

Lecture 8: Optimization III — The Optimizer Zoo

Prof. Noah Golowich

Today we will interpret optimizer updates through their choice of norm, then introduce Shampoo and SOAP.

1 Review: AdaGrad, Adam, and Muon

Let $\theta_t \in \mathbb{R}^d$ be the parameter vector, $g_t = \nabla f_t(\theta_t)$ the batch gradient, and $\alpha_t > 0$ the learning rate. Below we use the convention that squares, square roots, and division of vectors act coordinate-wise.

AdaGrad. The diagonal AdaGrad update is

$$s_t = \sum_{\tau=1}^t g_\tau^2, \quad \theta_{t+1,i} = \theta_{t,i} - \alpha_t \frac{g_{t,i}}{\sqrt{s_{t,i}} + \epsilon}. \quad (1)$$

Full-matrix AdaGrad. Instead of keeping only squared coordinates, accumulate outer products [1]. The unconstrained update is

$$C_t = \sum_{\tau=1}^t g_\tau g_\tau^\top, \quad \theta_{t+1} = \theta_t - \alpha_t C_t^{-1/2} g_t. \quad (2)$$

Here $C_t^{-1/2}$ inverts the square roots of its positive eigenvalues and is zero on its kernel. A stabilized version replaces it by $(C_t + \epsilon I_d)^{-1/2}$. Taking only $\text{diag}(C_t)$ in this stabilized version recovers (1).

Adam. Initialize $m_0 = v_0 = 0$ and choose $\beta_1, \beta_2 \in [0, 1)$. Omitting bias correction, as in Lecture 6, the updates are

$$\begin{aligned} m_t &= \beta_1 m_{t-1} + (1 - \beta_1) g_t, \\ v_t &= \beta_2 v_{t-1} + (1 - \beta_2) g_t^2, \\ \theta_{t+1,i} &= \theta_{t,i} - \alpha_t \frac{m_{t,i}}{\sqrt{v_{t,i}} + \epsilon}. \end{aligned} \quad (3)$$

We keep Lecture 6's convention of placing ϵ inside the square root.

Muon (idealized). For a weight matrix $W_t \in \mathbb{R}^{m \times n}$, write $G_t = \nabla_W f_t(W_t)$. Initialize $B_0 = 0$, choose $\mu \in [0, 1)$, and compute

$$B_t = \mu B_{t-1} + G_t, \quad B_t = U_t \Sigma_t V_t^\top, \quad W_{t+1} = W_t - \alpha_t U_t V_t^\top. \quad (4)$$

In the context of the above, we write $\text{Ortho}(B_t) := U_t V_t^\top$; we set $\text{Ortho}(0) = 0$.

2 Optimizer updates as steepest descent

A gradient gives a linear approximation to the loss. In this section we interpret various GD alternatives as steepest descent of this linear approximation with respect to appropriate norms on the parameter vectors.

2.1 Gradient descent and sign gradient descent

Fix a differentiable loss $f : \mathbb{R}^d \rightarrow \mathbb{R}$, a current point θ , and $g = \nabla f(\theta)$. Its linearization at θ is $f(\theta) + \langle g, \Delta \rangle$, where Δ is the proposed displacement.

Proposition 2.1 (Gradient descent). *For every $\alpha > 0$, the unique minimizer of the linearized loss with a squared ℓ_2 penalty on the displacement is*

$$\arg \min_{\Delta \in \mathbb{R}^d} \left\{ f(\theta) + \langle g, \Delta \rangle + \frac{1}{2\alpha} \|\Delta\|_2^2 \right\} = -\alpha g. \quad (5)$$

Thus the next point is $\theta + \Delta = \theta - \alpha g$.

Proof. Completing the square gives

$$f(\theta) + \langle g, \Delta \rangle + \frac{1}{2\alpha} \|\Delta\|_2^2 = f(\theta) - \frac{\alpha}{2} \|g\|_2^2 + \frac{1}{2\alpha} \|\Delta + \alpha g\|_2^2.$$

The last term is nonnegative and equals zero exactly at $\Delta = -\alpha g$. $\square$

Next, recall that Adam without any momentum terms (i.e., $\beta_1 = \beta_2 = 0$) is equivalent to sign gradient descent (SignGD). As we show next, this may be interpreted as steepest descent in the ℓ_∞ norm.

