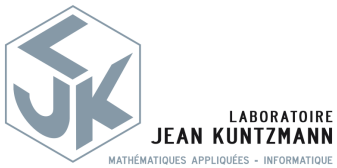

Lecture 6 : Adaptative methods - Adagrad and variants

In this lecture, we investigate adaptative methods, a family of optimization algorithms that become popular recently. Different from gradient descent or Nesterov acceleration iterations where all coefficients use the same step-size, algorithms of this family uses different step-sizes for different coordinates and

1 Adaptative methods - Adatative Gradien Descent

2 Motivations

Here are two examples of motivations:

1. **Quadratic optimization with gradient descent** : Consider an instance of quadratic functions $f : \mathbb{R}^2 \to \mathbb{R} : \theta \mapsto \frac{1}{2}\theta^\top \mathbf{A}\theta$ where:

$$\mathbf{A} = \begin{pmatrix} \mu & 0 \\ 0 & L \end{pmatrix}, L \geq \mu > 0.$$

Because f is μ-strongly convex and $\nabla f(\theta) = \mathbf{A}\theta$, f admits a unique optimal solution $\theta^\star = \mathbf{0}$. Moreover, we can explicitly calculate the update of gradient descent as:

$$\theta_k = \begin{pmatrix} 1-\mu\alpha & 0 \\ 0 & 1-L\alpha \end{pmatrix} \theta_{k-1} = \ldots = \begin{pmatrix} (1-\mu\alpha)^k & 0 \\ 0 & (1-L\alpha)^k \end{pmatrix} \theta_0$$

We observe that the convergence rate of two coefficients are different: $(1-\alpha\mu)^k$ and $(1-\alpha L)^k$.

In the regime where $\alpha\mu \approx 0$, the progress of the first coefficient is very slow because to reach ϵ, we will need:

$$k \approx \frac{1}{\alpha\mu} \log \frac{1}{\epsilon}.$$

A natural idea is to increase α because it will increase $\alpha\mu$ and reduce the number of necessary iterations to reach ϵ error. However, there is a limit that we can increase α: $\alpha \geq \frac{2}{L}$. Otherwise, the second coefficient will diverge because $|1 - \alpha L| > 1$.

Another possible solution is to choose distinct steps $(\alpha_1, \ldots, \alpha_d)$ for each coefficient. In our example, if we choose $\alpha_1 = \frac{1}{\mu}$ and $\alpha_2 = \frac{1}{L}$, we arrive at the optimal solution in one step.

2. **(Still) quadratic case and stochastic gradient descent**: The second motivation comes from the optimization problems in machine learning: Imagin that you have a linear regression of the following form:

$$\underset{\theta\in\mathbb{R}^d}{\text{Minimize}} \quad \frac{1}{n}\|\mathbf{X}\theta - y\|^2 = \frac{1}{n}\sum_{\ell=1}^{n}(\theta^\top x_i - y_i)^2.$$

Suppose that there are important features but it rarely appears (for example, being millionaire, ambidextrous, etc.), the stochastic gradient descent risk not to update the coefficients sufficiently regular because:

$$\theta_{k+1} = \theta_k - \alpha x_{i_k}(\theta_k^\top x_{i_k} - y_{i_k})$$

and most of the time, the coefficients of x_{i_k} corresponding to the important features are nulls.

The idea of adaptativity appears once again as an interesting solution. By using larger step-size from these coefficients, we can accelerate the convergence.

Nevertheless, this idea raise new problems: a priori, it is not trivial to choose good step-sizes for a specific optimization problem. This question pushed the following innovative idea: instead of choosing a good step-size from the beginnning, we use adaptative step-sizes that evolve through the iterations and incorporate (gradient) information. In the following section, we look at such an algorithm: Adaptative Gradient Descent (AdaGrad).

2.1 Adaptative Gradient Descent - AdaGrad

Consider the following generic optimization algorithm (OP),

$$\underset{\theta}{\text{Minimize}} \quad f(\theta) \tag{OP}$$

AdaGrad applies the following update rule:

$$\begin{aligned} g_{k+1} &= g_k + \nabla f(\theta_k) \odot \nabla f(\theta_k) \\ \theta_{k+1} &= \theta_k - \alpha_k \texttt{diag}(g_{k+1})^{-1/2}\nabla f(\theta_k), \end{aligned} \tag{1}$$

where $g_0 = \mathbf{0}$. By induction, we can prove that:

$$[g_{k+1}]_\ell = \sum_{i=0}^{k}([\nabla f(\theta_i)]_\ell)^2.$$

If this quantity is small, i.e., the ℓth coordinate is rarely updated, Adagrad will increase its step-size by dividing $\sqrt{[g_{k+1}]_\ell}$.

