Making "Model Alchemy" More Scientific (Part 9): Classic Adaptive Gradient Algorithms

Su Jianlin

2026-09-05

The previous eight articles in this series all revolved around SGD and its learning rate. Starting from this article, we formally enter the world of adaptive gradient algorithms. It can be said that all current adaptive gradient algorithms share a common origin, namely the classic 2011 work *Adaptive Subgradient Methods for Online Learning and Stochastic Optimization*—the famous AdaGrad paper.

This paper generalizes the classical convergence results of SGD to a general preconditioner matrix form, then derives that the optimal preconditioner matrix is precisely the second moment of the gradient (raised to the $-1/2$ power), and finally takes the diagonal approximation to obtain AdaGrad. Although AdaGrad itself is not particularly popular, the follow-up works it inspired, such as RMSProp and Adam, are without question the mainstream optimizers. Moreover, later optimizers such as Shampoo and KL-Shampoo can still be considered its descendants.

In this article, let us revisit this classic work together.

1 Difficulty Analysis

Actually, in earlier articles we already briefly discussed the idea of an adaptive learning rate, for example in *Making "Model Alchemy" More Scientific (Part 5): Fine-Tuning the Learning Rate Based on Gradients*, but that was still tuning a scalar learning rate within the SGD framework. This time we consider the general update rule

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \eta_t \boldsymbol{H}_t^{-1} \boldsymbol{g}(\boldsymbol{x}_t, \boldsymbol{\theta}_t) \tag{1}$$

where $\boldsymbol{H}_t$ is a positive definite symmetric matrix; $\boldsymbol{H}_t^{-1}$ is usually also called the "preconditioner", which we can view as a matrix version of the learning rate. Clearly, when $\boldsymbol{H}_t = \boldsymbol{I}$ it reduces to SGD.

Why generalize to a matrix learning rate? Purely from input to output, we want to input a gradient vector and output an update vector; the most general linear transformation between two vectors is matrix multiplication, so a matrix learning rate is the ultimate

generalization. As for the positive definiteness restriction, it guarantees that $\boldsymbol{g}^\top \boldsymbol{H}_t^{-1}\boldsymbol{g} \geq 0$ holds for any $\boldsymbol{g}$, so that $-\boldsymbol{H}_t^{-1}\boldsymbol{g}$ remains a direction that decreases the loss.

In this section we first briefly analyze what difficulties arise for a general $\boldsymbol{H}_t$ if we directly apply the proof procedure of SGD. The starting point of all previous proofs is the following identity

$$\begin{aligned}\|\boldsymbol{\theta}_{t+1} - \boldsymbol{\varphi}\|^2 &= \|\boldsymbol{\theta}_t - \eta_t \boldsymbol{g}(\boldsymbol{x}_t, \boldsymbol{\theta}_t) - \boldsymbol{\varphi}\|^2 \\ &= \|\boldsymbol{\theta}_t - \boldsymbol{\varphi}\|^2 - 2\eta_t(\boldsymbol{\theta}_t - \boldsymbol{\varphi})\cdot \boldsymbol{g}(\boldsymbol{x}_t, \boldsymbol{\theta}_t) + \eta_t^2 \|\boldsymbol{g}(\boldsymbol{x}_t, \boldsymbol{\theta}_t)\|^2\end{aligned} \tag{2}$$

Then we manage to isolate $(\boldsymbol{\theta}_t - \boldsymbol{\varphi})\cdot \boldsymbol{g}(\boldsymbol{x}_t, \boldsymbol{\theta}_t)$ on one side, and use the convexity assumption to relate it to the loss values:

$$L(\boldsymbol{x}_t, \boldsymbol{\theta}_t) - L(\boldsymbol{x}_t, \boldsymbol{\varphi}) \leq (\boldsymbol{\theta}_t - \boldsymbol{\varphi})\cdot \boldsymbol{g}(\boldsymbol{x}_t, \boldsymbol{\theta}_t) \tag{3}$$

However, when we consider the general update rule (1), if we still use the same identity, then $(\boldsymbol{\theta}_t - \boldsymbol{\varphi})\cdot \boldsymbol{H}_t^{-1}\boldsymbol{g}(\boldsymbol{x}_t, \boldsymbol{\theta}_t)$ appears; because of the extra $\boldsymbol{H}_t^{-1}$, even if we assume convexity of the loss function, we can no longer relate this term to the loss function—the right-hand side of the convexity inequality (3) can only be a clean $(\boldsymbol{\theta}_t - \boldsymbol{\varphi})\cdot \boldsymbol{g}(\boldsymbol{x}_t, \boldsymbol{\theta}_t)$; inserting an $\boldsymbol{H}_t^{-1}$ in the middle makes it inapplicable.

