Recall: When we analyzed GD and SGD for strongly convex objectives, the convergence **rate** depended on the **condition** number $\kappa = L/\mu$.

For example, the noise ball for SGD w/ constant step size was determined by

$$\mathbb{E}[f(\omega_t) - f^*] \leq (1-\alpha\mu)^T (f(\omega_0) - f^*) + \frac{\alpha\sigma^2 L}{2\mu}$$

$$= (1-\alpha\mu)^T (f(\omega_0) - f^*) + \frac{\alpha\sigma^2 \kappa}{2}$$

and the convergence rate for descent w/ constant step size was

$$f(\omega_T) - f^* \leq \exp\left(-\frac{\mu T}{2}\right)(f(\omega_0) - f^*)$$

$$= \exp\left(-\frac{T}{\kappa}\right)(f(\omega_0) - f^*)$$

Takeaway: When κ high, convergence slow!

How can we speed up when κ high?

Let's consider the **simplest** possible setting with a high condition number: a two-dimensional quadratic.

Specifically,

$$f(w) = f(w_1, w_2) = \frac{L}{2} w_1^2 + \frac{\mu}{2} w_2^2 = \frac{1}{2} \begin{bmatrix} w_1 \\ w_2 \end{bmatrix}^T \begin{bmatrix} L & 0 \\ 0 & \mu \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \end{bmatrix}$$

Then

$$\nabla f(w) = \begin{bmatrix} L w_1 \\ \mu w_2 \end{bmatrix}, \quad \nabla^2 f(w) = \begin{bmatrix} L & 0 \\ 0 & \mu \end{bmatrix}$$

Here, the optimum value occurs at $w^* = 0$, and the Hessian is constan.

Eigenvalues of $\nabla^2 f(w)$ are L, μ wher $L \geq \mu > 0$.
Thus f is strongly convex w/ strong convexity constant μ and its second derivative is bounded by L.

Gradient descent w/ constant learning rate on this problem has update rule

$$(w_{t+1})_1 = (w_t)_1 - \alpha L (w_t)_1$$
$$(w_{t+1})_2 = (w_t)_2 - \alpha \mu (w_t)_2$$

$$\Rightarrow (w_K)_1 = (1 - \alpha L)^K (w_0)_1$$
$$(w_K)_2 = (1 - \alpha \mu)^K (w_0)_2$$

and so

$$f(w_K) = \frac{L}{2} (1 - \alpha_2)^{2K} ((w_0)_1)^2 + \frac{\mu}{2} (1 - \alpha \mu)^{2K} ((w_0)_2)^2$$

Asymptotically, this will be dominated by whichever term grows slower. That is

$$f(\omega_k) = O\left(\max\left(|1-\alpha L|, |1-\alpha\mu|\right)^{2k}\right)$$

Looking at the inner maximum function of α, a curious effort emerges.

So while we would **want** to pick a smaller learning rate for the L term, and a larger rate for the μ term, we're forced to pick something in the middle.

Aside

optimal learning rate

$$\alpha = \frac{2}{L+\mu} \Rightarrow \max(|1-\alpha L|, |1-\alpha\mu|) = \frac{L-\mu}{L+\mu} = \frac{\kappa-1}{\kappa+1} = 1 - \frac{2}{\kappa+1}$$

That is, we'd like to set the step size larger for dimensions with less curvature, and smaller for dimensions with **more** curvature.

Can't do w/ GD or SGD b/c only one parameter.

Today's fix?

Momentum

Motivation: try to tell the difference b/t more and less curved directions using information already available in GD.

Idea: In 1-D case, if gradients are **reversing sign**, step size is too large, b/c we're overshooting the optimum. Conversely, if the gradients are staying in the same direction, then the step size is too small.

Polyak Momentum Step: Adds an extra momentum term to gradient descent.

$$w_{t+1} = w_t - \alpha \nabla f(w_t) + \beta(w_t - w_{t-1})$$

Intuition: If current gradient is in the same direction as the previous step, move a little further in the same direction. If it's in the opposite direction, move a little less far.

Analysis of Polyak Momentum

Let's observe the case of one dimensional quadratics.

$$f(w) = \frac{\lambda}{2} w^2$$

Using this, Polyak update step looks like

$$w_{t+1} = w_t - \alpha\lambda w_t + \beta(w_t - w_{t-1})$$

$$= (1+\beta-\alpha\lambda)w_t - \beta w_{t-1}$$

Write in terms of matrix multiplication as

$$\begin{bmatrix} w_{t+1} \\ w_t \end{bmatrix} = \begin{bmatrix} 1+\beta-\alpha\lambda & -\beta \\ 1 & 0 \end{bmatrix} \begin{bmatrix} w_t \\ w_{t-1} \end{bmatrix}$$

This is called the **companion** matrix of the process.
Call this matrix M.
By induction, we get

$$\begin{bmatrix} w_{t+1} \\ w_t \end{bmatrix} = M \begin{bmatrix} w_t \\ w_{t-1} \end{bmatrix} = M^t \begin{bmatrix} w_1 \\ w_0 \end{bmatrix}$$

