

Which Way Is Down? First-Order Optimizers as a Choice of Norm

Mikkel Paltorp (mpasc@dtu.dk), 13 October 2026

Department of Applied Mathematics and Computer Science, Technical University of Denmark

We consider the general optimization problem

$$\min_{\theta \in \Theta} \mathcal{J}(\theta), \quad (1)$$

where Θ is a vector space and $\mathcal{J}$ is differentiable. First-order methods solve it iteratively: starting from θ_0 , they take steps $\theta_{t+1} = \theta_t + h_t$, where each step h_t is computed from the derivative $D\mathcal{J}(\theta_t)$, and possibly from earlier ones. This post is about how to choose h_t , and in particular how the choice of a norm on Θ affects it.

1. Derivatives and gradients

The derivative $D\mathcal{J}(\theta)$ maps a direction $h \in \Theta$ to a number, so it is an element of the *dual space* Θ^* of linear maps $\Theta \rightarrow \mathbb{R}$, and we write its value at h as the *pairing* $\langle D\mathcal{J}(\theta), h \rangle$. Both Θ^* and the pairing come with Θ itself, so nothing is chosen here. An inner product $\langle \cdot, \cdot \rangle_\Theta$ on Θ , on the other hand, is a choice, for example the Frobenius inner product. With an inner product chosen, the linearization reads

$$\mathcal{J}(\theta + h) \approx \mathcal{J}(\theta) + \langle D\mathcal{J}(\theta), h \rangle = \mathcal{J}(\theta) + \langle \nabla_\theta \mathcal{J}(\theta), h \rangle_\Theta. \quad (2)$$

The second form holds because every linear map $\Theta \rightarrow \mathbb{R}$ is $h \mapsto \langle k, h \rangle_\Theta$ for exactly one $k \in \Theta$, its Riesz representative,¹ and the gradient $\nabla_\theta \mathcal{J}(\theta)$ is the representative of $D\mathcal{J}(\theta)$. Unlike the derivative, the gradient lies in Θ , but it depends on the inner product: another inner product gives another gradient of the same function. Only the gradient needs an inner product. The derivative and the steps below need none.

Example 1 (Two inner products, two gradients) Let $\Theta = \mathbb{R}^2$ and $\mathcal{J}(\theta) = \theta_1^2 + \theta_2$, so that $\langle D\mathcal{J}(\theta), h \rangle = 2\theta_1 h_1 + h_2$.² Then

$$\begin{aligned} \langle h, k \rangle_\Theta = h_1 k_1 + h_2 k_2 &\Rightarrow \nabla_\theta \mathcal{J}(\theta) = (2\theta_1, 1), \\ \langle h, k \rangle_\Theta = h_1 k_1 + 4h_2 k_2 &\Rightarrow \nabla_\theta \mathcal{J}(\theta) = (2\theta_1, 1/4), \end{aligned} \quad (3)$$

since both give $\langle \nabla_\theta \mathcal{J}(\theta), h \rangle_\Theta = 2\theta_1 h_1 + h_2$.

2. Two surrogates

The linearization $\mathcal{J}(\theta) + \langle g, h \rangle$, with $g = D\mathcal{J}(\theta)$, is a good model of $\mathcal{J}(\theta + h)$ only for small h , and on its own it has no minimizer: unless $g = 0$, it decreases without bound along any direction with $\langle g, h \rangle < 0$. A step therefore needs a second ingredient that says how large h we allow. The length of $h \in \Theta$ is measured by a norm $\|\cdot\|$, which is something that we again have to choose. There are two natural ways to combine the norm with the linearization: as a constraint, or as a penalty,³

Contents

1. Derivatives and gradients
2. Two surrogates
3. The dual norm and the sharp operator
4. The effect of the chosen norm
 - 4.1. Norms from an inner product
 - 4.2. Norms not from inner products
5. Incorporating the past
 - 5.1. Momentum
 - 5.2. Adaptive norms
6. Summary
- Bibliography

¹ This is the Riesz representation theorem, as in *Batched Backpropagation as a Linear Solve*: $k \mapsto \langle k, \cdot \rangle_\Theta$ is linear and one-to-one, and Θ and Θ^* have the same dimension.

Example 1: Two inner products, two gradients

² This follows from $\mathcal{J}$ alone, as the directional derivative $d/(dt)\mathcal{J}(\theta + th)|_{t=0} = 2\theta_1 h_1 + h_2$, with no inner product involved. Writing the derivative as the array $(2\theta_1, 1)$ uses the coordinates of $\Theta = \mathbb{R}^2$, but the value of the pairing does not depend on them.

³ The norm here can be any norm on Θ , not just the one induced by the inner product on Θ .

$$\begin{aligned}
h_C &= \arg \min_{\|h\| \leq \eta} \langle g, h \rangle, \\
h_P &= \arg \min_{h \in \Theta} \langle g, h \rangle + \frac{\lambda}{2} \|h\|^2.
\end{aligned} \tag{4}$$

The first is a trust-region model: it trusts the linearization only within a radius η , which caps the length of the step. The second instead penalizes long steps, and as the penalty grows faster than the linearization can decrease, it always has a minimum.⁴

