Оптимизация в машинном обучении: краткий обзор

Александр Дьяконов

12 октября 2020 года

Методы оптимизации (пока безусловная оптимизация)

Методы нулевого порядка / метаэвристики

– используют лишь значения функции

- Покоординартный спуск
- Стохастическая оптимизация

ещё вспомним при селекции признаков

Методы первого порядка

– используют первые производные

- Градинтный спуск (+стохастический, наискорейший и т.п.)
- Квазиньютоновские методы (BFGS, …)
- Stochastic Average Gradient, momentum, Nesterov, …

Методы второго порядка

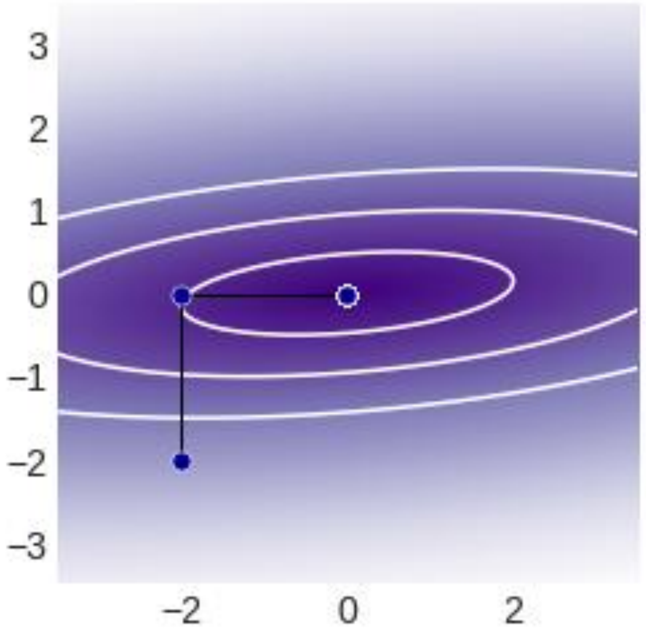
– используют вторые производные

- Метод Ньютона

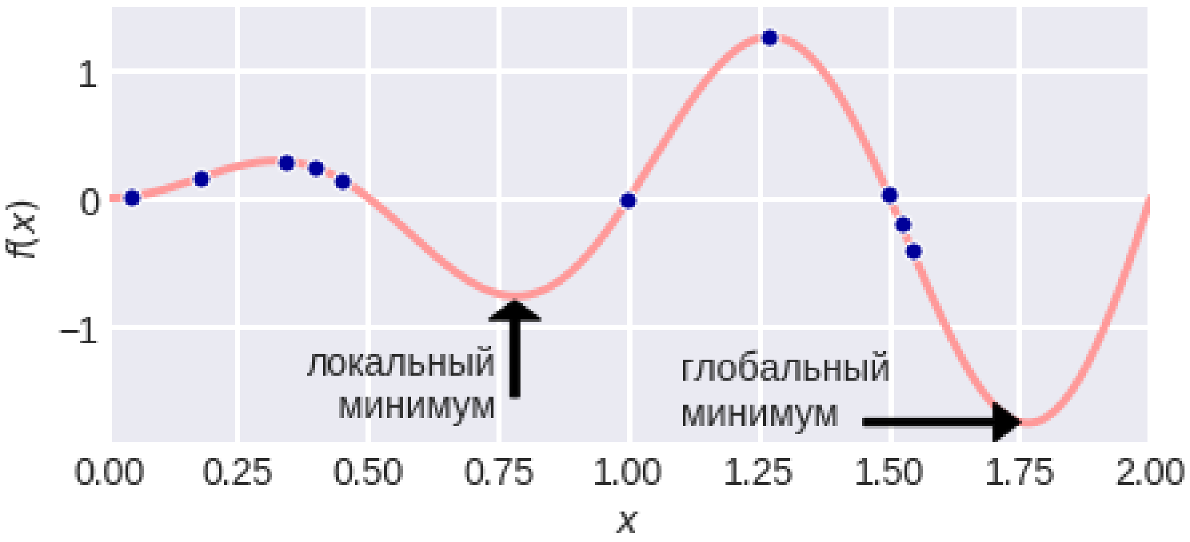
Покоординатный спуск (Coordinate descent)

Перебираем координаты вектора параметров
Оптимизируем по каждой координате (любым способом)

Итерации быстрые, но сходимость медленная
Можно, как и любой метод нулевого порядка, использовать, когда производная не вычисляется



Стохастическая оптимизация



Полный перебор

Направленный перебор

Стохастические алгоритмы

генетические алгоритмы

имитация отжига

когда будем говорить про селекцию

Градиент

$\nabla f(w_0)$ – **направление наискорейшего возрастания функции**

$$f(w) = f(w_0) + (w - w_0)^{\mathrm{т}} \nabla f(w_0) + o(\| w - w_0 \|)$$

$$f(w) - f(w_0) \approx (w - w_0)^{\mathrm{т}} \nabla f(w_0)$$

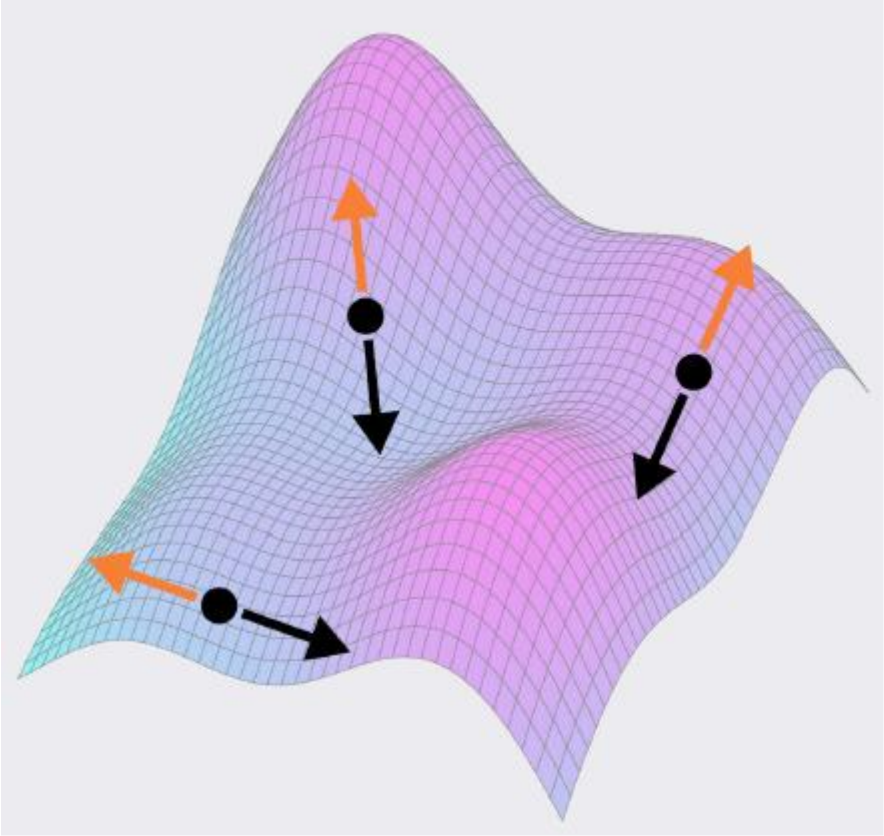
если выбирать из всех векторов $w - w_0$ **единичной нормы,**

