Методы оптимизации. Семинар 5.

Методы градиентного спуска, Ньютона и сопряженных градиентов. Подготовка к практическому заданию 1.

ФКН ВШЭ

7 февраля 2017

Методы спуска. Общая схема

Рассматриваемая задача: $\min_{x \in \mathbb{R}^n} f(x)$.

Общая схема метода спуска:

1. Выбрать направление спуска $d_k \in \mathbb{R}^n$: $\nabla f(x_k)^T d_k < 0$.
2. *(Линейный поиск)* Выбрать длину шага $\alpha_k \geq 0$.
3. *(Обновление)* $x_{k+1} \leftarrow x_k + \alpha_k d_k$

Линейный поиск

Линейный поиск: $\phi_k(\alpha) := f(x_k + \alpha_k d_k), \qquad \alpha \geq 0.$
Типичные стратегии выбора шага:

- **Постоянный шаг:** $\alpha_k = \text{const}$.
- **Бэктрэкинг:** Начать с $\alpha := \alpha_0$ и уменьшать вдвое, пока не выполнится **условие Армихо** ($c_1 \in (0, 0.5)$)
$$\phi_k(\alpha) \leq \phi_k(0) + c_1 \alpha \phi_k'(0).$$
- **Сильные условия Вульфа** ($c_1 \in (0, 0.5)$, $c_1 < c_2 < 1$):
$$\phi_k(\alpha) \leq \phi_k(0) + c_1 \alpha \phi_k'(0)$$
$$|\phi_k'(\alpha)| \leq c_2 |\phi_k'(0)|$$
Типичные методы применяют квадратичные/кубические интерполяции/экстраполяции.

Градиентный спуск

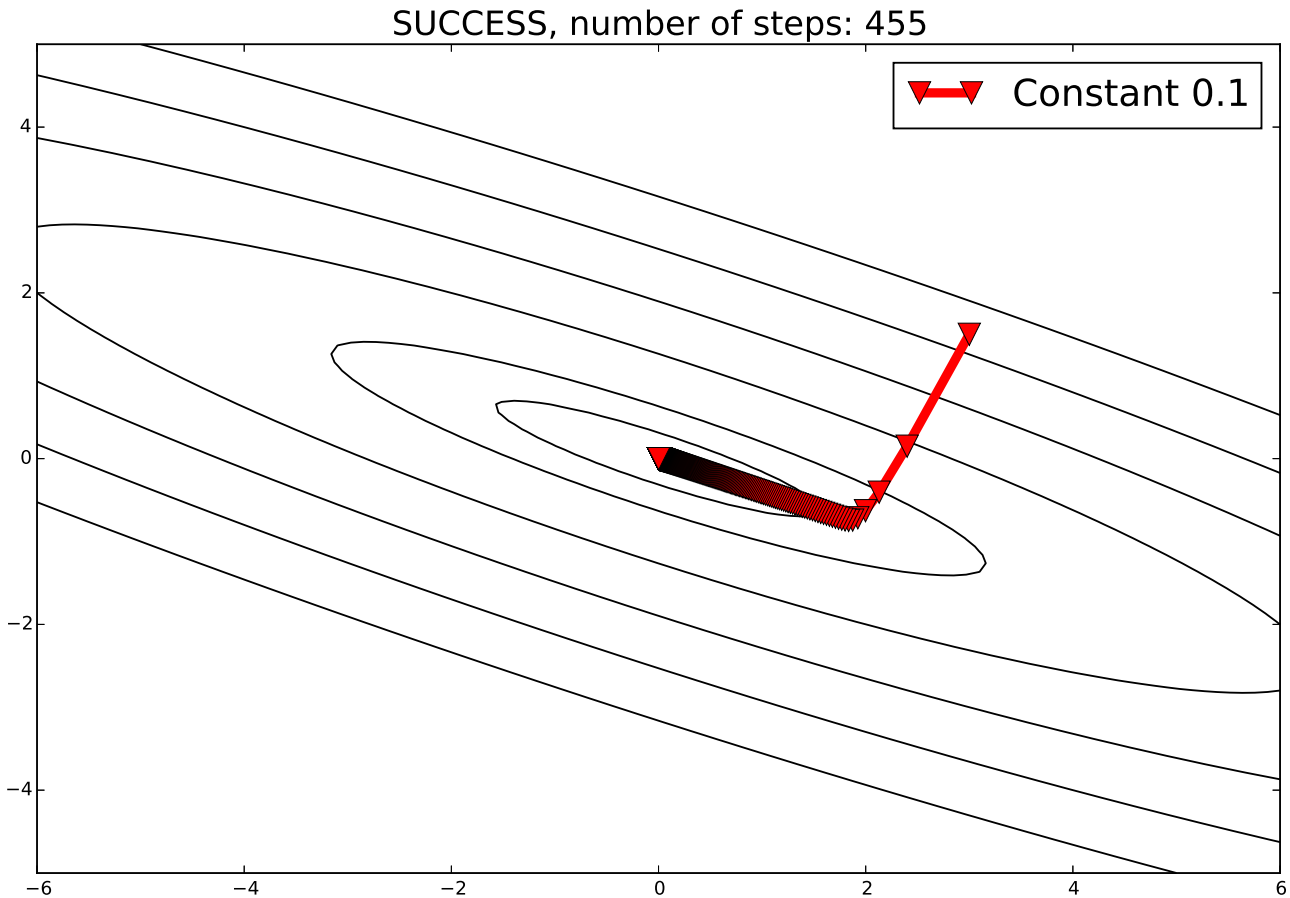
Градиентный спуск:

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

Может рассматриваться как метод спуска с направлением $d_k = -\nabla f(x_k)$.

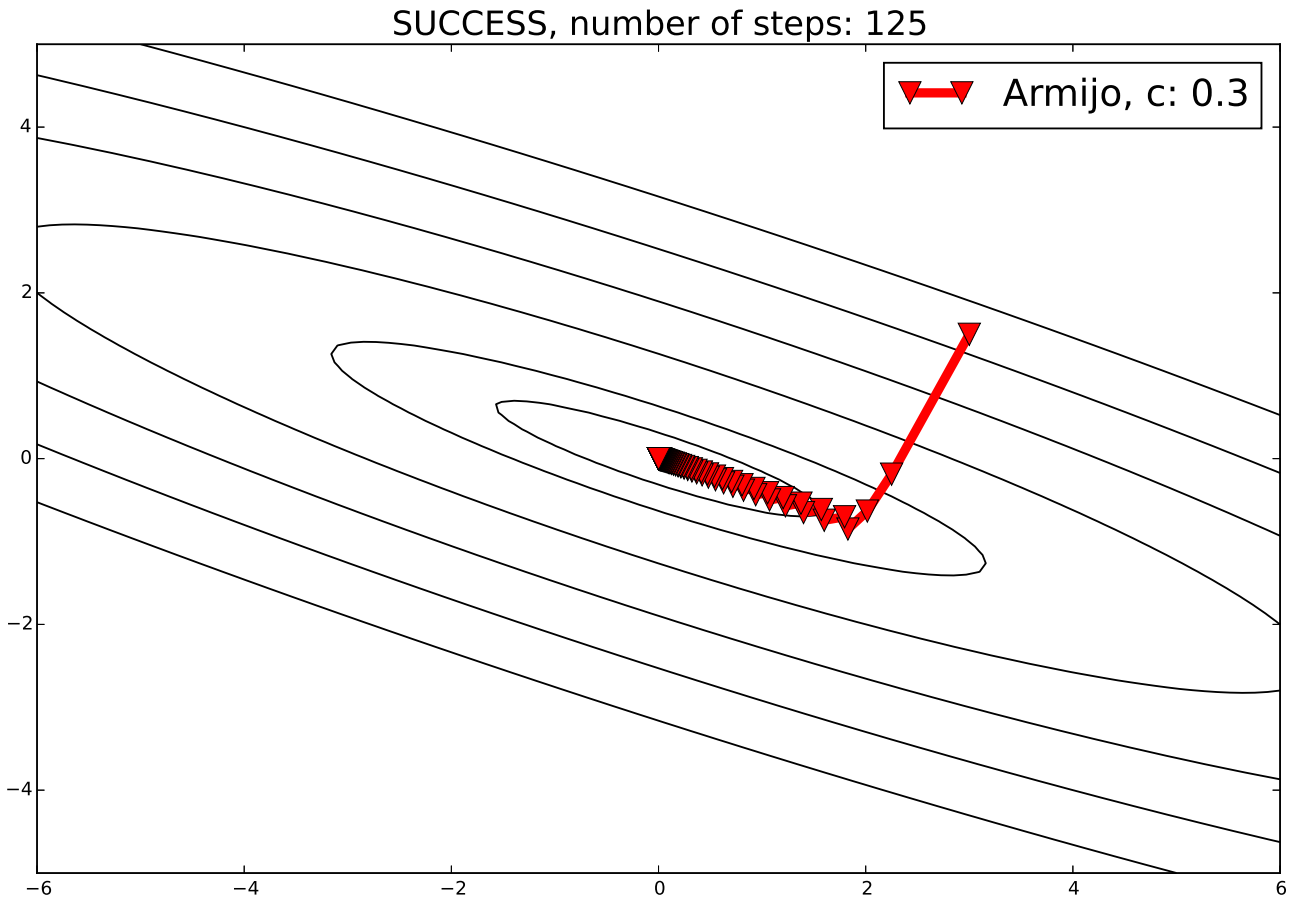
Градиентный спуск: константный шаг

Квадратичная функция: $f(x) = \frac{1}{2}x^T Ax, \quad A = \begin{pmatrix} 1 & 2 \\ 2 & 5 \end{pmatrix}.$