Proposition 2.2 (Sign gradient descent [2, Proposition 2]). *For $g \in \mathbb{R}^d$ and $\lambda > 0$, a minimizer is*

$$-\frac{\|g\|_1}{\lambda} \text{sign}(g) \in \arg \min_{\Delta \in \mathbb{R}^d} \left\{ \langle g, \Delta \rangle + \frac{\lambda}{2} \|\Delta\|_\infty^2 \right\}. \quad (6)$$

Proof. For any Δ , let $a = \|\Delta\|_\infty$. Since $g_i \Delta_i \geq -|g_i|a$ in every coordinate,

$$\langle g, \Delta \rangle + \frac{\lambda}{2} \|\Delta\|_\infty^2 \geq -a \|g\|_1 + \frac{\lambda}{2} a^2 = \frac{\lambda}{2} \left(a - \frac{\|g\|_1}{\lambda} \right)^2 - \frac{\|g\|_1^2}{2\lambda} \geq -\frac{\|g\|_1^2}{2\lambda}.$$

If $g \neq 0$, choosing $\Delta = -(\|g\|_1 / \lambda) \text{sign}(g)$, so that $a = \|g\|_1 / \lambda$, makes both inequalities equalities. $\square$

This gives the SignGD update with step size $\alpha = \|g\|_1 / \lambda$.

2.2 The matrix analogue: spectral norm

What is the correct analogue of ℓ_∞ norm for matrix-valued parameters?

If we want the norm to be unchanged by orthogonal changes of input and output bases, a natural analogue is the *spectral norm*:

$$\|A\|_{\text{op}} := \sup_{\|x\|_2=1} \|Ax\|_2 = \max_i \sigma_i(A). \quad (7)$$

It takes the maximum singular value, just as ℓ_∞ takes the maximum absolute coordinate; in particular, $\|\text{diag}(x)\|_{\text{op}} = \|x\|_\infty$.

For matrices of the same shape, define the Frobenius inner product $\langle G, \Delta \rangle_F = \text{tr}(G^\top \Delta) = \sum_{i,j} G_{ij} \Delta_{ij}$. The counterpart of the ℓ_1 norm for matrices is the *nuclear norm* $\|G\|_* := \sum_i \sigma_i(G)$.

Proposition 2.3 (Spectral steepest descent [2, Proposition 5, $L = 1$]). *Let $G \in \mathbb{R}^{m \times n}$ and $\lambda > 0$. If $G = U\Sigma V^\top$ is its SVD restricted to positive singular values, then*

$$\Delta^* = -\frac{\|G\|_*}{\lambda} UV^\top \in \arg \min_{\Delta \in \mathbb{R}^{m \times n}} \left\{ \langle G, \Delta \rangle_F + \frac{\lambda}{2} \|\Delta\|_{\text{op}}^2 \right\}. \quad (8)$$

Proof. Write $G = \sum_{i=1}^q \sigma_i u_i v_i^\top$, where $q = \text{rank}(G)$ and $\sigma_i > 0$. For any Δ , put $a = \|\Delta\|_{\text{op}}$. Since u_i, v_i are unit vectors, Cauchy–Schwarz gives $u_i^\top \Delta v_i \geq -\|\Delta v_i\|_2 \geq -a$. Hence

$$\begin{aligned} \langle G, \Delta \rangle_F + \frac{\lambda}{2} \|\Delta\|_{\text{op}}^2 &= \sum_{i=1}^q \sigma_i u_i^\top \Delta v_i + \frac{\lambda}{2} a^2 \\ &\geq -a \|G\|_* + \frac{\lambda}{2} a^2 = \frac{\lambda}{2} \left(a - \frac{\|G\|_*}{\lambda} \right)^2 - \frac{\|G\|_*^2}{2\lambda} \geq -\frac{\|G\|_*^2}{2\lambda}. \end{aligned}$$

For $G \neq 0$, the matrix $UV^\top$ has spectral norm 1 and $\langle G, UV^\top \rangle_F = \sum_i \sigma_i = \|G\|_*$. Thus Δ^* attains the lower bound. $\square$

Thus orthogonalizing the gradient gives the spectral steepest-descent direction.

3 Optimizer Zoo

We can organize these optimizers by how they modify stochastic gradient descent (SGD), $\theta_{t+1} = \theta_t - \alpha_t g_t$.