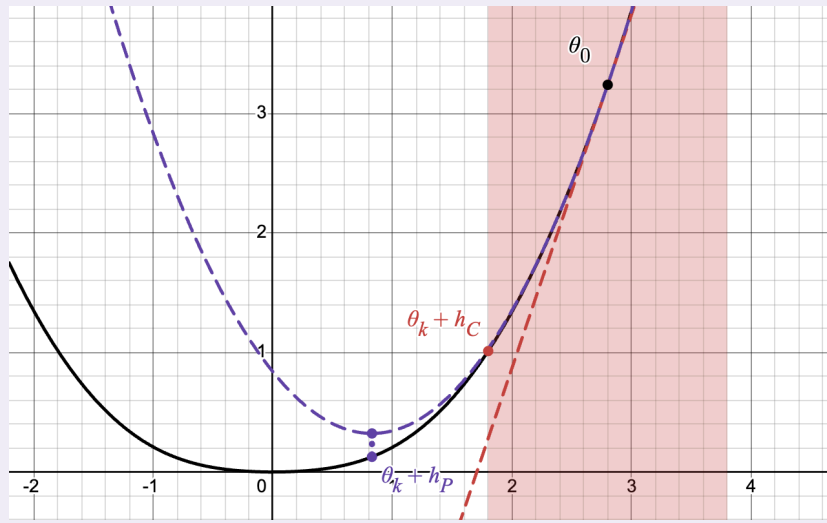
⁴ If $\mathcal{J}$ is L -smooth with respect to $\|\cdot\|$, meaning that $\mathcal{J}(\theta + h) \leq \mathcal{J}(\theta) + \langle g, h \rangle + \frac{L}{2} \|h\|^2$ for all h , then for $\lambda \geq L$ the second surrogate is an upper bound on $\mathcal{J}$, and h_P decreases $\mathcal{J}$ by at least the decrease of the surrogate. Convergence analyses use this; in practice λ is often chosen smaller.

Interactive figure 1: The two surrogates over many steps

Interactive figure 1 (The two surrogates over many steps) The graph shows both steps for

$$\mathcal{J}(\theta) = 3\theta^2/4 - 5/4 \log \cosh \theta, \tag{5}$$

with the norm $|h|$. In one dimension every norm is a multiple of $|h|$, so the choice of norm only matters from two dimensions on. The constrained step always has length η , since the linearization has no minimum inside the trust region, so close to the minimum it jumps back and forth over it. The penalized step shrinks with the derivative, but with $\lambda = L$ set by the largest curvature it crawls where $\mathcal{J}$ is flat. Constrained methods such as sign descent and Muon therefore let η decrease over the iterations, while adaptive norms, later in the post, take longer steps where $\mathcal{J}$ is flat.



Interactive version on the website →

At iteration k , the dashed line is the linearization at the constrained iterate θ_k , with the trust region $|h| \leq \eta$ shaded, and the dashed parabola is the quadratic model at the penalized iterate θ_k . The graph shows $k = 0$, with $\eta = 1$ and $\lambda = 3/2$. On the website, θ_0 , η , λ , and the iteration k can be changed.

3. The dual norm and the sharp operator

To write the solutions of the constrained and penalized problems in Equation 4 explicitly, we need two things: the dual norm, which measures the size of a derivative in Θ^* , and the sharp operator, which maps a derivative back to an element of

Θ . In both cases it is important to note that they depend on the choice of a norm on Θ , rather than the inner product on Θ .

Definition 1: Dual norm and sharp operator

Definition 1 (Dual norm and sharp operator) For a norm $\|\cdot\|$ on Θ , the dual norm of $g \in \Theta^*$ and the *sharp* of g , $g^\sharp \in \Theta$, are

$$\|g\|_* = \max_{\|h\| \leq 1} \langle g, h \rangle, \quad g^\sharp = -\arg \min_{h \in \Theta} \langle g, h \rangle + \frac{1}{2}\|h\|^2. \quad (6)$$

The dual norm is the slope of the linearization in its steepest direction; that is, a step of unit length with respect to the chosen norm changes the linearization by at most $\|g\|_*$, so that $|\langle g, h \rangle| \leq \|g\|_* \|h\|$ for all $h \in \Theta$. The *sharp operator*, the map $g \mapsto g^\sharp$, takes g back to Θ : $-g^\sharp$ is the penalized step for $\lambda = 1$, and its direction and length are set by the dual norm,⁵

$$\langle g, g^\sharp \rangle = \|g\|_*^2 \quad \text{and} \quad \|g^\sharp\| = \|g\|_*. \quad (7)$$

The minus sign is a convention. Like a gradient, $g^\sharp$ points where $\mathcal{J}$ increases, and a step subtracts it. When the minimizer is not unique, $g^\sharp$ is any one of them.

The sharp operator is the map from Θ^* back to Θ that a norm provides, while the Riesz map is the one that an inner product provides.⁶ The naming convention and notation $g^\sharp$ follow Kelner et al. [1].⁷ Both the constrained and the penalized steps can be written explicitly using the sharp operator. The constrained problem is bounded below by $\langle g, h \rangle \geq -\|g\|_* \|h\| \geq -\eta \|g\|_*$,⁸ which the step below attains, and writing $h = k/\lambda$ turns the penalized problem into the definition. For $g \neq 0$,

$$\begin{aligned} h_C &= \arg \min_{\|h\| \leq \eta} \langle g, h \rangle = -\eta \frac{g^\sharp}{\|g\|_*}, \\ h_P &= \frac{1}{\lambda} \arg \min_{k \in \Theta} \langle g, k \rangle + \frac{1}{2}\|k\|^2 = -\frac{1}{\lambda} g^\sharp. \end{aligned} \quad (8)$$

Here $g^\sharp/\|g\|_*$ is the unit vector, in the chosen norm, along $g^\sharp$, so h_C goes straight to the border of the trust region. Both steps point along $-g^\sharp$, which depends on the chosen norm, and differ only in length: η for h_C and $\|g\|_*/\lambda$ for h_P , so they coincide for $\lambda = \|g\|_*/\eta$.

Example 2 (The dual norm and the sharp operator in the plane) Let $\Theta = \mathbb{R}^2$ and $\langle g, h \rangle = 3h_1 + 4h_2$. For the Euclidean norm $\|h\|_2 = \sqrt{h_1^2 + h_2^2}$,

$$\begin{aligned} \|g\|_* &= \max_{\|h\|_2 \leq 1} 3h_1 + 4h_2 = 5, \\ g^\sharp &= -\arg \min_{h \in \mathbb{R}^2} 3h_1 + 4h_2 + \frac{1}{2}(h_1^2 + h_2^2) = (3, 4). \end{aligned} \quad (9)$$

By the Cauchy–Schwarz inequality, $3h_1 + 4h_2 \leq 5\|h\|_2$, with equality at $h = (3, 4)/5$, and in the problem for $g^\sharp$, setting the derivative $(3 + h_1, 4 + h_2)$ of the objective to zero gives the minimizer $h = (-3, -4)$, so $g^\sharp = (3, 4)$, the gradient for the Euclidean inner product. For the max-norm $\|h\|_\infty = \max(|h_1|, |h_2|)$,

⁵ Write $h = tu$ with $t \geq 0$ and $\|u\| = 1$. For a fixed t , the best u has $\langle g, u \rangle = -\|g\|_*$, which leaves $-t\|g\|_* + t^2/2$, smallest at $t = \|g\|_*$.