2 Generalizing the Identity

So we must generalize this identity at the same time. The generalization idea uses the identity $\boldsymbol{x}\cdot\boldsymbol{y} = \boldsymbol{x}^\top\boldsymbol{y} = \mathrm{tr}(\boldsymbol{y}\boldsymbol{x}^\top)$; note that $\boldsymbol{y}\boldsymbol{x}^\top$ is a vector outer product, resulting in a matrix. This means we can first establish a matrix version of the identity at the outer-product level, and then take tr on both sides at the appropriate moment to recover the scalar form. In the matrix version of the identity, $\boldsymbol{H}_t^{-1}$ can then be moved out smoothly.

Concretely, we have (for simplicity, below we write $\boldsymbol{g}(\boldsymbol{x}_t, \boldsymbol{\theta}_t)$ as $\boldsymbol{g}_t$):

$$\begin{aligned}&(\boldsymbol{\theta}_{t+1} - \boldsymbol{\varphi})(\boldsymbol{\theta}_{t+1} - \boldsymbol{\varphi})^\top \\ =&(\boldsymbol{\theta}_t - \eta_t\boldsymbol{H}_t^{-1}\boldsymbol{g}_t - \boldsymbol{\varphi})(\boldsymbol{\theta}_t - \eta_t\boldsymbol{H}_t^{-1}\boldsymbol{g}_t - \boldsymbol{\varphi})^\top \\ =&(\boldsymbol{\theta}_t - \boldsymbol{\varphi})(\boldsymbol{\theta}_t - \boldsymbol{\varphi})^\top - \eta_t(\boldsymbol{\theta}_t - \boldsymbol{\varphi})\boldsymbol{g}_t^\top\boldsymbol{H}_t^{-1} - \eta_t\boldsymbol{H}_t^{-1}\boldsymbol{g}_t(\boldsymbol{\theta}_t - \boldsymbol{\varphi})^\top + \eta_t^2\boldsymbol{H}_t^{-1}\boldsymbol{g}_t\boldsymbol{g}_t^\top\boldsymbol{H}_t^{-1}\end{aligned} \tag{4}$$

Right-multiplying both sides by $\boldsymbol{H}_t$ and rearranging gives

$$\begin{aligned}&\eta_t(\boldsymbol{\theta}_t - \boldsymbol{\varphi})\boldsymbol{g}_t^\top + \eta_t\boldsymbol{H}_t^{-1}\boldsymbol{g}_t(\boldsymbol{\theta}_t - \boldsymbol{\varphi})^\top\boldsymbol{H}_t \\ =&(\boldsymbol{\theta}_t - \boldsymbol{\varphi})(\boldsymbol{\theta}_t - \boldsymbol{\varphi})^\top\boldsymbol{H}_t - (\boldsymbol{\theta}_{t+1} - \boldsymbol{\varphi})(\boldsymbol{\theta}_{t+1} - \boldsymbol{\varphi})^\top\boldsymbol{H}_t + \eta_t^2\boldsymbol{H}_t^{-1}\boldsymbol{g}_t\boldsymbol{g}_t^\top\end{aligned} \tag{5}$$

Taking tr on both sides and repeatedly using $\mathrm{tr}(\boldsymbol{A}\boldsymbol{B}) = \mathrm{tr}(\boldsymbol{B}\boldsymbol{A})$ gives

$$2\eta_t(\boldsymbol{\theta}_t - \boldsymbol{\varphi})\cdot\boldsymbol{g}_t = \|\boldsymbol{\theta}_t - \boldsymbol{\varphi}\|_{\boldsymbol{H}_t}^2 - \|\boldsymbol{\theta}_{t+1} - \boldsymbol{\varphi}\|_{\boldsymbol{H}_t}^2 + \eta_t^2\|\boldsymbol{g}_t\|_{\boldsymbol{H}_t^{-1}}^2 \tag{6}$$

This is the identity we wanted, where $\|\boldsymbol{x}\|_{\boldsymbol{H}} \triangleq \sqrt{\boldsymbol{x}^\top \boldsymbol{H}\boldsymbol{x}}$, which we call the Mahalanobis norm; the Euclidean norm is $\|\boldsymbol{x}\|_{\boldsymbol{I}}$, which we abbreviate as $\|\boldsymbol{x}\|$. Now the left-hand side recovers $(\boldsymbol{\theta}_t - \boldsymbol{\varphi}) \cdot \boldsymbol{g}_t$, allowing us to relate it to the convexity inequality (3), at the cost that the distance metric on the right-hand side changes from the Euclidean norm to a time-varying Mahalanobis norm.

