Концепция методов адаптивной метрики. Метод Ньютона. Квазиньютоновские методы

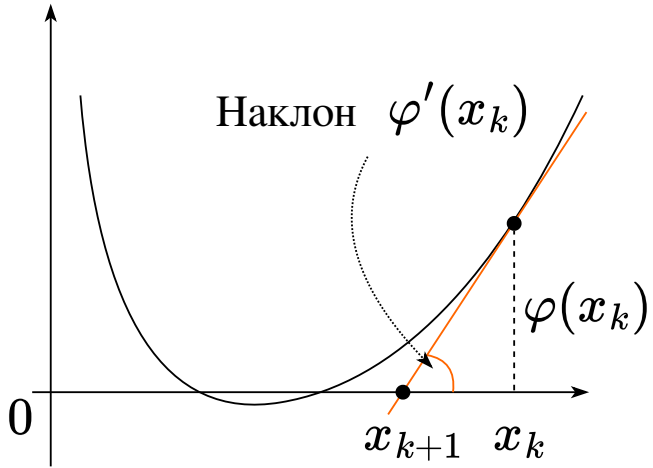
Даниил Меркулов

Методы оптимизации. МФТИ

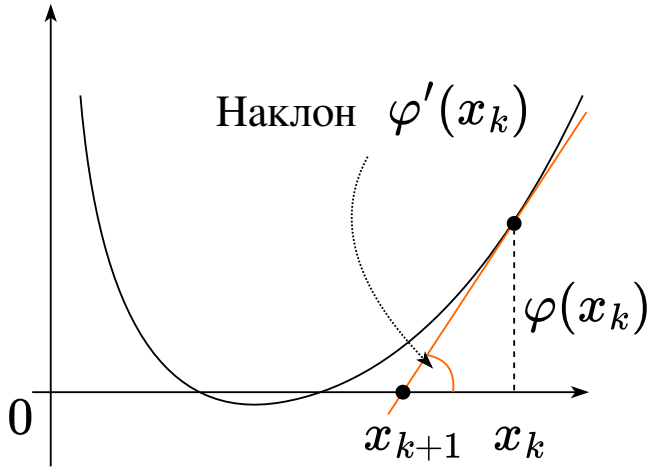
Метод Ньютона

Идея метода Ньютона для нахождения корней функции

Рассмотрим функцию $\varphi(x) : \mathbb{R} \to \mathbb{R}$.

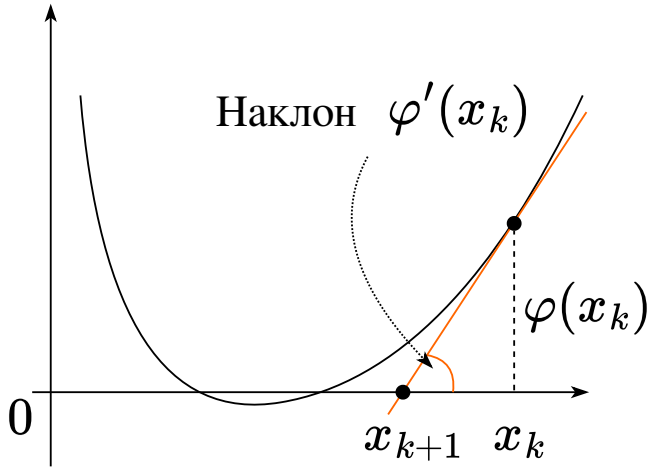


Идея метода Ньютона для нахождения корней функции



Рассмотрим функцию $\varphi(x) : \mathbb{R} \to \mathbb{R}$. Основная идея заключается в том, чтобы построить линейное приближение в точке x_k и найти его корень, который будет новой точкой итерации:

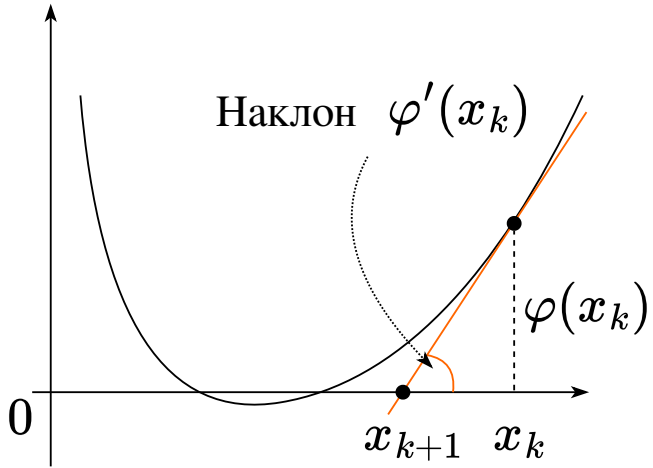
Идея метода Ньютона для нахождения корней функции



Рассмотрим функцию $\varphi(x) : \mathbb{R} \to \mathbb{R}$.
Основная идея заключается в том, чтобы построить линейное приближение в точке x_k и найти его корень, который будет новой точкой итерации:

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

Идея метода Ньютона для нахождения корней функции

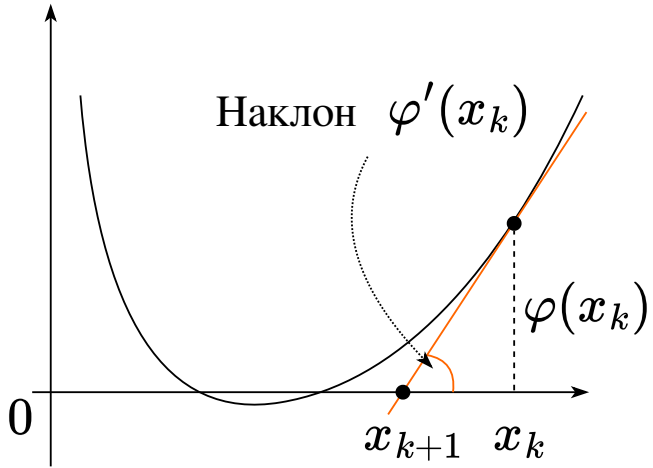


Рассмотрим функцию $\varphi(x) : \mathbb{R} \to \mathbb{R}$.
Основная идея заключается в том, чтобы построить линейное приближение в точке x_k и найти его корень, который будет новой точкой итерации:

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

Мы получаем итерационную схему:

Идея метода Ньютона для нахождения корней функции



Рассмотрим функцию $\varphi(x) : \mathbb{R} \to \mathbb{R}$.
Основная идея заключается в том, чтобы построить линейное приближение в точке x_k и найти его корень, который будет новой точкой итерации:

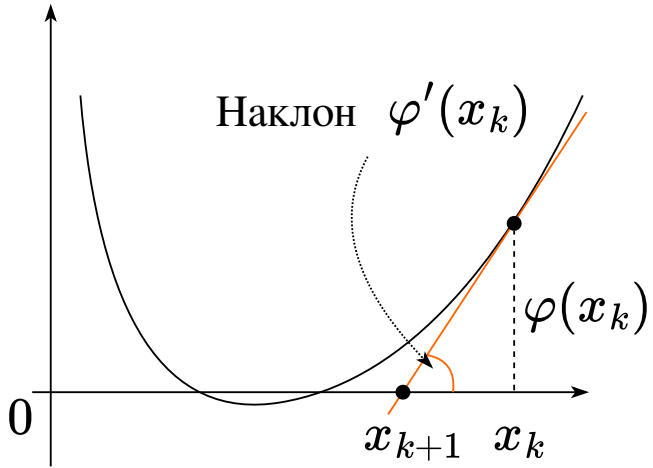
$$\varphi'(x_k) = \frac{\varphi(x_k)}{x_k - x_{k+1}}$$

Мы получаем итерационную схему:

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

[1] Мы фактически решаем задачу нахождения стационарных точек $\nabla f(x) = 0$

Идея метода Ньютона для нахождения корней функции



Рассмотрим функцию $\varphi(x) : \mathbb{R} \to \mathbb{R}$. Основная идея заключается в том, чтобы построить линейное приближение в точке x_k и найти его корень, который будет новой точкой итерации:

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

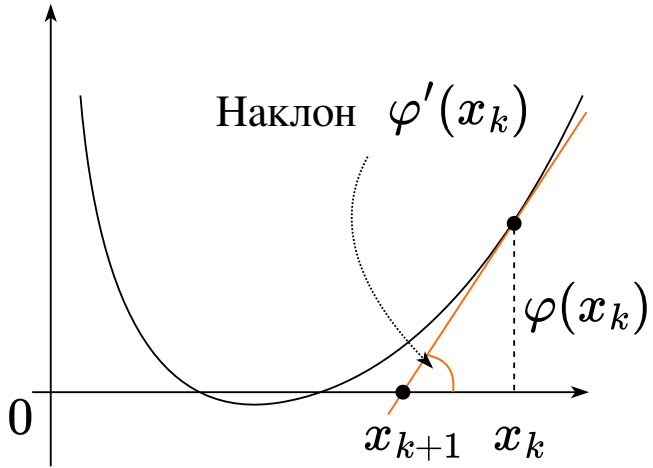
Мы получаем итерационную схему:

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

Этот метод станет методом оптимизации Ньютона в случае $f'(x) = \varphi(x)$[1]:

[1] Мы фактически решаем задачу нахождения стационарных точек $\nabla f(x) = 0$

Идея метода Ньютона для нахождения корней функции



Рассмотрим функцию $\varphi(x) : \mathbb{R} \to \mathbb{R}$. Основная идея заключается в том, чтобы построить линейное приближение в точке x_k и найти его корень, который будет новой точкой итерации:

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

Мы получаем итерационную схему:

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

Этот метод станет методом оптимизации Ньютона в случае $f'(x) = \varphi(x)$[1]:

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

[1] Мы фактически решаем задачу нахождения стационарных точек $\nabla f(x) = 0$

Метод Ньютона как оптимизация локальной квадратичной аппроксимации

Пусть у нас есть функция $f(x)$ и некоторая точка x_k. Рассмотрим квадратичное приближение этой функции в окрестности x_k:

Метод Ньютона как оптимизация локальной квадратичной аппроксимации

Пусть у нас есть функция $f(x)$ и некоторая точка x_k. Рассмотрим квадратичное приближение этой функции в окрестности x_k:

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

Метод Ньютона как оптимизация локальной квадратичной аппроксимации

Пусть у нас есть функция $f(x)$ и некоторая точка x_k. Рассмотрим квадратичное приближение этой функции в окрестности x_k:

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

Идея метода заключается в том, чтобы найти точку x_{k+1}, которая минимизирует функцию $f_{x_k}^{II}(x)$, т.е. $\nabla f_{x_k}^{II}(x_{k+1}) = 0$.

Метод Ньютона как оптимизация локальной квадратичной аппроксимации

Пусть у нас есть функция $f(x)$ и некоторая точка x_k. Рассмотрим квадратичное приближение этой функции в окрестности x_k:

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

Идея метода заключается в том, чтобы найти точку x_{k+1}, которая минимизирует функцию $f_{x_k}^{II}(x)$, т.е. $\nabla f_{x_k}^{II}(x_{k+1}) = 0$.

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

Метод Ньютона как оптимизация локальной квадратичной аппроксимации

Пусть у нас есть функция $f(x)$ и некоторая точка x_k. Рассмотрим квадратичное приближение этой функции в окрестности x_k:

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

Идея метода заключается в том, чтобы найти точку x_{k+1}, которая минимизирует функцию $f_{x_k}^{II}(x)$, т.е. $\nabla f_{x_k}^{II}(x_{k+1}) = 0$.

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

Метод Ньютона как оптимизация локальной квадратичной аппроксимации

Пусть у нас есть функция $f(x)$ и некоторая точка x_k. Рассмотрим квадратичное приближение этой функции в окрестности x_k:

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

Идея метода заключается в том, чтобы найти точку x_{k+1}, которая минимизирует функцию $f_{x_k}^{II}(x)$, т.е. $\nabla f_{x_k}^{II}(x_{k+1}) = 0$.

$$\begin{aligned} \nabla f_{x_k}^{II}(x_{k+1}) &= \nabla f(x_k) + \nabla^2 f(x_k)(x_{k+1} - x_k) = 0 \\ \nabla^2 f(x_k)(x_{k+1} - x_k) &= -\nabla f(x_k) \\ \left[\nabla^2 f(x_k)\right]^{-1} \nabla^2 f(x_k)(x_{k+1} - x_k) &= -\left[\nabla^2 f(x_k)\right]^{-1} \nabla f(x_k) \end{aligned}$$

Метод Ньютона как оптимизация локальной квадратичной аппроксимации

Пусть у нас есть функция $f(x)$ и некоторая точка x_k. Рассмотрим квадратичное приближение этой функции в окрестности x_k:

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

Идея метода заключается в том, чтобы найти точку x_{k+1}, которая минимизирует функцию $f_{x_k}^{II}(x)$, т.е. $\nabla f_{x_k}^{II}(x_{k+1}) = 0$.

$$\begin{aligned}
\nabla f_{x_k}^{II}(x_{k+1}) &= \nabla f(x_k) + \nabla^2 f(x_k)(x_{k+1} - x_k) = 0 \\
\nabla^2 f(x_k)(x_{k+1} - x_k) &= -\nabla f(x_k) \\
\left[\nabla^2 f(x_k)\right]^{-1} \nabla^2 f(x_k)(x_{k+1} - x_k) &= -\left[\nabla^2 f(x_k)\right]^{-1} \nabla f(x_k) \\
x_{k+1} &= x_k - \left[\nabla^2 f(x_k)\right]^{-1} \nabla f(x_k).
\end{aligned}$$

Метод Ньютона как оптимизация локальной квадратичной аппроксимации

Пусть у нас есть функция $f(x)$ и некоторая точка x_k. Рассмотрим квадратичное приближение этой функции в окрестности x_k:

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

Идея метода заключается в том, чтобы найти точку x_{k+1}, которая минимизирует функцию $f_{x_k}^{II}(x)$, т.е. $\nabla f_{x_k}^{II}(x_{k+1}) = 0$.

$$\begin{aligned}
\nabla f_{x_k}^{II}(x_{k+1}) &= \nabla f(x_k) + \nabla^2 f(x_k)(x_{k+1} - x_k) = 0 \\
\nabla^2 f(x_k)(x_{k+1} - x_k) &= -\nabla f(x_k) \\
\left[\nabla^2 f(x_k)\right]^{-1} \nabla^2 f(x_k)(x_{k+1} - x_k) &= -\left[\nabla^2 f(x_k)\right]^{-1} \nabla f(x_k) \\
x_{k+1} &= x_k - \left[\nabla^2 f(x_k)\right]^{-1} \nabla f(x_k).
\end{aligned}$$

Метод Ньютона как оптимизация локальной квадратичной аппроксимации

Пусть у нас есть функция $f(x)$ и некоторая точка x_k. Рассмотрим квадратичное приближение этой функции в окрестности x_k:

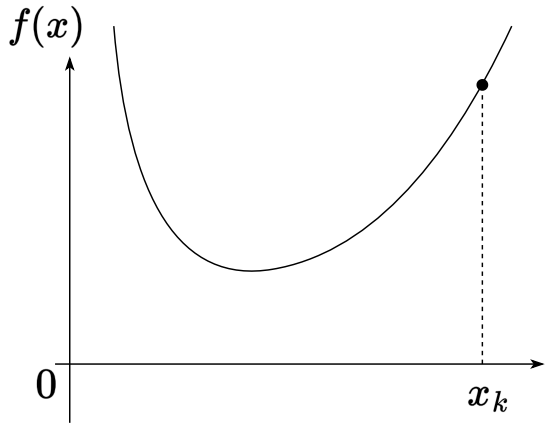
$$f_{x_k}^{II}(x) = f(x_k) + \langle \nabla f(x_k), x - x_k \rangle + \frac{1}{2}\langle \nabla^2 f(x_k)(x - x_k), x - x_k \rangle.$$

Идея метода заключается в том, чтобы найти точку x_{k+1}, которая минимизирует функцию $f_{x_k}^{II}(x)$, т.е. $\nabla f_{x_k}^{II}(x_{k+1}) = 0$.

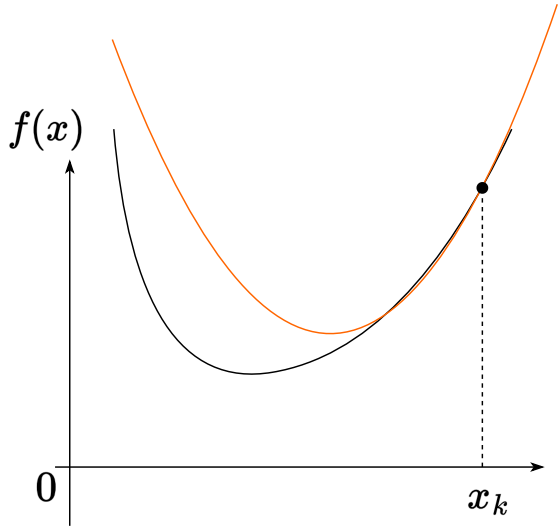
$$\begin{aligned}
\nabla f_{x_k}^{II}(x_{k+1}) &= \nabla f(x_k) + \nabla^2 f(x_k)(x_{k+1} - x_k) = 0 \\
\nabla^2 f(x_k)(x_{k+1} - x_k) &= -\nabla f(x_k) \\
\left[\nabla^2 f(x_k)\right]^{-1} \nabla^2 f(x_k)(x_{k+1} - x_k) &= -\left[\nabla^2 f(x_k)\right]^{-1} \nabla f(x_k) \\
x_{k+1} &= x_k - \left[\nabla^2 f(x_k)\right]^{-1} \nabla f(x_k).
\end{aligned}$$

Необходимо отметить ограничения, связанные с необходимостью невырожденности (для существования метода) и положительной определенности (для гарантии сходимости) гессиана.

