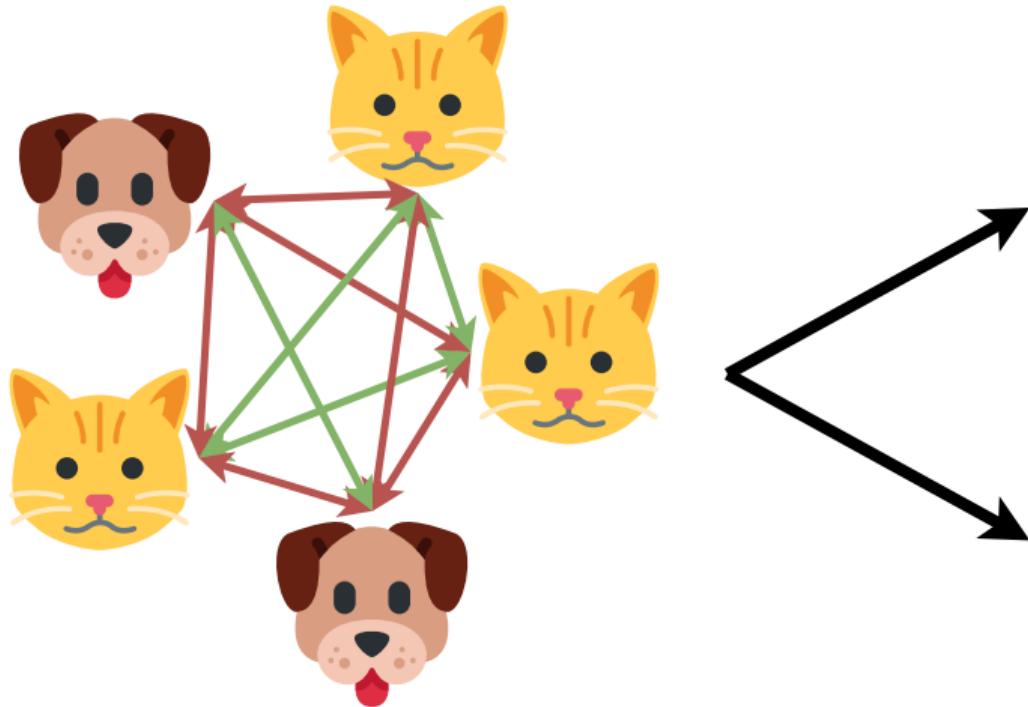
**Условие Слейтера:** Существует  $\tilde{x}$  такое, что:

- $g_i(\tilde{x}) < 0$  для всех  $i$  (строгое неравенство)
- $h_j(\tilde{x}) = 0$  для всех  $j$

**Теорема:** Для выпуклой задачи при выполнении условия Слейтера выполняется сильная двойственность.

**Важно:** Двойственная функция  $g(\lambda, \nu)$  всегда вогнутая, двойственная задача — всегда выпуклая.

## Двухстороннее разбиение: сильная двойственность для невыпуклой задачи



## Линейное программирование (Лекция 8)

## Задача LP

**Стандартная форма:**

$$c^T x \rightarrow \min \quad \text{s.t.} \quad Ax = b, \quad x \geq 0$$

## Задача LP

**Стандартная форма:**

$$c^T x \rightarrow \min \quad \text{s.t.} \quad Ax = b, \quad x \geq 0$$

**Каноническая форма:**

$$c^T x \rightarrow \min \quad \text{s.t.} \quad Ax \leq b$$

## Задача LP

**Стандартная форма:**

$$c^T x \rightarrow \min \quad \text{s.t.} \quad Ax = b, \quad x \geq 0$$

**Каноническая форма:**

$$c^T x \rightarrow \min \quad \text{s.t.} \quad Ax \leq b$$

**Свойства:**

- Допустимое множество — полиэдр (выпуклый многогранник)

## Задача LP

**Стандартная форма:**

$$c^T x \rightarrow \min \quad \text{s.t.} \quad Ax = b, \quad x \geq 0$$

**Каноническая форма:**

$$c^T x \rightarrow \min \quad \text{s.t.} \quad Ax \leq b$$

**Свойства:**

- Допустимое множество — полиэдр (выпуклый многогранник)
- Оптимум достигается в вершине полиэдра

## Задача LP

**Стандартная форма:**

$$c^T x \rightarrow \min \quad \text{s.t.} \quad Ax = b, \quad x \geq 0$$

**Каноническая форма:**

$$c^T x \rightarrow \min \quad \text{s.t.} \quad Ax \leq b$$

**Свойства:**

- Допустимое множество — полиэдр (выпуклый многогранник)
- Оптимум достигается в вершине полиэдра
- Решение: симплекс-метод или методы внутренней точки

## Симплекс-метод (идея)

1. Начинаем с базисного допустимого решения (вершины)

## Симплекс-метод (идея)

1. Начинаем с базисного допустимого решения (вершины)
2. Находим улучшающее направление вдоль ребра

## Симплекс-метод (идея)

1. Начинаем с базисного допустимого решения (вершины)
2. Находим улучшающее направление вдоль ребра
3. Переходим к соседней вершине

## Симплекс-метод (идея)

1. Начинаем с базисного допустимого решения (вершины)
2. Находим улучшающее направление вдоль ребра
3. Переходим к соседней вершине
4. Повторяем до достижения оптимума

## Симплекс-метод (идея)

1. Начинаем с базисного допустимого решения (вершины)
2. Находим улучшающее направление вдоль ребра
3. Переходим к соседней вершине
4. Повторяем до достижения оптимума

## Симплекс-метод (идея)

1. Начинаем с базисного допустимого решения (вершины)
2. Находим улучшающее направление вдоль ребра
3. Переходим к соседней вершине
4. Повторяем до достижения оптимума

**Сложность:**

- В худшем случае экспоненциальная

## Симплекс-метод (идея)

1. Начинаем с базисного допустимого решения (вершины)
2. Находим улучшающее направление вдоль ребра
3. Переходим к соседней вершине
4. Повторяем до достижения оптимума

**Сложность:**

- В худшем случае экспоненциальная
- На практике обычно  $O(m)$  итераций

## Симплекс-метод (идея)

1. Начинаем с базисного допустимого решения (вершины)
2. Находим улучшающее направление вдоль ребра
3. Переходим к соседней вершине
4. Повторяем до достижения оптимума

**Сложность:**

- В худшем случае экспоненциальная
- На практике обычно  $O(m)$  итераций
- LP разрешима за полиномиальное время (методы внутренней точки)



# Прямая и двойственная задачи LP

Прямая задача:

$$c^T x \rightarrow \min \quad \text{s.t.} \quad Ax \leq b$$

# Прямая и двойственная задачи LP

**Прямая задача:**

$$c^T x \rightarrow \min \quad \text{s.t.} \quad Ax \leq b$$

**Двойственная задача:**

$$b^T y \rightarrow \max \quad \text{s.t.} \quad A^T y = c, \quad y \geq 0$$

# Прямая и двойственная задачи LP

**Прямая задача:**

$$c^T x \rightarrow \min \quad \text{s.t.} \quad Ax \leq b$$

**Двойственная задача:**

$$b^T y \rightarrow \max \quad \text{s.t.} \quad A^T y = c, \quad y \geq 0$$

**Сильная двойственность для LP:** Если прямая задача имеет решение, то двойственная тоже, и  $c^T x^* = b^T y^*$ .

## Теорема о максимальном потоке и минимальном разрезе

**Max-flow min-cut:** В задаче о максимальном s-t потоке в сети:

Максимальный s-t поток = Минимальная пропускная способность s-t разреза

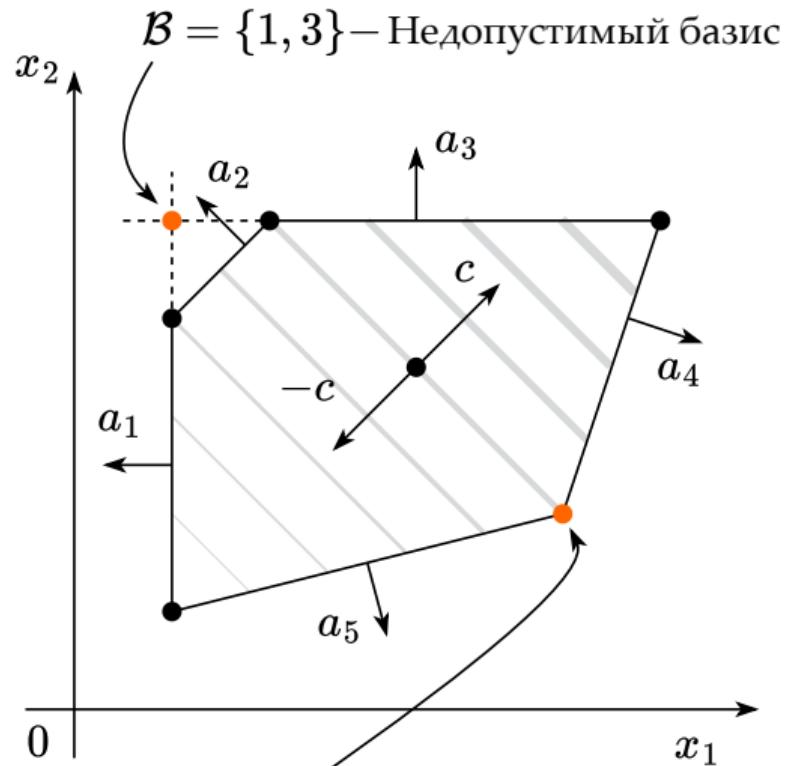
## Теорема о максимальном потоке и минимальном разрезе

**Max-flow min-cut:** В задаче о максимальном s-t потоке в сети:

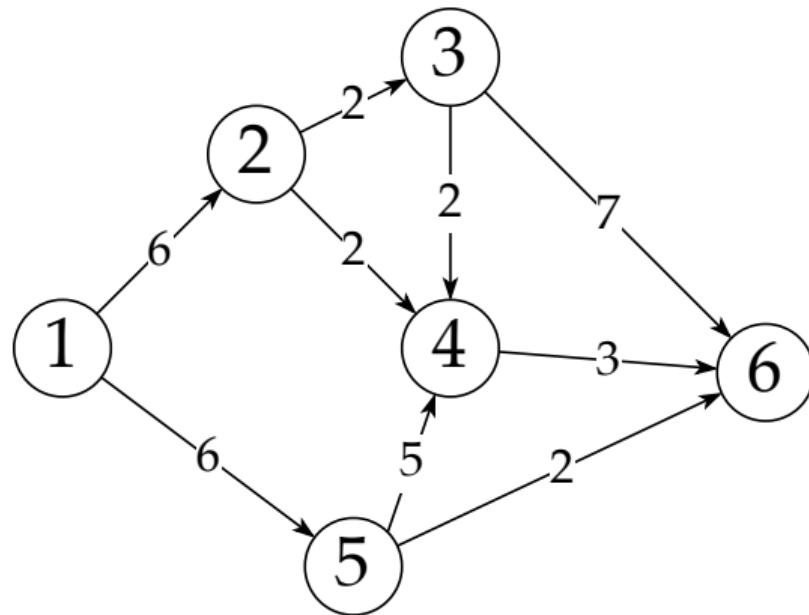
Максимальный s-t поток = Минимальная пропускная способность s-t разреза

Это следствие сильной двойственности LP!

## Геометрия линейного программирования



## Максимальный поток в сети





## Классификация скоростей сходимости

Последовательность  $\{x_k\}$  сходится к  $x^*$  со скоростью:

## Классификация скоростей сходимости

Последовательность  $\{x_k\}$  сходится к  $x^*$  со скоростью:

**Сублинейная:**  $\|x_k - x^*\| \leq O(1/k^\alpha)$ ,  $\alpha > 0$

## Классификация скоростей сходимости

Последовательность  $\{x_k\}$  сходится к  $x^*$  со скоростью:

**Сублинейная:**  $\|x_k - x^*\| \leq O(1/k^\alpha)$ ,  $\alpha > 0$

**Линейная (геометрическая):**  $\|x_{k+1} - x^*\| \leq q \|x_k - x^*\|$ ,  $q \in (0, 1)$

## Классификация скоростей сходимости

Последовательность  $\{x_k\}$  сходится к  $x^*$  со скоростью:

**Сублинейная:**  $\|x_k - x^*\| \leq O(1/k^\alpha)$ ,  $\alpha > 0$

**Линейная (геометрическая):**  $\|x_{k+1} - x^*\| \leq q \|x_k - x^*\|$ ,  $q \in (0, 1)$

**Суперлинейная:**  $\lim_{k \rightarrow \infty} \frac{\|x_{k+1} - x^*\|}{\|x_k - x^*\|} = 0$

# Классификация скоростей сходимости

Последовательность  $\{x_k\}$  сходится к  $x^*$  со скоростью:

**Сублинейная:**  $\|x_k - x^*\| \leq O(1/k^\alpha)$ ,  $\alpha > 0$

**Линейная (геометрическая):**  $\|x_{k+1} - x^*\| \leq q\|x_k - x^*\|$ ,  $q \in (0, 1)$

**Суперлинейная:**  $\lim_{k \rightarrow \infty} \frac{\|x_{k+1} - x^*\|}{\|x_k - x^*\|} = 0$

**Квадратичная:**  $\|x_{k+1} - x^*\| \leq C\|x_k - x^*\|^2$

# Одномерный поиск

Методы одномерной оптимизации:

- Дихотомия (бисекция)

# Одномерный поиск

Методы одномерной оптимизации:

- Дихотомия (бисекция)
- Метод золотого сечения

# Одномерный поиск

Методы одномерной оптимизации:

- Дихотомия (бисекция)
- Метод золотого сечения
- Метод парабол

# Одномерный поиск

Методы одномерной оптимизации:

- Дихотомия (бисекция)
- Метод золотого сечения
- Метод парабол

# Одномерный поиск

Методы одномерной оптимизации:

- Дихотомия (бисекция)
- Метод золотого сечения
- Метод парабол

Условия Вольфе (для выбора шага):

1. **Armijo:**  $f(x_k + \alpha d_k) \leq f(x_k) + c_1 \alpha \langle \nabla f(x_k), d_k \rangle$

где  $0 < c_1 < c_2 < 1$ .

# Одномерный поиск

Методы одномерной оптимизации:

- Дихотомия (бисекция)
- Метод золотого сечения
- Метод парабол

Условия Вольфе (для выбора шага):

1. **Armijo:**  $f(x_k + \alpha d_k) \leq f(x_k) + c_1 \alpha \langle \nabla f(x_k), d_k \rangle$
2. **Кривизна:**  $\langle \nabla f(x_k + \alpha d_k), d_k \rangle \geq c_2 \langle \nabla f(x_k), d_k \rangle$

где  $0 < c_1 < c_2 < 1$ .

## Задача (скорости сходимости)

- i** Напишите асимптотическую верхнюю оценку числа итераций для достижения  $f(x) - f^* \leq \varepsilon$ :
- 1) Субградиентный метод,  $f$  выпукла,  $\|\partial f(x)\|_2 \leq G$

## Задача (скорости сходимости)

- i** Напишите асимптотическую верхнюю оценку числа итераций для достижения  $f(x) - f^* \leq \varepsilon$ :
- 1) Субградиентный метод,  $f$  выпукла,  $\|\partial f(x)\|_2 \leq G$
  - 2) Градиентный спуск,  $f$   $L$ -гладкая и  $\mu$ -сильно выпукла

## Задача (скорости сходимости)

- i** Напишите асимптотическую верхнюю оценку числа итераций для достижения  $f(x) - f^* \leq \varepsilon$ :
- 1) Субградиентный метод,  $f$  выпукла,  $\|\partial f(x)\|_2 \leq G$
  - 2) Градиентный спуск,  $f$   $L$ -гладкая и  $\mu$ -сильно выпукла
  - 3) Heavy Ball для  $\mu$ -сильно выпуклой квадратичной функции

## Задача (скорости сходимости)

- i** Напишите асимптотическую верхнюю оценку числа итераций для достижения  $f(x) - f^* \leq \varepsilon$ :
- 1) Субградиентный метод,  $f$  выпукла,  $\|\partial f(x)\|_2 \leq G$
  - 2) Градиентный спуск,  $f$   $L$ -гладкая и  $\mu$ -сильно выпукла
  - 3) Heavy Ball для  $\mu$ -сильно выпуклой квадратичной функции
  - 4) Метод дихотомии для унимодальной функции

## Задача (скорости сходимости)

- i** Напишите асимптотическую верхнюю оценку числа итераций для достижения  $f(x) - f^* \leq \varepsilon$ :
- 1) Субградиентный метод,  $f$  выпукла,  $\|\partial f(x)\|_2 \leq G$
  - 2) Градиентный спуск,  $f$   $L$ -гладкая и  $\mu$ -сильно выпукла
  - 3) Heavy Ball для  $\mu$ -сильно выпуклой квадратичной функции
  - 4) Метод дихотомии для унимодальной функции

## Задача (скорости сходимости)

- i** Напишите асимптотическую верхнюю оценку числа итераций для достижения  $f(x) - f^* \leq \varepsilon$ :
- 1) Субградиентный метод,  $f$  выпукла,  $\|\partial f(x)\|_2 \leq G$
  - 2) Градиентный спуск,  $f$   $L$ -гладкая и  $\mu$ -сильно выпукла
  - 3) Heavy Ball для  $\mu$ -сильно выпуклой квадратичной функции
  - 4) Метод дихотомии для унимодальной функции

**Решение:**

- 1)  $N = \mathcal{O}(1/\varepsilon^2)$  — сублинейная

## Задача (скорости сходимости)

- i** Напишите асимптотическую верхнюю оценку числа итераций для достижения  $f(x) - f^* \leq \varepsilon$ :
- 1) Субградиентный метод,  $f$  выпукла,  $\|\partial f(x)\|_2 \leq G$
  - 2) Градиентный спуск,  $f$   $L$ -гладкая и  $\mu$ -сильно выпукла
  - 3) Heavy Ball для  $\mu$ -сильно выпуклой квадратичной функции
  - 4) Метод дихотомии для унимодальной функции

**Решение:**

- 1)  $N = \mathcal{O}(1/\varepsilon^2)$  — сублинейная
- 2)  $N = \mathcal{O}(\kappa \log(1/\varepsilon))$  — линейная

## Задача (скорости сходимости)

- i** Напишите асимптотическую верхнюю оценку числа итераций для достижения  $f(x) - f^* \leq \varepsilon$ :
- 1) Субградиентный метод,  $f$  выпукла,  $\|\partial f(x)\|_2 \leq G$
  - 2) Градиентный спуск,  $f$   $L$ -гладкая и  $\mu$ -сильно выпукла
  - 3) Heavy Ball для  $\mu$ -сильно выпуклой квадратичной функции
  - 4) Метод дихотомии для унимодальной функции

**Решение:**

- 1)  $N = \mathcal{O}(1/\varepsilon^2)$  — сублинейная
- 2)  $N = \mathcal{O}(\kappa \log(1/\varepsilon))$  — линейная
- 3)  $N = \mathcal{O}(\sqrt{\kappa} \log(1/\varepsilon))$  — ускоренная линейная

## Задача (скорости сходимости)

**i**

Напишите асимптотическую верхнюю оценку числа итераций для достижения  $f(x) - f^* \leq \varepsilon$ :

- 1) Субградиентный метод,  $f$  выпукла,  $\|\partial f(x)\|_2 \leq G$
- 2) Градиентный спуск,  $f$   $L$ -гладкая и  $\mu$ -сильно выпукла
- 3) Heavy Ball для  $\mu$ -сильно выпуклой квадратичной функции
- 4) Метод дихотомии для унимодальной функции

**Решение:**

- 1)  $N = \mathcal{O}(1/\varepsilon^2)$  — сублинейная
- 2)  $N = \mathcal{O}(\kappa \log(1/\varepsilon))$  — линейная
- 3)  $N = \mathcal{O}(\sqrt{\kappa} \log(1/\varepsilon))$  — ускоренная линейная
- 4)  $N = \mathcal{O}(\log(1/\varepsilon))$  — логарифмическая



