Лекция 9

Ilya Yaroshevskiy

24 марта 2021 г.

Содержание

1 Метод сопряженных градиентов

$$x^{k+1} = x^k + \alpha_k p^k \quad k = 0, 1, \dots \tag{1}$$

$$p^k = -\nabla f(x^*)$$

Направление убывания может носить зигзагообразный характер. Будем находить вектор p^k не только через антиградиент, но и через p^{k-1}.

$$p^{k+1} = -\nabla(x^{k+1}) + \beta_k p^k \tag{2}$$

β_k выбираются так, чтобы получалась последовательность A-ортогональных векторов $p^0, p^1, \dots$. Из условия:

$$(Ap^{k+1}, p^k) = 0$$

$$\beta_k = \frac{(A\nabla f(x^{k+1}), p^k)}{(Ap^k, p^k)} \tag{3}$$

Для квадратичных функция:

$$\alpha_k = -\frac{(\nabla f(x^k), p^k)}{(Ap^k, p^k)} \tag{4}$$

Утвержение: итерационный процесс, который описывается формулами 1, 2, 3, 4, с положительно определенной симметричной матрицей A дает точки $x^0, \dots, x^k$ и векторы, такие что если $\nabla f(x^i) \neq 0$, $0 \leq i < k \leq n - 1$, то векторы $p^0, \dots, p^k$ — A-ортогональны, а градиенты $\nabla f(x^0), \dots, \nabla f(x^i)$ — взаимно ортогональны.

Т.к. p^k в 2 A-ортогональны, то метод гарантирует нахождение точки минимума сильно выпуклой квадратичной функции не более чем за n шагов

$$x^{k+1} = x^k + \alpha_k p^k \quad k = 0, 1, \dots \quad x^0 \in E_k \quad p^0 = -\nabla f(x^0) \tag{5}$$

$$f(x^k + \alpha_k p^k) = \min_{\alpha > 0} f(x^k + \alpha p^k) \quad k = 0, 1, \dots \tag{6}$$

$$p^{k+1} = -\nabla f(x^{k+1}) + \beta_k p^k \quad k = 0, 1, \dots \tag{7}$$

$$\beta_k = \frac{\|\nabla f(x^{k+1})\|^2}{\|\nabla f(x^k)\|^2} \tag{8}$$

Точное определение α_k возможно только в редких случаях, т.к. p^k могут быть не A-ортогональными. В этом методе используется следующий практический прием: через N шагов производится обновление метода, т.е. $\beta_{m \cdot N} = 0 \quad m = 1, 2, \dots$, где $m \cdot N$ — момент обновления метода(рестарта), часто полагают $N = n$ — размерность пространства E_n. Ретарт необходим для устранения накопленной погрешности метода, из-за которой вектора p^k перестанут указывать на направление убывания функции $f(x)$

Если функция хорошо апроксимируется квадратичной функции, то метод сопряженных градиентов даст маленькое количество шагов

2 Метод стохастического градиентного спуска

Этот метод по большей части связан с большими выборками. Обычные методы пострадют, из-за дорогого вычисления функции на большом наборе данных.

Наборы разбивают на K тренировочных наборов, части тренировочных наборов размера M называют minibatch. Тогда набор можно предсавить как:

$$X^{(k)} = \{x_i | i = M_k, \dots, (M_k + M - 1)\}$$

$$Y^{(k)} = \{y_i | i = M_k, \dots, (M_k + M - 1)\}$$

Определяют некоторую функцию, которую будем оптимизировать. Для каждого набора она будет выглядеть так:

$$L^{(k)}(\omega) = \sum_{i=0}^{M} L(\omega, x_{M_k+i}, y_{M_k+i}) \quad k = 0, \dots, (K-1)$$

, где ω — точк минимума

Когда определяем функцию для каждого набора, каждая составляющая ω будет находится на мини итерации:

$$\omega_p^{(k+1)} = \omega_p^{(k)} - \eta \cdot \nabla L^{(k)}(\omega_p^{(k)}) \quad k = 0, \dots, (K-1)$$

$$\omega_{p+1}^{(0)} = \omega_p^{(k)}$$

Большая итерация: $p = 0, 1, \dots$ завершается когда проходим весь набор миниитераций. Такая большая итерация называется эпохой. Когда переходим к следующей эпохе, перемешивает тренировочный набор. В результате пермешивания, элементы будут попадать в разные minibatch'и на каждой эпохе.

2.1 Adagrad (модификация)

Предлагается использовать разные η, для каждого minibatch'a.

$$\eta_p = (\eta_p^{(1)}, \dots, \eta_p^{(d)})$$

$$\eta_0 = \text{const} \quad \eta_0^{(i)} = \eta \quad i = 1, \dots, d$$

$$\omega_p = (\omega_p^{(1)}, \dots, \omega_p^{(d)})$$

$$\nabla L(\omega_p) = (g_p^{(1)}, \dots, g_p^{(d)})$$

Определим вспомогательный вектор:

$$G_p^{(i)} = (G_p^{(1)}, \dots, G_p^{(d)})???$$

$$G_p^{(i)} = \sum_{j=1}^{p} (g_i^{(i)})^2 \quad i = 1, \dots, d$$

$$\eta_p^{(i)} = \frac{\eta}{\sqrt{G_p^{(i)} + e}}$$

, где e — коэффицент $\sim 1e-8$

$$\omega_{p+1} = \omega_p - \eta_p \odot L(\omega_p)$$

, где $\odot$ — поэлементное умножение двух векторов

3 Метод покоординатного спуска

$$f(x) \to \min_{x \in E_n}$$

Алгоритм:

- Выбираем вектор $x_0 \in E_n$

$\forall i$:

1. фиксируем значение всех перменных, кроме x_i
2. $f(x_i) \to min$ любым методом одномерной оптимизации(золотое сечение наиболее популярный)
3. Проверка выполнения критерия останова:
   - $\|x^{k+1} - x^k\| \leq \varepsilon_1$
   - $\|f(x^{k+1}) - f(x^k)\| \leq \varepsilon_2$