3 Classical Bounding

The next steps are largely similar to those in *Making "Model Alchemy" More Scientific (Part 1): Convergence of SGD's Average Loss* and *Making "Model Alchemy" More Scientific (Part 2): Extending the Result to Unbounded Domains*. On the left-hand side we likewise use inequality (3) to relate it to the loss, then sum over $t = 1, 2, \cdots, T$ to get

$$2\sum_{t=1}^{T} \eta_t [L(\boldsymbol{x}_t, \boldsymbol{\theta}_t) - L(\boldsymbol{x}_t, \boldsymbol{\varphi})] \le \sum_{t=1}^{T} (\|\boldsymbol{\theta}_t - \boldsymbol{\varphi}\|_{\boldsymbol{H}_t}^2 - \|\boldsymbol{\theta}_{t+1} - \boldsymbol{\varphi}\|_{\boldsymbol{H}_t}^2) + \sum_{t=1}^{T} \eta_t^2 \|\boldsymbol{g}_t\|_{\boldsymbol{H}_t^{-1}}^2 \tag{7}$$

However, since in the first sum on the right-hand side $\|\boldsymbol{\theta}_{t+1} - \boldsymbol{\varphi}\|_{\boldsymbol{H}_t}^2$ is not $\|\boldsymbol{\theta}_{t+1} - \boldsymbol{\varphi}\|_{\boldsymbol{H}_{t+1}}^2$, the sum does not telescope. Here we use a classical trick from *Making "Model Alchemy" More Scientific (Part 1): Convergence of SGD's Average Loss*:

$$\begin{aligned}
&\sum_{t=1}^{T} (\|\boldsymbol{\theta}_t - \boldsymbol{\varphi}\|_{\boldsymbol{H}_t}^2 - \|\boldsymbol{\theta}_{t+1} - \boldsymbol{\varphi}\|_{\boldsymbol{H}_t}^2) \\
&= \|\boldsymbol{\theta}_1 - \boldsymbol{\varphi}\|_{\boldsymbol{H}_1}^2 - \|\boldsymbol{\theta}_{T+1} - \boldsymbol{\varphi}\|_{\boldsymbol{H}_T}^2 + \sum_{t=2}^{T} (\|\boldsymbol{\theta}_t - \boldsymbol{\varphi}\|_{\boldsymbol{H}_t}^2 - \|\boldsymbol{\theta}_t - \boldsymbol{\varphi}\|_{\boldsymbol{H}_{t-1}}^2) \\
&= \|\boldsymbol{\theta}_1 - \boldsymbol{\varphi}\|_{\boldsymbol{H}_1}^2 - \|\boldsymbol{\theta}_{T+1} - \boldsymbol{\varphi}\|_{\boldsymbol{H}_T}^2 + \sum_{t=2}^{T} \|\boldsymbol{\theta}_t - \boldsymbol{\varphi}\|_{\boldsymbol{H}_t - \boldsymbol{H}_{t-1}}^2
\end{aligned} \tag{8}$$

To continue the bounding, we need to add one more assumption on $\boldsymbol{H}_t$: for every t, $\boldsymbol{H}_t - \boldsymbol{H}_{t-1}$ is positive semidefinite, written $\boldsymbol{H}_t \succeq \boldsymbol{H}_{t-1}$; this corresponds to the earlier non-increasing learning rate assumption. In addition, for any positive semidefinite matrix $\boldsymbol{H}$, we have $\boldsymbol{H} \preceq \operatorname{tr}(\boldsymbol{H})\boldsymbol{I}$, so $\|\boldsymbol{x}\|_{\boldsymbol{H}} \le \sqrt{\operatorname{tr}(\boldsymbol{H})}\|\boldsymbol{x}\|$; this bound is actually quite loose, but it suffices for qualitative conclusions.