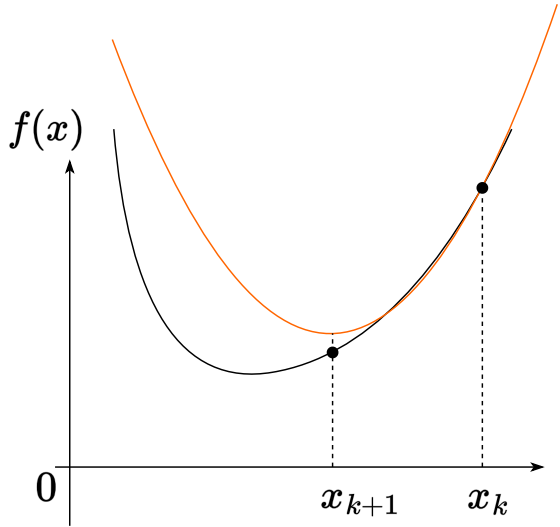
Метод Ньютона как оптимизация локальной квадратичной аппроксимации



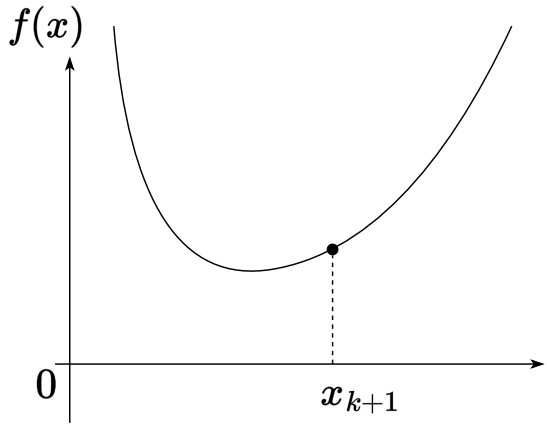
Метод Ньютона как оптимизация локальной квадратичной аппроксимации



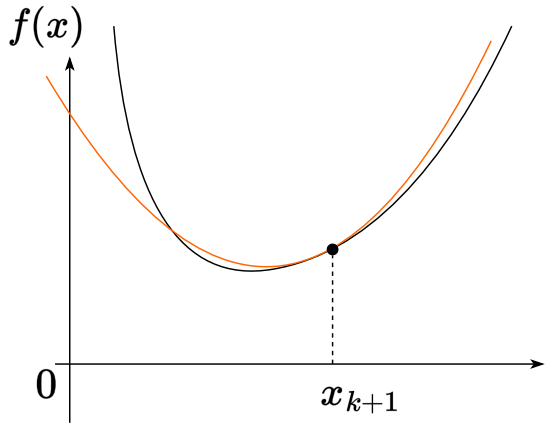
Метод Ньютона как оптимизация локальной квадратичной аппроксимации



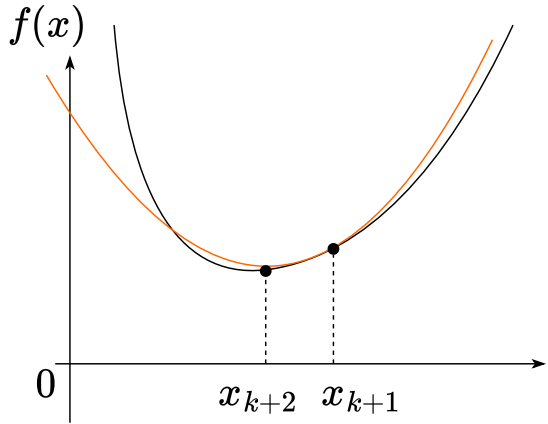
Метод Ньютона как оптимизация локальной квадратичной аппроксимации



Метод Ньютона как оптимизация локальной квадратичной аппроксимации



Метод Ньютона как оптимизация локальной квадратичной аппроксимации



Сходимость

Theorem

Пусть $f(x)$ - сильно выпуклая дважды непрерывно дифференцируемая функция на $\mathbb{R}^n$, для второй производной которой выполняются неравенства: $\mu I_n \preceq \nabla^2 f(x) \preceq L I_n$. Пусть также гессиан функции M-липшицев. Тогда метод Ньютона сходится локально к решению с квадратичной скоростью, т.е. при $\|x_0 - x^*\| < \frac{2\mu}{M}$:

$$\|x_{k+1} - x^*\| \leq \frac{M}{2\mu}\|x_k - x^*\|^2$$

Сходимость

Theorem

Пусть $f(x)$ - сильно выпуклая дважды непрерывно дифференцируемая функция на $\mathbb{R}^n$, для второй производной которой выполняются неравенства: $\mu I_n \preceq \nabla^2 f(x) \preceq L I_n$. Пусть также гессиан функции M-липшицев. Тогда метод Ньютона сходится локально к решению с квадратичной скоростью, т.е. при $\|x_0 - x^*\| < \frac{2\mu}{M}$:

$$\|x_{k+1} - x^*\| \leq \frac{M}{2\mu}\|x_k - x^*\|^2$$

Доказательство

Сходимость

Theorem

Пусть $f(x)$ - сильно выпуклая дважды непрерывно дифференцируемая функция на $\mathbb{R}^n$, для второй производной которой выполняются неравенства: $\mu I_n \preceq \nabla^2 f(x) \preceq L I_n$. Пусть также гессиан функции M-липшицев. Тогда метод Ньютона сходится локально к решению с квадратичной скоростью, т.е. при $\|x_0 - x^*\| < \frac{2\mu}{M}$:

$$\|x_{k+1} - x^*\| \leq \frac{M}{2\mu}\|x_k - x^*\|^2$$

Доказательство

1. Так как x^* - минимум сильно выпуклой функции, $\nabla f(x^*) = 0$. Используя формулу Ньютона-Лейбница:

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

Сходимость

Theorem

Пусть $f(x)$ - сильно выпуклая дважды непрерывно дифференцируемая функция на $\mathbb{R}^n$, для второй производной которой выполняются неравенства: $\mu I_n \preceq \nabla^2 f(x) \preceq L I_n$. Пусть также гессиан функции M-липшицев. Тогда метод Ньютона сходится локально к решению с квадратичной скоростью, т.е. при $\|x_0 - x^*\| < \frac{2\mu}{M}$:

$$\|x_{k+1} - x^*\| \leq \frac{M}{2\mu}\|x_k - x^*\|^2$$

Доказательство

1. Так как x^* - минимум сильно выпуклой функции, $\nabla f(x^*) = 0$. Используя формулу Ньютона-Лейбница:

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

2. Мы будем отслеживать расстояние до решения

Сходимость

Theorem

Пусть $f(x)$ - сильно выпуклая дважды непрерывно дифференцируемая функция на $\mathbb{R}^n$, для второй производной которой выполняются неравенства: $\mu I_n \preceq \nabla^2 f(x) \preceq L I_n$. Пусть также гессиан функции M-липшицев. Тогда метод Ньютона сходится локально к решению с квадратичной скоростью, т.е. при $\|x_0 - x^*\| < \frac{2\mu}{M}$:

$$\|x_{k+1} - x^*\| \leq \frac{M}{2\mu}\|x_k - x^*\|^2$$

Доказательство

1. Так как x^* - минимум сильно выпуклой функции, $\nabla f(x^*) = 0$. Используя формулу Ньютона-Лейбница:

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

2. Мы будем отслеживать расстояние до решения

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

Сходимость

Theorem

Пусть $f(x)$ - сильно выпуклая дважды непрерывно дифференцируемая функция на $\mathbb{R}^n$, для второй производной которой выполняются неравенства: $\mu I_n \preceq \nabla^2 f(x) \preceq L I_n$. Пусть также гессиан функции M-липшицев. Тогда метод Ньютона сходится локально к решению с квадратичной скоростью, т.е. при $\|x_0 - x^*\| < \frac{2\mu}{M}$:

$$\|x_{k+1} - x^*\| \leq \frac{M}{2\mu}\|x_k - x^*\|^2$$

Доказательство

1. Так как x^* - минимум сильно выпуклой функции, $\nabla f(x^*) = 0$. Используя формулу Ньютона-Лейбница:

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

2. Мы будем отслеживать расстояние до решения

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

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

Сходимость

3.

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

Сходимость

3.

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

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

Сходимость

3.

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

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

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

Сходимость

3.

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

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

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

$$= \left[\nabla^2 f(x_k)\right]^{-1} G_k (x_k - x^*)$$

Сходимость

3.

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

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

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

$$= \left[\nabla^2 f(x_k)\right]^{-1} G_k (x_k - x^*)$$

4. Введём:

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

Сходимость

5. Попробуем оценить размер G_k с помощью $r_k = \|x_k - x^*\|$:

Сходимость

5. Попробуем оценить размер G_k с помощью $r_k = \|x_k - x^*\|$:

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

Сходимость

5. Попробуем оценить размер G_k с помощью $r_k = \|x_k - x^*\|$:

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

$$\leq \int_0^1 \left\|\nabla^2 f(x_k) - \nabla^2 f(x^* + \tau(x_k - x^*))\right\| d\tau \leq \qquad \text{(Липшицевость гессиана)}$$

Сходимость

5. Попробуем оценить размер G_k с помощью $r_k = \|x_k - x^*\|$:

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

$$\leq \int_0^1 \left\|\nabla^2 f(x_k) - \nabla^2 f(x^* + \tau(x_k - x^*))\right\| d\tau \leq \qquad \text{(Липшицевость гессиана)}$$

$$\leq \int_0^1 M\|x_k - x^* - \tau(x_k - x^*)\| d\tau = \int_0^1 M\|x_k - x^*\|(1-\tau) d\tau = \frac{r_k}{2} M,$$

Сходимость

5. Попробуем оценить размер G_k с помощью $r_k = \|x_k - x^*\|$:

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

$$\leq \int_0^1 \left\|\nabla^2 f(x_k) - \nabla^2 f(x^* + \tau(x_k - x^*))\right\| d\tau \leq \qquad \text{(Липшицевость гессиана)}$$

$$\leq \int_0^1 M\|x_k - x^* - \tau(x_k - x^*)\| d\tau = \int_0^1 M\|x_k - x^*\|(1-\tau) d\tau = \frac{r_k}{2} M,$$

6. Из сильной выпуклости $\nabla^2 f(x) \succeq \mu I_n$ непосредственно следует:

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

Подставляя в оценку:

$$r_{k+1} \leq \frac{1}{\mu} \cdot \frac{r_k}{2} M \cdot r_k = \frac{M}{2\mu} r_k^2$$

Сходимость

5. Попробуем оценить размер G_k с помощью $r_k = \|x_k - x^*\|$:

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

$$\leq \int_0^1 \left\|\nabla^2 f(x_k) - \nabla^2 f(x^* + \tau(x_k - x^*))\right\| d\tau \leq \qquad \text{(Липшицевость гессиана)}$$

$$\leq \int_0^1 M\|x_k - x^* - \tau(x_k - x^*)\| d\tau = \int_0^1 M\|x_k - x^*\|(1-\tau) d\tau = \frac{r_k}{2} M,$$

6. Из сильной выпуклости $\nabla^2 f(x) \succeq \mu I_n$ непосредственно следует:

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

Подставляя в оценку:

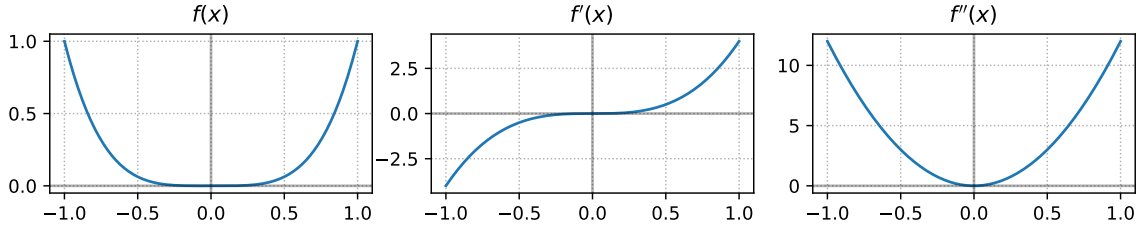
$$r_{k+1} \leq \frac{1}{\mu} \cdot \frac{r_k}{2} M \cdot r_k = \frac{M}{2\mu} r_k^2$$

7. Сжимающее отображение при $\frac{M}{2\mu} r_k < 1$, т.е. $r_k < \frac{2\mu}{M}$.
Таким образом, мы получили важный результат: метод Ньютона для функции с липшицевым положительно определённым гессианом сходится **квадратично** вблизи решения.

Свойства метода Ньютона

Отсутствие квадратичной сходимости, если некоторые предположения нарушаются

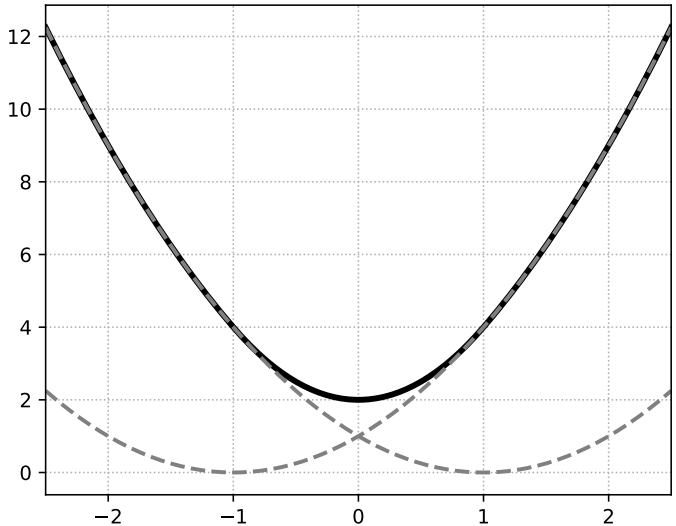
$$f(x) = x^4 \qquad f'(x) = 4x^3 \qquad f''(x) = 12x^2$$



$$x_{k+1} = x_k - \frac{f'(x_k)}{f''(x_k)} = x_k - \frac{4x_k^3}{12x_k^2} = x_k - \frac{1}{3}x_k = \frac{2}{3}x_k,$$

сходится к 0, единственному решению задачи, с линейной скоростью.

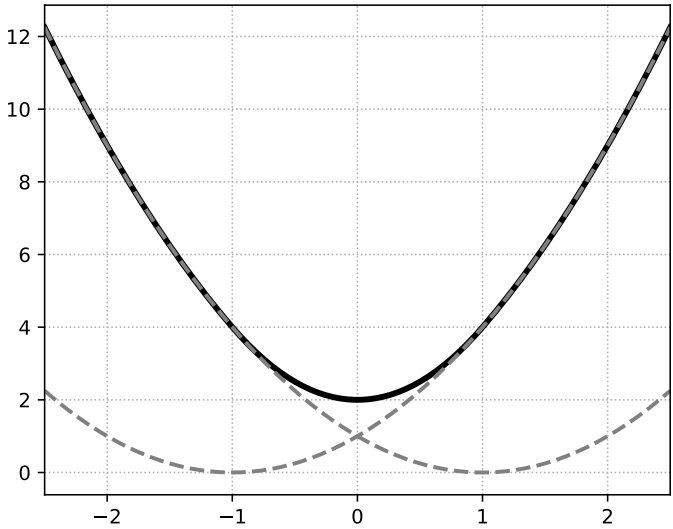
Локальная сходимость метода Ньютона для гладкой сильно выпуклой $f(x)$



$$f(x) = \begin{cases} (x-1)^2, & x \le -1 \\ 2x^2 + 2, & -1 < x < 1 \\ (x+1)^2, & x \ge 1 \end{cases}$$

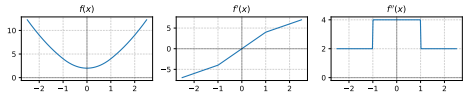
Эта функция сильно выпукла, но вторая производная не является липшицевой.

Локальная сходимость метода Ньютона для гладкой сильно выпуклой $f(x)$

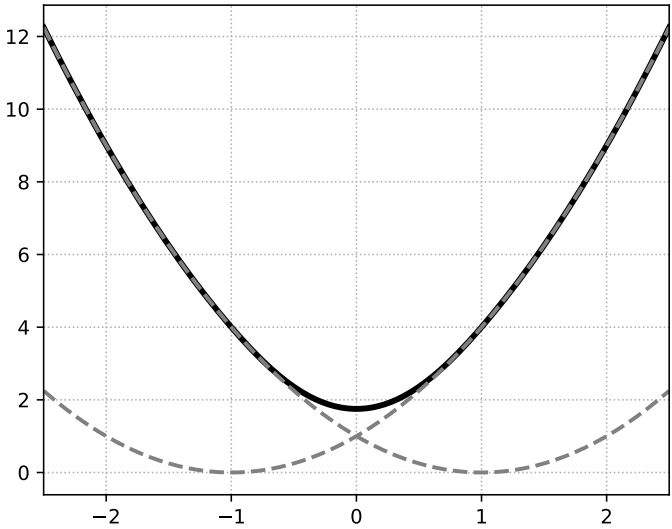


$$f(x) = \begin{cases} (x-1)^2, & x \le -1 \\ 2x^2 + 2, & -1 < x < 1 \\ (x+1)^2, & x \ge 1 \end{cases}$$

Эта функция сильно выпукла, но вторая производная не является липшицевой.

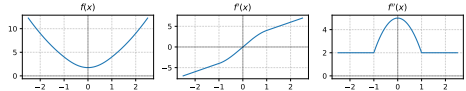


Локальная сходимость метода Ньютона даже если $\nabla^2 f$ липшицев

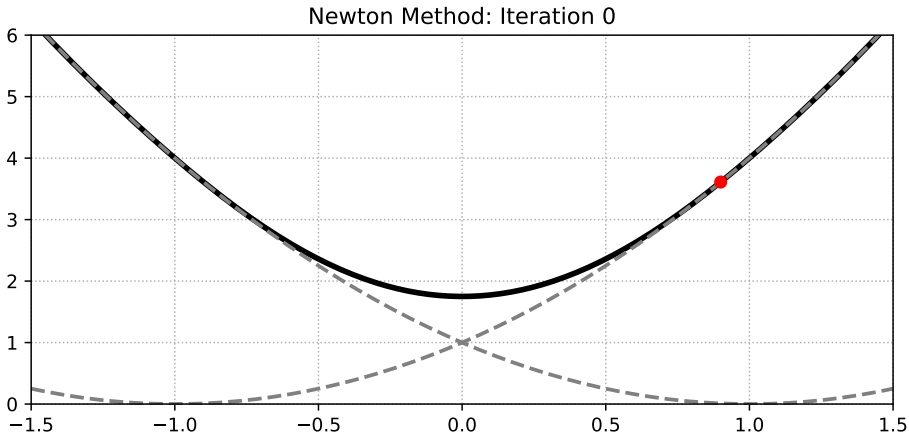


$$f(x) = \begin{cases} (x-1)^2, & x \le -1 \\ -\frac{1}{4}x^4 + \frac{5}{2}x^2 + \frac{7}{4}, & -1 < x < 1 \\ (x+1)^2, & x \ge 1 \end{cases}$$

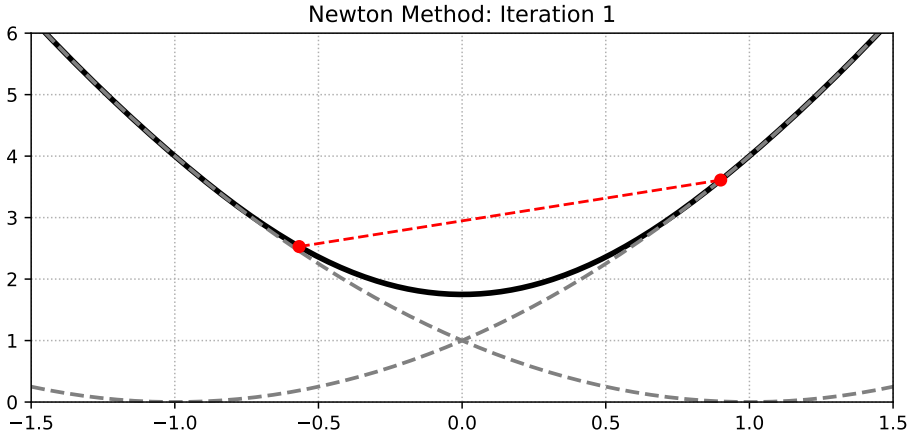
Эта функция сильно выпукла и вторая производная является липшицевой.