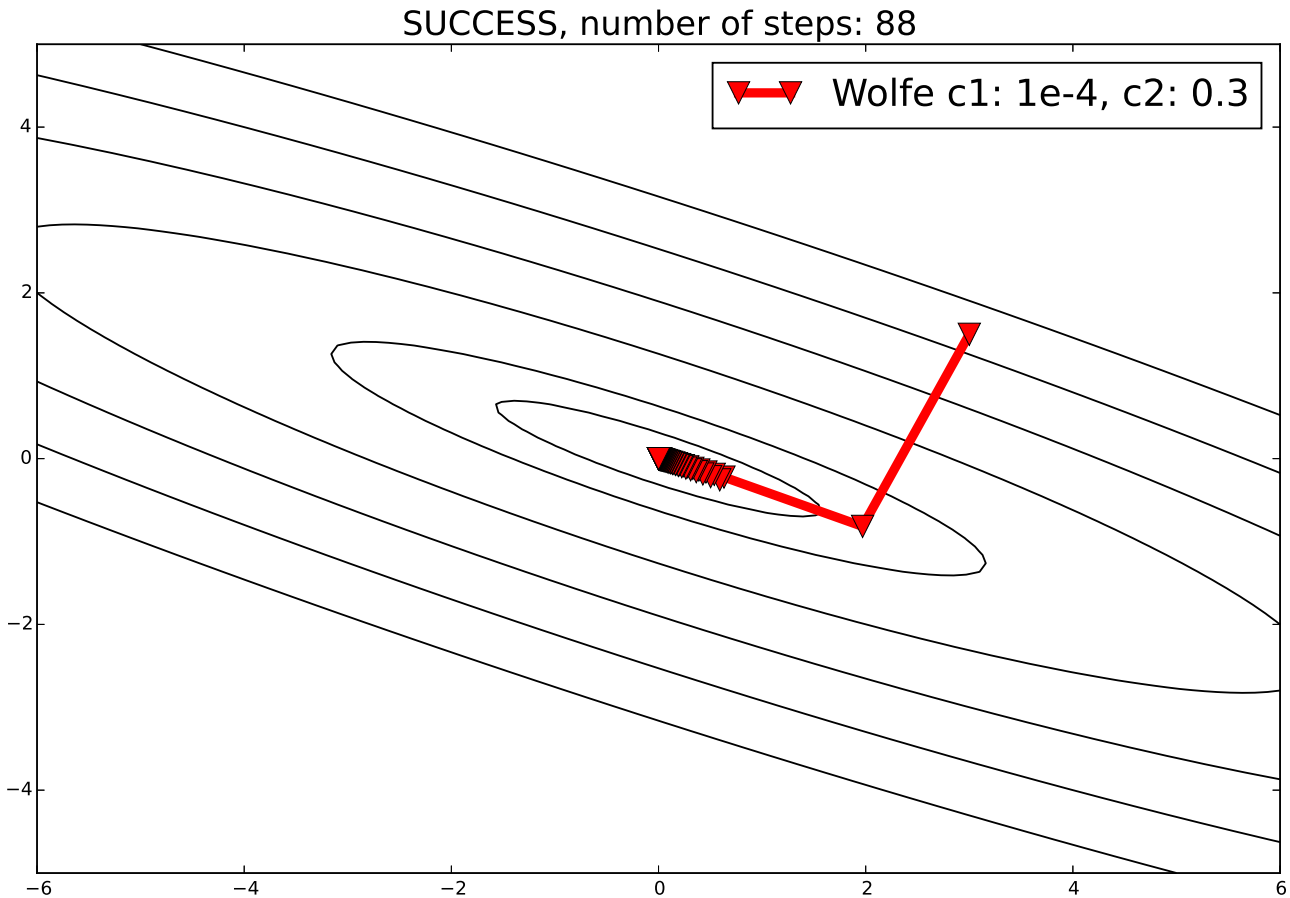
Градиентный спуск: бэктрекинг

Квадратичная функция: $f(x) = \frac{1}{2}x^T Ax, \quad A = \begin{pmatrix} 1 & 2 \\ 2 & 5 \end{pmatrix}.$



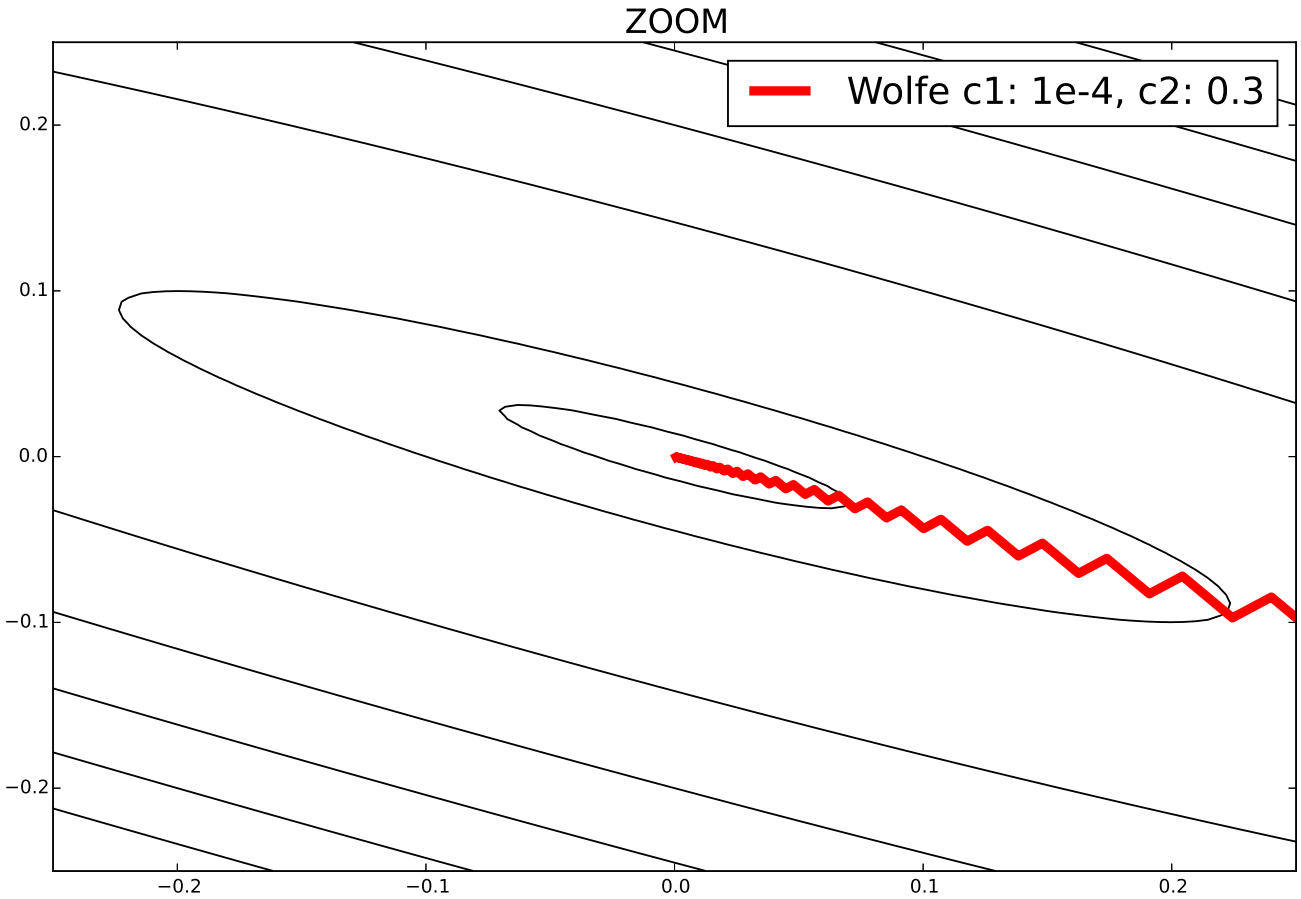
Градиентный спуск: стратегия Вульфа

Квадратичная функция: $f(x) = \frac{1}{2} x^T A x, \quad A = \begin{pmatrix} 1 & 2 \\ 2 & 5 \end{pmatrix}.$



Градиентный спуск: стратегия Вульфа

Квадратичная функция: $f(x) = \frac{1}{2} x^T A x, \quad A = \begin{pmatrix} 1 & 2 \\ 2 & 5 \end{pmatrix}.$