Using the above assumption and bound, and discarding the non-positive $-\|\boldsymbol{\theta}_{T+1} - \boldsymbol{\varphi}\|_{\boldsymbol{H}_T}^2$, we have

$$\begin{aligned}
&\sum_{t=1}^{T} (\|\boldsymbol{\theta}_t - \boldsymbol{\varphi}\|_{\boldsymbol{H}_t}^2 - \|\boldsymbol{\theta}_{t+1} - \boldsymbol{\varphi}\|_{\boldsymbol{H}_t}^2) \\
&\le \|\boldsymbol{\theta}_1 - \boldsymbol{\varphi}\|^2 \operatorname{tr}(\boldsymbol{H}_1) + \sum_{t=2}^{T} \|\boldsymbol{\theta}_t - \boldsymbol{\varphi}\|^2 \operatorname{tr}(\boldsymbol{H}_t - \boldsymbol{H}_{t-1}) \\
&\le R^2 \operatorname{tr}(\boldsymbol{H}_T)
\end{aligned} \tag{9}$$

Here $R = \max(\|\boldsymbol{\theta}_1 - \boldsymbol{\varphi}\|, \cdots, \|\boldsymbol{\theta}_T - \boldsymbol{\varphi}\|)$, i.e., a common upper bound of all $\|\boldsymbol{\theta}_t - \boldsymbol{\varphi}\|$. If you have doubts about this step, we can simply assume, as in *Making "Model Alchemy" More Scientific (Part 1): Convergence of SGD's Average Loss*, that the optimization takes place in a bounded domain; then all $\|\boldsymbol{\theta}_t - \boldsymbol{\varphi}\|$ naturally have a common upper bound R.

4 The Result Emerges

Substituting the bounds from the previous section back into (7) and taking $\boldsymbol{\varphi}$ to be the theoretical optimum $\boldsymbol{\theta}^*$, we obtain

$$2\sum_{t=1}^{T}\eta_t[L(\boldsymbol{x}_t,\boldsymbol{\theta}_t) - L(\boldsymbol{x}_t,\boldsymbol{\theta}^*)] \leq R^2\operatorname{tr}(\boldsymbol{H}_T) + \sum_{t=1}^{T}\eta_t^2\|\boldsymbol{g}_t\|_{\boldsymbol{H}_t^{-1}}^2 \tag{10}$$

For simplicity, first consider the case where η_t is a constant; rearranging gives

$$\frac{1}{T}\sum_{t=1}^{T}L(\boldsymbol{x}_t,\boldsymbol{\theta}_t) - L(\boldsymbol{x}_t,\boldsymbol{\theta}^*) \leq \frac{R^2}{2T\eta}\operatorname{tr}(\boldsymbol{H}_T) + \frac{\eta}{2T}\sum_{t=1}^{T}\|\boldsymbol{g}_t\|_{\boldsymbol{H}_t^{-1}}^2 \tag{11}$$

This is the general convergence result for $\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \eta\boldsymbol{H}_t^{-1}\boldsymbol{g}_t$. Now let us think in reverse: for a good optimizer, the right-hand side should be as small as possible, so minimizing the upper bound may find a better optimizer, consistent with the idea of *Making "Model Alchemy" More Scientific (Part 5): Fine-Tuning the Learning Rate Based on Gradients*. To this end, we first use the assumption $\boldsymbol{H}_1 \preceq \boldsymbol{H}_2 \preceq \cdots \preceq \boldsymbol{H}_T$ to get $\|\boldsymbol{g}_t\|_{\boldsymbol{H}_t^{-1}}^2 \geq \|\boldsymbol{g}_t\|_{\boldsymbol{H}_T^{-1}}^2$, hence

$$\frac{R^2}{2T\eta}\operatorname{tr}(\boldsymbol{H}_T) + \frac{\eta}{2T}\sum_{t=1}^{T}\|\boldsymbol{g}_t\|_{\boldsymbol{H}_t^{-1}}^2 \geq \frac{R^2}{2T\eta}\operatorname{tr}(\boldsymbol{H}_T) + \frac{\eta}{2T}\sum_{t=1}^{T}\|\boldsymbol{g}_t\|_{\boldsymbol{H}_T^{-1}}^2 \tag{12}$$

