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

Направление спуска: $\nabla f(x_k)^T d_k < 0$.

Линейный поиск

Функция прогресса: $\phi_k(\alpha) := f(x_k + \alpha_k d_k), \qquad \alpha \geq 0.$

Стратегии выбора шага:

- **Постоянный шаг:** $\alpha_k := \text{const}.$
- **Наискорейший спуск:** $\alpha_k := \text{argmin}_{\alpha \geq 0} \phi_k(\alpha).$
- **Неточный линейный поиск:**
  - **Бэктрэкинг + условие Армихо:**

$\alpha \leftarrow \alpha_0^{(k)}$
while not $\phi_k(\alpha) \leq \phi_k(0) + c_1 \alpha \phi_k'(0)$ **do**
$\alpha \leftarrow \alpha/2$
end while

  - Методы, основанные на сильных условиях Вульфа:

$$\phi_k(\alpha) \leq \phi_k(0) + c_1 \alpha \phi_k'(0)$$
$$|\phi_k'(\alpha)| \leq c_2 |\phi_k'(0)|$$

Используют внутри квадратичные/кубические аппроксимации.

Градиентный спуск

Градиентный спуск:

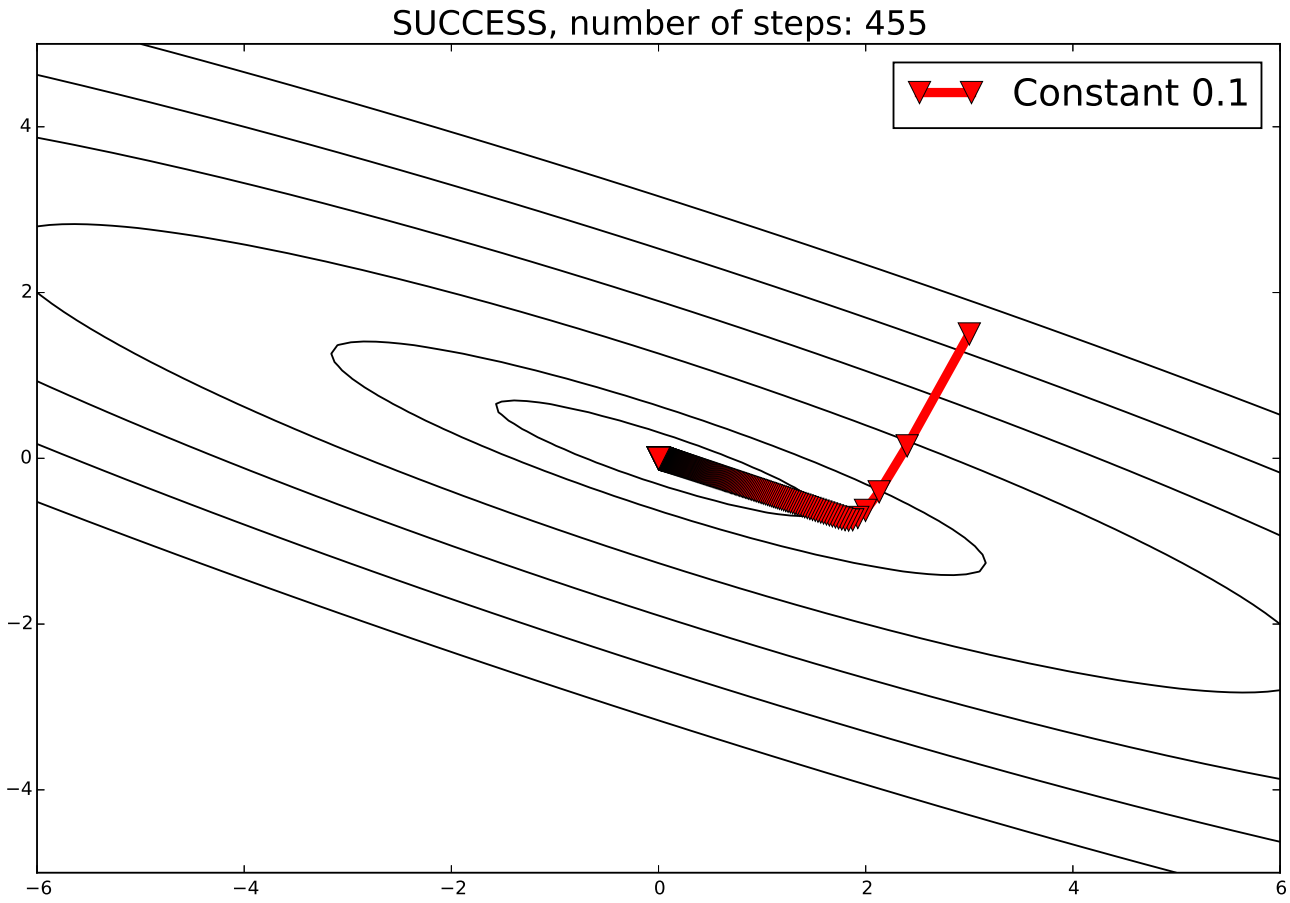
$$x_{k+1} = x_k - \alpha_k \nabla f(x_k)$$

