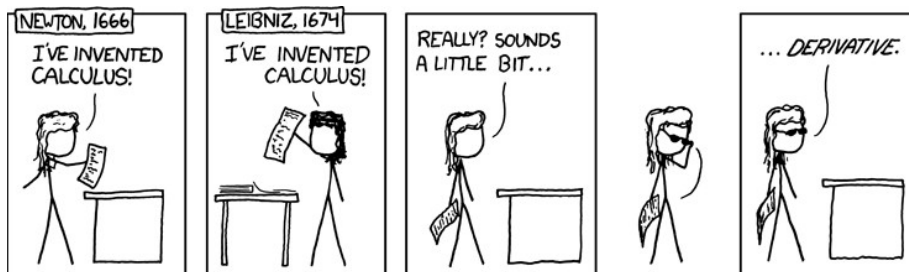


Дифференциальное исчисление и градиентный спуск

Некоторые обозначения.....	2
Понятие производной (derrivative).....	2
Касательная к графику.....	3
Таблица производных.....	3
Правила дифференцирования.....	3
Примеры.....	5
Производная и экстремумы функции.....	7
См. также.....	8
Частная производная (partial derivative).....	9
Производная по направлению.....	10
Градиент (Gradient).....	10
Свойства градиента.....	11
Матрица Якоби – матрица производных.....	11
Примеры.....	12
Оптимизация (Mathematical optimization or mathematical programming).....	14
Постановка задачи.....	14
Некоторая классификация методов оптимизации.....	15
Градиентный спуск (Gradient descent).....	17
Алгоритм.....	18
Проблемы метода.....	19
Метод покоординатного спуска (Coordinate descent).....	21
Стохастический градиентный спуск (Stochastic gradient descent, SGD)....	22
Инерционные или ускоренные градиентные методы (momentum).....	23
Пример.....	25
Метод Нестерова (Nesterov Momentum).....	25
Метод сопряжённых градиентов (Nonlinear conjugate gradient method)....	26
Некоторые определения.....	26
***.....	27
Ссылки.....	28
Задачи.....	29
Задача 1.....	29
Задача 2.....	29
Задача 3.....	29
Задача 4.....	29
Задача 5.....	29



Сложные и важные части отмечены оранжевым в тексте конспекта или в выносках

Некоторые обозначения

\mathbb{R} – множество действительных чисел

$U(x_0)$ – окрестность точки x_0 ,

$f \in C^r(\Omega)$ – функция f определена на множестве Ω и имеет непрерывные производные до r -й включительно.

Понятие производной (derrivative)

Пусть в некоторой окрестности точки $x_0 \in \mathbb{R}$ определена функция

$$f: U(x_0) \subset \mathbb{R} \rightarrow \mathbb{R}$$

Производной функции f в точке x_0 называется предел (если он существует):

$$f'(x_0) = \lim_{x \rightarrow x_0} \frac{f(x) - f(x_0)}{x - x_0} = \lim_{\Delta x \rightarrow 0} \frac{f(x_0 + \Delta x) - f(x_0)}{\Delta x} = \lim_{\Delta x \rightarrow 0} \frac{\Delta f(x)}{\Delta x}.$$

функция определённая на множестве действительных чисел, с областью определения в области действительных чисел

Общепринятые обозначения производной функции $y = f(x)$ в точке

x_0

$$f'(x_0) = f'_x(x_0) = Df(x_0) = \frac{df}{dx}(x_0) = \left. \frac{dy}{dx} \right|_{x=x_0} = \dot{y}(x_0)$$

Последнее обозначение \dot{y} обычно обозначает производную по времени (в теоретической механике и физике, исторически часто тоже).

Производная $f'(x_0)$ функции f в точке x_0 , будучи пределом, может не существовать или существовать и быть конечной или бесконечной.

Функция f является дифференцируемой в точке x_0 тогда и только тогда, когда её производная в этой точке существует и конечна:

$$f \in \mathcal{D}(x_0) \Leftrightarrow \exists f'(x_0) \in (-\infty; \infty)$$

См. также разрывы функции в точке.

Дифференцируемую функцию f можно аппроксимировать в окрестности $U(x_0)$ следующей функцией:

$$f(x) = f(x_0) + f'(x_0)(x - x_0) + o(x - x_0) \text{ при } x \rightarrow x_0$$

см. также ряд Тейлора.

где $o(x - x_0)$ – «о малое от $x - x_0$ » обозначает «бесконечно малое относительно $x - x_0$ »

см. также уравнение прямой по двум точкам

Касательная к графику

Пусть $f: U(x_0) \rightarrow \mathbb{R}$ и $x_1 \in U(x_0)$. Тогда прямая линия, проходящая через точки $(x_0, f(x_0))$ и $(x_1, f(x_1))$ задаётся уравнением

$$y = f(x_0) + \frac{f(x_1) - f(x_0)}{x_1 - x_0}(x - x_0).$$

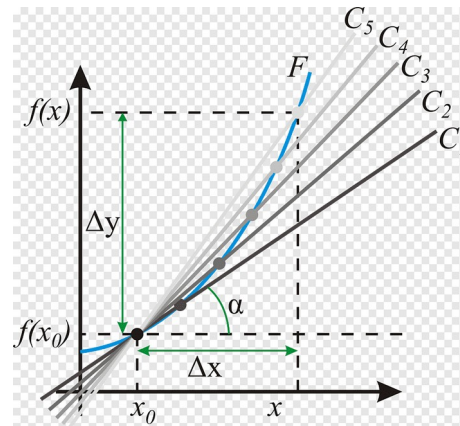
Эта прямая проходит через точку $(x_0, f(x_0))$ для любого $x_1 \in U(x_0)$, и её угол наклона $\alpha(x_1)$ удовлетворяет уравнению

$$\operatorname{tg} \alpha(x_1) = \frac{f(x_1) - f(x_0)}{x_1 - x_0}.$$

В силу существования производной функции f в точке x_0 , переходя к пределу при $x_1 \rightarrow x_0$, получаем, что существует предел $\lim_{x_1 \rightarrow x_0} \operatorname{tg} \alpha(x_1) = f'(x_0)$, а в силу непрерывности арктангенса и предельный угол $\alpha = \operatorname{arctg} f'(x_0)$.

Прямая, проходящая через точку $(x_0, f(x_0))$ и имеющая предельный угол наклона, удовлетворяющий $\operatorname{tg} \alpha = f'(x_0)$, задаётся уравнением касательной:

$$y = f(x_0) + f'(x_0)(x - x_0)$$



Секунга переходит в касательную при уменьшении приращения x

Таблица производных

ru.wikipedia.org/wiki/Таблица_производных

Правила дифференцирования


Если C – постоянное число, $f = f(x)$, $g = g(x)$ – некоторые дифференцируемые функции, то справедливы следующие правила дифференцирования:

- $C' = 0$
- $x' = 1$
- $(f + g)' = f' + g'$
- $(fg)' = f'g + fg'$
- $(Cf)' = Cf'$
- $\left(\frac{f}{g}\right)' = \frac{f'g - fg'}{g^2} \left(\frac{f}{g}\right)' = \frac{f'g - fg'}{g^2}$ для $(g \neq 0)$
- Если функция задана параметрически:

$$\begin{cases} x = x(t), \\ y = y(t), \end{cases} t \in [T_1; T_2] \quad \begin{cases} x = x(t), \\ y = y(t), \end{cases} t \in [T_1; T_2]$$

то

$$y'_x = \frac{dy}{dx} = \frac{dy}{dt} \cdot \frac{dt}{dx} = y'_t \cdot t'_x = \frac{y'_t}{x'_t} y'_x = \frac{dy}{dx} = \frac{dy}{dt} \cdot \frac{dt}{dx} = y'_t \cdot t'_x = \frac{y'_t}{x'_t}$$

- Дифференцирование сложной функции – chain rule (см. примеры 1 и 2 ниже) 

$$\frac{d}{dx}f(g(x)) = \frac{df(g)}{dg} \cdot \frac{dg(x)}{dx} = f'_g g'_x \frac{d}{dx}f(g(x)) = \frac{df(g)}{dg} \cdot \frac{dg(x)}{dx} = f'_g g'_x$$

- Формулы производной произведения и отношения обобщаются на случай n-кратного дифференцирования (формула Лейбница):

$$(fg)^{(n)} = \sum_{k=0}^n C_n^k f^{(n-k)} g^{(k)}, (fg)^{(n)} = \sum_{k=0}^n C_n^k f^{(n-k)} g^{(k)}, \text{ где } C_n^k —$$

биномиальные коэффициенты (количество сочетаний из n по k)

$$\binom{n}{k} = C_n^k = \frac{n!}{k!(n-k)!}$$

Примеры

$$y = \cos 2x \quad \left. \begin{array}{l} f(g(x)) \\ \text{сложная функция} \end{array} \right\} g(x)$$

$$y' = f'_g \cdot g'_x ; \quad g'_x = (2x)' = 2$$

$$f'_g = -\sin 2x$$

$$y' = -2 \sin 2x$$

$$y = \sqrt{3 - \cos^3(\ln[x + \sqrt{x}])}$$

$$y' = \frac{1}{2\sqrt{3 - \cos^3(\ln[x + \sqrt{x}])}} \cdot (-\cos^3(\ln[x + \sqrt{x}]))'$$

$$= \frac{1}{2\sqrt{3 - \cos^3(\ln[x + \sqrt{x}])}} \cdot (-3\cos^2(\ln[x + \sqrt{x}])) \cdot (\cos(\ln[x + \sqrt{x}]))'$$

$$= \frac{+1}{2\sqrt{\dots}} \cdot 3\cos^2(\ln[x + \sqrt{x}]) \cdot \sin(\ln[x + \sqrt{x}]) \cdot (\ln[x + \sqrt{x}])'$$

$$= \frac{3\cos^2(\ln[x + \sqrt{x}]) \cdot \sin(\ln[x + \sqrt{x}]) \cdot (x + \sqrt{x})'}{2\sqrt{3 - \cos^3(\ln[x + \sqrt{x}])} \cdot (x + \sqrt{x})}$$

$$= \frac{3\cos^2(\ln[x + \sqrt{x}]) \cdot \sin(\ln[x + \sqrt{x}])}{2 \cdot \sqrt{3 - \cos^3(\ln[x + \sqrt{x}])} \cdot (x + \sqrt{x})} \cdot (1 + \frac{1}{2\sqrt{x}})$$

Пример 3. Иногда стоит функцию упростить перед взятием производной

Упростим выражение:

- степени, под логарифмом, можно вынести: $\ln a^b = b \ln a$
- логарифм разности на разность: $\ln(a/b) = \ln a - \ln b$

$$y = \ln \sqrt[5]{\frac{1-x}{1+x}}$$
$$y = \ln \sqrt[5]{\frac{1-x}{1+x}} = \frac{1}{5} \ln \left(\frac{1-x}{1+x} \right) =$$
$$= \frac{1}{5} [\ln(1-x) - \ln(1+x)] =$$
$$y' = \frac{1}{5} \left[\frac{-1}{1-x} - \frac{+1}{1+x} \right]$$

Пример 4. Вычислить приблизительное значение функции в точках, если известно её значение и значение её производной в некоторой, близкой к заданным, точке. Сравнить эти значения с точным значением функции.