That is to say, taking the same $\boldsymbol{H}$ for all $\boldsymbol{H}_t$ achieves a smaller upper bound. Next, we rewrite the right-hand side as

$$\frac{R^2}{2T\eta}\operatorname{tr}(\boldsymbol{H}) + \frac{\eta}{2T}\sum_{t=1}^{T}\|\boldsymbol{g}_t\|_{\boldsymbol{H}^{-1}}^2 = \frac{R^2}{2T\eta}\operatorname{tr}(\boldsymbol{H}) + \frac{\eta}{2T}\operatorname{tr}(\boldsymbol{\Sigma}\boldsymbol{H}^{-1}) \tag{13}$$

where $\boldsymbol{\Sigma} = \sum_{t=1}^{T}\boldsymbol{g}_t\boldsymbol{g}_t^\top$ is the (unnormalized) second moment matrix. It is easy to see that η is actually a redundant parameter; the true independent variable is $\boldsymbol{H}/\eta$, which tells us that η can be any positive number. For simplicity, we take $\eta = R$, which makes $\frac{R^2}{2T\eta} = \frac{\eta}{2T}$; thus minimizing the upper bound is equivalent to

$$\min_{\boldsymbol{H}\succeq\boldsymbol{0}}\operatorname{tr}(\boldsymbol{H}) + \operatorname{tr}(\boldsymbol{\Sigma}\boldsymbol{H}^{-1}) \tag{14}$$

Interestingly, this is essentially the optimization objective of PSGD!

5 Solving the Objective

To solve the above objective, we again use the cyclic property of the trace to rewrite the two terms as

$$\begin{aligned}\operatorname{tr}(\boldsymbol{H}) &= \operatorname{tr}(\boldsymbol{H}^{1/2}\boldsymbol{H}^{1/2}) = \|\boldsymbol{H}^{1/2}\|_F^2 \\ \operatorname{tr}(\boldsymbol{\Sigma}\boldsymbol{H}^{-1}) &= \operatorname{tr}(\boldsymbol{\Sigma}^{1/2}\boldsymbol{H}^{-1/2}\boldsymbol{H}^{-1/2}\boldsymbol{\Sigma}^{1/2}) = \|\boldsymbol{\Sigma}^{1/2}\boldsymbol{H}^{-1/2}\|_F^2\end{aligned} \tag{15}$$

Then, using the AM–GM inequality,

$$\operatorname{tr}(\boldsymbol{H}) + \operatorname{tr}(\boldsymbol{\Sigma}\boldsymbol{H}^{-1}) = \|\boldsymbol{H}^{1/2}\|_F^2 + \|\boldsymbol{\Sigma}^{1/2}\boldsymbol{H}^{-1/2}\|_F^2 \geq 2\,\|\boldsymbol{H}^{1/2}\|_F\,\|\boldsymbol{\Sigma}^{1/2}\boldsymbol{H}^{-1/2}\|_F \tag{16}$$

and then, using the Cauchy–Schwarz inequality,

$$\|\boldsymbol{H}^{1/2}\|_F\,\|\boldsymbol{\Sigma}^{1/2}\boldsymbol{H}^{-1/2}\|_F \geq \operatorname{tr}(\boldsymbol{H}^{1/2}\boldsymbol{\Sigma}^{1/2}\boldsymbol{H}^{-1/2}) = \operatorname{tr}(\boldsymbol{\Sigma}^{1/2}) \tag{17}$$

The equality condition of the Cauchy–Schwarz inequality is that there exists $\lambda \geq 0$ such that $\boldsymbol{H}^{1/2} = \lambda\boldsymbol{\Sigma}^{1/2}\boldsymbol{H}^{-1/2}$, i.e., $\boldsymbol{H} = \lambda\boldsymbol{\Sigma}^{1/2}$; the equality condition of the AM–GM inequality is $\|\boldsymbol{H}^{1/2}\|_F = \|\boldsymbol{\Sigma}^{1/2}\boldsymbol{H}^{-1/2}\|_F$, which further determines $\lambda = 1$ after substitution. So the optimal solution is

$$\boldsymbol{H}^* = \boldsymbol{\Sigma}^{1/2} = \left(\sum_{t=1}^{T} \boldsymbol{g}_t\boldsymbol{g}_t^\top\right)^{1/2} \tag{18}$$

This is the most classical result of the AdaGrad paper; it can be regarded as the "primogenitor" of second-moment-based adaptive learning rate optimizers. From this we can see that tuning the learning rate based on the second moment is not a harebrained heuristic trick, but the natural product of the principle of "minimizing the convergence upper bound". When $\boldsymbol{\Sigma}$ is not invertible, we can add a $\kappa\boldsymbol{I}$ to it; this is an implementation detail and is not expanded upon in the theoretical derivation.

Intuitively, $\boldsymbol{H}^{-1}\boldsymbol{g}_t = \boldsymbol{\Sigma}^{-1/2}\boldsymbol{g}_t$ performs a "whitening" operation on the gradient, making the update more isotropic—which coincides exactly with the idea behind today's Muon optimizer.