- **Интерпретация:** Метод спуска с направлением

$$d_k = -\nabla f(x_k)$$

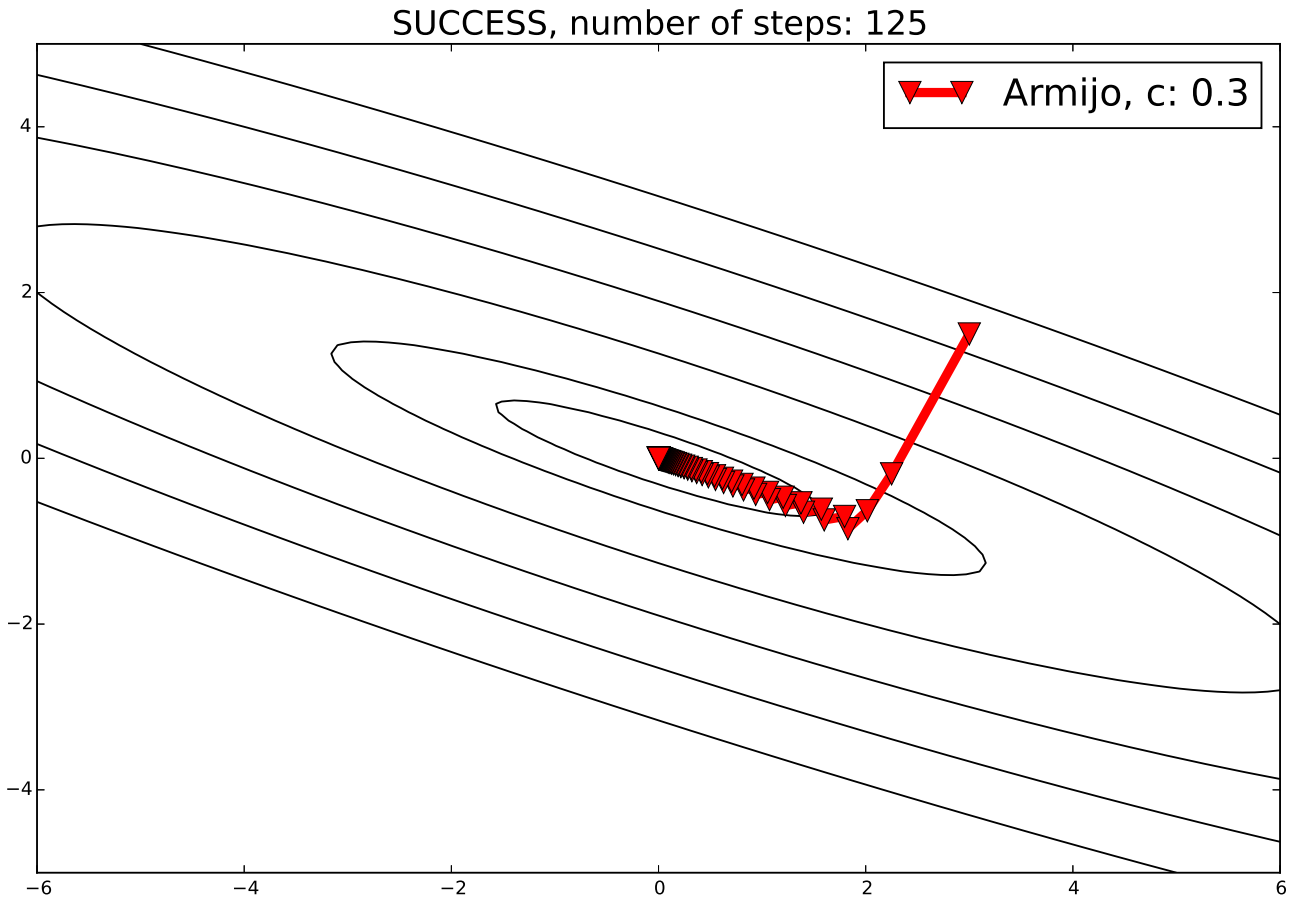
Градиентный спуск: постоянный шаг

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



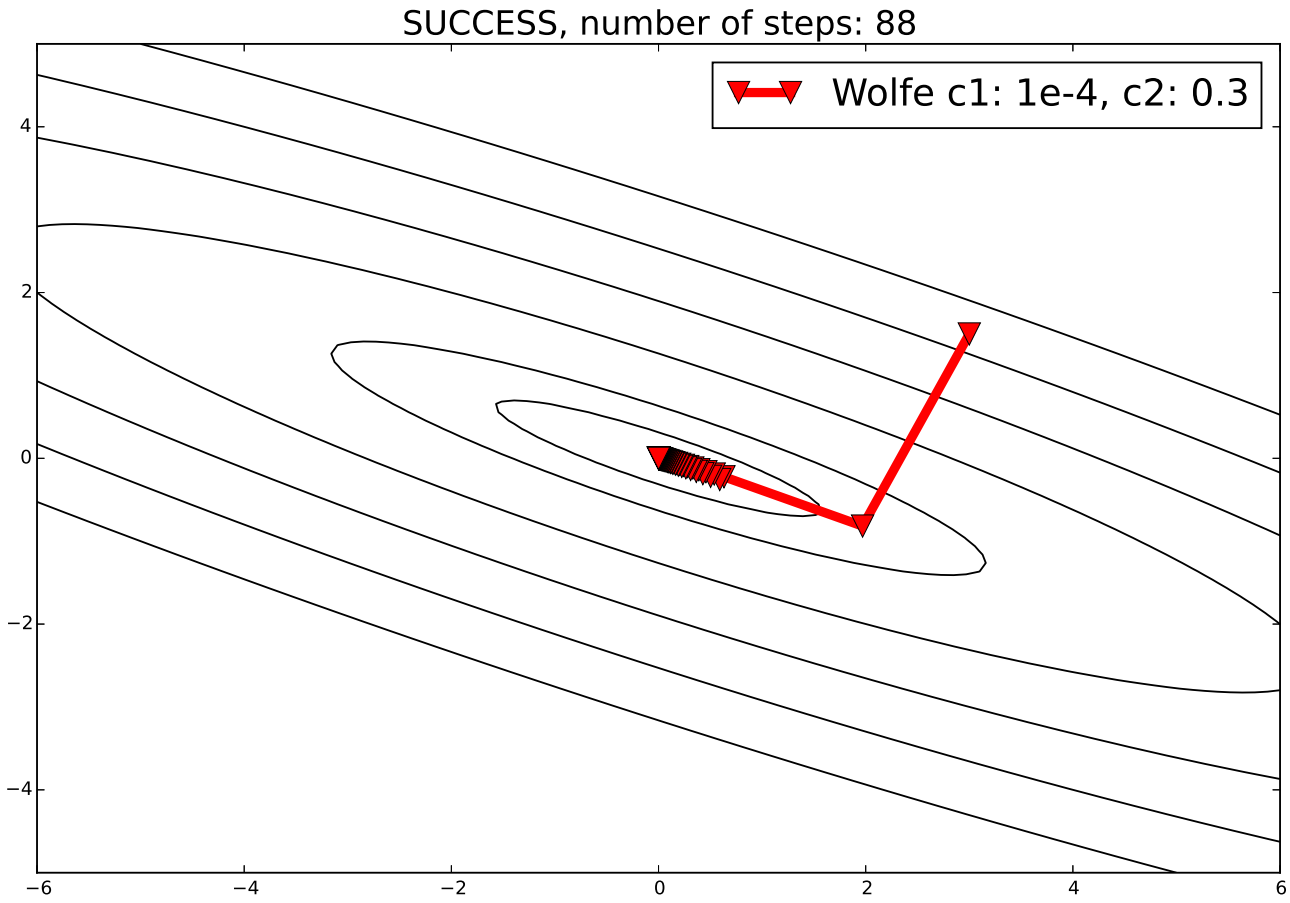
Градиентный спуск: бэктрекинг

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



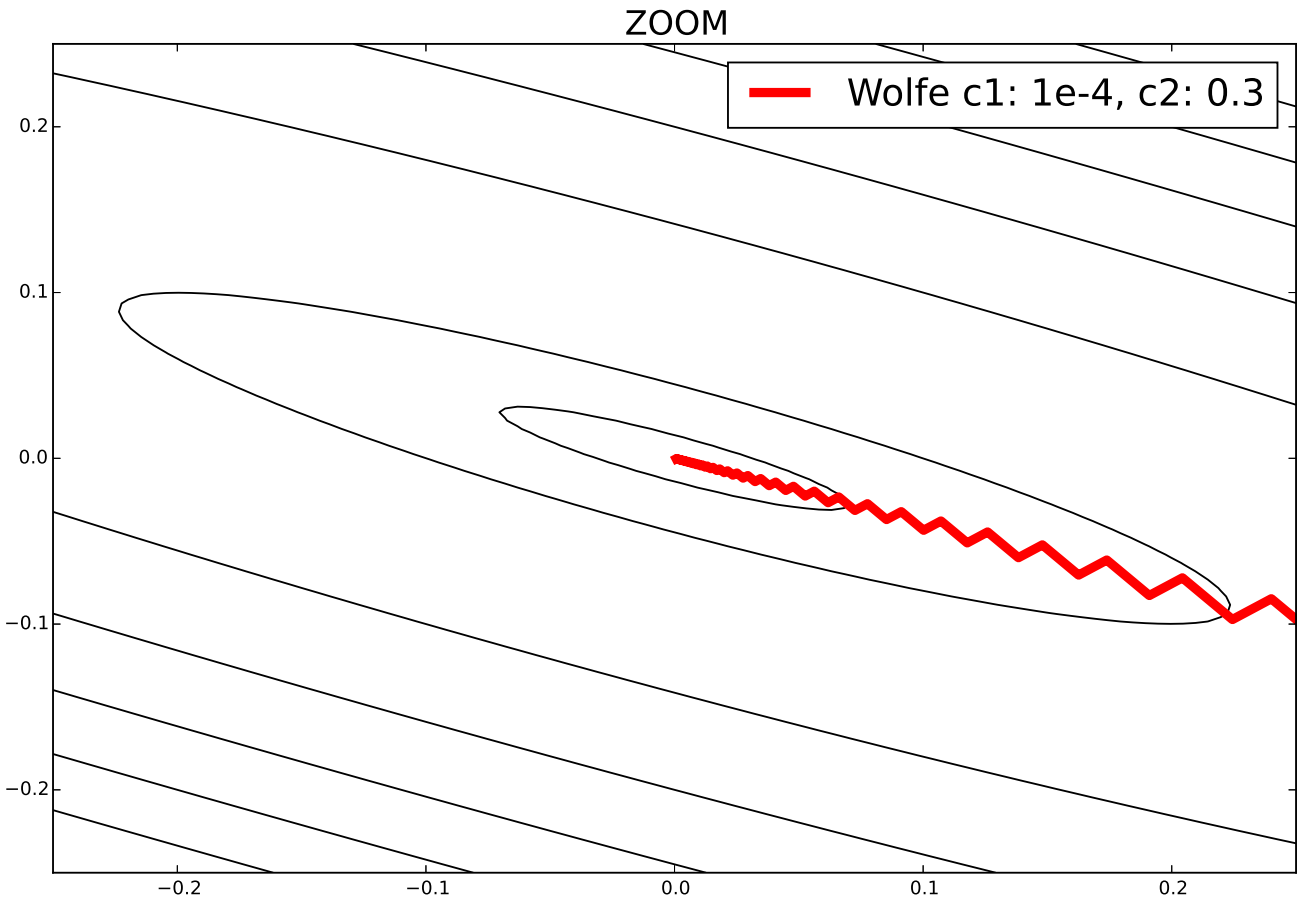
Градиентный спуск: стратегия Вульфа

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

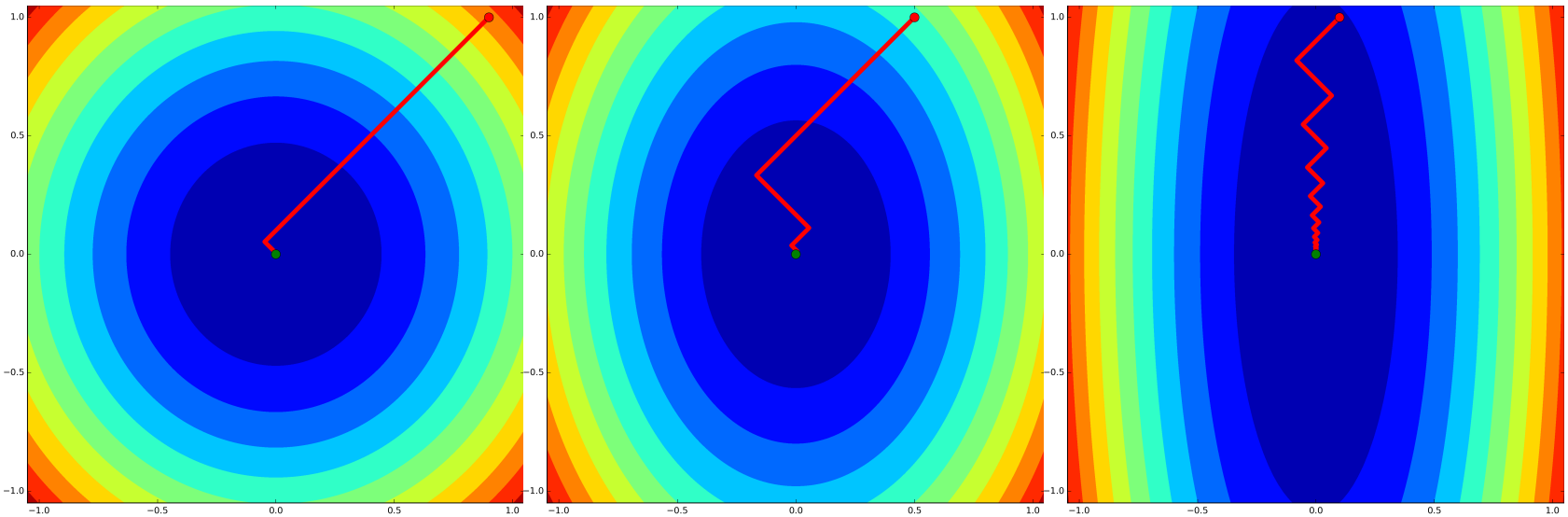
Квадратичная функция: $f(x) = \frac{1}{2} x^T A x, \quad A = \begin{pmatrix} 1 & 2 \\ 2 & 5 \end{pmatrix}$.



Градиентный спуск: типичная траектория

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

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

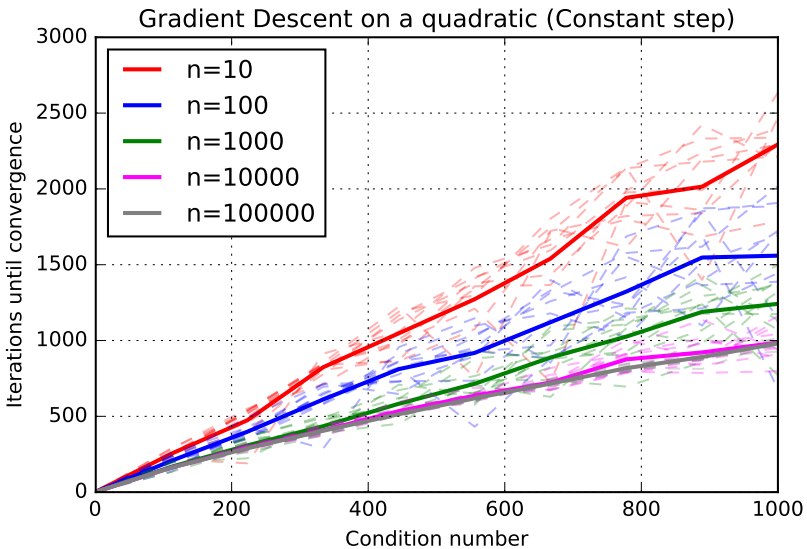


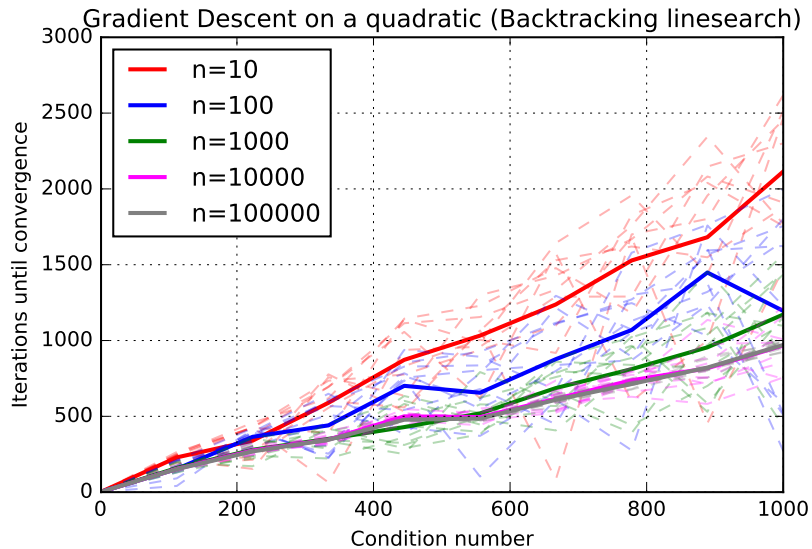
Плохая обусловленность + неудачный старт = **зигзаг**

Зависимость от обусловленности и размерности задачи

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

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