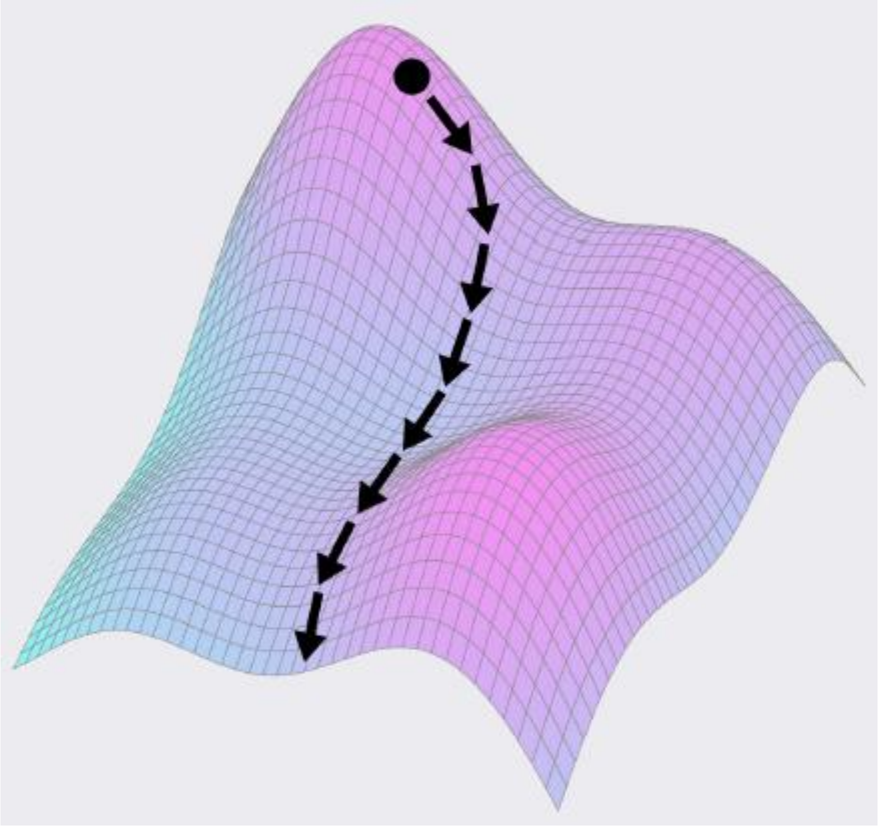
то по неравенству К-Б-Ш

$$|(w - w_0)^{\mathrm{т}} \nabla f(w_0)| \le 1 \| \nabla f(w_0) \| = \frac{\nabla f(w_0)^{\mathrm{т}}}{\| \nabla f(w_0) \|} \nabla f(w_0)$$

Антиградиент $(-\nabla f(w_0))$ – **направление наискорейшего убывания функции**

Градиент и антиградиент





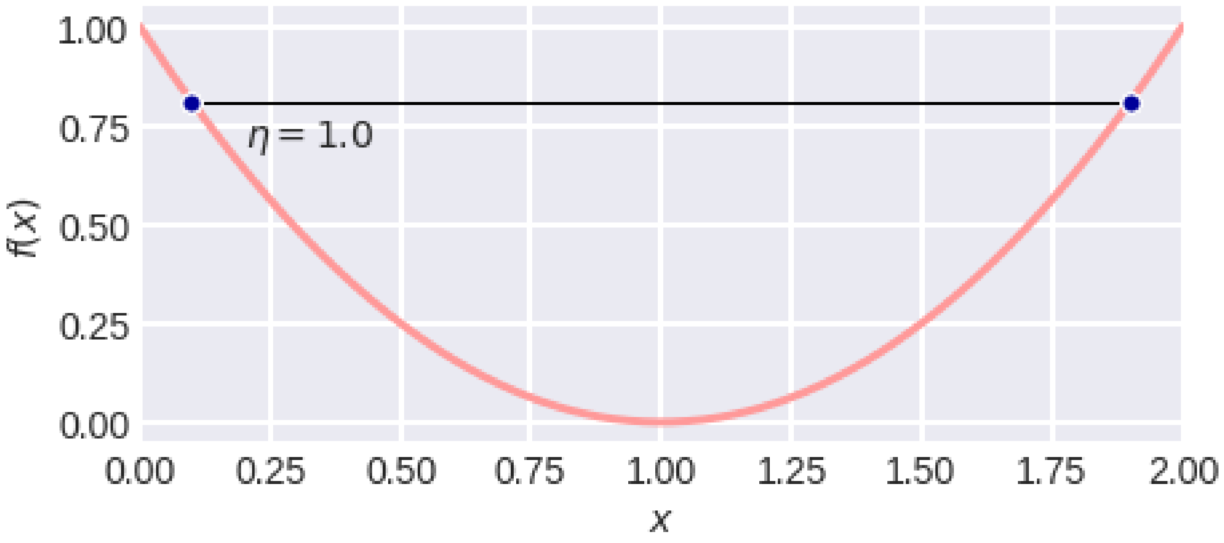
[Glassner]

Градиентный спуск (GD = Gradient Descent)

$$w^{(t+1)} = w^{(t)} - \eta \nabla L(w^{(t)})$$

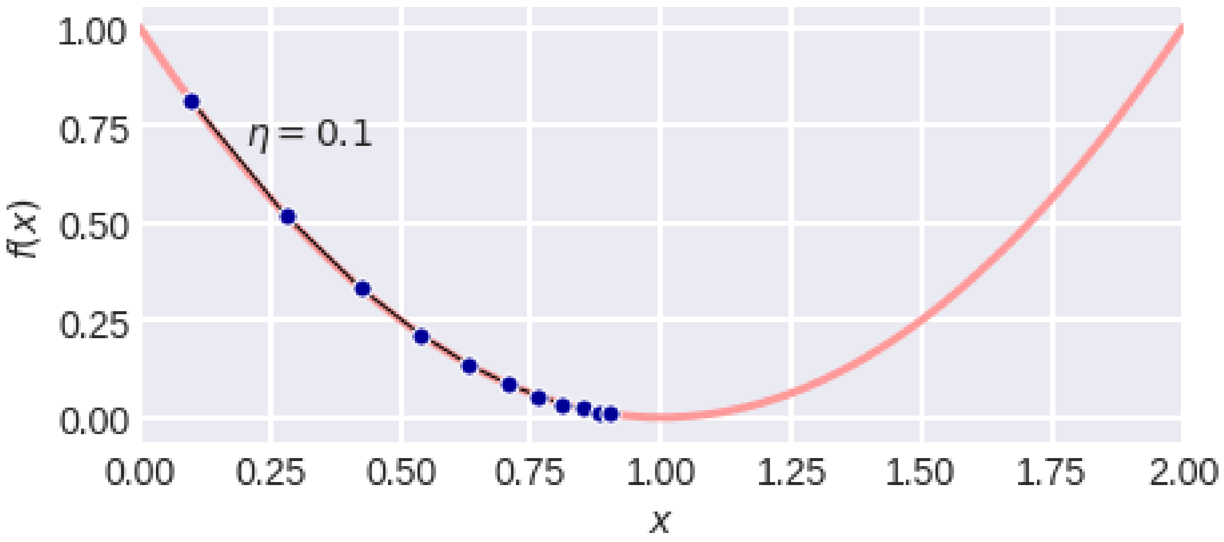
$\eta > 0$ **– шаг / темп обучения (step size / learning rate)**