⁶ The two coincide exactly when the norm comes from the inner product, $\|h\|^2 = \langle h, h \rangle_\Theta$.

⁷ Bernstein and Newhouse [2] call $g^\sharp/\|g\|_*$ the duality map.

⁸ The first inequality is the bound $|\langle g, h \rangle| \leq \|g\|_* \|h\|$ from the definition of the dual norm. The second holds because the constraint gives $\|h\| \leq \eta$ and $\|g\|_* \geq 0$.

Example 2: The dual norm and the sharp operator in the plane

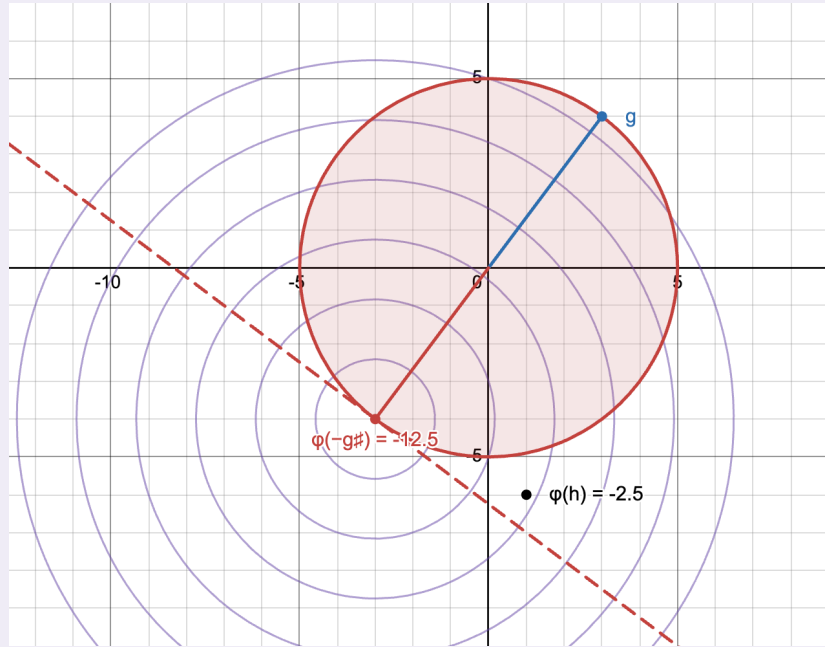
$$\begin{aligned}\|g\|_* &= \max_{\|h\|_\infty \leq 1} 3h_1 + 4h_2 = 7, \\ g^\sharp &= -\arg \min_{h \in \mathbb{R}^2} 3h_1 + 4h_2 + \frac{1}{2} \max(|h_1|, |h_2|)^2 = (7, 7).\end{aligned}\tag{10}$$

Both coordinates can be moved as far as allowed, so the maximum is $3 + 4$ at $h = (1, 1)$. For the minimum, fix $t = \max(|h_1|, |h_2|)$: the linear part is then smallest at $h = (-t, -t)$, which leaves $-7t + t^2/2$, smallest at $t = 7$. In both cases $\langle g, g^\sharp \rangle = \|g\|_*^2$ and $\|g^\sharp\| = \|g\|_*$. The same derivative gives different directions for different norms.⁹

⁹ The same argument gives $g^\sharp = \|g\|_1 \operatorname{sign}(g)$ for the max-norm, and this map is not linear: $(3, 4)^\sharp + (-1, 2)^\sharp = (7, 7) + (-3, 3) = (4, 10)$, while $(2, 6)^\sharp = (8, 8)$. So the max-norm is not the norm of any inner product, whose sharp operator, the Riesz map, is linear.

Interactive figure 2: The sharp operator

Interactive figure 2 (The sharp operator) The graph shows the definition for $g = (3, 4)$ of the example above and the ℓ_p -norms $\|h\|_p = (|h_1|^p + |h_2|^p)^{1/p}$, which give the Euclidean norm for $p = 2$ and the max-norm for $p = \infty$. Its contours are those of $\varphi(h) = \langle g, h \rangle + 1/2 \|h\|^2$, which is smallest at $-g^\sharp$. The dual norm shows in two places: $-g^\sharp$ lies on the border $\|h\| = \|g\|_*$ of the ball $\|h\| \leq \|g\|_*$, and the level set $\langle g, h \rangle + \|g\|_*^2 = 0$, a line in the plane, touches the ball there. Only $p = 2$ comes from an inner product. For every other p the sharp operator is nonlinear, and for p near 1 or ∞ it jumps between corners of the ball as g is dragged, which no linear map can do.



Interactive version on the website →

The ball is shaded and the line dashed, with $-g^\sharp$ drawn from the origin. The graph shows $p = 2$. On the website, g , p , and a trial point h can be changed, to check that no point has a smaller value $\varphi(h)$ than $-g^\sharp$.