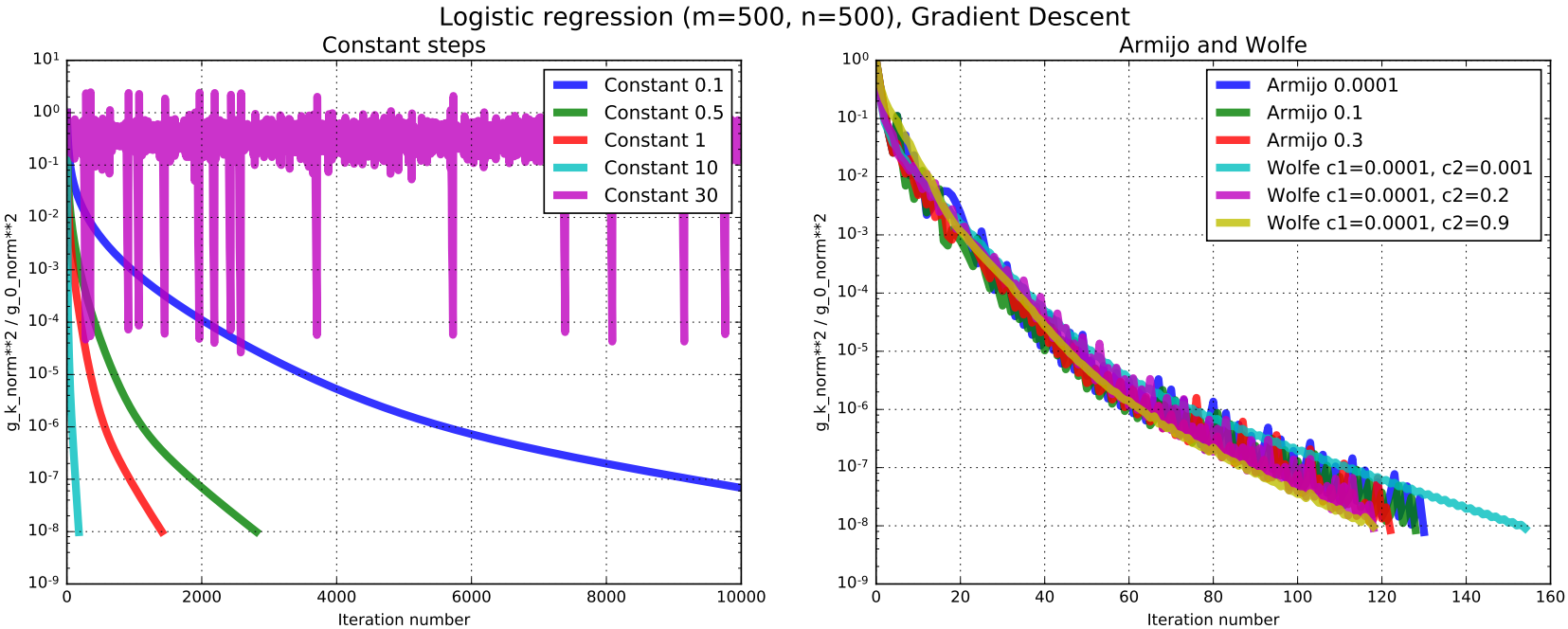


- Линейная зависимость от числа обусловленности.
- С ростом размерности число итераций не увеличивается!

Стратегии выбора длины шага в градиентном спуске

Логистическая регрессия с ℓ_2-регуляризатором:

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



- Сильная чувствительность к значению постоянного шага.
- Адаптивные стратегии сами подбирают «хорошую» длину шага.
- Разница между адаптивными стратегиями незначительная.

Учет структуры функции

Пусть $f(x) := \psi(Ax)$, где

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

Примеры:

- Квадратичная функция:

$$f(x) := \frac{1}{2} x^T A x - b^T x, \qquad \psi(y) := \frac{1}{2} x^T y - b^T x$$

- Логистическая регрессия:

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

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

Учет структуры функции – 2

Рассматриваемая ситуация: $f(x) = \psi(Ax)$.
Линейный поиск: $\phi_k(\alpha) := f(x_k + \alpha d_k)$.
Если считать «в лоб»:

- Стоимость вычисления для одного α: $\sim mn$.
- Стоимость вычисления для s разных α: $\sim smn$.