Используем разложение функции в ряд Тейлора, ограничившись двумя членами ряда.

см. выше формулу для аппроксимацию функции в окрестности точки

$y = \sin(x)$
 $x_0 = \pi/4, \quad \sin \frac{\pi}{4} \approx 0.71$
 $x_1 = 0.9 \text{ рад.}, \quad \cos \frac{\pi}{4} \approx 0.71$
 $x_2 = 1 \text{ рад.}$

$$y \approx y\left(\frac{\pi}{4}\right) + y'\left(\frac{\pi}{4}\right) \cdot \left(x - \frac{\pi}{4}\right)$$
$$y \approx 0.71 + 0.71 \left(x - \frac{\pi}{4}\right)$$
$$y(0.9) \approx 0.71 + 0.71 \cdot 0.11 \approx 0.79$$
$$\sin(0.9) = 0.78$$
$$y(1) \approx 0.71 + 0.71 \cdot 0.21 \approx 0.86$$
$$\sin(1) = 0.84$$

Производная и экстремумы функции

Критической точкой дифференцируемой функции $f: \mathbb{R}^n \rightarrow \mathbb{R}$ называется точка, в которой её дифференциал обращается в нуль. Это условие эквивалентно тому, что в данной точке все частные производные первого порядка обращаются в нуль, геометрически оно означает, что касательная гиперплоскость к графику функции горизонтальна.

Экстремум — максимальное или минимальное значение функции на заданном множестве.

Точкой экстремума может быть только критическая точка. Но не все критические точки — точки экстремума. Пример — точка перегиба в кубической параболе.

Теорема Ферма: производная дифференцируемой функции в точке локального экстремума равна нулю.

Доказательство.

Предположим, что $f(x_0) = \max_{x \in (a,b)} f(x)$. Тогда $\forall x \in (a,b): f(x) \leq f(x_0)$.

Поэтому:

$$f'_-(x_0) = \lim_{x \rightarrow x_0-0} \left(\frac{f(x) - f(x_0)}{x - x_0} \right) \geq 0,$$

$$f'_+(x_0) = \lim_{x \rightarrow x_0+0} \left(\frac{f(x) - f(x_0)}{x - x_0} \right) \leq 0.$$

Что значит $x_0 \neq 0$ в примере?

Если производная $f'(x_0)$ определена, то получаем

$$0 \leq f'_-(x_0) = f'(x_0) = f'_+(x_0) \leq 0, \text{ то есть } f'(x_0) = 0$$

Если x_0 — точка локального минимума функции f то доказательство аналогично.

Достаточные условия существования локальных экстремумов

Пусть дана функция $f: M \subset \mathbb{R} \rightarrow \mathbb{R}$ и $x_0 \in M^0$ — внутренняя точка области определения f . Тогда:

- x_0 называется точкой локального максимума функции f , если существует проколота окрестность $\dot{U}(x_0)$ такая, что $\forall x \in \dot{U}(x_0) \rightarrow f(x) \leq f(x_0)$
- x_0 называется точкой локального минимума функции f , если существует проколота окрестность $\dot{U}(x_0)$ такая, что $\forall x \in \dot{U}(x_0) \rightarrow f(x) \geq f(x_0)$
- x_0 называется точкой глобального (абсолютного) максимума, если $\forall x \in M \rightarrow f(x) \leq f(x_0)$
- x_0 называется точкой глобального (абсолютного) минимума, если $\forall x \in M \rightarrow f(x) \geq f(x_0)$

Достаточное условие экстремума: если в точке x производная $f'(x)$ функции $f(x)$ меняет знак, то точка x является экстремумом.
Если производная становится

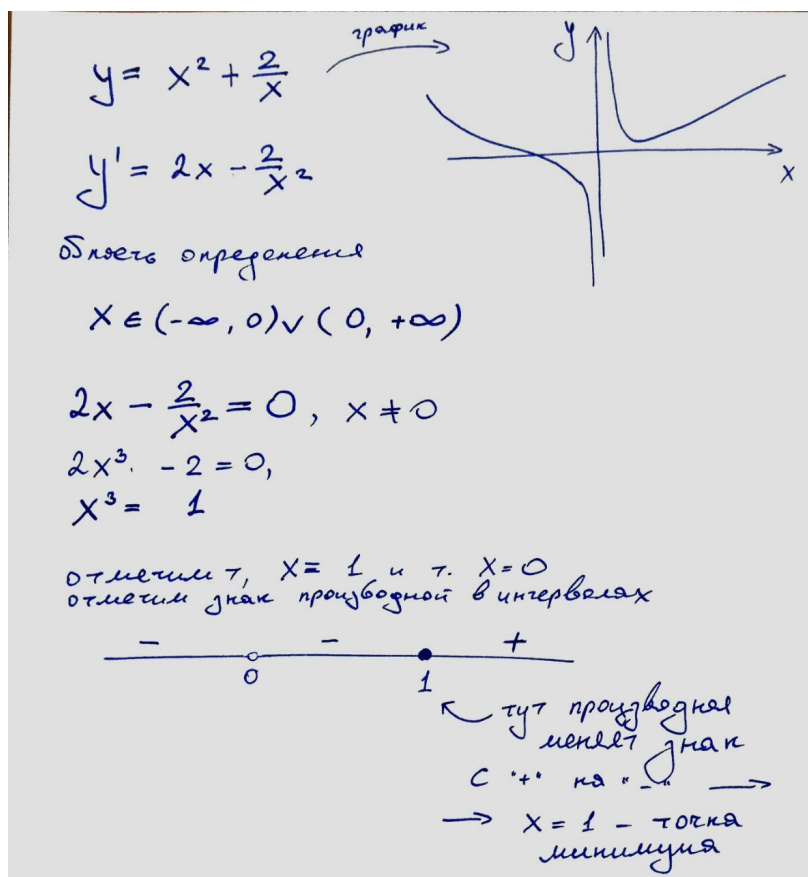
положительной, то это точка минимума, в противном случае точка максимума.

См. также

ru.wikipedia.org/wiki/Численное_дифференцирование

wolframalpha.com/examples/mathematics/calculus-and-analysis/derivatives – вычисление производных в WolframAlpha

Пример 5. Поиск экстремумов



Если функция задана уравнением $y = f(x)$, разрешённым относительно y , то функция задана в явном виде (явная функция).

Под неявным заданием функции понимают задание функции в виде уравнения $F(x; y) = 0$, не разрешённого относительно y .

Всякую явно заданную функцию $y = f(x)$ можно записать как неявно заданную уравнением $f(x) - y = 0$, но не наоборот.

Пример. Производная функции заданной неявно [!!!]

$3x^2y^2 - 5x + \sin y = 3y - 1$ ← неявно заданная функция
 Найдём производную.
 — будем брать производную и от x и от y
 — при взятии производной от y нужно помнить, что это производная сложной функции
 $3(x^2y^2)' - 5 + (\sin y)' = 3y'$
 производн. произведения, производн. сложной функ. производная сложной функции
 $6xy^2 + 6x^2yy' - 5 + \cos y \cdot y' = 3y'$
 производные от y влево:
 $6x^2yy' + \cos y \cdot y' = -6xy^2 + 5$
 $y'(6x^2y + \cos y) = 5 - 6xy^2$
 $y' = \frac{5 - 6xy^2}{6x^2y + \cos y}$

это не производная функции нескольких переменных

Частная производная (partial derivative)

Частная производная — одно из обобщений понятия производной на случай функции нескольких переменных.

Частная производная функции $f(x_1, x_2, \dots, x_n)$ в точке (a_1, a_2, \dots, a_n) определяется так:

$$\frac{\partial f}{\partial x_k}(a_1, \dots, a_n) = \lim_{\Delta x \rightarrow 0} \frac{f(a_1, \dots, a_k + \Delta x, \dots, a_n) - f(a_1, \dots, a_k, \dots, a_n)}{\Delta x}$$

Обозначение $\frac{\partial f}{\partial x} \neq \frac{df}{dx}$

Производная по направлению

Производная по направлению — одно из обобщений понятия производной на случай функции нескольких переменных. Производная по направлению показывает, как быстро значение функции изменяется при движении в данном направлении.