Assume M diagonalizable. (basis of eigenvectors $\Rightarrow M = QDQ^{-1}$)
then

$$\begin{bmatrix} w_{t+1} \\ w_t \end{bmatrix} = (QDQ^{-1})^t \begin{bmatrix} w_1 \\ w_0 \end{bmatrix} = QD\,Q^{-1} \begin{bmatrix} w_1 \\ w_0 \end{bmatrix}$$

Taking norms,

Induced norm: $\|A\| = \max_{\|u\|=1} \frac{\|Au\|}{\|u\|}$

$$\left\| \begin{bmatrix} w_{t+1} \\ w_t \end{bmatrix} \right\| = \left\| QD^tQ^{-1} \begin{bmatrix} w_1 \\ w_0 \end{bmatrix} \right\| \le \|Q\|\|D^t\|\|Q^{-1}\| \left\| \begin{bmatrix} w_1 \\ w_0 \end{bmatrix} \right\|$$

$$= \|Q\|\|Q^{-1}\| \left\| \begin{bmatrix} w_1 \\ w_0 \end{bmatrix} \right\| \cdot \max_{i \in \text{eigenvals}} \chi_i^t$$

So, this algorithm will converge at a linear rate determined by the maximum absolute value of an eigenvalue of M.

To understand how **Polyak** momentum converges, we need to understand the eigenvalues of M.

To do this, observe characteristic polynomial of M is

$$|xI - M| = \left| \begin{bmatrix} x - (1+\beta-\alpha\lambda) & \beta \\ -1 & x \end{bmatrix} \right|$$

$$= x^2 - (1 + \beta - \alpha\lambda)x + \beta$$

So, the eigenvalues of M are

$$x = \frac{(1 + \beta - \alpha\lambda) \pm \sqrt{(1+\beta-\alpha\lambda)^2 - 4\beta}}{2}$$

Now we note that if

$$(1+\beta-\alpha\lambda)^2 - 4\beta < 0 \quad \longleftrightarrow \quad -1 \leq \frac{1+\beta-\alpha\lambda}{2\sqrt{\beta}} \leq 1$$

then eigenvalues are complex conjugates!

That is,

$$x_1^* = x_2 \quad \text{and} \quad |x_1|^2 = |x_2|^2 = x_1^* x_1 = x_1 x_2$$

But $x_1 x_2$ is the product of eigenvalues of M.

So

$$x_1 x_2 = \beta \quad \leftarrow \det(M)$$

So

$$|x_1| = |x_2| = \sqrt{\beta}$$

This means momentum converges at a rate of $\sqrt{\beta}$ in this case!

Or, equivalently w_t^2 will approach zero at a linear rate of β^t for ANY step size (as long as condition above satisfied). i.e $\|w_t\|^2 = O(\beta^t)$

What does this mean for multidimensional quadratics?

$$f(w) = w^T A w \quad \leftarrow A \text{ symmetric, PSD assumed}$$

$$\nabla f(w) = Aw$$

$$w_{t+1} = w_t - \alpha A w_t + \beta (w_t - w_{t-1})$$

If u is an eigenvector of A, $Au = \lambda u \rightarrow u^T A^T = \lambda u^T$

$$u^T w_{t+1} = u^T w_t - \alpha u^T A w_t + \beta (u^T w_t - u^T w_{t-1})$$

$$= u^T w_t - \alpha \lambda u^T w_t + \beta (u^T w_t - u^T w_{t-1})$$

Importantly, multidimensional quadratics will converge at the same rate β^t, EVEN for directions of different curvature!

This will happen as long as

$$-1 \leq \frac{1+\beta-\alpha\lambda}{2\sqrt{\beta}} \leq 1.$$

Since $\mu \leq \lambda \leq L$, this will hold iff

$$-1 = \frac{1+\beta-\alpha L}{2\sqrt{\beta}} \quad \text{and} \quad \frac{1+\beta-\alpha\mu}{2\sqrt{\beta}} = 1$$

And through solving for α, β we get

$$\alpha = \frac{2+2\beta}{L+\mu} \quad \text{and} \quad \sqrt{\beta} = 1 - \frac{2}{\sqrt{\kappa}+1}$$

$\sqrt{\kappa}$!

that is,

$$\|w_T\| = O\left(\left(1-\frac{2}{\sqrt{\kappa}+1}\right)^T\right)$$

$$= O\left(\exp\left(\frac{-2T}{\sqrt{\kappa}+1}\right)\right)$$

Momentum has $O(\delta)$ extra memory and computational cost.

How to set momentum?

- Typically, just set $\beta = 0.9$
- Hyperparameter optimization.

Nesterov Momentum Step: Slightly different then Polyak momentum — GUARANTEED to work for convex functions.

$$v_{t+1} = w_t - \alpha \nabla f(w_t)$$
$$w_{t+1} = v_{t+1} + \beta (v_{t+1} - v_t)$$

Main difference: seperate momentum state from the point we are calculating gradient at.

Momentum with SGD

$$v_{t+1} = \beta v_t - \alpha \nabla f_{i_t}(w_t)$$
$$w_{t+1} = w_t + v_{t+1}$$