Учтем структуру функции: $\phi_k(\alpha) = \psi(Ax_k + \alpha Ad_k)$.

- Нужно один раз вычислить и запомнить Ax_k и Ad_k.
  Стоимость: $2mn$.
- Дальнейшее вычисление ϕ_k для любого α стоит $O(m)$.
- Суммарная стоимость для s разных α: $\sim 2mn$.
- Эффективно, когда $s \geq 2$.

Полностью аналогично для производной:

$$\phi'(\alpha) = \nabla\psi(Ax_k + \alpha Ad_k)^T Ad_k.$$

Детали реализации

Итерация метода спуска для $f(x) = \psi(Ax)$:

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 d_k (оракул не вызывается)
Линейный поиск:
$\phi(0) = \psi(Ax_k)$, $\phi'(0) = \nabla \psi(Ax_k)^T Ad_k$
$\phi(\bar{\alpha}_1) = \psi(Ax_k + \bar{\alpha}_1 Ad_k)$, $\phi'(\bar{\alpha}_1) = \nabla \psi(Ax_k + \bar{\alpha}_1 Ad_k)^T Ad_k$
$\dots$
$\phi(\bar{\alpha}_s) = \psi(Ax_k + \bar{\alpha}_s Ad_k)$, $\phi'(\bar{\alpha}_s) = \nabla \psi(Ax_k + \bar{\alpha}_s Ad_k)^T Ad_k$
$x_{k+1} \leftarrow x_k + \bar{\alpha}_s d_k$ $\triangleright$ $Ax_{k+1} = Ax_k + \bar{\alpha}_s Ad_k$

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

Детали реализации – 2. Пример

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

```
1  class CubicOptimizedOracle(BaseSmoothOracle):
2    def __init__(self, A):
3      self.A, self.last_x, self.last_x_trial = A, None, None
4
5    def func(self, x):
6      self._update_Ax(x); return (1/3) * self.norm_Ax**3
7
8    def grad(self, x):
9      self._update_Ax(x); return self.norm_Ax**3 * self.A.T.dot(self.Ax)
10
11   def func_directional(self, x, d, alpha):
12     self._update_Ax(x)
13     self._update_Ad(d) # TODO: IMPLEMENT
14     self.last_x_trial = x + alpha * d
15     self.Ax_trial = self.Ax + alpha * self.Ad
16     return (1/3) * np.linalg.norm(self.Ax_trial)**3
17
18   def _update_Ax(self, x):
19     if np.array_equal(x, self.last_x): return
20     if np.array_equal(x, self.last_x_trial):
21       self.last_x = self.last_x_trial
22       self.Ax, self.norm_Ax = self.Ax_trial, self.norm_Ax_trial
23       return
24     self.last_x = np.copy(x)
25     self.Ax = self.A.dot(x)
26     self.norm_Ax = np.linalg.norm(self.Ax)
```

Чистый (классический) метод Ньютона

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

Чистый метод Ньютона:

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

- **Интерпретация:** Минимизация квадратичной модели:

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

- **Скорость сходимости:** квадратичная (в невырожденном случае).
- Для плохих начальных приближений x_0 не гарантируется даже сходимости.

Демпфированный метод Ньютона

Демпфированный метод Ньютона:

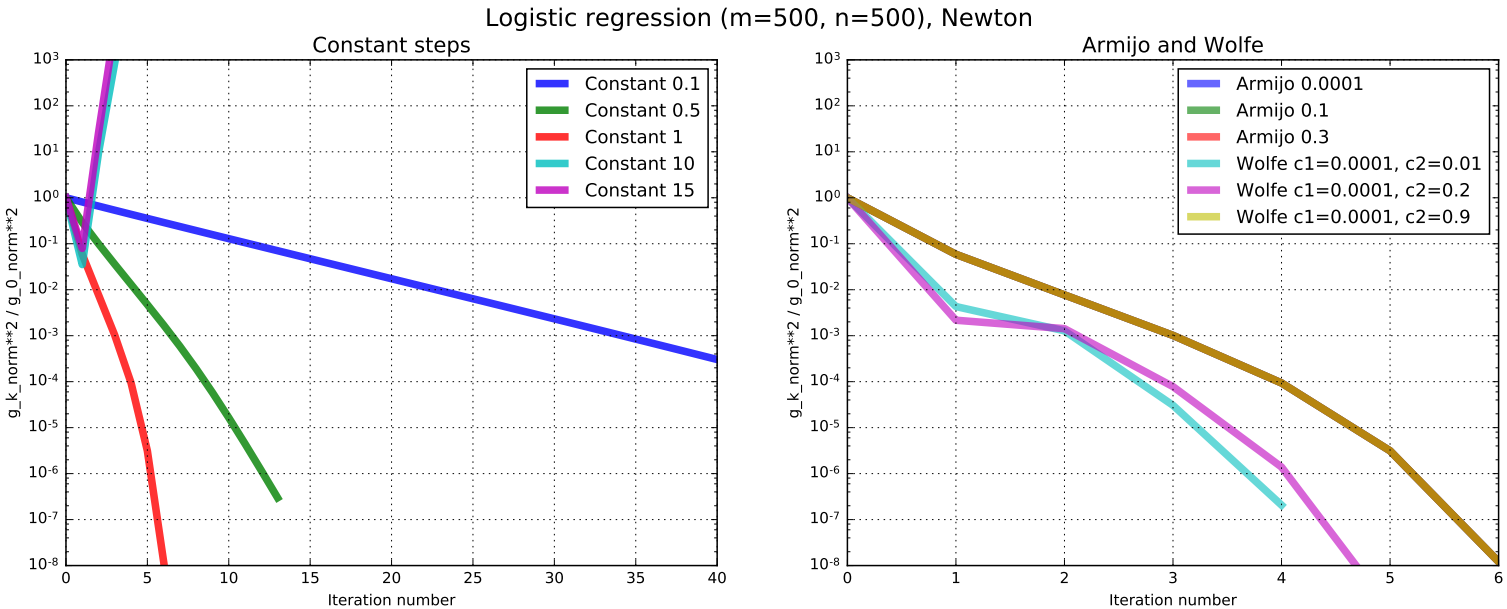
$$x_{k+1} = x_k - \alpha_k [\nabla^2 f(x_k)]^{-1} \nabla f(x_k).$$

- Добавляется «демпфирующая» длина шага $\alpha_k \in [0, 1]$.
- **Интерпретация:** Метод спуска с $d_k := -[\nabla^2 f(x_k)]^{-1} \nabla f(x_k)$:
  - Если $\nabla^2 f(x_k) \succ 0$, то
    $$\nabla f(x_k)^T d_k = -\nabla f(x_k)^T [\nabla^2 f(x_k)]^{-1} \nabla f(x_k) < 0.$$
  - Если $\nabla^2 f(x_k) \not\succ 0$, то применяют модификацию гессиана.
- 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.

Сравнение стратегий выбора шага в методе Ньютона

Логистическая регрессия с ℓ_2-регуляризатором:

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

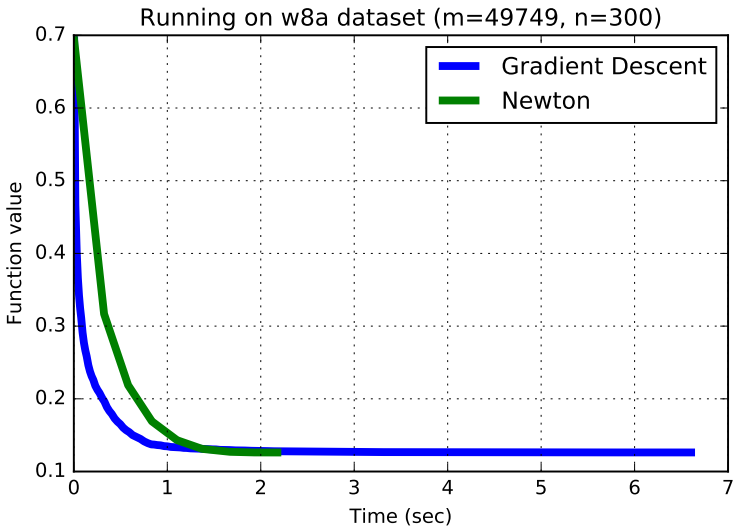


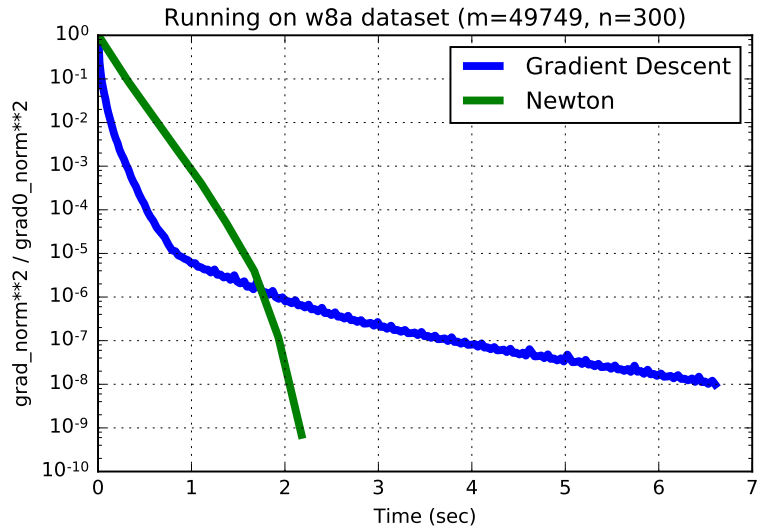
- Постоянный шаг:
  - При $\alpha > 1$ наблюдается расходимость.
  - При $\alpha < 1$ сходимость может быть линейной.
  - Наилучший выбор: $\alpha = 1$ (квадратичная сходимость).
- Адаптивные стратегии сами выбирают «хорошую» длину шага.
- Разница между адаптивными стратегиями незначительная.