## Метод градиентного спуска

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

## Метод градиентного спуска

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

**Теорема (сходимость для L-гладких выпуклых функций):**

При  $\alpha = \frac{1}{L}$ :

$$f(x_k) - f^* \leq \frac{L\|x_0 - x^*\|^2}{2k}$$

## Метод градиентного спуска

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

**Теорема (сходимость для L-гладких выпуклых функций):**

При  $\alpha = \frac{1}{L}$ :

$$f(x_k) - f^* \leq \frac{L\|x_0 - x^*\|^2}{2k}$$

**Теорема (сходимость для сильно выпуклых функций):**

При  $\alpha = \frac{1}{L}$  для  $\mu$ -сильно выпуклых:

$$\|x_k - x^*\|^2 \leq \left(1 - \frac{\mu}{L}\right)^k \|x_0 - x^*\|^2$$

## Задача (градиентный спуск для Ridge регрессии)

**i** Выпишите итерацию градиентного спуска и оценку скорости сходимости для:

$$\min_{x \in \mathbb{R}^n} \|Ax - b\|_2^2 + \lambda \|x\|_2^2$$

## Задача (градиентный спуск для Ridge регрессии)

**i** Выпишите итерацию градиентного спуска и оценку скорости сходимости для:

$$\min_{x \in \mathbb{R}^n} \|Ax - b\|_2^2 + \lambda \|x\|_2^2$$

**Решение:**  $\nabla f(x) = 2(A^T A + \lambda I)x - 2A^T b$

## Задача (градиентный спуск для Ridge регрессии)

**i** Выпишите итерацию градиентного спуска и оценку скорости сходимости для:

$$\min_{x \in \mathbb{R}^n} \|Ax - b\|_2^2 + \lambda \|x\|_2^2$$

**Решение:**  $\nabla f(x) = 2(A^T A + \lambda I)x - 2A^T b$

$$x_{k+1} = x_k - \alpha[(A^T A + \lambda I)x_k - A^T b]$$

## Задача (градиентный спуск для Ridge регрессии)

**i** Выпишите итерацию градиентного спуска и оценку скорости сходимости для:

$$\min_{x \in \mathbb{R}^n} \|Ax - b\|_2^2 + \lambda \|x\|_2^2$$

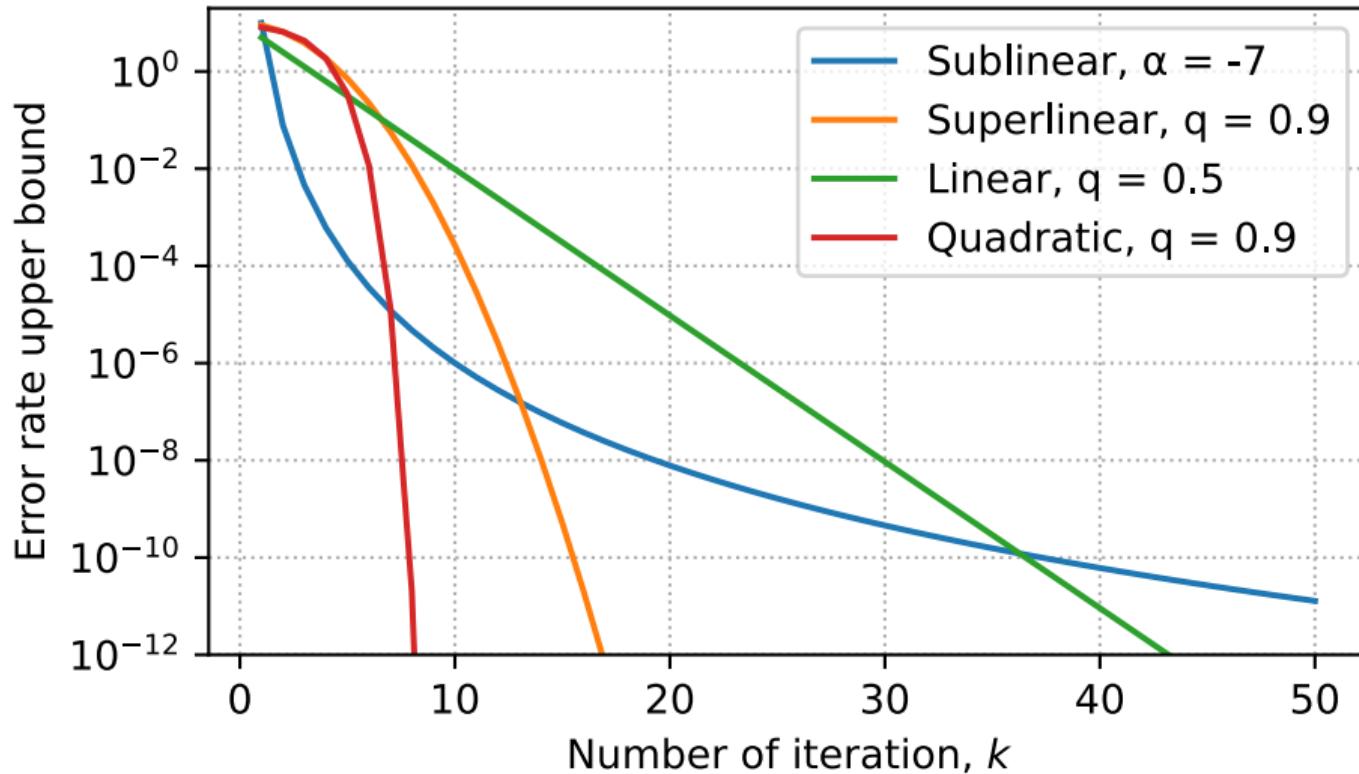
**Решение:**  $\nabla f(x) = 2(A^T A + \lambda I)x - 2A^T b$

$$x_{k+1} = x_k - \alpha[(A^T A + \lambda I)x_k - A^T b]$$

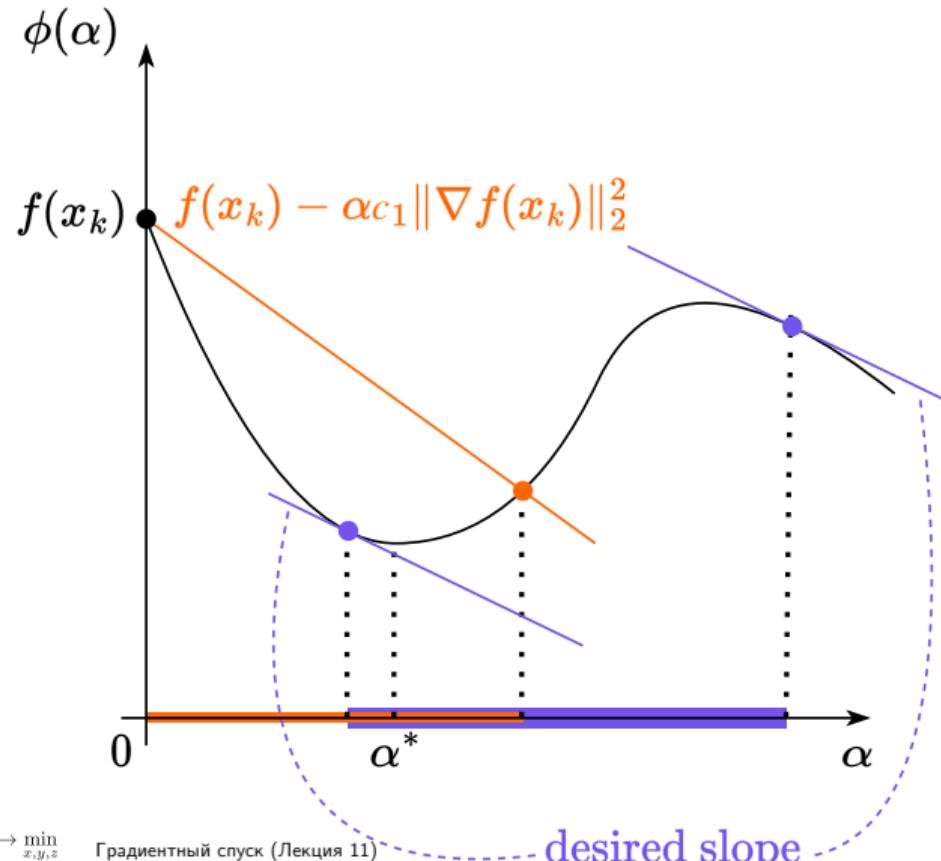
Число обусловленности:  $\kappa = \frac{\lambda_{\max}(A^T A) + \lambda}{\lambda_{\min}(A^T A) + \lambda}$

Скорость сходимости:  $\|x_k - x^*\| \leq \left(\frac{\kappa-1}{\kappa+1}\right)^k \|x_0 - x^*\|$

## Типы скоростей сходимости

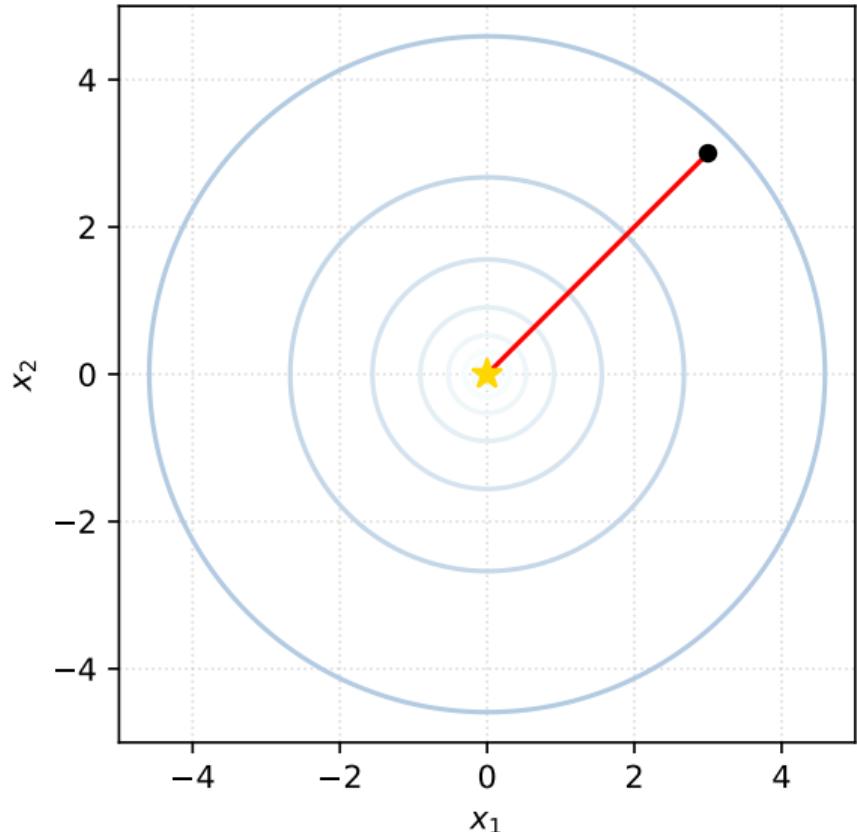


## Условия Вольфе

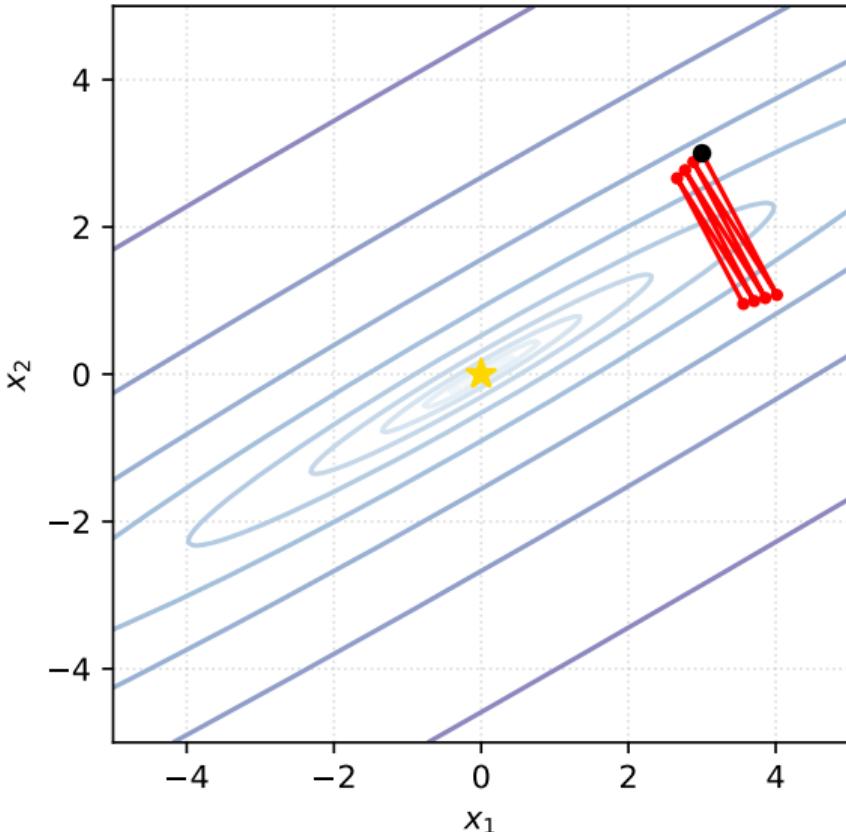


## Влияние числа обусловленности на градиентный спуск

$$\kappa = 1.0$$



$$\kappa = 100.0$$



## Нижние оценки и ускорение (Лекция 12)

## Нижние оценки для методов первого порядка

**Теорема (Нестеров):** Для любого метода первого порядка существует  $L$ -гладкая  $\mu$ -сильно выпуклая функция такая, что:

$$f(x_k) - f^* \geq \frac{\mu}{2} \left( \frac{\sqrt{\kappa} - 1}{\sqrt{\kappa} + 1} \right)^{2k} \|x_0 - x^*\|^2$$

## Нижние оценки для методов первого порядка

**Теорема (Нестеров):** Для любого метода первого порядка существует  $L$ -гладкая  $\mu$ -сильно выпуклая функция такая, что:

$$f(x_k) - f^* \geq \frac{\mu}{2} \left( \frac{\sqrt{\kappa} - 1}{\sqrt{\kappa} + 1} \right)^{2k} \|x_0 - x^*\|^2$$

Сравните с GD:  $\left(1 - \frac{1}{\kappa}\right)^k$  vs оптимальное  $\left(1 - \frac{2}{\sqrt{\kappa}}\right)^k$

## Ускоренный градиентный метод Нестерова

$$\begin{aligned}y_k &= x_k + \frac{k-1}{k+2}(x_k - x_{k-1}) \\x_{k+1} &= y_k - \frac{1}{L}\nabla f(y_k)\end{aligned}$$

## Ускоренный градиентный метод Нестерова

$$\begin{aligned}y_k &= x_k + \frac{k-1}{k+2}(x_k - x_{k-1}) \\x_{k+1} &= y_k - \frac{1}{L}\nabla f(y_k)\end{aligned}$$

**Теорема:** Для  $L$ -гладких выпуклых функций:

$$f(x_k) - f^* \leq \frac{2L\|x_0 - x^*\|^2}{(k+1)^2}$$

## Ускоренный градиентный метод Нестерова

$$\begin{aligned}y_k &= x_k + \frac{k-1}{k+2}(x_k - x_{k-1}) \\x_{k+1} &= y_k - \frac{1}{L}\nabla f(y_k)\end{aligned}$$

**Теорема:** Для  $L$ -гладких выпуклых функций:

$$f(x_k) - f^* \leq \frac{2L\|x_0 - x^*\|^2}{(k+1)^2}$$

Это **оптимальная** скорость для методов первого порядка!

## Метод тяжёлого шарика (Heavy Ball)

$$x_{k+1} = x_k - \alpha \nabla f(x_k) + \beta(x_k - x_{k-1})$$

## Метод тяжёлого шарика (Heavy Ball)

$$x_{k+1} = x_k - \alpha \nabla f(x_k) + \beta(x_k - x_{k-1})$$

Оптимальные параметры для квадратичных функций:

$$\alpha = \frac{4}{(\sqrt{L} + \sqrt{\mu})^2}, \quad \beta = \left( \frac{\sqrt{\kappa} - 1}{\sqrt{\kappa} + 1} \right)^2$$

## Задача (Heavy Ball)

**i** Для  $f(x) = \frac{\lambda}{2}x^2$  ( $\lambda > 0$ ,  $x \in \mathbb{R}$ ) и Heavy Ball  $x_{k+1} = x_k - \alpha \nabla f(x_k) + \beta(x_k - x_{k-1})$ :

1) Запишите итерацию в матричном виде  $\begin{pmatrix} x_{k+1} \\ x_k \end{pmatrix} = T \begin{pmatrix} x_k \\ x_{k-1} \end{pmatrix}$

## Задача (Heavy Ball)

**i** Для  $f(x) = \frac{\lambda}{2}x^2$  ( $\lambda > 0$ ,  $x \in \mathbb{R}$ ) и Heavy Ball  $x_{k+1} = x_k - \alpha \nabla f(x_k) + \beta(x_k - x_{k-1})$ :

1) Запишите итерацию в матричном виде  $\begin{pmatrix} x_{k+1} \\ x_k \end{pmatrix} = T \begin{pmatrix} x_k \\ x_{k-1} \end{pmatrix}$

2) При  $\alpha = 1/\lambda$  найдите собственные числа  $T$ . При каких  $\beta$  метод сходится?

## Задача (Heavy Ball)

**i** Для  $f(x) = \frac{\lambda}{2}x^2$  ( $\lambda > 0$ ,  $x \in \mathbb{R}$ ) и Heavy Ball  $x_{k+1} = x_k - \alpha \nabla f(x_k) + \beta(x_k - x_{k-1})$ :

1) Запишите итерацию в матричном виде  $\begin{pmatrix} x_{k+1} \\ x_k \end{pmatrix} = T \begin{pmatrix} x_k \\ x_{k-1} \end{pmatrix}$

2) При  $\alpha = 1/\lambda$  найдите собственные числа  $T$ . При каких  $\beta$  метод сходится?

## Задача (Heavy Ball)

**i** Для  $f(x) = \frac{\lambda}{2}x^2$  ( $\lambda > 0$ ,  $x \in \mathbb{R}$ ) и Heavy Ball  $x_{k+1} = x_k - \alpha \nabla f(x_k) + \beta(x_k - x_{k-1})$ :

1) Запишите итерацию в матричном виде  $\begin{pmatrix} x_{k+1} \\ x_k \end{pmatrix} = T \begin{pmatrix} x_k \\ x_{k-1} \end{pmatrix}$

2) При  $\alpha = 1/\lambda$  найдите собственные числа  $T$ . При каких  $\beta$  метод сходится?

**Решение:** 1)  $T = \begin{pmatrix} 1 - \alpha\lambda + \beta & -\beta \\ 1 & 0 \end{pmatrix}$ . При  $\alpha = 1/\lambda$ :  $T = \begin{pmatrix} \beta & -\beta \\ 1 & 0 \end{pmatrix}$

## Задача (Heavy Ball)

**i** Для  $f(x) = \frac{\lambda}{2}x^2$  ( $\lambda > 0$ ,  $x \in \mathbb{R}$ ) и Heavy Ball  $x_{k+1} = x_k - \alpha \nabla f(x_k) + \beta(x_k - x_{k-1})$ :

1) Запишите итерацию в матричном виде  $\begin{pmatrix} x_{k+1} \\ x_k \end{pmatrix} = T \begin{pmatrix} x_k \\ x_{k-1} \end{pmatrix}$

2) При  $\alpha = 1/\lambda$  найдите собственные числа  $T$ . При каких  $\beta$  метод сходится?