Рассмотрим дифференцируемую функцию $f(x_1, \dots, x_n)$ от n аргументов в окрестности точки $\vec{x}^0 = (x_1^0, \dots, x_n^0)$. Для любого единичного вектора $\vec{e} = (e_1, \dots, e_n)$ определим производную функции f в точке \vec{x}^0 по направлению \vec{e} :

$$\nabla_{\vec{e}} f(\mathbf{x}) = \lim_{h \rightarrow 0} \frac{f(\vec{x}^0 + h \cdot \vec{e}) - f(\vec{x}^0)}{h}$$

∇ — оператор набла

Значение этого выражения показывает, как быстро меняется значение функции при сдвиге аргумента в направлении вектора \vec{e} .

Если направление сонаправленно с координатной осью, то производная по этому направлению совпадает с частной производной по этой координате.

Производная по направлению и частные производные

$$\nabla_{\vec{e}} f(\mathbf{x}) = \frac{\partial f}{\partial x_1} e_1 + \dots + \frac{\partial f}{\partial x_n} e_n$$

Формулу можно записать короче, через скалярное произведение:

$$\nabla_{\vec{e}} f(\mathbf{x}) = \nabla f \cdot \vec{e}$$

$$\vec{x} \cdot \vec{y} = \sum_{i=1}^n x_i \cdot y_i \quad \leftarrow \begin{array}{l} \text{скалярное} \\ \text{произведение} \end{array}$$

Градиент (Gradient)

Для случая трёхмерного пространства градиентом скалярной функции $\varphi = \varphi(x, y, z)$ координат x, y, z называется векторная функция с компонентами

$$\frac{\partial \varphi}{\partial x}, \frac{\partial \varphi}{\partial y}, \frac{\partial \varphi}{\partial z}.$$

Или, используя для единичных векторов по осям прямоугольных декартовых координат $\vec{e}_x, \vec{e}_y, \vec{e}_z$:

$$\text{grad } \varphi = \nabla \varphi = \frac{\partial \varphi}{\partial x} \vec{e}_x + \frac{\partial \varphi}{\partial y} \vec{e}_y + \frac{\partial \varphi}{\partial z} \vec{e}_z$$

Если φ — функция n переменных x_1, \dots, x_n , то её градиентом называется n -мерный вектор

$$\left(\frac{\partial \varphi}{\partial x_1}, \dots, \frac{\partial \varphi}{\partial x_n} \right)$$

компоненты которого равны частным производным φ по всем её аргументам.

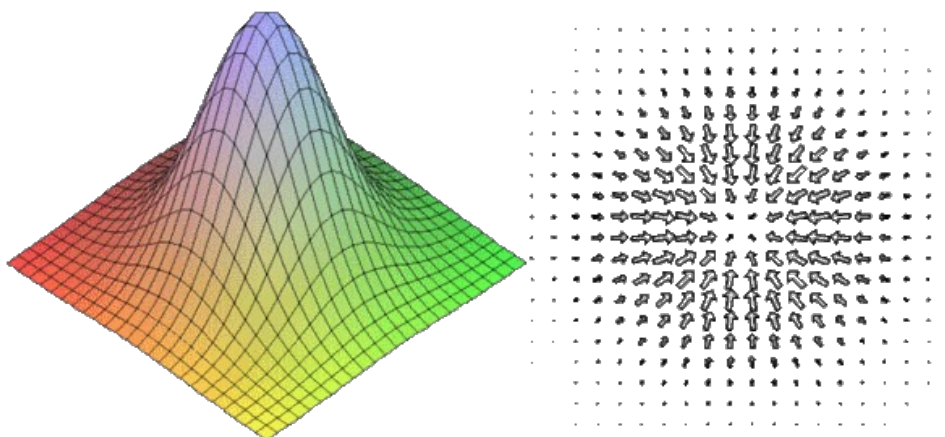


График функции (слева) и вектора её градиента (справа)

Свойства градиента

- $\text{grad } c = 0$, где c – константа
- Свойства линейности:
 - $\text{grad}(c \cdot \vec{u}) = c \cdot \text{grad}(\vec{u})$
 - $\text{grad}(\vec{u} + \vec{v}) = \text{grad}(\vec{u}) + \text{grad}(\vec{v})$
 - $\text{grad}(\vec{u} \cdot \vec{v}) = \vec{u} \cdot \text{grad}(\vec{v}) + \vec{v} \cdot \text{grad}(\vec{u})$

см. правила
дифференцирования выше

Матрица Якоби – матрица производных

Пусть задано отображение $\mathbf{u} : \mathbb{R}^n \rightarrow \mathbb{R}^m$, $\mathbf{u} = (u_1, \dots, u_m)$,
 $u_i = u_i(x_1, \dots, x_n)$, $i = 1, \dots, m$, имеющее в некоторой точке x все частные производные первого порядка. Матрица J , составленная из частных производных этих функций в точке x , называется матрицей Якоби данной системы функций.

$$J(x) = \begin{pmatrix} \frac{\partial u_1}{\partial x_1}(x) & \frac{\partial u_1}{\partial x_2}(x) & \cdots & \frac{\partial u_1}{\partial x_n}(x) \\ \frac{\partial u_2}{\partial x_1}(x) & \frac{\partial u_2}{\partial x_2}(x) & \cdots & \frac{\partial u_2}{\partial x_n}(x) \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial u_m}{\partial x_1}(x) & \frac{\partial u_m}{\partial x_2}(x) & \cdots & \frac{\partial u_m}{\partial x_n}(x) \end{pmatrix}$$

Иными словами, матрица Якоби является производной векторной функции от векторного аргумента.

Определитель этой матрицы называется якобианом.

Примеры

Пример 1. Найти градиент функции в точке

$$u = x + \ln(z^2 + y^2)$$
$$M_0(2, 1, 1)$$
$$\frac{\partial u}{\partial x} = 1; \quad \frac{\partial u}{\partial y} = \frac{2y}{z^2 + y^2}; \quad \frac{\partial u}{\partial z} = \frac{2z}{z^2 + y^2}$$
$$\left. \frac{\partial u}{\partial x} \right|_{M_0} = 1 \quad \left. \frac{\partial u}{\partial y} \right|_{M_0} = 1 \quad \left. \frac{\partial u}{\partial z} \right|_{M_0} = 1$$
$$\text{grad } u = (1, 1, 1)$$

$$z = e^{x^2 y - x}$$
$$M_0(2, 0),$$

Найти производн. в т. M_0 в направлении, составляющем угол $\frac{\pi}{6}$ с положительным направлением оси Ox

$$\frac{\partial z}{\partial x} = (2xy - 1)e^{x^2 y - x}; \quad \left. \frac{\partial z}{\partial x} \right|_{M_0} = -e^{-2}$$
$$\frac{\partial z}{\partial y} = x^2 \cdot e^{x^2 y - x}; \quad \left. \frac{\partial z}{\partial y} \right|_{M_0} = 4e^{-2}$$
$$\ell = (\cos \frac{\pi}{6}, \sin \frac{\pi}{6}) \leftarrow \text{вектор направления}$$
$$\nabla_{\ell} z(M_0) = (-e^{-2} \cdot \cos \frac{\pi}{6}, 4e^{-2} \cdot \sin \frac{\pi}{6}) \approx (-0.12, 0.27)$$

Пример 2. Найти производную функции в точке по направлению

вектор направления должен быть единичным. Т.е. его нужно нормировать при необходимости.

Пример 3

Найти минимум функции двух переменных.

Описать итерации метода градиентного спуска до тех пор, пока изменение значения минимизируемой функции по модулю будет изменяться больше чем на ε

Точку минимума определить из множества критических, для которой выполняется условие

- а) если $AC - B^2 > 0$ и $A < 0$, то в точке M имеется максимум;
- б) если $AC - B^2 > 0$ и $A > 0$, то в точке M имеется минимум;
- в) если $AC - B^2 < 0$, то экстремума нет;

г) если $AC - B^2 = 0$, то вопрос о наличии экстремума остается открытым;

где $A = \left(\frac{\partial^2 f}{\partial x^2} \right)_M$, $B = \left(\frac{\partial^2 f}{\partial x \partial y} \right)_M$, $C = \left(\frac{\partial^2 f}{\partial y^2} \right)_M$, M – исследуемый экстремум.

$$f(x, y) = \frac{x^2}{10} + y^2$$

Найдём минимум.

Определим критические точки
из решения системы уравнений

$$\begin{cases} \frac{\partial f}{\partial x} = 0; \\ \frac{\partial f}{\partial y} = 0; \end{cases} \begin{cases} x/5 = 0; \\ 2y = 0; \end{cases} \begin{cases} x = 0 \\ y = 0 \end{cases}$$

критическая т. $X_1 = (0, 0)$

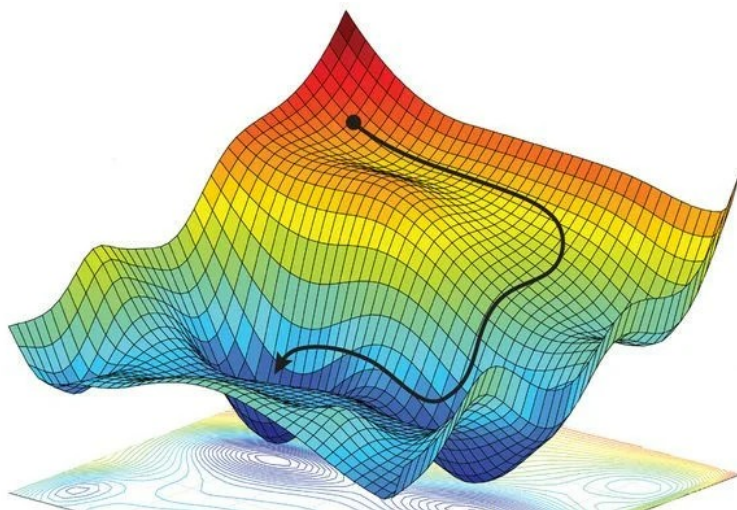
$$A = \left(\frac{\partial^2 f}{\partial x^2} \right)_{x_1}, \quad B = \left(\frac{\partial^2 f}{\partial x \partial y} \right)_{x_1}, \quad C = \left(\frac{\partial^2 f}{\partial y^2} \right)_{x_1}$$

$$A = 0.2 \quad B = 0 \quad C = 2$$

$$\left. \begin{array}{l} AC - B^2 = 0.4 > 0 \\ A > 0 \end{array} \right\} \Rightarrow \text{в т. } X_1 = (0, 0) - \text{минимум функции } f$$

Ответ можно
сравнить с примером
численного
определения
минимума

Оптимизация (Mathematical optimization or mathematical programming)



Оптимизация — задача нахождения экстремума (минимума или максимума) целевой функции в некоторой области конечномерного векторного пространства, ограниченной набором линейных и/или нелинейных равенств и/или неравенств.