Сравнение градиентного спуска и Ньютона

Логистическая регрессия с ℓ_2-регуляризатором:

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

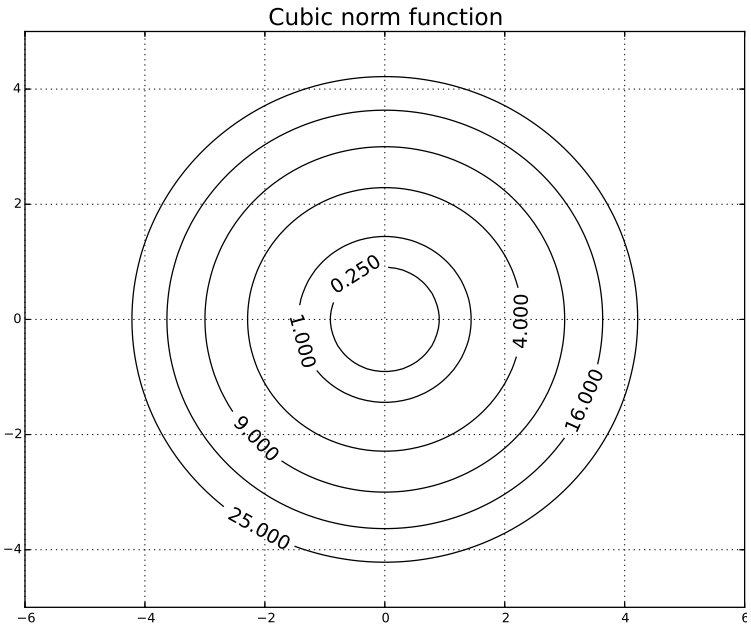


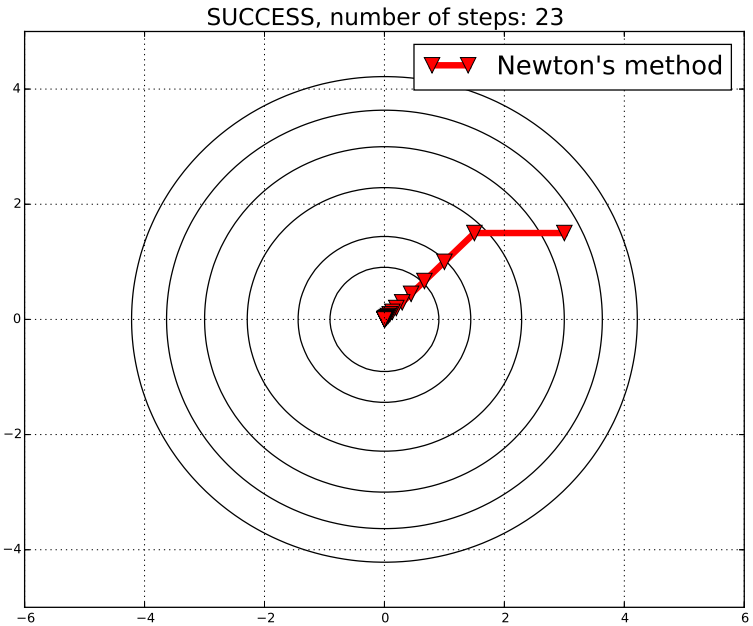


- Типичная ситуация: более простой метод быстрее работает в начале (для небольшой точности), но медленее в конце (для большой точности).

Линейная сходимость метода Ньютона

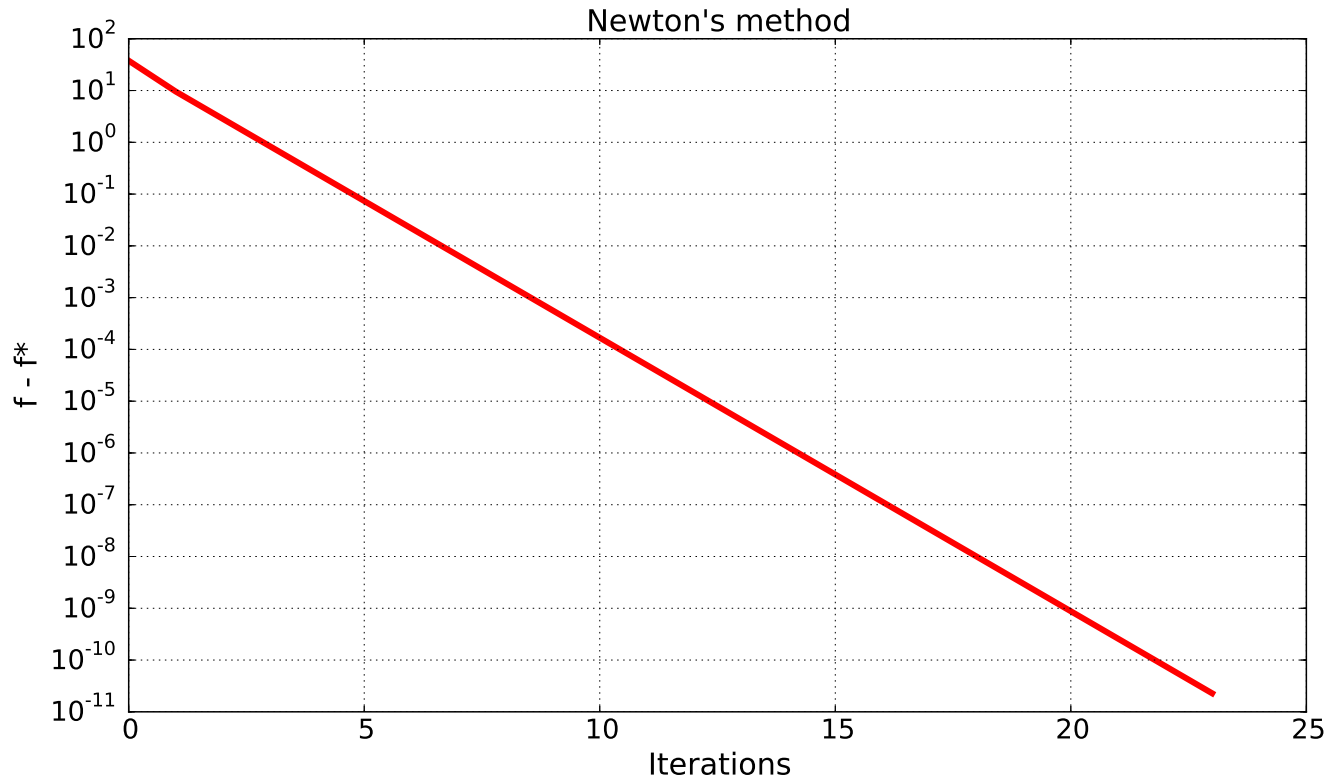
Функция — куб евклидовой нормы: $f(x) := \|x\|_2^3$.





Запускается метод Ньютона с **единичным шагом**.