**Решение:** 1)  $T = \begin{pmatrix} 1 - \alpha\lambda + \beta & -\beta \\ 1 & 0 \end{pmatrix}$ . При  $\alpha = 1/\lambda$ :  $T = \begin{pmatrix} \beta & -\beta \\ 1 & 0 \end{pmatrix}$

2) Характеристическое уравнение:  $r^2 - \beta r + \beta = 0$ , откуда  $r_{1,2} = \frac{\beta \pm \sqrt{\beta^2 - 4\beta}}{2}$

Условие сходимости  $|r_{1,2}| < 1$  выполняется при  $-\frac{1}{2} < \beta < 1$ .

## Метод сопряжённых градиентов (Лекция 13)

## Идея метода

Для квадратичной функции  $f(x) = \frac{1}{2}x^T Ax - b^T x, A \succ 0$ :

## Идея метода

Для квадратичной функции  $f(x) = \frac{1}{2}x^T Ax - b^T x, A \succ 0$ :

**$A$ -сопряжённость (ортогональность):**

$$\langle d_i, d_j \rangle_A = d_i^T A d_j = 0 \quad \text{для } i \neq j$$

## Идея метода

Для квадратичной функции  $f(x) = \frac{1}{2}x^T Ax - b^T x$ ,  $A \succ 0$ :

**$A$ -сопряжённость (ортогональность):**

$$\langle d_i, d_j \rangle_A = d_i^T A d_j = 0 \quad \text{для } i \neq j$$

**Теорема:** Метод сопряжённых градиентов сходится **не более чем за  $n$  итераций** для квадратичной функции в  $\mathbb{R}^n$ .

# Алгоритм сопряжённых градиентов

1.  $r_0 = b - Ax_0, d_0 = r_0$

## Алгоритм сопряжённых градиентов

1.  $r_0 = b - Ax_0, d_0 = r_0$
2. Для  $k = 0, 1, \dots$ :

## Алгоритм сопряжённых градиентов

1.  $r_0 = b - Ax_0, d_0 = r_0$

2. Для  $k = 0, 1, \dots$ :

- $\alpha_k = \frac{r_k^T r_k}{d_k^T A d_k}$

## Алгоритм сопряжённых градиентов

1.  $r_0 = b - Ax_0, d_0 = r_0$

2. Для  $k = 0, 1, \dots$ :

- $\alpha_k = \frac{r_k^T r_k}{d_k^T A d_k}$

- $x_{k+1} = x_k + \alpha_k d_k$

## Алгоритм сопряжённых градиентов

1.  $r_0 = b - Ax_0, d_0 = r_0$

2. Для  $k = 0, 1, \dots$ :

- $\alpha_k = \frac{r_k^T r_k}{d_k^T A d_k}$
- $x_{k+1} = x_k + \alpha_k d_k$
- $r_{k+1} = r_k - \alpha_k A d_k$

## Алгоритм сопряжённых градиентов

1.  $r_0 = b - Ax_0, d_0 = r_0$

2. Для  $k = 0, 1, \dots$ :

- $\alpha_k = \frac{r_k^T r_k}{d_k^T A d_k}$
- $x_{k+1} = x_k + \alpha_k d_k$
- $r_{k+1} = r_k - \alpha_k A d_k$
- $\beta_k = \frac{r_{k+1}^T r_{k+1}}{r_k^T r_k}$

## Алгоритм сопряжённых градиентов

1.  $r_0 = b - Ax_0, d_0 = r_0$

2. Для  $k = 0, 1, \dots$ :

- $\alpha_k = \frac{r_k^T r_k}{d_k^T A d_k}$
- $x_{k+1} = x_k + \alpha_k d_k$
- $r_{k+1} = r_k - \alpha_k A d_k$
- $\beta_k = \frac{r_{k+1}^T r_{k+1}}{r_k^T r_k}$
- $d_{k+1} = r_{k+1} + \beta_k d_k$

## Алгоритм сопряжённых градиентов

1.  $r_0 = b - Ax_0, d_0 = r_0$

2. Для  $k = 0, 1, \dots$ :

- $\alpha_k = \frac{r_k^T r_k}{d_k^T A d_k}$
- $x_{k+1} = x_k + \alpha_k d_k$
- $r_{k+1} = r_k - \alpha_k A d_k$
- $\beta_k = \frac{r_{k+1}^T r_{k+1}}{r_k^T r_k}$
- $d_{k+1} = r_{k+1} + \beta_k d_k$

# Алгоритм сопряжённых градиентов

1.  $r_0 = b - Ax_0, d_0 = r_0$

2. Для  $k = 0, 1, \dots$ :

- $\alpha_k = \frac{r_k^T r_k}{d_k^T A d_k}$
- $x_{k+1} = x_k + \alpha_k d_k$
- $r_{k+1} = r_k - \alpha_k A d_k$
- $\beta_k = \frac{r_{k+1}^T r_{k+1}}{r_k^T r_k}$
- $d_{k+1} = r_{k+1} + \beta_k d_k$

Преимущества:

- Не нужно хранить матрицу  $A$  (только умножение  $Av$ )

# Алгоритм сопряжённых градиентов

1.  $r_0 = b - Ax_0, d_0 = r_0$

2. Для  $k = 0, 1, \dots$ :

- $\alpha_k = \frac{r_k^T r_k}{d_k^T A d_k}$
- $x_{k+1} = x_k + \alpha_k d_k$
- $r_{k+1} = r_k - \alpha_k A d_k$
- $\beta_k = \frac{r_{k+1}^T r_{k+1}}{r_k^T r_k}$
- $d_{k+1} = r_{k+1} + \beta_k d_k$

Преимущества:

- Не нужно хранить матрицу  $A$  (только умножение  $Av$ )
- Для разреженных систем:  $O(n)$  операций на итерацию

# Алгоритм сопряжённых градиентов

1.  $r_0 = b - Ax_0, d_0 = r_0$

2. Для  $k = 0, 1, \dots$ :

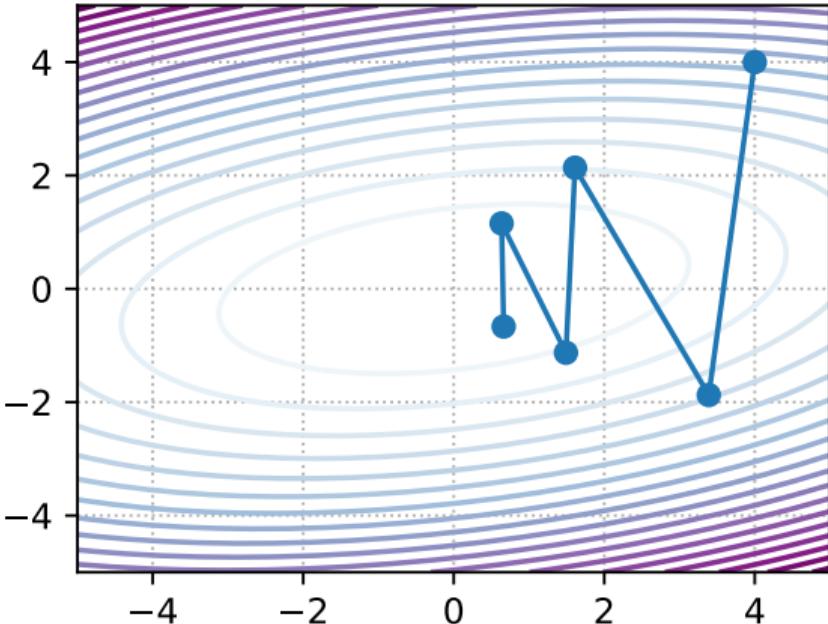
- $\alpha_k = \frac{r_k^T r_k}{d_k^T A d_k}$
- $x_{k+1} = x_k + \alpha_k d_k$
- $r_{k+1} = r_k - \alpha_k A d_k$
- $\beta_k = \frac{r_{k+1}^T r_{k+1}}{r_k^T r_k}$
- $d_{k+1} = r_{k+1} + \beta_k d_k$

Преимущества:

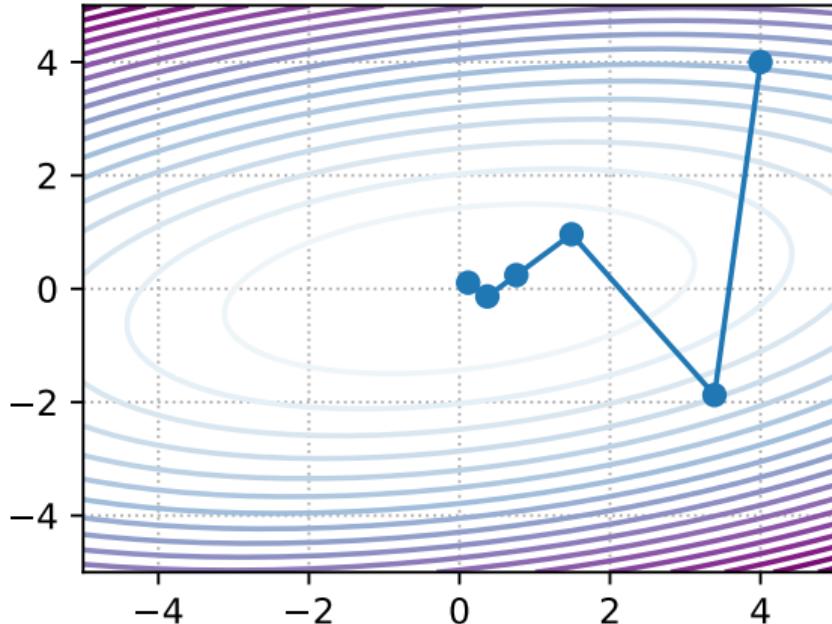
- Не нужно хранить матрицу  $A$  (только умножение  $Av$ )
- Для разреженных систем:  $O(n)$  операций на итерацию
- На практике сходится быстрее, чем за  $n$  итераций