Постановка задачи

Среди элементов x , образующих множества X , найти такой элемент x^* , который доставляет минимальное (максимальное) значение $f(x^*)$ заданной функции $f(x)$.

Определим:

- Допустимое множество $X = \{\vec{x} \mid g_i(\vec{x}) \leq 0, i = 1, \dots, m\} \subset \mathbb{R}^n$
- Целевую функцию $f : X \rightarrow \mathbb{R}$
- Критерий поиска (max или min).

Тогда решить задачу $f(x) \rightarrow \min_{\vec{x} \in X}$ означает одно из:

- Показать, что $X = \emptyset$
- Показать, что целевая функция $f(\vec{x})$ не ограничена снизу.
- Найти $\vec{x}^* \in X : f(\vec{x}^*) = \min_{\vec{x} \in X} f(\vec{x})$
- Если $\nexists \vec{x}^*$, то найти $\inf_{\vec{x} \in X} f(\vec{x})$.

Если целевая функция не является выпуклой, то часто ограничиваются поиском локальных минимумов и максимумов: точек x_0 таких, что всюду в некоторой их окрестности выполняется $f(x) \geq f(x_0)$ для минимума и $f(x) \leq f(x_0)$ для максимума.

функцию $y = f(x)$ называют ограниченной снизу на мн-ве X , если существует такое число b , что для любого x из мн-ва X выполнено неравенство $f(x) > b$

Инфимум (inf) подмножества X упорядоченного множества (или класса) M , называется наибольший элемент M , который равен или меньше всех элементов множества X . Привести пример инфинума для подмножества натуральных чисел

Некоторая классификация методов оптимизации

Если допустимое множество $\mathbb{X} = \mathbb{R}^n$, то такая задача называется задачей *безусловной оптимизации*, в противном случае — задачей *условной оптимизации*. Задачу условной оптимизации линейной функции решает, например, симплекс-метод.

Если оптимизация связана с расчётом оптимальных значений параметров при заданной структуре объекта, то она называется *параметрической оптимизацией*. Задача выбора оптимальной структуры является *структурной оптимизацией*.

Классификация по детерминированности.

- Детерминированные;
- случайные – стохастические (например стохастический градиентный спуск);
- комбинированные.

Классификация по требованиям к гладкости и наличию у целевой функции частных производных:

- прямые методы, требующие только вычислений целевой функции в точках приближений (например метод деформируемого многогранника);
- методы первого порядка: требуют вычисления первых частных производных функции (например градиентный спуск)
- методы второго порядка: требуют вычисления вторых частных производных, то есть гессиана целевой функции (например Бroyдена — Флетчера — Гольдфарба — Шанно (BFGS))

Сходимость (convergence) – это стремление значений решения метода к соответствующим значениям решения исходной задачи при стремлении к нулю параметра дискретизации (например, шага интегрирования).

С ростом порядка метода скорость сходимости возрастает, но и возрастает вычислительная сложность каждой итерации.

Особый интерес могут для оптимизации могут представлять квадратичный функции вида

$$f(x) = f(x_1, x_2, \dots, x_n) = \frac{1}{2} \sum_{i,j=1}^n a_{ij} x_i x_j - \sum_{i=1}^n b_i x_i + c$$

Оптимизация таких функций – одна из самых простых задач оптимизации. Если используются методы первого порядка, то для квадратичной функции получается линейный градиент

Вторая причина – эти функции часто встречаются на практике.
Например при построении линейной регрессии.

Градиентный спуск (Gradient descent)

Градиентный спуск — метод первого порядка для нахождения локального экстремума (минимума или максимума) функции с помощью движения вдоль градиента.

Пусть целевая функция имеет вид:

$$F(\vec{x}) : \mathbb{X} \rightarrow \mathbb{R}$$

И задача оптимизации задана следующим образом:

$$F(\vec{x}) \rightarrow \min_{\vec{x} \in \mathbb{X}}$$

В случае, когда требуется найти максимум, вместо $F(\vec{x})$ используется $-F(\vec{x})$

Основная идея метода заключается в том, чтобы идти в направлении наискорейшего спуска, а это направление задаётся антиградиентом $-\nabla F$

$$\vec{x}^{[j+1]} = \vec{x}^{[j]} - \lambda^{[j]} \nabla F(\vec{x}^{[j]})$$

где $\lambda^{[j]}$ задает скорость градиентного спуска. Верхний индекс — номер шага.

λ может быть выбрана:

- постоянной;
- убывающей в процессе градиентного спуска;
- гарантирующей наискорейший спуск:

1. Для поиска минимума $F(\vec{x})$ получаем

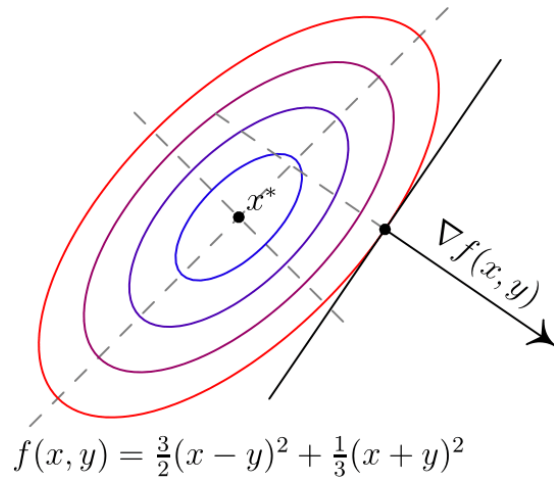
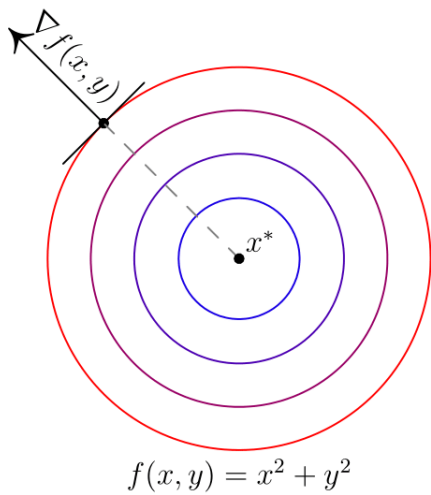
$$\lambda^{[j]} = \operatorname{argmin}_{\lambda} F(\vec{x}^{[j+1]}) = \operatorname{argmin}_{\lambda} F(\vec{x}^{[j]} - \lambda \nabla F(\vec{x}^{[j]}))$$

2. Для поиска максимума $F(\vec{x})$ получаем

$$\lambda^{[j]} = \operatorname{argmax}_{\lambda} F(\vec{x}^{[j+1]}) = \operatorname{argmax}_{\lambda} F(\vec{x}^{[j]} + \lambda \nabla F(\vec{x}^{[j]}))$$

Здесь неизвестные обозначаются как x . Однако на практике, например при подборе параметров (коэффициентов) уравнения регрессии, x — это независимые признаки (которые считаются константами), а неизвестные которые нужно подобрать — w

см. картинку
ниже



[!!!] Вектор градиента указывает на направление наискорейшего возрастания функции. Но вектор антиградиента в общем случае не указывает точно на точку минимума функции

Алгоритм

1. Задают начальное приближение и точность расчёта \vec{x}^0, ε

2. Рассчитывают $\vec{x}^{[j+1]} = \vec{x}^{[j]} - \lambda^{[j]} \nabla F(\vec{x}^{[j]})$,

где $\lambda^{[j]} = \operatorname{argmin}_{\lambda} F(\vec{x}^{[j]} - \lambda \nabla F(\vec{x}^{[j]}))$

или $\lambda^{[j]} = \lambda = \text{const}$

3. Проверяют условие остановки:

1. Если $|\vec{x}^{[j+1]} - \vec{x}^{[j]}| > \varepsilon$, $|F(\vec{x}^{[j+1]}) - F(\vec{x}^{[j]})| > \varepsilon$ или $\|\nabla F(\vec{x}^{[j+1]})\| > \varepsilon$ (выбирают одно из условий), то $j = j + 1$ и переход к шагу 2.

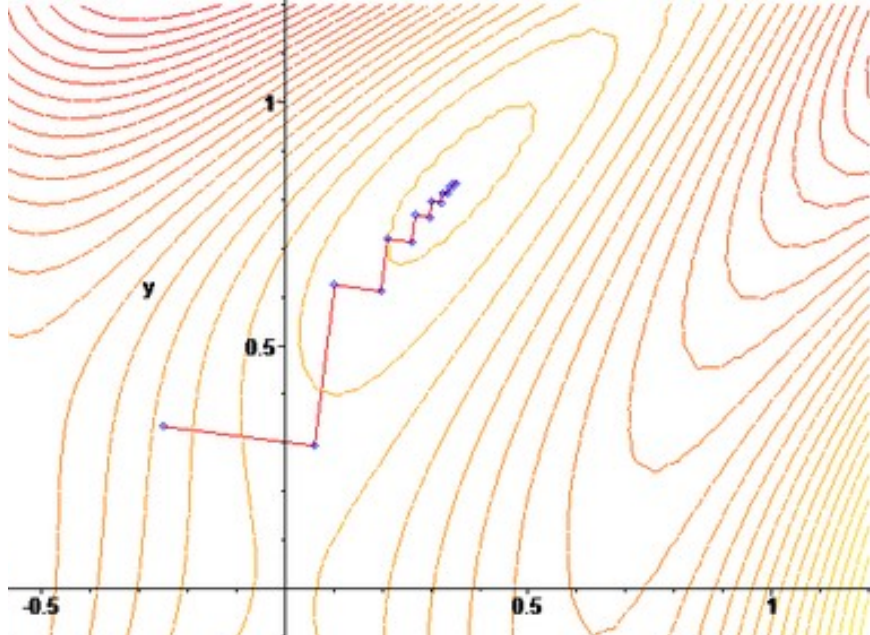
2. Иначе $\vec{x} = \vec{x}^{[j+1]}$ и останов.

argmin по λ приведено для общности.