Линейная сходимость метода Ньютона

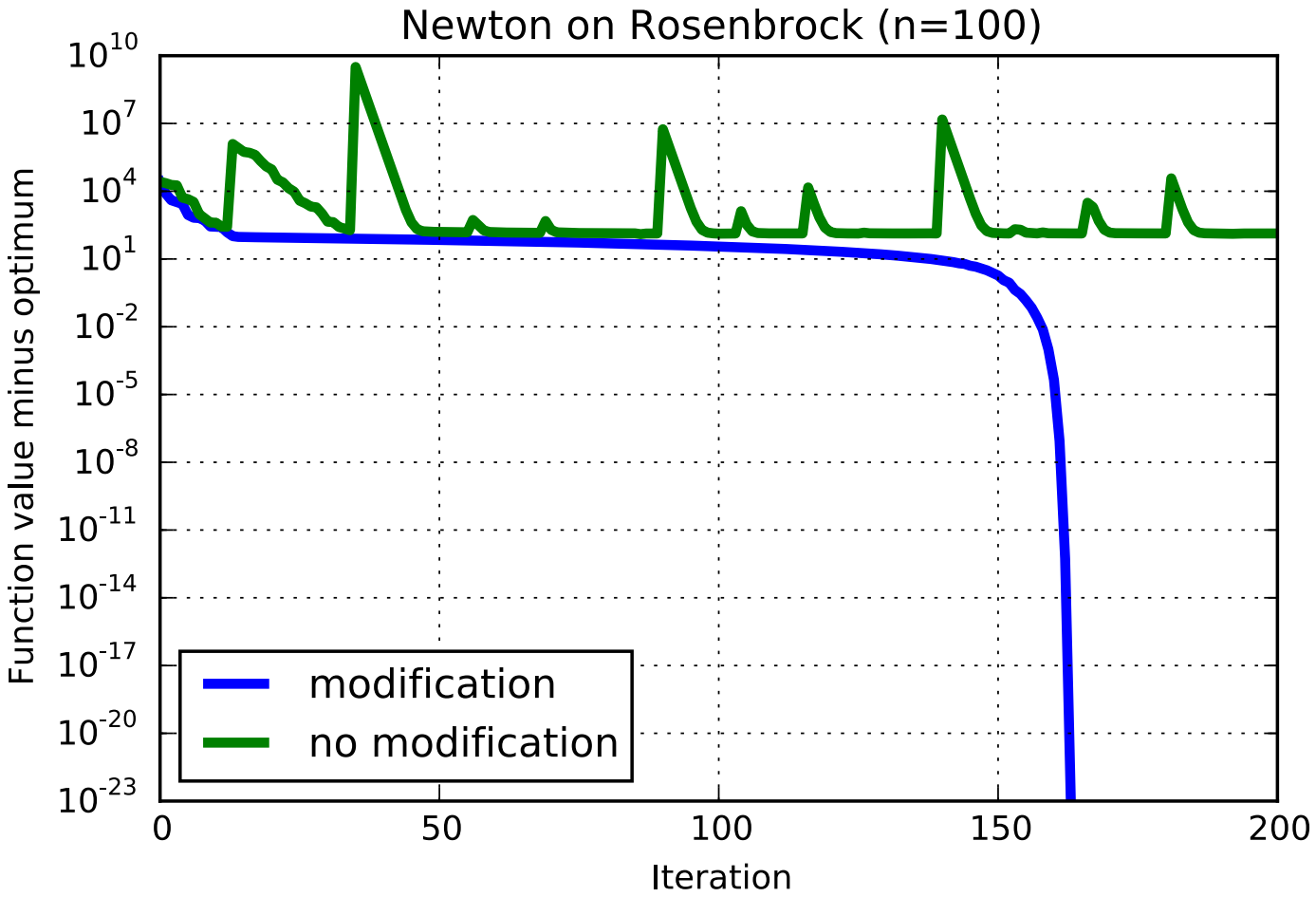


Почему нет квадратичной сходимости?

Модификация гессиана: метод Левенберга-Маркварта

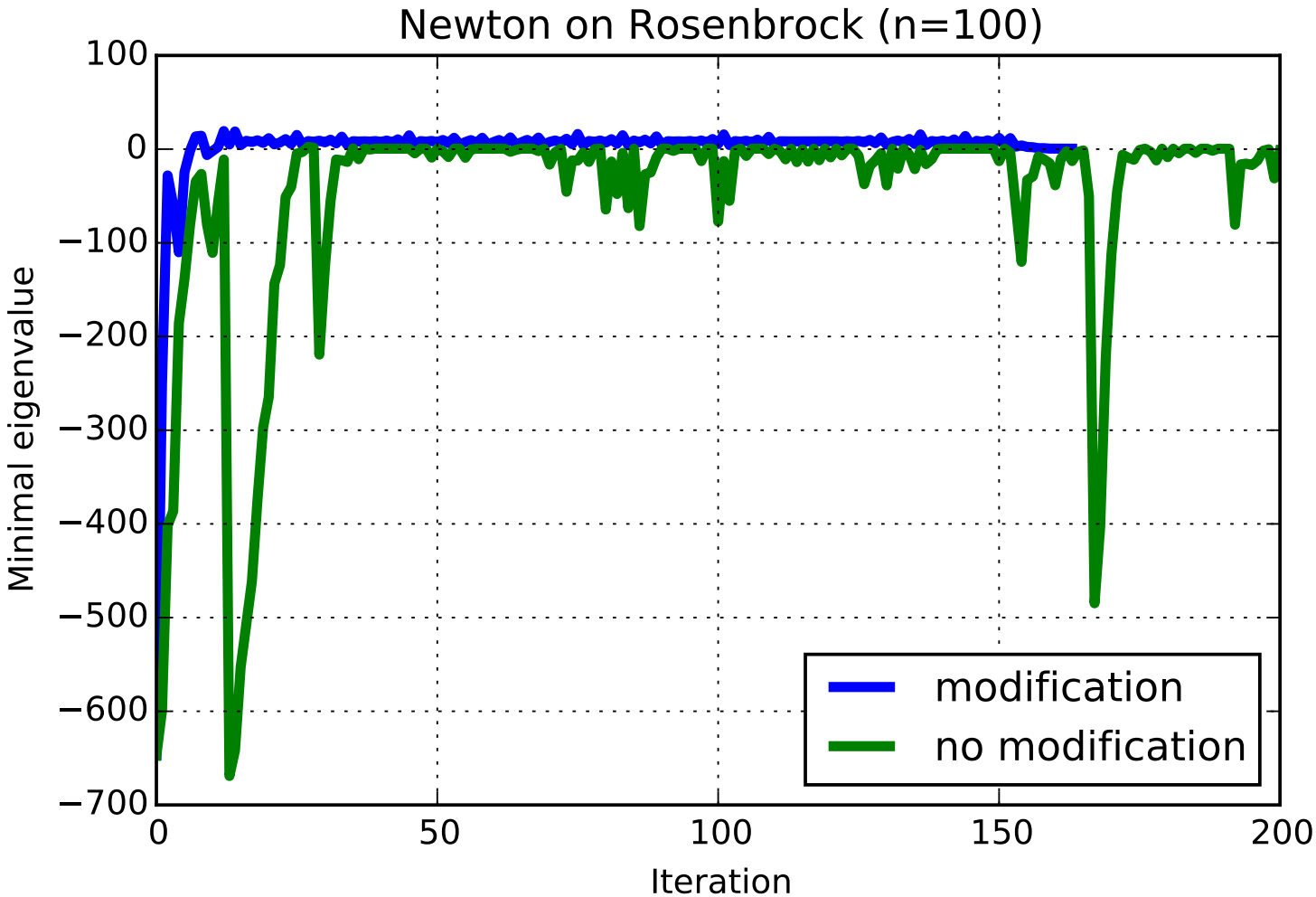
Многомерная функция Розенброка:

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



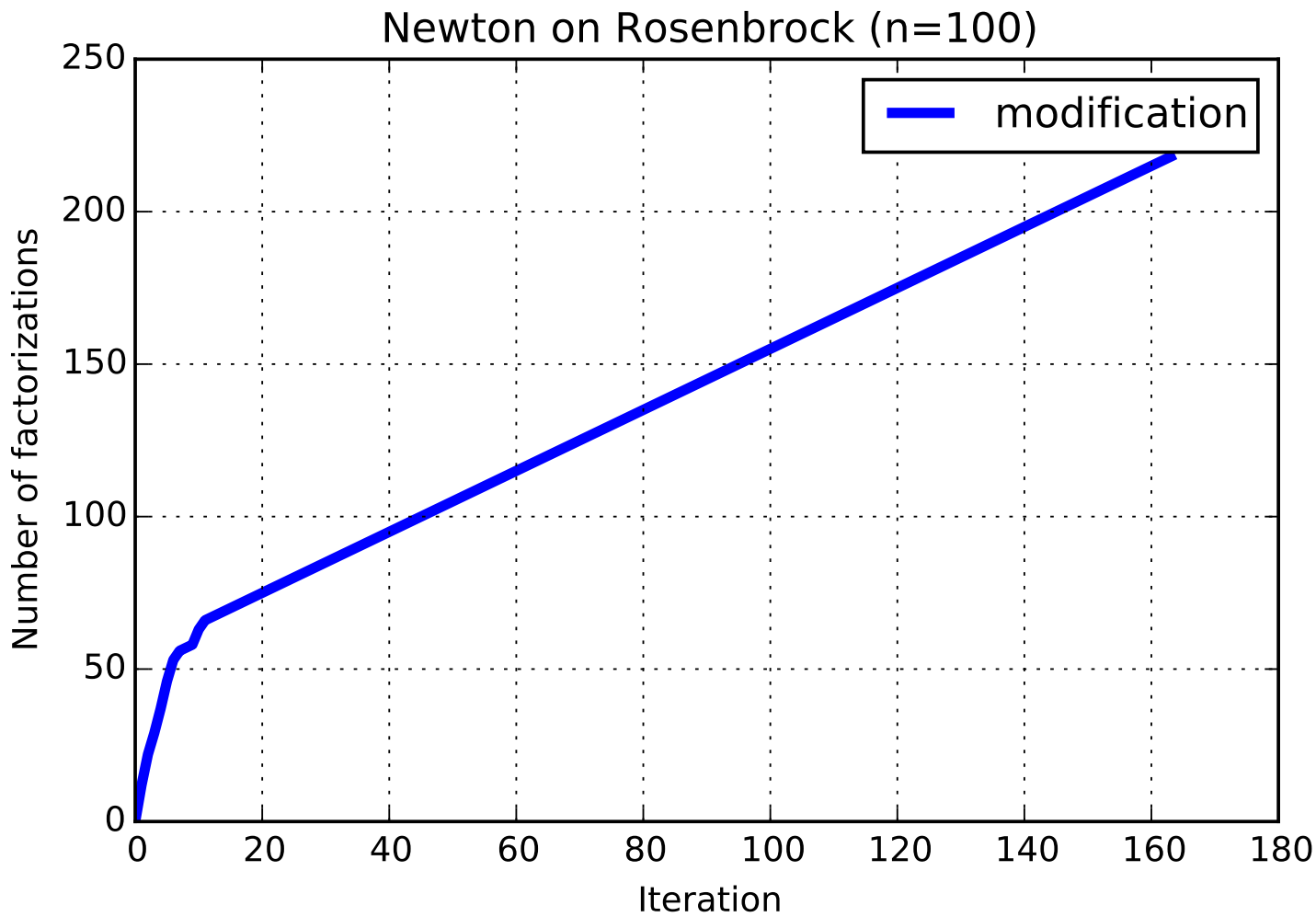
Модификация гессиана: метод Левенберга-Маркварта