## Градиентный спуск vs Метод тяжёлого шарика

Gradient Descent

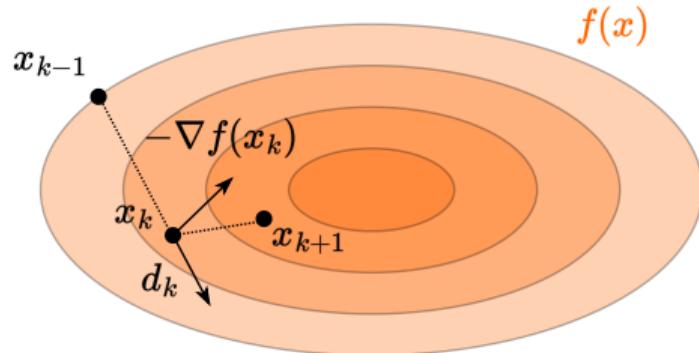


Heavy Ball

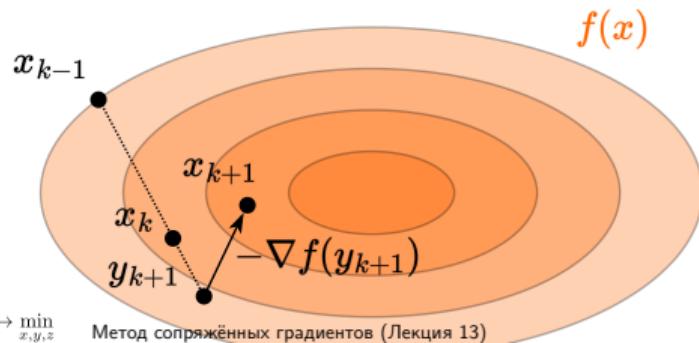


## Ускоренный градиентный метод

### Polyak momentum

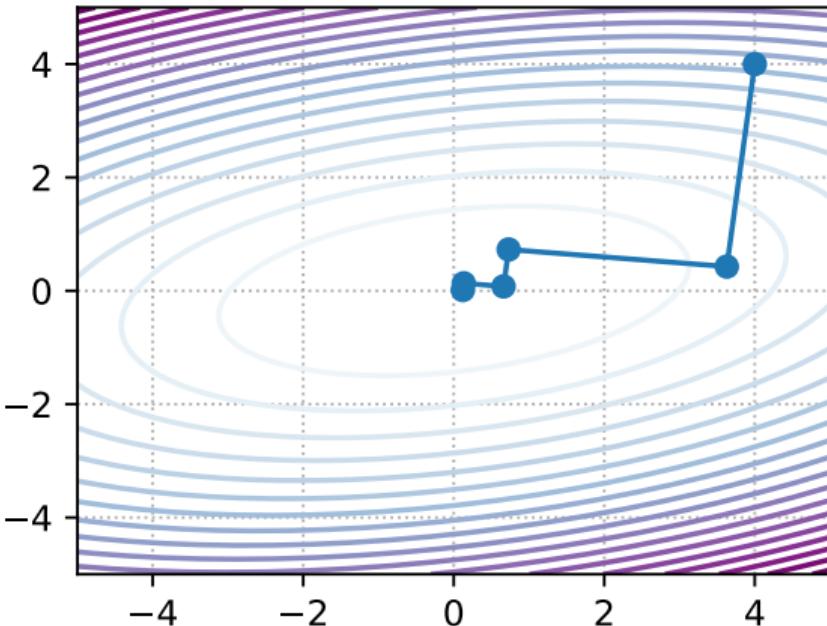


### Nesterov momentum

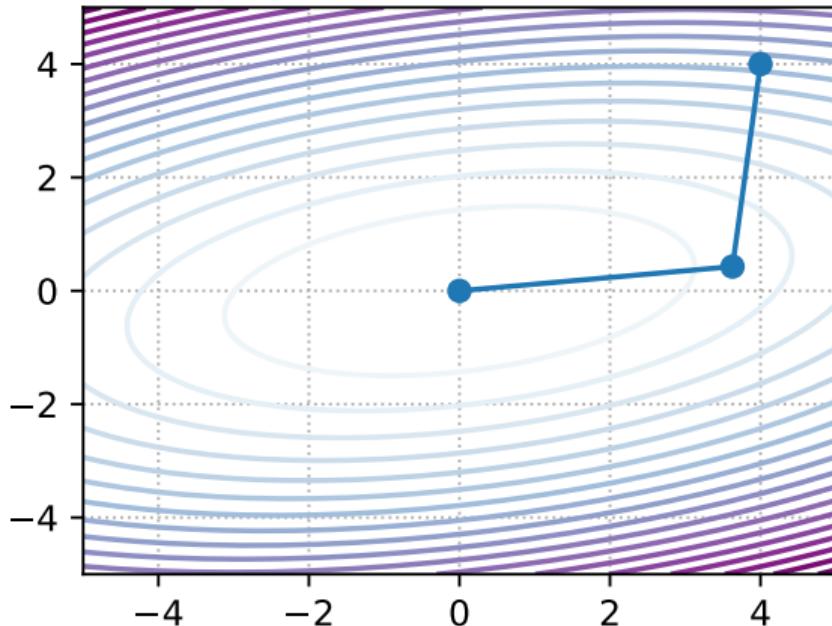


## Градиентный спуск vs Метод сопряжённых градиентов

Steepest Descent



Conjugate Gradient

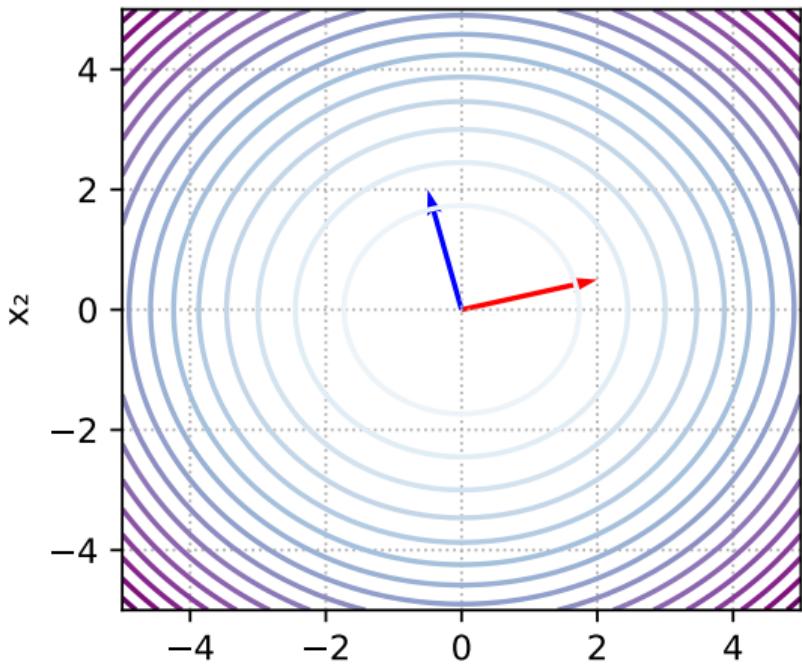


## *A*-ортогональность

$v_1$  and  $v_2$  are orthogonal

$$v_1^T v_2 = 0.00$$

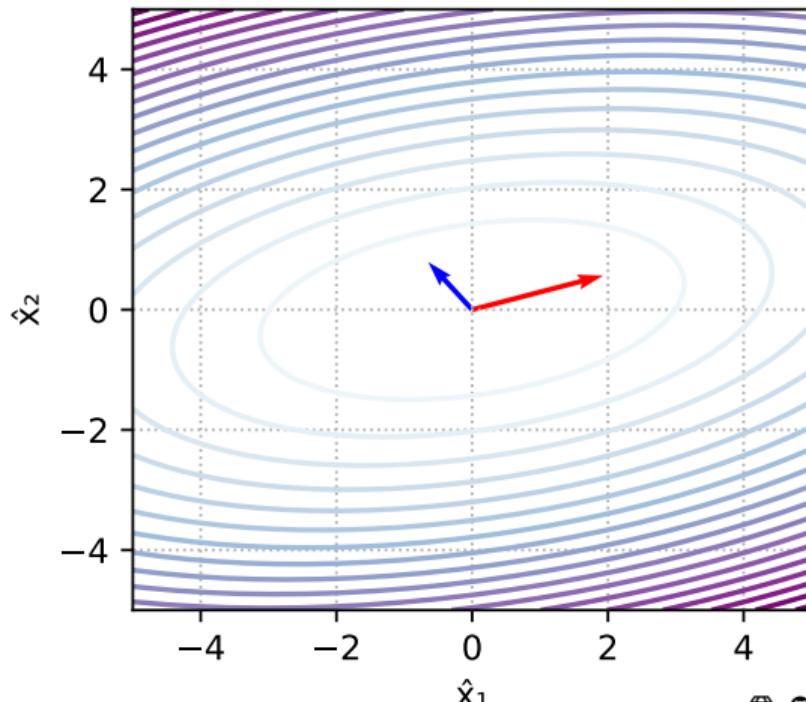
$$v_1^T A v_2 = 1.19$$



$\hat{v}_1$  and  $\hat{v}_2$  are  $A$ -orthogonal

$$\hat{v}_1^T \hat{v}_2 = -0.80$$

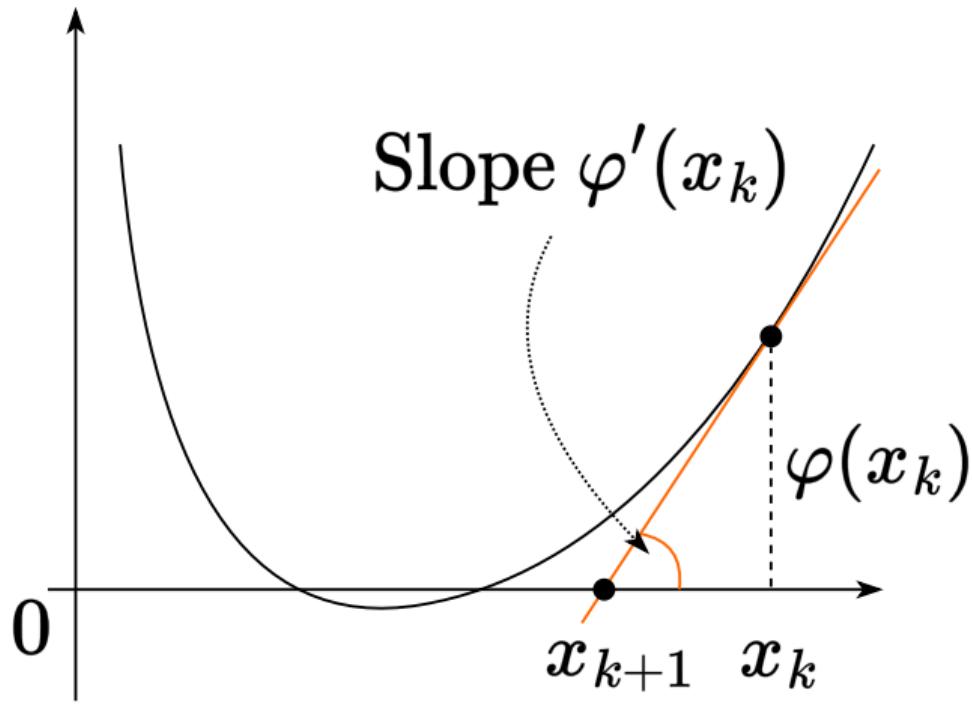
$$\hat{v}_1^T A \hat{v}_2 = -0.00$$



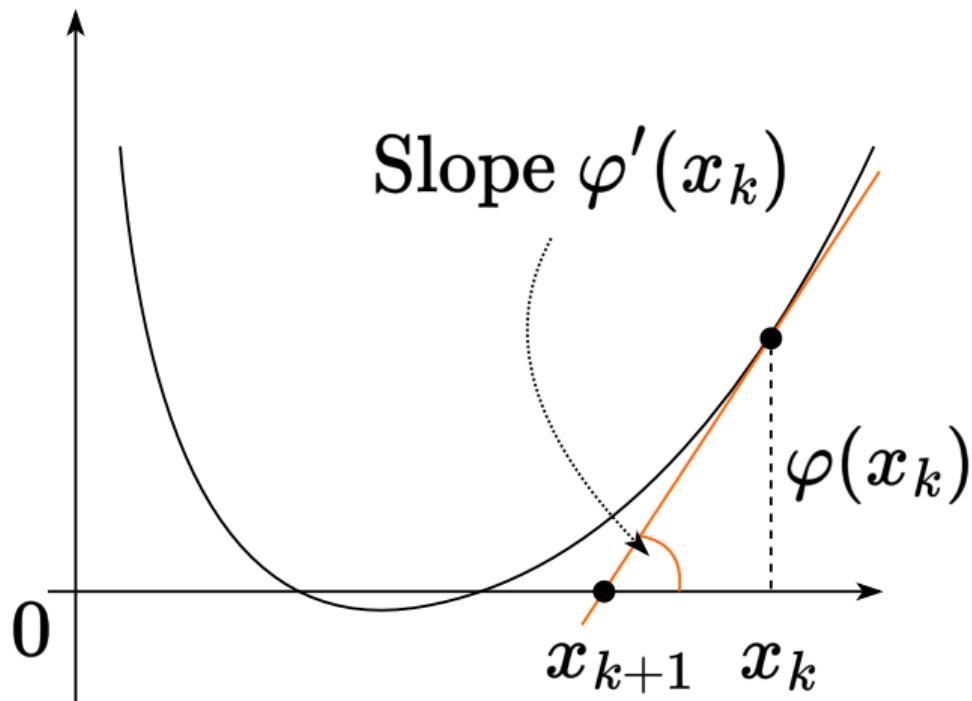
## Newton method

## Idea of Newton method of root finding

Consider the function  $\varphi(x) : \mathbb{R} \rightarrow \mathbb{R}$ .

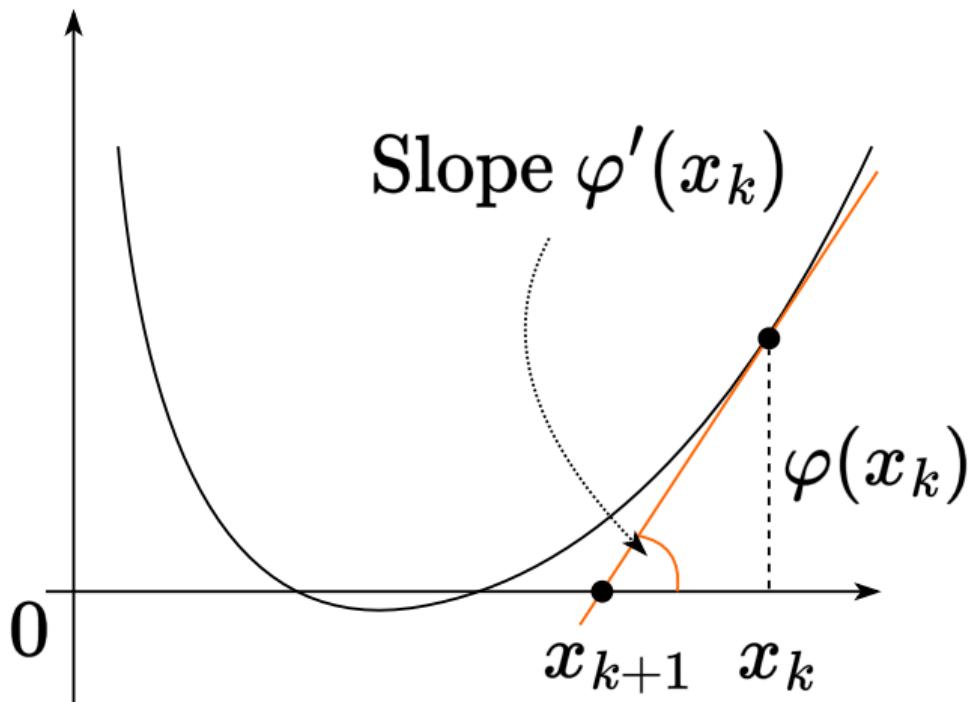


## Idea of Newton method of root finding



Consider the function  $\varphi(x) : \mathbb{R} \rightarrow \mathbb{R}$ .  
The whole idea came from building a linear approximation at the point  $x_k$  and find its root, which will be the new iteration point:

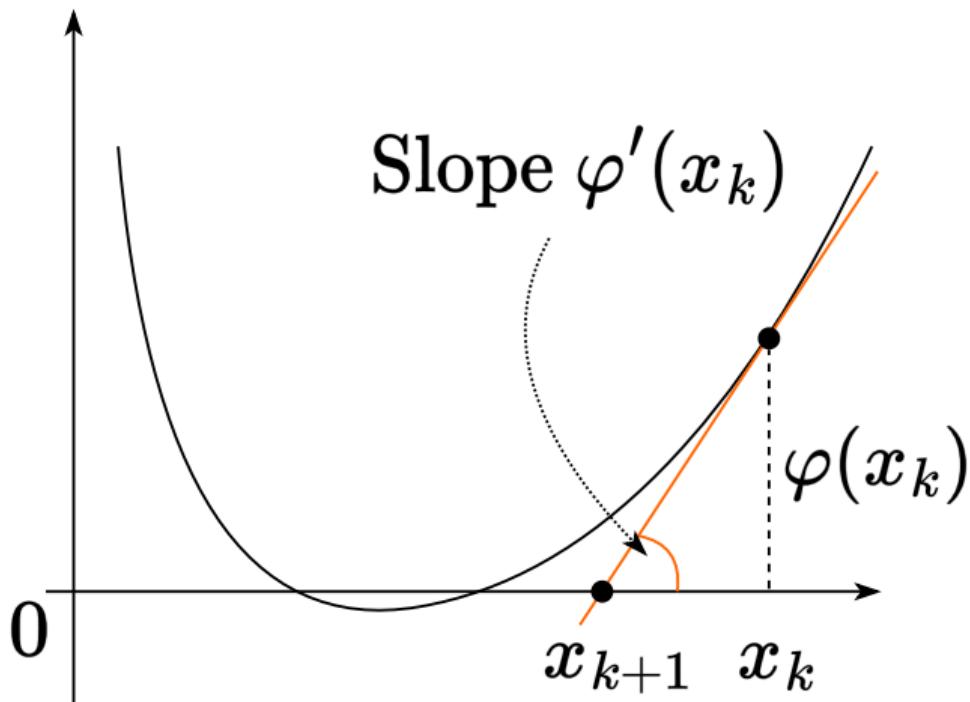
## Idea of Newton method of root finding



Consider the function  $\varphi(x) : \mathbb{R} \rightarrow \mathbb{R}$ .  
The whole idea came from building a linear approximation at the point  $x_k$  and find its root, which will be the new iteration point:

$$\varphi'(x_k) = \frac{\varphi(x_k)}{x_{k+1} - x_k}$$

## Idea of Newton method of root finding

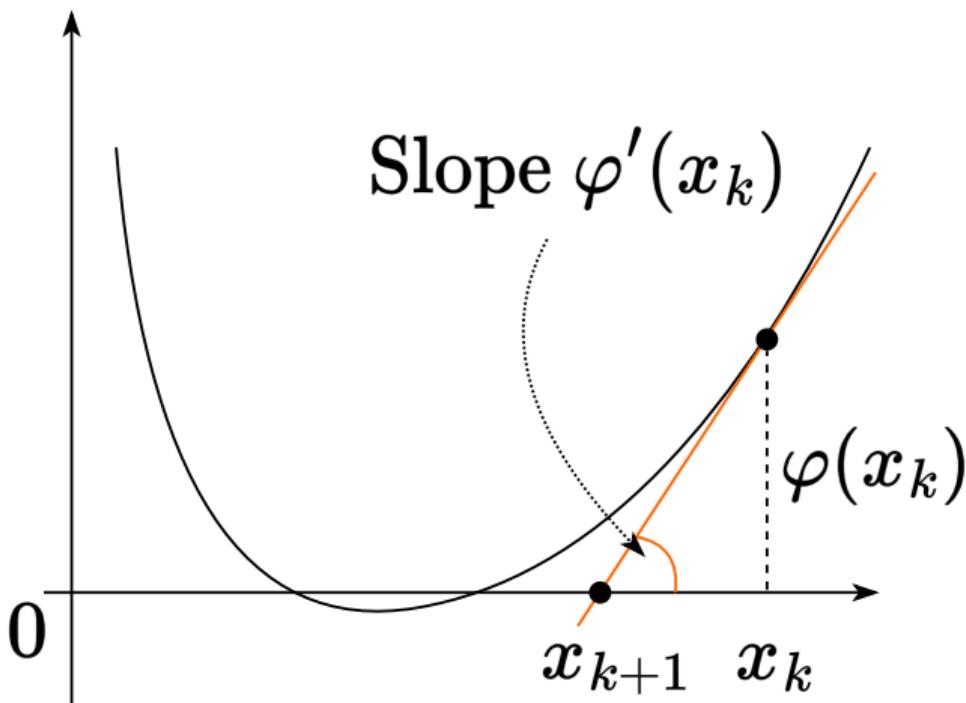


Consider the function  $\varphi(x) : \mathbb{R} \rightarrow \mathbb{R}$ .  
The whole idea came from building a linear approximation at the point  $x_k$  and find its root, which will be the new iteration point:

$$\varphi'(x_k) = \frac{\varphi(x_k)}{x_{k+1} - x_k}$$

We get an iterative scheme:

## Idea of Newton method of root finding



Consider the function  $\varphi(x) : \mathbb{R} \rightarrow \mathbb{R}$ . The whole idea came from building a linear approximation at the point  $x_k$  and find its root, which will be the new iteration point:

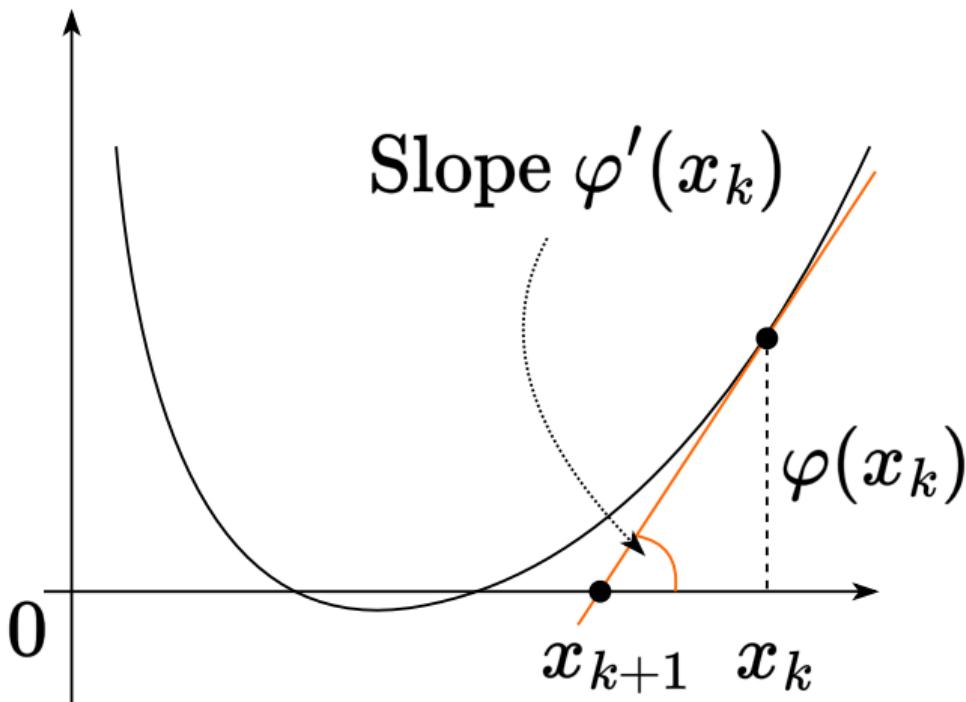
$$\varphi'(x_k) = \frac{\varphi(x_k)}{x_{k+1} - x_k}$$

We get an iterative scheme:

$$x_{k+1} = x_k - \frac{\varphi(x_k)}{\varphi'(x_k)}.$$

<sup>1</sup>Literally we aim to solve the problem of finding stationary points  $\nabla f(x) = 0$

## Idea of Newton method of root finding



Consider the function  $\varphi(x) : \mathbb{R} \rightarrow \mathbb{R}$ .  
The whole idea came from building a linear approximation at the point  $x_k$  and find its root, which will be the new iteration point:

$$\varphi'(x_k) = \frac{\varphi(x_k)}{x_{k+1} - x_k}$$

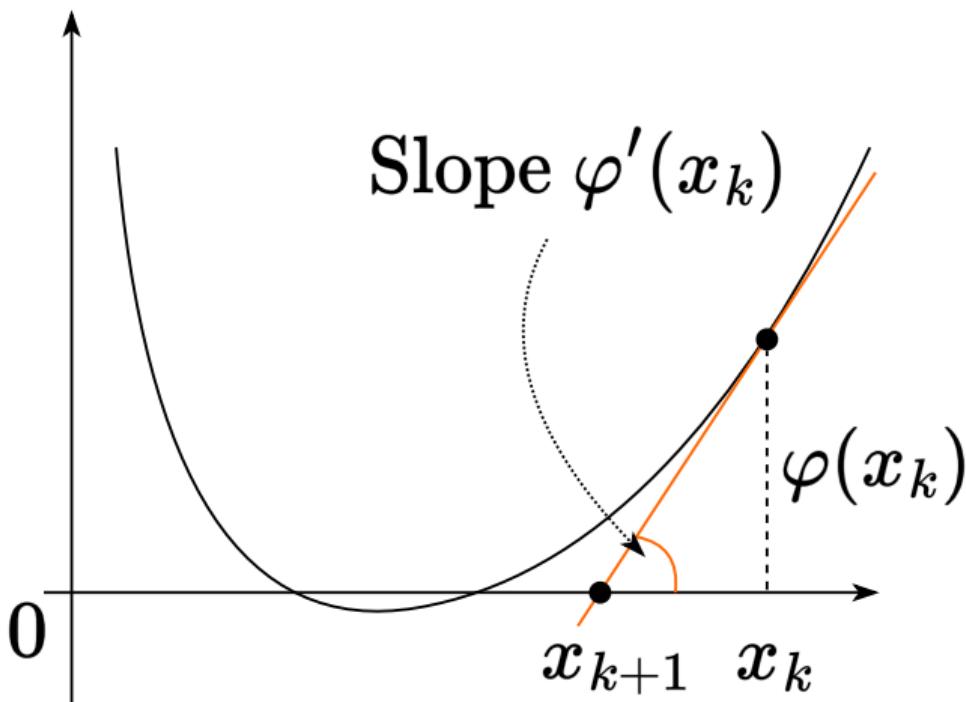
We get an iterative scheme:

$$x_{k+1} = x_k - \frac{\varphi(x_k)}{\varphi'(x_k)}.$$

Which will become a Newton optimization method in case  $f'(x) = \varphi(x)$ <sup>1</sup>:

<sup>1</sup>Literally we aim to solve the problem of finding stationary points  $\nabla f(x) = 0$

## Idea of Newton method of root finding



Consider the function  $\varphi(x) : \mathbb{R} \rightarrow \mathbb{R}$ . The whole idea came from building a linear approximation at the point  $x_k$  and find its root, which will be the new iteration point:

$$\varphi'(x_k) = \frac{\varphi(x_k)}{x_{k+1} - x_k}$$

We get an iterative scheme:

$$x_{k+1} = x_k - \frac{\varphi(x_k)}{\varphi'(x_k)}.$$

Which will become a Newton optimization method in case  $f'(x) = \varphi(x)$ <sup>1</sup>:

$$x_{k+1} = x_k - [\nabla^2 f(x_k)]^{-1} \nabla f(x_k)$$

<sup>1</sup>Literally we aim to solve the problem of finding stationary points  $\nabla f(x) = 0$

## Newton method as a local quadratic Taylor approximation minimizer

Let us now have the function  $f(x)$  and a certain point  $x_k$ . Let us consider the quadratic approximation of this function near  $x_k$ :

## Newton method as a local quadratic Taylor approximation minimizer

Let us now have the function  $f(x)$  and a certain point  $x_k$ . Let us consider the quadratic approximation of this function near  $x_k$ :

$$f_{x_k}^{II}(x) = f(x_k) + \langle \nabla f(x_k), x - x_k \rangle + \frac{1}{2} \langle \nabla^2 f(x_k)(x - x_k), x - x_k \rangle.$$

## Newton method as a local quadratic Taylor approximation minimizer

Let us now have the function  $f(x)$  and a certain point  $x_k$ . Let us consider the quadratic approximation of this function near  $x_k$ :

$$f_{x_k}^{II}(x) = f(x_k) + \langle \nabla f(x_k), x - x_k \rangle + \frac{1}{2} \langle \nabla^2 f(x_k)(x - x_k), x - x_k \rangle.$$

The idea of the method is to find the point  $x_{k+1}$ , that minimizes the function  $f_{x_k}^{II}(x)$ , i.e.  $\nabla f_{x_k}^{II}(x_{k+1}) = 0$ .