4. The effect of the chosen norm

Once the norm is chosen, the step is fixed. To write the steps down, we need coordinates. Let $\Theta = \mathbb{R}^{m \times n}$, so that θ is a matrix, for example the weights of a layer

in a neural network. The derivative is then represented by the matrix G of partial derivatives $\partial \mathcal{J} / \partial \theta_{ab}$, with

$$\langle g, h \rangle = \langle G, h \rangle_F = \text{tr}(G^\top h), \quad g = D\mathcal{J}(\theta). \quad (11)$$

This uses the Frobenius inner product, which makes G the Riesz representative of g : the gradient for the Frobenius inner product.¹⁰ Some norms on Θ , such as the max-norm, only look at the entries of h , as if the matrix were one long vector of mn numbers. Others, such as the spectral norm, use that θ is a matrix, a linear map from inputs to outputs, which is what the weights of a layer are. For matrices, $\|\cdot\|_1$ and $\|\cdot\|_\infty$ act on the entries as for vectors, while $\|\cdot\|_2$ is the spectral norm, the largest singular value. Each norm then gives a dual norm and a sharp operator in terms of G , and with them a known method. Since both the constrained and the penalized steps follow from $\|g\|_*$ and $g^\sharp$, each example shows only the one that gives the known method. The other step is a rescaled version of it, for example normalized gradient descent in place of gradient descent. The table below collects the norms of this section.¹¹

Norm on Θ	$\ g\ _*$	$g^\sharp$	Method
Inner product	$\ \mathcal{A}^{-1}(G)\ _{\mathcal{A}}$	$\mathcal{A}^{-1}(G)$	Gradient descent
Max-norm	$\ G\ _1$	$\ G\ _1 \text{ sign}(G)$	Sign descent
Spectral	$\ G\ _{\text{nuc}}$	$\ G\ _{\text{nuc}} UV^\top$	Muon

The norms fall into two families. A norm from an inner product has a linear sharp operator, the Riesz map, and its step is a preconditioned gradient step. Every other norm has a nonlinear sharp operator, and its step is not the gradient for any fixed inner product, one that does not depend on g .

4.1. Norms from an inner product

Every inner product on Θ can be written as

$$\langle h, k \rangle_{\mathcal{A}} = \langle \mathcal{A}(h), k \rangle_F \quad (12)$$

where the linear map $\mathcal{A} : \Theta \rightarrow \Theta$ is symmetric and positive definite, i.e.,

$$\langle \mathcal{A}(h), k \rangle_F = \langle h, \mathcal{A}(k) \rangle_F \quad \text{and} \quad \langle \mathcal{A}(h), h \rangle_F > 0 \quad \text{for all } h \neq 0. \quad (13)$$

Since $\langle g, h \rangle = \langle G, h \rangle_F = \langle \mathcal{A}^{-1}(G), h \rangle_{\mathcal{A}}$,¹² the gradient for $\langle \cdot, \cdot \rangle_{\mathcal{A}}$ is $\nabla_{\mathcal{A}} \mathcal{J} = \mathcal{A}^{-1}(G)$.¹³ The inverse $\mathcal{A}^{-1}$ may look unusual, but it is the familiar preconditioner of scientific computing, where a problem is rescaled before it is solved: a gradient step with $\nabla_{\mathcal{A}} \mathcal{J}$ is plain gradient descent in the rescaled variables $\xi = \mathcal{A}^{1/2}(\theta)$.¹⁴ With $\mathcal{A}$ close to the curvature of $\mathcal{J}$, the rescaled problem is well conditioned, and the step comes close to Newton's method.

Example 3 (Preconditioned gradient descent) Let $\|h\|^2 = \langle h, h \rangle_{\mathcal{A}}$, so that $\langle g, h \rangle = \langle \nabla_{\mathcal{A}} \mathcal{J}, h \rangle_{\mathcal{A}}$ with the gradient $\nabla_{\mathcal{A}} \mathcal{J} = \mathcal{A}^{-1}(G)$. Then

¹⁰ This might seem like a limitation, but as shown in Section 4.1 every inner product on Θ can be written using the Frobenius inner product.

¹¹ The downloadable code computes $g^\sharp$ for the Frobenius norm, a diagonal inner product, the max-norm, and the spectral norm by solving the defining problem as a small conic program [3], and prints it next to the closed form.

Table 1: Three norms and the methods they give. For the first row the method is the penalized step $-g^\sharp/\lambda$, and for the last two it is the constrained step $-\eta g^\sharp/\|g\|_*$.

¹² We look for the $k \in \Theta$ with $\langle k, h \rangle_{\mathcal{A}} = \langle G, h \rangle_F$ for all h . By the definition of $\langle \cdot, \cdot \rangle_{\mathcal{A}}$, the left side is $\langle \mathcal{A}(k), h \rangle_F$, so $\langle \mathcal{A}(k) - G, h \rangle_F = 0$ for all h , which gives $\mathcal{A}(k) = G$ and $k = \mathcal{A}^{-1}(G)$. The inverse exists because $\mathcal{A}$ is positive definite.

¹³ In Example 1, the second inner product is $\mathcal{A} = \text{diag}(1, 4)$, and its gradient $(2\theta_1, 1/4)$ is $\mathcal{A}^{-1}$ applied to the first.

¹⁴ With $\xi = \mathcal{A}^{1/2}(\theta)$, the derivative of $\mathcal{J}(\mathcal{A}^{-1/2}(\xi))$ is represented by $\mathcal{A}^{-1/2}(G)$, so a gradient step in ξ changes ξ by $-\frac{1}{\lambda} \mathcal{A}^{-1/2}(G)$, and $\theta = \mathcal{A}^{-1/2}(\xi)$ by $-\frac{1}{\lambda} \mathcal{A}^{-1}(G)$.

Example 3: Preconditioned gradient descent

$$\begin{aligned}\|g\|_* &= \max_{\|h\|_{\mathcal{A}} \leq 1} \langle \nabla_{\mathcal{A}} \mathcal{J}, h \rangle_{\mathcal{A}} = \|\nabla_{\mathcal{A}} \mathcal{J}\|_{\mathcal{A}} = \langle G, \mathcal{A}^{-1}(G) \rangle_F^{1/2}, \\ g^\sharp &= -\arg \min_{h \in \Theta} \langle \nabla_{\mathcal{A}} \mathcal{J}, h \rangle_{\mathcal{A}} + \frac{1}{2} \|h\|_{\mathcal{A}}^2 = \nabla_{\mathcal{A}} \mathcal{J}.\end{aligned}\tag{14}$$