Хотим $\lim\limits_{t\to\infty} w^{(t)} = \arg\min\limits_{w} L(w)$

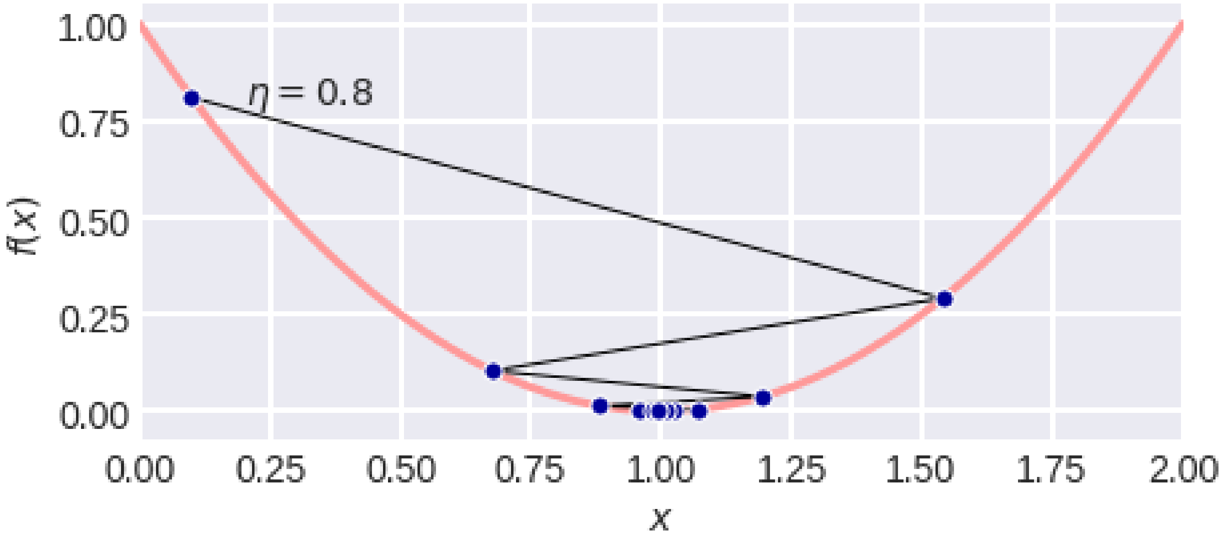


неудачно выбран темп

Градиентный спуск: проблема выбора темпа

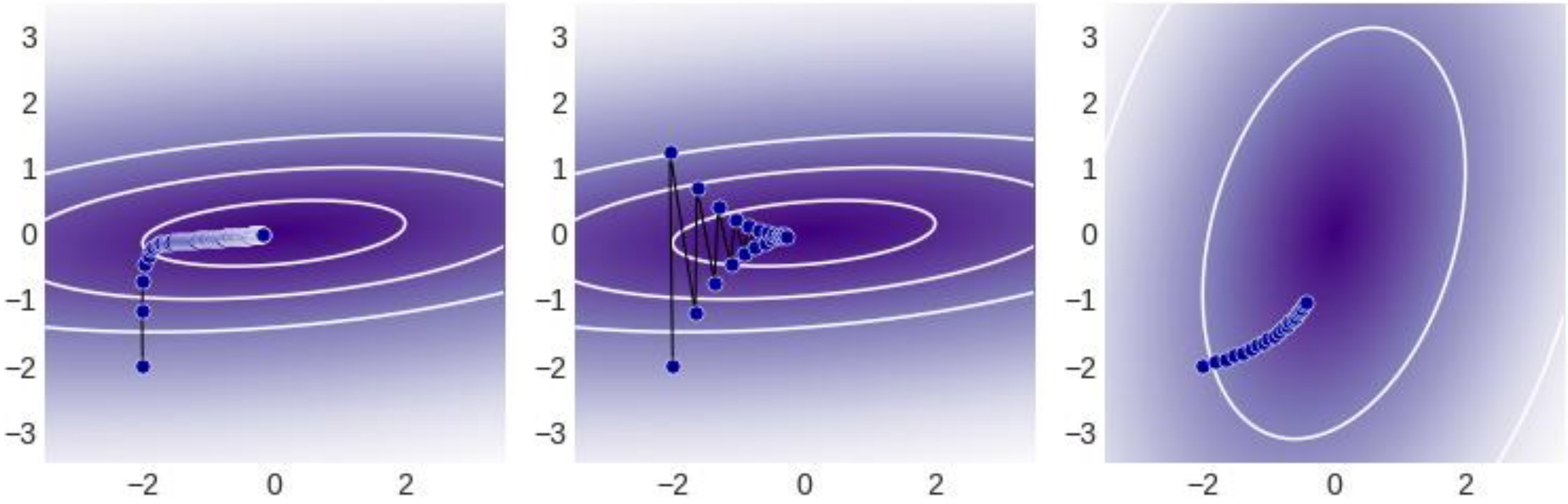


темп, возможно, маленький



темп, возможно, большой

Градиентный спуск: проблема масштаба признаков



вот для чего нормируют признаки

Проблема постоянного шага

Выбор шага – важно!
Большой – можем не сойтись
Маленький – долгая сходимость

Теорема (просто GD)

Пусть $L:\mathbb{R}^d \to \mathbb{R}$ **выпукла и дифференцируема,**
∇L **липшецева (Lipschitz continuous) с константой** $\lambda > 0$**:**

$$\| \nabla L(z_1) - \nabla L(z_2) \| \le \lambda \| z_1 - z_2 \|$$

для любых $z_1, z_2 \in \mathbb{R}^d$**.**

Тогда метод градиентного спуска с фиксированной скоростью $\eta \le 1/\lambda$ **сходится, в частности,**

$$L(z^{(t)}) - L(z^*) \le \frac{\| z^{(0)} - z^* \|^2}{2\eta t}$$

Переменный шаг

$$w^{(t+1)} = w^{(t)} - \eta^{(t)} \nabla L(w^{(t)})$$

Достаточные условия сходимости:

(иногда условия Роббинса-Монро)

$$\sum_{t=1}^{+\infty} \eta^{(t)} = +\infty$$

$$\sum_{t=1}^{+\infty} (\eta^{(t)})^2 < +\infty$$

Пример

$$\eta^{(t)} = \frac{1}{t}$$

Leon Bottou's «Tricks» http://research.microsoft.com/pubs/192769/tricks-2012.pdf

Скорость сходимости

Для выпуклых функций

$$L(w^{(t)}) - \min L(w) \le O\left(\frac{1}{\sqrt{k}}\right)$$

Для строго выпуклых функций

$$L(w^{(t)}) - \min L(w) \le O\left(\frac{1}{k}\right)$$

Без дополнительных предположений нельзя улучшить оценки

Оптимальный шаг

Наискорейший градиентный спуск

$$w^{(t+1)} = w^{(t)} - \eta^{(t)} \nabla L(w^{(t)})$$

$$\eta^{(t)} = \arg\min_{\eta} L(w^{(t)} - \eta \nabla L(w^{(t)}))$$

точная оптимизация в направлении антиградиента

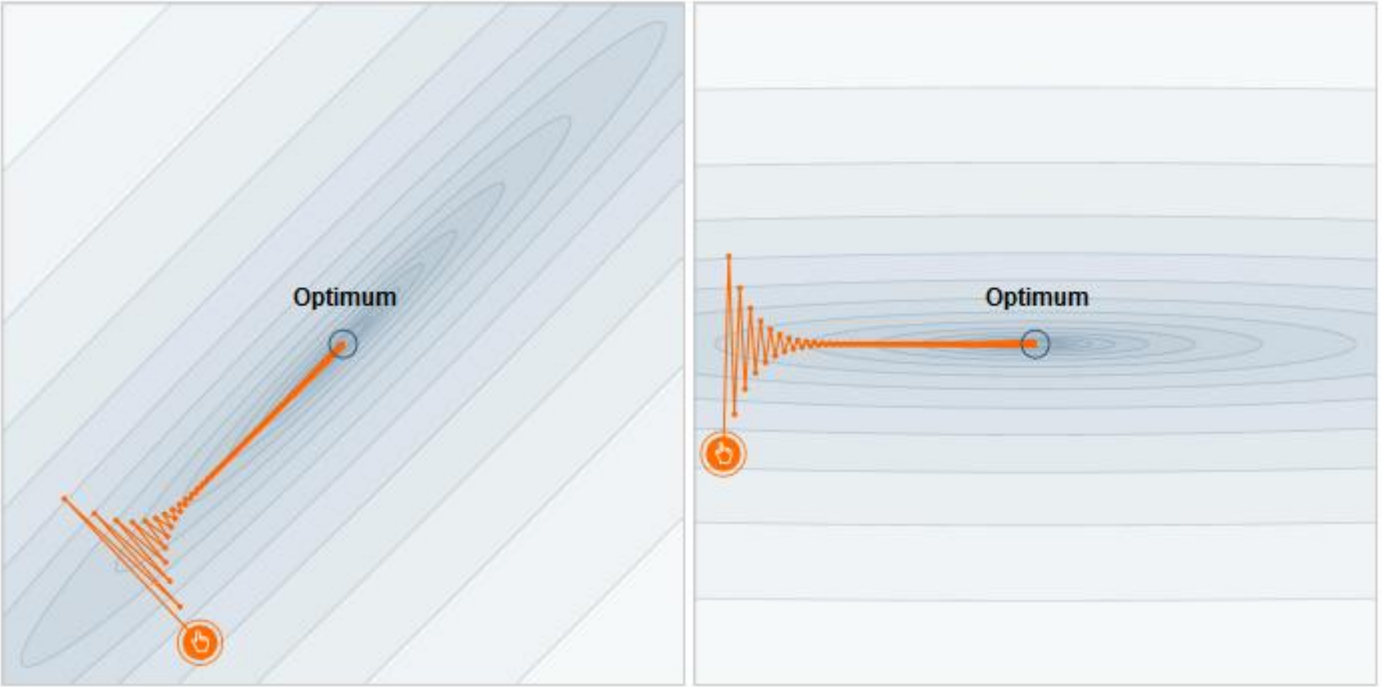
Свойства градиентного спуска

+ если функция выпуклая градиентный спуск сойдётся в минимум (при правильном выборе шагов)

– если нет – в один из локальных минимумов

+ простой метод

+ может использоваться в онлайн-режиме (см. дальше)



https://distill.pub/2017/momentum/

Стохастический градиентный спуск (SGD = Stochastic gradient descent)

Если есть «большая» сумма

(если без регуляризации)

$$L(w)=\sum_{t=1}^{m}L_t(w)$$

Слишком долго вычислять полный градиент!

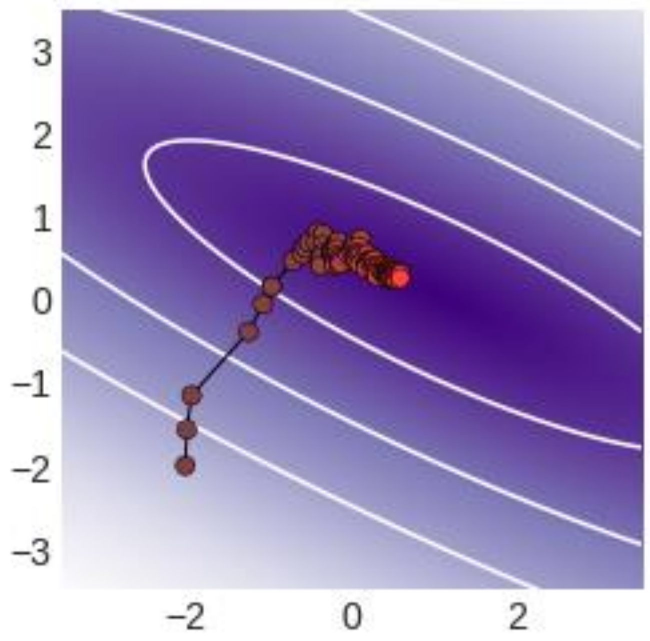
Не вычисляем полный градиент:

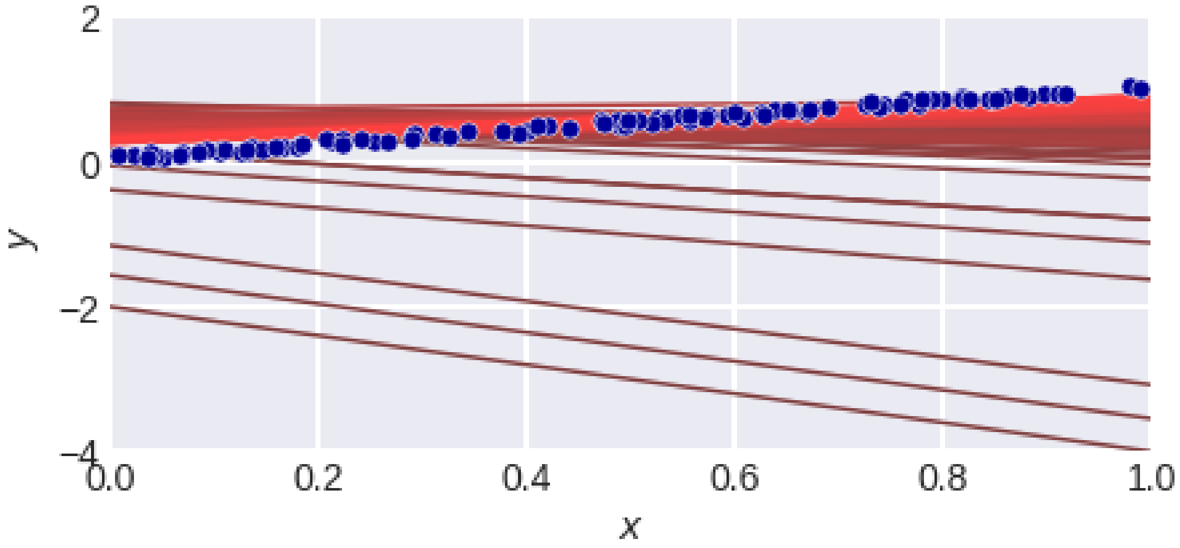
$$\nabla L(w)=\sum_{t=1}^{m}\nabla L_t(w)$$

А выцепляем случайное (!) слагаемое и
делаем шаг с помощью такого частичного антиградиента:

$$w^{(t+1)}=w^{(t)}-\eta\nabla L_t(w^{(t)})$$

Стохастический градиентный спуск (SGD)





- **Можно учиться в online-режиме**

(когда функция становится известна по частям – некоторые слагаемые),

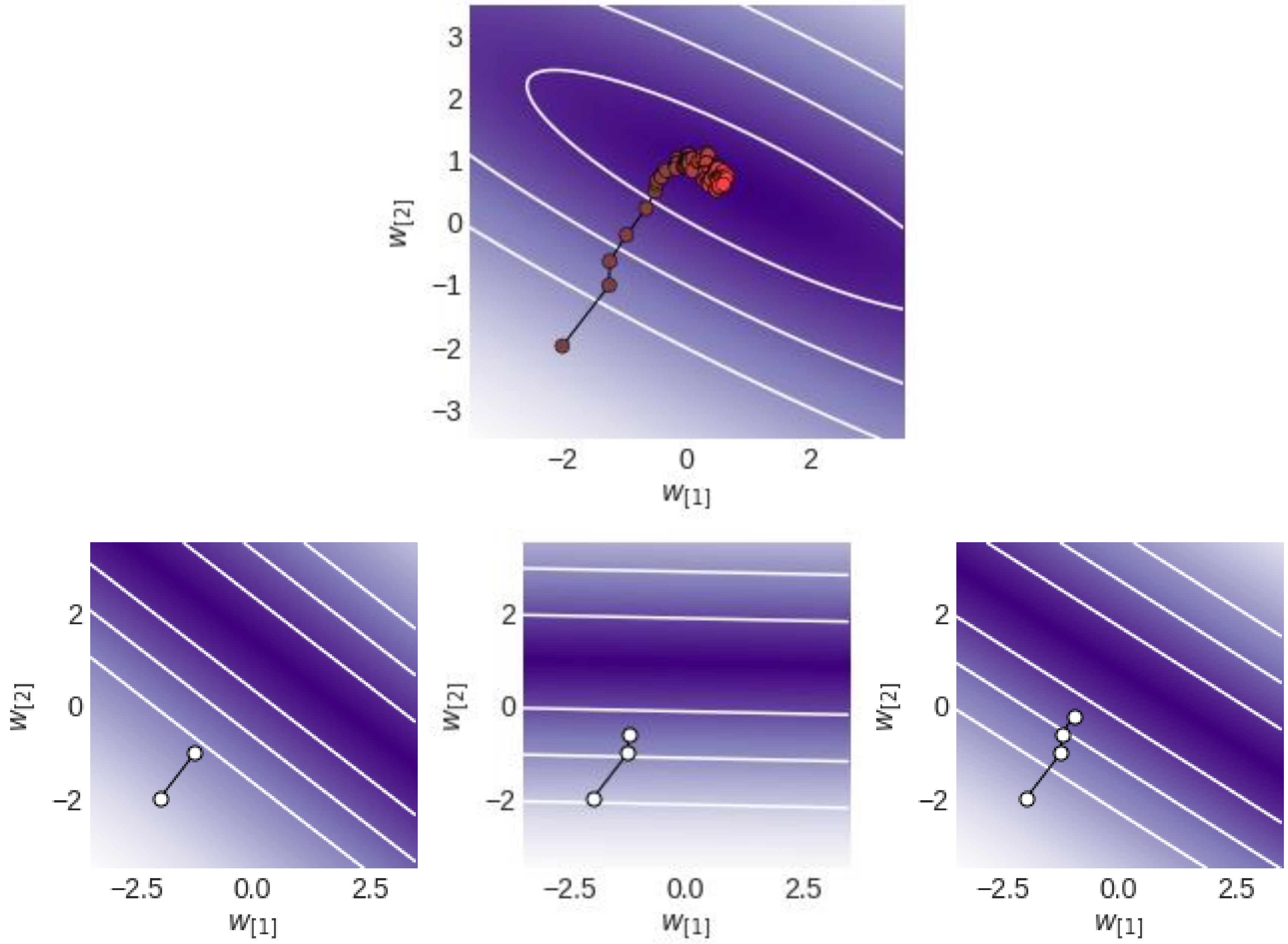
но порядок здесь не совсем случайный

- **Метод быстрый**

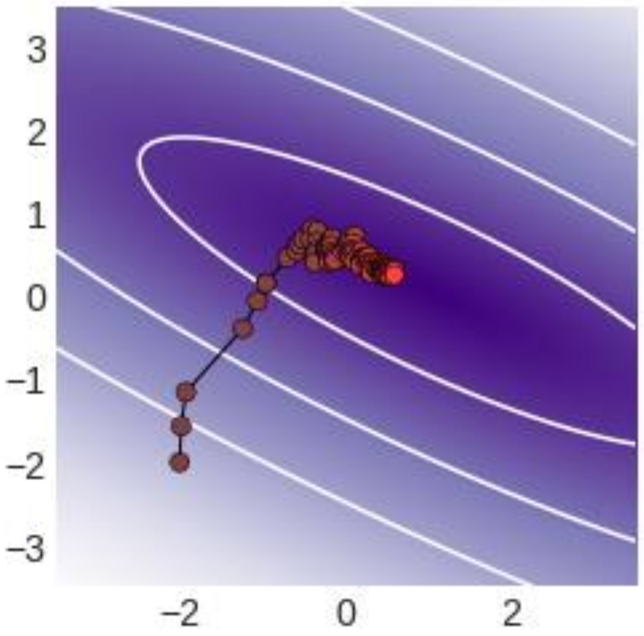
(не надо вычислять градиенты всех слагаемых на каждом шаге)