6 Various Variants

However, although $\boldsymbol{H}^* = \boldsymbol{\Sigma}^{1/2}$ is theoretically optimal, it violates causality, because it assumes we know the gradients of all time steps in advance. This episode should be familiar: we already encountered it in *Making "Model Alchemy" More Scientific (Part 5): Fine-Tuning the Learning Rate Based on Gradients*. The solution is to truncate the sum to the current time step:

$$\boldsymbol{\Sigma}_t = \sum_{\tau=1}^{t} \boldsymbol{g}_\tau\boldsymbol{g}_\tau^\top, \qquad \boldsymbol{H}_t = \boldsymbol{\Sigma}_t^{1/2} \tag{19}$$

It is worth pointing out that $\boldsymbol{H}_t$ defined this way still satisfies $\boldsymbol{H}_t \succeq \boldsymbol{H}_{t-1}$, so the convergence result up to (11) still holds. However, keeping the full matrix $\boldsymbol{H}_t$ is extremely expensive in both storage and computation, and thus impractical. A naive solution is to consider the diagonal approximation, i.e., taking only the diagonal part of $\boldsymbol{\Sigma}_t$ and then taking the square root:

$$\boldsymbol{H}_t \approx \operatorname{diag}(\boldsymbol{\Sigma}_t)^{1/2} \qquad \Rightarrow \qquad \boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \eta \frac{\boldsymbol{g}_t}{\sqrt{\sum_{\tau=1}^t \boldsymbol{g}_\tau \odot \boldsymbol{g}_\tau}} \tag{20}$$

This is what we call the AdaGrad optimizer, where $\odot$ denotes the Hadamard product, and the division of two vectors is likewise componentwise in the Hadamard sense. Note that AdaGrad's per-component learning rate is monotonically decreasing, which is usually considered a bad property in practice (training tends to "stall" in the later stages). So RMSProp replaces the sum with a moving average (EMA):

$$\boldsymbol{v}_t = \beta \boldsymbol{v}_{t-1} + (1-\beta)\boldsymbol{g}_t \odot \boldsymbol{g}_t, \qquad \boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \eta \frac{\boldsymbol{g}_t}{\sqrt{\boldsymbol{v}_t}} \tag{21}$$

Note that after switching to the EMA, practical performance is often better, but $\boldsymbol{H}_t \succeq \boldsymbol{H}_{t-1}$ can no longer be guaranteed—this is where theory and practice begin to "part ways". If we further add momentum, we get the classic of classics, the Adam optimizer (for simplicity, the bias correction is omitted):

$$\begin{aligned} \boldsymbol{m}_t &= \beta_1 \boldsymbol{m}_{t-1} + (1-\beta_1)\boldsymbol{g}_t \\ \boldsymbol{v}_t &= \beta_2 \boldsymbol{v}_{t-1} + (1-\beta_2)\boldsymbol{g}_t \odot \boldsymbol{g}_t \end{aligned}, \qquad \boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \eta \frac{\boldsymbol{m}_t}{\sqrt{\boldsymbol{v}_t}} \tag{22}$$

Before Muon, Adam had always been the most mainstream optimizer, bar none; even with Muon, Adam remains an important component of the mainstream optimizers (for Embedding layers, LM Heads, etc.). As for RMSProp, it is usually used for training generative models, especially GAN models, because for such models adding momentum tends to cause mode collapse.

Beyond these classical optimizers, newer ones such as PSGD, Shampoo, KL-Shampoo, and AdaBK are, in fact, just other structured approximations of $\boldsymbol{\Sigma}$ or $\boldsymbol{\Sigma}^{-1/2}$ for matrix-multiplication parameters. In other words, even today, the preconditioner matrix $\boldsymbol{\Sigma}^{-1/2}$ derived from AdaGrad remains one of the important guiding principles for optimizers.

7 Article Summary

This article reviewed the foundational work of adaptive gradient algorithms—AdaGrad—and reproduced its complete derivation along the route of "convergence analysis $\to$ minimizing the upper bound $\to$ optimal preconditioner matrix". It can be said that AdaGrad's idea of "tuning the learning rate based on the second moment of the gradient" is almost the shared underlying principle of all subsequent adaptive learning rate optimizers.

When reprinting, please include the address of this article: https://kexue.fm/archives/11882
For more detailed reprinting matters, please refer to: *Scientific Spaces FAQ*