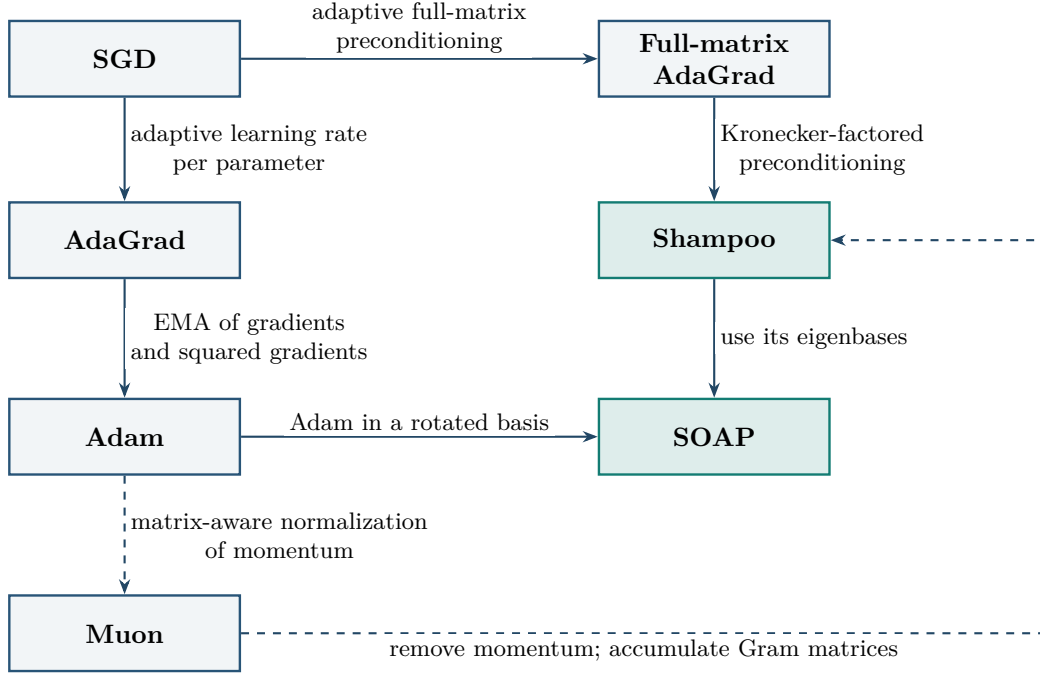


Figure 1: Relationships between update rules.

Summarizing the previous optimizers we have discussed: AdaGrad gives SGD an adaptive learning rate for each parameter; full-matrix AdaGrad also mixes coordinates. Adam adds a moving average of the gradients and replaces AdaGrad’s cumulative second moments by an exponential moving average. Muon likewise averages gradients, but replaces Adam’s coordinatewise division by matrix orthogonalization; it has no separate second-moment buffer. Next, we add two more optimizers to the zoo: Shampoo and SOAP.

3.1 Shampoo

Shampoo maintains one matrix for the rows of a gradient and one for its columns. In particular, for $W_t \in \mathbb{R}^{m \times n}$ and $G_t = \nabla f_t(W_t)$, the matrices $G_t G_t^\top \in \mathbb{R}^{m \times m}$ and $G_t^\top G_t \in \mathbb{R}^{n \times n}$ record inner products between rows and between columns, respectively. Their accumulated versions serve as preconditioners for each update.

We define matrix powers to act on eigenvalues: if $A = Q \text{diag}(\lambda_1, \dots, \lambda_k) Q^\top$ with each $\lambda_i > 0$, then $A^{-1/4} = Q \text{diag}(\lambda_1^{-1/4}, \dots, \lambda_k^{-1/4}) Q^\top$. See algorithm 1.

Removing accumulation gives Muon without momentum. We next show how Shampoo is related to Muon: to do so, we set $\epsilon = 0$ and replace L_t, R_t by the current Gram matrices $G_t G_t^\top, G_t^\top G_t$. Writing the positive-singular-value SVD as $G_t = U_t \Sigma_t V_t^\top$, we obtain that the Shampoo update may be written as

$$\begin{aligned}
 (G_t G_t^\top)^{-1/4} G_t (G_t^\top G_t)^{-1/4} &= (U_t \Sigma_t^{-1/2} U_t^\top) (U_t \Sigma_t V_t^\top) (V_t \Sigma_t^{-1/2} V_t^\top) \\
 &= U_t \Sigma_t^{-1/2} \Sigma_t \Sigma_t^{-1/2} V_t^\top = U_t V_t^\top.
 \end{aligned} \tag{9}$$

Algorithm 1 Shampoo for matrices [4, Algorithm 1]**Require:** Learning rate $\alpha > 0$, stabilizer $\epsilon > 0$, number of rounds T .