Если выбирается оптимальная длина шага, то такой метод называется методом наискорейшего спуска.

Оптимальная длина шага в этом случае выбирается методом одномерной оптимизации

знаком $\| \cdot \|$ обозначается норма



Пример нескольких шагов метода. Причём здесь шаги метода не направлены точно в точку минимума, а приближаются к ней зигзагами.

Метафора метода: поиск спуска с горы в сильный туман, когда путь к подножью не видно.

Теорема о сходимости.

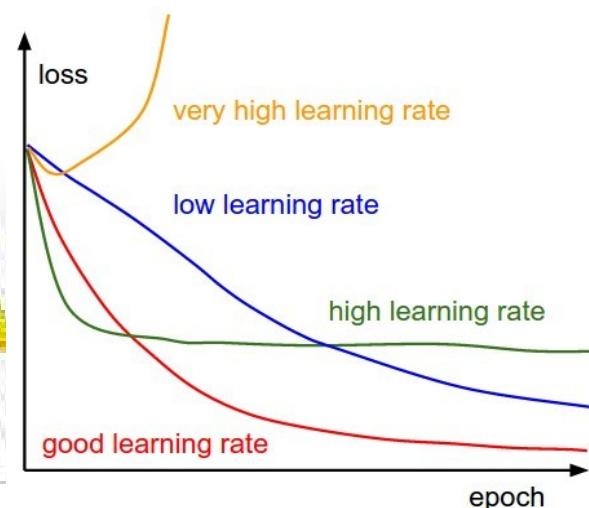
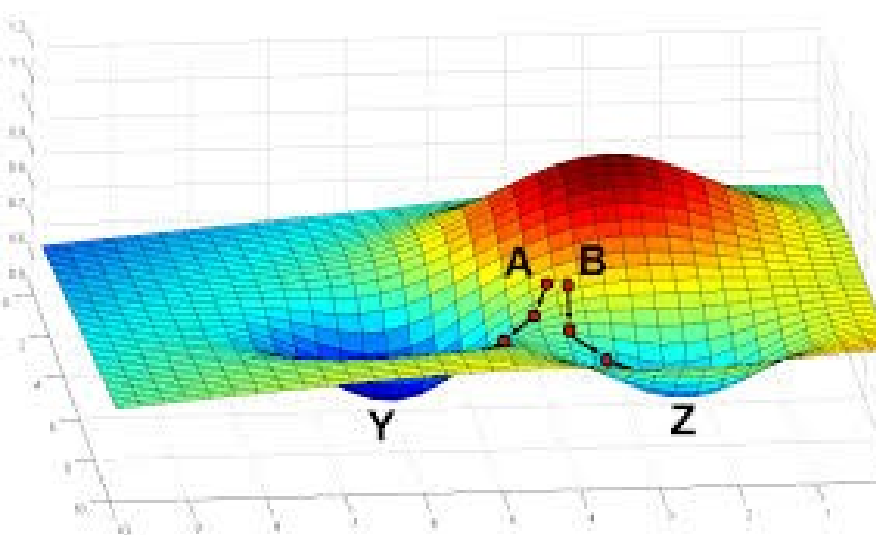
Если $\lambda = \text{const}$, $0 < \lambda < 2/L$, $f(x)$ – гладкая функция и ограничена снизу, ее градиент $f'(x)$ удовлетворяет условию Липшица

$\|f'(x) - f'(y)\| = L\|x - y\|$ для любых x и y , тогда $\lim_{k \rightarrow \infty} f'(x^{[k]}) = 0$,
 $f(x^{[k+1]}) < f(x^{[k]})$ при любом выборе начального приближения x_0 .

k – номер шага
 L – некоторое число

Проблемы метода

- Функция должна иметь непрерывную первую производную
- Проблема медленной сходимости – проблема оврага (рисунок выше) – метод медленно спускается к минимуму прыгая по станкам оврага.
- Может зайти в локальный минимум (общая проблема для многих методов оптимизации)
- Нужно подбирать параметр λ : при неудачном выборе метод может сходиться либо слишком медленно, либо пропускать минимум.



Найденный минимум может сильно зависеть от начальной точки

Выбор неправильного значения для длины шага (learning rate) может сильно сказаться на минимизации функции (на примере функции ошибки – loss)

Пример. Две итерации метода градиентного спуска с постоянным шагом.

$$f(x, y) = \frac{x^2}{10} + y^2; \quad X_0 = (1, 1) \quad \lambda = 0.1 \quad \varepsilon = 0.3$$

$$\text{grad } f = \left(\frac{x}{5}, 2y \right)$$

итерация 1. - - - - -

$$X_1 = X_0 - \text{grad } f_{X_0} \cdot \lambda = (1, 1) - (0.2, 2) \cdot 0.1$$

$$X_1 = (0.98, 0.8)$$

$$|X_1 - X_0| \approx 0.2$$

$$|f(X_1) - f(X_0)| \approx 0.37 > \varepsilon. \text{ продолжаем.}$$

итерация 2. - - - - -

$$X_2 = X_1 - \text{grad } f_{X_1} \cdot \lambda = (0.98, 0.8) - (0.196, 1.6) \cdot 0.1$$

$$X_2 = (0.96, 0.64)$$

$$|X_2 - X_1| \approx 0.026$$

$$|f(X_2) - f(X_1)| \approx 0.23 < \varepsilon. \text{ останов.}$$

Найденный минимум

$$X = (0.96, 0.64)$$

см. также ru.wikipedia.org/wiki/Тестовые_функции_для_оптимизации

Вопросы

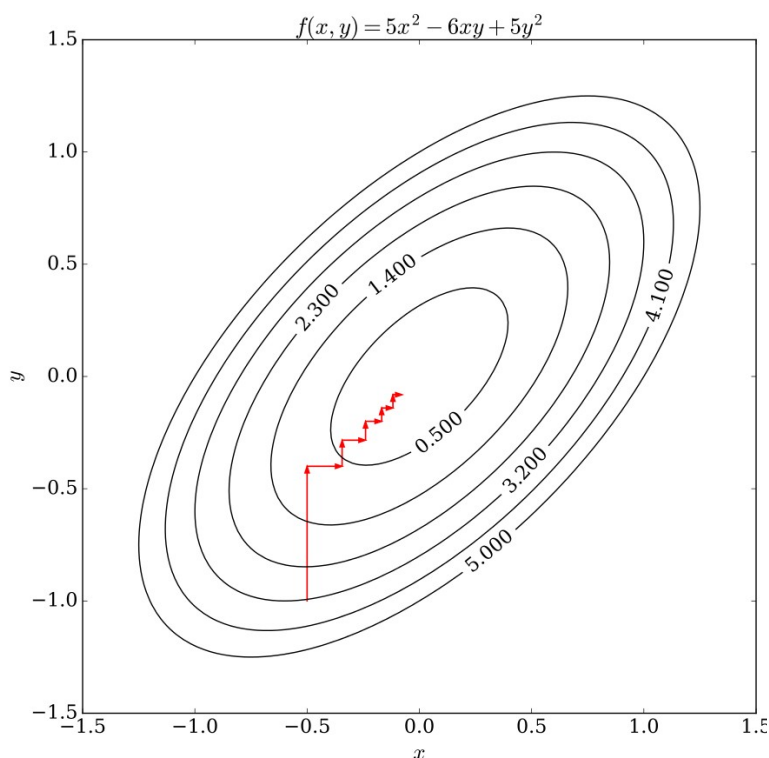
1. Зачем нужен метод градиентного спуска (ГС)?
2. ГС — это метод какого порядка?
3. Почему на практике чаще всего используют именно ГС и его модификации для минимизации функции потерь, а не методы других порядков?
4. В чём идея метода градиентного спуска? Как выглядит шаг метода?
5. Какие условия останова можно использовать?
6. Какие параметры есть у метода?
7. Опишите проблемы выбора длины шага.
8. Может ли метод найти вместо глобального минимума локальный?

Метод покоординатного спуска (Coordinate descent)

Алгоритм

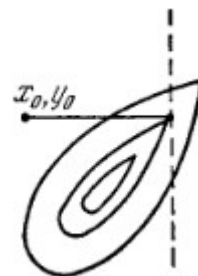
1. Задают начальное приближение и точность расчёта \vec{x}^0, ε
2. Для всех компонент вектора $i=1..n$:
 1. Зафиксировать все компоненты вектора x кроме i -ой
 2. Решить задачу одномерной оптимизации¹ по этой компоненте, получить новое значение i -й координаты
 3. Проверить условие остановки:
4. Если $|\vec{x}^{[j+1]} - \vec{x}^{[j]}| > \varepsilon$, $|F(\vec{x}^{[j+1]}) - F(\vec{x}^{[j]})| > \varepsilon$ или $\|\nabla F(\vec{x}^{[j+1]})\| > \varepsilon$ (выбирают одно из условий), то $j = j + 1$ и переход к шагу 2 основного алгоритма.

Иначе $\vec{x} = \vec{x}^{[j+1]}$ и останов.