## Newton method as a local quadratic Taylor approximation minimizer

Let us now have the function  $f(x)$  and a certain point  $x_k$ . Let us consider the quadratic approximation of this function near  $x_k$ :

$$f_{x_k}^{II}(x) = f(x_k) + \langle \nabla f(x_k), x - x_k \rangle + \frac{1}{2} \langle \nabla^2 f(x_k)(x - x_k), x - x_k \rangle.$$

The idea of the method is to find the point  $x_{k+1}$ , that minimizes the function  $f_{x_k}^{II}(x)$ , i.e.  $\nabla f_{x_k}^{II}(x_{k+1}) = 0$ .

$$\nabla f_{x_k}^{II}(x_{k+1}) = \nabla f(x_k) + \nabla^2 f(x_k)(x_{k+1} - x_k) = 0$$

## Newton method as a local quadratic Taylor approximation minimizer

Let us now have the function  $f(x)$  and a certain point  $x_k$ . Let us consider the quadratic approximation of this function near  $x_k$ :

$$f_{x_k}^{II}(x) = f(x_k) + \langle \nabla f(x_k), x - x_k \rangle + \frac{1}{2} \langle \nabla^2 f(x_k)(x - x_k), x - x_k \rangle.$$

The idea of the method is to find the point  $x_{k+1}$ , that minimizes the function  $f_{x_k}^{II}(x)$ , i.e.  $\nabla f_{x_k}^{II}(x_{k+1}) = 0$ .

$$\nabla f_{x_k}^{II}(x_{k+1}) = \nabla f(x_k) + \nabla^2 f(x_k)(x_{k+1} - x_k) = 0$$

$$\nabla^2 f(x_k)(x_{k+1} - x_k) = -\nabla f(x_k)$$

## Newton method as a local quadratic Taylor approximation minimizer

Let us now have the function  $f(x)$  and a certain point  $x_k$ . Let us consider the quadratic approximation of this function near  $x_k$ :

$$f_{x_k}^{II}(x) = f(x_k) + \langle \nabla f(x_k), x - x_k \rangle + \frac{1}{2} \langle \nabla^2 f(x_k)(x - x_k), x - x_k \rangle.$$

The idea of the method is to find the point  $x_{k+1}$ , that minimizes the function  $f_{x_k}^{II}(x)$ , i.e.  $\nabla f_{x_k}^{II}(x_{k+1}) = 0$ .

$$\nabla f_{x_k}^{II}(x_{k+1}) = \nabla f(x_k) + \nabla^2 f(x_k)(x_{k+1} - x_k) = 0$$

$$\nabla^2 f(x_k)(x_{k+1} - x_k) = -\nabla f(x_k)$$

$$[\nabla^2 f(x_k)]^{-1} \nabla^2 f(x_k)(x_{k+1} - x_k) = -[\nabla^2 f(x_k)]^{-1} \nabla f(x_k)$$

## Newton method as a local quadratic Taylor approximation minimizer

Let us now have the function  $f(x)$  and a certain point  $x_k$ . Let us consider the quadratic approximation of this function near  $x_k$ :

$$f_{x_k}^{II}(x) = f(x_k) + \langle \nabla f(x_k), x - x_k \rangle + \frac{1}{2} \langle \nabla^2 f(x_k)(x - x_k), x - x_k \rangle.$$

The idea of the method is to find the point  $x_{k+1}$ , that minimizes the function  $f_{x_k}^{II}(x)$ , i.e.  $\nabla f_{x_k}^{II}(x_{k+1}) = 0$ .

$$\nabla f_{x_k}^{II}(x_{k+1}) = \nabla f(x_k) + \nabla^2 f(x_k)(x_{k+1} - x_k) = 0$$

$$\nabla^2 f(x_k)(x_{k+1} - x_k) = -\nabla f(x_k)$$

$$[\nabla^2 f(x_k)]^{-1} \nabla^2 f(x_k)(x_{k+1} - x_k) = -[\nabla^2 f(x_k)]^{-1} \nabla f(x_k)$$

$$x_{k+1} = x_k - [\nabla^2 f(x_k)]^{-1} \nabla f(x_k).$$

## Newton method as a local quadratic Taylor approximation minimizer

Let us now have the function  $f(x)$  and a certain point  $x_k$ . Let us consider the quadratic approximation of this function near  $x_k$ :

$$f_{x_k}^{II}(x) = f(x_k) + \langle \nabla f(x_k), x - x_k \rangle + \frac{1}{2} \langle \nabla^2 f(x_k)(x - x_k), x - x_k \rangle.$$

The idea of the method is to find the point  $x_{k+1}$ , that minimizes the function  $f_{x_k}^{II}(x)$ , i.e.  $\nabla f_{x_k}^{II}(x_{k+1}) = 0$ .

$$\nabla f_{x_k}^{II}(x_{k+1}) = \nabla f(x_k) + \nabla^2 f(x_k)(x_{k+1} - x_k) = 0$$

$$\nabla^2 f(x_k)(x_{k+1} - x_k) = -\nabla f(x_k)$$

$$[\nabla^2 f(x_k)]^{-1} \nabla^2 f(x_k)(x_{k+1} - x_k) = -[\nabla^2 f(x_k)]^{-1} \nabla f(x_k)$$

$$x_{k+1} = x_k - [\nabla^2 f(x_k)]^{-1} \nabla f(x_k).$$

## Newton method as a local quadratic Taylor approximation minimizer

Let us now have the function  $f(x)$  and a certain point  $x_k$ . Let us consider the quadratic approximation of this function near  $x_k$ :

$$f_{x_k}^{II}(x) = f(x_k) + \langle \nabla f(x_k), x - x_k \rangle + \frac{1}{2} \langle \nabla^2 f(x_k)(x - x_k), x - x_k \rangle.$$

The idea of the method is to find the point  $x_{k+1}$ , that minimizes the function  $f_{x_k}^{II}(x)$ , i.e.  $\nabla f_{x_k}^{II}(x_{k+1}) = 0$ .

$$\nabla f_{x_k}^{II}(x_{k+1}) = \nabla f(x_k) + \nabla^2 f(x_k)(x_{k+1} - x_k) = 0$$

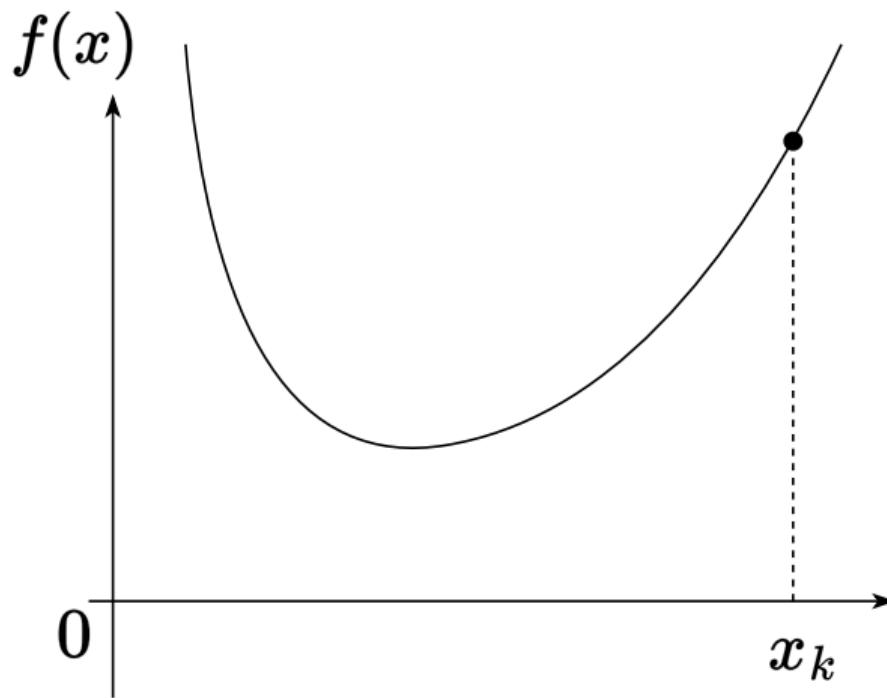
$$\nabla^2 f(x_k)(x_{k+1} - x_k) = -\nabla f(x_k)$$

$$[\nabla^2 f(x_k)]^{-1} \nabla^2 f(x_k)(x_{k+1} - x_k) = -[\nabla^2 f(x_k)]^{-1} \nabla f(x_k)$$

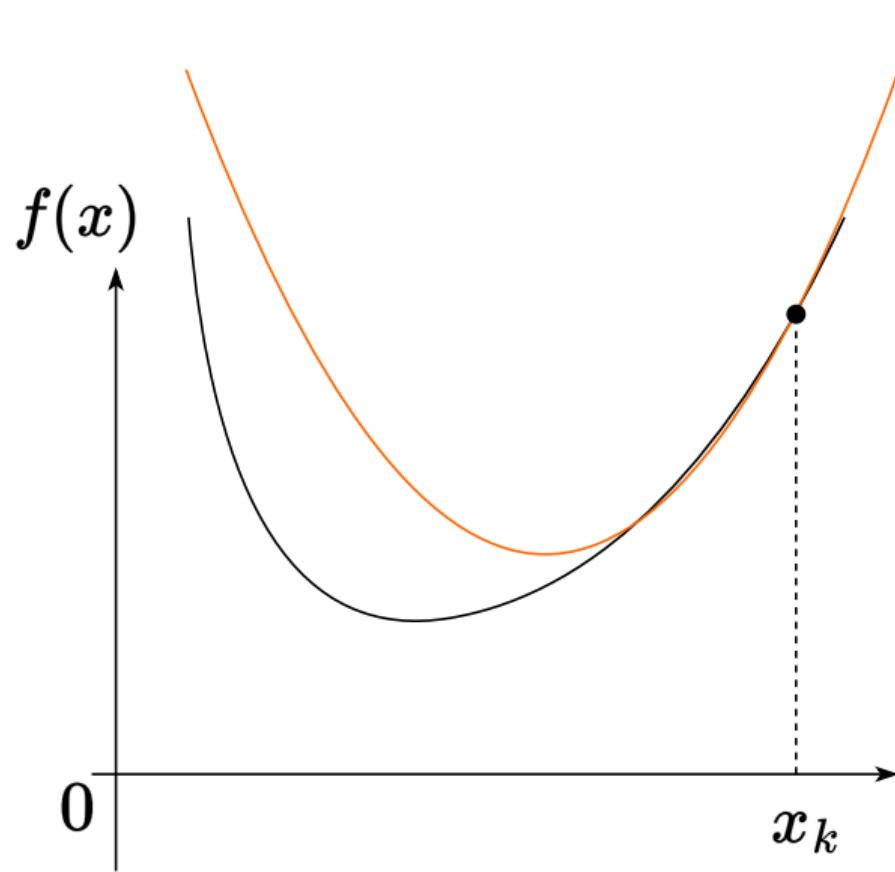
$$x_{k+1} = x_k - [\nabla^2 f(x_k)]^{-1} \nabla f(x_k).$$

Let us immediately note the limitations related to the necessity of the Hessian's non-degeneracy (for the method to exist), as well as its positive definiteness (for the convergence guarantee).

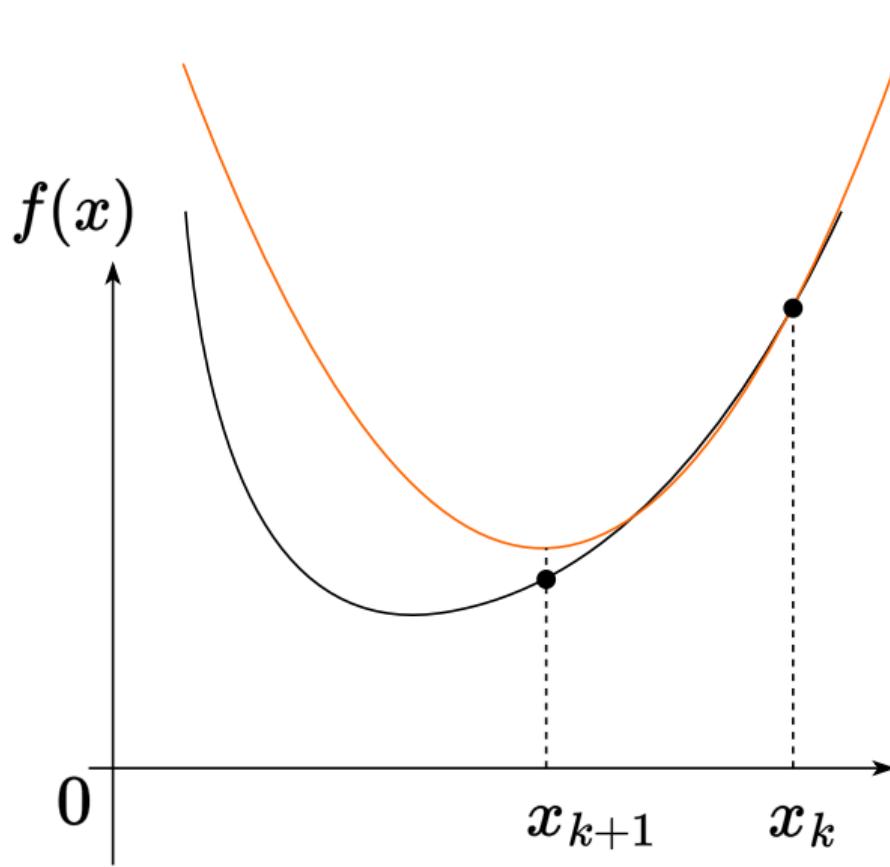
## Newton method as a local quadratic Taylor approximation minimizer



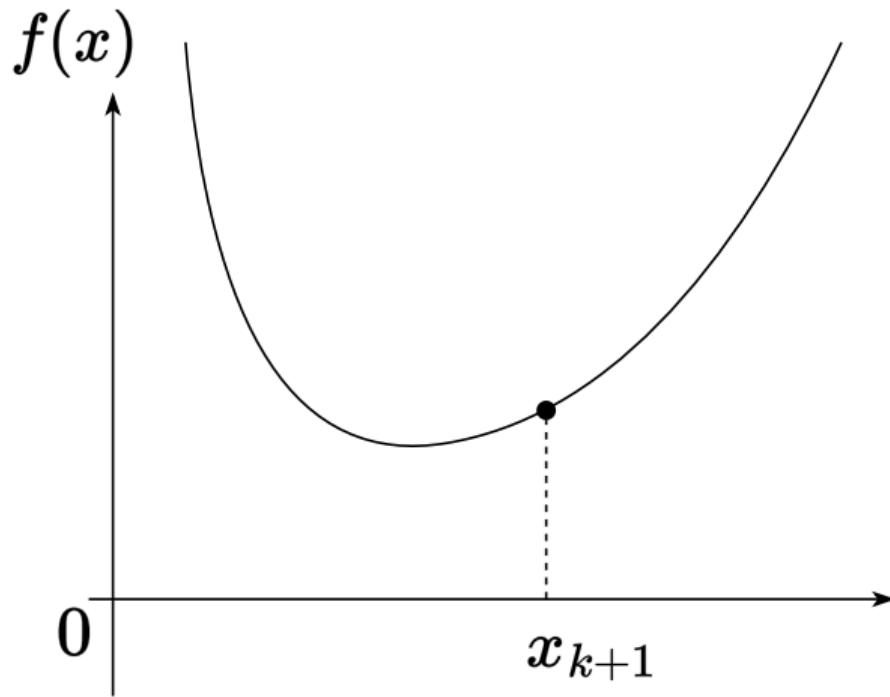
## Newton method as a local quadratic Taylor approximation minimizer



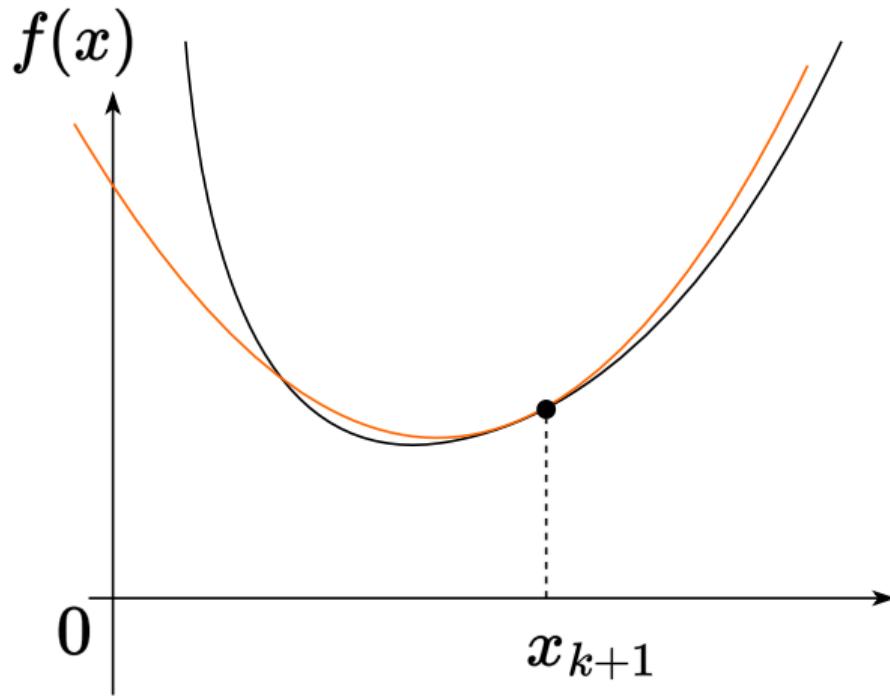
## Newton method as a local quadratic Taylor approximation minimizer



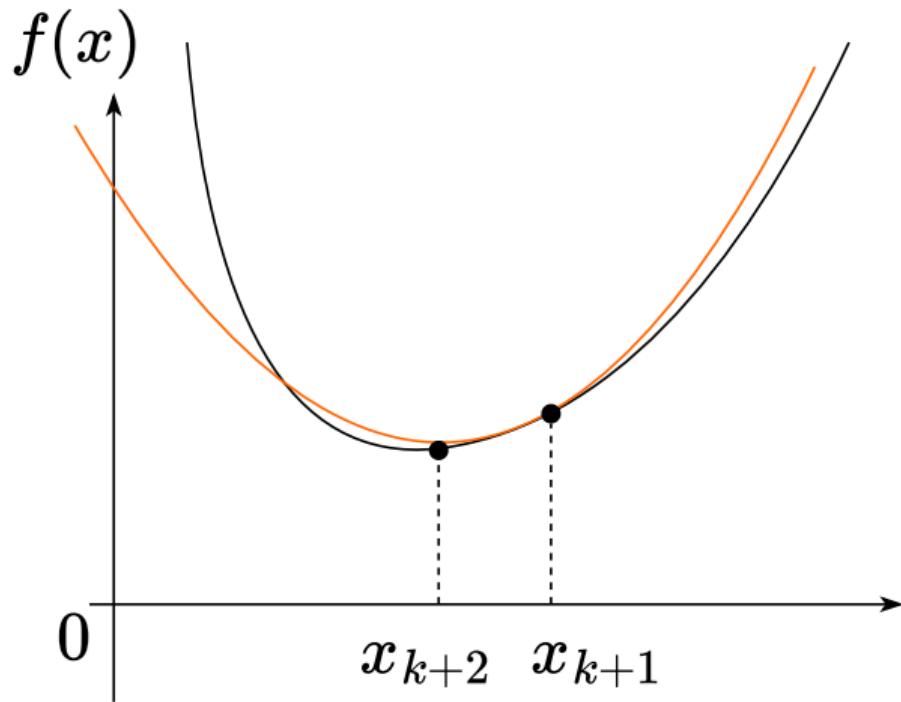
## Newton method as a local quadratic Taylor approximation minimizer



## Newton method as a local quadratic Taylor approximation minimizer



## Newton method as a local quadratic Taylor approximation minimizer



# Convergence

## i Theorem

Let  $f(x)$  be a strongly convex twice continuously differentiable function at  $\mathbb{R}^n$ , for the second derivative of which inequalities are executed:  $\mu I_n \preceq \nabla^2 f(x) \preceq L I_n$ . Then Newton's method with a constant step locally converges to solving the problem with superlinear speed. If, in addition, Hessian is  $M$ -Lipschitz continuous, then this method converges locally to  $x^*$  at a quadratic rate.

# Convergence

## i Theorem

Let  $f(x)$  be a strongly convex twice continuously differentiable function at  $\mathbb{R}^n$ , for the second derivative of which inequalities are executed:  $\mu I_n \preceq \nabla^2 f(x) \preceq L I_n$ . Then Newton's method with a constant step locally converges to solving the problem with superlinear speed. If, in addition, Hessian is  $M$ -Lipschitz continuous, then this method converges locally to  $x^*$  at a quadratic rate.