- 1: $W_1 = 0_{m \times n}$, $L_0 = \epsilon I_m$, $R_0 = \epsilon I_n$.
- 2: **for** $t = 1, \dots, T$ **do**
- 3: Receive f_t and compute $G_t = \nabla f_t(W_t)$.
- 4: $L_t = L_{t-1} + G_t G_t^\top$.
- 5: $R_t = R_{t-1} + G_t^\top G_t$.
- 6: $W_{t+1} = W_t - \alpha L_t^{-1/4} G_t R_t^{-1/4}$.
- 7: **end for**

Here we used $U_t^\top U_t = V_t^\top V_t = I$. This is exactly (4) with $\mu = 0$ [2, 3]. In this sense, Shampoo is Muon with no momentum but with accumulation in the second-moment matrices.

A structured version of full-matrix AdaGrad. To express the update as a single preconditioned vector step, let $\text{vec}(G)$ stack the *rows* of G into a column vector:

$$\text{vec}(G) = (G_{11}, \dots, G_{1n}, G_{21}, \dots, G_{mn})^\top \in \mathbb{R}^{mn}.$$

For $A \in \mathbb{R}^{m \times m}$ and $B \in \mathbb{R}^{n \times n}$, their Kronecker product $A \otimes B$ is the $mn \times mn$ block matrix whose (i, k) block is $A_{ik}B$.

Lemma 3.1 ([4, Lemma 4]). For $G \in \mathbb{R}^{m \times n}$, $L \in \mathbb{R}^{m \times m}$, and $R \in \mathbb{R}^{n \times n}$,

$$\text{vec}(LGR) = (L \otimes R^\top) \text{vec}(G). \quad (10)$$

Proof. Index a coordinate of the vectorized matrix by (i, j) . The corresponding coordinate on the right is $\sum_{k=1}^m \sum_{\ell=1}^n L_{ik}(R^\top)_{j\ell} G_{k\ell} = \sum_{k,\ell} L_{ik} G_{k\ell} R_{\ell j} = (LGR)_{ij}$, which is the corresponding coordinate on the left. $\square$

Set $w_t = \text{vec}(W_t)$ and $g_t = \text{vec}(G_t)$. Since the preconditioners are symmetric, lemma 3.1 gives

$$H_t := L_t^{1/4} \otimes R_t^{1/4}, \quad w_{t+1} = w_t - \alpha H_t^{-1} g_t. \quad (11)$$

Full-matrix AdaGrad uses $C_t^{1/2}$ in place of H_t , where $C_t = \sum_{s=1}^t g_s g_s^\top$. Shampoo replaces this unrestricted matrix by the Kronecker-factored preconditioner $L_t^{1/4} \otimes R_t^{1/4}$, storing $m^2 + n^2$ entries instead of $m^2 n^2$. It need not equal the full AdaGrad preconditioner.

Online convex optimization with matrix parameters. We next discuss a regret bound for Shampoo in the online convex optimization setting [4]. Recall that in this setting, at each round t , we choose $W_t \in \mathbb{R}^{m \times n}$ before seeing a differentiable convex loss $f_t : \mathbb{R}^{m \times n} \rightarrow \mathbb{R}$, incur $f_t(W_t)$, and observe $G_t = \nabla f_t(W_t)$. For a fixed comparator $W^* \in \mathbb{R}^{m \times n}$, we aim to minimize the *regret*

$$\mathcal{R}_T(W^*) := \sum_{t=1}^T [f_t(W_t) - f_t(W^*)]. \quad (12)$$

Theorem 3.2 (Shampoo regret [4, Theorem 7]). Suppose $\text{rank}(G_t) \leq r$ for every t , where $1 \leq r \leq \min\{m, n\}$, and fix W^* and $D > 0$ with $\max_{1 \leq t \leq T} \|W_t - W^*\|_F \leq D$. For algorithm 1 with $\alpha = D/\sqrt{2r}$,

$$\mathcal{R}_T(W^*) \leq \sqrt{2r} D \text{tr}(L_T^{1/4}) \text{tr}(R_T^{1/4}), \quad (13)$$

where

$$L_T = \epsilon I_m + \sum_{t=1}^T G_t G_t^\top, \quad R_T = \epsilon I_n + \sum_{t=1}^T G_t^\top G_t.$$

