Optimization and Computational Linear Algebra for Data Science
Lecture 12: Gradient descent

Léo Miolane · leo.miolane@gmail.com

November 28, 2019

Warning: *This material is not meant to be lecture notes. It only gathers the main concepts and results from the lecture, without any additional explanation, motivation, examples, figures...*

In these notes, f denotes a twice differentiable **convex** function from $\mathbb{R}^n$ to $\mathbb{R}$.

1 Gradient descent

Given an initial point $x_0 \in \mathbb{R}^n$, the gradient descent algorithm follows the updates:

$$x_{t+1} = x_t - \alpha_t \nabla f(x_t), \tag{1}$$

where the step-size α_t remains to be determined. The step (1) is a very natural strategy to minimize f, since $-\nabla f(x)$ is the direction of steepest descent at x. Since $f(x+h) = f(x) + \langle \nabla f(x), h \rangle + o(\|h\|)$ we have

$$\begin{aligned} f(x_{t+1}) &= f(x_t) - \alpha_t \|\nabla f(x_t)\|^2 + o(\alpha_t) \\ &< f(x_t) \end{aligned}$$

for α_t small enough (provided that $\nabla f(x_t) \neq 0$). Hence is the step-sizes α_t are chosen very small, the sequence $(f(x_t))_{k \geq 0}$ is decreasing! However, if α_t are too small, the algorithm may never converge.

1.1 Convergence analysis

Notation: Given a symmetric matrix M we will denote by $\lambda_{\min}(M)$ and $\lambda_{\max}(M)$ the smallest and largest eigenvalues of M.

Definition 1.1

For $L, \mu > 0$, we say that a twice-differentiable convex function $f : \mathbb{R}^n \to \mathbb{R}$ is

- *L-smooth if for all $x \in \mathbb{R}^n$, $\lambda_{\max}(H_f(x)) \leq L$.*
- *μ-strongly convex if for all $x \in \mathbb{R}^n$, $\lambda_{\min}(H_f(x)) \geq \mu$.*

Remark 1.1. *L-smooth and μ-strongly convex functions are very convenient since they can be "sandwiched" as follows (see homework 9 for a proof):*

$$f(x) + \langle h, \nabla f(x) \rangle + \frac{\mu}{2}\|h\|^2 \leq f(x+h) \leq f(x) + \langle h, \nabla f(x) \rangle + \frac{L}{2}\|h\|^2,$$

for all $x, h \in \mathbb{R}^n$.

Theorem 1.1

Assume that f is L-smooth and that f admits a (global) minimizer $x^* \in \mathbb{R}^n$. Then the gradient descent iterates (1) with constant step-size $\alpha_k = 1/L$ verify

$$f(x_t) - f(x^*) \leq \frac{2L\|x_0 - x^*\|^2}{t+4}.$$

See Section 2.1.5 from [2] for a proof.

Why did we used step sizes of $1/L$? f is L-smooth, hence (see Remark 1.1) for all $x, h \in \mathbb{R}^n$:

$$f(x+h) \leq f(x) + \langle \nabla f(x), h \rangle + \frac{L}{2}\|h\|^2. \tag{2}$$

Then, one can check (exercise!) that when x is fixed, the minimum of the right-hand side is minimum for $h = -\frac{1}{L}\nabla f(x)$.

Theorem 1.2

Assume that f is L-smooth and μ-strongly convex. Then f admits a unique minimizer global x^* and the gradient descent iterates (1) with constant step-size $\alpha_k = 1/L$ verify

$$f(x_t) - f(x^*) \leq \Big(1 - \frac{\mu}{L}\Big)^t (f(x_0) - f(x^*)).$$

Remark 1.2. *The ratio $\kappa = \frac{L}{\mu} \in (0,1]$ is called the condition number. The smaller the condition number, the faster the convergence.*

Remark 1.3. *The μ-strong convexity of f implies (using Remark 1.1) that for all $x \in \mathbb{R}^n$,*

$$\frac{\mu}{2}\|x - x^*\|^2 \leq f(x) - f(x^*).$$

Combining this with Theorem 1.2 gives a bound of the distance to the minimizer x^*:

$$\|x_t - x^*\|^2 \leq \frac{2}{\mu}\Big(1 - \frac{\mu}{L}\Big)^t (f(x_0) - f(x^*)).$$