Шаги производятся только вдоль координат

Существуют функции, когда метод координатного спуска не приводит даже в локальный оптимум – рисунок справа.



Эффективность метода (количество итераций) зависит от ориентации линий уровня относительно осей координат. В идеальном случае овраг ориентирован вдоль одной из осей координат.

1 Можно решать задачу например методом золотого сечения или методом Ньютона для одной переменной

Метод сопряжённых градиентов (Nonlinear conjugate gradient method)

Некоторые определения

Пусть заданы $\vec{S}_1, \dots, \vec{S}_n \in \mathbb{X} \subset \mathbb{R}^n$.

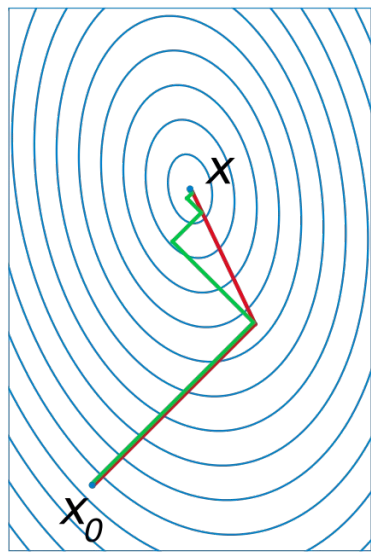
Введём на X целевую функцию $f(\vec{x}) \in C^2(\mathbb{X})$

Векторы $\vec{S}_1, \dots, \vec{S}_n$ называются сопряжёнными, если:

- $\vec{S}_i^T H \vec{S}_j = 0, \quad i \neq j, \quad i, j = 1, \dots, n$
- $\vec{S}_i^T H \vec{S}_i \geq 0, \quad i = 1, \dots, n$

где H — матрица Гессе $f(\vec{x})$ — матрица состоящая из вторых производных функции:

$$H(f) = \begin{bmatrix} \frac{\partial^2 f}{\partial x_1^2} & \frac{\partial^2 f}{\partial x_1 \partial x_2} & \dots & \frac{\partial^2 f}{\partial x_1 \partial x_n} \\ \frac{\partial^2 f}{\partial x_2 \partial x_1} & \frac{\partial^2 f}{\partial x_2^2} & \dots & \frac{\partial^2 f}{\partial x_2 \partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 f}{\partial x_n \partial x_1} & \frac{\partial^2 f}{\partial x_n \partial x_2} & \dots & \frac{\partial^2 f}{\partial x_n^2} \end{bmatrix}$$



метод
наискорейшего спуска – зелёная
ломаная; метода сопряжённых
градиентов – красная ломаная

- метод сходится быстрее чем обычный метод градиентного спуска
- при этом (как и в вышеприведённых модификациях) не требуется вычислять вторую производную.
- В случае квадратичной функции в R^n минимум находится не более чем за n шагов.

Стохастический градиентный спуск (Stochastic gradient descent, SGD)

Стохастический градиентный спуск (Stochastic gradient descent, SGD) — это метод итерации для оптимизации целевой функции с подходящими свойствами гладкости (например, дифференцируемость или субдифференцируемость). Его можно расценивать как стохастическую аппроксимацию оптимизации методом градиентного спуска, поскольку он заменяет реальный градиент (вычисленный из полного набора данных) путём оценки такового (вычисленного из случайно выбранного подмножества данных). Особенно в приложениях обработки больших данных это сокращает вычислительные ресурсы, достигая более быстрых итераций в обмен на более низкую скорость сходимости.

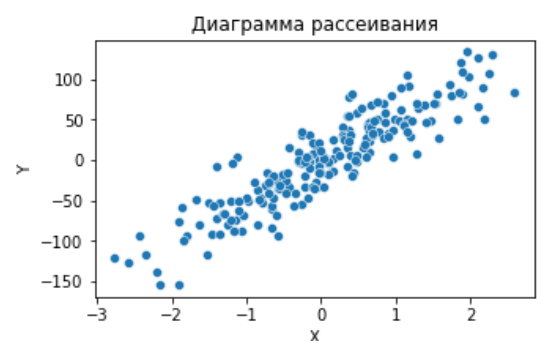
Математическая статистика и машинное обучение рассматривают задачу минимизации целевой функции, представленной суммой функций Q_i

$$Q(x) = \frac{1}{n} \sum_{i=1}^n Q_i(x)$$

Каждый член суммы Q_i обычно ассоциируется i -ым наблюдением в наборе данных (использованном для обучения).

В случаях, когда вычисление градиента вышеприведённой суммы может потребовать ресурсоёмких вычислений градиентов для всех суммируемых функций стохастический градиентный спуск отбирает (случайно, поэтому метод и стохастический) подмножество суммируемых функций на каждом шаге алгоритма. Например на каждом шаге для суммы из $n=100$ слагаемых будет рассмотрено только 20. Такое множество называется **батчем (batch)**.

Таким образом, алгоритм метода практически не изменяется, за исключением того, что оптимизируемая сумма на каждой итерации метода будет не полной.



Например Q_i — ошибка на i -й точке данных в задаче построения линейной регрессии

Под ресурсоёмкостью можно понимать высокие требования к оперативной памяти (все данных не помещаются в ОЗУ) или вычисление ошибки для всех данных занимает слишком много времени.

Алгоритм метода изменится следующим образом

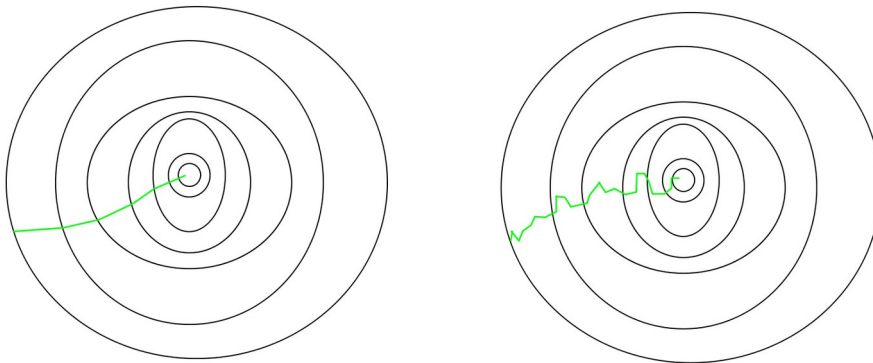
по сравнению с алгоритмом
градиентного спуска, здесь
добавился новый пункт



1. Задают начальное приближение и точность расчёта \vec{x}^0, ε
2. Цикл по выборкам частичных сумм из n . Далее для каждой частичной суммы:
 1. Рассчитывают $\vec{x}^{[j+1]} = \vec{x}^{[j]} - \lambda^{[j]} \nabla F(\vec{x}^{[j]})$,
3. Проверяют условие остановки:
 1. Если $|\vec{x}^{[j+1]} - \vec{x}^{[j]}| > \varepsilon$, $|F(\vec{x}^{[j+1]}) - F(\vec{x}^{[j]})| > \varepsilon$ или $\|\nabla F(\vec{x}^{[j+1]})\| > \varepsilon$ (выбирают одно из условий), то $j = j + 1$ и переход к шагу 2.
 2. Иначе $\vec{x} = \vec{x}^{[j+1]}$ и останов.

Стоит отметить, что эта модификация метода призвана решить исключительно техническую проблему – высокую ресурсоёмкость вычисления целевой функции как суммы других функций.

При этом приходится идти на компромисс – вычисленный градиент неполной суммы может плохо совпадать с градиентом целевой функции в случае полной суммы.

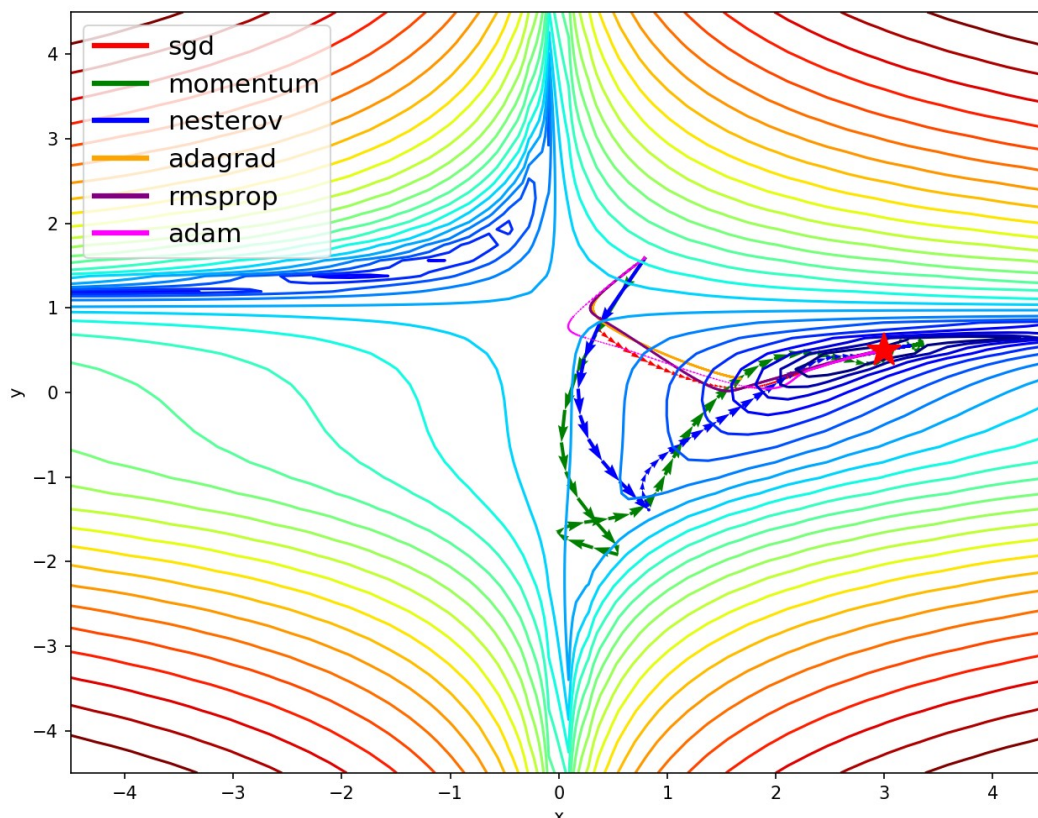


Пример траекторий градиентного спуска (слева) и стохастического градиентного спуска (справа)

Вопросы

1. Какую проблему применения градиентного спуска решает метод стохастического градиентного спуска?
2. Какие проблемы могут возникать в следствии этого?