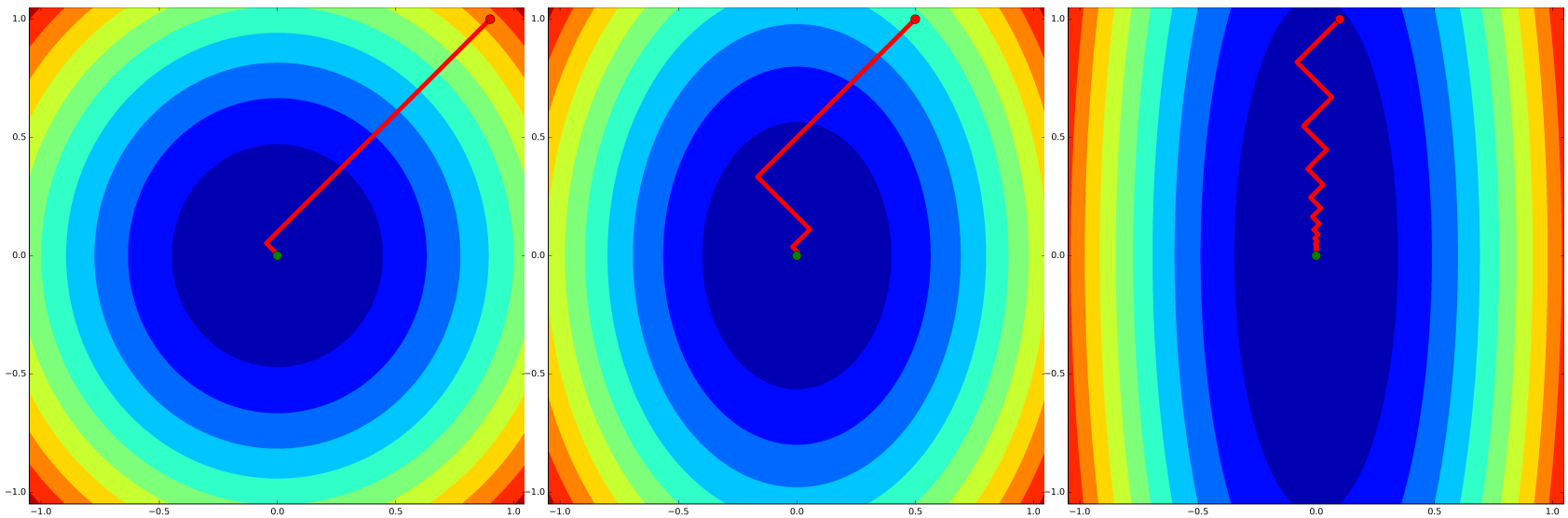


Градиентный спуск: число обусловленности

Квадратичная функция:

$$f(x) = \frac{1}{2}x^T Ax - b^T x, \qquad A \in \mathbb{S}^n_{++}, b \in \mathbb{R}^n.$$

Число обусловленности: $\kappa = \frac{\lambda_{\max}(A)}{\lambda_{\min}(A)} \geq 1.$

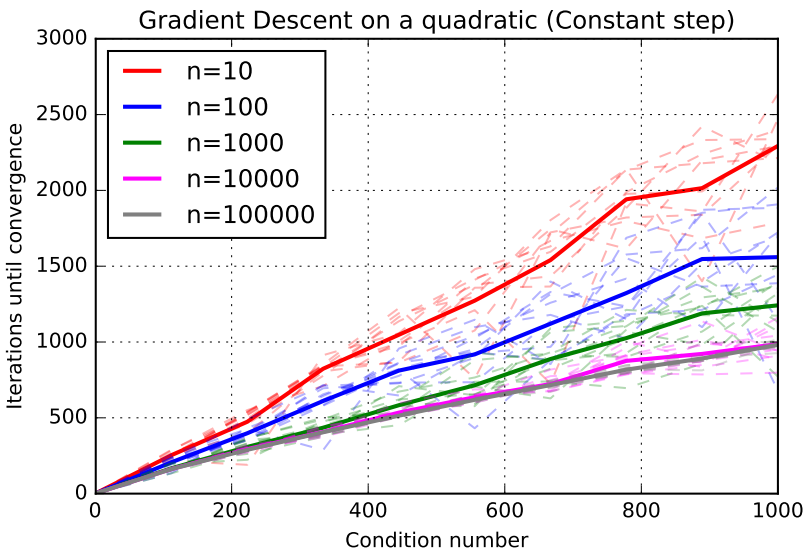


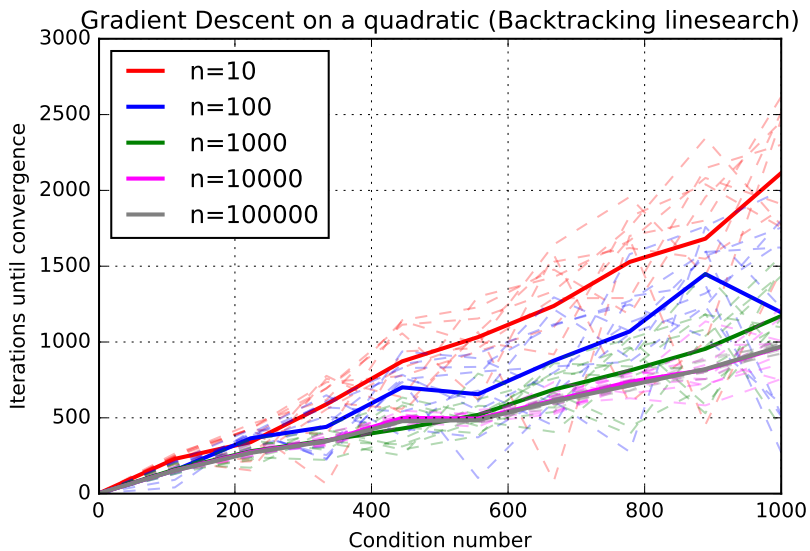
Плохая обусловленность + неудачный старт = **зигзаг**

Зависимость градиентного спуска от числа обусловленности и размерности задачи

Квадратичная функция: $f(x) = \frac{1}{2}x^T Ax - b^T x$.

$$A = \mathrm{Diag}(a), \qquad b \sim \mathcal{N}(0, I_n), \qquad a_i = \begin{cases} 1, & i = 1 \\ \sim \mathrm{Unif}(1, \kappa), & 2 \le i \le n-1 \\ \kappa, & i = n \end{cases}$$

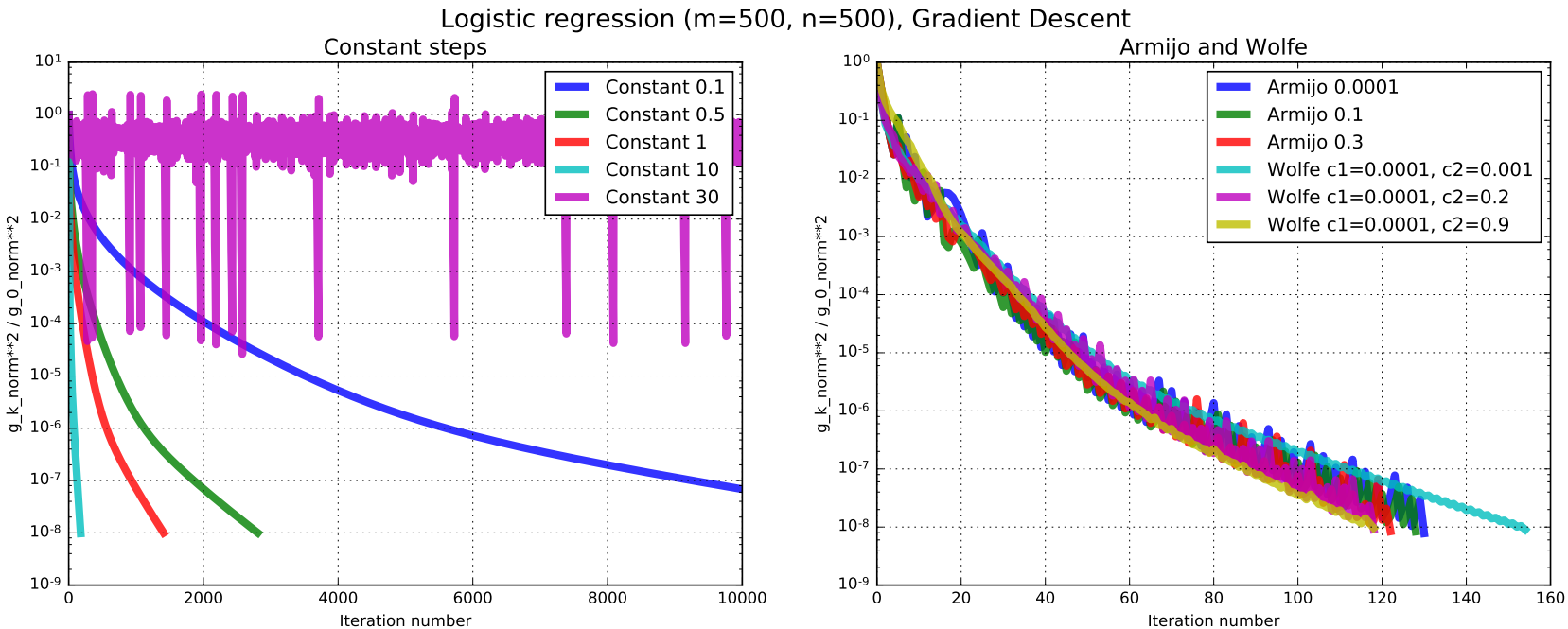




Стратегии выбора длины шага в градиентном спуске

Логистическая регрессия с ℓ_2-регуляризатором:

$$f(x) := \frac{1}{m}\sum_{i=1}^{m} \ln(1 + \exp(-b_i a_i^T x)) + \frac{\lambda}{2}\|x\|_2^2 \quad \to \quad \min_{x \in \mathbb{R}^n}.$$



Учет структуры функции

Рассматриваемая задача: $\min_{x \in \mathbb{R}^n} f(x)$.
Пусть $f(x) := \psi(Ax)$, где

1. $A \in \mathbb{R}^{m \times n}$: некоторая матрица.
2. Функция $\psi : \mathbb{R}^m \to \mathbb{R}$ считается за $O(m)$.

