

Lecture 6: Optimization I

Prof. Noah Golowich

Today we will analyze stochastic gradient descent and introduce adaptive optimizers.

1 Optimization setup

Training a model means choosing parameters $\theta \in \mathbb{R}^d$ to minimize an average loss. For a training example z , let $f(\theta, z)$ be its loss, and let $\mathcal{D}$ be the data distribution. Our objective is

$$\min_{\theta \in \mathcal{B}} F(\theta), \quad F(\theta) := \mathbb{E}_{z \sim \mathcal{D}}[f(\theta, z)], \quad (1)$$

where $\mathcal{B} \subseteq \mathbb{R}^d$ is the feasible set. The above generalizes empirical risk minimization on examples $z_1, \dots, z_n$; indeed, take $\mathcal{D}$ to be uniform on the training set, so that $F(\theta) = n^{-1} \sum_{i=1}^n f(\theta, z_i)$.

Stochastic gradients. Computing the full gradient at every step can be expensive. At iteration t , we instead draw a fresh batch $B_t = (z_{t,1}, \dots, z_{t,b})$ of b independent examples from $\mathcal{D}$ and set

$$f_t(\theta) := \frac{1}{b} \sum_{r=1}^b f(\theta, z_{t,r}), \quad g_t := \nabla f_t(\theta_t) = \frac{1}{b} \sum_{r=1}^b \nabla_{\theta} f(\theta_t, z_{t,r}). \quad (2)$$

Thus f_t and g_t change with the randomly chosen batch. Let $\mathcal{H}_{t-1}$ be the history before drawing B_t , including θ_t , and write $\mathbb{E}_t[\cdot] := \mathbb{E}[\cdot \mid \mathcal{H}_{t-1}]$. We have

$$\mathbb{E}_t[g_t] = \nabla F(\theta_t), \quad \mathbb{E}_t[\|g_t - \nabla F(\theta_t)\|_2^2] \leq \frac{\sigma_0^2}{b} =: \sigma^2, \quad (3)$$

provided each example gradient has variance at most σ_0^2 .

2 Convergence of stochastic gradient descent

Stochastic gradient descent (SGD) follows the gradient of the current batch loss:

$$\theta_{t+1} = \theta_t - \alpha_t g_t, \quad (4)$$

where $\alpha_t > 0$ is the step size. The proof uses smoothness to compare the progress along $-g_t$ with the error introduced by the random batch.

We say F is L -smooth on a convex region if it is differentiable there and $\|\nabla F(\theta) - \nabla F(\theta')\|_2 \leq L \|\theta - \theta'\|_2$ for all points in that region. Smoothness implies the *descent inequality*

$$F(\theta + s) \leq F(\theta) + \langle \nabla F(\theta), s \rangle + \frac{L}{2} \|s\|_2^2 \quad (5)$$

whenever the segment from θ to $\theta + s$ lies in the region. Indeed, subtracting the linear term and integrating along the segment gives

$$F(\theta + s) - F(\theta) - \langle \nabla F(\theta), s \rangle = \int_0^1 \langle \nabla F(\theta + us) - \nabla F(\theta), s \rangle du \leq \int_0^1 Lu \|s\|_2^2 du.$$

Theorem 2.1 (SGD finds an approximate stationary point). *Let θ_1 be fixed. Suppose the unprojected SGD iterates $\theta_1, \dots, \theta_{T+1}$ lie almost surely in a compact convex set $\mathcal{B}$, and F is L -smooth on a neighborhood of $\mathcal{B}$, with $L > 0$. Let $F_* := \min_{\theta \in \mathcal{B}} F(\theta)$ and $\Delta := F(\theta_1) - F_*$. Assume (3), and use a constant step size $\alpha_t = \alpha$ with $0 < \alpha \leq 1/L$. Then*

$$\frac{1}{T} \sum_{t=1}^T \mathbb{E}[\|\nabla F(\theta_t)\|_2^2] \leq \frac{2\Delta}{\alpha T} + L\alpha\sigma^2. \quad (6)$$

Equivalently, the left side is $\mathbb{E}[\|\nabla F(\theta_\tau)\|_2^2]$ for an independent uniform index $\tau \in \{1, \dots, T\}$.