By the Cauchy–Schwarz inequality, $\langle \nabla_{\mathcal{A}} \mathcal{J}, h \rangle_{\mathcal{A}} \leq \|\nabla_{\mathcal{A}} \mathcal{J}\|_{\mathcal{A}} \|h\|_{\mathcal{A}}$, with equality at $h = \nabla_{\mathcal{A}} \mathcal{J} / \|\nabla_{\mathcal{A}} \mathcal{J}\|_{\mathcal{A}}$. In the problem for $g^\sharp$, completing the square gives

$$\langle \nabla_{\mathcal{A}} \mathcal{J}, h \rangle_{\mathcal{A}} + \frac{1}{2} \|h\|_{\mathcal{A}}^2 = \frac{1}{2} \|h + \nabla_{\mathcal{A}} \mathcal{J}\|_{\mathcal{A}}^2 - \frac{1}{2} \|\nabla_{\mathcal{A}} \mathcal{J}\|_{\mathcal{A}}^2,\tag{15}$$

which is smallest at $h = -\nabla_{\mathcal{A}} \mathcal{J}$, so $g^\sharp = \nabla_{\mathcal{A}} \mathcal{J}$. The sharp operator is the Riesz map, and the penalized step

$$h_P = -\frac{1}{\lambda} \mathcal{A}^{-1}(G)\tag{16}$$

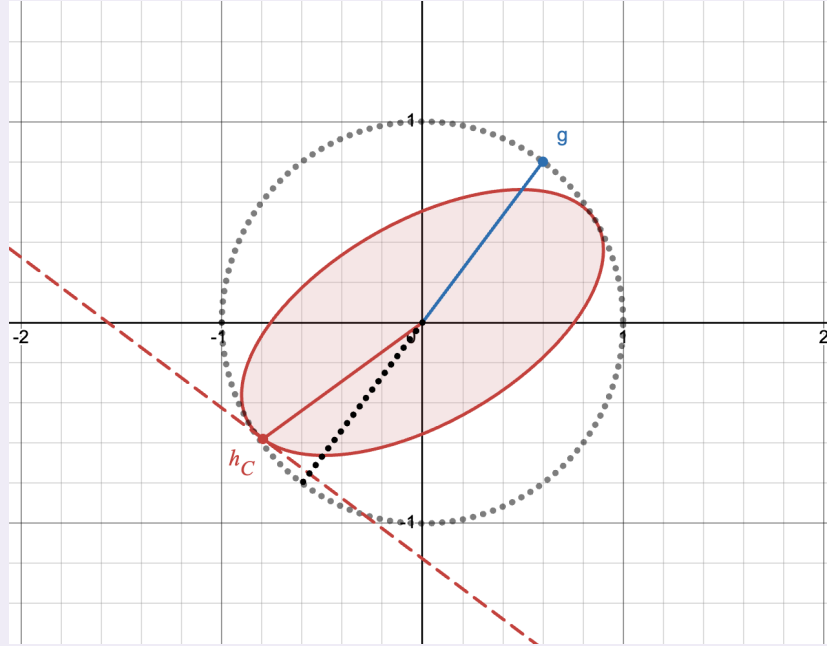
is preconditioned gradient descent with the preconditioner $\mathcal{A}^{-1}$, and for the Frobenius norm, $\mathcal{A} = \mathcal{I}$, plain gradient descent.

Interactive figure 3: A norm from an inner product

Interactive figure 3 (A norm from an inner product) The graph shows the constrained step for an inner product on $\mathbb{R}^2$, $\langle h, k \rangle_{\mathcal{A}} = h^\top A k$, where

$$A = R_\varphi \begin{bmatrix} 1 & 0 \\ 0 & \kappa \end{bmatrix} R_\varphi^\top, \quad R_\varphi = \begin{bmatrix} \cos \varphi & -\sin \varphi \\ \sin \varphi & \cos \varphi \end{bmatrix},\tag{17}$$

has the eigenvalues 1 and κ , with eigenvectors turned by the angle φ . The unit ball is an ellipse, and the step $h_C = -A^{-1}g/\|g\|_*$, for $\eta = 1$, is where the line $\langle g, h \rangle + \|g\|_* = 0$ touches it. Unlike the steps for $p = 1$ and $p = \infty$, it never jumps between corners, and unlike the steps of the other ℓ_p -norms, its direction depends linearly on g , since the sharp operator A^{-1} is linear.



Interactive version on the website →

The Euclidean unit circle and the Euclidean step $-g/\|g\|$ are dotted.

4.2. Norms not from inner products

Recall that the norm need not come from an inner product. Every method still minimizes the same linearization $\langle g, h \rangle$, with the derivative $g = D\mathcal{J}(\theta) \in \Theta^*$. What changes is the norm that limits the step, and with it the sharp operator, the map from Θ^* back to Θ . For a norm that does not come from an inner product, the sharp operator is nonlinear,¹⁵ so it is not the Riesz map of any fixed inner product, and the step is not a gradient for any inner product that does not depend on g . The max-norm is one such norm: as in the plane, $g^\sharp = \|G\|_1 \text{sign}(G)$, and its constrained step $h_C = -\eta \text{sign}(G)$ is sign descent. The example below shows the spectral norm, whose constrained step is Muon.

¹⁵ If $g \mapsto g^\sharp$ were linear, then $\|g\|_*^2 = \langle g, g^\sharp \rangle$ would be a quadratic form in g , so the dual norm would come from an inner product on Θ^* , and then so would the norm on Θ , its dual.