Примеры:

1. Квадратичная функция: $f(x) := \frac{1}{2} x^T A x - b^T x$. Здесь $\psi(y) := \frac{1}{2} x^T y - b^T x$.
2. Логистическая регрессия:

$$f(x) := \sum_{i=1}^{m} \ln(1 + \exp(-b_i a_i^T x))$$

   В этом случае

$$\phi(y) := \sum_{i=1}^{m} \ln(1 + \exp(-b_i y_i)).$$

Учет структуры функции – 2

Рассматриваемая ситуация: $f(x) = \psi(Ax)$.
Линейный поиск: $\phi_k(\alpha) := f(x_k + \alpha d_k)$. Производная:

$$\phi_k'(\alpha) = \nabla f(x_k + \alpha d_k)^T d_k, \qquad \nabla f(x) = A^T \nabla \psi(Ax).$$

Для произвольного $\alpha \geq 0$ нужно считать:

1. (Армихо) Значение функции $\phi_k(\alpha)$.
2. (Вульф) Значение функции $\phi_k(\alpha)$ и производной $\phi_k'(\alpha)$.

Если считать «в лоб», то значение функции и производной стоит $O(mn)$. Если их нужно посчитать для s разных значений α, то стоимость составляет $O(smn)$.
Учтем структуру функции:

$$\phi_k(\alpha) = \psi(Ax_k + \alpha Ad_k), \qquad \phi_k'(\alpha) = \nabla \psi(Ax_k + \alpha Ad_k)^T Ad_k.$$

В этом случае нужно один раз посчитать и запомнить Ax_k, Ad_k (стоимость $O(mn)$). Дальнейшее вычисление ϕ_k и ϕ_k' для произвольного α стоит $O(m)$. Итого суммарная стоимость $O(mn)$.

Детали реализации

Рассматриваемая задача: $\min_{x \in \mathbb{R}^n} \{f(x) := \psi(Ax)\}$.

Типичная структура метода спуска:

for $k \leftarrow 0, \ldots, K$ **do**

Вызвать оракул в точке x_k:

$f(x_k) = \psi(Ax_k)$, $\nabla f(x_k) = A^T \nabla \psi(Ax_k)$ и пр.;

Вычислить направление спуска d_k (оракул не вызывается);

Выполнить линейный поиск для нахождения длины шага:

$\phi(0) = \psi(Ax_k)$, $\phi'(0) = \nabla \psi(Ax_k)^T Ad_k$;

$\phi(\bar{\alpha}_1) = \psi(Ax_k + \bar{\alpha}_1 Ad_k)$,

$\phi'(\bar{\alpha}_1) = \nabla \psi(Ax_k + \bar{\alpha}_1 Ad_k)^T Ad_k$;

$\ldots$;

$\phi(\bar{\alpha}_s) = \psi(Ax_k + \bar{\alpha}_s Ad_k)$, $\phi'(\bar{\alpha}_s) = \nabla \psi(Ax_k + \bar{\alpha}_s Ad_k)^T Ad_k$;

$x_{k+1} \leftarrow x_k + \bar{\alpha}_s d_k$; // $Ax_{k+1} = Ax_k + \bar{\alpha}_s Ad_k$

end

NB: Последовательные вызовы используют много одинаковой информации: Ax_k, Ad_k. Имеет смысл запоминать эти величины.

Детали реализации – 2

$$f(x) := \frac{1}{3}\|Ax\|_2^3, \qquad \nabla f(x) = \|Ax\|_2 A^T Ax.$$

```
class CubicOracle(BaseSmoothOracle):
    def __init__(self, A):
        self.A
        self.last_x = None

    def func(self, x):
        self._update_Ax(x); return 1.0 / 3 * self.Ax_norm ** 3.0

    def grad(self, x):
        self._update_Ax(x); return self.Ax_norm * self.A.T.dot(self.Ax)

    def _update_Ax(self, x):
        if not np.array_equal(self.last_x, x):
            self.last_x = np.copy(x)
            self.Ax, self.Ax_norm = self.A.dot(x), np.linalg.norm(self.Ax)
```

Чистый (классический) метод Ньютона

Задача: $\min_{x \in \mathbb{R}^n} f(x)$.
Чистый метод Ньютона:

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

Откуда берется формула? Рассмотрим квадратичную модель:

$$f(x+h) \approx f(x) + \nabla f(x)^T h + \frac{1}{2} h^T \nabla^2 f(x) h.$$

Точка минимума модели по $h \in \mathbb{R}^n$ равна $-[\nabla^2 f(x_k)]^{-1} \nabla f(x_k)$.
Чистый метод Ньютона (в невырожденной ситуации) имеет квадратичную локальную сходимость.
NB: Для плохих начальных приближений x_0 ничего не гарантируется. Метод может расходиться.

Демпфированный метод Ньютона

Демпфированный метод Ньютона:

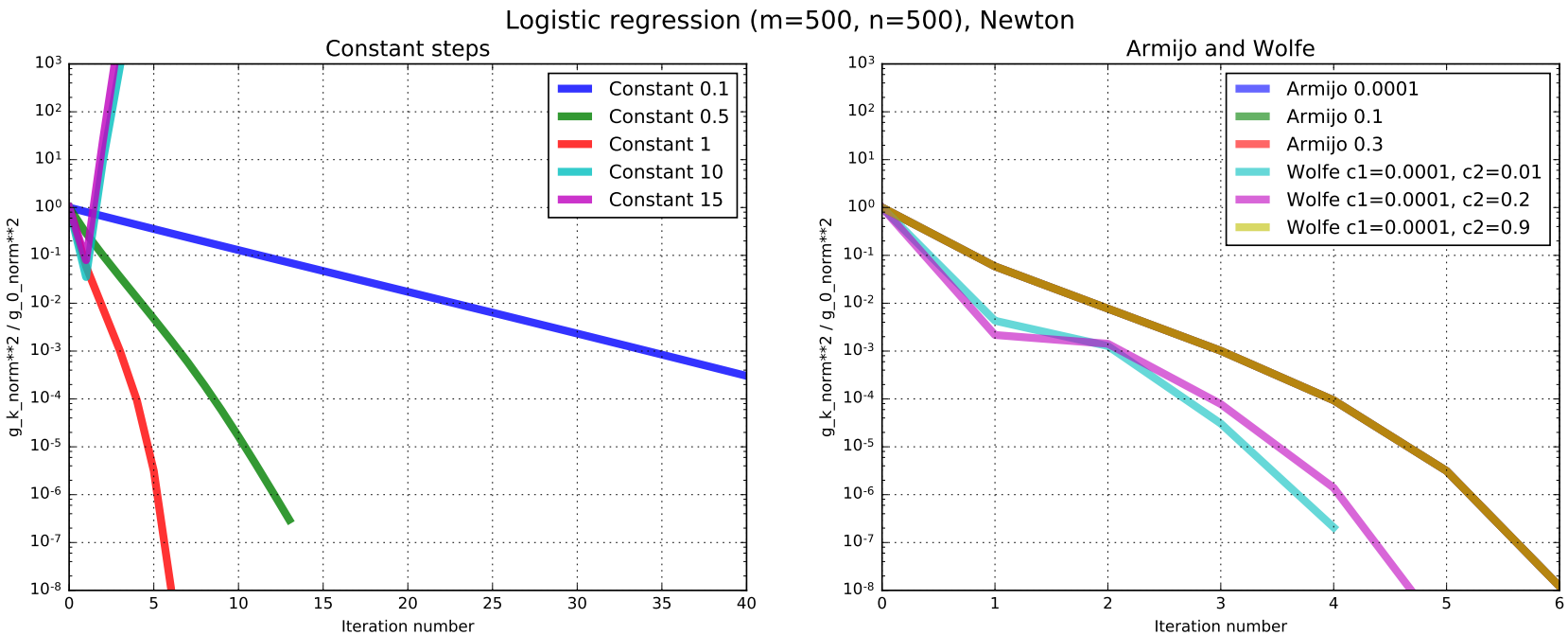
$$x_{k+1} = x_k - \alpha_k [\nabla^2 f(x_k)]^{-1} \nabla f(x_k).$$

- Добавляется «демпфирующая» длина шага α_k. Обычно $\alpha_k \in [0, 1]$.
- Если $\nabla^2 f(x_k) \succ 0$, то $d_k := -[\nabla^2 f(x_k)]^{-1} \nabla f(x_k)$ будет направлением спуска:
$$\nabla f(x_k)^T d_k = -\nabla f(x_k)^T [\nabla^2 f(x_k)]^{-1} \nabla f(x_k) \leq 0.$$
- Если $\nabla^2 f(x_k) \not\succ 0$, то применяют модификацию гессиана.
- Таким образом, демпфированный метод Ньютона погружается в класс методов спуска. Длина шага α_k настраивается с помощью линейного поиска.
- Важный момент: линейный поиск всегда нужно начинать с $\alpha_0 = 1$. Иначе не будет квадратичной сходимости.