Proof. Apply (5) with $s = -\alpha g_t$ and take conditional expectations:

$$\begin{aligned} \mathbb{E}_t[F(\theta_{t+1})] &\leq F(\theta_t) - \alpha \langle \nabla F(\theta_t), \mathbb{E}_t[g_t] \rangle + \frac{L\alpha^2}{2} \mathbb{E}_t[\|g_t\|_2^2] \\ &\leq F(\theta_t) - \alpha \|\nabla F(\theta_t)\|_2^2 + \frac{L\alpha^2}{2} (\|\nabla F(\theta_t)\|_2^2 + \sigma^2) \\ &\leq F(\theta_t) - \frac{\alpha}{2} \|\nabla F(\theta_t)\|_2^2 + \frac{L\alpha^2\sigma^2}{2}. \end{aligned}$$

The second line uses unbiasedness and the identity $\mathbb{E}_t[\|g_t\|_2^2] = \|\nabla F(\theta_t)\|_2^2 + \mathbb{E}_t[\|g_t - \nabla F(\theta_t)\|_2^2]$; the last uses $L\alpha \leq 1$. Taking full expectations and summing over $t = 1, \dots, T$ telescopes the objective values:

$$\frac{\alpha}{2} \sum_{t=1}^T \mathbb{E}[\|\nabla F(\theta_t)\|_2^2] \leq F(\theta_1) - \mathbb{E}[F(\theta_{T+1})] + \frac{TL\alpha^2\sigma^2}{2} \leq \Delta + \frac{TL\alpha^2\sigma^2}{2}.$$

Dividing by $\alpha T/2$ proves the bound. The random-iterate formulation follows by averaging over the independent uniform index τ . $\square$

For a known horizon T , the simple choice $\alpha = 1/(L\sqrt{T})$ yields

$$\mathbb{E}[\|\nabla F(\theta_\tau)\|_2^2] \leq \frac{2L\Delta + \sigma_0^2/b}{\sqrt{T}}. \quad (7)$$

Thus SGD approaches stationarity without assuming convexity. This is a bound on the *squared* gradient norm; a small gradient need not identify a global minimum. A fixed positive step size, independent of T , instead leaves the noise term $L\alpha\sigma^2$ in (6). With exact gradients, that term vanishes and $\alpha = 1/L$ gives a $2L\Delta/T$ bound.

3 Adaptive optimizers

Next, we discuss *adaptive optimizers*, which are motivated by the fact that a single step size can be too large for some coordinates and unnecessarily small for others. In this section, we take $\mathcal{B} = \mathbb{R}^d$. Adaptive methods maintain a gradient-based direction $m_t \in \mathbb{R}^d$ and a diagonal scaling matrix $V_t = \text{diag}(v_t)$, using the past gradients:

$$m_t = \phi_t(g_1, \dots, g_t), \quad V_t = \psi_t(g_1, \dots, g_t).$$

For positive definite V_t , the generic update is

$$\theta_{t+1} = \theta_t - \alpha_t V_t^{-1/2} m_t. \quad (8)$$

All vector squares, square roots, and divisions below act coordinatewise. In particular, the effective learning rate of coordinate i is $\alpha_t/\sqrt{v_{t,i}}$. Ordinary SGD has $m_t = g_t$ and $V_t = I$.

AdaGrad. For a numerical stabilizer $\epsilon > 0$, AdaGrad takes

$$\phi_t(g_1, \dots, g_t) = g_t, \quad \psi_t(g_1, \dots, g_t) = \frac{1}{t} \text{diag} \left(\epsilon \mathbf{1} + \sum_{j=1}^t g_j^2 \right), \quad (9)$$

where $\mathbf{1} \in \mathbb{R}^d$ is the all-ones vector. Thus it reduces the learning rate of a coordinate according to its accumulated squared gradients. Recursively, initialize $s_0 = 0$ and set

$$s_t = s_{t-1} + g_t^2, \quad m_t = g_t, \quad v_t = \frac{s_t + \epsilon \mathbf{1}}{t}, \quad \alpha_t = \frac{\alpha}{\sqrt{t}}. \quad (10)$$

The factors of $\sqrt{t}$ cancel, giving

$$\theta_{t+1,i} = \theta_{t,i} - \frac{\alpha g_{t,i}}{\sqrt{\epsilon + s_{t,i}}} = \theta_{t,i} - \frac{\alpha g_{t,i}}{\sqrt{\epsilon + \sum_{j=1}^t g_{j,i}^2}}. \quad (11)$$

Thus the average convention (10) is equivalent to a cumulative-sum convention with $V_t = \text{diag}(s_t + \epsilon \mathbf{1})$ and constant scalar step size α .