There are, in fact, other variants of Adagrad, with different norms, given by:

$$\begin{aligned} g_{k+1} &= g_k + \nabla f(\theta_k)^p \\ \theta_{k+1} &= \theta_k - \alpha_k \texttt{diag}(g_{k+1})^{-1/p}\nabla f(\theta_k). \end{aligned} \tag{2}$$

For $p = 2$, (2) reduces to (1). It is noteworthy that Adagrad does not belong to the set of first-order methods explained in the previous lecture because:

$$\theta_k \notin x_0 + \text{Lin}\{\nabla f(\theta_0), \nabla f(\theta_1), \ldots, \nabla f(\theta_{k-1})\}.$$

It is still a first-order method, though!!! (in a much larger sense).

2.2 Theoretical guarantee for AdaGrad in the minimization of convex functions

We prove a theoretical guarantee for Adagrad. In general, the performance of AdaGrad is not always better than that of gradient descent (with or without acceleration) from theoretical viewpoint. However, the following result implies that AdaGrad will perform much better when the gradients are sparse, which is a regular situation in reinforcement learning. In addition, the numerical result of plenty of models in deep learning suggests that the adaptative methods can greatly accelerate the training.

Theorem 2.1 (Convergence of AdaGrad for convex functions)**.** *Consider f a convex function which admits at least one optimal solution $\theta^\star$. Assume that $\|\theta_k - \theta^\star\|_\infty \leq D_\infty, \forall k \in \mathbb{N}$ and $\alpha_k = \alpha := \frac{D_\infty}{\sqrt{2}}$, we have:*

$$f(\bar{\theta}_{K+1}) - f(\theta^\star) \leq \frac{1}{K+1}\left(\alpha + \frac{1}{2\alpha}D_\infty^2\right)\sum_{\ell=1}^{d}\|g_{0:K}^\ell\|_2$$

where $\bar{\theta}_K = \frac{1}{K+1}\sum_{k=1}^{K+1}\theta_k$ and $g_{0:k}^\ell = \begin{pmatrix}[\nabla f(\theta_0)]_\ell & \dots [\nabla f(\theta_k)]_\ell\end{pmatrix}^\top$.

Proof. Denote $G_k = \texttt{diag}(g_k)^{1/2}$. By (1), we have:

$$\begin{aligned}\theta_{k+1} - \theta^\star &= \theta_k - \theta^\star - \alpha G_{k+1}^{-1}\nabla f(\theta_k)\\ G_{k+1}(\theta_{k+1} - \theta^\star) &= G_{k+1}(\theta_k - \theta^\star) - \alpha\nabla f(\theta_k)\end{aligned}$$

By taking the scalar product of the two above equations, we obtain:

$$(\underbrace{\theta_{k+1} - \theta^\star}_{\delta_{k+1}})^\top G_{k+1}(\theta_{k+1}-\theta^\star) = (\underbrace{\theta_k - \theta^\star}_{\delta_k})^\top G_{k+1}(\theta_k-\theta^\star) - 2\alpha\nabla f(\theta_k)^\top(\theta_k-\theta^\star) + \alpha^2\nabla f(\theta_k)^\top G_{k+1}^{-1}\nabla f(\theta_k).$$

By consequence, we have:

$$\nabla f(\theta_k)^\top\delta_k = \frac{\alpha}{2}\nabla f(\theta_k)^\top G_{k+1}^{-1}\nabla f(\theta_k) + \frac{1}{2\alpha}\left(\delta_k^\top G_{k+1}\delta_k - \delta_{k+1}^\top G_{k+1}\delta_{k+1}\right).$$

By summing this inequality for $k = 0, \dots, K$, we obtain:

$$\begin{aligned}\sum_{k=0}^{K}\nabla f(\theta_k)^\top\delta_k &= \frac{\alpha}{2}\sum_{k=0}^{K}\nabla f(\theta_k)^\top G_{k+1}^{-1}\nabla f(\theta_k) + \frac{1}{2\alpha}\sum_{k=0}^{K}\left(\delta_k^\top G_k\delta_k - \delta_{k+1}^\top G_{k+1}\delta_{k+1}\right)\\ &\quad + \frac{1}{2\alpha}\sum_{k=0}^{K}\delta_k^\top(G_{k+1} - G_k)\delta_k\\ &\leq \frac{\alpha}{2}\sum_{k=0}^{K}\nabla f(\theta_k)^\top G_{k+1}^{-1}\nabla f(\theta_k) + \frac{1}{2\alpha}\sum_{k=0}^{K}\delta_k^\top(G_{k+1} - G_k)\delta_k.\end{aligned}$$

since $g_0 = 0$. We analyse two terms:

$$\sum_{k=0}^{K}\nabla f(\theta_k)^\top G_{k+1}^{-1}\nabla f(\theta^k) = \sum_{k=0}^{K}\sum_{\ell=1}^{d}\frac{[|\nabla f(\theta_k)|]_\ell^2}{\|g_{0:k}^\ell\|_2} = \sum_{\ell=1}^{d}\sum_{k=0}^{K}\frac{[|\nabla f(\theta_k)|]_\ell^2}{\|g_{0:k}^\ell\|_2} \leq 2\sum_{\ell=1}^{d}\|g_{0:K}^\ell\|_2.$$