## Proof

# Convergence

## i Theorem

Let  $f(x)$  be a strongly convex twice continuously differentiable function at  $\mathbb{R}^n$ , for the second derivative of which inequalities are executed:  $\mu I_n \preceq \nabla^2 f(x) \preceq L I_n$ . Then Newton's method with a constant step locally converges to solving the problem with superlinear speed. If, in addition, Hessian is  $M$ -Lipschitz continuous, then this method converges locally to  $x^*$  at a quadratic rate.

## Proof

1. We will use Newton-Leibniz formula

$$\nabla f(x_k) - \nabla f(x^*) = \int_0^1 \nabla^2 f(x^* + \tau(x_k - x^*))(x_k - x^*) d\tau$$

# Convergence

## i Theorem

Let  $f(x)$  be a strongly convex twice continuously differentiable function at  $\mathbb{R}^n$ , for the second derivative of which inequalities are executed:  $\mu I_n \preceq \nabla^2 f(x) \preceq L I_n$ . Then Newton's method with a constant step locally converges to solving the problem with superlinear speed. If, in addition, Hessian is  $M$ -Lipschitz continuous, then this method converges locally to  $x^*$  at a quadratic rate.

## Proof

1. We will use Newton-Leibniz formula

$$\nabla f(x_k) - \nabla f(x^*) = \int_0^1 \nabla^2 f(x^* + \tau(x_k - x^*))(x_k - x^*) d\tau$$

2. Then we track the distance to the solution

# Convergence

## i Theorem

Let  $f(x)$  be a strongly convex twice continuously differentiable function at  $\mathbb{R}^n$ , for the second derivative of which inequalities are executed:  $\mu I_n \preceq \nabla^2 f(x) \preceq L I_n$ . Then Newton's method with a constant step locally converges to solving the problem with superlinear speed. If, in addition, Hessian is  $M$ -Lipschitz continuous, then this method converges locally to  $x^*$  at a quadratic rate.

## Proof

1. We will use Newton-Leibniz formula

$$\nabla f(x_k) - \nabla f(x^*) = \int_0^1 \nabla^2 f(x^* + \tau(x_k - x^*))(x_k - x^*) d\tau$$

2. Then we track the distance to the solution

$$x_{k+1} - x^* = x_k - [\nabla^2 f(x_k)]^{-1} \nabla f(x_k) - x^* = x_k - x^* - [\nabla^2 f(x_k)]^{-1} \nabla f(x_k) =$$

# Convergence

## i Theorem

Let  $f(x)$  be a strongly convex twice continuously differentiable function at  $\mathbb{R}^n$ , for the second derivative of which inequalities are executed:  $\mu I_n \preceq \nabla^2 f(x) \preceq L I_n$ . Then Newton's method with a constant step locally converges to solving the problem with superlinear speed. If, in addition, Hessian is  $M$ -Lipschitz continuous, then this method converges locally to  $x^*$  at a quadratic rate.

## Proof

1. We will use Newton-Leibniz formula

$$\nabla f(x_k) - \nabla f(x^*) = \int_0^1 \nabla^2 f(x^* + \tau(x_k - x^*))(x_k - x^*) d\tau$$

2. Then we track the distance to the solution

$$\begin{aligned} x_{k+1} - x^* &= x_k - [\nabla^2 f(x_k)]^{-1} \nabla f(x_k) - x^* = x_k - x^* - [\nabla^2 f(x_k)]^{-1} \nabla f(x_k) = \\ &= x_k - x^* - [\nabla^2 f(x_k)]^{-1} \int_0^1 \nabla^2 f(x^* + \tau(x_k - x^*))(x_k - x^*) d\tau \end{aligned}$$

## Convergence

3.

$$= \left( I - [\nabla^2 f(x_k)]^{-1} \int_0^1 \nabla^2 f(x^* + \tau(x_k - x^*)) d\tau \right) (x_k - x^*) =$$

## Convergence

3.

$$\begin{aligned} &= \left( I - [\nabla^2 f(x_k)]^{-1} \int_0^1 \nabla^2 f(x^* + \tau(x_k - x^*)) d\tau \right) (x_k - x^*) = \\ &= [\nabla^2 f(x_k)]^{-1} \left( \nabla^2 f(x_k) - \int_0^1 \nabla^2 f(x^* + \tau(x_k - x^*)) d\tau \right) (x_k - x^*) = \end{aligned}$$

## Convergence

3.

$$\begin{aligned} &= \left( I - [\nabla^2 f(x_k)]^{-1} \int_0^1 \nabla^2 f(x^* + \tau(x_k - x^*)) d\tau \right) (x_k - x^*) = \\ &= [\nabla^2 f(x_k)]^{-1} \left( \nabla^2 f(x_k) - \int_0^1 \nabla^2 f(x^* + \tau(x_k - x^*)) d\tau \right) (x_k - x^*) = \\ &= [\nabla^2 f(x_k)]^{-1} \left( \int_0^1 (\nabla^2 f(x_k) - \nabla^2 f(x^* + \tau(x_k - x^*))) d\tau \right) (x_k - x^*) = \end{aligned}$$

## Convergence

3.

$$\begin{aligned} &= \left( I - [\nabla^2 f(x_k)]^{-1} \int_0^1 \nabla^2 f(x^* + \tau(x_k - x^*)) d\tau \right) (x_k - x^*) = \\ &= [\nabla^2 f(x_k)]^{-1} \left( \nabla^2 f(x_k) - \int_0^1 \nabla^2 f(x^* + \tau(x_k - x^*)) d\tau \right) (x_k - x^*) = \\ &= [\nabla^2 f(x_k)]^{-1} \left( \int_0^1 (\nabla^2 f(x_k) - \nabla^2 f(x^* + \tau(x_k - x^*))) d\tau \right) (x_k - x^*) = \\ &\quad = [\nabla^2 f(x_k)]^{-1} G_k(x_k - x^*) \end{aligned}$$

## Convergence

3.

$$\begin{aligned} &= \left( I - [\nabla^2 f(x_k)]^{-1} \int_0^1 \nabla^2 f(x^* + \tau(x_k - x^*)) d\tau \right) (x_k - x^*) = \\ &= [\nabla^2 f(x_k)]^{-1} \left( \nabla^2 f(x_k) - \int_0^1 \nabla^2 f(x^* + \tau(x_k - x^*)) d\tau \right) (x_k - x^*) = \\ &= [\nabla^2 f(x_k)]^{-1} \left( \int_0^1 (\nabla^2 f(x_k) - \nabla^2 f(x^* + \tau(x_k - x^*))) d\tau \right) (x_k - x^*) = \\ &\quad = [\nabla^2 f(x_k)]^{-1} G_k (x_k - x^*) \end{aligned}$$

4. We have introduced:

$$G_k = \int_0^1 (\nabla^2 f(x_k) - \nabla^2 f(x^* + \tau(x_k - x^*))) d\tau .$$

## Convergence

5. Let's try to estimate the size of  $G_k$ :

where  $r_k = \|x_k - x^*\|$ .

## Convergence

5. Let's try to estimate the size of  $G_k$ :

$$\|G_k\| = \left\| \int_0^1 (\nabla^2 f(x_k) - \nabla^2 f(x^* + \tau(x_k - x^*)) d\tau \right\| \leq$$

where  $r_k = \|x_k - x^*\|$ .

## Convergence

5. Let's try to estimate the size of  $G_k$ :

$$\begin{aligned}\|G_k\| &= \left\| \int_0^1 (\nabla^2 f(x_k) - \nabla^2 f(x^* + \tau(x_k - x^*)) d\tau \right\| \leq \\ &\leq \int_0^1 \|\nabla^2 f(x_k) - \nabla^2 f(x^* + \tau(x_k - x^*))\| d\tau \leq \quad \text{(Hessian's Lipschitz continuity)}\end{aligned}$$

where  $r_k = \|x_k - x^*\|$ .

## Convergence

5. Let's try to estimate the size of  $G_k$ :

$$\begin{aligned}\|G_k\| &= \left\| \int_0^1 (\nabla^2 f(x_k) - \nabla^2 f(x^* + \tau(x_k - x^*)) d\tau \right\| \leq \\ &\leq \int_0^1 \|\nabla^2 f(x_k) - \nabla^2 f(x^* + \tau(x_k - x^*))\| d\tau \leq \quad \text{(Hessian's Lipschitz continuity)} \\ &\leq \int_0^1 M \|x_k - x^* - \tau(x_k - x^*)\| d\tau = \int_0^1 M \|x_k - x^*\| (1 - \tau) d\tau = \frac{r_k}{2} M,\end{aligned}$$

where  $r_k = \|x_k - x^*\|$ .

## Convergence

5. Let's try to estimate the size of  $G_k$ :

$$\begin{aligned} \|G_k\| &= \left\| \int_0^1 (\nabla^2 f(x_k) - \nabla^2 f(x^* + \tau(x_k - x^*)) d\tau \right\| \leq \\ &\leq \int_0^1 \|\nabla^2 f(x_k) - \nabla^2 f(x^* + \tau(x_k - x^*))\| d\tau \leq \quad (\text{Hessian's Lipschitz continuity}) \\ &\leq \int_0^1 M \|x_k - x^* - \tau(x_k - x^*)\| d\tau = \int_0^1 M \|x_k - x^*\| (1 - \tau) d\tau = \frac{r_k}{2} M, \end{aligned}$$

where  $r_k = \|x_k - x^*\|$ .

6. So, we have:

$$r_{k+1} \leq \left\| [\nabla^2 f(x_k)]^{-1} \right\| \cdot \frac{r_k}{2} M \cdot r_k$$

and we need to bound the norm of the inverse hessian

## Convergence

7. Because of Hessian's Lipschitz continuity and symmetry:

## Convergence

7. Because of Hessian's Lipschitz continuity and symmetry:

$$\nabla^2 f(x_k) - \nabla^2 f(x^*) \succeq -Mr_k I_n$$

## Convergence

7. Because of Hessian's Lipschitz continuity and symmetry:

$$\nabla^2 f(x_k) - \nabla^2 f(x^*) \succeq -Mr_k I_n$$

$$\nabla^2 f(x_k) \succeq \nabla^2 f(x^*) - Mr_k I_n$$

## Convergence

7. Because of Hessian's Lipschitz continuity and symmetry:

$$\nabla^2 f(x_k) - \nabla^2 f(x^*) \succeq -Mr_k I_n$$

$$\nabla^2 f(x_k) \succeq \nabla^2 f(x^*) - Mr_k I_n$$

$$\nabla^2 f(x_k) \succeq \mu I_n - Mr_k I_n$$

## Convergence

7. Because of Hessian's Lipschitz continuity and symmetry:

$$\nabla^2 f(x_k) - \nabla^2 f(x^*) \succeq -Mr_k I_n$$

$$\nabla^2 f(x_k) \succeq \nabla^2 f(x^*) - Mr_k I_n$$

$$\nabla^2 f(x_k) \succeq \mu I_n - Mr_k I_n$$

$$\nabla^2 f(x_k) \succeq (\mu - Mr_k) I_n$$

## Convergence

7. Because of Hessian's Lipschitz continuity and symmetry:

$$\nabla^2 f(x_k) - \nabla^2 f(x^*) \succeq -Mr_k I_n$$

$$\nabla^2 f(x_k) \succeq \nabla^2 f(x^*) - Mr_k I_n$$

$$\nabla^2 f(x_k) \succeq \mu I_n - Mr_k I_n$$

$$\nabla^2 f(x_k) \succeq (\mu - Mr_k) I_n$$

Convexity implies  $\nabla^2 f(x_k) \succ 0$ , i.e.  $r_k < \frac{\mu}{M}$ .

$$\left\| [\nabla^2 f(x_k)]^{-1} \right\| \leq (\mu - Mr_k)^{-1}$$

$$r_{k+1} \leq \frac{r_k^2 M}{2(\mu - Mr_k)}$$

## Convergence

7. Because of Hessian's Lipschitz continuity and symmetry:

$$\nabla^2 f(x_k) - \nabla^2 f(x^*) \succeq -Mr_k I_n$$

$$\nabla^2 f(x_k) \succeq \nabla^2 f(x^*) - Mr_k I_n$$

$$\nabla^2 f(x_k) \succeq \mu I_n - Mr_k I_n$$

$$\nabla^2 f(x_k) \succeq (\mu - Mr_k) I_n$$

Convexity implies  $\nabla^2 f(x_k) \succ 0$ , i.e.  $r_k < \frac{\mu}{M}$ .

$$\left\| [\nabla^2 f(x_k)]^{-1} \right\| \leq (\mu - Mr_k)^{-1}$$

$$r_{k+1} \leq \frac{r_k^2 M}{2(\mu - Mr_k)}$$

8. The convergence condition  $r_{k+1} < r_k$  imposes additional conditions on  $r_k$ :  $r_k < \frac{2\mu}{3M}$

Thus, we have an important result: Newton's method for the function with Lipschitz positive-definite Hessian converges **quadratically** near ( $\|x_0 - x^*\| < \frac{2\mu}{3M}$ ) to the solution.

## Affine Invariance of Newton's Method

An important property of Newton's method is **affine invariance**. Given a function  $f$  and a nonsingular matrix  $A \in \mathbb{R}^{n \times n}$ , let  $x = Ay$ , and define  $g(y) = f(Ay)$ . Note, that  $\nabla g(y) = A^T \nabla f(x)$  and  $\nabla^2 g(y) = A^T \nabla^2 f(x)A$ . The Newton steps on  $g$  are expressed as:

$$y_{k+1} = y_k - (\nabla^2 g(y_k))^{-1} \nabla g(y_k)$$

## Affine Invariance of Newton's Method

An important property of Newton's method is **affine invariance**. Given a function  $f$  and a nonsingular matrix  $A \in \mathbb{R}^{n \times n}$ , let  $x = Ay$ , and define  $g(y) = f(Ay)$ . Note, that  $\nabla g(y) = A^T \nabla f(x)$  and  $\nabla^2 g(y) = A^T \nabla^2 f(x)A$ . The Newton steps on  $g$  are expressed as:

$$y_{k+1} = y_k - (\nabla^2 g(y_k))^{-1} \nabla g(y_k)$$

Expanding this, we get:

$$y_{k+1} = y_k - (A^T \nabla^2 f(Ay_k)A)^{-1} A^T \nabla f(Ay_k)$$

## Affine Invariance of Newton's Method

An important property of Newton's method is **affine invariance**. Given a function  $f$  and a nonsingular matrix  $A \in \mathbb{R}^{n \times n}$ , let  $x = Ay$ , and define  $g(y) = f(Ay)$ . Note, that  $\nabla g(y) = A^T \nabla f(x)$  and  $\nabla^2 g(y) = A^T \nabla^2 f(x)A$ . The Newton steps on  $g$  are expressed as:

$$y_{k+1} = y_k - (\nabla^2 g(y_k))^{-1} \nabla g(y_k)$$

Expanding this, we get:

$$y_{k+1} = y_k - (A^T \nabla^2 f(Ay_k) A)^{-1} A^T \nabla f(Ay_k)$$

Using the property of matrix inverse  $(AB)^{-1} = B^{-1}A^{-1}$ , this simplifies to:

$$\begin{aligned} y_{k+1} &= y_k - A^{-1} (\nabla^2 f(Ay_k))^{-1} \nabla f(Ay_k) \\ Ay_{k+1} &= Ay_k - (\nabla^2 f(Ay_k))^{-1} \nabla f(Ay_k) \end{aligned}$$

## Affine Invariance of Newton's Method

An important property of Newton's method is **affine invariance**. Given a function  $f$  and a nonsingular matrix  $A \in \mathbb{R}^{n \times n}$ , let  $x = Ay$ , and define  $g(y) = f(Ay)$ . Note, that  $\nabla g(y) = A^T \nabla f(x)$  and  $\nabla^2 g(y) = A^T \nabla^2 f(x)A$ . The Newton steps on  $g$  are expressed as:

$$y_{k+1} = y_k - (\nabla^2 g(y_k))^{-1} \nabla g(y_k)$$

Expanding this, we get:

$$y_{k+1} = y_k - (A^T \nabla^2 f(Ay_k) A)^{-1} A^T \nabla f(Ay_k)$$

Using the property of matrix inverse  $(AB)^{-1} = B^{-1}A^{-1}$ , this simplifies to:

$$\begin{aligned} y_{k+1} &= y_k - A^{-1} (\nabla^2 f(Ay_k))^{-1} \nabla f(Ay_k) \\ Ay_{k+1} &= Ay_k - (\nabla^2 f(Ay_k))^{-1} \nabla f(Ay_k) \end{aligned}$$

Thus, the update rule for  $x$  is:

$$x_{k+1} = x_k - (\nabla^2 f(x_k))^{-1} \nabla f(x_k)$$

## Affine Invariance of Newton's Method

An important property of Newton's method is **affine invariance**. Given a function  $f$  and a nonsingular matrix  $A \in \mathbb{R}^{n \times n}$ , let  $x = Ay$ , and define  $g(y) = f(Ay)$ . Note, that  $\nabla g(y) = A^T \nabla f(x)$  and  $\nabla^2 g(y) = A^T \nabla^2 f(x)A$ . The Newton steps on  $g$  are expressed as:

$$y_{k+1} = y_k - (\nabla^2 g(y_k))^{-1} \nabla g(y_k)$$

Expanding this, we get:

$$y_{k+1} = y_k - (A^T \nabla^2 f(Ay_k) A)^{-1} A^T \nabla f(Ay_k)$$

Using the property of matrix inverse  $(AB)^{-1} = B^{-1}A^{-1}$ , this simplifies to:

$$\begin{aligned} y_{k+1} &= y_k - A^{-1} (\nabla^2 f(Ay_k))^{-1} \nabla f(Ay_k) \\ Ay_{k+1} &= Ay_k - (\nabla^2 f(Ay_k))^{-1} \nabla f(Ay_k) \end{aligned}$$

Thus, the update rule for  $x$  is:

$$x_{k+1} = x_k - (\nabla^2 f(x_k))^{-1} \nabla f(x_k)$$

This shows that the progress made by Newton's method is independent of problem scaling. This property is not shared by the gradient descent method!