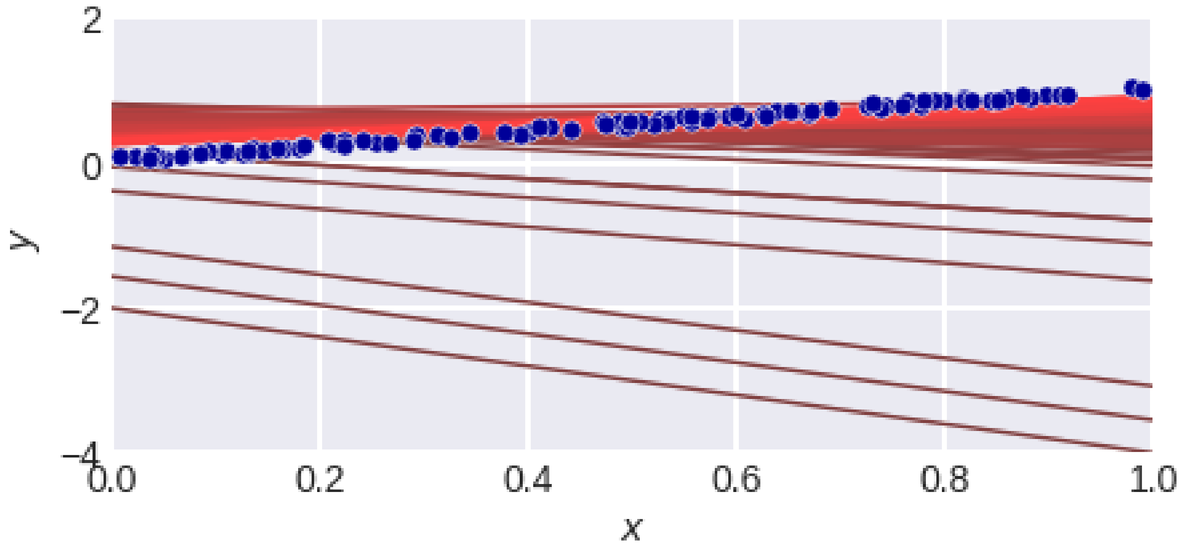
- **темп сходимости определяется на CV**

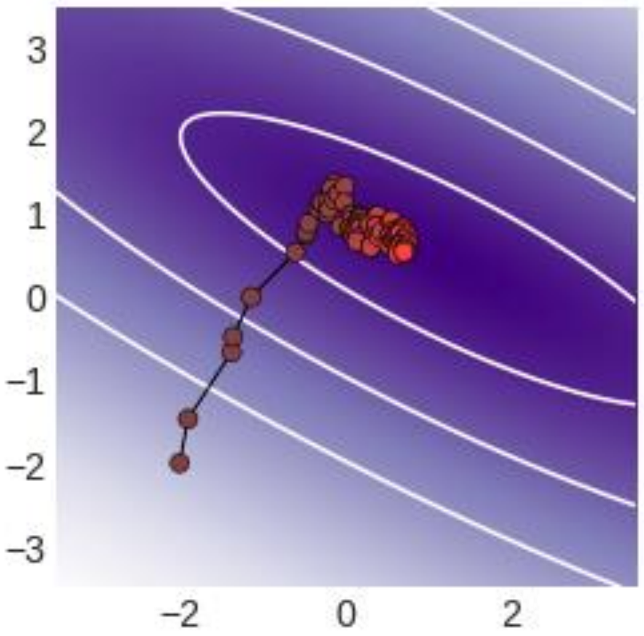
Стохастический градиентный спуск (SGD)

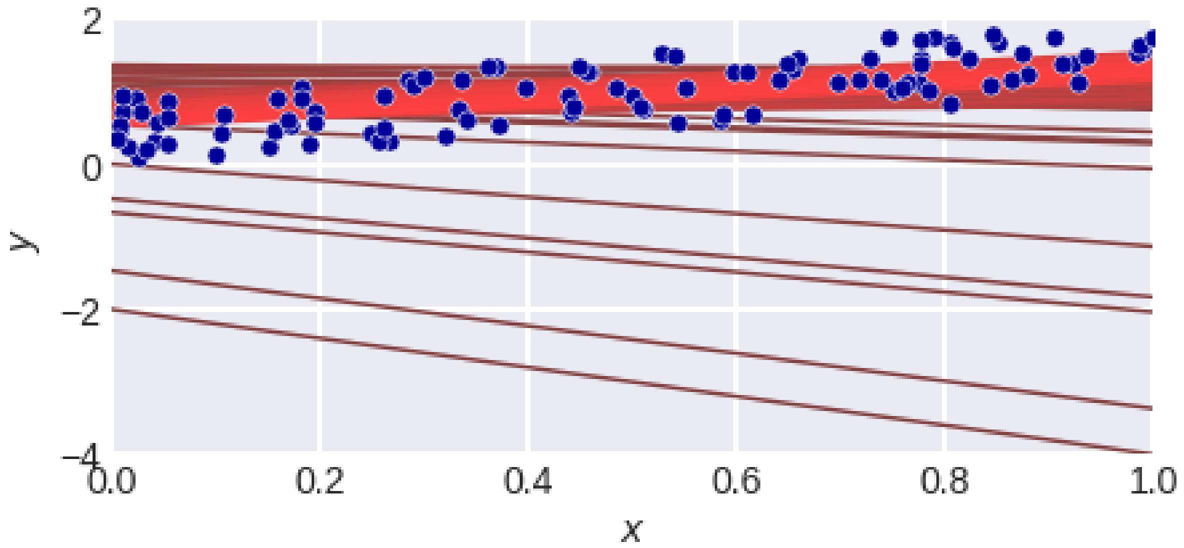


Стохастический градиентный спуск (SGD)









Критерии останова

- **слабо меняется значение функции**

$$|L(w^{(t+1)}) - L(w^{(t)})| < \varepsilon$$

- **слабо меняется аргумент**

$$\| w^{(t+1)} - w^{(t)} \| < \varepsilon$$

- **слишком много итераций**

$$t \geq t_{\max}$$

нормализация…

многое зависит от начальной точки…

Пакетное (Batch / Offline)-обучение

$$w^{(t+1)} = w^{(t)} - \eta \sum_{i=1}^{m} \nabla L_i(w^{(t)})$$

Онлайн (Online)-обучение

stochastic gradient descent – если слагаемые случайные

$$w^{(t+1)} = w^{(t)} - \eta \nabla L_i(w^{(t)})$$

Minibatch Online обучение

$$w^{(t+1)} = w^{(t)} - \eta \sum_{i \in I} \nabla L_i(w^{(t)})$$

See Yoshua Bengio's «Practical recommendations for gradient-based training of deep architectures» http://arxiv.org/abs/1206.5533