More generally, for any constant $\alpha > 0$, the bound is $\mathcal{R}_T(W^*) \leq (D^2/(2\alpha) + \alpha r) \text{tr}(L_T^{1/4}) \text{tr}(R_T^{1/4})$.

If D stays bounded and $\|G_t\|_{\text{op}} \leq K$, then $L_T \preceq (\epsilon + TK^2)I_m$ and $R_T \preceq (\epsilon + TK^2)I_n$. Thus the right side of (13) is at most $\sqrt{2r} D m n \sqrt{\epsilon + TK^2}$, giving $O(\sqrt{T})$ regret for fixed dimensions and K . Here $A \preceq B$ means that $B - A$ is positive semidefinite.

Comparison with AdaGrad. With $g_t = \text{vec}(G_t)$ and $C_T = \sum_t g_t g_t^\top$, the corresponding full-matrix AdaGrad guarantee, with learning rate $D/\sqrt{2}$, is

$$\mathcal{R}_T(W^*) \leq \sqrt{2} D \text{tr}(C_T^{1/2}) \quad (14)$$

under its own iterate-distance bound D [1]. For comparison, diagonal AdaGrad with learning rate $D_\infty/\sqrt{2}$ satisfies

$$\mathcal{R}_T(W^*) \leq \sqrt{2} D_\infty \sum_{i=1}^m \sum_{j=1}^n \sqrt{\sum_{t=1}^T G_{t,ij}^2}, \quad D_\infty \geq \max_{t,i,j} |W_{t,ij} - W_{ij}^*|. \quad (15)$$

Evaluating the bounds for rank- r gradients. Fix $1 \leq r \leq \min\{m, n\}$ and matrices $U \in \mathbb{R}^{m \times r}$ and $V \in \mathbb{R}^{n \times r}$ with orthonormal columns: $U^\top U = V^\top V = I_r$. Take $G_t = UV^\top$ on every round, so each gradient has rank r and all its nonzero singular values equal 1. Compare the two upper bounds using the same D and let $\epsilon \downarrow 0$ in the Shampoo bound. With $g = \text{vec}(UV^\top)$, the limiting accumulators are

$$C_T = T g g^\top, \quad L_T = T U U^\top, \quad R_T = T V V^\top.$$

Since $\|g\|_2^2 = \|UV^\top\|_F^2 = r$, the matrix C_T has one nonzero eigenvalue, rT . Each of L_T, R_T has r nonzero eigenvalues, all equal to T . Consequently,

$$\text{tr}(C_T^{1/2}) = \sqrt{rT}, \quad \text{tr}(L_T^{1/4}) = rT^{1/4}, \quad \text{tr}(R_T^{1/4}) = rT^{1/4},$$

and the guarantees evaluate to

$$\text{Full-matrix AdaGrad: } \sqrt{2} D \text{tr}(C_T^{1/2}) = \sqrt{2r} D \sqrt{T},$$

$$\text{Shampoo: } \sqrt{2r} D \text{tr}(L_T^{1/4}) \text{tr}(R_T^{1/4}) = r^2 \sqrt{2r} D \sqrt{T}.$$

In fact, on a common gradient sequence with the same D , the stated Shampoo bound cannot be smaller. For gradients of rank at most r , we have [4, Lemma 8]

$$\frac{1}{r} C_T \preceq L_T^{1/2} \otimes R_T^{1/2}.$$

Taking positive semidefinite square roots preserves this ordering, and taking traces gives

$$\text{tr}(C_T^{1/2}) \leq \sqrt{r} \text{tr}(L_T^{1/4} \otimes R_T^{1/4}) = \sqrt{r} \text{tr}(L_T^{1/4}) \text{tr}(R_T^{1/4}).$$

Multiplying by $\sqrt{2}D$ compares the two upper bounds directly. In this example, Shampoo's regret bound is a factor r^2 larger than full-matrix AdaGrad's, with equality when $r = 1$, while storing only $m^2 + n^2$ preconditioner entries instead of $m^2 n^2$.