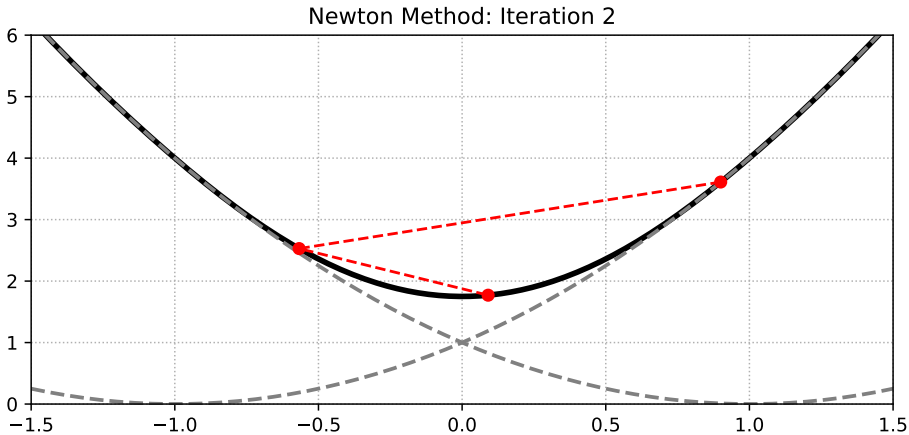
Локальная сходимость метода Ньютона. Хорошая инициализация



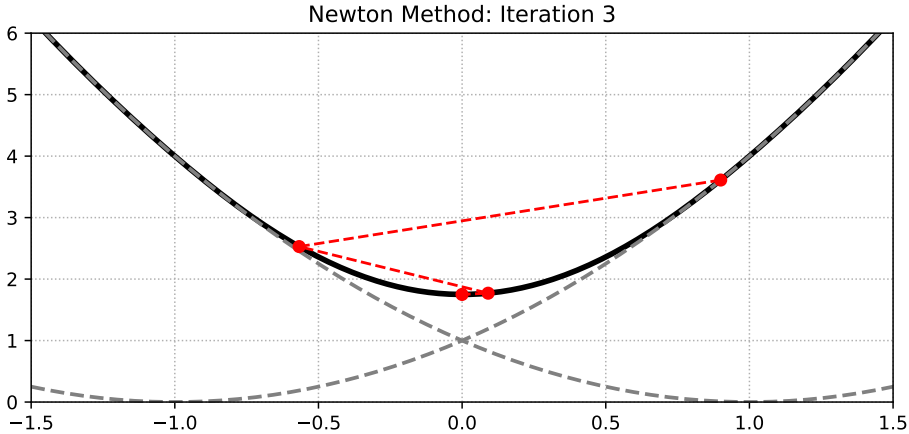
Локальная сходимость метода Ньютона. Хорошая инициализация



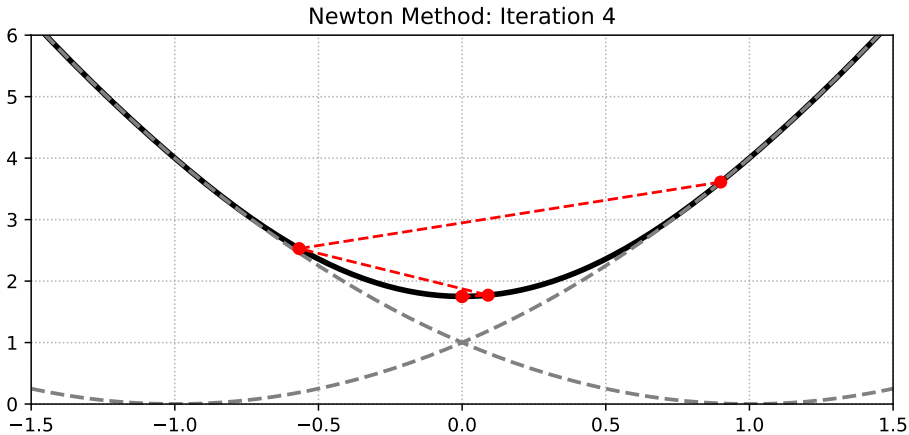
Локальная сходимость метода Ньютона. Хорошая инициализация



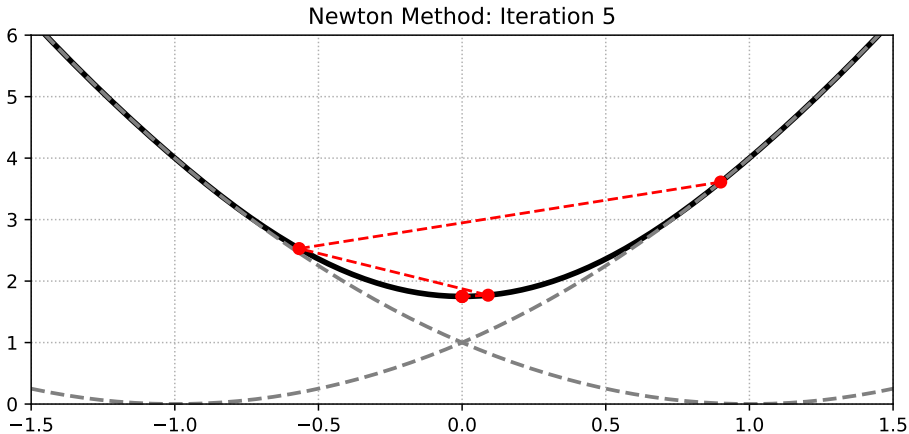
Локальная сходимость метода Ньютона. Хорошая инициализация



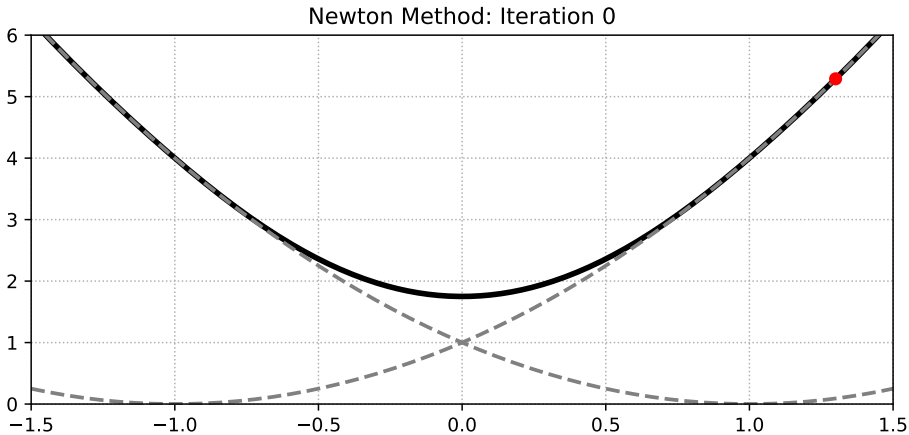
Локальная сходимость метода Ньютона. Хорошая инициализация



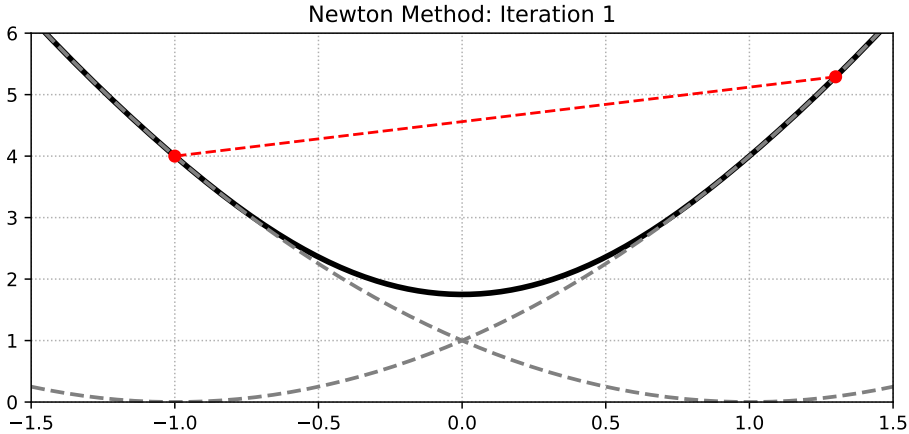
Локальная сходимость метода Ньютона. Хорошая инициализация



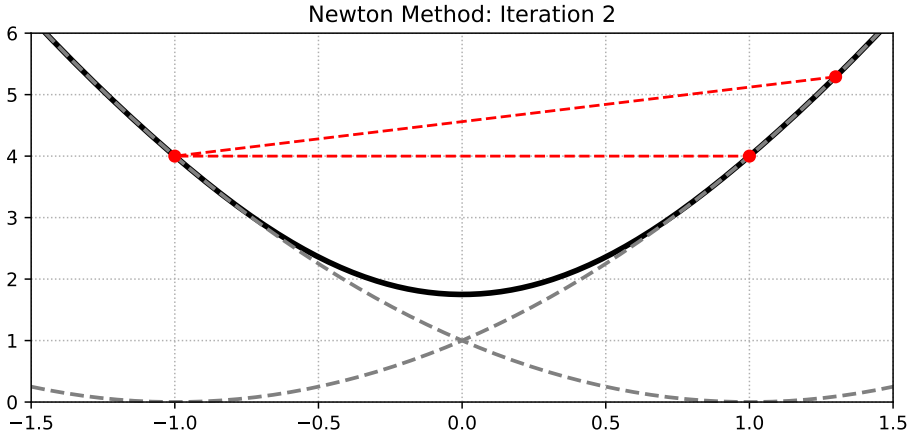
Локальная сходимость метода Ньютона. Плохая инициализация



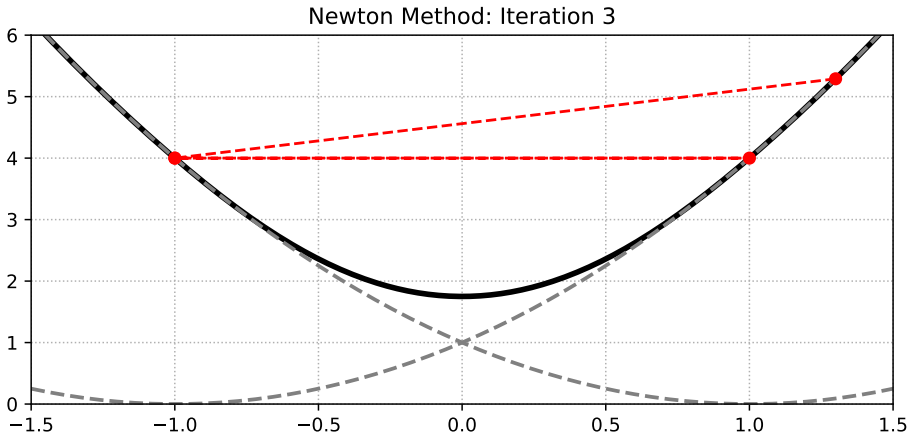
Локальная сходимость метода Ньютона. Плохая инициализация



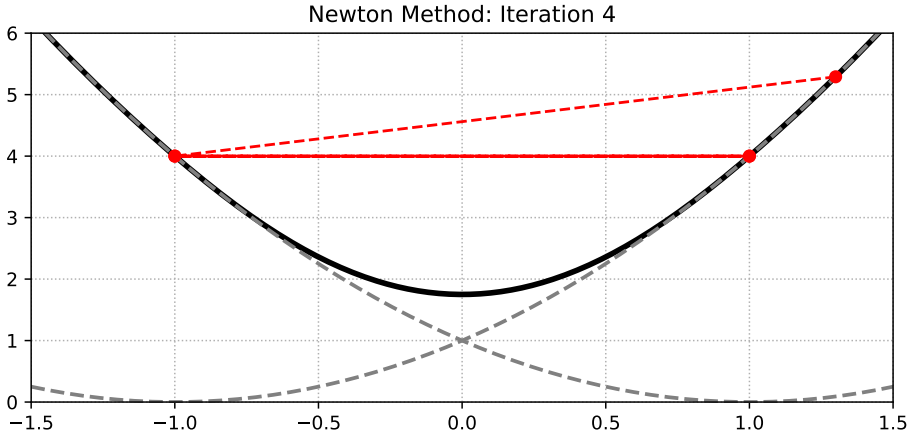
Локальная сходимость метода Ньютона. Плохая инициализация



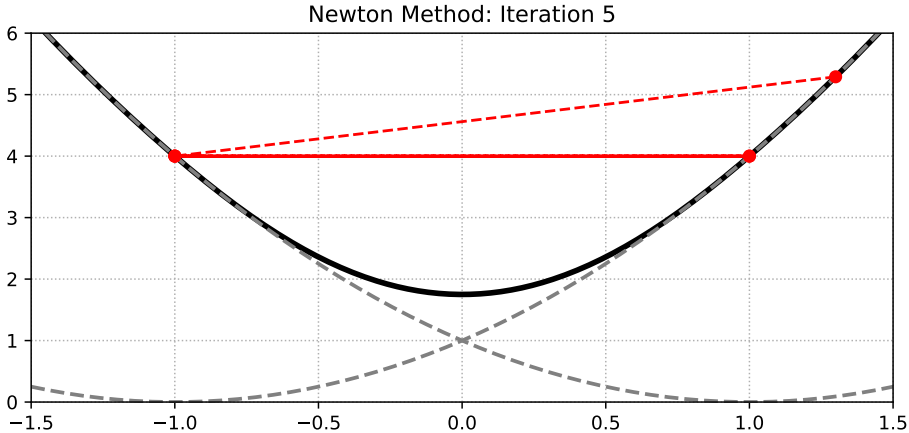
Локальная сходимость метода Ньютона. Плохая инициализация



Локальная сходимость метода Ньютона. Плохая инициализация



Локальная сходимость метода Ньютона. Плохая инициализация



Проблемы метода Ньютона

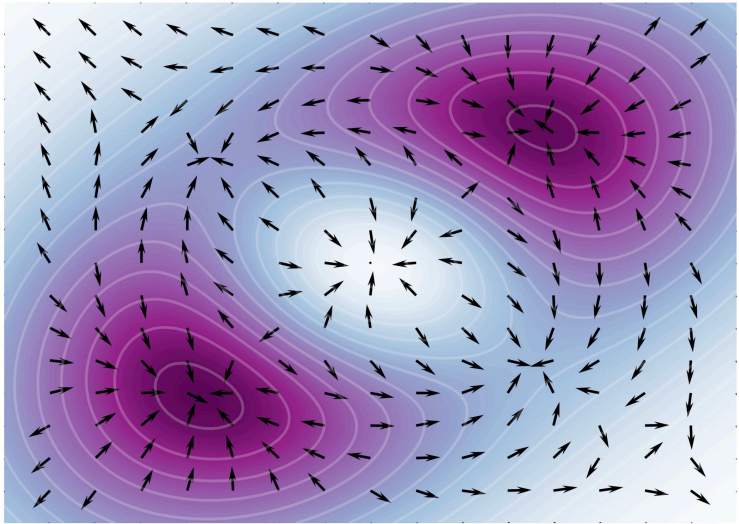


Рис. 1: Animation

Проблемы метода Ньютона

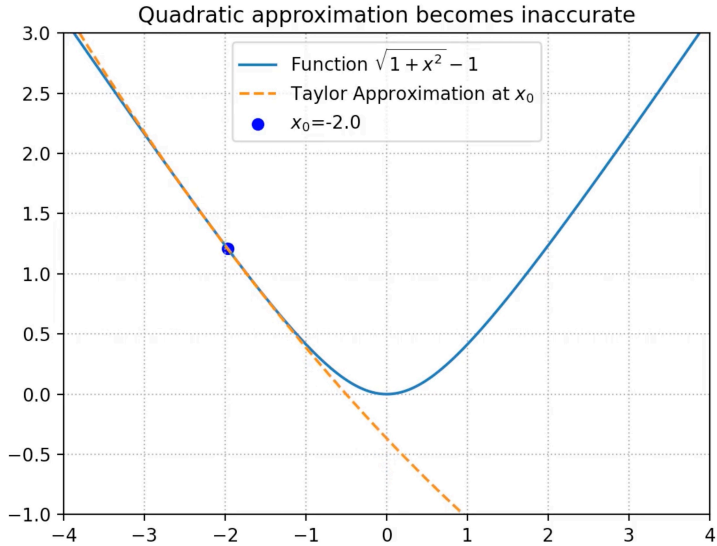


Рис. 2: Animation

Метод Ньютона для квадратичной задачи (линейной регрессии)

$$\min_{x \in \mathbb{R}^n} \frac{1}{2} x^\top A x - b^\top x, \qquad A \in \mathbb{R}^{n \times n}, \quad \lambda(A) \in [\mu; L].$$

Strongly convex quadratics: n=60, random matrix, μ=1, L=10

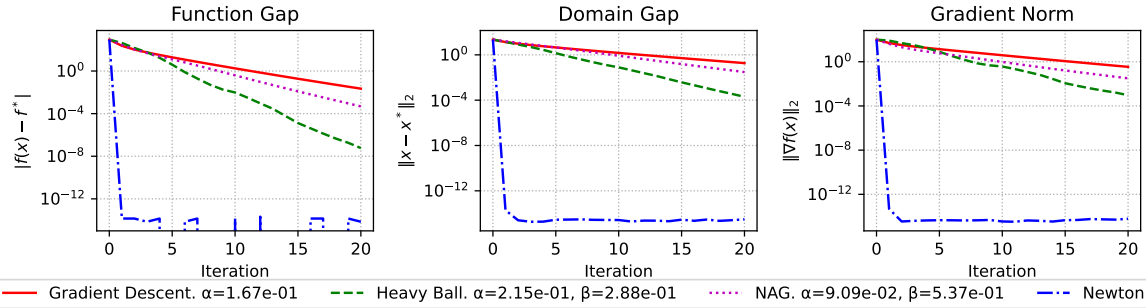


Рис. 3: Так как задача - квадратичная, то метод Ньютона сходится за один шаг.