Proof. Let $t \geq 0$. Applying (2) for $x = x_t$ and $h = x_t - L^{-1}\nabla f(x_t)$, we get

$$f(x_{t+1}) \leq f(x_t) - \frac{1}{L}\|\nabla f(x_t)\|^2 + \frac{1}{2L}\|\nabla f(x_t)\|^2 = f(x_t) - \frac{1}{2L}\|\nabla f(x_t)\|^2.$$

Now, since f is μ-strongly convex, we have (exercise!) for all $x \in \mathbb{R}^n$

$$f(x) - f(x^*) \leq 2\mu\|\nabla f(x)\|^2. \tag{3}$$

We get that $f(x_{t+1}) \leq f(x_t) - \frac{\mu}{L}(f(x_t) - f(x^*))$, hence

$$f(x_{t+1}) - f(x^*) \leq (1 - \frac{\mu}{L})(f(x_t) - f(x^*)),$$

from which the theorem follows. $\square$

1.2 Choosing the step size in practice

In practice, one may not have access to L and need hence to choose the step size α_t. A popular method is the so-called "backtracking line search" a goes as follows. Fix a parameter $\beta \in (0,1)$. Start with $\alpha = 1$ and while

$$f(x_t - \alpha \nabla f(x_t)) > f(x_t) - \frac{\alpha}{2} \|\nabla f(x_t)\|^2,$$

update $\alpha = \beta \alpha$. Then choose $\alpha_t = \alpha$.

1.3 Accelerated gradient method

Gradient descent with momentum. Also known as "heavy ball" method, this scheme was introduced by Polyak in 1964. This is a way to prevent zigzagging trajectories when doing gradient descent by adding a momentum term:

$$x_{t+1} = x_t + v_t \quad \text{where} \quad v_t = \alpha_t v_{t-1} - \beta_t \nabla f(x_t),$$

for some α_t, β_t. The idea is to keep momentum from past iterations in order to avoid zigzagging.

Nesterov's accelerated gradient descent. Nesterov's accelerated gradient descent is a amelioration of idea of momentum.

$$x_{t+1} = x_t + v_t \quad \text{where} \quad v_t = \alpha_t v_{t-1} - \beta_t \nabla f(x_t + \alpha_t v_{t-1})$$

When α_t, β_t are properly chosen, it improves on the convergence rates of gradient descent (given by Theorems 1.1-1.2). Namely:

- if f is L-smooth and if its minimum is attained at some x^*, then for $\alpha_t = \frac{t-1}{t+2}$ and $\beta_t = 1/L$ we have
$$f(x_t) - f(x^*) \leq \frac{2L\|x_0 - x^*\|^2}{(t+1)^2}.$$
- if f is L-smooth and μ-strongly convex, then for $\alpha_t = \frac{1-\sqrt{\mu/L}}{1+\sqrt{\mu/L}}$ and $\beta_t = 1/L$ we have
$$f(x_t) - f(x^*) \leq L\|x_0 - x^*\|^2 \Big(1 - \sqrt{\mu/L}\Big)^t.$$

See for instance [4] for proofs of these results.

2 Newton's method

2.1 Newton's method

We assume here that f is μ-strongly convex and L-smooth. Newton's method performs updates according to

$$x_{t+1} = x_t - H_f(x_t)^{-1} \nabla f(x_t). \tag{4}$$

The (important!) difference with gradient descent is that the step-size α_k is now replaced by the inverse[1] of the Hessian of f. The idea begin Newton's method is to minimize the second order approximation of f at x_t :

$$f(x_t + h) \simeq f(x_t) + \langle \nabla f(x_t), h \rangle + \frac{1}{2} h^{\mathsf{T}} H_f(x_t) h \tag{5}$$

[1]The Hessian of f if indeed invertible at all x since its smallest eigenvalue is always greater than $\mu > 0$.

with respect to h and then choose $x_{t+1} = x_t + h$. It is an easy exercise to see that the minimizer of the right-hand side of (5) is $h = -H_f(x_t)^{-1}\nabla f(x_t)$, leading to the recursion (6).

It can be shown (see for instance [1]) that for t large enough

$$\|x_t - x^*\|^2 \leq Ce^{-\rho 2^t}, \tag{6}$$

where C, ρ are constants depending on f and x_0. We say that Newton's method converges *quadratically* to the minimizer x^*. Newton's method is much faster than gradient descent, whose speed (given by Theorem 1.2) is of order $C'e^{-\sqrt{\mu/L}\,t}$.