Stochastic average gradient (SAG)

на каждом t-м шаге выбираем случайный индекс $j \in \{1, 2, \ldots, m\}$

$$g_j = \nabla L_j(w^{(t)})$$

$$w^{(t+1)} = w^{(t)} - \frac{\eta}{m} \sum_{i=1}^{m} g_i$$

Другие приёмы

- **Momentum**
- **адаптивные шаги**

см. http://github.com/Dyakonov/DL/

(оптимизация в DL)

Метод градиентного спуска в машинном обучении

Оптимизация в ML: минимизация эмпирического риска (empirical training loss) + регуляризатора (regularizer term)

пока пусть нет регуляризатора

$$\frac{1}{2}\sum_{i=1}^{m}(a(x_i \mid w) - y_i)^2 \to \min$$

Gradient Descent

$$w^{(t+1)} = w^{(t)} - \eta\sum_{i=1}^{m}(a(x_i \mid w^{(t)}) - y_i)\frac{\partial a(x_i \mid w^{(t)})}{\partial w}$$

Метод градиентного спуска в машинном обучении

Gradient Descent в линейной модели

$$a(x \mid w) = w^{\mathrm{T}} x$$

$$w^{(t+1)} = w^{(t)} - \eta \sum_{i=1}^{m} (a(x_i \mid w^{(t)}) - y_i) x_i$$

Есть аналитическое решение,
но данные м.б. большими
функция ошибки чуть сложнее

в матричной форме $w^{(t+1)} = w^{(t)} - \eta X^{\mathrm{T}} (a - y)$

Stochastic Gradient Descent (SGD)

из обучения выбирается случайный объект x_i

$$w^{(t+1)} = w^{(t)} - \eta_t (a(x_i \mid w^{(t)}) - y_i) x_i$$

Метод градиентного спуска в машинном обучении

если заменить в формуле значение $a(x_i \mid w^{(t)})$,
т.е. оценку принадлежности к классу 1 на округлённое значение,
т.е. предсказываемую метку... то получим
алгоритм персептрона

– один из первых алгоритмов линейной классификации
(Розенблат, 1958)

Гарантированно находит разделяющую классы прямую,
если она существует

SGD может применяться в онлайн-режиме (Online Learning),
когда объекты поступают по одному и на больших данных

Пример градиентного спуска – квадратичный функционал (*)

Рассмотрим функцию

$$f(w)=\frac{1}{2}w^{\mathrm{т}}Aw-b^{\mathrm{т}}w$$

пусть матрица симметричная и невырожденная

$$w^{(t+1)}=w^{(t)}-\alpha(Aw^{(t)}-b)$$

трюк… симметричная матрица допускает разложение

$$A=Q\Lambda Q^{\mathrm{т}}$$

$$\Lambda=\mathrm{diag}(\lambda_1,\ldots,\lambda_n)$$

переход к новым координатам:

$$v=Q^{\mathrm{т}}(w-w^*),$$

где $w^*=A^{-1}b=Q\Lambda^{-1}Q^{\mathrm{т}}b$ **– оптимальное решение, тогда**

$$Q^{\mathrm{т}}w^{(t+1)}=Q^{\mathrm{т}}w^{(t)}-\alpha Q^{\mathrm{т}}(Q\Lambda Q^{\mathrm{т}}w^{(t)}-b)$$

Пример градиентного спуска – квадратичный функционал (*)

$$Q^{\mathrm{T}} w^{(t+1)} = Q^{\mathrm{T}} w^{(t)} - \alpha Q^{\mathrm{T}} (Q \Lambda Q^{\mathrm{T}} w^{(t)} - b)$$

$$Q^{\mathrm{T}} (w^{(t+1)} - w^{*}) = Q^{\mathrm{T}} (w^{(t)} - w^{*}) - \alpha (\Lambda Q^{\mathrm{T}} w^{(t)} - Q^{\mathrm{T}} b)$$

$$v^{(t+1)} = v^{(t)} - \alpha (\Lambda Q^{\mathrm{T}} w^{(t)} - \Lambda Q^{\mathrm{T}} w^{*})$$

$$v^{(t+1)} = v^{(t)} - \alpha \Lambda v^{(t)} = (I - \alpha \Lambda) v^{(t)}$$

в новом пространстве всё покоординатно...

$$v_{[i]}^{(t+1)} = (1 - \alpha \lambda_i) v_{[i]}^{(t)}$$

Пример градиентного спуска – квадратичный функционал (*)

Норма вектора в новом пространстве – расстояние до оптимума

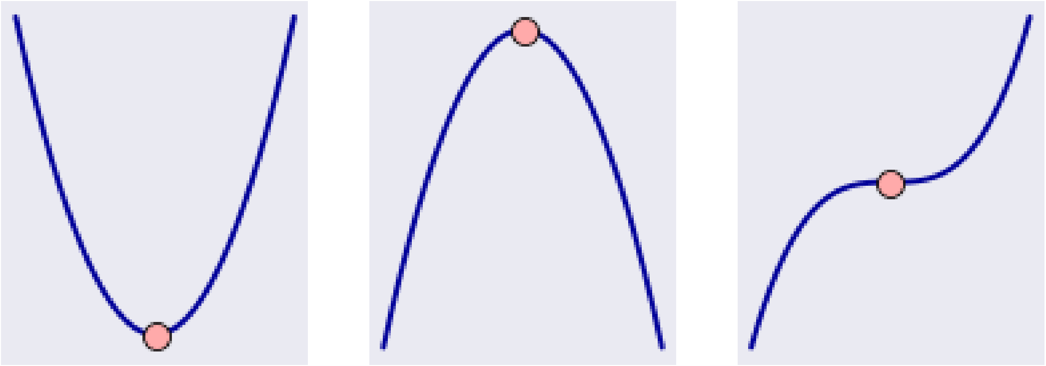
$$v_{[i]}^{(t)} = (1-\alpha\lambda_i)^t v_{[i]}^{(0)}$$

Для сходимости

$$|1-\alpha\lambda_i| < 1$$

Вопрос – какой темп сходимости оптимален?