Стратегии выбора шага в методе Ньютона

Логистическая регрессия с ℓ_2-регуляризатором:

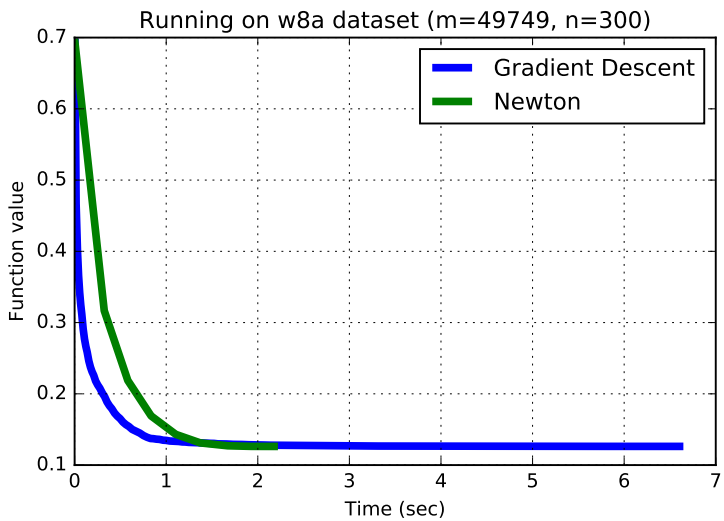
$$f(x) := \frac{1}{m}\sum_{i=1}^{m} \ln(1 + \exp(-b_i a_i^T x)) + \frac{\lambda}{2}\|x\|_2^2 \quad \to \quad \min_{x \in \mathbb{R}^n}.$$

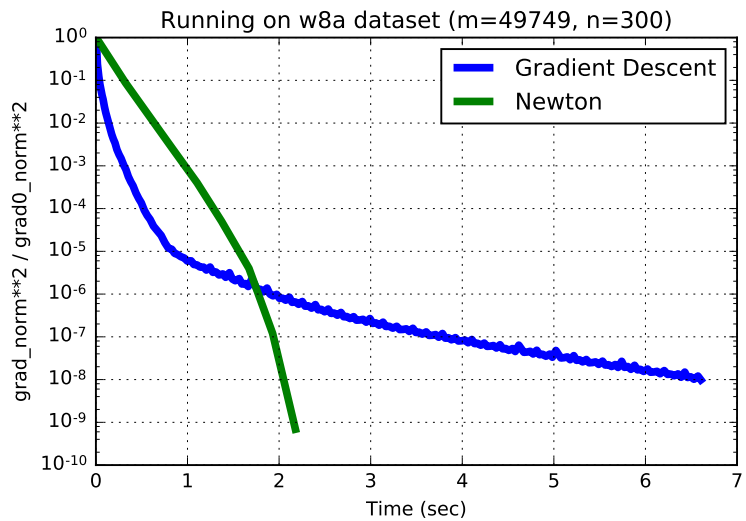


Сравнение градиентного спуска и Ньютона

Логистическая регрессия с ℓ_2-регуляризатором:

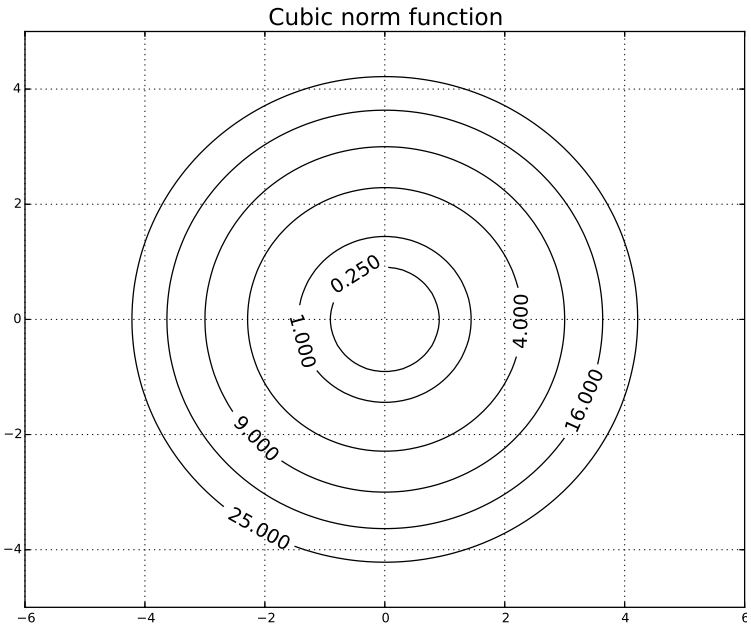
$$f(x) := \frac{1}{m}\sum_{i=1}^{m} \ln(1 + \exp(-b_i a_i^T x)) + \frac{\lambda}{2}\|x\|_2^2 \quad \to \quad \min_{x \in \mathbb{R}^n}.$$

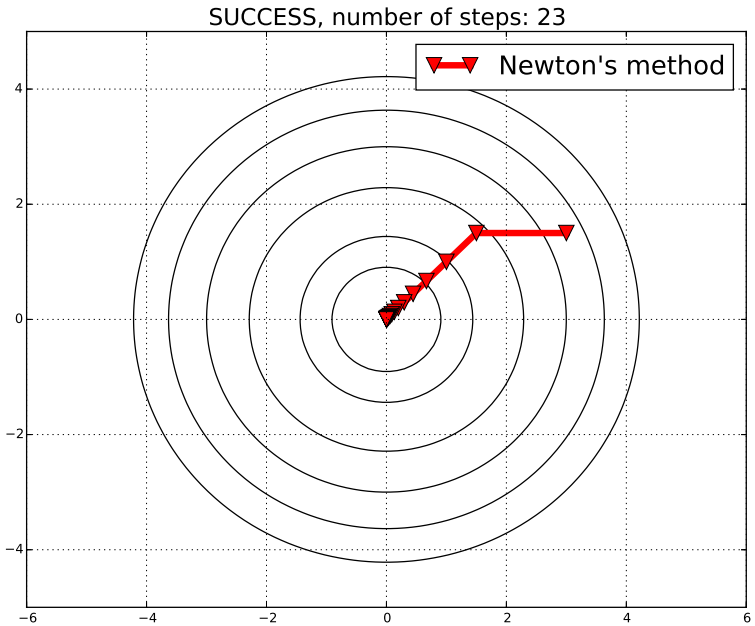




Линейная сходимость метода Ньютона

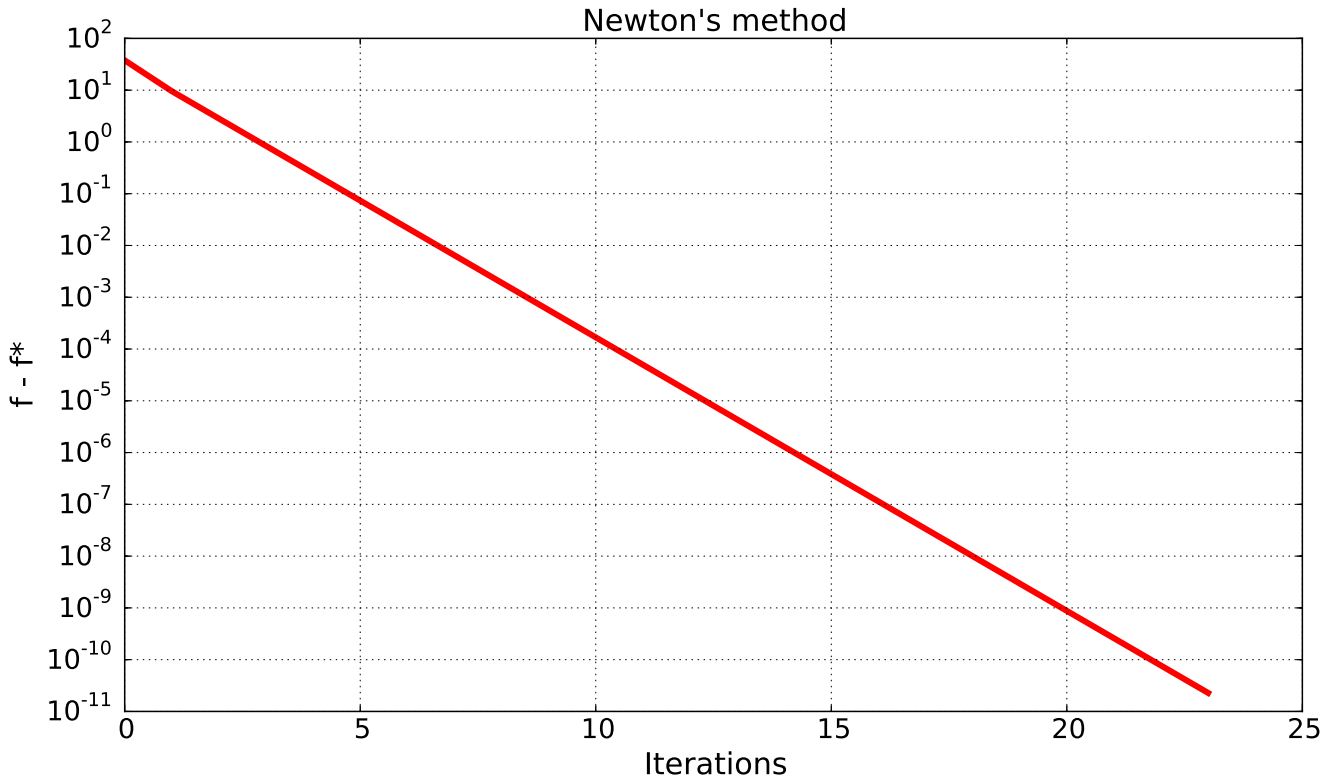
Функция — куб евклидовой нормы: $f(x) := \|x\|_2^3$





Запускается метод Ньютона с **единичным шагом**.