Метод Ньютона для квадратичной задачи (линейной регрессии)

$$\min_{x \in \mathbb{R}^n} \frac{1}{2} x^\top A x - b^\top x, \qquad A \in \mathbb{R}^{n \times n}, \quad \lambda(A) \in [\mu; L].$$

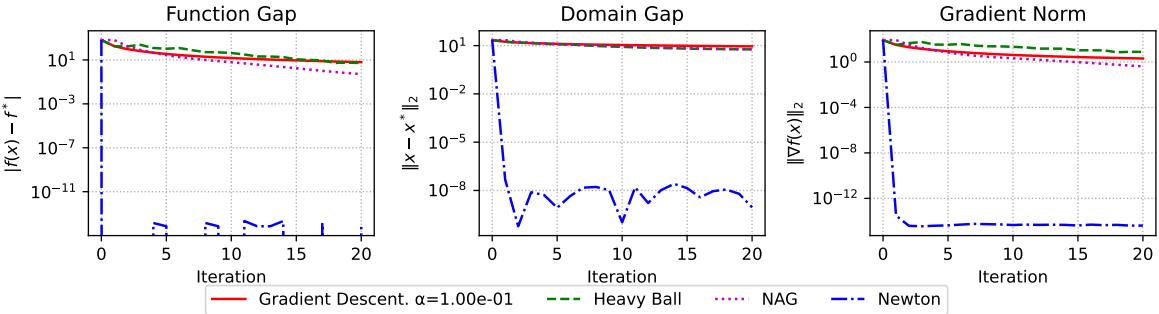


Рис. 4: В этом случае метод Ньютона тоже крайне быстро сходится, однако, отметим, что это происходит благодаря тому, что минимальное собственное число гессиана не 0, а около 10^{-8}. Если применять метод Ньютона в наивной форме с обращением матрицы, то получится ошибка, так как матрица вырождена. На практике все равно можно использовать метод, если для направления итерации решать линейную систему $\nabla^2 f(x_k) d_k = -\nabla f(x_k)$ методом наименьших квадратов.

Метод Ньютона для квадратичной задачи (линейной регрессии)

$$\min_{x \in \mathbb{R}^n} \frac{1}{2} x^\top A x - b^\top x, \qquad A \in \mathbb{R}^{n \times n}, \quad \lambda(A) \in [\mu; L].$$

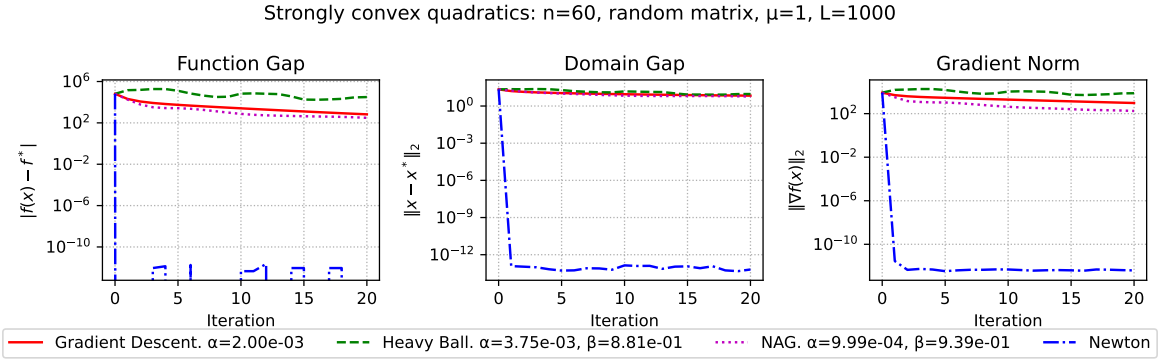


Рис. 5: Здесь число обусловленности гессиана в 1000 раз больше, чем в предыдущем случае, и метод Ньютона сходится за 1 итерацию.

Метод Ньютона для задачи бинарной логистической регрессии

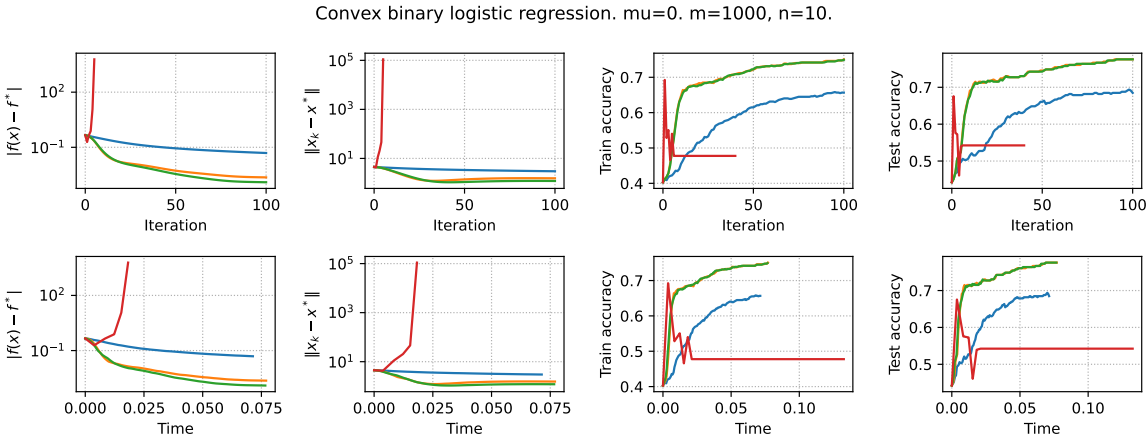


Рис. 6: Наблюдается расходимость метода Ньютона. Сразу отметим, что в задаче нет регуляризации и гарантии сильной выпуклости. А также нет гарантий того, что мы инициализируем метод в окрестности решения.

Метод Ньютона для задачи бинарной логистической регрессии

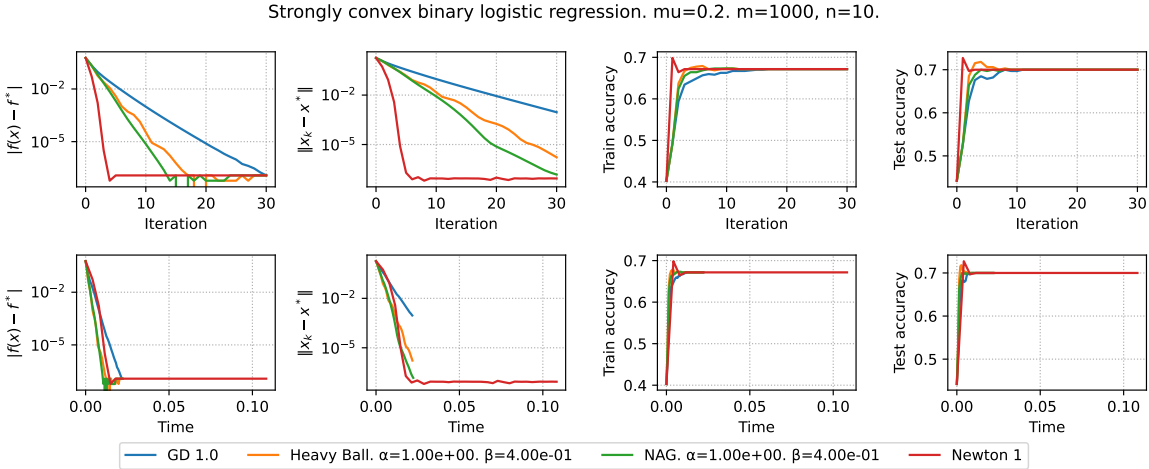


Рис. 7: Добавление регуляризации гарантирует сильную выпуклость, наблюдается сходимость метода Ньютона.

Метод Ньютона для задачи бинарной логистической регрессии

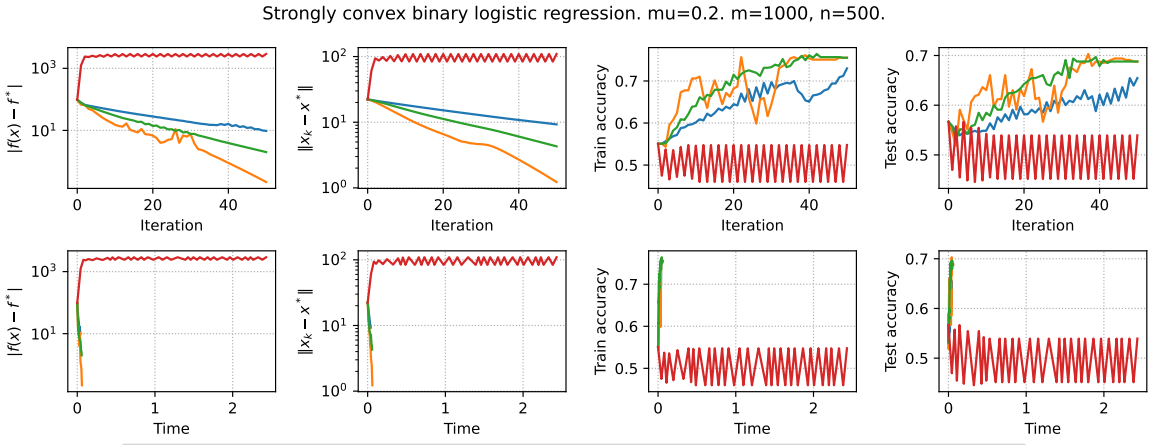


Рис. 8: Увеличим размерность в 50 раз и наблюдаем расходимость метода Ньютона. Это можно связать с тем, что мы инициализируем метод в точке, далекой от решения

Метод Ньютона для задачи бинарной логистической регрессии

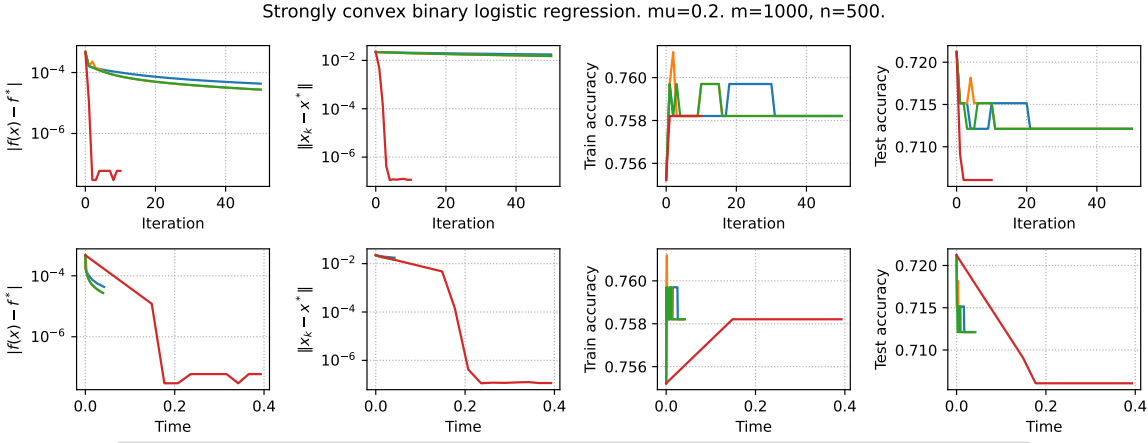


Рис. 9: Не меняя задачу, изменим начальную точку и наблюдаем квадратичную сходимость метода Ньютона. Однако, обратите внимание на время работы. Уже при небольшой размерности, метод Ньютона работает значительно дольше, чем градиентные методы.

Задача нахождения аналитического центра многогранника

Найти точку $x \in \mathbb{R}^n$, которая максимизирует сумму логарифмов расстояний до границ политопа:

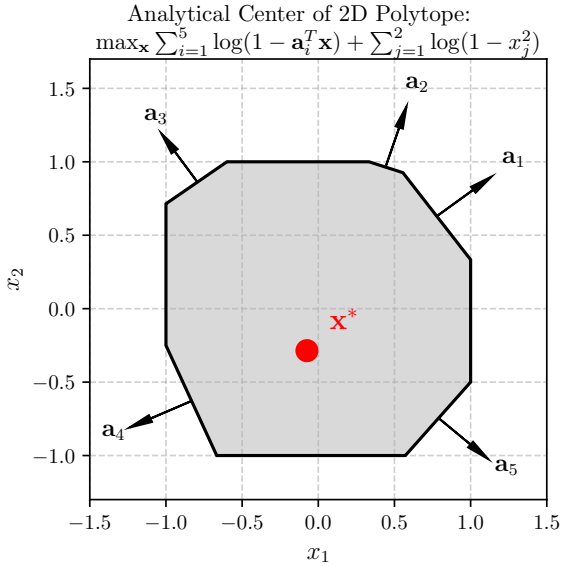
$$\max_x \sum_{i=1}^{m} \log(1 - a_i^T x) + \sum_{j=1}^{n} \log(1 - x_j^2)$$

или, эквивалентно, минимизирует:

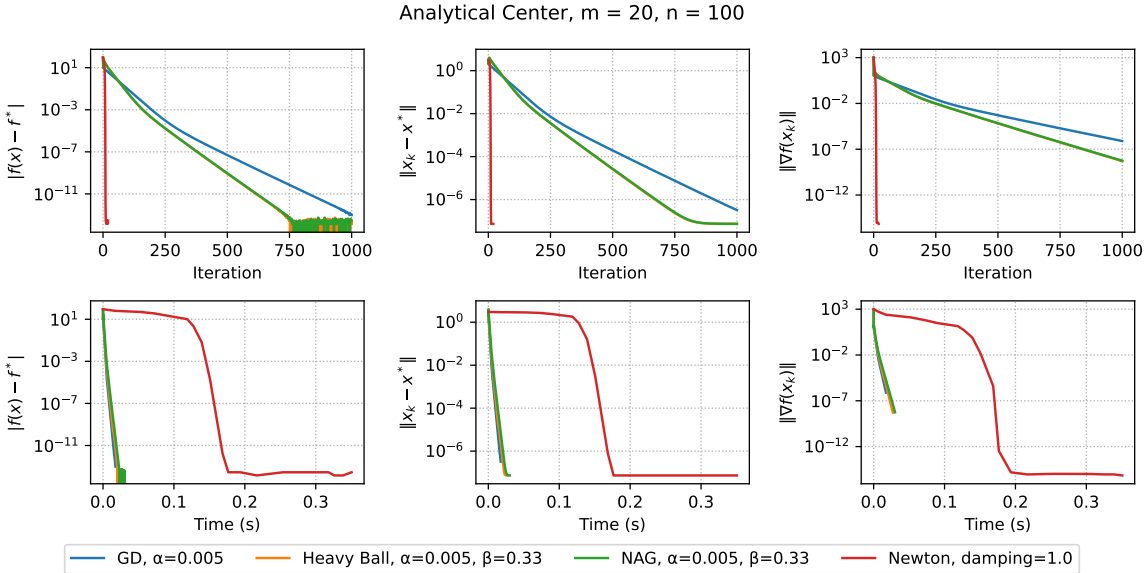
$$\min_x -\sum_{i=1}^{m} \log(1 - a_i^T x) - \sum_{j=1}^{n} \log(1 - x_j^2)$$

при ограничениях: - $a_i^T x < 1$ для всех $i = 1, ..., m$, где a_i - строки матрицы A^T - $|x_j| < 1$ для всех $j = 1, ..., n$

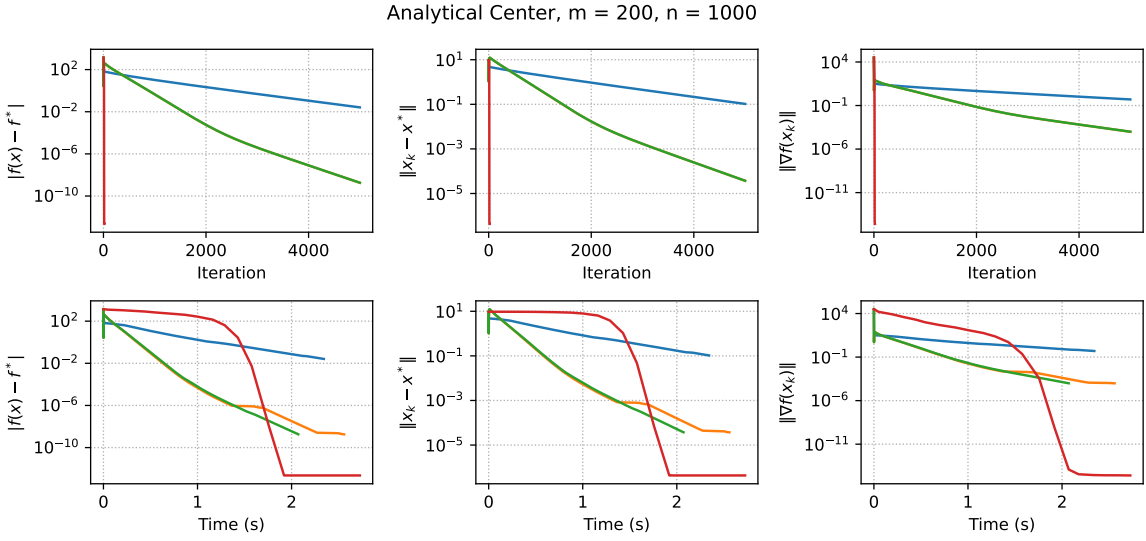
Аналитический центр многогранника - это точка, которая максимально удалена от всех границ многогранника в смысле логарифмического барьера. Эта концепция широко используется в методах внутренней точки для выпуклой оптимизации.