3.2 SOAP

SOAP uses the eigenvectors of Shampoo's row and column statistics to choose coordinates, then applies Adam's entrywise normalization in those coordinates [5]. We first express this as a change of basis in the vectorized parameter space, then show how to perform the same update using matrices.

The tensor-product basis. Retain the row-stacking convention of lemma 3.1, and let $\text{mat} : \mathbb{R}^{mn} \rightarrow \mathbb{R}^{m \times n}$ denote its inverse. Write $w_t = \text{vec}(W_t)$ and define the vectorized loss $F_t(w) := f_t(\text{mat}(w))$. For orthogonal matrices $Q_{L,t} \in \mathbb{R}^{m \times m}$ and $Q_{R,t} \in \mathbb{R}^{n \times n}$, set

$$Q_t := Q_{L,t} \otimes Q_{R,t} \in \mathbb{R}^{mn \times mn}. \quad (16)$$

This is an orthogonal change of basis, since

$$Q_t^\top Q_t = (Q_{L,t}^\top Q_{L,t}) \otimes (Q_{R,t}^\top Q_{R,t}) = I_m \otimes I_n = I_{mn}.$$

Its basis vectors are all pairwise tensor products of a column of $Q_{L,t}$ and a column of $Q_{R,t}$. In algorithm 2 we introduce SOAP in vectorized coordinates. Write $\odot$ for entrywise multiplication and $\mathbf{1} \in \mathbb{R}^{mn}$ for the all-ones vector.

Algorithm 2 SOAP in vectorized coordinates

Require: Initial $w_1 \in \mathbb{R}^{mn}$, learning rates $\alpha_t > 0$, $\beta_1, \beta_2 \in [0, 1)$, $\epsilon > 0$, number of rounds T .

- 1: Initialize $L_0 = 0_{m \times m}$, $R_0 = 0_{n \times n}$, $m_0 = v_0 = 0_{mn}$.
 - 2: **for** $t = 1, \dots, T$ **do**
 - 3: Compute orthogonal eigendecompositions:

$$L_{t-1} = Q_{L,t} \Lambda_{L,t} Q_{L,t}^\top, \quad R_{t-1} = Q_{R,t} \Lambda_{R,t} Q_{R,t}^\top.$$
 - 4: $Q_t = Q_{L,t} \otimes Q_{R,t}$.
 - 5: Sample a batch with loss F_t and compute $g_t = \nabla F_t(w_t)$.
 - 6: $g'_t = Q_t^\top g_t$.
 - 7: $m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t$.
 - 8: $v_t = \beta_2 v_{t-1} + (1 - \beta_2) (g'_t \odot g'_t)$.
 - 9: $D_t = \text{diag}(v_t + \epsilon \mathbf{1})^{-1/2}$.
 - 10: $w_{t+1} = w_t - \alpha_t Q_t D_t Q_t^\top m_t$.
 - 11: $G_t = \text{mat}(g_t)$.
 - 12: $L_t = \beta_2 L_{t-1} + (1 - \beta_2) G_t G_t^\top$.
 - 13: $R_t = \beta_2 R_{t-1} + (1 - \beta_2) G_t^\top G_t$.
 - 14: **end for**
-

At $t = 1$ we choose $Q_{L,1} = I_m$ and $Q_{R,1} = I_n$, which diagonalize the zero matrices.

The matrix implementation. The matrix Q_t has $m^2 n^2$ entries, but we can apply its change of basis by multiplying on the left and right by the two smaller factors. The following algorithm stores $M_t, V_t \in \mathbb{R}^{m \times n}$ in place of m_t, v_t and uses the same initialization and eigendecomposition convention.

Algorithm 3 SOAP in matrix coordinates, with an exact eigenbasis at every step [5, Algorithm 3]

Require: Initial $W_1 \in \mathbb{R}^{m \times n}$, learning rates $\alpha_t > 0$, $\beta_1, \beta_2 \in [0, 1)$, $\epsilon > 0$, number of rounds T .