Линейная сходимость метода Ньютона

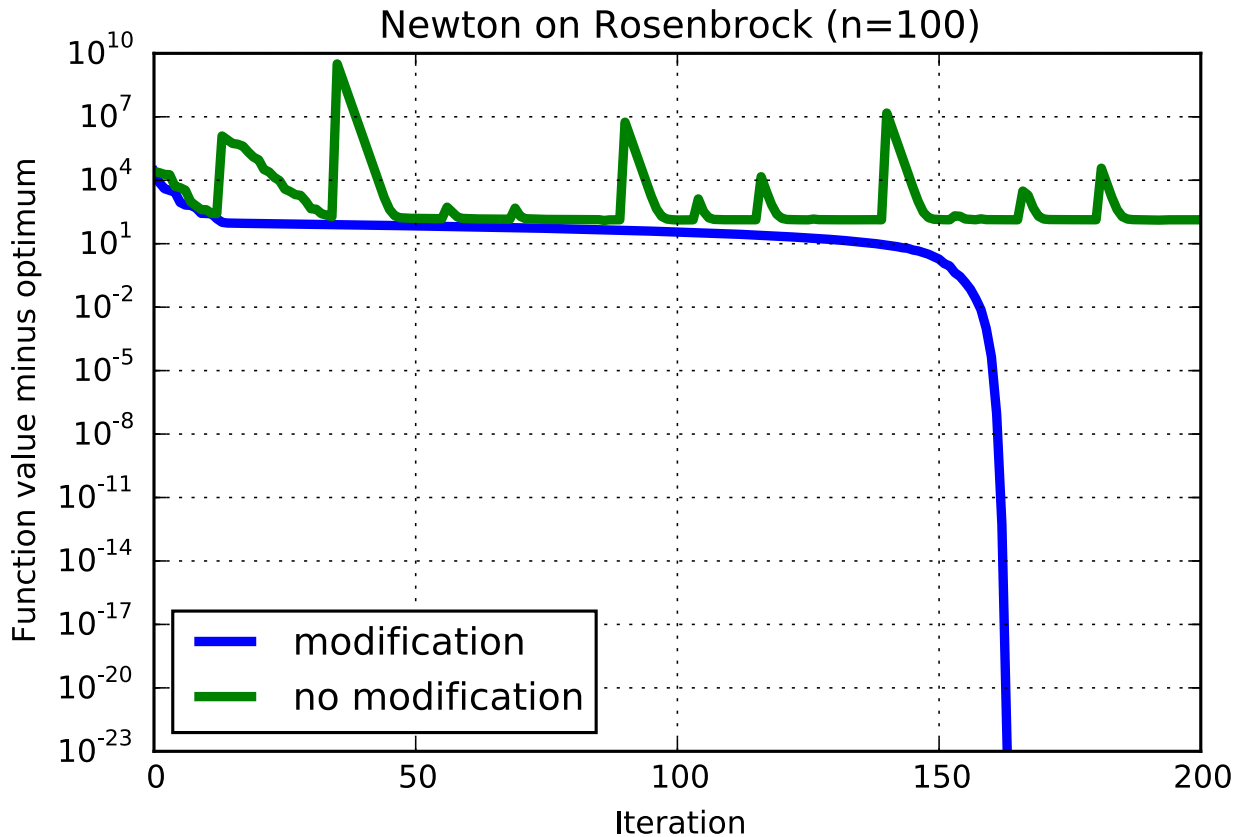


Почему нет квадратичной сходимости?

Модификация гессиана: метод Левенберга-Маркварта

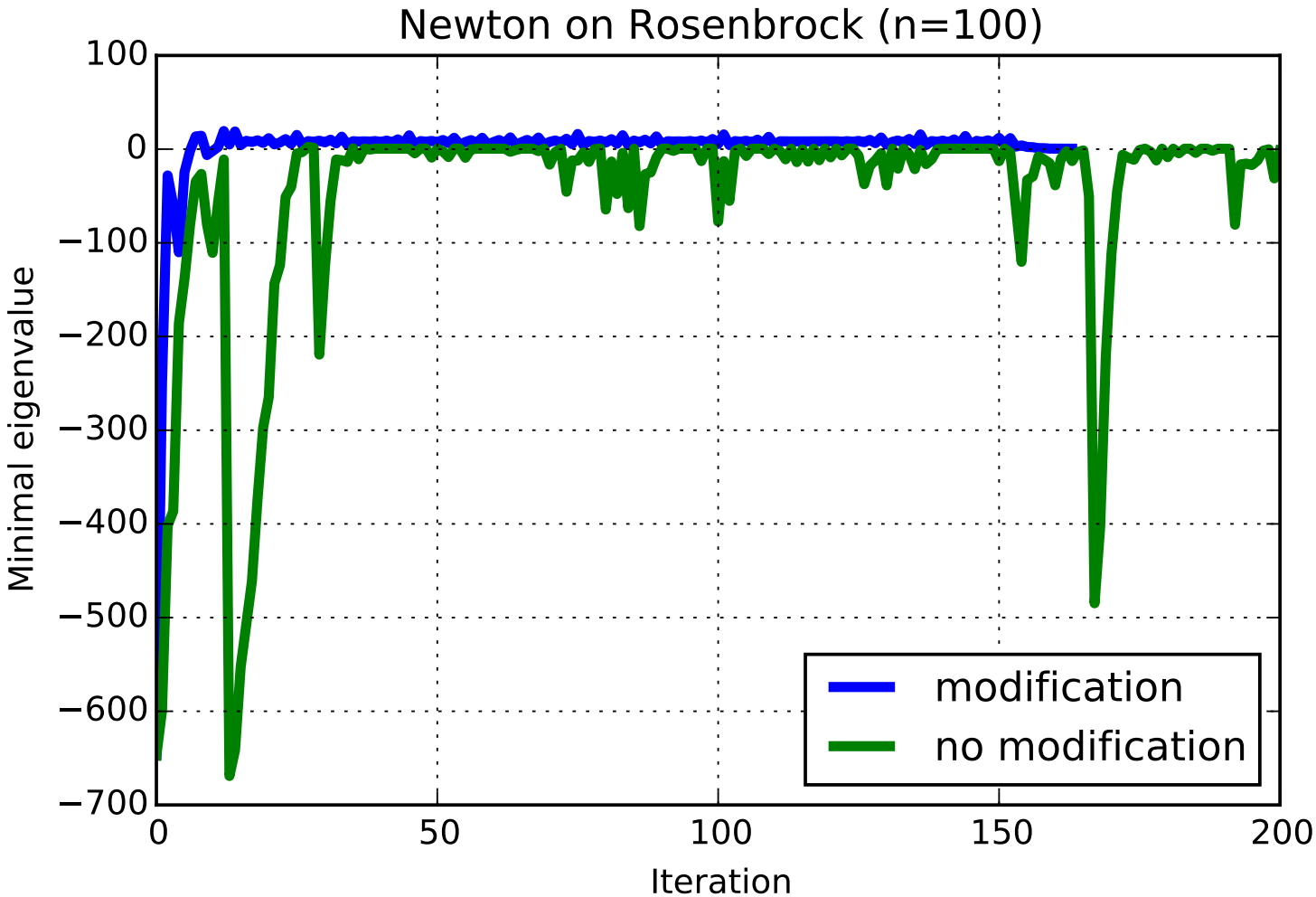
Многомерная функция Розенброка:

$$f(x) := \sum_{i=1}^{n-1} [100(x_{i+1} - x_i^2)^2 + (1 - x_i)^2]$$



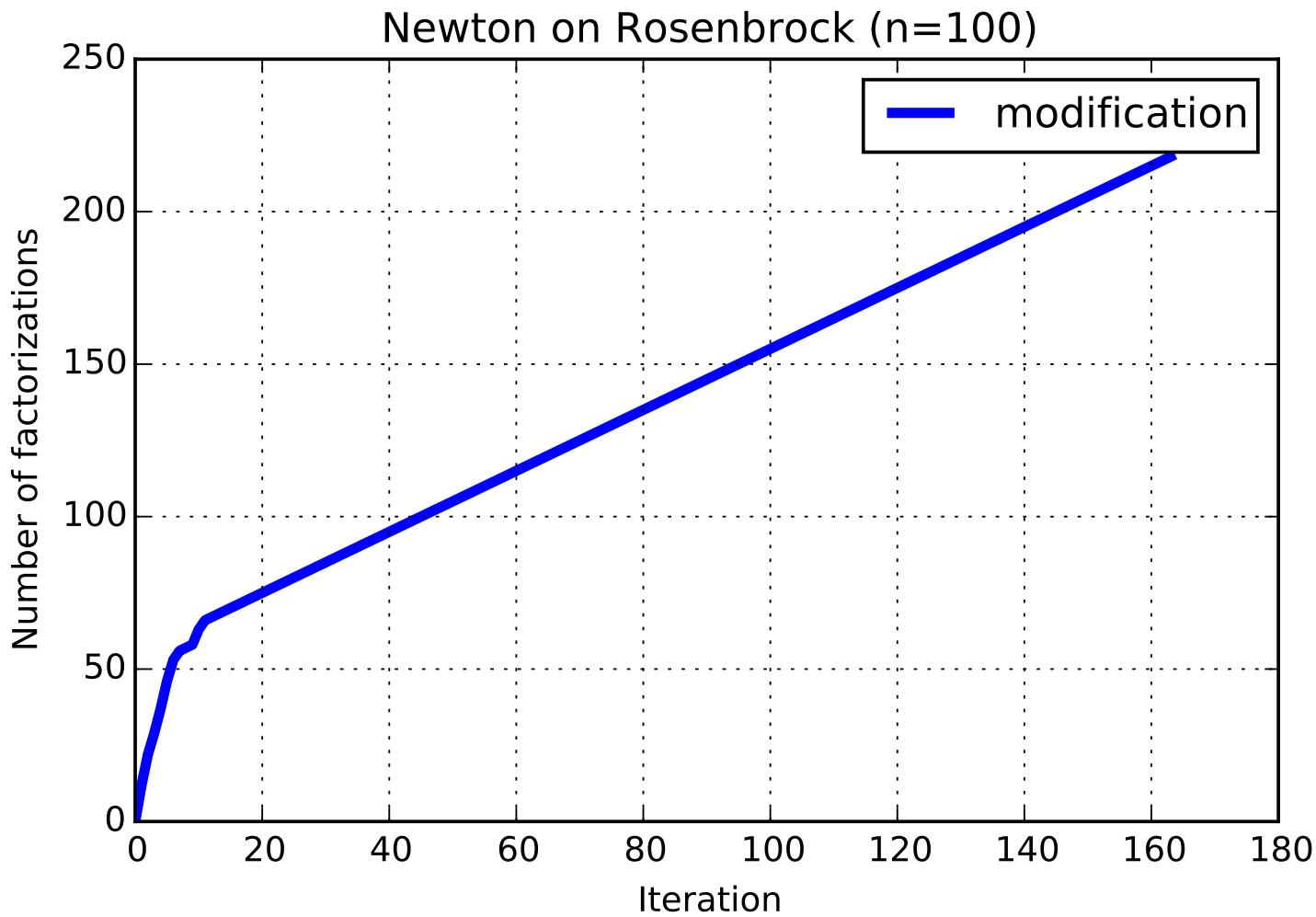
Модификация гессиана: метод Левенберга-Маркварта