Задача нахождения аналитического центра многогранника



Задача нахождения аналитического центра многогранника



Аффинная инвариантность метода Ньютона

Важным свойством метода Ньютона является **аффинная инвариантность**. Пусть дана функция f и невырожденная матрица $A \in \mathbb{R}^{n \times n}$, пусть $x = Ay$, и пусть $g(y) = f(Ay)$. Заметим, что $\nabla g(y) = A^T \nabla f(x)$ и $\nabla^2 g(y) = A^T \nabla^2 f(x) A$. Шаги Ньютона на g выражаются как:

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

Аффинная инвариантность метода Ньютона

Важным свойством метода Ньютона является **аффинная инвариантность**. Пусть дана функция f и невырожденная матрица $A \in \mathbb{R}^{n \times n}$, пусть $x = Ay$, и пусть $g(y) = f(Ay)$. Заметим, что $\nabla g(y) = A^T \nabla f(x)$ и $\nabla^2 g(y) = A^T \nabla^2 f(x) A$. Шаги Ньютона на g выражаются как:

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

Раскрывая это, мы получаем:

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

Аффинная инвариантность метода Ньютона

Важным свойством метода Ньютона является **аффинная инвариантность**. Пусть дана функция f и невырожденная матрица $A \in \mathbb{R}^{n \times n}$, пусть $x = Ay$, и пусть $g(y) = f(Ay)$. Заметим, что $\nabla g(y) = A^T \nabla f(x)$ и $\nabla^2 g(y) = A^T \nabla^2 f(x) A$. Шаги Ньютона на g выражаются как:

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

Раскрывая это, мы получаем:

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

Используя свойство обратной матрицы $(AB)^{-1} = B^{-1} A^{-1}$, это упрощается до:

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

$$Ay_{k+1} = Ay_k - \left(\nabla^2 f(Ay_k)\right)^{-1} \nabla f(Ay_k)$$

Аффинная инвариантность метода Ньютона

Важным свойством метода Ньютона является **аффинная инвариантность**. Пусть дана функция f и невырожденная матрица $A \in \mathbb{R}^{n \times n}$, пусть $x = Ay$, и пусть $g(y) = f(Ay)$. Заметим, что $\nabla g(y) = A^T \nabla f(x)$ и $\nabla^2 g(y) = A^T \nabla^2 f(x) A$. Шаги Ньютона на g выражаются как:

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

Раскрывая это, мы получаем:

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

Используя свойство обратной матрицы $(AB)^{-1} = B^{-1} A^{-1}$, это упрощается до:

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

Таким образом, правило обновления для x выглядит так:

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

Аффинная инвариантность метода Ньютона

Важным свойством метода Ньютона является **аффинная инвариантность**. Пусть дана функция f и невырожденная матрица $A \in \mathbb{R}^{n \times n}$, пусть $x = Ay$, и пусть $g(y) = f(Ay)$. Заметим, что $\nabla g(y) = A^T \nabla f(x)$ и $\nabla^2 g(y) = A^T \nabla^2 f(x) A$. Шаги Ньютона на g выражаются как:

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

Раскрывая это, мы получаем:

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

Используя свойство обратной матрицы $(AB)^{-1} = B^{-1} A^{-1}$, это упрощается до:

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

Таким образом, правило обновления для x выглядит так:

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

Это показывает, что итерация метода Ньютона не зависит от масштаба задачи. У градиентного спуска такого свойства нет!

Итоги

Плюсы:

- Квадратичная сходимость вблизи решения x^*

Минусы:

Итоги

Плюсы:

- Квадратичная сходимость вблизи решения x^*
- Аффинная инвариантность

Минусы:

Итоги

Плюсы:

- Квадратичная сходимость вблизи решения x^*
- Аффинная инвариантность
- Отсутствие параметров у метода

Минусы:

Итоги

Плюсы:

- Квадратичная сходимость вблизи решения x^*
- Аффинная инвариантность
- Отсутствие параметров у метода
- Сходимость можно сделать глобальной, если использовать демпфированный метод Ньютона (добавить процедуру линейного поиска и шага метода)

Минусы:

Итоги

Плюсы:

- Квадратичная сходимость вблизи решения x^*
- Аффинная инвариантность
- Отсутствие параметров у метода
- Сходимость можно сделать глобальной, если использовать демпфированный метод Ньютона (добавить процедуру линейного поиска и шага метода)

Минусы:

- Необходимо хранить (обратный) гессиан на каждой итерации: $\mathcal{O}(n^2)$ памяти

Итоги

Плюсы:

- Квадратичная сходимость вблизи решения x^*
- Аффинная инвариантность
- Отсутствие параметров у метода
- Сходимость можно сделать глобальной, если использовать демпфированный метод Ньютона (добавить процедуру линейного поиска и шага метода)

Минусы:

- Необходимо хранить (обратный) гессиан на каждой итерации: $\mathcal{O}(n^2)$ памяти
- Необходимо решать линейные системы: $\mathcal{O}(n^3)$ операций

Итоги

Плюсы:

- Квадратичная сходимость вблизи решения x^*
- Аффинная инвариантность
- Отсутствие параметров у метода
- Сходимость можно сделать глобальной, если использовать демпфированный метод Ньютона (добавить процедуру линейного поиска и шага метода)

Минусы:

- Необходимо хранить (обратный) гессиан на каждой итерации: $\mathcal{O}(n^2)$ памяти
- Необходимо решать линейные системы: $\mathcal{O}(n^3)$ операций
- Гессиан может быть вырожденным в x^*

Итоги

Плюсы:

- Квадратичная сходимость вблизи решения x^*
- Аффинная инвариантность
- Отсутствие параметров у метода
- Сходимость можно сделать глобальной, если использовать демпфированный метод Ньютона (добавить процедуру линейного поиска и шага метода)

Минусы:

- Необходимо хранить (обратный) гессиан на каждой итерации: $\mathcal{O}(n^2)$ памяти
- Необходимо решать линейные системы: $\mathcal{O}(n^3)$ операций
- Гессиан может быть вырожденным в x^*
- Гессиан может не быть положительно определенным $\rightarrow$ направление $-(f''(x))^{-1}f'(x)$ может не быть направлением спуска

Квазиньютоновские методы

Идея методов адаптивной метрики

Зафиксируем точку x_k и зададим метрику пространства симметричной положительно определённой матрицей $A \succ 0$:

$$\|x - x_k\|_A^2 = (x - x_k)^\top A(x - x_k)$$

Идея методов адаптивной метрики

Зафиксируем точку x_k и зададим метрику пространства симметричной положительно определённой матрицей $A \succ 0$:

$$\|x - x_k\|_A^2 = (x - x_k)^\top A(x - x_k)$$

Определим **направление наискорейшего спуска** относительно метрики A как направление наибольшего убывания линеаризации f на единичной A-сфере:

Идея методов адаптивной метрики

Зафиксируем точку x_k и зададим метрику пространства симметричной положительно определённой матрицей $A \succ 0$:

$$\|x - x_k\|_A^2 = (x - x_k)^\top A(x - x_k)$$

Определим **направление наискорейшего спуска** относительно метрики A как направление наибольшего убывания линеаризации f на единичной A-сфере:

$$d = \arg \min_{\|\delta x\|_A = 1} \nabla f(x_k)^\top \delta x$$

Идея методов адаптивной метрики

Зафиксируем точку x_k и зададим метрику пространства симметричной положительно определённой матрицей $A \succ 0$:

$$\|x - x_k\|_A^2 = (x - x_k)^\top A (x - x_k)$$

Определим **направление наискорейшего спуска** относительно метрики A как направление наибольшего убывания линеаризации f на единичной A-сфере:

$$d = \arg \min_{\|\delta x\|_A = 1} \nabla f(x_k)^\top \delta x$$

Лагранжиан:

$$\mathcal{L}(\delta x, \lambda) = \nabla f(x_k)^\top \delta x + \lambda(\delta x^\top A \delta x - 1)$$

Условие стационарности $\nabla_{\delta x} \mathcal{L} = 0$:

$$\nabla f(x_k) + 2\lambda A \delta x = 0 \implies \delta x \propto -A^{-1} \nabla f(x_k)$$

Направление наискорейшего спуска в метрике A:

Идея методов адаптивной метрики

Зафиксируем точку x_k и зададим метрику пространства симметричной положительно определённой матрицей $A \succ 0$:

$$\|x - x_k\|_A^2 = (x - x_k)^\top A(x - x_k)$$

Определим **направление наискорейшего спуска** относительно метрики A как направление наибольшего убывания линеаризации f на единичной A-сфере:

$$d = \arg \min_{\|\delta x\|_A = 1} \nabla f(x_k)^\top \delta x$$

Лагранжиан:

$$\mathcal{L}(\delta x, \lambda) = \nabla f(x_k)^\top \delta x + \lambda(\delta x^\top A \delta x - 1)$$

Условие стационарности $\nabla_{\delta x} \mathcal{L} = 0$:

$$\nabla f(x_k) + 2\lambda A \delta x = 0 \implies \delta x \propto -A^{-1} \nabla f(x_k)$$

Направление наискорейшего спуска в метрике A:

$$\boxed{d = -A^{-1} \nabla f(x_k)}$$

Идея методов адаптивной метрики

Зафиксируем точку x_k и зададим метрику пространства симметричной положительно определённой матрицей $A \succ 0$:

$$\|x - x_k\|_A^2 = (x - x_k)^\top A(x - x_k)$$

Определим **направление наискорейшего спуска** относительно метрики A как направление наибольшего убывания линеаризации f на единичной A-сфере:

$$d = \arg \min_{\|\delta x\|_A = 1} \nabla f(x_k)^\top \delta x$$

Лагранжиан:

$$\mathcal{L}(\delta x, \lambda) = \nabla f(x_k)^\top \delta x + \lambda(\delta x^\top A \delta x - 1)$$

Условие стационарности $\nabla_{\delta x} \mathcal{L} = 0$:

$$\nabla f(x_k) + 2\lambda A \delta x = 0 \implies \delta x \propto -A^{-1} \nabla f(x_k)$$

Направление наискорейшего спуска в метрике A:

$$\boxed{d = -A^{-1} \nabla f(x_k)}$$

Выбор метрики определяет направление метода:

Идея методов адаптивной метрики

Зафиксируем точку x_k и зададим метрику пространства симметричной положительно определённой матрицей $A \succ 0$:

$$\|x - x_k\|_A^2 = (x - x_k)^\top A(x - x_k)$$

Определим **направление наискорейшего спуска** относительно метрики A как направление наибольшего убывания линеаризации f на единичной A-сфере:

$$d = \arg \min_{\|\delta x\|_A = 1} \nabla f(x_k)^\top \delta x$$

Лагранжиан:

$$\mathcal{L}(\delta x, \lambda) = \nabla f(x_k)^\top \delta x + \lambda(\delta x^\top A \delta x - 1)$$

Условие стационарности $\nabla_{\delta x} \mathcal{L} = 0$:

$$\nabla f(x_k) + 2\lambda A \delta x = 0 \implies \delta x \propto -A^{-1} \nabla f(x_k)$$

Направление наискорейшего спуска в метрике A:

$$\boxed{d = -A^{-1} \nabla f(x_k)}$$

Выбор метрики определяет направление метода:

Идея методов адаптивной метрики

Зафиксируем точку x_k и зададим метрику пространства симметричной положительно определённой матрицей $A \succ 0$:

$$\|x - x_k\|_A^2 = (x - x_k)^\top A(x - x_k)$$

Определим **направление наискорейшего спуска** относительно метрики A как направление наибольшего убывания линеаризации f на единичной A-сфере:

$$d = \arg\min_{\|\delta x\|_A = 1} \nabla f(x_k)^\top \delta x$$

Лагранжиан:

$$\mathcal{L}(\delta x, \lambda) = \nabla f(x_k)^\top \delta x + \lambda(\delta x^\top A \delta x - 1)$$

Условие стационарности $\nabla_{\delta x}\mathcal{L} = 0$:

$$\nabla f(x_k) + 2\lambda A \delta x = 0 \implies \delta x \propto -A^{-1}\nabla f(x_k)$$

Направление наискорейшего спуска в метрике A:

$$\boxed{d = -A^{-1}\nabla f(x_k)}$$

Выбор метрики определяет направление метода: Градиентный спуск предполагает *изотропное* пространство ($A = I$), в то время как метод Ньютона адаптирует геометрию к *локальной кривизне* функции через гессиан. Квазиньютоновские методы аппроксимируют эту кривизну дешевле - за $O(n^2)$ вместо $O(n^3)$.

Метрика A	Направление	Метод
I_n	$-\nabla f(x_k)$	Градиентный спуск
$\nabla^2 f(x_k)$	$-[\nabla^2 f(x_k)]^{-1}\nabla f(x_k)$	Ньютон
$B_k \approx \nabla^2 f(x_k)$	$-B_k^{-1}\nabla f(x_k)$	Квазиньютон

Интуиция квазиньютоновских методов

Для классической задачи безусловной оптимизации $f(x) \to \min\limits_{x \in \mathbb{R}^n}$ общий алгоритм итерационного метода записывается как:

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

Интуиция квазиньютоновских методов

Для классической задачи безусловной оптимизации $f(x) \to \min\limits_{x \in \mathbb{R}^n}$ общий алгоритм итерационного метода записывается как:

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

В методе Ньютона направление p_k (направление Ньютона) устанавливается решением линейной системы на каждом шаге:

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

Интуиция квазиньютоновских методов

Для классической задачи безусловной оптимизации $f(x) \to \min\limits_{x \in \mathbb{R}^n}$ общий алгоритм итерационного метода записывается как:

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

В методе Ньютона направление p_k (направление Ньютона) устанавливается решением линейной системы на каждом шаге:

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

т.е. на каждой итерации необходимо **вычислить** гессиан и градиент и **решить** линейную систему.

Интуиция квазиньютоновских методов

Для классической задачи безусловной оптимизации $f(x) \to \min\limits_{x \in \mathbb{R}^n}$ общий алгоритм итерационного метода записывается как:

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

В методе Ньютона направление p_k (направление Ньютона) устанавливается решением линейной системы на каждом шаге:

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

т.е. на каждой итерации необходимо **вычислить** гессиан и градиент и **решить** линейную систему.

Обратите внимание, что если мы возьмем единичную матрицу $B_k = I_n$ в качестве B_k на каждом шаге, мы получим в точности метод градиентного спуска.

Общий алгоритм квазиньютоновских методов основан на выборе матрицы B_k так, чтобы она в некотором смысле стремилась к истинному значению гессиана $\nabla^2 f(x_k)$ при $k \to \infty$.

Шаблон квазиньютоновского метода

Пусть $x_0 \in \mathbb{R}^n$, $B_0 \succ 0$. Для $k = 0, 1, 2, ...$, повторяем:

1. Решить $B_k p_k = -\nabla f(x_k)$

Шаблон квазиньютоновского метода

Пусть $x_0 \in \mathbb{R}^n$, $B_0 \succ 0$. Для $k = 0, 1, 2, ...$, повторяем:

1. Решить $B_k p_k = -\nabla f(x_k)$
2. Обновить $x_{k+1} = x_k + \alpha_k p_k$

Шаблон квазиньютоновского метода

Пусть $x_0 \in \mathbb{R}^n$, $B_0 \succ 0$. Для $k = 0, 1, 2, ...$, повторяем:

1. Решить $B_k p_k = -\nabla f(x_k)$
2. Обновить $x_{k+1} = x_k + \alpha_k p_k$
3. Вычислить B_{k+1} из B_k

Шаблон квазиньютоновского метода

Пусть $x_0 \in \mathbb{R}^n$, $B_0 \succ 0$. Для $k = 0, 1, 2, ...$, повторяем:

1. Решить $B_k p_k = -\nabla f(x_k)$
2. Обновить $x_{k+1} = x_k + \alpha_k p_k$
3. Вычислить B_{k+1} из B_k

Шаблон квазиньютоновского метода

Пусть $x_0 \in \mathbb{R}^n$, $B_0 \succ 0$. Для $k = 0, 1, 2, ...$, повторяем:

1. Решить $B_k p_k = -\nabla f(x_k)$
2. Обновить $x_{k+1} = x_k + \alpha_k p_k$
3. Вычислить B_{k+1} из B_k

Разные квазиньютоновские методы реализуют шаг 3 по-разному. Мы скоро увидим, что обычно мы можем вычислить $(B_{k+1})^{-1}$ из $(B_k)^{-1}$.

Шаблон квазиньютоновского метода

Пусть $x_0 \in \mathbb{R}^n$, $B_0 \succ 0$. Для $k = 0, 1, 2, ...$, повторяем:

1. Решить $B_k p_k = -\nabla f(x_k)$
2. Обновить $x_{k+1} = x_k + \alpha_k p_k$
3. Вычислить B_{k+1} из B_k

Разные квазиньютоновские методы реализуют шаг 3 по-разному. Мы скоро увидим, что обычно мы можем вычислить $(B_{k+1})^{-1}$ из $(B_k)^{-1}$.

Основная идея: Поскольку B_k уже содержит информацию о гессиане, используем подходящее обновление матрицы для формирования B_{k+1}.

Шаблон квазиньютоновского метода

Пусть $x_0 \in \mathbb{R}^n$, $B_0 \succ 0$. Для $k = 0, 1, 2, ...$, повторяем:

1. Решить $B_k p_k = -\nabla f(x_k)$
2. Обновить $x_{k+1} = x_k + \alpha_k p_k$
3. Вычислить B_{k+1} из B_k

Разные квазиньютоновские методы реализуют шаг 3 по-разному. Мы скоро увидим, что обычно мы можем вычислить $(B_{k+1})^{-1}$ из $(B_k)^{-1}$.

Основная идея: Поскольку B_k уже содержит информацию о гессиане, используем подходящее обновление матрицы для формирования B_{k+1}.

Разумное требование для B_{k+1} (вдохновленное методом секущих). Обозначим шаг $s_k = x_{k+1} - x_k = \alpha_k p_k$ и разность градиентов $y_k = \nabla f(x_{k+1}) - \nabla f(x_k)$:

$$y_k = B_{k+1} s_k$$

Шаблон квазиньютоновского метода

Пусть $x_0 \in \mathbb{R}^n$, $B_0 \succ 0$. Для $k = 0, 1, 2, ...$, повторяем:

1. Решить $B_k p_k = -\nabla f(x_k)$
2. Обновить $x_{k+1} = x_k + \alpha_k p_k$
3. Вычислить B_{k+1} из B_k

Разные квазиньютоновские методы реализуют шаг 3 по-разному. Мы скоро увидим, что обычно мы можем вычислить $(B_{k+1})^{-1}$ из $(B_k)^{-1}$.

Основная идея: Поскольку B_k уже содержит информацию о гессиане, используем подходящее обновление матрицы для формирования B_{k+1}.

Разумное требование для B_{k+1} (вдохновленное методом секущих). Обозначим шаг $s_k = x_{k+1} - x_k = \alpha_k p_k$ и разность градиентов $y_k = \nabla f(x_{k+1}) - \nabla f(x_k)$:

$$y_k = B_{k+1}\, s_k$$

Помимо уравнения секущей, мы хотим:

- B_{k+1} симметричная

Шаблон квазиньютоновского метода

Пусть $x_0 \in \mathbb{R}^n$, $B_0 \succ 0$. Для $k = 0, 1, 2, ...$, повторяем:

1. Решить $B_k p_k = -\nabla f(x_k)$
2. Обновить $x_{k+1} = x_k + \alpha_k p_k$
3. Вычислить B_{k+1} из B_k

Разные квазиньютоновские методы реализуют шаг 3 по-разному. Мы скоро увидим, что обычно мы можем вычислить $(B_{k+1})^{-1}$ из $(B_k)^{-1}$.

Основная идея: Поскольку B_k уже содержит информацию о гессиане, используем подходящее обновление матрицы для формирования B_{k+1}.