- 1: Initialize $L_0 = 0_{m \times m}$, $R_0 = 0_{n \times n}$, $M_0 = V_0 = 0_{m \times n}$.
 - 2: **for** $t = 1, \dots, T$ **do**
 - 3: Compute orthogonal eigendecompositions at the beginning of the iteration:

$$L_{t-1} = Q_{L,t} \Lambda_{L,t} Q_{L,t}^\top, \quad R_{t-1} = Q_{R,t} \Lambda_{R,t} Q_{R,t}^\top.$$
 - 4: Sample a batch with loss f_t and compute $G_t = \nabla f_t(W_t)$.
 - 5: $G'_t = Q_{L,t}^\top G_t Q_{R,t}$.
 - 6: $M_t = \beta_1 M_{t-1} + (1 - \beta_1) G_t$.
 - 7: $M'_t = Q_{L,t}^\top M_t Q_{R,t}$.
 - 8: $V_t = \beta_2 V_{t-1} + (1 - \beta_2)(G'_t \odot G'_t)$.
 - 9: $N'_t = M'_t / \sqrt{V_t + \epsilon}$ ▷ entrywise operations
 - 10: $N_t = Q_{L,t} N'_t Q_{R,t}^\top$.
 - 11: $W_{t+1} = W_t - \alpha_t N_t$.
 - 12: $L_t = \beta_1 L_{t-1} + (1 - \beta_1) G_t G_t^\top$.
 - 13: $R_t = \beta_2 R_{t-1} + (1 - \beta_2) G'_t G'_t$.
 - 14: **end for**
-

Proposition 3.3 (Equivalence of the vector and matrix updates). *Run algorithms 2 and 3 with $w_1 = \text{vec}(W_1)$, the same batches and hyperparameters, and the same eigendecomposition choices. Then for every t ,*

$$w_t = \text{vec}(W_t), \quad m_t = \text{vec}(M_t), \quad v_t = \text{vec}(V_t),$$

and both algorithms maintain identical L_t, R_t . In particular, the two-sided matrix rotation is the tensor-product change of basis:

$$\text{vec}(Q_{L,t}^\top A Q_{R,t}) = Q_t^\top \text{vec}(A) \quad \text{for every } A \in \mathbb{R}^{m \times n}. \quad (17)$$

Proof. First, lemma 3.1 gives

$$\text{vec}(Q_{L,t}^\top A Q_{R,t}) = (Q_{L,t}^\top \otimes Q_{R,t}^\top) \text{vec}(A) = Q_t^\top \text{vec}(A),$$

and likewise $\text{vec}(Q_{L,t} A Q_{R,t}^\top) = Q_t \text{vec}(A)$.

We prove the state identities by induction. They hold at initialization. Suppose they hold at the start of round t . The eigendecomposition convention gives the same $Q_{L,t}, Q_{R,t}$ in both algorithms. For any $h \in \mathbb{R}^{mn}$, the definition of F_t and preservation of inner products give

$$\left. \frac{d}{ds} F_t(w_t + sh) \right|_{s=0} = \langle G_t, \text{mat}(h) \rangle_F = \langle \text{vec}(G_t), h \rangle.$$

Thus $g_t = \nabla F_t(w_t) = \text{vec}(G_t)$. Applying (17) to G_t gives $\text{vec}(G'_t) = Q_t^\top g_t = g'_t$. Linearity of vectorization yields

$$\text{vec}(M_t) = \beta_1 m_{t-1} + (1 - \beta_1) g_t = m_t, \quad \text{vec}(M'_t) = Q_t^\top m_t.$$

Vectorization also preserves entrywise products, so

$$\text{vec}(V_t) = \beta_2 v_{t-1} + (1 - \beta_2)(g'_t \odot g'_t) = v_t.$$

Entrywise division by $\sqrt{V_t + \epsilon}$ therefore gives $\text{vec}(N'_t) = D_t Q_t^\top m_t$. Rotating back and updating the parameters yields

$$\text{vec}(N_t) = Q_t D_t Q_t^\top m_t, \quad \text{vec}(W_{t+1}) = w_t - \alpha_t Q_t D_t Q_t^\top m_t = w_{t+1}.$$

Finally, both algorithms update L_t, R_t using the same G_t and the same previous Gram matrices. This proves the induction step. $\square$