Собственные значения в ходе итераций:



Пример с модификацией гессиана

Число факторизаций Холецкого в итерациях:



Метод сопряженных градиентов (CG). Общая схема

Решаемая задача: $Ax = b$, где $A \in \mathbb{S}^n_{++}$, $b \in \mathbb{R}^n$.
Эквивалентно:

$$f(x) := \frac{1}{2} x^T A x - b^T x \qquad \to \min_{x \in \mathbb{R}^n}.$$

- Метод спуска: $x_{k+1} = x_k + \alpha_k d_k$.
- Обозначение: $g_k := \nabla f(x_k) = Ax_k - b$.
- В этом случае можно аналитически найти наилучший α_k:

$$\alpha_k = \operatorname*{argmin}_{\alpha \geq 0} f(x_k + \alpha_k d_k) = \frac{-g_k^T d_k}{d_k^T A d_k}$$

- Основная идея CG — строить сопряженные направления: $d_i^T A d_j = 0$ для $i \neq j$.
- В качестве $d_1, \ldots, d_n$ можно взять базис из ортогональных собственных векторов $q_1, \ldots, q_n$ матрицы A. Но это слишком дорого (находить спектральное разложение).

Метод сопряженных градиентов (CG). Общая схема – 2

Решаемая задача: $Ax = b$, где $A \in \mathbb{S}^n_{++}$, $b \in \mathbb{R}^n$.

Обозначение: $g_k := Ax_k - b$.

Метод: $x_{k+1} = x_k + \alpha_k d_k$, где $\alpha_k := \frac{-g_k^T d_k}{d_k^T A d_k}$.

Основная идея — строить d_k онлайн (в итерациях алгоритма):

- Пусть уже есть сопряженные $d_0, \ldots, d_k$, т.е. $d_i^T A d_j = 0$ $(i \neq j)$.
- Будем искать d_{k+1} как линейную комбинацию g_{k+1} и d_k:

$$d_{k+1} = -g_{k+1} + \beta_k d_k.$$

- Коэффициент b_k можно найти из требуемого условия ортогональности:

$$0 = d_k^T A d_{k+1} = -d_k^T A g_{k+1} + \beta_k d_k^T A d_k \quad \Rightarrow \quad \beta_k = \frac{d_k^T A g_{k+1}}{d_k^T A d_k}$$

- Таким образом мы обеспечили лишь $d_k^T A d_{k+1} = 0$.
- Оказывается, что если выбрать $d_0 = -g_0$ (это важно!), то автоматически будет $d_i^T A d_{k+1} = 0$ для $i < k$.

Метод сопряженных градиентов (CG). Общая схема – 3

Итак, метод сопряженных градиентов (предварительная версия):

1. $g_k := Ax_k - b$;
2. $\alpha_k := \frac{-g_k^T d_k}{d_k^T A d_k}$;
3. $x_{k+1} := x_k + \alpha_k d_k$;
4. $g_{k+1} := Ax_{k+1} - b$;
5. $\beta_k := \frac{d_k^T A g_{k+1}}{d_k^T A d_k}$;
6. $d_{k+1} := -g_{k+1} + \beta_k d_k$;

Недостаток: четыре матрично-векторных произведения за итерацию.

Важное свойство CG: $g_{k+1}^T g_i = 0$ и $g_{k+1}^T d_i = 0$ для $i <= k$.

1. Заметим, что $g_{k+1} = A(x_k + \alpha_k d_k) - b = g_k + \alpha_k A d_k$.
2. Отсюда $A d_k = \alpha_k^{-1}(g_{k+1} - g_k)$. Значит,

$$\beta_k = \frac{(g_{k+1} - g_k)^T g_{k+1}}{d_k^T (g_{k+1} - g_k)} = \frac{g_{k+1}^T g_{k+1}}{-d_k^T g_k}$$

$$= \frac{g_{k+1}^T g_{k+1}}{-(-g_k + \beta_{k-1} d_{k-1})^T g_k} = \frac{g_{k+1}^T g_{k+1}}{g_k^T g_k}$$

3. Аналогично $\alpha_k := \frac{g_k^T g_k}{d_k^T A d_k}$.

Метод сопряженных градиентов (CG). Общая схема – 4

В итоге:

$g_0 := Ax_0 - b$;

$d_0 := -g_0$;

$k := 0$;

while $\|g_k\|_2 > \varepsilon \|g_0\|_2$ **do**

$\alpha_k := \frac{g_k^T g_k}{d_k^T A d_k}$;

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

$g_{k+1} := g_k + \alpha_k A d_k$;

$\beta_k := \frac{g_{k+1}^T g_{k+1}}{g_k^T g_k}$;

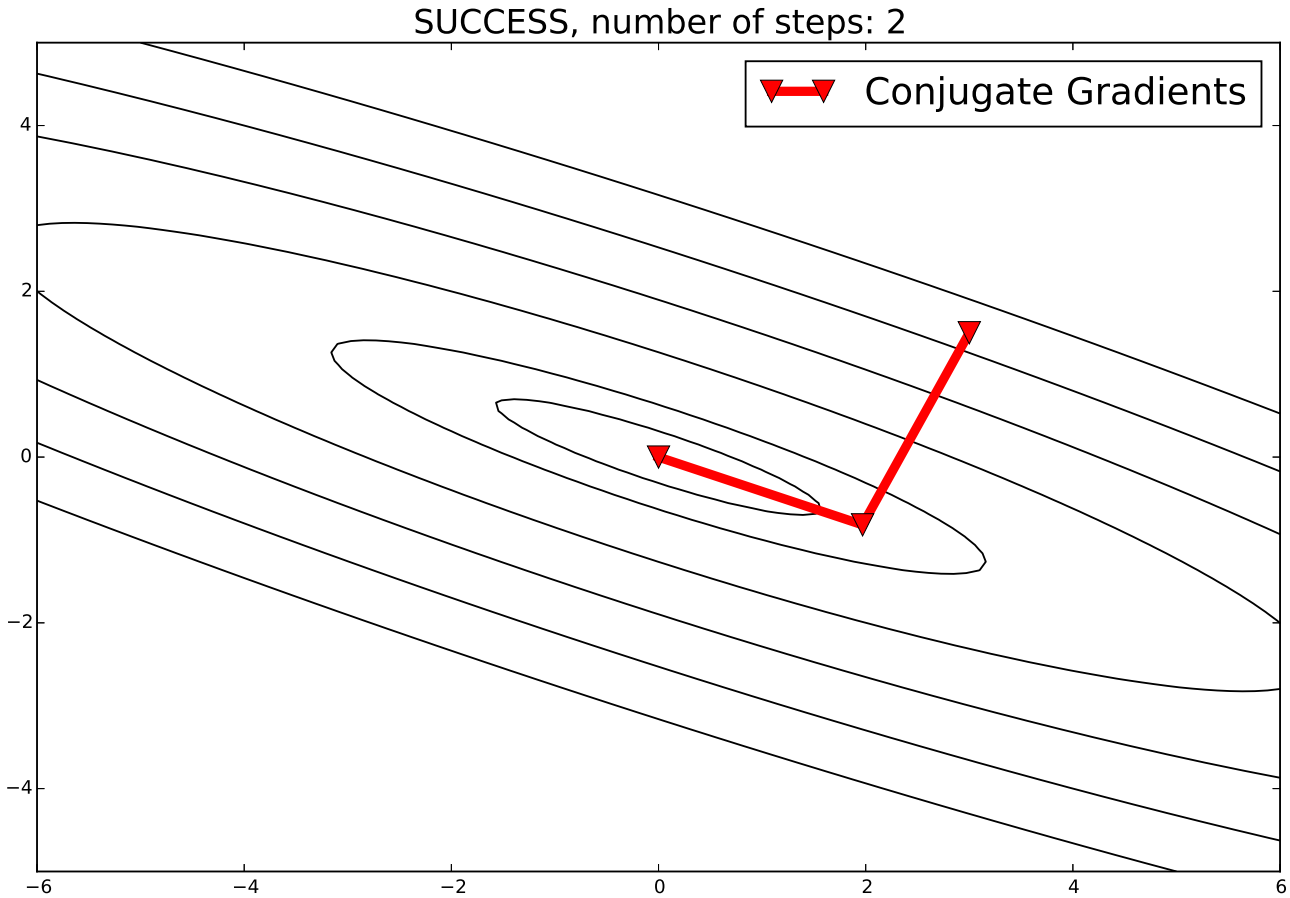
$d_{k+1} := -g_{k+1} + \beta_k d_k$;