Example 4: The spectral norm: Muon

Example 4 (The spectral norm: Muon) The spectral norm of a matrix h is its largest singular value, $\|h\|_2 = \sigma_{\max}(h)$. Let $\theta \in \mathbb{R}^{m \times n}$ be the weights of a linear layer $Z = \theta X$. A step h changes the outputs by hX , and the spectral norm is the largest such change relative to the inputs,

$$\|hX\|_F \leq \|h\|_2 \|X\|_F \quad \text{for all } X. \quad (18)$$

As such, bounding the spectral norm effectively bounds the change of the outputs for every input. The dual norm and the sharp follow from the compact singular value decomposition $G = U\Sigma V^\top = \sum_k \sigma_k u_k v_k^\top$, which keeps only the nonzero singular values,

$$\begin{aligned} \|g\|_* &= \max_{\|h\|_2 \leq 1} \text{tr}(G^\top h) = \|G\|_{\text{nuc}}, \\ g^\sharp &= -\arg \min_{h \in \Theta} \text{tr}(G^\top h) + \frac{1}{2} \|h\|_2^2 = \|G\|_{\text{nuc}} UV^\top. \end{aligned} \quad (19)$$

By the cyclic property of the trace,

$$\begin{aligned} \text{tr}(G^\top h) &= \text{tr}(V\Sigma U^\top h) = \text{tr}(\Sigma U^\top hV) \\ &= \sum_k \sigma_k (U^\top hV)_{kk} \leq \sum_k \sigma_k = \|G\|_{\text{nuc}}, \end{aligned} \quad (20)$$

since each diagonal entry of $U^\top hV$ is at most its spectral norm, $\|U^\top hV\|_2 \leq \|h\|_2 \leq 1$. Equality holds at $h = UV^\top$, where $U^\top hV = I$. For the minimum, fix $t = \|h\|_2$: the linear part is then smallest at $h = -tUV^\top$, which leaves $-t\|G\|_{\text{nuc}} + t^2/2$, smallest at $t = \|G\|_{\text{nuc}}$. The nuclear norm, the sum of the singular values, is therefore not chosen but appears as the dual of the spectral norm. The constrained step

$$h_C = -\eta UV^\top \quad (21)$$

is minus the matrix with the singular vectors of G and all its nonzero singular values set to η , while the zero ones stay zero. It is the update of Muon [4], applied to an averaged derivative as in the next section. In a network, the constraint is applied to each block.¹⁶ In practice, Muon computes $UV^\top$ with a few matrix products by a Newton–Schulz iteration, whose polynomials are improved by The Polar Express Algorithm¹⁷, and uses $UV^\top$ only for the weight matrices of the hidden layers. Avoiding the SVD is what makes Muon practical for large networks: an SVD runs

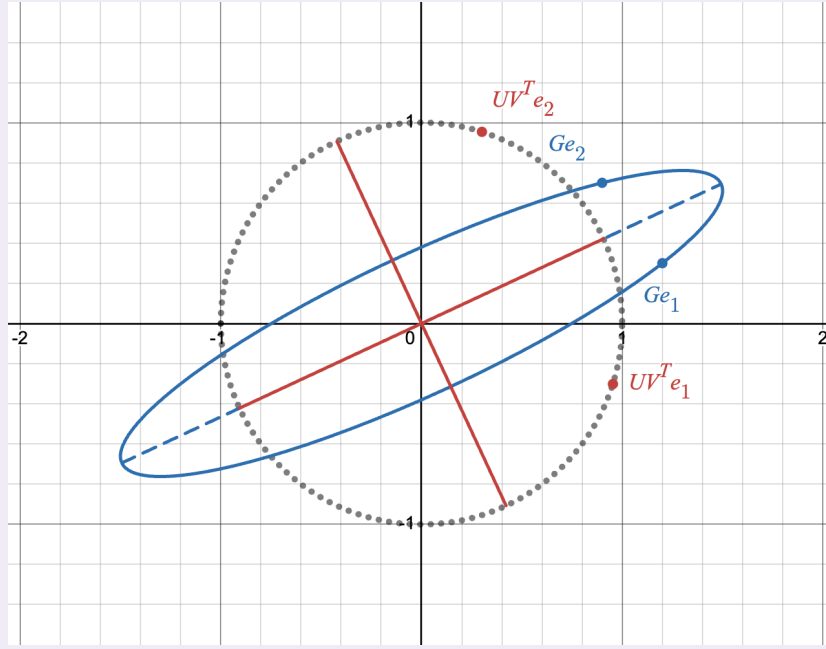
¹⁶ That is, the norm is $\max_i \|h_i\|_2$, as in Bernstein and Newhouse [2]. Its dual norm is $\sum_i \|G_i\|_{\text{nuc}}$, and each block takes the step $-\eta U_i V_i^\top$.

¹⁷ Earlier post: *The Polar Express Algorithm*. mipals.github.io/blog/2026/01-03-polar-express/

poorly on GPUs and needs high precision, while matrix products are fast even in low precision.

Interactive figure 4: The Muon step

Interactive figure 4 (The Muon step) Unlike the graphs above, this plane is not Θ , which here is four-dimensional: it holds the vectors a 2×2 block acts on, and each matrix is shown through its image of the unit circle. The block G maps the unit circle to an ellipse with the semi-axes $\sigma_1 u_1$ and $\sigma_2 u_2$. The matrix $UV^\top$ keeps the singular vectors and sets both singular values to 1: it maps the circle to itself, and its columns $UV^\top e_1$ and $UV^\top e_2$, which take the place of Ge_1 and Ge_2 , are orthonormal.



Interactive version on the website →

5. Incorporating the past

The surrogates above have no memory of the past. The linearization is created using only the current iterate θ_t and its derivative $g_t = D\mathcal{J}(\theta_t)$, and the norm is fixed. Methods with memory use past iterations in two ways: momentum methods change the linearization, and adaptive methods change the norm.