Rough interpretation: Adam in a rotated basis. First hold $Q = Q_L \otimes Q_R$ fixed and set $z_t = Q^\top w_t$ and $m'_t = Q^\top m_t$. By the chain rule, the gradient of $z \mapsto F_t(Qz)$ at z_t is $g'_t = Q^\top g_t$. Multiplying the first-moment and parameter updates in algorithm 2 by $Q^\top$ gives

$$\begin{aligned} m'_t &= \beta_1 m'_{t-1} + (1 - \beta_1) g'_t, \\ v_t &= \beta_2 v_{t-1} + (1 - \beta_2) (g'_t \odot g'_t), \\ z_{t+1} &= z_t - \alpha_t D_t m'_t = z_t - \alpha_t \frac{m'_t}{\sqrt{v_t + \epsilon \mathbf{1}}}. \end{aligned}$$

These are exactly Adam's recurrences in the z coordinates. By (17), $z_t = \text{vec}(Q_L^\top W_t Q_R)$. Choosing $Q_L = I_m$ and $Q_R = I_n$ gives ordinary Adam in the original coordinates.

In SOAP the bases change with t . Reprojecting the original-coordinate m_t gives the exact first moment in the current basis, whereas the second-moment recurrence retains the squares measured in past bases:

$$\begin{aligned} Q_t^\top m_t &= (1 - \beta_1) \sum_{s=1}^t \beta_1^{t-s} Q_s^\top g_s, \\ v_t &= (1 - \beta_2) \sum_{s=1}^t \beta_2^{t-s} [(Q_s^\top g_s) \odot (Q_s^\top g_s)]. \end{aligned}$$

The connection to Shampoo. The same tensor-product basis diagonalizes Shampoo's preconditioner. Write the eigendecompositions of the accumulated Shampoo statistics as $L_t = P_{L,t} \Lambda_{L,t} P_{L,t}^\top$ and $R_t = P_{R,t} \Lambda_{R,t} P_{R,t}^\top$, and set $P_t = P_{L,t} \otimes P_{R,t}$. Then the preconditioner from (11) satisfies

$$H_t = L_t^{1/4} \otimes R_t^{1/4} = P_t (\Lambda_{L,t}^{1/4} \otimes \Lambda_{R,t}^{1/4}) P_t^\top. \quad (18)$$

This follows by substituting the two eigendecompositions and using $(A \otimes B)(C \otimes D) = (AC) \otimes (BD)$. Consequently, Shampoo's vector update is

$$w_{t+1} = w_t - \alpha P_t (\Lambda_{L,t}^{-1/4} \otimes \Lambda_{R,t}^{-1/4}) P_t^\top g_t.$$

If $\ell_{t,i}$ and $\rho_{t,j}$ are the corresponding row and column eigenvalues, the diagonal scaling for coordinate (i, j) is $(\ell_{t,i} \rho_{t,j})^{-1/4}$.

Comparison to SOAP. SOAP chooses its basis from exponentially averaged Gram matrices (as opposed from the accumulated Gram matrices L_t, R_t) and instead uses the learned Adam scaling D_t (as opposed to the $(\ell_{t,i} \rho_{t,j})^{-1/4}$ above).

References

- [1] John Duchi, Elad Hazan, and Yoram Singer. Adaptive Subgradient Methods for Online Learning and Stochastic Optimization. *Journal of Machine Learning Research*, 12:2121–2159, 2011. <https://www.jmlr.org/papers/volume12/duchi11a/duchi11a.pdf>.
- [2] Jeremy Bernstein and Laker Newhouse. Old Optimizer, New Norm: An Anthology. 2024. <https://arxiv.org/abs/2409.20325>.
- [3] Keller Jordan, Yuchen Jin, Vlado Boza, Jiacheng You, Franz Cesista, Laker Newhouse, and Jeremy Bernstein. Muon: An optimizer for hidden layers in neural networks. 2024. <https://kellerjordan.github.io/posts/muon/>.
- [4] Vineet Gupta, Tomer Koren, and Yoram Singer. Shampoo: Preconditioned Stochastic Tensor Optimization. *International Conference on Machine Learning*, 2018. <https://arxiv.org/abs/1802.09568>.
- [5] Nikhil Vyas, Depen Morwani, Rosie Zhao, Mujin Kwun, Itai Shapira, David Brandfonbrener, Lucas Janson, and Sham Kakade. SOAP: Improving and Stabilizing Shampoo using Adam. 2024. <https://arxiv.org/abs/2409.11321>.

AI Use. I prepared these lecture notes with the aid of AI assistance; I take full responsibility for all content in the notes.