2.2 Quasi-Newton methods

The main drawback of Newton's method is its computational complexity. Each step of the method require to compute the inverse of the $n \times n$ Hessian matrix of f at x_t, which require $O(n^3)$ operations. This makes Newton's method unpractical for large scale applications.

Quasi-Newton methods have been developed to face these limitations. The idea behind quasi-Newton methods is to try to mimic the inverse Hessian $H_f(x_t)^{-1}$ by a sequence of symmetric positive semidefinite matrices $(Q_t)_{t\geq 0}$ that are recursively computed in an efficient way. We refer to Chapter 6 of [3] for a detailed introduction to this topic.

3 Stochastic gradient descent

3.1 Setting

A lot of machine learning problems fall in the following framework. Assume that there is a probability distribution P_0 on $\mathbb{R}^k \times \mathbb{R}^\ell$ and for $(X, Y) \sim P_0$ we would like to be able to estimate Y using X. To do so, we define a model depending on parameters $\theta \in \mathbb{R}^n$ that takes the form of a function

$$\varphi_\theta : \mathbb{R}^k \to \mathbb{R}^\ell.$$

Our estimate for Y will then be $\varphi_\theta(X)$. It remains then to find "good parameters" θ so that $\varphi_\theta(X) \simeq Y$. Hence we would like to find θ that minimize the risk

$$R(\theta) = \mathbb{E}\left[L(Y, \varphi_\theta(X))\right], \tag{7}$$

where $L : \mathbb{R}^\ell \times \mathbb{R}^\ell \to \mathbb{R}$ is some loss function, for instance $L(y, y') = \|y - y'\|^2$.

However we do not have access to the function R in general, because we do not know the distribution P_0. Instead, we usually have access to N samples $(X_i, Y_i) \overset{\text{i.i.d.}}{\sim} P_0$ and minimize then the *empirical risk*:

$$R_N(\theta) = \frac{1}{N}\sum_{i=1}^{N} L(Y_i, \varphi_\theta(X_i)) = \frac{1}{N}\sum_{i=1}^{N} f_i(\theta),$$

where we write $f_i(\theta) = L(Y_i, \varphi_\theta(X_i))$ for simplicity. In many cases, one minimizes $R_N(\theta)$ by gradient descent, following the opposite direction of the gradients:

$$\nabla R_N(\theta) = \frac{1}{N}\sum_{i=1}^{N} \nabla f_i(\theta).$$

However, when N (the number of training examples) and n (the number of parameters) are large, gradient descent becomes intractable since computing the gradient at a single point requires at least $N \times n$ computer operations.

3.2 Stochastic gradient descent

In order to face this issue, stochastic gradient descent (SGD) uses a single gradient $\nabla f_i(\theta)$ instead of the full-gradient $\nabla R_N(\theta) = \frac{1}{N}\sum_{i=1}^N \nabla f_i(\theta)$ to move:

$$\begin{aligned} &\text{Pick} \quad i \quad \text{uniformly at random in} \quad \{1,\dots,N\}, \\ &\text{Update} \quad \theta_{t+1} = \theta_t - \alpha_t \nabla f_i(\theta_t), \end{aligned}$$

where α_t is a step-size that has to be determined.

The direction of $\nabla f_i(\theta)$ is of course less accurate than the full gradient $\nabla R_N(\theta)$ and can hence be seen as a *noisy observation* of the full gradient. Hence, if we want SGD to converge we need that $\alpha_t \xrightarrow[t\to\infty]{} 0$. However, the step sizes should not decrease to fast, otherwise we may stay forever near the initial position θ_0 which will not perform well (because usually chosen at random). We have to make a trade-off:

- slowly decaying step-sizes: the gradient iterates θ_t moves fast, but keep a high variance.
- rapidly decaying step-sizes: the variance is reduced, but the iterates may move to slow to "forget" the initial condition θ_0.

3.3 Convergence analysis

In this section we assume N to be very large, and typically larger than the total number of steps of SGD that we are going to make. This allows us to never use at each time t a different gradient $\nabla f_{i_t}(\theta_t)$.