5.1. Momentum

The idea behind momentum is to build the linearization from past derivatives as well. The surrogates keep their form, with a chosen derivative $\hat{g}_t \in \Theta^*$ in place of g_t ,

$$\begin{aligned} h_C &= \arg \min_{\|h\| \leq \eta} \langle \hat{g}_t, h \rangle, \\ h_P &= \arg \min_{h \in \Theta} \langle \hat{g}_t, h \rangle + \frac{\lambda}{2} \|h\|^2. \end{aligned} \tag{22}$$

The derivative $\hat{g}_t$ enters before the sharp operator, in Θ^* . When the sharp operator is linear, as for a norm from an inner product, averaging the derivatives is the same as averaging the steps they give.¹⁸

Example 5 (Averaged linearizations: Adam and Muon) Average the past derivatives with weights $(1 - \beta)\beta^k$ for $\beta \in (0, 1)$, giving

$$\hat{g}_t = (1 - \beta) \sum_{k \geq 0} \beta^k g_{t-k}. \quad (23)$$

The pairing is linear in g for any norm, so $\langle \hat{g}_t, h \rangle$ is the same weighted average of the past linearizations $\langle g_{t-k}, h \rangle$. Both steps keep their form with $\hat{g}_t$ in place of g_t ,

$$h_C = -\eta \frac{\hat{g}_t^\#}{\|\hat{g}_t\|_*}, \quad h_P = -\frac{1}{\lambda} \hat{g}_t^\#. \quad (24)$$

The past derivatives were taken at earlier points, so as a model of $\mathcal{J}$ near θ_t this is worse than the linearization with g_t . But when each g_t is computed from a minibatch, $\hat{g}_t$ is a less noisy estimate of the expected derivative. In coordinates, $\hat{G} \leftarrow \beta \hat{G} + (1 - \beta)G$. Muon takes the step of Example 4 with $\hat{G}$ in place of G , and Adam does the same with the adaptive norm of Example 6.

5.2. Adaptive norms

Past information can also enter the norm. An adaptive method builds an inner product $\langle \cdot, \cdot \rangle_{\mathcal{A}_t}$ from past derivatives, so that the norm changes at every step. Each step is still the one of Example 3, with $\mathcal{A}_t$ in place of $\mathcal{A}$,

$$h_P = -\frac{1}{\lambda} \mathcal{A}_t^{-1}(G). \quad (25)$$

Storing a general $\mathcal{A}_t$ is out of the question for a network, so two cheap structures are used: a diagonal one, which stores as many numbers as θ , and a Kronecker-factored one.

Example 6 (A diagonal norm: AdaGrad and Adam) AdaGrad [5] sums the squared partial derivatives of all past iterations, and Adam [6] takes a moving average of them instead,

$$v \leftarrow v + G^2 \quad (\text{AdaGrad}), \quad v \leftarrow \beta_2 v + (1 - \beta_2)G^2 \quad (\text{Adam}), \quad (26)$$

where the square, like the square root below, acts entry by entry. So v has the same shape as θ , one number per entry. Both measure the step with the weighted norm

$$\|h\|_t^2 = \sum_a (\sqrt{v_a} + \varepsilon) h_a^2 = \langle \mathcal{A}_t(h), h \rangle_F = \langle h, h \rangle_{\mathcal{A}_t}, \quad (27)$$

$$\mathcal{A}_t(h) = (\sqrt{v} + \varepsilon) \odot h,$$

where a runs over all entries of θ , and $\odot$ multiplies two elements of Θ entry by entry, so that $\mathcal{A}_t$ multiplies each h_a by $\sqrt{v_a} + \varepsilon$. As $\|\cdot\|_t$ comes from an inner product, Example 3 gives

$$g^\# = \mathcal{A}_t^{-1}(G), \quad (28)$$

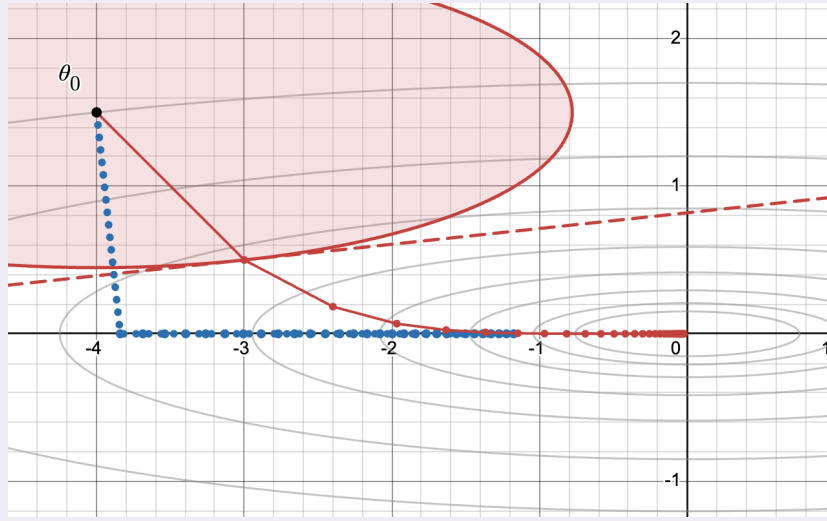
¹⁸ A linear sharp operator commutes with averaging, $(\sum_k w_k g_k)^\# = \sum_k w_k g_k^\#$. A nonlinear one does not: for Muon, the orthogonalized average of the derivatives is not the average of the orthogonalized derivatives.

Example 5: Averaged linearizations: Adam and Muon

Example 6: A diagonal norm: AdaGrad and Adam

which divides each G_a by $\sqrt{v_a} + \varepsilon$, and the penalized step with $\lambda = 1/\eta$ is $-\eta \mathcal{A}_t^{-1}(G)$. Adam also uses the averaged derivative $\hat{G}$ of Section 5.1 in place of G , so it carries memory in both the linearization and the norm, and its step, without bias correction,¹⁹ is $-\eta \mathcal{A}_t^{-1}(\hat{G})$.

Interactive figure 5 (The adaptive norm of AdaGrad) The graph shows AdaGrad on the quadratic $\mathcal{J}(\theta) = 1/2(\theta_1^2 + \kappa\theta_2^2)$, next to gradient descent with $\lambda = L = \kappa$. At each iterate, the ellipse is the ball of $\|\cdot\|_t$ through the next iterate, which the step reaches where a line of constant linearization touches the ball. Already the first derivative makes the ball narrow in the steep direction θ_2 , where the derivatives are large, and wide along the flat coordinate θ_1 . As v accumulates, the ball, and with it the step, shrinks. AdaGrad thereby takes long steps where gradient descent crawls.



Interactive version on the website →

Gradient descent is dotted and AdaGrad solid, with thirty steps each. The graph shows $k = 0$, with $\eta = 1$ and $\kappa = 25$. On the website, θ_0 , η , κ , and the iteration k can be changed.