Улучшения SGD



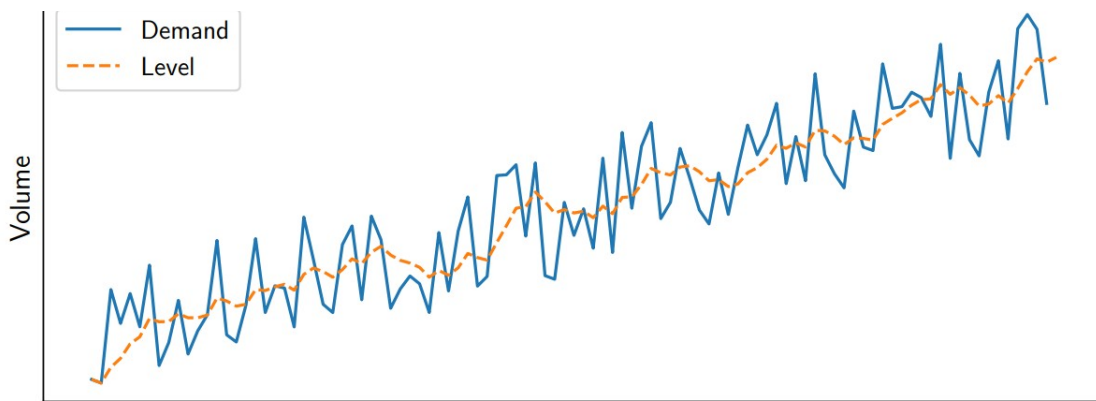
В глубоких нейросетях частные производные функции потерь по параметрам состоят из большого числа произведений (см. chain rule). Такие производные могут быть очень близкими к нулю, особенно на пологих участках функции и в седловых точках. Минимизация может быть очень медленной. Минимизация может остановиться в точке локального минимума.

Экспоненциальное сглаживание

Экспоненциальное сглаживание — метод математического преобразования, используемый при прогнозировании временных рядов. Метод также известен как метод простого экспоненциального сглаживания, или метод Брауна.

$$\begin{aligned} s_0 &= x_0 \\ s_t &= \alpha x_t + (1 - \alpha)s_{t-1}, \quad t > 0 \end{aligned} \quad \begin{aligned} s_t & \text{ — сглаженный ряд,} \\ c_t & \text{ — исходный ряд,} \\ \alpha & \text{ — коэффициент сглаживания,} \\ 0 < \alpha < 1 \end{aligned}$$

Сглаживание называется экспоненциальным потому, что каждое следующее сглаженное значение зависит от сглаживающего коэффициента на предыдущих шагах: $\alpha, \alpha^2, \alpha^3, \alpha^4 \dots$ [[wikipedia](#)]

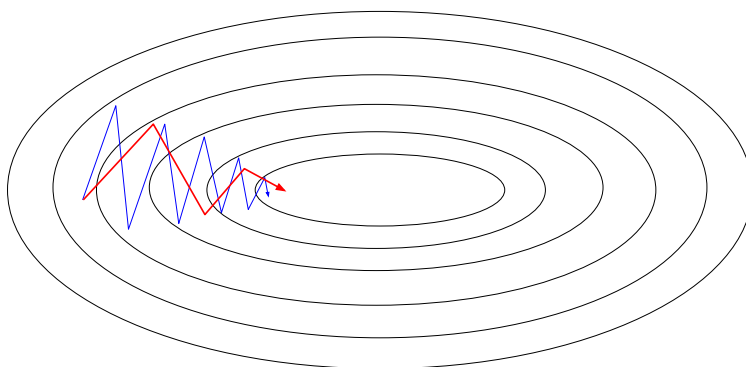


Экспоненциальное сглаживание применяют в том числе для лучшей визуализации кривых с большим уровнем шума. Например на графиках обучения в сервисе wandb.ai

Инерционные или ускоренные градиентные методы (momentum)

Далее в этом методе и последующих будем рассматривать алгоритм градиентного спуска, указывая только изменения в его отдельных частях.

Проблема SGD: направление градиента может часто меняться на отдельных шагах, при этом направление в среднем меняется мало, но и продвижение медленное.



Красная ломанная — инерционный метод градиентного спуска, синяя ломанная — классический метод градиентного спуска

github.com/ilguyi/optimizers.numpy

Размер стрелки — длина одного шага.

Предлагается учитывать направление одного или нескольких предыдущих шагов:

$$p_{t+1} = \beta \nabla f(x_t) + (1 - \beta) p_t$$

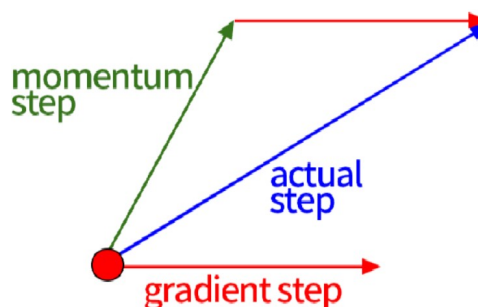
$$x_{t+1} = x_t - \lambda p_{t+1}$$

нижний индекс обозначает номер шага метода, x и p — векторы.

researchgate.net/publication/333469047_The_Frontier_of_SGD_and_Its_Variants_in_Machine_Learning

Информация о предыдущих шагах метода содержится в p , коэффициент β (выбирается несколько меньше единицы) отвечает за

← см. пункт 2 метода градиентного спуска. Здесь считаем λ постоянным. Иногда коэф. Длины шага λ ставят перед p



При $\beta = 0$ значения на предыдущем шаге не учитываются — обычный градиентный спуск.

Для $0 < \beta < 1$ значения градиента с предыдущего шага всегда меньше, чем с текущего.

вес каждого предыдущего сделанного шага в выборе направления для текущего. Такой подход, когда каждое следующее значение рекуррентно учитывает несколько предыдущих называется **экспоненциальным сглаживанием**. Можно говорить, что ρ определяет инерцию движения (momentum), т. к. шаг изменяется с учётом предыдущих значений градиента.

В задачах обучения нейросетей, где метод минимизирует функцию ошибки, рекомендуется выбирать значение β в диапазоне от 0.8 до 0.999.

Таким образом удаётся сгладить направление каждого из шагов, что решает проблему частой смены направления шага (в том числе из-за небольшого батча) и как следствие решается проблема общей низкой скорости сходимости.

В дополнение к этому, удаётся перешагивать некоторые локальные минимумы, за счёт инерции шага.

Один из первых таких методов появился в середине 20 века и назывался **метод тяжелого шарика**, что передавало природу инерционности метода: в этом методе не зависят от и аккуратно подбираются в зависимости от целевой функции.



Инерция помогает преодолевать локальные минимумы, седловые точки и быстрее проходить плато.

источник изображения

Визуализация пути: distill.pub/2017/momentum демонстрация (и описание) работы метода в зависимости от параметров: длины шага и коэффициента β

Пример, инерционный градиентный спуск

$$f(x, y) = \frac{x^2}{10} + y^2 \quad X_0 = (1, 1) \quad \lambda = 0.1$$

$$\text{grad } f = \left(\frac{x}{5}, 2y \right) \quad \beta = 0.8$$

$$P_1 = 0 + \text{grad } f_{X_0} \quad \varepsilon = 0.3$$

$$X_1 = X_0 - \text{grad } f_{X_0} \cdot \lambda = (1, 1) - (0.2, 2) \cdot 0.1$$

$$X_1 = (0.98, 0.8)$$

$$|X_1 - X_0| \approx 0.2$$

$$|f(X_1) - f(X_0)| \approx 0.37$$

$$P_2 = \beta \cdot P_1 + \text{grad } f_{X_1}$$

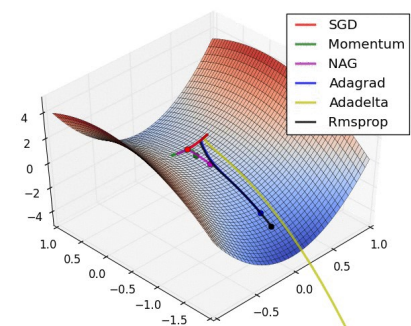
$$P_2 = 0.8 \cdot (0.2, 2) + (0.196, 1.6) = (0.36, 3.2)$$

$$X_2 = X_1 - \lambda \cdot P_2$$

$$X_2 = (0.98, 0.8) - 0.1 \cdot (0.36, 3.2) = (0.94, 0.48)$$

$$|X_2 - X_1| \approx 0.32$$

$$|f(X_2) - f(X_1)| \approx 0.42$$

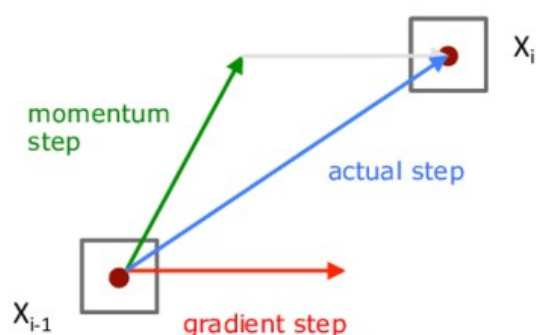


<https://www.linkedin.com/pulse/deep-learning-explaining-optimization-gradient-adam-sobh-phd>

Метод Нестерова (Nesterov Momentum)

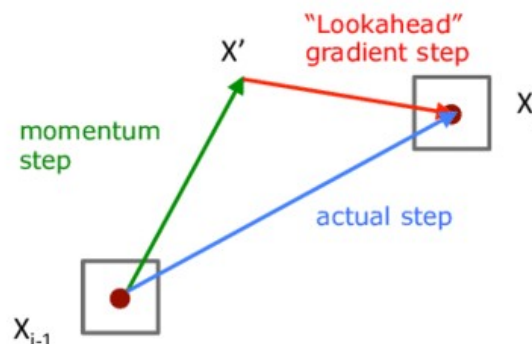
Нестеров Юрий Евгеньевич

Если представить, что происходит оптимизация функции двух переменных, то работу предыдущего метода можно представить левой частью рисунка ниже. Шаг, изменяющий параметры функции (синий вектор на рис. ниже), складывался из двух векторов p – вектор накопленных градиентов (на рисунке ниже это зелёный вектор) и градиента функции с некоторым множителем (на рисунке ниже это



Regular Momentum Update

красный вектор).