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

- Метод спуска: $x_{k+1} = x_k + \alpha_k d_k$.
- Для квадратичной функции можно аналитически найти

$$\alpha_k = \underset{\alpha \geq 0}{\operatorname{argmin}} f(x_k + \alpha_k d_k) = \frac{-g_k^T d_k}{d_k^T A d_k}, \qquad g_k := Ax_k - b$$

Сопряженные направления: $d_i^T A d_j = 0$ для $i \neq j$.

Возможные варианты:

- Ортогональные собственные векторы $q_1, \ldots, q_n$ матрицы A.
- Любые л.н.з. векторы + модифицированный процесс Грамма-Шмидта.

Это все неэффективно для больших матриц!

Метод сопряженных градиентов (CG). Общая схема – 2

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

- Пусть уже есть $d_0, \ldots, d_k$, т. ч. $d_i^T A d_j = 0$ для $i \neq j$.
- Ищем d_{k+1} как линейную комбинацию g_{k+1} и d_k:

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

- Коэффициент β_k найдем из условия сопряженности:

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

- 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

Метод сопряженных градиентов (неэффективная версия):

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

- Недостаток: четыре матрично-векторных произведения.

Устраним этот недостаток:

1. Важное свойство CG: $g_{k+1}^T g_i = 0$ и $g_{k+1}^T d_i = 0$ для $i \leq k$.
2. Заметим, что $g_{k+1} = A(x_k + \alpha_k d_k) - b = g_k + \alpha_k A d_k$.
3. Отсюда $Ad_k = \alpha_k^{-1}(g_{k+1} - g_k)$. Значит,

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

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

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

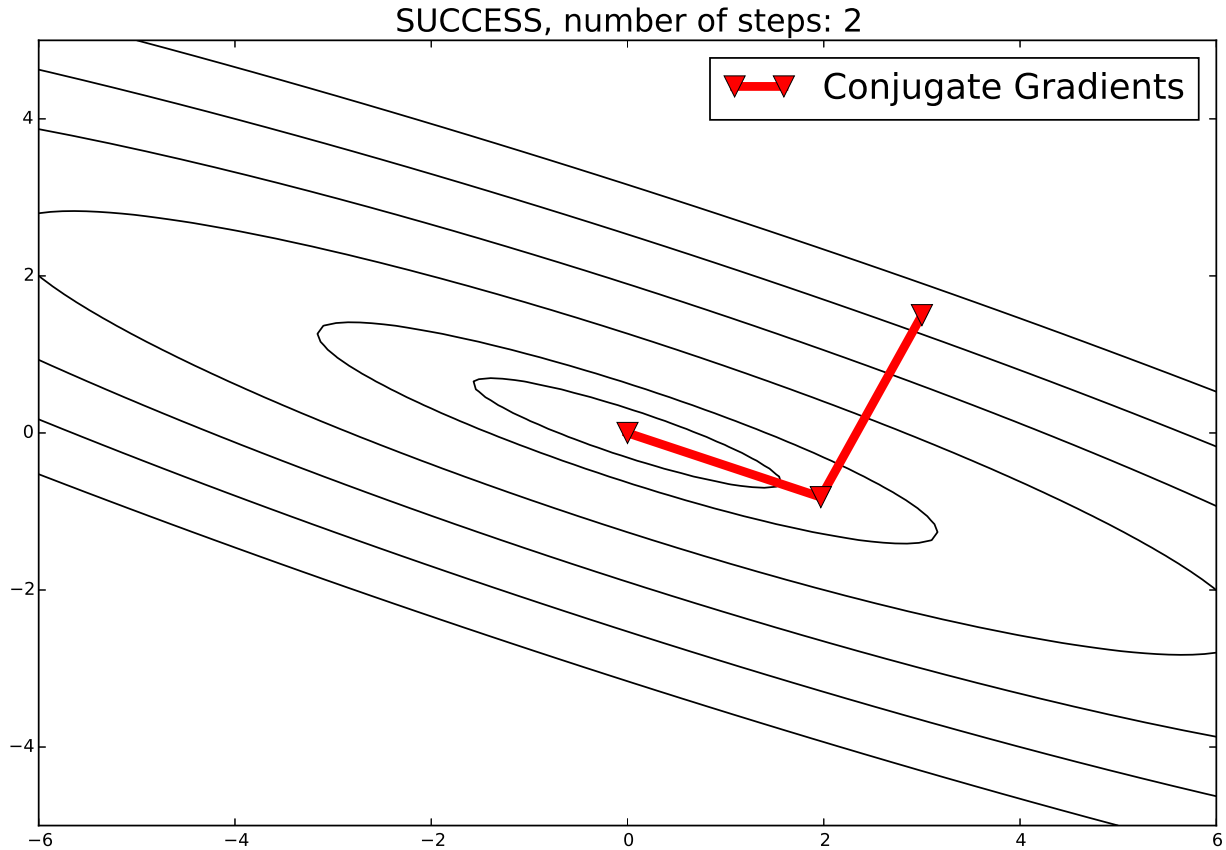
Метод сопряженных градиентов (CG). Общая схема – 4

Метод сопряженных градиентов (эффективная версия):

$g_0 \leftarrow Ax_0 - b$
$d_0 \leftarrow -g_0$
$k \leftarrow 0$
while $\|g_k\|_2 > \varepsilon \|g_0\|_2$ **do**
  $\alpha_k \leftarrow \frac{g_k^T g_k}{d_k^T A d_k}$
  $x_{k+1} \leftarrow x_k + \alpha_k d_k$
  $g_{k+1} \leftarrow g_k + \alpha_k A d_k$
  $\beta_k \leftarrow \frac{g_{k+1}^T g_{k+1}}{g_k^T g_k}$
  $d_{k+1} \leftarrow -g_{k+1} + \beta_k d_k$
  $k \leftarrow k + 1$
end while

- Одно матрично-векторное произведение за итерацию!

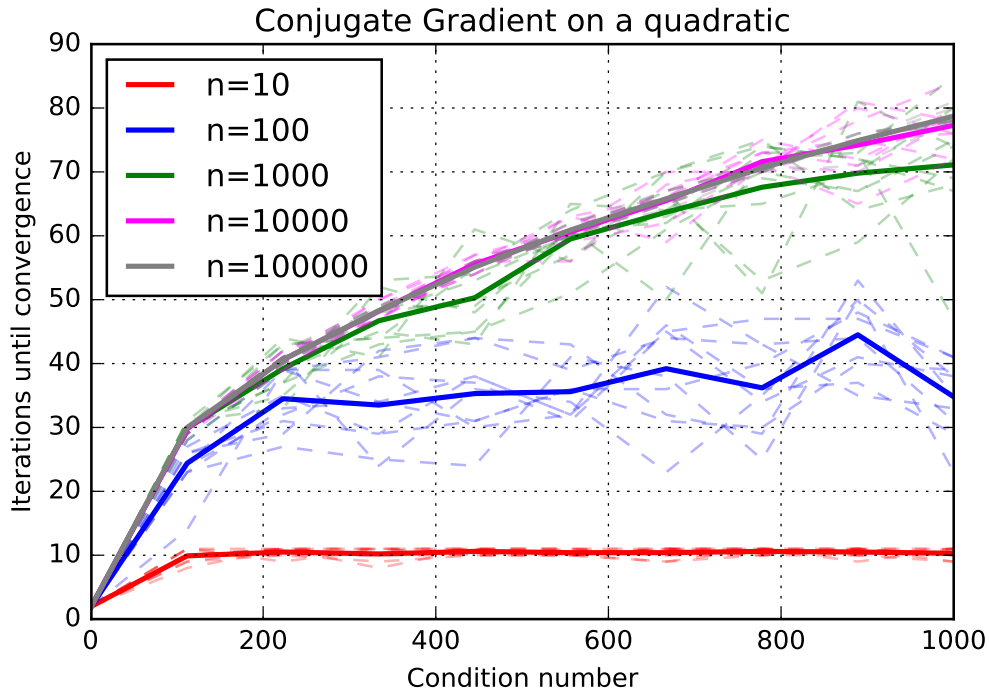
Метод сопряженных градиентов: траектория



- Всегда ≤ 2 итерации в двумерном случае.