Example 7 (A Kronecker-factored norm: K-FAC and Shampoo) Let $\theta \in \mathbb{R}^{m \times n}$ be the weights of a linear layer $Z = \theta X$ with inputs $X \in \mathbb{R}^{n \times b}$. A step h changes the outputs by hX . Measuring this change, weighted on the output side by a symmetric positive definite $L \in \mathbb{R}^{m \times m}$, gives²⁰

$$\|L^{1/2}hX\|_F^2 = \langle \mathcal{A}(h), h \rangle_F, \quad R = XX^\top, \quad \mathcal{A}(h) = LhR, \quad (29)$$

which is positive definite only if $XX^\top$ is. Unlike the spectral norm of Example 4, which is the worst case over all inputs, this norm measures the change on the given inputs. Its inverse, $\mathcal{A}^{-1}(G) = L^{-1}GR^{-1}$, needs only an $m \times m$ and an $n \times n$ matrix, not a general one of size $mn \times mn$.²¹ Since the inputs change from batch to batch, K-FAC [7] takes moving averages over the iterations of $XX^\top$ and of $\Delta\Delta^\top$, where Δ holds the derivatives of the loss with respect to the outputs Z ,²²

¹⁹ Adam starts both averages at zero, so after t steps their weights sum to $1 - \beta^t$, and early averages are too small. It divides $\hat{G}$ by $1 - \beta_1^t$ and v by $1 - \beta_2^t$, which scales the step by $\sqrt{1 - \beta_2^t}/(1 - \beta_1^t)$ when ε is ignored. The factor approaches 1 on the time scale of $1/(1 - \beta_2)$ steps.

Interactive figure 5: The adaptive norm of AdaGrad

Example 7: A Kronecker-factored norm: K-FAC and Shampoo

²⁰ In a network, each layer has its own L and R .

²¹ Stacking the columns of h into one long vector $\text{vec}(h)$ gives $\text{vec}(LhR) = (R \otimes L) \text{vec}(h)$, so $\mathcal{A}$ is the Kronecker product $R \otimes L$. This is why such norms and preconditioners are called *Kronecker-factored*, the KF in K-FAC.

²² In K-FAC, X has an extra row of ones for the bias, and Δ is computed with targets sampled from the network's own predictive distribution rather than taken from the data, so that $R \otimes L$ approximates a block of the Fisher information matrix.

$$R \leftarrow \rho R + (1 - \rho)XX^\top, \quad L \leftarrow \rho L + (1 - \rho)\Delta\Delta^\top, \quad (30)$$

and its basic step is $-\eta L^{-1}GR^{-1}$. K-FAC also adds damping to L and R , and takes the step $\alpha d_t + \mu h_{t-1}$ with $d_t = -L^{-1}GR^{-1}$, where α and μ are chosen at every step by minimizing a quadratic model. This is a momentum term whose coefficients change from step to step. Shampoo [8] instead sums the products of the derivative with itself, starting from $L = \varepsilon I$ and $R = \varepsilon I$,

$$L \leftarrow L + GG^\top, \quad R \leftarrow R + G^\top G, \quad (31)$$

and takes the step $-\eta L^{-1/4}GR^{-1/4}$, so its norm uses the fourth roots of L and R .

6. Summary

Table 2 summarizes the methods of the post by their norm and by where past information enters.

Method	Norm	$\hat{g}_t$	Norm memory
Gradient descent	Inner product	g_t	—
AdaGrad	Diagonal	g_t	Sum
Adam	Diagonal	Average	Average
K-FAC	Kronecker-factored	g_t	Average
Shampoo	Kronecker-factored	g_t	Sum
Sign descent	Max-norm	g_t	—
Muon	Spectral	Average	—

Table 2: The methods of the post as choices of the norm and of where past information enters: the linearization through $\hat{g}_t$, and the norm. K-FAC also uses the previous step, and implementations of Shampoo often add momentum as well.

Bibliography

- [1] J. A. Kelner, Y. T. Lee, L. Orecchia, and A. Sidford, “An Almost-Linear-Time Algorithm for Approximate Max Flow in Undirected Graphs, and its Multicommodity Generalizations,” in *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2014, pp. 217–226. doi: 10.1137/1.9781611973402.16.
- [2] J. Bernstein and L. Newhouse, “Modular Duality in Deep Learning.” [Online]. Available: <https://arxiv.org/abs/2410.21265>
- [3] P. J. Goulart and Y. Chen, “Clarabel: An interior-point solver for conic programs with quadratic objectives.” [Online]. Available: <https://arxiv.org/abs/2405.12762>
- [4] K. Jordan *et al.*, “Muon: An optimizer for hidden layers in neural networks.” [Online]. Available: <https://kellerjordan.github.io/posts/muon/>
- [5] J. Duchi, E. Hazan, and Y. Singer, “Adaptive Subgradient Methods for Online Learning and Stochastic Optimization,” *Journal of Machine Learning Research*, vol. 12, no. 61, pp. 2121–2159, 2011, [Online]. Available: <https://jmlr.org/papers/v12/duchi11a.html>

- [6] D. P. Kingma and J. Ba, “Adam: A Method for Stochastic Optimization,” in *International Conference on Learning Representations (ICLR)*, 2015. [Online]. Available: <https://arxiv.org/abs/1412.6980>
- [7] J. Martens and R. Grosse, “Optimizing Neural Networks with Kronecker-factored Approximate Curvature,” in *Proceedings of the 32nd International Conference on Machine Learning*, in Proceedings of Machine Learning Research, vol. 37. 2015, pp. 2408–2417. [Online]. Available: <https://proceedings.mlr.press/v37/martens15.html>
- [8] V. Gupta, T. Koren, and Y. Singer, “Shampoo: Preconditioned Stochastic Tensor Optimization,” in *Proceedings of the 35th International Conference on Machine Learning*, in Proceedings of Machine Learning Research, vol. 80. 2018, pp. 1842–1850. [Online]. Available: <https://proceedings.mlr.press/v80/gupta18a.html>