We start by rewriting SGD can be rewritten as

$$\theta_{t+1} = \theta_t - \alpha_t(\nabla R(\theta_t) + \varepsilon_t) \tag{8}$$

where $\varepsilon_t \overset{\text{def}}{=} \nabla f_{i_t}(\theta_t) - \nabla R(\theta_t)$. We use this notation to make clear that $\nabla f_{i_t}(\theta_t)$ is a noisy observation of $\nabla R(\theta_t)$, where ε_t denotes the error. Notice that $\mathbb{E}[\nabla f_{i_t}(\theta_t)|\theta_t] = \nabla R(\theta_t)$, thus $\mathbb{E}[\varepsilon_t|\theta_t] = 0$. We make the following additional assumptions:

- There exists $\sigma > 0$ such that $\mathbb{E}[\|\varepsilon_t\|^2|\theta_t] \le \sigma^2$ for all $t \ge 0$.
- R is μ-strongly convex and L-smooth.
- $\min_{\theta\in\mathbb{R}^n} R(\theta) = 0$. (This can be easily verified by shifting R by a constant.)

Applying R on both sides of (8) and using the L-smoothness of L, we get

$$R(\theta_{t+1}) \le R(\theta_t) - \alpha_t\langle \nabla R(\theta_t), \nabla R(\theta_t) + \varepsilon_t\rangle + \frac{L}{2}\alpha_t\|\nabla R(\theta_t) + \varepsilon_t\|^2.$$

Taking expectations on both sides gives:

$$\begin{aligned} \mathbb{E}R(\theta_{t+1}) &\le \mathbb{E}R(\theta_t) - \alpha_t\mathbb{E}\big\|\nabla R(\theta_t)\big\|^2 + \frac{L}{2}\alpha_t^2\left(\mathbb{E}\|\nabla R(\theta_t)\|^2 + \sigma^2\right). \\ &\le \mathbb{E}R(\theta_t) - \frac{\alpha_t}{2}\mathbb{E}\big\|\nabla R(\theta_t)\big\|^2 + \frac{L}{2}\alpha_t^2\sigma^2, \end{aligned}$$

assuming $\alpha_t \le 1/L$. By strong convexity we have (as in (3)): $R(\theta) \le 2\mu\|\nabla R(\theta)\|^2$. Hence:

$$\mathbb{E}R(\theta_{t+1}) \le (1-\mu\alpha_t)\,\mathbb{E}R(\theta_t) + \frac{L}{2}\alpha_t^2\sigma^2.$$

Using this inequalities, assuming that $(\alpha_t)_{t\ge 0}$ is non-increasing and less that $1/L$, one can prove that for all $0 \le t \le T$:

$$\mathbb{E}[R(\theta_T)] \le \mathbb{E}[R(\theta_0)]\prod_{t=1}^T(1-\mu\alpha_t) + \frac{L\sigma^2}{2}\exp\Big(-\mu\sum_{s=t+1}^T \alpha_t\Big)\sum_{t=1}^T \alpha_t^2 + \frac{\alpha_t}{\mu}.$$

Analysis of the first term. When $\alpha_t \to 0$,

$$\log \prod_{t=1}^{T}(1-\mu\alpha_t) = \sum_{t=1}^{T}\log(1-\mu\alpha_t) \sim -\mu\sum_{t=0}^{T}\alpha_t.$$

Hence, we need that $\sum_{t\geq 0}\alpha_t = +\infty$ in order to make the first term go to zero.

One can for instance take $\alpha_t = 1/t$, in which case one gets

$$\mathbb{E}[R(\theta_T)] = O(1/T).$$

3.4 Conclusion

We summarize below the performances of gradient descent and stochastic gradient descent for optimizing R, when it is L-smooth and μ-strongly convex:

	Cost per step	Error after t steps	Number of steps to get ε-error	Cost to get ε-error
GD	Nn	$O(e^{-\rho t})$	$O(\log(1/\varepsilon))$	$O(Nn\log(1/\varepsilon))$
SGD	n	$O(1/t)$	$O(1/\varepsilon)$	$O(n/\varepsilon)$

Further reading

See chapter 9 of [1] for more background on gradient descent and Newton's method.

References

[1] Stephen Boyd and Lieven Vandenberghe. *Convex optimization.* Cambridge university press, https://web.stanford.edu/~boyd/cvxbook/, 2004.

[2] Yurii Nesterov. *Lectures on convex optimization*, volume 137. Springer, 2018.

[3] Jorge Nocedal and Stephen Wright. *Numerical optimization.* Springer Science & Business Media, 2006.

[4] Mark Schmidt, Nicolas L Roux, and Francis R Bach. Convergence rates of inexact proximal-gradient methods for convex optimization. In *Advances in neural information processing systems*, pages 1458–1466, 2011.