$k := k + 1$;

end

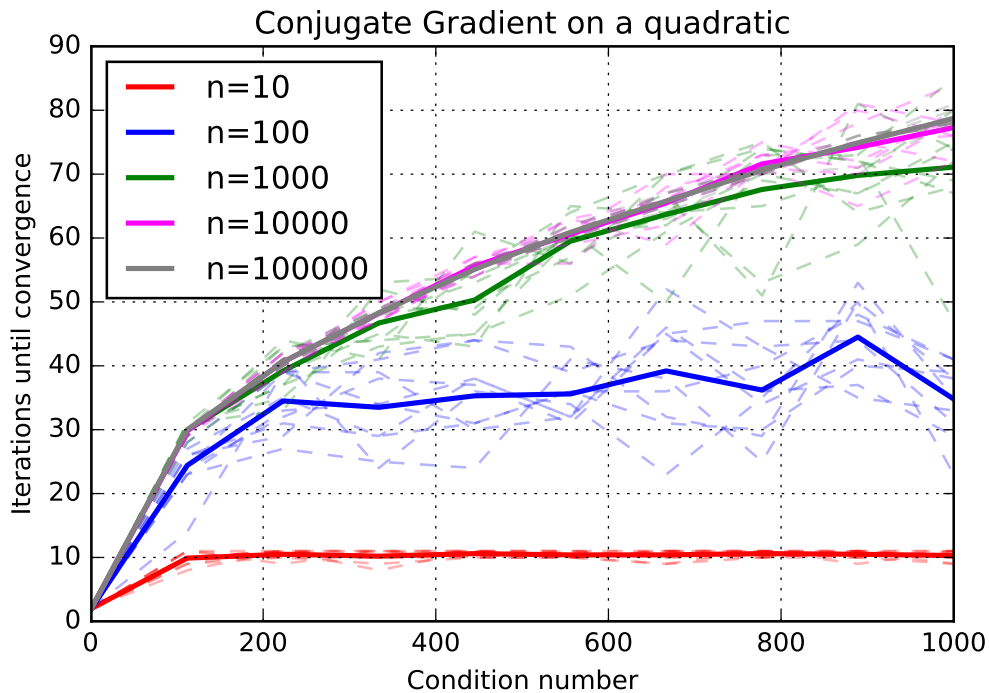
Одно матрично-векторное произведение за итерацию!

Метод сопряженных градиентов: траектория



Зависимость от числа обусловленности и размерности пространства

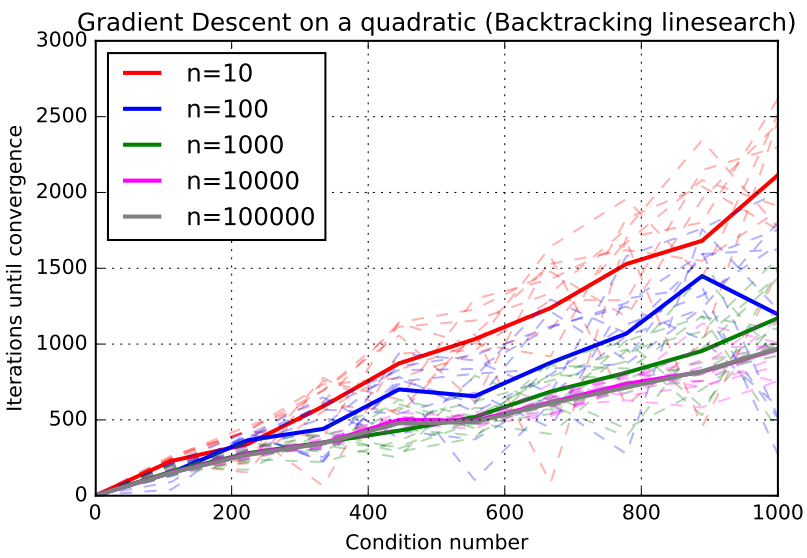
$$A = \mathrm{Diag}(a), \qquad b \sim \mathcal{N}(0, I_n), \qquad a_i = \begin{cases} 1, & i = 1 \\ \sim \mathrm{Unif}(1, \kappa), & 2 \le i \le n-1 \\ \kappa, & i = n \end{cases}$$

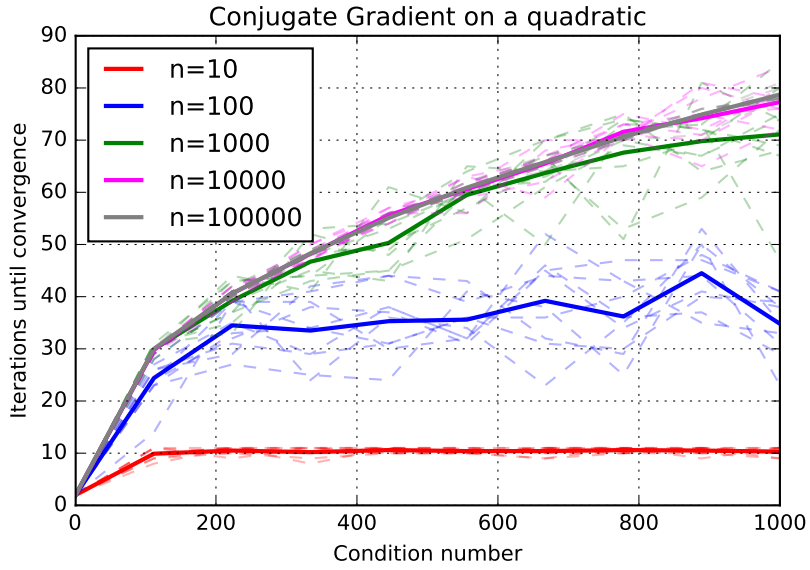


Зависимость от числа обусловленности: сравнение с градиентным спуском

Квадратичная функция: $f(x) = \frac{1}{2}x^T Ax - b^T x$.

$$A = \mathrm{Diag}(a), \qquad b \sim \mathcal{N}(0, I_n), \qquad a_i = \begin{cases} 1, & i = 1 \\ \sim \mathrm{Unif}(1, \kappa), & 2 \le i \le n-1 \\ \kappa, & i = n \end{cases}$$





Предобуславливание в методе сопряженных градиентов

- Решаемая задача: $Ax = b$, где $A \in \mathbb{S}^n_{++}$.
- Скорость работы метода определяется спектром матрицы A.
- Спектр можно улучшить с помощью эквивалентного преобразования системы:

$$Ax = b \quad \Leftrightarrow \quad (S^{-T}AS^{-1})(Sx) = S^{-T}b,$$

  где $S \in \mathbb{R}^{n \times n}$ — невырожденная матрица.
- Новая система: $\tilde{A}\tilde{x} = \tilde{b}$, где

$$\tilde{A} := S^{-T}AS^{-1}, \qquad \tilde{b} := S^{-T}b.$$

  Решение исходной системы: $x = S^{-1}\tilde{x}$.
- Матрица $M := S^T S$ называется предобуславливателем.
- Если $M \approx A$, то $\tilde{A} \approx I \Rightarrow$ сходимость $\approx$ за одну итерацию.

CG vs PCG

Обычный CG:

$r_0 \leftarrow Ax_0 - b$;
$d_0 \leftarrow -r_0$;
$k \leftarrow 0$;
while $\|r_k\|_2 > \varepsilon\|r_0\|_2$ **do**
  $\alpha_k \leftarrow \frac{r_k^T r_k}{d_k^T A d_k}$;
  $x_{k+1} \leftarrow x_k + \alpha_k d_k$;
  $r_{k+1} \leftarrow r_k + \alpha_k A d_k$;
  $\beta_k \leftarrow \frac{r_{k+1}^T r_{k+1}}{r_k^T r_k}$;
  $d_{k+1} \leftarrow -r_{k+1} + \beta_k d_k$;
  $k \leftarrow k + 1$;
end

CG с предобуславливателем:

$r_0 \leftarrow Ax_0 - b$;
$d_0 \leftarrow -M^{-1} r_0$;
$k \leftarrow 0$;
while $\|r_k\|_2 > \varepsilon\|r_0\|_2$ **do**
  $\alpha_k \leftarrow \frac{r_k^T M^{-1} r_k}{d_k^T A d_k}$;
  $x_{k+1} \leftarrow x_k + \alpha_k d_k$;
  $r_{k+1} \leftarrow r_k + \alpha_k A d_k$;
  $\beta_k \leftarrow \frac{r_{k+1}^T M^{-1} r_{k+1}}{r_k^T M^{-1} r_k}$;
  $d_{k+1} \leftarrow -M^{-1} r_{k+1} + \beta_k d_k$;
  $k \leftarrow k + 1$;
end

- Дополнительно нужна процедура решения $Mz_k = r_k$.

Предобуславливание: пример

- Система $Ax = b$ размера $n = 500$, где $b = (1, \ldots, 1)$ и

$$a_{ij} = \begin{cases} 1 + i^{1.2} & \text{если } i = j \\ 1 & \text{если } |i - j| = 1 \text{ или } |i - j| = 100 \\ 0 & \text{иначе} \end{cases}$$

- Диагональный предобуславливатель: $M = \mathrm{Diag}(A)$.