## Summary

What's nice:

- quadratic convergence near the solution  $x^*$

## Summary

What's nice:

- quadratic convergence near the solution  $x^*$
- affine invariance

## Summary

What's nice:

- quadratic convergence near the solution  $x^*$
- affine invariance
- the parameters have little effect on the convergence rate

## Summary

What's nice:

- quadratic convergence near the solution  $x^*$
- affine invariance
- the parameters have little effect on the convergence rate

## Summary

What's nice:

- quadratic convergence near the solution  $x^*$
- affine invariance
- the parameters have little effect on the convergence rate

What's not nice:

- it is necessary to store the (inverse) hessian on each iteration:  $\mathcal{O}(n^2)$  memory

## Summary

What's nice:

- quadratic convergence near the solution  $x^*$
- affine invariance
- the parameters have little effect on the convergence rate

What's not nice:

- it is necessary to store the (inverse) hessian on each iteration:  $\mathcal{O}(n^2)$  memory
- it is necessary to solve linear systems:  $\mathcal{O}(n^3)$  operations

## Summary

What's nice:

- quadratic convergence near the solution  $x^*$
- affine invariance
- the parameters have little effect on the convergence rate

What's not nice:

- it is necessary to store the (inverse) hessian on each iteration:  $\mathcal{O}(n^2)$  memory
- it is necessary to solve linear systems:  $\mathcal{O}(n^3)$  operations
- the Hessian can be degenerate at  $x^*$

# Summary

What's nice:

- quadratic convergence near the solution  $x^*$
- affine invariance
- the parameters have little effect on the convergence rate

What's not nice:

- it is necessary to store the (inverse) hessian on each iteration:  $\mathcal{O}(n^2)$  memory
- it is necessary to solve linear systems:  $\mathcal{O}(n^3)$  operations
- the Hessian can be degenerate at  $x^*$
- the hessian may not be positively determined → direction  $-(f''(x))^{-1}f'(x)$  may not be a descending direction

## Задача (метод Ньютона)

**i** Пусть  $f(x) = e^{x_1} + x_1^2 + x_2^2 - 2x_1 - x_2$ . Найдите  $x_1$  методом Ньютона, если  $x_0 = (0, 0)^T$ .

## Задача (метод Ньютона)

**i** Пусть  $f(x) = e^{x_1} + x_1^2 + x_2^2 - 2x_1 - x_2$ . Найдите  $x_1$  методом Ньютона, если  $x_0 = (0, 0)^T$ .

**Решение:**  $\nabla f(x) = (e^{x_1} + 2x_1 - 2, 2x_2 - 1)^T$ ,  $\nabla^2 f(x) = \begin{pmatrix} e^{x_1} + 2 & 0 \\ 0 & 2 \end{pmatrix}$

## Задача (метод Ньютона)

**i** Пусть  $f(x) = e^{x_1} + x_1^2 + x_2^2 - 2x_1 - x_2$ . Найдите  $x_1$  методом Ньютона, если  $x_0 = (0, 0)^T$ .

**Решение:**  $\nabla f(x) = (e^{x_1} + 2x_1 - 2, 2x_2 - 1)^T$ ,  $\nabla^2 f(x) = \begin{pmatrix} e^{x_1} + 2 & 0 \\ 0 & 2 \end{pmatrix}$

В точке  $x_0 = (0, 0)^T$ :  $\nabla f(x_0) = (-1, -1)^T$ ,  $\nabla^2 f(x_0) = \begin{pmatrix} 3 & 0 \\ 0 & 2 \end{pmatrix}$

## Задача (метод Ньютона)

**i** Пусть  $f(x) = e^{x_1} + x_1^2 + x_2^2 - 2x_1 - x_2$ . Найдите  $x_1$  методом Ньютона, если  $x_0 = (0, 0)^T$ .

**Решение:**  $\nabla f(x) = (e^{x_1} + 2x_1 - 2, 2x_2 - 1)^T$ ,  $\nabla^2 f(x) = \begin{pmatrix} e^{x_1} + 2 & 0 \\ 0 & 2 \end{pmatrix}$

В точке  $x_0 = (0, 0)^T$ :  $\nabla f(x_0) = (-1, -1)^T$ ,  $\nabla^2 f(x_0) = \begin{pmatrix} 3 & 0 \\ 0 & 2 \end{pmatrix}$

$$x_1 = x_0 - [\nabla^2 f(x_0)]^{-1} \nabla f(x_0) = \begin{pmatrix} 1/3 \\ 1/2 \end{pmatrix}$$

## Newton method problems

# Newton

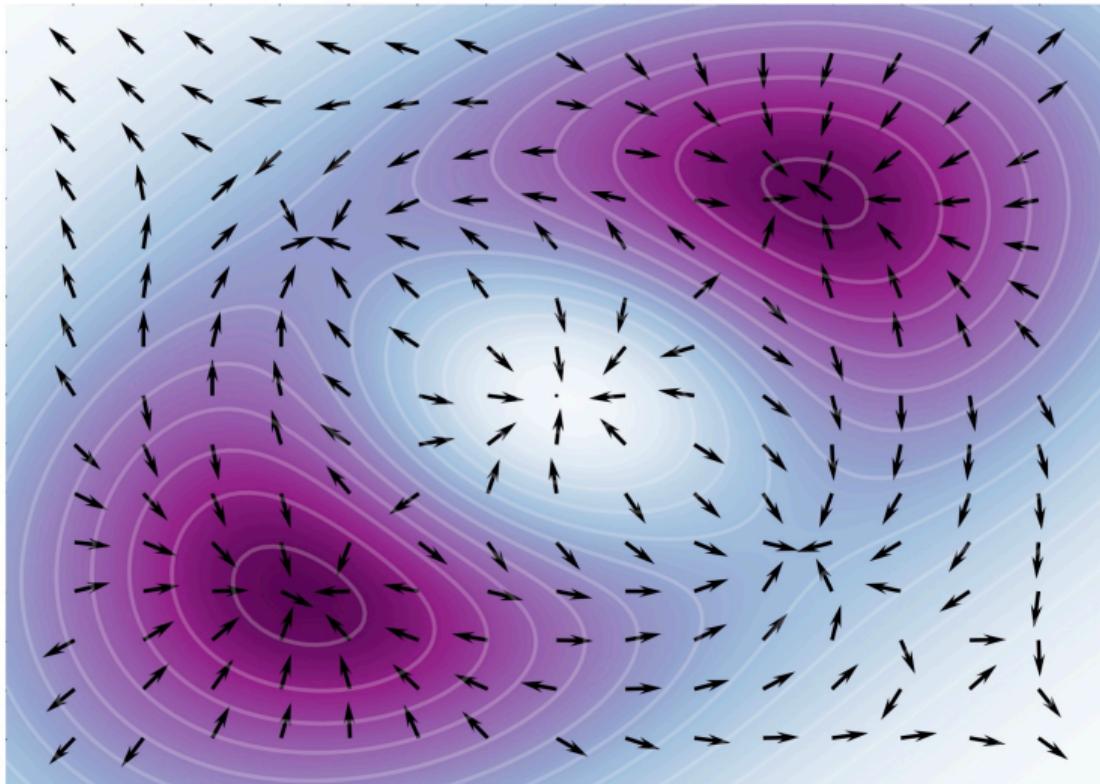


Рис. 7: Animation

## Newton method problems

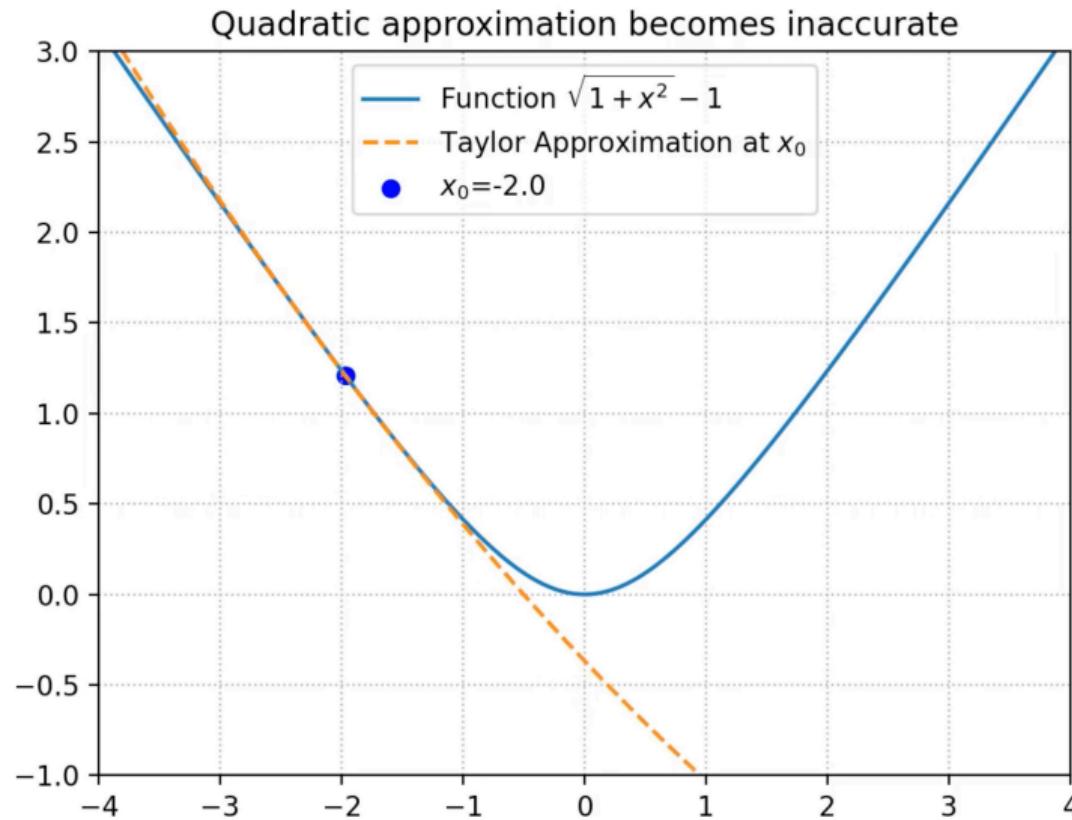


Рис. 8: Animation

## The idea of adaptive metrics

Given  $f(x)$  and a point  $x_0$ . Define

$B_\varepsilon(x_0) = \{x \in \mathbb{R}^n : d(x, x_0) = \varepsilon^2\}$  as the set of points with distance  $\varepsilon$  to  $x_0$ . Here we presume the existence of a distance function  $d(x, x_0)$ .

## The idea of adaptive metrics

Given  $f(x)$  and a point  $x_0$ . Define

$B_\varepsilon(x_0) = \{x \in \mathbb{R}^n : d(x, x_0) = \varepsilon^2\}$  as the set of points with distance  $\varepsilon$  to  $x_0$ . Here we presume the existence of a distance function  $d(x, x_0)$ .

$$x^* = \arg \min_{x \in B_\varepsilon(x_0)} f(x)$$

## The idea of adaptive metrics

Given  $f(x)$  and a point  $x_0$ . Define

$B_\varepsilon(x_0) = \{x \in \mathbb{R}^n : d(x, x_0) = \varepsilon^2\}$  as the set of points with distance  $\varepsilon$  to  $x_0$ . Here we presume the existence of a distance function  $d(x, x_0)$ .

$$x^* = \arg \min_{x \in B_\varepsilon(x_0)} f(x)$$

Then, we can define another *steepest descent* direction in terms of minimizer of function on a sphere:

## The idea of adaptive metrics

Given  $f(x)$  and a point  $x_0$ . Define

$B_\varepsilon(x_0) = \{x \in \mathbb{R}^n : d(x, x_0) = \varepsilon^2\}$  as the set of points with distance  $\varepsilon$  to  $x_0$ . Here we presume the existence of a distance function  $d(x, x_0)$ .

$$x^* = \arg \min_{x \in B_\varepsilon(x_0)} f(x)$$

Then, we can define another *steepest descent* direction in terms of minimizer of function on a sphere:

$$s = \lim_{\varepsilon \rightarrow 0} \frac{x^* - x_0}{\varepsilon}$$

## The idea of adaptive metrics

Given  $f(x)$  and a point  $x_0$ . Define

$B_\varepsilon(x_0) = \{x \in \mathbb{R}^n : d(x, x_0) = \varepsilon\}$  as the set of points with distance  $\varepsilon$  to  $x_0$ . Here we presume the existence of a distance function  $d(x, x_0)$ .

$$x^* = \arg \min_{x \in B_\varepsilon(x_0)} f(x)$$

Then, we can define another *steepest descent* direction in terms of minimizer of function on a sphere:

$$s = \lim_{\varepsilon \rightarrow 0} \frac{x^* - x_0}{\varepsilon}$$

Let us assume that the distance is defined locally by some metric  $A$ :

$$d(x, x_0) = (x - x_0)^\top A(x - x_0)$$

## The idea of adaptive metrics

Given  $f(x)$  and a point  $x_0$ . Define  $B_\varepsilon(x_0) = \{x \in \mathbb{R}^n : d(x, x_0) = \varepsilon\}$  as the set of points with distance  $\varepsilon$  to  $x_0$ . Here we presume the existence of a distance function  $d(x, x_0)$ .

$$x^* = \arg \min_{x \in B_\varepsilon(x_0)} f(x)$$

Then, we can define another *steepest descent* direction in terms of minimizer of function on a sphere:

$$s = \lim_{\varepsilon \rightarrow 0} \frac{x^* - x_0}{\varepsilon}$$

Let us assume that the distance is defined locally by some metric  $A$ :

$$d(x, x_0) = (x - x_0)^\top A(x - x_0)$$

Let us also consider first order Taylor approximation of a function  $f(x)$  near the point  $x_0$ :

$$f(x_0 + \delta x) \approx f(x_0) + \nabla f(x_0)^\top \delta x \quad (1)$$

Now we can explicitly pose a problem of finding  $s$ , as it was stated above.

$$\begin{aligned} & \min_{\delta x \in \mathbb{R}^n} f(x_0 + \delta x) \\ \text{s.t. } & \delta x^\top A \delta x = \varepsilon^2 \end{aligned}$$

## The idea of adaptive metrics

Given  $f(x)$  and a point  $x_0$ . Define  $B_\varepsilon(x_0) = \{x \in \mathbb{R}^n : d(x, x_0) = \varepsilon\}$  as the set of points with distance  $\varepsilon$  to  $x_0$ . Here we presume the existence of a distance function  $d(x, x_0)$ .

$$x^* = \arg \min_{x \in B_\varepsilon(x_0)} f(x)$$

Then, we can define another *steepest descent* direction in terms of minimizer of function on a sphere:

$$s = \lim_{\varepsilon \rightarrow 0} \frac{x^* - x_0}{\varepsilon}$$

Let us assume that the distance is defined locally by some metric  $A$ :

$$d(x, x_0) = (x - x_0)^\top A(x - x_0)$$

Let us also consider first order Taylor approximation of a function  $f(x)$  near the point  $x_0$ :

$$f(x_0 + \delta x) \approx f(x_0) + \nabla f(x_0)^\top \delta x \quad (1)$$

Now we can explicitly pose a problem of finding  $s$ , as it was stated above.

$$\begin{aligned} & \min_{\delta x \in \mathbb{R}^n} f(x_0 + \delta x) \\ \text{s.t. } & \delta x^\top A \delta x = \varepsilon^2 \end{aligned}$$

Using equation (1) it can be written as:

$$\begin{aligned} & \min_{\delta x \in \mathbb{R}^n} \nabla f(x_0)^\top \delta x \\ \text{s.t. } & \delta x^\top A \delta x = \varepsilon^2 \end{aligned}$$

## The idea of adaptive metrics

Given  $f(x)$  and a point  $x_0$ . Define  $B_\varepsilon(x_0) = \{x \in \mathbb{R}^n : d(x, x_0) = \varepsilon\}$  as the set of points with distance  $\varepsilon$  to  $x_0$ . Here we presume the existence of a distance function  $d(x, x_0)$ .

$$x^* = \arg \min_{x \in B_\varepsilon(x_0)} f(x)$$

Then, we can define another *steepest descent* direction in terms of minimizer of function on a sphere:

$$s = \lim_{\varepsilon \rightarrow 0} \frac{x^* - x_0}{\varepsilon}$$

Let us assume that the distance is defined locally by some metric  $A$ :

$$d(x, x_0) = (x - x_0)^\top A(x - x_0)$$

Let us also consider first order Taylor approximation of a function  $f(x)$  near the point  $x_0$ :

$$f(x_0 + \delta x) \approx f(x_0) + \nabla f(x_0)^\top \delta x \quad (1)$$

Now we can explicitly pose a problem of finding  $s$ , as it was stated above.

$$\begin{aligned} & \min_{\delta x \in \mathbb{R}^n} f(x_0 + \delta x) \\ \text{s.t. } & \delta x^\top A \delta x = \varepsilon^2 \end{aligned}$$

Using equation (1) it can be written as:

$$\begin{aligned} & \min_{\delta x \in \mathbb{R}^n} \nabla f(x_0)^\top \delta x \\ \text{s.t. } & \delta x^\top A \delta x = \varepsilon^2 \end{aligned}$$

Using Lagrange multipliers method, we can easily conclude, that the answer is:

$$\delta x = -\frac{2\varepsilon^2}{\nabla f(x_0)^\top A^{-1} \nabla f(x_0)} A^{-1} \nabla f$$

## The idea of adaptive metrics

Given  $f(x)$  and a point  $x_0$ . Define  $B_\varepsilon(x_0) = \{x \in \mathbb{R}^n : d(x, x_0) = \varepsilon\}$  as the set of points with distance  $\varepsilon$  to  $x_0$ . Here we presume the existence of a distance function  $d(x, x_0)$ .

$$x^* = \arg \min_{x \in B_\varepsilon(x_0)} f(x)$$

Then, we can define another *steepest descent* direction in terms of minimizer of function on a sphere:

$$s = \lim_{\varepsilon \rightarrow 0} \frac{x^* - x_0}{\varepsilon}$$

Let us assume that the distance is defined locally by some metric  $A$ :

$$d(x, x_0) = (x - x_0)^\top A(x - x_0)$$

Let us also consider first order Taylor approximation of a function  $f(x)$  near the point  $x_0$ :

$$f(x_0 + \delta x) \approx f(x_0) + \nabla f(x_0)^\top \delta x$$

Now we can explicitly pose a problem of finding  $s$ , as it was stated above.

$$\begin{aligned} & \min_{\delta x \in \mathbb{R}^n} f(x_0 + \delta x) \\ \text{s.t. } & \delta x^\top A \delta x = \varepsilon^2 \end{aligned}$$

Using equation (1) it can be written as:

$$\begin{aligned} & \min_{\delta x \in \mathbb{R}^n} \nabla f(x_0)^\top \delta x \\ \text{s.t. } & \delta x^\top A \delta x = \varepsilon^2 \end{aligned}$$

Using Lagrange multipliers method, we can easily conclude, that the answer is:

$$\delta x = -\frac{2\varepsilon^2}{\nabla f(x_0)^\top A^{-1} \nabla f(x_0)} A^{-1} \nabla f$$

Which means, that new direction of steepest descent is nothing else, but  $A^{-1} \nabla f(x_0)$ .

Indeed, if the space is isotropic and  $A = I$ , we immediately have gradient descent formula, while Newton method uses local Hessian as a metric matrix.

## Quasi-Newton methods

## Quasi-Newton methods intuition

For the classic task of unconditional optimization  $f(x) \rightarrow \min_{x \in \mathbb{R}^n}$  the general scheme of iteration method is written as:

$$x_{k+1} = x_k + \alpha_k d_k$$

## Quasi-Newton methods intuition

For the classic task of unconditional optimization  $f(x) \rightarrow \min_{x \in \mathbb{R}^n}$  the general scheme of iteration method is written as:

$$x_{k+1} = x_k + \alpha_k d_k$$

In the Newton method, the  $d_k$  direction (Newton's direction) is set by the linear system solution at each step:

$$B_k d_k = -\nabla f(x_k), \quad B_k = \nabla^2 f(x_k)$$

## Quasi-Newton methods intuition

For the classic task of unconditional optimization  $f(x) \rightarrow \min_{x \in \mathbb{R}^n}$  the general scheme of iteration method is written as:

$$x_{k+1} = x_k + \alpha_k d_k$$

In the Newton method, the  $d_k$  direction (Newton's direction) is set by the linear system solution at each step:

$$B_k d_k = -\nabla f(x_k), \quad B_k = \nabla^2 f(x_k)$$

i.e. at each iteration it is necessary to **compute** hessian and gradient and **solve** linear system.

## Quasi-Newton methods intuition

For the classic task of unconditional optimization  $f(x) \rightarrow \min_{x \in \mathbb{R}^n}$  the general scheme of iteration method is written as:

$$x_{k+1} = x_k + \alpha_k d_k$$

In the Newton method, the  $d_k$  direction (Newton's direction) is set by the linear system solution at each step:

$$B_k d_k = -\nabla f(x_k), \quad B_k = \nabla^2 f(x_k)$$

i.e. at each iteration it is necessary to **compute** hessian and gradient and **solve** linear system.

Note here that if we take a single matrix of  $B_k = I_n$  as  $B_k$  at each step, we will exactly get the gradient descent method.

The general scheme of quasi-Newton methods is based on the selection of the  $B_k$  matrix so that it tends in some sense at  $k \rightarrow \infty$  to the truth value of the Hessian  $\nabla^2 f(x_k)$ .