Indeed, it is sufficient to prove the following inequality: if we denote $\nu = \begin{pmatrix}\nu_0 & \dots \nu_d\end{pmatrix}^\top$ et $\nu_{0:\ell} = \begin{pmatrix}\nu_0 & \dots \nu_\ell\end{pmatrix}^\top$, then:

$$\sum_{i=0}^{d}\frac{\nu_i^2}{\|\nu_{0:t}\|_2} \leq 2\|\nu\|_2.$$

The proof can be done by induction. It is sufficient to show that:

$$2\|\nu_{0:d-1}\|_2 + \frac{\nu_d^2}{\|\nu\|} \leq 2\|\nu\|,$$

which is correct because for all $\beta \leq \alpha, \beta > 0$, we have:

$$2\sqrt{\beta^2 - \alpha^2} + \frac{\alpha^2}{\beta} \leq 2\beta.$$

We provide an upperbound for the second term as:

$$\sum_{k=0}^{K} \delta_k^\top (G_{k+1} - G_k)\delta_k \leq \sum_{k=0}^{K} D_\infty^2 \texttt{Trace}(G_{k+1} - G_k) \leq D_\infty^2 \texttt{Trace}(G_{k+1}) = D_\infty^2 \sum_{\ell=1}^{d} \|g_{0:K}^\ell\|_2$$

By combining the two upperbound and subtituting $\alpha = \frac{D_\infty}{\sqrt{2}}$, we get:

$$\sum_{k=0}^{K} \nabla f(\theta_k)^\top \delta_k \leq \left(\alpha + \frac{1}{2\alpha} D_\infty^2\right) \sum_{\ell=1}^{d} \|g_{0:K}^\ell\|_2 = \sqrt{2} D_\infty^2 \sum_{\ell=1}^{d} \|g_{0:K}^\ell\|_2.$$

The proof is finished by using the two following inequalities:

$$\nabla f(\theta_k)^\top \delta_k \geq f(\theta_k) - f(\theta^\star)$$
$$\frac{1}{K+1} \sum_{k=0}^{K} f(\theta_k) \geq f(\bar{\theta}_K)$$

since the function f is convex. □

3 Hybrid methods : RMSProp et Adam

Despite being one of the pioneer of the adaptative methods, AdaGrad is not the most frequently used algorithm in practice. In fact, for many learning tasks, first-order methods and hybrid methods that combine linear combination of past iterations and adaptative step-size are more preferable due to their excel on pratical performance. To concludethis course, we introduce two more hybrid methods: *Root Mean Square Propagation* (RMSProp) and *Adaptive Moment Estimation* (ADAM). We ignore their theoretical guarantees since the mathematical arguments are too technical and not intuitive. The theoretical guarantees for these methods constitute an active research domain until today.

RMSProp RMSProp uses two hyperparameters : $\alpha > 0$ (learning rate), $0 < \beta_2 < 1$ (parameters for the adaptative step-sizes)

$$\begin{aligned} v_{k+1} &= \beta_2 v_k + (1-\beta_2)[\nabla f(\theta_k) \odot \nabla f(\theta_k)] \\ \theta_{k+1} &= \theta_k - \alpha \cdot \texttt{diag}(\sqrt{v_{k+1}} + \epsilon)^{-1} \nabla f(\theta_k) \end{aligned} \tag{3}$$

ADAM ADAM uses three hyperparameters: $\alpha > 0$, and $\beta_1, \beta_2 \in (0, 1)$.

$$\begin{aligned} v_{k+1} &= \beta_2 v_k + (1-\beta_2)[\nabla f(\theta_k) \odot \nabla f(\theta_k)] \\ m_{k+1} &= \beta_1 m_k + (1-\beta_1)\nabla f(\theta_k) \\ \widehat{v_{k+1}} &= \frac{1}{1-\beta_2^{k+1}} v_{k+1} \\ \widehat{m_{k+1}} &= \frac{1}{1-\beta_1^{k+1}} m_{k+1} \\ \theta_{k+1} &= \theta_k - \alpha \cdot \texttt{diag}(\sqrt{\widehat{v_{k+1}}} + \epsilon)^{-1} \widehat{m_{k+1}} \end{aligned} \tag{4}$$

The factors $1 - \beta_2^{k+1}$ and $1 - \beta_1^{k+1}$ are bias correction in the sum.

In practice, ADAM is used more than RMSProp and is crucial in the training of Large Language Models (LLMs).