Стационарные точки



локальный минимум | локальный максимум | седловая точка

Особенность многомерных пространств

В пространствах большой размерности стационарные точки, как правило, седловые (а не локальные минимумы и максимумы)

$$f(w) = f(w_0) + (w - w_0)^{\mathrm{т}} \nabla f(w_0) + \frac{1}{2}(w - w_0)^{\mathrm{т}} H (w - w_0) + o(\| w - w_0 \|^2)$$

зависит от с.з. матрицы Гессе

Если есть и положительные и отрицательные – седло

Если представить, что знак определяется подбрасыванием монетки…

В любом случае, полезно смотреть за нормой градиента – попали ли в стационарную точку

Другой взгляд на градиентный метод

$$f(w) = f(w_0) + (w - w_0)^{\mathrm{т}} \nabla f(w_0) + \frac{1}{2}(w - w_0)^{\mathrm{т}} H (w - w_0) + o(\| w - w_0 \|^2)$$

$$f(w) \approx f(w_0) + (w - w_0)^{\mathrm{т}} \nabla f(w_0) + \frac{1}{2}(w - w_0)^{\mathrm{т}} H (w - w_0)$$

$$\min f(w) \approx \min \left[f(w_0) + (w - w_0)^{\mathrm{т}} \nabla f(w_0) + \frac{1}{2}(w - w_0)^{\mathrm{т}} H (w - w_0) \right]$$

$$\nabla_w \left[f(w_0) + (w - w_0)^{\mathrm{т}} \nabla f(w_0) + \frac{1}{2}(w - w_0)^{\mathrm{т}} H (w - w_0) \right] = 0$$

$$\nabla f(w_0) + (w - w_0)^{\mathrm{т}} H = 0$$

Другой взгляд на градиентный метод

Получаем формулу

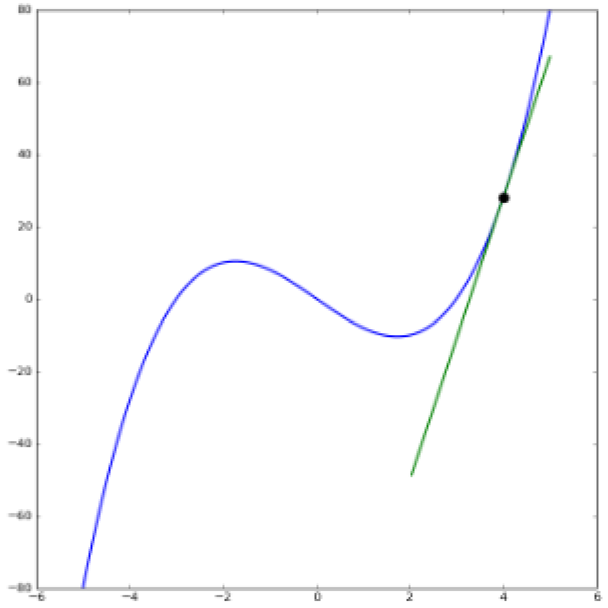
$$w = w_0 - H^{-1}\nabla f(w_0)$$

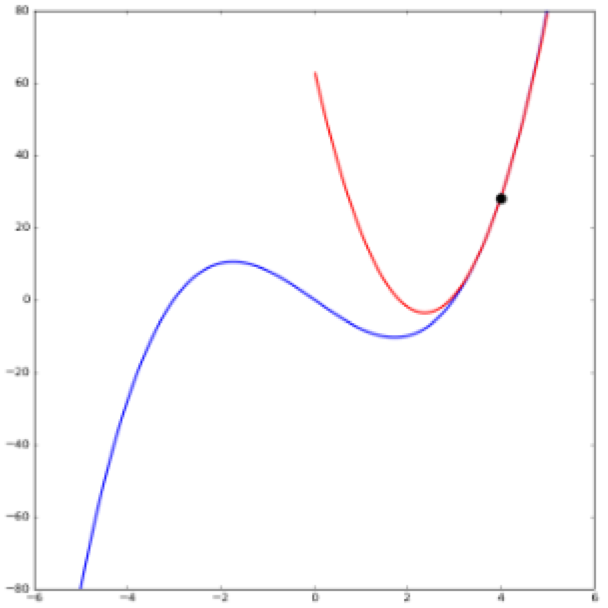
1) если положить $H = I$, то получаем метод градиентного спуска

2) если применяем формулу так – метод Ньютона

$$w^{(t+1)} = w^{(t)} - H_{(t)}^{-1} g^{(t)}$$

Метод Ньютона





Градиентный спуск использует только первые производные

~ Локальная линейная аппроксимация

Метод Ньютона использует вторые производные

~ аппроксимация рядом Тейлора до 2го порядка

Шаг по методу Ньютона

$$w^{(t+1)} = w^{(t)} - H_{(t)}^{-1} g^{(t)}$$

Нет гиперпараметров и темпа обучения!

Применим, если матрица Гессе положительно определённая

$$w^{(t+1)} = w^{(t)} - (H_{(t)} + \alpha I)^{-1} g^{(t)}$$

обращение матрицы трудоёмко (но не всегда необходимо –
– там умножается на вектор

Матрица Гессе – матрица вторых производных

$$H = \frac{\partial}{\partial w}\left[\frac{\partial L(w)}{\partial w}\right]^{\mathrm{T}}$$

Квази-ньютоновские методы

- **BFGS = Бройдена-Флетчера-Гольдфарба-Шанно**

вместо обращение Гессиана $H_{(t)}^{-1}$ – (O(n³) операций)

– низкоранговая аппроксимация обратного гессиана $M_{(t)} \approx H_{(t)}^{-1}$,

которая итеративно уточняется (O(n²) для хранения)

$$w^{(t+1)} = w^{(t)} - \varepsilon M_{(t)} g^{(t)}$$

ε специально подбирается линейным поиском

http://fa.bianp.net/teaching/2018/eecs227at/quasi_newton.html

- **Limited memory BFGS – с ограниченной памятью**

Хорошо на всех данных (не мини батчах)

Le et al, «On optimization methods for deep learning, ICML 2011»

Ba et al, «Distributed second-order optimization using Kronecker-factored approximations», ICLR 2017

Дальше, что понадобится в SVM
(немного про условную оптимизацию)

Оптимизация с ограничениями

$$f(w) \to \min$$
$$g_i(w) \le 0,\ i \in I,$$
$$h_j(w) = 0,\ j \in J.$$

Выпишем Лагранжиан

$$L(w,\alpha,\beta) = f(w) + \sum_{i \in I} \alpha_i g_i(w) + \sum_{j \in J} \beta_j h_j(w)$$

$$\alpha = (\alpha_i)_{i \in I} \ge 0,\ \beta = (\beta_j)_{j \in J}.$$

Заметим, что

$$\max_{\alpha,\beta} L(w,\alpha,\beta)$$

обращается в бесконечность, если нарушено хотя бы одно ограничение (по g_i или h_j), в противном случае, совпадает с $f(w)$

Оптимизация с ограничениями

Поэтому можно решать такую задачу:

$$\min_{w} \max_{\alpha,\beta} L(w,\alpha,\beta)$$

из-за выпуклости всех функций min и max можно переставлять

Условия Кунна-Таккера (Karush-Kuhn-Tucker (KKT) Conditions):

в оптимальной точке

$$\alpha_i g_i(w) = 0$$

Ссылки

Более продвинутые современные подходы к оптимизации в DL см. в

http://github.com/Dyakonov/DL/

Léon Bottou «Stochastic Gradient Descent Tricks» // Microsoft Research, Redmond, WA
https://www.microsoft.com/en-us/research/wp-content/uploads/2012/01/tricks-2012.pdf

Хороший обзор методов оптимизации с интерактивными примерами
http://fa.bianp.net/teaching/2018/eecs227at/