## Quasi-Newton Method Template

Let  $x_0 \in \mathbb{R}^n$ ,  $B_0 \succ 0$ . For  $k = 1, 2, 3, \dots$ , repeat:

1. Solve  $B_k d_k = -\nabla f(x_k)$

## Quasi-Newton Method Template

Let  $x_0 \in \mathbb{R}^n$ ,  $B_0 \succ 0$ . For  $k = 1, 2, 3, \dots$ , repeat:

1. Solve  $B_k d_k = -\nabla f(x_k)$
2. Update  $x_{k+1} = x_k + \alpha_k d_k$

## Quasi-Newton Method Template

Let  $x_0 \in \mathbb{R}^n$ ,  $B_0 \succ 0$ . For  $k = 1, 2, 3, \dots$ , repeat:

1. Solve  $B_k d_k = -\nabla f(x_k)$
2. Update  $x_{k+1} = x_k + \alpha_k d_k$
3. Compute  $B_{k+1}$  from  $B_k$

## Quasi-Newton Method Template

Let  $x_0 \in \mathbb{R}^n$ ,  $B_0 \succ 0$ . For  $k = 1, 2, 3, \dots$ , repeat:

1. Solve  $B_k d_k = -\nabla f(x_k)$
2. Update  $x_{k+1} = x_k + \alpha_k d_k$
3. Compute  $B_{k+1}$  from  $B_k$

## Quasi-Newton Method Template

Let  $x_0 \in \mathbb{R}^n$ ,  $B_0 \succ 0$ . For  $k = 1, 2, 3, \dots$ , repeat:

1. Solve  $B_k d_k = -\nabla f(x_k)$
2. Update  $x_{k+1} = x_k + \alpha_k d_k$
3. Compute  $B_{k+1}$  from  $B_k$

Different quasi-Newton methods implement Step 3 differently. As we will see, commonly we can compute  $(B_{k+1})^{-1}$  from  $(B_k)^{-1}$ .

## Quasi-Newton Method Template

Let  $x_0 \in \mathbb{R}^n$ ,  $B_0 \succ 0$ . For  $k = 1, 2, 3, \dots$ , repeat:

1. Solve  $B_k d_k = -\nabla f(x_k)$
2. Update  $x_{k+1} = x_k + \alpha_k d_k$
3. Compute  $B_{k+1}$  from  $B_k$

Different quasi-Newton methods implement Step 3 differently. As we will see, commonly we can compute  $(B_{k+1})^{-1}$  from  $(B_k)^{-1}$ .

**Basic Idea:** As  $B_k$  already contains information about the Hessian, use a suitable matrix update to form  $B_{k+1}$ .

## Quasi-Newton Method Template

Let  $x_0 \in \mathbb{R}^n$ ,  $B_0 \succ 0$ . For  $k = 1, 2, 3, \dots$ , repeat:

1. Solve  $B_k d_k = -\nabla f(x_k)$
2. Update  $x_{k+1} = x_k + \alpha_k d_k$
3. Compute  $B_{k+1}$  from  $B_k$

Different quasi-Newton methods implement Step 3 differently. As we will see, commonly we can compute  $(B_{k+1})^{-1}$  from  $(B_k)^{-1}$ .

**Basic Idea:** As  $B_k$  already contains information about the Hessian, use a suitable matrix update to form  $B_{k+1}$ .

**Reasonable Requirement for  $B_{k+1}$**  (motivated by the secant method):

$$\begin{aligned}\nabla f(x_{k+1}) - \nabla f(x_k) &= B_{k+1}(x_{k+1} - x_k) = B_{k+1}d_k \\ \Delta y_k &= B_{k+1}\Delta x_k\end{aligned}$$

## Quasi-Newton Method Template

Let  $x_0 \in \mathbb{R}^n$ ,  $B_0 \succ 0$ . For  $k = 1, 2, 3, \dots$ , repeat:

1. Solve  $B_k d_k = -\nabla f(x_k)$
2. Update  $x_{k+1} = x_k + \alpha_k d_k$
3. Compute  $B_{k+1}$  from  $B_k$

Different quasi-Newton methods implement Step 3 differently. As we will see, commonly we can compute  $(B_{k+1})^{-1}$  from  $(B_k)^{-1}$ .

**Basic Idea:** As  $B_k$  already contains information about the Hessian, use a suitable matrix update to form  $B_{k+1}$ .

**Reasonable Requirement for  $B_{k+1}$**  (motivated by the secant method):

$$\begin{aligned}\nabla f(x_{k+1}) - \nabla f(x_k) &= B_{k+1}(x_{k+1} - x_k) = B_{k+1}d_k \\ \Delta y_k &= B_{k+1}\Delta x_k\end{aligned}$$

In addition to the secant equation, we want:

- $B_{k+1}$  to be symmetric

## Quasi-Newton Method Template

Let  $x_0 \in \mathbb{R}^n$ ,  $B_0 \succ 0$ . For  $k = 1, 2, 3, \dots$ , repeat:

1. Solve  $B_k d_k = -\nabla f(x_k)$
2. Update  $x_{k+1} = x_k + \alpha_k d_k$
3. Compute  $B_{k+1}$  from  $B_k$

Different quasi-Newton methods implement Step 3 differently. As we will see, commonly we can compute  $(B_{k+1})^{-1}$  from  $(B_k)^{-1}$ .

**Basic Idea:** As  $B_k$  already contains information about the Hessian, use a suitable matrix update to form  $B_{k+1}$ .

**Reasonable Requirement for  $B_{k+1}$**  (motivated by the secant method):

$$\begin{aligned}\nabla f(x_{k+1}) - \nabla f(x_k) &= B_{k+1}(x_{k+1} - x_k) = B_{k+1}d_k \\ \Delta y_k &= B_{k+1}\Delta x_k\end{aligned}$$

In addition to the secant equation, we want:

- $B_{k+1}$  to be symmetric
- $B_{k+1}$  to be “close” to  $B_k$

## Quasi-Newton Method Template

Let  $x_0 \in \mathbb{R}^n$ ,  $B_0 \succ 0$ . For  $k = 1, 2, 3, \dots$ , repeat:

1. Solve  $B_k d_k = -\nabla f(x_k)$
2. Update  $x_{k+1} = x_k + \alpha_k d_k$
3. Compute  $B_{k+1}$  from  $B_k$

Different quasi-Newton methods implement Step 3 differently. As we will see, commonly we can compute  $(B_{k+1})^{-1}$  from  $(B_k)^{-1}$ .

**Basic Idea:** As  $B_k$  already contains information about the Hessian, use a suitable matrix update to form  $B_{k+1}$ .

**Reasonable Requirement for  $B_{k+1}$**  (motivated by the secant method):

$$\begin{aligned}\nabla f(x_{k+1}) - \nabla f(x_k) &= B_{k+1}(x_{k+1} - x_k) = B_{k+1}d_k \\ \Delta y_k &= B_{k+1}\Delta x_k\end{aligned}$$

In addition to the secant equation, we want:

- $B_{k+1}$  to be symmetric
- $B_{k+1}$  to be “close” to  $B_k$
- $B_k \succ 0 \Rightarrow B_{k+1} \succ 0$

## Symmetric Rank-One Update

Let's try an update of the form:

$$B_{k+1} = B_k + auu^T$$

## Symmetric Rank-One Update

Let's try an update of the form:

$$B_{k+1} = B_k + auu^T$$

The secant equation  $B_{k+1}d_k = \Delta y_k$  yields:

$$(au^T d_k)u = \Delta y_k - B_k d_k$$

## Symmetric Rank-One Update

Let's try an update of the form:

$$B_{k+1} = B_k + auu^T$$

The secant equation  $B_{k+1}d_k = \Delta y_k$  yields:

$$(au^T d_k)u = \Delta y_k - B_k d_k$$

This only holds if  $u$  is a multiple of  $\Delta y_k - B_k d_k$ . Putting  $u = \Delta y_k - B_k d_k$ , we solve the above,

$$a = \frac{1}{(\Delta y_k - B_k d_k)^T d_k},$$

## Symmetric Rank-One Update

Let's try an update of the form:

$$B_{k+1} = B_k + auu^T$$

The secant equation  $B_{k+1}d_k = \Delta y_k$  yields:

$$(au^T d_k)u = \Delta y_k - B_k d_k$$

This only holds if  $u$  is a multiple of  $\Delta y_k - B_k d_k$ . Putting  $u = \Delta y_k - B_k d_k$ , we solve the above,

$$a = \frac{1}{(\Delta y_k - B_k d_k)^T d_k},$$

which leads to

$$B_{k+1} = B_k + \frac{(\Delta y_k - B_k d_k)(\Delta y_k - B_k d_k)^T}{(\Delta y_k - B_k d_k)^T d_k}$$

called the symmetric rank-one (SR1) update or Broyden method.

## Symmetric Rank-One Update with inverse

How can we solve

$$B_{k+1}d_{k+1} = -\nabla f(x_{k+1}),$$

in order to take the next step? In addition to propagating  $B_k$  to  $B_{k+1}$ , let's propagate inverses, i.e.,  $C_k = B_k^{-1}$  to  $C_{k+1} = (B_{k+1})^{-1}$ .

### Sherman-Morrison Formula:

The Sherman-Morrison formula states:

$$(A + uv^T)^{-1} = A^{-1} - \frac{A^{-1}uv^TA^{-1}}{1 + v^TA^{-1}u}$$

Thus, for the SR1 update, the inverse is also easily updated:

$$C_{k+1} = C_k + \frac{(d_k - C_k \Delta y_k)(d_k - C_k \Delta y_k)^T}{(d_k - C_k \Delta y_k)^T \Delta y_k}$$

In general, SR1 is simple and cheap, but it has a key shortcoming: it does not preserve positive definiteness.

## Davidon-Fletcher-Powell Update

We could have pursued the same idea to update the inverse  $C$ :

$$C_{k+1} = C_k + auu^T + bvv^T.$$

## Davidon-Fletcher-Powell Update

We could have pursued the same idea to update the inverse  $C$ :

$$C_{k+1} = C_k + auu^T + bvv^T.$$

Multiplying by  $\Delta y_k$ , using the secant equation  $d_k = C_k \Delta y_k$ , and solving for  $a, b$ , yields:

$$C_{k+1} = C_k - \frac{C_k \Delta y_k \Delta y_k^T C_k}{\Delta y_k^T C_k \Delta y_k} + \frac{d_k d_k^T}{\Delta y_k^T d_k}$$

## Woodbury Formula Application

Woodbury then shows:

$$B_{k+1} = \left( I - \frac{\Delta y_k d_k^T}{\Delta y_k^T d_k} \right) B_k \left( I - \frac{d_k \Delta y_k^T}{\Delta y_k^T d_k} \right) + \frac{\Delta y_k \Delta y_k^T}{\Delta y_k^T d_k}$$

This is the Davidon-Fletcher-Powell (DFP) update. Also cheap:  $O(n^2)$ , preserves positive definiteness. Not as popular as BFGS.

## Broyden-Fletcher-Goldfarb-Shanno update

Let's now try a rank-two update:

$$B_{k+1} = B_k + auu^T + bvv^T.$$

## Broyden-Fletcher-Goldfarb-Shanno update

Let's now try a rank-two update:

$$B_{k+1} = B_k + auu^T + bvv^T.$$

The secant equation  $\Delta y_k = B_{k+1}d_k$  yields:

$$\Delta y_k - B_k d_k = (au^T d_k)u + (bv^T d_k)v$$

## Broyden-Fletcher-Goldfarb-Shanno update

Let's now try a rank-two update:

$$B_{k+1} = B_k + auu^T + bvv^T.$$

The secant equation  $\Delta y_k = B_{k+1}d_k$  yields:

$$\Delta y_k - B_k d_k = (au^T d_k)u + (bv^T d_k)v$$

Putting  $u = \Delta y_k$ ,  $v = B_k d_k$ , and solving for a, b we get:

$$B_{k+1} = B_k - \frac{B_k d_k d_k^T B_k}{d_k^T B_k d_k} + \frac{\Delta y_k \Delta y_k^T}{d_k^T \Delta y_k}$$

called the Broyden-Fletcher-Goldfarb-Shanno (BFGS) update.

## Broyden-Fletcher-Goldfarb-Shanno update with inverse

### Woodbury Formula

The Woodbury formula, a generalization of the Sherman-Morrison formula, is given by:

$$(A + UCV)^{-1} = A^{-1} - A^{-1}U(C^{-1} + VA^{-1}U)^{-1}VA^{-1}$$

## Broyden-Fletcher-Goldfarb-Shanno update with inverse

### Woodbury Formula

The Woodbury formula, a generalization of the Sherman-Morrison formula, is given by:

$$(A + UCV)^{-1} = A^{-1} - A^{-1}U(C^{-1} + VA^{-1}U)^{-1}VA^{-1}$$

Applied to our case, we get a rank-two update on the inverse  $C$ :

$$C_{k+1} = C_k + \frac{(d_k - C_k \Delta y_k) d_k^T}{\Delta y_k^T d_k} + \frac{d_k (d_k - C_k \Delta y_k)^T}{\Delta y_k^T d_k} - \frac{(d_k - C_k \Delta y_k)^T \Delta y_k}{(\Delta y_k^T d_k)^2} d_k d_k^T$$

$$C_{k+1} = \left( I - \frac{d_k \Delta y_k^T}{\Delta y_k^T d_k} \right) C_k \left( I - \frac{\Delta y_k d_k^T}{\Delta y_k^T d_k} \right) + \frac{d_k d_k^T}{\Delta y_k^T d_k}$$

This formulation ensures that the BFGS update, while comprehensive, remains computationally efficient, requiring  $O(n^2)$  operations. Importantly, BFGS update preserves positive definiteness. Recall this means  $B_k \succ 0 \Rightarrow B_{k+1} \succ 0$ . Equivalently,  $C_k \succ 0 \Rightarrow C_{k+1} \succ 0$

## Code

- Open In Colab

## Code

- Open In Colab
- Comparison of quasi Newton methods

## Code

- Open In Colab
- Comparison of quasi Newton methods
- Some practical notes about Newton method

## Задача

Рассмотрим квадратичную функцию

$$f(x) = 4x_1^2 + 2x_2^2, \quad x = (x_1, x_2)^\top \in \mathbb{R}^2.$$

Найдите константу сильной выпуклости  $\mu$  и константу гладкости  $L$  функции  $f$ .

## Задача

Рассмотрим квадратичную функцию  $f(x) = \lambda x^2$  (где  $\lambda > 0$ ,  $x \in \mathbb{R}$ ). Ускоренный градиентный метод Нестерова задаётся итерацией

$$x_{k+1} = y_k - \alpha \nabla f(y_k),$$
$$y_{k+1} = x_{k+1} + \beta (x_{k+1} - x_k),$$

где  $\alpha > 0$  — шаг,  $\beta \in \mathbb{R}$  — параметр «инерции».

1. Выпишите итерацию метода в виде умножения вектора состояния на предыдущей итерации на матрицу

$$\begin{pmatrix} y_{k+1} \\ x_{k+1} \end{pmatrix} = T \begin{pmatrix} y_k \\ x_k \end{pmatrix},$$

явно указав элементы матрицы  $T$  через  $\lambda$ ,  $\alpha$  и  $\beta$ .

## Задача

Рассмотрим квадратичную функцию  $f(x) = \lambda x^2$  (где  $\lambda > 0$ ,  $x \in \mathbb{R}$ ). Ускоренный градиентный метод Нестерова задаётся итерацией

$$x_{k+1} = y_k - \alpha \nabla f(y_k),$$
$$y_{k+1} = x_{k+1} + \beta (x_{k+1} - x_k),$$

где  $\alpha > 0$  — шаг,  $\beta \in \mathbb{R}$  — параметр «инерции».

1. Выпишите итерацию метода в виде умножения вектора состояния на предыдущей итерации на матрицу

$$\begin{pmatrix} y_{k+1} \\ x_{k+1} \end{pmatrix} = T \begin{pmatrix} y_k \\ x_k \end{pmatrix},$$

явно указав элементы матрицы  $T$  через  $\lambda$ ,  $\alpha$  и  $\beta$ .

2. Запишите явно вид матрицы  $T$  при  $\alpha = \frac{1}{\lambda}$ . Запишите характеристический полином для поиска собственных чисел матрицы  $T$  в этом случае  $p(\lambda) = \det(T - \lambda I)$ .

## Задача

Рассмотрим квадратичную функцию  $f(x) = \lambda x^2$  (где  $\lambda > 0$ ,  $x \in \mathbb{R}$ ). Ускоренный градиентный метод Нестерова задаётся итерацией

$$\begin{aligned}x_{k+1} &= y_k - \alpha \nabla f(y_k), \\y_{k+1} &= x_{k+1} + \beta (x_{k+1} - x_k),\end{aligned}$$

где  $\alpha > 0$  — шаг,  $\beta \in \mathbb{R}$  — параметр «инерции».

1. Выпишите итерацию метода в виде умножения вектора состояния на предыдущей итерации на матрицу

$$\begin{pmatrix} y_{k+1} \\ x_{k+1} \end{pmatrix} = T \begin{pmatrix} y_k \\ x_k \end{pmatrix},$$

явно указав элементы матрицы  $T$  через  $\lambda$ ,  $\alpha$  и  $\beta$ .

2. Запишите явно вид матрицы  $T$  при  $\alpha = \frac{1}{\lambda}$ . Запишите характеристический полином для поиска собственных чисел матрицы  $T$  в этом случае  $p(\lambda) = \det(T - \lambda I)$ .
3. Найдите значения  $\beta$ , при которых метод будет сходиться с этим шагом.\ Примечание: Для сходимости итерационного процесса нужно, чтобы оба собственных числа по модулю были  $< 1$  (спектральный радиус  $\rho(T) < 1$ ). Можно это сделать, найдя  $\lambda_1(\beta), \lambda_2(\beta)$  явно и найти такие  $\beta$ , чтобы  $|\lambda_1(\beta)| < 1, |\lambda_2(\beta)| < 1$ . Мы предлагаем воспользоваться готовым результатом, который называется критерий Жюри для полинома второго порядка. Пусть характеристический многочлен имеет вид  $p(\lambda) = \lambda^2 + a_1\lambda + a_2$ . Тогда спектральный радиус будет меньше единицы тогда и только тогда, когда:

$$|a_2| < 1 \quad 1 + a_1 + a_2 > 0 \quad 1 - a_1 + a_2 > 0$$



## Задача

Пусть  $A \in \mathbb{R}^{n \times n}$  - симметричная положительно определенная матрица,  $b \in \mathbb{R}^n$ . Найдите минимум функции:

$$f(x, y) = \frac{1}{2}x^T Ax + b^T x + \frac{1}{2}\|y\|_2^2 \rightarrow \min_{x+y=c}$$

где  $c \in \mathbb{R}^n$  - заданный вектор. Выпишите явно решение  $(x^*, y^*)$ .

## Задача

Упростите выражение:

$$\operatorname{tr}((A + \lambda I)^{-1} A), \quad \text{где } A \in \mathbb{S}_{++}^n, \lambda > 0$$

Выразите ответ через собственные значения матрицы  $A$ .

## Задача

Предложите метод решения задачи наименьших квадратов с регуляризацией:

$$\min_{x \in \mathbb{R}^n} \|Ax - b\|_2^2 + \lambda \|x\|_2^2$$

где  $A \in \mathbb{R}^{m \times n}$  (не обязательно квадратная),  $\lambda > 0$ . Опишите один итерационный метод оптимизации для её решения и приведите оценку скорости сходимости.

## Задача

Докажите, что функция

$$f(x) = \log \left( \sum_{i=1}^n e^{x_i} \right)$$

выпуклая, используя любой дифференциальный критерий выпуклости.

## Задача

Пусть  $X \in \mathbb{R}^{m \times n}$ , где  $\text{rk}X = n$ ,  $\Omega \in \mathbb{S}_{++}^m$ , и  $W \in \mathbb{R}^{k \times n}$ . Найдите матрицу  $G \in \mathbb{R}^{k \times m}$ , являющуюся решением следующей задачи оптимизации:

$$f(G) = \text{tr}(G\Omega G^\top) \rightarrow \min_{GX=W}$$