Зависимость от обусловленности и размерности задачи

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

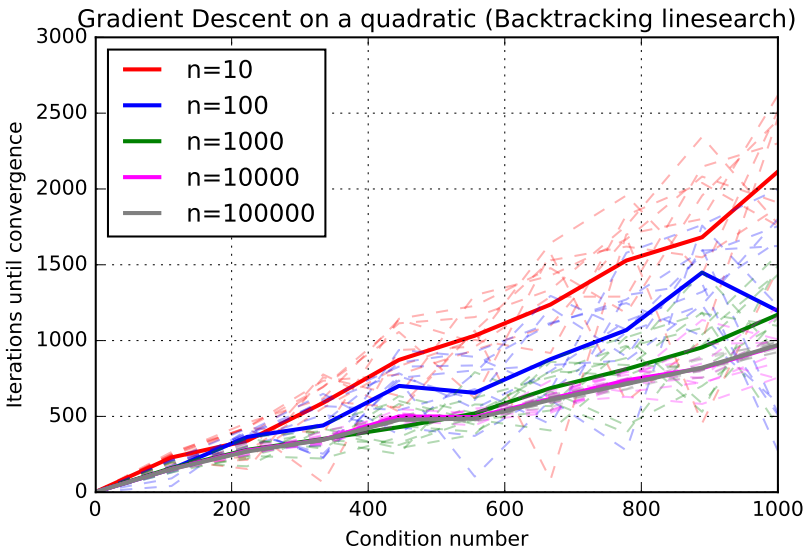


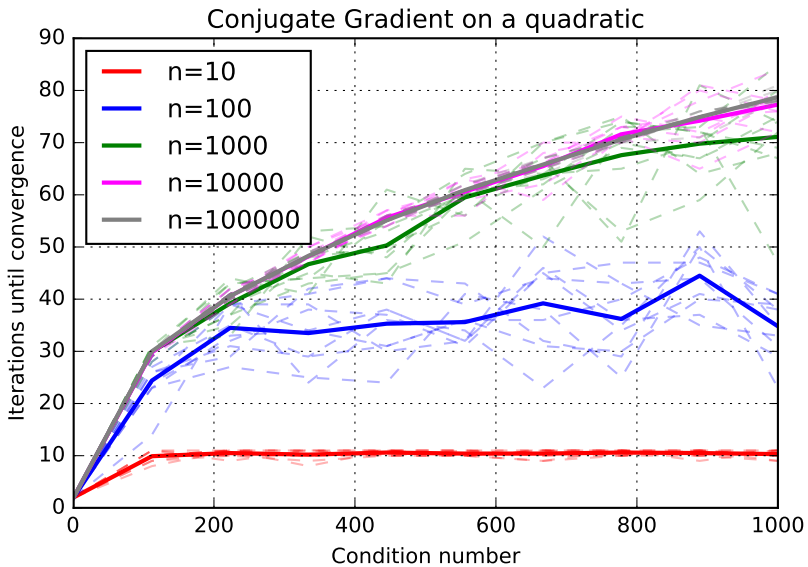
- Всегда $\le n$ итераций (м.б. чуть больше из-за погрешностей).
- В худшем случае: $O(\sqrt{\kappa})$ (κ — число обусловленности).
- С ростом размерности число итераций не увеличивается!

Сравнение CG с градиентным спуском

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

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





- $O(\kappa)$ против $O(\sqrt{\kappa})$ (κ — число обусловленности).
- Огромная разница уже даже при небольших κ!

Предобуславливание в методе сопряженных градиентов

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

- Выполним эквивалентное преобразование 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 задача: $\tilde{A}\tilde{x} = \tilde{b}$, где $\tilde{A} := S^{-T}AS^{-1}$, $\tilde{b} := S^{-T}b$.

- Решение исходной системы: $x = S^{-1}\tilde{x}$.

Предобуславливатель: $M := S^T S$.

- Идеальный предобуславливатель: $M \approx A$

$$\tilde{A} \approx S^{-T}(S^T S)S^{-1} = I_n$$

В этом случае сходимость будет примерно за одну итерацию.

Схема предобусловленного CG

Обычный CG:

$g_0 \leftarrow Ax_0 - b$
$d_0 \leftarrow -g_0$
$k \leftarrow 0$
while $\|g_k\|_2 > \varepsilon\|g_0\|_2$ **do**
  $\alpha_k \leftarrow \frac{g_k^T g_k}{d_k^T A d_k}$
  $x_{k+1} \leftarrow x_k + \alpha_k d_k$
  $g_{k+1} \leftarrow g_k + \alpha_k A d_k$
  $\beta_k \leftarrow \frac{g_{k+1}^T g_{k+1}}{g_k^T g_k}$
  $d_{k+1} \leftarrow -g_{k+1} + \beta_k d_k$
  $k \leftarrow k + 1$
end while

Предобусловленный CG:

$g_0 \leftarrow Ax_0 - b$
$d_0 \leftarrow -M^{-1} g_0$
$k \leftarrow 0$
while $\|g_k\|_2 > \varepsilon\|g_0\|_2$ **do**
  $\alpha_k \leftarrow \frac{g_k^T M^{-1} g_k}{d_k^T A d_k}$
  $x_{k+1} \leftarrow x_k + \alpha_k d_k$
  $g_{k+1} \leftarrow g_k + \alpha_k A d_k$
  $\beta_k \leftarrow \frac{g_{k+1}^T M^{-1} g_{k+1}}{g_k^T M^{-1} g_k}$
  $d_{k+1} \leftarrow -M^{-1} g_{k+1} + \beta_k d_k$
  $k \leftarrow k + 1$
end while

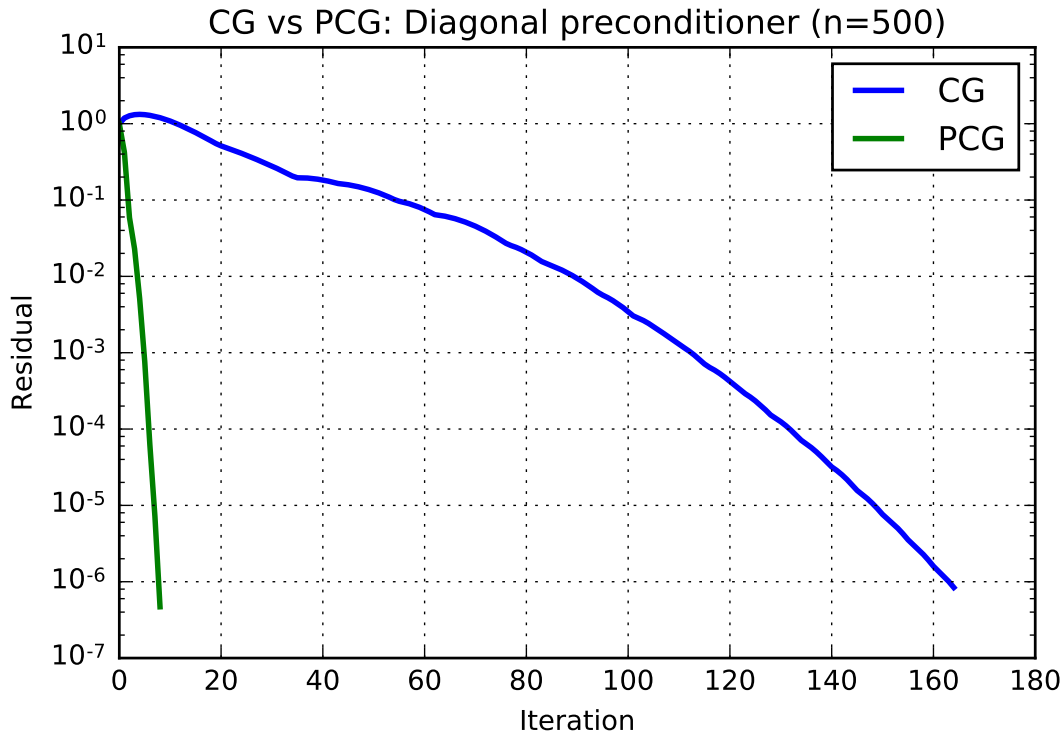
- Дополнительно нужна процедура вычисления $M^{-1} g_k$.
- Обычно применяют для хорошо структурированных M.
  - Примеры: диагональная, ленточная, разреженная и т.д.

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



- Хороший предобуславливатель существенно ускоряет метод.