Разумное требование для B_{k+1} (вдохновленное методом секущих). Обозначим шаг $s_k = x_{k+1} - x_k = \alpha_k p_k$ и разность градиентов $y_k = \nabla f(x_{k+1}) - \nabla f(x_k)$:

$$y_k = B_{k+1} s_k$$

Помимо уравнения секущей, мы хотим:

- B_{k+1} симметричная
- B_{k+1} близка к B_k

Шаблон квазиньютоновского метода

Пусть $x_0 \in \mathbb{R}^n$, $B_0 \succ 0$. Для $k = 0, 1, 2, ...$, повторяем:

1. Решить $B_k p_k = -\nabla f(x_k)$
2. Обновить $x_{k+1} = x_k + \alpha_k p_k$
3. Вычислить B_{k+1} из B_k

Разные квазиньютоновские методы реализуют шаг 3 по-разному. Мы скоро увидим, что обычно мы можем вычислить $(B_{k+1})^{-1}$ из $(B_k)^{-1}$.

Основная идея: Поскольку B_k уже содержит информацию о гессиане, используем подходящее обновление матрицы для формирования B_{k+1}.

Разумное требование для B_{k+1} (вдохновленное методом секущих). Обозначим шаг $s_k = x_{k+1} - x_k = \alpha_k p_k$ и разность градиентов $y_k = \nabla f(x_{k+1}) - \nabla f(x_k)$:

$$y_k = B_{k+1} s_k$$

Помимо уравнения секущей, мы хотим:

- B_{k+1} симметричная
- B_{k+1} близка к B_k
- $B_k \succ 0 \Rightarrow B_{k+1} \succ 0$

Симметричное одноранговое обновление

Попробуем обновление вида:

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

Симметричное одноранговое обновление

Попробуем обновление вида:

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

Уравнение секущей $B_{k+1}s_k = y_k$ дает:

$$(au^T s_k)u = y_k - B_k s_k$$

Симметричное одноранговое обновление

Попробуем обновление вида:

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

Уравнение секущей $B_{k+1}s_k = y_k$ дает:

$$(au^T s_k)u = y_k - B_k s_k$$

Это верно только если u является кратным $y_k - B_k s_k$. Положив $u = y_k - B_k s_k$, мы решаем уравнение,

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

Симметричное одноранговое обновление

Попробуем обновление вида:

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

Уравнение секущей $B_{k+1}s_k = y_k$ дает:

$$(au^T s_k)u = y_k - B_k s_k$$

Это верно только если u является кратным $y_k - B_k s_k$. Положив $u = y_k - B_k s_k$, мы решаем уравнение,

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

что приводит к

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

Это называется симметричным одноранговым (SR1) обновлением.

Симметричное одноранговое обновление с инверсией

Как мы можем решить

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

чтобы сделать следующий шаг? Помимо обновления $B_k \to B_{k+1}$, будем также обновлять обратную матрицу $H_k = B_k^{-1} \to H_{k+1} = B_{k+1}^{-1}$.

Формула Шермана-Моррисона:

Формула Шермана-Моррисона утверждает:

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

Таким образом, для SR1 обновления, обратная матрица также легко обновляется:

$$H_{k+1} = H_k + \frac{(s_k - H_k y_k)(s_k - H_k y_k)^T}{(s_k - H_k y_k)^T y_k}$$

В общем, SR1 прост и дешев, но у него есть ключевой недостаток: он не сохраняет положительную определенность.

Обновление Давидона-Флетчера-Пауэлла

Мы могли бы продолжить ту же идею для обновления обратной матрицы H:

$$H_{k+1} = H_k + auu^T + bvv^T.$$

Обновление Давидона-Флетчера-Пауэлла

Мы могли бы продолжить ту же идею для обновления обратной матрицы H:

$$H_{k+1} = H_k + auu^T + bvv^T.$$

Подставляя уравнение секущей $H_{k+1}y_k = s_k$ и решая для a, b, получаем:

$$H_{k+1} = H_k - \frac{H_k y_k y_k^T H_k}{y_k^T H_k y_k} + \frac{s_k s_k^T}{y_k^T s_k}$$

Применение формулы Вудбери

Вудбери показывает:

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

Это обновление Давидона-Флетчера-Пауэлла (DFP). Также дешево: $O(n^2)$, но сохраняет положительную определенность. Не так популярно, как BFGS.

Обновление Бройдена-Флетчера-Гольдштейна-Шенно

Попробуем теперь двухранговое обновление:

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

Обновление Бройдена-Флетчера-Гольдштейна-Шенно

Попробуем теперь двухранговое обновление:

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

Уравнение секущей $y_k = B_{k+1}s_k$ дает:

$$y_k - B_k s_k = (au^T s_k)u + (bv^T s_k)v$$

Обновление Бройдена-Флетчера-Гольдштейна-Шенно

Попробуем теперь двухранговое обновление:

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

Уравнение секущей $y_k = B_{k+1}s_k$ дает:

$$y_k - B_k s_k = (au^T s_k)u + (bv^T s_k)v$$

Положив $u = y_k$, $v = B_k s_k$ и решая для a, b, получаем:

$$B_{k+1} = B_k - \frac{B_k s_k s_k^T B_k}{s_k^T B_k s_k} + \frac{y_k y_k^T}{s_k^T y_k}$$

Это обновление Бройдена-Флетчера-Гольдштейна-Шенно (BFGS).

Обновление Бройдена-Флетчера-Гольдштейна-Шенно с инверсией

Формула Вудбери

Формула Вудбери, обобщение формулы Шермана-Моррисона, дается как:

$$(A + UCV)^{-1} = A^{-1} - A^{-1}U(C^{-1} + VA^{-1}U)^{-1}VA^{-1}$$

Обновление Бройдена-Флетчера-Гольдштейна-Шенно с инверсией

Формула Вудбери

Формула Вудбери, обобщение формулы Шермана-Моррисона, дается как:

$$(A + UCV)^{-1} = A^{-1} - A^{-1}U(C^{-1} + VA^{-1}U)^{-1}VA^{-1}$$

Применяя её к нашему случаю, получаем двухранговое обновление обратной матрицы H:

$$H_{k+1} = H_k + \frac{(s_k - H_k y_k)s_k^T}{y_k^T s_k} + \frac{s_k(s_k - H_k y_k)^T}{y_k^T s_k} - \frac{(s_k - H_k y_k)^T y_k}{(y_k^T s_k)^2} s_k s_k^T$$

$$H_{k+1} = \left(I - \frac{s_k y_k^T}{y_k^T s_k}\right) H_k \left(I - \frac{y_k s_k^T}{y_k^T s_k}\right) + \frac{s_k s_k^T}{y_k^T s_k}$$

Эта формулировка обеспечивает, что обновление BFGS, оставаясь достаточно общим, сохраняет вычислительную эффективность и требует $O(n^2)$ операций. Важно, что обновление BFGS сохраняет положительную определенность: $B_k \succ 0 \Rightarrow B_{k+1} \succ 0$. Эквивалентно, $H_k \succ 0 \Rightarrow H_{k+1} \succ 0$.

Алгоритм BFGS

Используем обозначения s_k, y_k из уравнения секущей. Определим $\rho_k = \frac{1}{y_k^T s_k}$.

Алгоритм BFGS

Используем обозначения s_k, y_k из уравнения секущей. Определим $\rho_k = \frac{1}{y_k^\top s_k}$.

Вход: $x_0 \in \mathbb{R}^n,\ H_0 = I_n$

Для $k = 0, 1, 2, \ldots$:

$$p_k = -H_k \nabla f(x_k)$$

$$\alpha_k = \text{LineSearch}(f, x_k, p_k) \quad \text{(условия Вульфа)}$$

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

$$s_k = x_{k+1} - x_k, \quad y_k = \nabla f(x_{k+1}) - \nabla f(x_k)$$

$$\rho_k = \frac{1}{y_k^\top s_k}$$

$$H_{k+1} = (I - \rho_k s_k y_k^\top)\, H_k\, (I - \rho_k y_k s_k^\top) + \rho_k s_k s_k^\top$$

Условия Вульфа

Для корректности обновления BFGS необходимо, чтобы $y_k^\top s_k > 0$ (иначе положительная определенность H_{k+1} нарушается). Это гарантируется условиями Вульфа для линейного поиска.

Условия Вульфа

Для корректности обновления BFGS необходимо, чтобы $y_k^\top s_k > 0$ (иначе положительная определенность H_{k+1} нарушается). Это гарантируется условиями Вульфа для линейного поиска.

Условия Вульфа

Шаг α_k выбирается так, чтобы выполнялись два условия:

1. **Достаточное убывание** (условие Армихо):

$$f(x_k + \alpha_k p_k) \leq f(x_k) + c_1 \alpha_k \nabla f(x_k)^\top p_k, \quad c_1 \in (0, 1)$$

Типичные значения: $c_1 = 10^{-4}$, $c_2 = 0.9$.

Условия Вульфа

Для корректности обновления BFGS необходимо, чтобы $y_k^\top s_k > 0$ (иначе положительная определенность H_{k+1} нарушается). Это гарантируется условиями Вульфа для линейного поиска.

Условия Вульфа

Шаг α_k выбирается так, чтобы выполнялись два условия:

1. **Достаточное убывание** (условие Армихо):

$$f(x_k + \alpha_k p_k) \leq f(x_k) + c_1 \alpha_k \nabla f(x_k)^\top p_k, \quad c_1 \in (0, 1)$$

2. **Условие кривизны**:

$$\nabla f(x_k + \alpha_k p_k)^\top p_k \geq c_2 \nabla f(x_k)^\top p_k, \quad c_2 \in (c_1, 1)$$

Типичные значения: $c_1 = 10^{-4}$, $c_2 = 0.9$.

Условия Вульфа

Для корректности обновления BFGS необходимо, чтобы $y_k^\top s_k > 0$ (иначе положительная определенность H_{k+1} нарушается). Это гарантируется условиями Вульфа для линейного поиска.

Условия Вульфа

Шаг α_k выбирается так, чтобы выполнялись два условия:

1. **Достаточное убывание** (условие Армихо):

$$f(x_k + \alpha_k p_k) \leq f(x_k) + c_1 \alpha_k \nabla f(x_k)^\top p_k, \quad c_1 \in (0, 1)$$

2. **Условие кривизны**:

$$\nabla f(x_k + \alpha_k p_k)^\top p_k \geq c_2 \nabla f(x_k)^\top p_k, \quad c_2 \in (c_1, 1)$$

Типичные значения: $c_1 = 10^{-4}$, $c_2 = 0.9$.

Условия Вульфа

Для корректности обновления BFGS необходимо, чтобы $y_k^\top s_k > 0$ (иначе положительная определенность H_{k+1} нарушается). Это гарантируется условиями Вульфа для линейного поиска.

Условия Вульфа

Шаг α_k выбирается так, чтобы выполнялись два условия:

1. **Достаточное убывание** (условие Армихо):

$$f(x_k + \alpha_k p_k) \leq f(x_k) + c_1 \alpha_k \nabla f(x_k)^\top p_k, \quad c_1 \in (0, 1)$$

2. **Условие кривизны**:

$$\nabla f(x_k + \alpha_k p_k)^\top p_k \geq c_2 \nabla f(x_k)^\top p_k, \quad c_2 \in (c_1, 1)$$

Типичные значения: $c_1 = 10^{-4}$, $c_2 = 0.9$.

Из условия кривизны непосредственно следует:

$$y_k^\top s_k = \left(\nabla f(x_{k+1}) - \nabla f(x_k)\right)^\top s_k \geq (c_2 - 1)\nabla f(x_k)^\top p_k \cdot \alpha_k > 0.$$

Сходимость BFGS

Теорема. Глобальная сходимость BFGS.

Пусть $f \in C^1(\mathbb{R}^n)$ - выпуклая функция с липшицевым градиентом, и множество уровня $\mathcal{L}_0 = \{x : f(x) \leq f(x_0)\}$ ограничено. Тогда метод BFGS с линейным поиском, удовлетворяющим условиям Вульфа, сходится глобально:

$$\lim_{k \to \infty} \nabla f(x_k) = 0.$$

Сходимость BFGS

Теорема. Глобальная сходимость BFGS.

Пусть $f \in C^1(\mathbb{R}^n)$ - выпуклая функция с липшицевым градиентом, и множество уровня $\mathcal{L}_0 = \{x : f(x) \leq f(x_0)\}$ ограничено. Тогда метод BFGS с линейным поиском, удовлетворяющим условиям Вульфа, сходится глобально:

$$\lim_{k\to\infty} \nabla f(x_k) = 0.$$

Более того, вблизи решения x^*, где $\nabla^2 f(x^*) \succ 0$ и гессиан липшицев, метод BFGS демонстрирует **локальную сверхлинейную сходимость** (Dennis, Moré, 1974):

$$\frac{\|x_{k+1} - x^*\|}{\|x_k - x^*\|} \to 0 \quad \text{при } k \to \infty.$$

Сходимость BFGS

Теорема. Глобальная сходимость BFGS.

Пусть $f \in C^1(\mathbb{R}^n)$ - выпуклая функция с липшицевым градиентом, и множество уровня $\mathcal{L}_0 = \{x : f(x) \leq f(x_0)\}$ ограничено. Тогда метод BFGS с линейным поиском, удовлетворяющим условиям Вульфа, сходится глобально:

$$\lim_{k\to\infty} \nabla f(x_k) = 0.$$

Более того, вблизи решения x^*, где $\nabla^2 f(x^*) \succ 0$ и гессиан липшицев, метод BFGS демонстрирует **локальную сверхлинейную сходимость** (Dennis, Moré, 1974):

$$\frac{\|x_{k+1} - x^*\|}{\|x_k - x^*\|} \to 0 \quad \text{при } k \to \infty.$$

Сверхлинейная сходимость BFGS - промежуточный результат между линейной сходимостью градиентного спуска и квадратичной сходимостью метода Ньютона. При этом стоимость одной итерации BFGS составляет $O(n^2)$ вместо $O(n^3)$ у метода Ньютона. Заметим, что последовательность B_k не обязана сходиться к $\nabla^2 f(x^*)$ - достаточно выполнения условия Денниса-Море.

Сравнение квазиньютоновских обновлений

Метод	Обновление	Сохраняет $\succ 0$	Стоимость
SR1	Ранг 1	Нет	$O(n^2)$
DFP	Ранг 2 (на H)	Да	$O(n^2)$
BFGS	Ранг 2 (на B)	Да	$O(n^2)$

Сравнение квазиньютоновских обновлений

Метод	Обновление	Сохраняет $\succ 0$	Стоимость
SR1	Ранг 1	Нет	$O(n^2)$
DFP	Ранг 2 (на H)	Да	$O(n^2)$
BFGS	Ранг 2 (на B)	Да	$O(n^2)$

На практике BFGS значительно превосходит DFP по устойчивости. Причина: DFP обновляет непосредственно H_k, и при неточном линейном поиске ошибки в H_k могут быстро накапливаться. BFGS обновляет B_k, а H_k получается через формулу Вудбери, что даёт лучшую численную устойчивость.

L-BFGS: метод BFGS с ограниченной памятью

От BFGS к L-BFGS

Метод BFGS хранит матрицу $H_k \in \mathbb{R}^{n \times n}$ - это $O(n^2)$ памяти. При $n = 10^6$ хранение H_k требует ~ 4 ТБ в `fp32` - неприемлемо.

От BFGS к L-BFGS

Метод BFGS хранит матрицу $H_k \in \mathbb{R}^{n \times n}$ - это $O(n^2)$ памяти. При $n = 10^6$ хранение H_k требует ~ 4 ТБ в `fp32` - неприемлемо.

Введём обозначение $V_i = I - \rho_i y_i s_i^\top$. Формула обновления BFGS из предыдущего раздела принимает компактный вид:

$$H_{k+1} = V_k^\top H_k V_k + \rho_k s_k s_k^\top$$

От BFGS к L-BFGS

Метод BFGS хранит матрицу $H_k \in \mathbb{R}^{n \times n}$ - это $O(n^2)$ памяти. При $n = 10^6$ хранение H_k требует ~ 4 ТБ в `fp32` - неприемлемо.

Введём обозначение $V_i = I - \rho_i y_i s_i^\top$. Формула обновления BFGS из предыдущего раздела принимает компактный вид:

$$H_{k+1} = V_k^\top H_k V_k + \rho_k s_k s_k^\top$$

Два структурных наблюдения, которые делают L-BFGS возможным:

От BFGS к L-BFGS

Метод BFGS хранит матрицу $H_k \in \mathbb{R}^{n \times n}$ - это $O(n^2)$ памяти. При $n = 10^6$ хранение H_k требует ~ 4 ТБ в `fp32` - неприемлемо.

Введём обозначение $V_i = I - \rho_i y_i s_i^\top$. Формула обновления BFGS из предыдущего раздела принимает компактный вид:

$$H_{k+1} = V_k^\top H_k V_k + \rho_k s_k s_k^\top$$

Два структурных наблюдения, которые делают L-BFGS возможным:

1. **Ранг-2 обновление.** H_k полностью определяется начальным приближением H_0 и историей пар $(s_i, y_i)_{i=0}^{k-1}$ - матрица является «функцией» этой истории.

От BFGS к L-BFGS

Метод BFGS хранит матрицу $H_k \in \mathbb{R}^{n \times n}$ - это $O(n^2)$ памяти. При $n = 10^6$ хранение H_k требует ~ 4 ТБ в `fp32` - неприемлемо.

Введём обозначение $V_i = I - \rho_i y_i s_i^\top$. Формула обновления BFGS из предыдущего раздела принимает компактный вид:

$$H_{k+1} = V_k^\top H_k V_k + \rho_k s_k s_k^\top$$

Два структурных наблюдения, которые делают L-BFGS возможным:

1. **Ранг-2 обновление.** H_k полностью определяется начальным приближением H_0 и историей пар $(s_i, y_i)_{i=0}^{k-1}$ - матрица является «функцией» этой истории.

От BFGS к L-BFGS

Метод BFGS хранит матрицу $H_k \in \mathbb{R}^{n \times n}$ - это $O(n^2)$ памяти. При $n = 10^6$ хранение H_k требует ~ 4 ТБ в `fp32` - неприемлемо.

Введём обозначение $V_i = I - \rho_i y_i s_i^\top$. Формула обновления BFGS из предыдущего раздела принимает компактный вид:

$$H_{k+1} = V_k^\top H_k V_k + \rho_k s_k s_k^\top$$

Два структурных наблюдения, которые делают L-BFGS возможным:

1. **Ранг-2 обновление.** H_k полностью определяется начальным приближением H_0 и историей пар $(s_i, y_i)_{i=0}^{k-1}$ - матрица является «функцией» этой истории.
2. **Нам не нужна сама матрица H_k.** На каждой итерации используется только произведение $p_k = -H_k \nabla f(x_k)$ - вектор, а не матрица.