Nesterov's Momentum Update



Причём градиент вычисляется для той же точки (x_{i-1}) , откуда предполагалось сделать шаг.

В методе Нестерова предложено вычислять градиент функции не в точке x_{i-1} , а начала изменить координаты по направлению накопленного градиента (правая часть рисунка выше), и вычислить градиент уже в новой точке x' .

Таким образом обновление координат в методе градиентного спуска будет выглядеть так

$$p_{t+1} = \beta \nabla f(x_t + \beta p_t) + (1 - \beta) p_t$$
$$x_{t+1} = x_t - \lambda p_{t+1}$$

← тут, по сравнению с предыдущим случаем, поменялась только точка, в которой вычисляется градиент

Такая «корректировка» вектора p сокращает количество итераций, для отыскания минимума по сравнению с инерционным методом.

Вопросы

1. В чём идея инерционного градиентного спуска? Как выглядит шаг этого метода?
2. Какую проблему решает этот метод?
3. В чём идея инерции Нестерова? Как выглядит шаг этого метода?

Адаптивный градиент (adagrad)

Если изменения градиента большие, то стоит сократить длину шага. Если же градиент от шага к шагу меняется мало, то стоит увеличить длину шага. Таким образом градиент будет нормирован на величину, зависящую от разброса значений на последних шагах:

$$\begin{aligned} \text{cache}_{t+1} &= \text{cache}_t + (\nabla f(x_t))^2 \\ x_{t+1} &= x_t - \lambda \frac{\nabla f(x_t)}{\sqrt{\text{cache}_{t+1} + \epsilon}} \end{aligned}$$

ϵ — небольшое значение (например 10^{-7}),
нужное чтобы избежать нулевого
числителя;
 $0 < \beta < 1$

Выражения вверху векторные, каждая компонента градиента оптимизируемой функции будет нормирована своим значением суммы. Можно говорить, что для каждого параметра будет свой коэффициент длины шага (learnig rate).

RMSProp — root mean square propagation

Однако знаменатель в формуле выше может неограниченно расти, а значит шаг будет уменьшаться. Проблема решается экспоненциальным сглаживанием величины cache_t , когда благодаря понижающему коэффициенту будут учитываться значения градиента преимущественно с последних шагов:

$$\text{cache}_{t+1} = \gamma (\nabla f(x_t))^2 + (1 - \gamma) \text{cache}_t \quad 0 < \gamma < 1, \text{ обычно } \gamma = 0,99$$

Adam

Объединим рассмотренные выше идеи:

1. Экспоненциальное сглаживание для градиента (momentum или ГД с инерцией)
2. Вычисление градиента в точке, определяемой инерцией (Nesterov Momentum)
3. Нормировка градиента, для адаптивного регулирования шага по разным его компонентам (adagrad + RMSProb)

$$p_{t+1} = \beta \nabla f(x_t + \beta p_t) + (1 - \beta) p_t$$

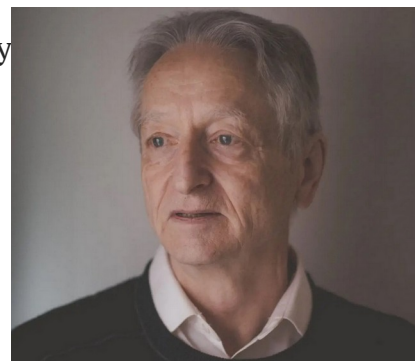
$$\text{cache}_{t+1} = \gamma (\nabla f(x_t))^2 + (1 - \gamma) \text{cache}_t$$

$$x_{t+1} = x_t - \lambda \frac{p_{t+1}}{\sqrt{\text{cache}_{t+1} + \epsilon}}$$

Для ускорения поиска минимума на первых этапах используется специальная процедура разогрева (warm up).

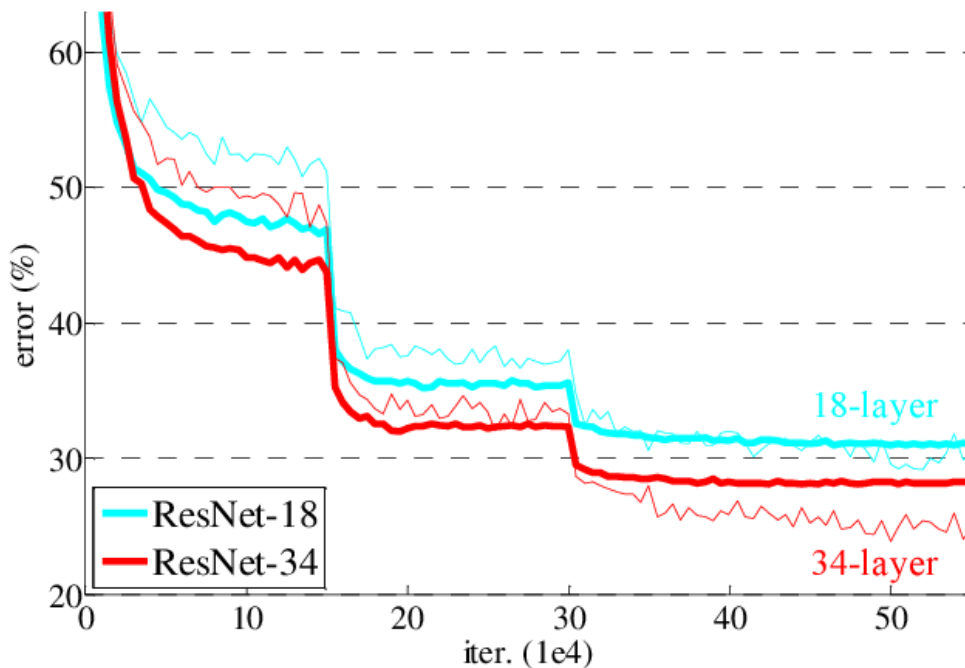
См. также AdamW — метод Adam с регуляризацией.

Geoffrey Everest Hinton
(Джеффри Хинтон) — автор
идеи RMSProp



В pytorch методы оптимизации находятся в модуле torch.optim:
pytorch.org/docs/stable/optim.html#algorithms

Несмотря на улучшения метода градиентного пуска у него всё равно остаётся важный гиперпараметр — коэффициент длины шага, learning rate λ . Остальные гиперпараметры как правило не меняют. Learning rate можно регулировать (см. изначальный алгоритм ГС). Один из простых подходов — понижать (обычно в разы) λ каждые несколько шагов или понижать при достижении плато. На рис. ниже пороги — это результаты понижения λ .



semanticscholar.org/paper/How-Does-Learning-Rate-Decay-Help-Modern-Neural-Networks-Long/8e79f3bfb2b1fc3cfd188a6aef046185a72170a

Классы для регулировки скорости learning rate в pytorch:
pytorch.org/docs/stable/optim.html#how-to-adjust-learning-rate

ССЫЛКИ

1. academy.yandex.ru/handbook/ml/article/optimizaciya-v-ml — Оптимизация в ML
2. habr.com/ru/post/413853 – Обзор градиентных методов в задачах математической оптимизации
3. cs231n.stanford.edu/slides/2023/lecture_3.pdf — Stanford CS231n: Deep Learning for Computer Vision
4. youtu.be/WZOYMG4xE – Обучение нейронных сетей.
5. programforyou.ru/projects/gradient-descent-optimization – живая анимация работы метода градиентного спуска и его модификаций
6. Современные численные методы оптимизации. Метод универсального градиентного спуска: Учебное пособие. — 2-е изд., испр. — М.: МЦНМО, 2021. — 272 с.
7. Пример на Python:
colab.research.google.com/drive/1VuiBRDRLVFzE5SxST4KwX3Uwn3MgdIAg?usp=sharing

ЗАДАЧИ

Задача 1

Найдите производные функций

$$y = \frac{3^x + 5}{\cos x}, \quad y = \frac{1}{2} \sqrt[3]{x^2} \cos x$$

Задача 2

Найти производную функции, заданной неявно

$$3x^4 y^5 = e^{7x-4y} = 4x^5 + 2y^4$$

Задача 3

Найти значение всех частных производных первого порядка в точке

$M(2, 1, 0)$

$$u = \arctg(xy^2 + z)$$

Задача 4

Найти значение градиента функции $u = \sin(x + 2y) + \sqrt{xyz}$ в точке

$$M_0 \left(\frac{\pi}{2}, 3\frac{\pi}{2}, 2 \right) \text{ по направлению вектора } \vec{l} = (4, 3)$$

Задача 5

Опишите две итерации метода градиентного спуска для функции

$$f(x, y) = (x^2 + y - 11)^2 + (x + y^2 - 7)^2 \text{ для } x_0 = (0, 3), \quad \lambda = 0.1.$$

Производные должны быть вычислены аналитически.