От BFGS к L-BFGS

Метод BFGS хранит матрицу $H_k \in \mathbb{R}^{n \times n}$ - это $O(n^2)$ памяти. При $n = 10^6$ хранение H_k требует ~ 4 ТБ в `fp32` - неприемлемо.

Введём обозначение $V_i = I - \rho_i y_i s_i^\top$. Формула обновления BFGS из предыдущего раздела принимает компактный вид:

$$H_{k+1} = V_k^\top H_k V_k + \rho_k s_k s_k^\top$$

Два структурных наблюдения, которые делают L-BFGS возможным:

1. **Ранг-2 обновление.** H_k полностью определяется начальным приближением H_0 и историей пар $(s_i, y_i)_{i=0}^{k-1}$ - матрица является «функцией» этой истории.
2. **Нам не нужна сама матрица H_k.** На каждой итерации используется только произведение $p_k = -H_k \nabla f(x_k)$ - вектор, а не матрица.

От BFGS к L-BFGS

Метод BFGS хранит матрицу $H_k \in \mathbb{R}^{n \times n}$ - это $O(n^2)$ памяти. При $n = 10^6$ хранение H_k требует ~ 4 ТБ в `fp32` - неприемлемо.

Введём обозначение $V_i = I - \rho_i y_i s_i^\top$. Формула обновления BFGS из предыдущего раздела принимает компактный вид:

$$H_{k+1} = V_k^\top H_k V_k + \rho_k s_k s_k^\top$$

Два структурных наблюдения, которые делают L-BFGS возможным:

1. **Ранг-2 обновление.** H_k полностью определяется начальным приближением H_0 и историей пар $(s_i, y_i)_{i=0}^{k-1}$ - матрица является «функцией» этой истории.
2. **Нам не нужна сама матрица H_k.** На каждой итерации используется только произведение $p_k = -H_k \nabla f(x_k)$ - вектор, а не матрица.

> **Идея L-BFGS**
>
> Храним m последних пар (s_i, y_i) и вычисляем $H_k \nabla f(x_k)$ **неявно**, без хранения матрицы. Память: $O(mn)$ при типичном $m \in [3, 20]$.

Раскрытие рекурсии

Раскрывая рекурсию $H_{k+1} = V_k^\top H_k V_k + \rho_k s_k s_k^\top$ на m шагов назад:

$$H_k = \underbrace{\left(V_{k-1}^\top \cdots V_{k-m}^\top\right) H_{k-m} \left(V_{k-m} \cdots V_{k-1}\right)}_{\text{вклад «забытой» истории}} + \rho_{k-m} \left(V_{k-1}^\top \cdots V_{k-m+1}^\top\right) s_{k-m} s_{k-m}^\top \left(V_{k-m+1} \cdots V_{k-1}\right) + \ldots + \rho_{k-1} s_{k-1} s_{k-1}^\top$$

Раскрытие рекурсии

Раскрывая рекурсию $H_{k+1} = V_k^\top H_k V_k + \rho_k s_k s_k^\top$ на m шагов назад:

$$H_k = \underbrace{\left(V_{k-1}^\top \cdots V_{k-m}^\top\right) H_{k-m} \left(V_{k-m} \cdots V_{k-1}\right)}_{\text{вклад «забытой» истории}} + \rho_{k-m} \left(V_{k-1}^\top \cdots V_{k-m+1}^\top\right) s_{k-m} s_{k-m}^\top \left(V_{k-m+1} \cdots V_{k-1}\right) + \ldots + \rho_{k-1} s_{k-1} s_{k-1}^\top$$

Усечение: матрица H_{k-m} содержит информацию от пар старше m итераций. В L-BFGS мы «забываем» H_{k-m} и заменяем её простым скалярным приближением $H_k^0 = \gamma_k I$. Но как выбрать γ_k?

Выбор масштаба γ_k

Рассмотрим квадратичную задачу $f(x) = \frac{1}{2}x^\top Ax$ с гессианом $A \succ 0$. Мы ищем $H_k^0 \approx A^{-1}$, но позволяем себе только скаляр, умноженный на единичную матрицу.

Выбор масштаба γ_k

Рассмотрим квадратичную задачу $f(x) = \frac{1}{2}x^\top Ax$ с гессианом $A \succ 0$. Мы ищем $H_k^0 \approx A^{-1}$, но позволяем себе только скаляр, умноженный на единичную матрицу.

Наивный выбор $H_k^0 = I$ — плохая идея. Если собственные числа A равны $\lambda_1 = 1, \lambda_n = 1000$, то A^{-1} имеет собственные числа от 0.001 до 1. Единичная матрица — это даже не в том же масштабе.

Выбор масштаба γ_k

Рассмотрим квадратичную задачу $f(x) = \frac{1}{2}x^\top Ax$ с гессианом $A \succ 0$. Мы ищем $H_k^0 \approx A^{-1}$, но позволяем себе только скаляр, умноженный на единичную матрицу.

Наивный выбор $H_k^0 = I$ — плохая идея. Если собственные числа A равны $\lambda_1 = 1, \lambda_n = 1000$, то A^{-1} имеет собственные числа от 0.001 до 1. Единичная матрица — это даже не в том же масштабе.

Идея: извлечь масштаб из последней пары (s_{k-1}, y_{k-1}). Для квадратичной функции $y_{k-1} = As_{k-1}$, поэтому:

$$\gamma_k = \frac{s_{k-1}^\top y_{k-1}}{\|y_{k-1}\|^2} = \frac{s_{k-1}^\top A\, s_{k-1}}{\|A\, s_{k-1}\|^2}$$

Выбор масштаба γ_k

Рассмотрим квадратичную задачу $f(x) = \frac{1}{2}x^\top Ax$ с гессианом $A \succ 0$. Мы ищем $H_k^0 \approx A^{-1}$, но позволяем себе только скаляр, умноженный на единичную матрицу.

Наивный выбор $H_k^0 = I$ — плохая идея. Если собственные числа A равны $\lambda_1 = 1, \lambda_n = 1000$, то A^{-1} имеет собственные числа от 0.001 до 1. Единичная матрица — это даже не в том же масштабе.

Идея: извлечь масштаб из последней пары (s_{k-1}, y_{k-1}). Для квадратичной функции $y_{k-1} = As_{k-1}$, поэтому:

$$\gamma_k = \frac{s_{k-1}^\top y_{k-1}}{\|y_{k-1}\|^2} = \frac{s_{k-1}^\top A\, s_{k-1}}{\|A\, s_{k-1}\|^2}$$

Это отношение Рэлея-подобного типа. Если s_{k-1} — собственный вектор A с λ_i, то $\gamma_k = 1/\lambda_i$ — в точности нужный масштаб. Для общего s_{k-1} это взвешенное среднее $1/\lambda_i$, величина порядка $1/\bar{\lambda}(A)$.

Выбор масштаба γ_k

Рассмотрим квадратичную задачу $f(x) = \frac{1}{2}x^\top Ax$ с гессианом $A \succ 0$. Мы ищем $H_k^0 \approx A^{-1}$, но позволяем себе только скаляр, умноженный на единичную матрицу.

Наивный выбор $H_k^0 = I$ — плохая идея. Если собственные числа A равны $\lambda_1 = 1, \lambda_n = 1000$, то A^{-1} имеет собственные числа от 0.001 до 1. Единичная матрица — это даже не в том же масштабе.

Идея: извлечь масштаб из последней пары (s_{k-1}, y_{k-1}). Для квадратичной функции $y_{k-1} = As_{k-1}$, поэтому:

$$\gamma_k = \frac{s_{k-1}^\top y_{k-1}}{\|y_{k-1}\|^2} = \frac{s_{k-1}^\top A\, s_{k-1}}{\|A\, s_{k-1}\|^2}$$

Это отношение Рэлея-подобного типа. Если s_{k-1} — собственный вектор A с λ_i, то $\gamma_k = 1/\lambda_i$ — в точности нужный масштаб. Для общего s_{k-1} это взвешенное среднее $1/\lambda_i$, величина порядка $1/\bar{\lambda}(A)$.

> **i** Таким образом, $\gamma_k I$ задаёт правильный **масштаб** приближения A^{-1}: все элементы диагонали имеют верный порядок величины. А m пар (s_i, y_i) через ранг-2 обновления корректируют **форму** — анизотропию матрицы.

Алгоритм двух циклов: идея

Задача: вычислить $r = H_k \nabla f(x_k)$, используя только m пар $(s_i, y_i)_{i=k-m}^{k-1}$ и скаляр γ_k.

Алгоритм двух циклов: идея

Задача: вычислить $r = H_k \nabla f(x_k)$, используя только m пар $(s_i, y_i)_{i=k-m}^{k-1}$ и скаляр γ_k.

После усечения и подстановки $H_k^0 = \gamma_k I$:

$$H_k = \underbrace{V_{k-1}^\top \cdots V_{k-m}^\top}_{\text{«снять» обновления}} \quad \gamma_k I \quad \underbrace{V_{k-m} \cdots V_{k-1}}_{\text{«надеть» обновления}} \quad + \text{ранг-1 слагаемые}$$

Алгоритм двух циклов: идея

Задача: вычислить $r = H_k \nabla f(x_k)$, используя только m пар $(s_i, y_i)_{i=k-m}^{k-1}$ и скаляр γ_k.

После усечения и подстановки $H_k^0 = \gamma_k I$:

$$H_k = \underbrace{V_{k-1}^\top \cdots V_{k-m}^\top}_{\text{«снять» обновления}} \quad \gamma_k I \quad \underbrace{V_{k-m} \cdots V_{k-1}}_{\text{«надеть» обновления}} \quad + \text{ранг-1 слагаемые}$$

Чтобы вычислить $H_k g$ для $g = \nabla f(x_k)$, не формируя матрицу, применяем операторы V_i последовательно к вектору. Каждый $V_i g = g - \rho_i y_i (s_i^\top g)$ — это вычитание проекции, стоимость $O(n)$.

Алгоритм двух циклов: идея

Задача: вычислить $r = H_k \nabla f(x_k)$, используя только m пар $(s_i, y_i)_{i=k-m}^{k-1}$ и скаляр γ_k.

После усечения и подстановки $H_k^0 = \gamma_k I$:

$$H_k = \underbrace{V_{k-1}^\top \cdots V_{k-m}^\top}_{\text{«снять» обновления}} \quad \gamma_k I \quad \underbrace{V_{k-m} \cdots V_{k-1}}_{\text{«надеть» обновления}} \quad + \text{ранг-1 слагаемые}$$

Чтобы вычислить $H_k g$ для $g = \nabla f(x_k)$, не формируя матрицу, применяем операторы V_i последовательно к вектору. Каждый $V_i g = g - \rho_i y_i (s_i^\top g)$ — это вычитание проекции, стоимость $O(n)$.

Стратегия:

1. Цикл 1 (справа налево): последовательно «снимаем» обновления $V_{k-1}, V_{k-2}, \ldots, V_{k-m}$ с вектора g, запоминая коэффициенты $\alpha_i = \rho_i s_i^\top q$

Алгоритм двух циклов: идея

Задача: вычислить $r = H_k \nabla f(x_k)$, используя только m пар $(s_i, y_i)_{i=k-m}^{k-1}$ и скаляр γ_k.

После усечения и подстановки $H_k^0 = \gamma_k I$:

$$H_k = \underbrace{V_{k-1}^\top \cdots V_{k-m}^\top}_{\text{«снять» обновления}} \quad \gamma_k I \quad \underbrace{V_{k-m} \cdots V_{k-1}}_{\text{«надеть» обновления}} \quad + \text{ранг-1 слагаемые}$$

Чтобы вычислить $H_k g$ для $g = \nabla f(x_k)$, не формируя матрицу, применяем операторы V_i последовательно к вектору. Каждый $V_i g = g - \rho_i y_i (s_i^\top g)$ — это вычитание проекции, стоимость $O(n)$.

Стратегия:

1. Цикл 1 (справа налево): последовательно «снимаем» обновления $V_{k-1}, V_{k-2}, \ldots, V_{k-m}$ с вектора g, запоминая коэффициенты $\alpha_i = \rho_i s_i^\top q$
2. Середина: умножаем на $\gamma_k I$ — просто масштабирование

Алгоритм двух циклов: идея

Задача: вычислить $r = H_k \nabla f(x_k)$, используя только m пар $(s_i, y_i)_{i=k-m}^{k-1}$ и скаляр γ_k.

После усечения и подстановки $H_k^0 = \gamma_k I$:

$$H_k = \underbrace{V_{k-1}^\top \cdots V_{k-m}^\top}_{\text{«снять» обновления}} \quad \gamma_k I \quad \underbrace{V_{k-m} \cdots V_{k-1}}_{\text{«надеть» обновления}} \quad + \text{ранг-1 слагаемые}$$

Чтобы вычислить $H_k g$ для $g = \nabla f(x_k)$, не формируя матрицу, применяем операторы V_i последовательно к вектору. Каждый $V_i g = g - \rho_i y_i (s_i^\top g)$ — это вычитание проекции, стоимость $O(n)$.

Стратегия:

1. Цикл 1 (справа налево): последовательно «снимаем» обновления $V_{k-1}, V_{k-2}, \ldots, V_{k-m}$ с вектора g, запоминая коэффициенты $\alpha_i = \rho_i s_i^\top q$
2. Середина: умножаем на $\gamma_k I$ — просто масштабирование
3. Цикл 2 (слева направо): «надеваем» обновления обратно, используя запомненные α_i для коррекции

Алгоритм двух циклов (two-loop recursion)

$q = \nabla f(x_k)$

Цикл 1 (от новых к старым):

Для $i = k{-}1, k{-}2, \dots, k{-}m$:

$\alpha_i = \rho_i \, s_i^\top q$

$q = q - \alpha_i \, y_i$

$r = \gamma_k \, q$

Цикл 2 (от старых к новым):

Для $i = k{-}m, k{-}m{+}1, \dots, k{-}1$:

$\beta = \rho_i \, y_i^\top r$

$r = r + (\alpha_i - \beta) \, s_i$

$p_k = -r$

Алгоритм двух циклов (two-loop recursion)

$$q = \nabla f(x_k)$$

Цикл 1 (от новых к старым):

Для $i = k{-}1, k{-}2, \dots, k{-}m$:

$$\alpha_i = \rho_i \, s_i^\top q$$

$$q = q - \alpha_i \, y_i$$

$$r = \gamma_k \, q$$

Цикл 2 (от старых к новым):

Для $i = k{-}m, k{-}m{+}1, \dots, k{-}1$:

$$\beta = \rho_i \, y_i^\top r$$

$$r = r + (\alpha_i - \beta) \, s_i$$

$$p_k = -r$$

Каждая итерация цикла — одно скалярное произведение ($O(n)$) и одна операция axpy ($O(n)$). Два цикла по m итераций: стоимость $4mn$ умножений — **линейно** по n, сопоставимо с градиентным спуском.

Свойства L-BFGS

Вычислительная сложность:

- Память: $O(mn)$ вместо $O(n^2)$

Свойства L-BFGS

Вычислительная сложность:

- Память: $O(mn)$ вместо $O(n^2)$
- Вычисление $p_k = -H_k \nabla f(x_k)$: $O(mn)$

Свойства L-BFGS

Вычислительная сложность:

- Память: $O(mn)$ вместо $O(n^2)$
- Вычисление $p_k = -H_k \nabla f(x_k)$: $O(mn)$
- Типичные значения m: от 3 до 20

Свойства L-BFGS

Вычислительная сложность:

- Память: $O(mn)$ вместо $O(n^2)$
- Вычисление $p_k = -H_k \nabla f(x_k)$: $O(mn)$
- Типичные значения m: от 3 до 20
- Стоимость итерации сопоставима с GD

Свойства L-BFGS

Вычислительная сложность:

- Память: $O(mn)$ вместо $O(n^2)$
- Вычисление $p_k = -H_k \nabla f(x_k)$: $O(mn)$
- Типичные значения m: от 3 до 20
- Стоимость итерации сопоставима с GD

Свойства L-BFGS

Вычислительная сложность:

- Память: $O(mn)$ вместо $O(n^2)$
- Вычисление $p_k = -H_k \nabla f(x_k)$: $O(mn)$
- Типичные значения m: от 3 до 20
- Стоимость итерации сопоставима с GD

Практические рекомендации:

- $m = 10$ - хорошее значение по умолчанию

Свойства L-BFGS

Вычислительная сложность:

- Память: $O(mn)$ вместо $O(n^2)$
- Вычисление $p_k = -H_k \nabla f(x_k)$: $O(mn)$
- Типичные значения m: от 3 до 20
- Стоимость итерации сопоставима с GD

Практические рекомендации:

- $m = 10$ - хорошее значение по умолчанию
- Увеличение m улучшает аппроксимацию, но замедляет итерацию

Свойства L-BFGS

Вычислительная сложность:

- Память: $O(mn)$ вместо $O(n^2)$
- Вычисление $p_k = -H_k \nabla f(x_k)$: $O(mn)$
- Типичные значения m: от 3 до 20
- Стоимость итерации сопоставима с GD

Практические рекомендации:

- $m = 10$ - хорошее значение по умолчанию
- Увеличение m улучшает аппроксимацию, но замедляет итерацию
- γ_k адаптирует масштаб автоматически на каждой итерации

Свойства L-BFGS

Вычислительная сложность:

- Память: $O(mn)$ вместо $O(n^2)$
- Вычисление $p_k = -H_k \nabla f(x_k)$: $O(mn)$
- Типичные значения m: от 3 до 20
- Стоимость итерации сопоставима с GD

Практические рекомендации:

- $m = 10$ - хорошее значение по умолчанию
- Увеличение m улучшает аппроксимацию, но замедляет итерацию
- γ_k адаптирует масштаб автоматически на каждой итерации

Свойства L-BFGS

Вычислительная сложность:

- Память: $O(mn)$ вместо $O(n^2)$
- Вычисление $p_k = -H_k \nabla f(x_k)$: $O(mn)$
- Типичные значения m: от 3 до 20
- Стоимость итерации сопоставима с GD

Практические рекомендации:

- $m = 10$ - хорошее значение по умолчанию
- Увеличение m улучшает аппроксимацию, но замедляет итерацию
- γ_k адаптирует масштаб автоматически на каждой итерации

Сходимость:

- Глобальная: $\nabla f(x_k) \to 0$ с условиями Вульфа

Свойства L-BFGS

Вычислительная сложность:

- Память: $O(mn)$ вместо $O(n^2)$
- Вычисление $p_k = -H_k \nabla f(x_k)$: $O(mn)$
- Типичные значения m: от 3 до 20
- Стоимость итерации сопоставима с GD

Практические рекомендации:

- $m = 10$ - хорошее значение по умолчанию
- Увеличение m улучшает аппроксимацию, но замедляет итерацию
- γ_k адаптирует масштаб автоматически на каждой итерации

Сходимость:

- Глобальная: $\nabla f(x_k) \to 0$ с условиями Вульфа
- При фиксированном $m \ll n$: **линейная** скорость сходимости

Свойства L-BFGS

Вычислительная сложность:

- Память: $O(mn)$ вместо $O(n^2)$
- Вычисление $p_k = -H_k \nabla f(x_k)$: $O(mn)$
- Типичные значения m: от 3 до 20
- Стоимость итерации сопоставима с GD

Практические рекомендации:

- $m = 10$ - хорошее значение по умолчанию
- Увеличение m улучшает аппроксимацию, но замедляет итерацию
- γ_k адаптирует масштаб автоматически на каждой итерации

Сходимость:

- Глобальная: $\nabla f(x_k) \to 0$ с условиями Вульфа
- При фиксированном $m \ll n$: **линейная** скорость сходимости
- При $m = n$: эквивалентен полному BFGS $\to$ сверхлинейная сходимость

Свойства L-BFGS

Вычислительная сложность:

- Память: $O(mn)$ вместо $O(n^2)$
- Вычисление $p_k = -H_k \nabla f(x_k)$: $O(mn)$
- Типичные значения m: от 3 до 20
- Стоимость итерации сопоставима с GD

Практические рекомендации:

- $m = 10$ - хорошее значение по умолчанию
- Увеличение m улучшает аппроксимацию, но замедляет итерацию
- γ_k адаптирует масштаб автоматически на каждой итерации

Сходимость:

- Глобальная: $\nabla f(x_k) \to 0$ с условиями Вульфа
- При фиксированном $m \ll n$: **линейная** скорость сходимости
- При $m = n$: эквивалентен полному BFGS $\to$ сверхлинейная сходимость
- На практике: значительно быстрее GD, сопоставимо с BFGS

Свойства L-BFGS

Вычислительная сложность:

- Память: $O(mn)$ вместо $O(n^2)$
- Вычисление $p_k = -H_k \nabla f(x_k)$: $O(mn)$
- Типичные значения m: от 3 до 20
- Стоимость итерации сопоставима с GD

Практические рекомендации:

- $m = 10$ - хорошее значение по умолчанию
- Увеличение m улучшает аппроксимацию, но замедляет итерацию
- γ_k адаптирует масштаб автоматически на каждой итерации

Сходимость:

- Глобальная: $\nabla f(x_k) \to 0$ с условиями Вульфа
- При фиксированном $m \ll n$: **линейная** скорость сходимости
- При $m = n$: эквивалентен полному BFGS $\to$ сверхлинейная сходимость
- На практике: значительно быстрее GD, сопоставимо с BFGS

Свойства L-BFGS

Вычислительная сложность:

- Память: $O(mn)$ вместо $O(n^2)$
- Вычисление $p_k = -H_k \nabla f(x_k)$: $O(mn)$
- Типичные значения m: от 3 до 20
- Стоимость итерации сопоставима с GD

Практические рекомендации:

- $m = 10$ - хорошее значение по умолчанию
- Увеличение m улучшает аппроксимацию, но замедляет итерацию
- γ_k адаптирует масштаб автоматически на каждой итерации

Сходимость:

- Глобальная: $\nabla f(x_k) \to 0$ с условиями Вульфа
- При фиксированном $m \ll n$: **линейная** скорость сходимости
- При $m = n$: эквивалентен полному BFGS $\to$ сверхлинейная сходимость
- На практике: значительно быстрее GD, сопоставимо с BFGS

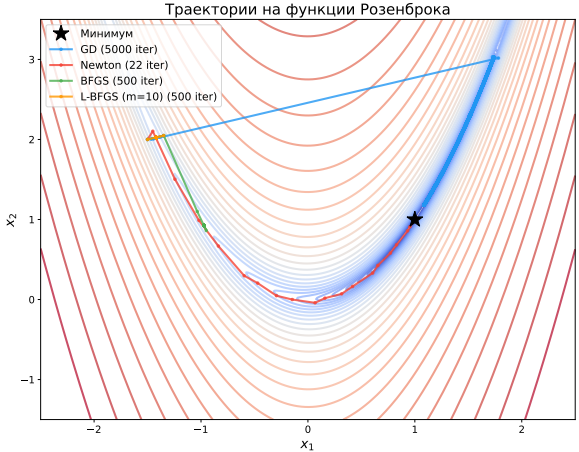
Сравнение методов:

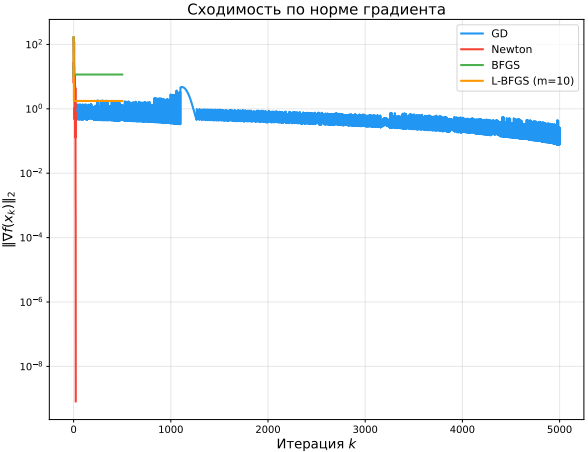
	GD	L-BFGS	BFGS	Newton
Память	$O(n)$	$O(mn)$	$O(n^2)$	$O(n^2)$
Итерация	$O(n)$	$O(mn)$	$O(n^2)$	$O(n^3)$
Сходимость	Лин.	Лин.	Сверхлин.	Квадр.

Эксперименты

Сравнение методов на функции Розенброка

$f(x_1, x_2) = (1 - x_1)^2 + 100(x_2 - x_1^2)^2$. Начальная точка $x_0 = (-1.5, 2)$.

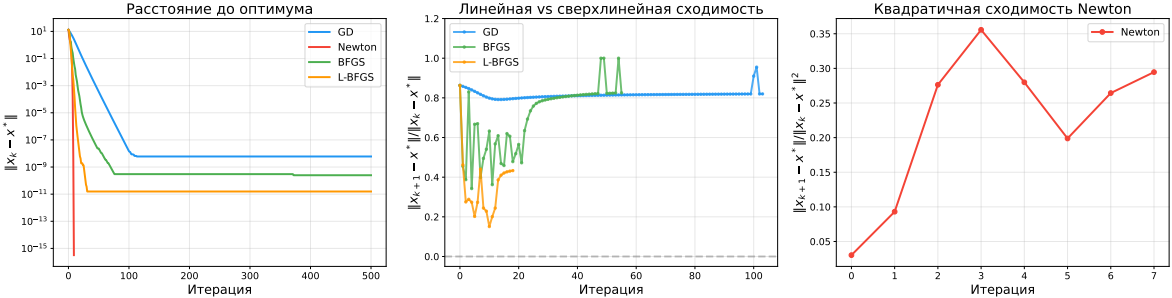




Сверхлинейная сходимость BFGS

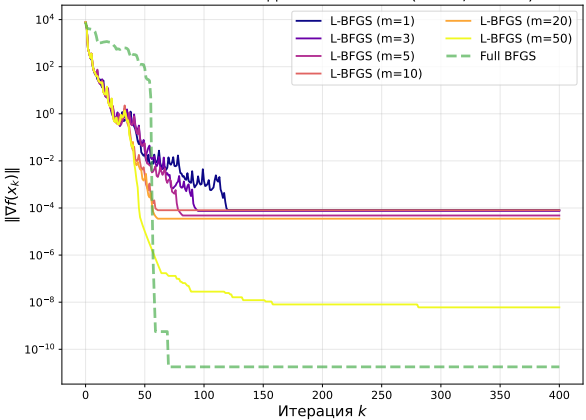
Логистическая регрессия, $n = 30$, $\lambda = 0.1$. Теория предсказывает: GD - линейная $\frac{\|x_{k+1} - x^*\|}{\|x_k - x^*\|} \to c$, BFGS - сверхлинейная $\frac{\|x_{k+1} - x^*\|}{\|x_k - x^*\|} \to 0$, Newton - квадратичная $\frac{\|x_{k+1} - x^*\|}{\|x_k - x^*\|^2} \to C$.

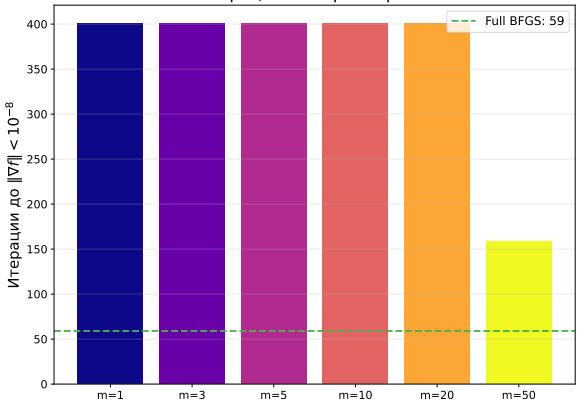
Скорости сходимости (логистическая регрессия, n=30)



Влияние параметра памяти m в L-BFGS

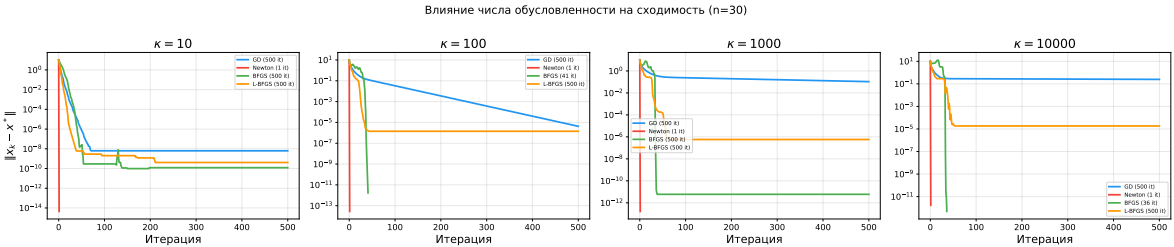
Квадратичная функция, $n = 50$, $\kappa = 500$. При $m = 1$ L-BFGS сходится медленно, при $m = 50 = n$ совпадает с полным BFGS.





Влияние числа обусловленности κ

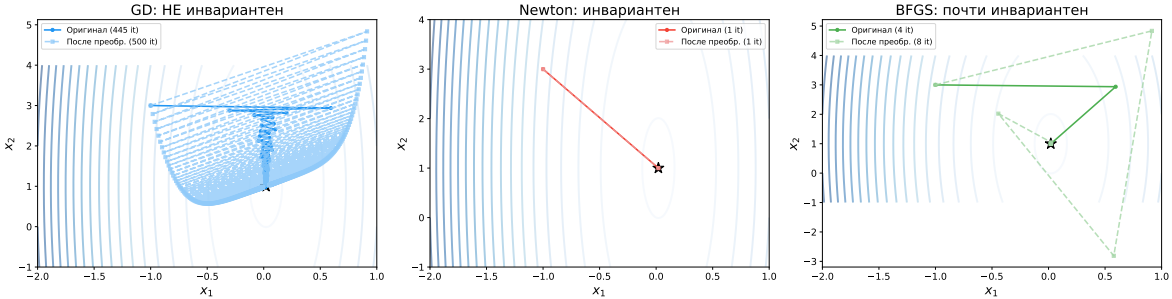
Квадратичная функция, $n = 30$. GD деградирует с ростом κ, метод Ньютона инвариантен - число итераций не зависит от κ.



Аффинная инвариантность

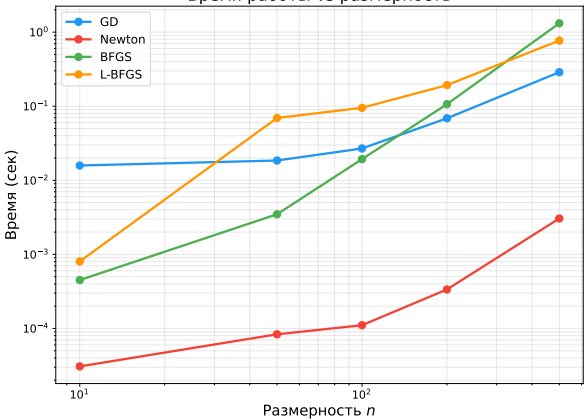
$f(x) = \frac{1}{2}x^\top \text{diag}(50, 1)\, x$, замена координат $x = Ty$, $T = \begin{pmatrix} 2 & 1 \\ 1 & 3 \end{pmatrix}$, $g(y) = f(Ty)$. Запускаем метод на $g(y)$ в y-координатах, затем переводим траекторию обратно: $x_k = Ty_k$. Если метод аффинно инвариантен, обе траектории (в x-пространстве) совпадают.

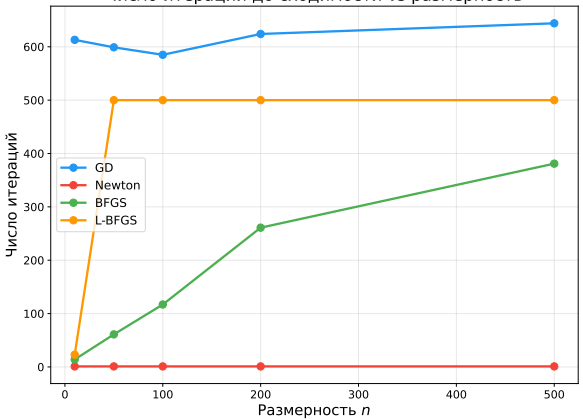
Аффинная инвариантность: траектории в оригинальных и преобразованных координатах



Масштабируемость: время vs размерность

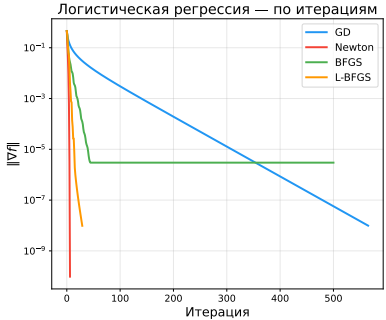
Квадратичная функция, $\kappa = 100$. Newton: $O(n^3)$ на итерацию, но минимум итераций. L-BFGS масштабируется линейно по n.

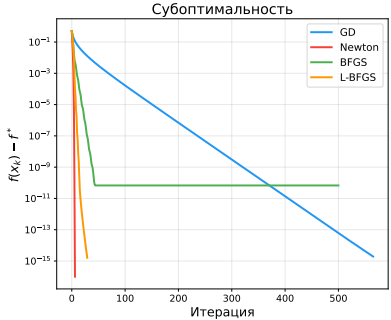


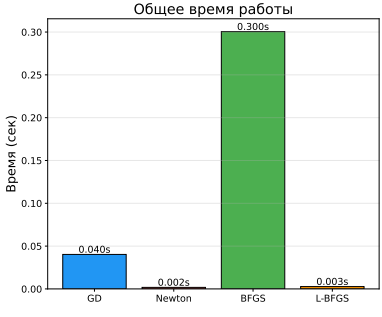


Логистическая регрессия

Бинарная логистическая регрессия с ℓ_2-регуляризацией ($\lambda = 0.01$), $n = 50$ признаков, $N = 300$ наблюдений.

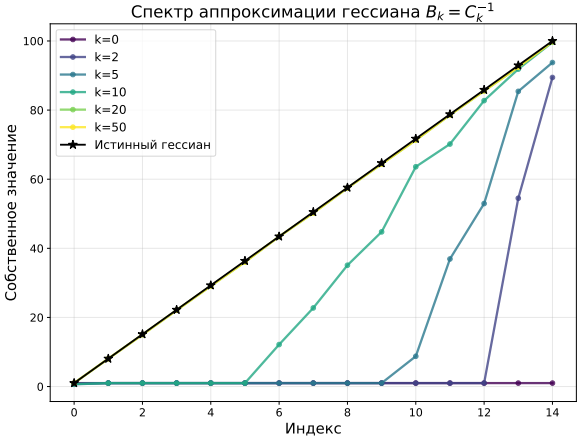


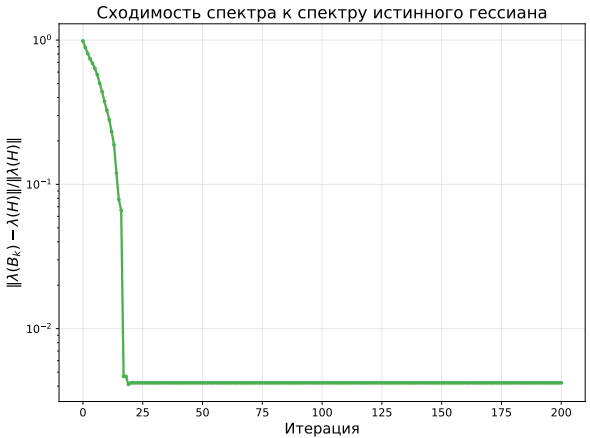




Спектр аппроксимации гессиана в BFGS

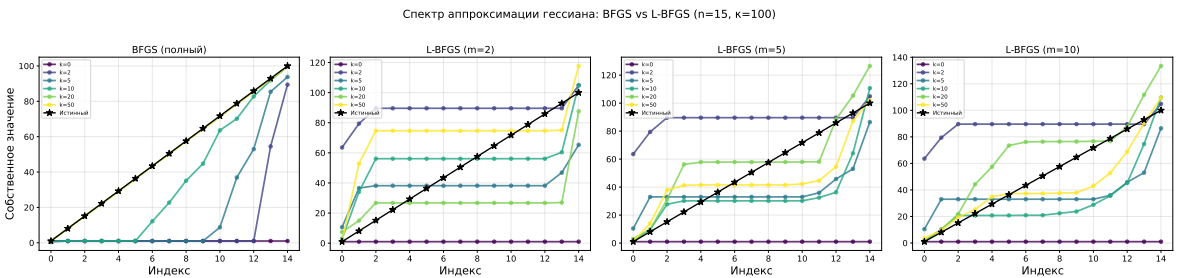
Квадратичная функция, $n = 15$, $\kappa = 100$. Спектр $B_k = H_k^{-1}$ сходится к спектру истинного гессиана.





Спектр аппроксимации гессиана: BFGS vs L-BFGS

Квадратичная функция, $n = 15$, $\kappa = 100$. Полный BFGS восстанавливает весь спектр гессиана. L-BFGS с памятью m аппроксимирует только m последних направлений кривизны - при малом m спектр B_k далёк от истинного, при $m = n$ совпадает с полным BFGS.



Код

- Открыть в Colab

Код

- Открыть в Colab
- Сравнение квазиньютоновских методов

Код

- Открыть в Colab
- Сравнение квазиньютоновских методов
- Некоторые практические замечания о методе Ньютона