Intuition for adaptivity. The square root of a mean squared gradient estimates its typical magnitude. Dividing by this magnitude prevents large-scale coordinates from dominating the update. For example, ignoring ϵ , multiplying every historical gradient in coordinate i by $c > 0$ multiplies its numerator by c and its denominator by c , leaving the AdaGrad update unchanged. A coordinate that is rarely active accumulates fewer squared gradients and retains a larger learning rate. A limitation is that the cumulative denominator never decreases, even when old gradients no longer reflect the current optimization problem.

Adam. For decay parameters $\beta_1, \beta_2 \in [0, 1)$, Adam takes

$$\begin{aligned} \phi_t(g_1, \dots, g_t) &= (1 - \beta_1) \sum_{j=1}^t \beta_1^{t-j} g_j, \\ \psi_t(g_1, \dots, g_t) &= \text{diag} \left(\epsilon \mathbf{1} + (1 - \beta_2) \sum_{j=1}^t \beta_2^{t-j} g_j^2 \right). \end{aligned} \quad (12)$$

These are exponentially moving averages (EMAs) of gradients and squared gradients, with ϵ added to stabilize the denominator. Recursively, initialize $m_0 = v_0 = 0$ and set

$$\begin{aligned} m_t &= \beta_1 m_{t-1} + (1 - \beta_1) g_t, \\ v_t &= \beta_2 v_{t-1} + (1 - \beta_2) g_t^2, \\ \theta_{t+1,i} &= \theta_{t,i} - \alpha_t \frac{m_{t,i}}{\sqrt{v_{t,i} + \epsilon}}. \end{aligned} \quad (13)$$

This corresponds to $V_t = \text{diag}(v_t + \epsilon \mathbf{1})$ in (8). For clarity, these are the updates without bias correction. Bias correction replaces m_t, v_t by $m_t/(1 - \beta_1^t), v_t/(1 - \beta_2^t)$, respectively, to compensate for initialization at zero. We place ϵ inside the square root throughout; the convention $\sqrt{v_{t,i}} + \epsilon$ is a different stabilization rule.

Remark 3.1 (Standard hyperparameters). Common Adam defaults are $\beta_1 = 0.9$, $\beta_2 = 0.999$, and $\epsilon = 10^{-8}$, with ϵ added *outside* the square root [1].

The momentum term m_t reinforces directions that persist across batches while averaging out some batch noise. For illustration, if independent scalar gradients have a fixed mean and variance ν^2 , their EMA with decay β has limiting variance $(1 - \beta)^2 \sum_{k \geq 0} \beta^{2k} \nu^2 = (1 - \beta) \nu^2 / (1 + \beta)$. Actual gradients change with θ_t , so longer averaging also introduces lag. The second EMA estimates recent gradient magnitudes and forgets old ones. Unlike AdaGrad, it can increase a coordinate's effective learning rate when recent gradients become smaller.

3.1 A regret bound for AdaGrad

We now analyze AdaGrad for differentiable convex losses on $\mathbb{R}^d$, measuring its cumulative performance against a fixed parameter vector.

Online convex optimization and regret. At round t , we choose $\theta_t \in \mathbb{R}^d$ before observing a differentiable convex loss $f_t : \mathbb{R}^d \rightarrow \mathbb{R}$, incur loss $f_t(\theta_t)$, and compute $g_t = \nabla f_t(\theta_t)$. The losses need not be independent or randomly sampled. For a fixed comparator $\theta^* \in \mathbb{R}^d$, define the *regret*

$$R_T(\theta^*) := \sum_{t=1}^T [f_t(\theta_t) - f_t(\theta^*)]. \quad (14)$$

A sublinear regret bound gives an upper bound on excess loss per round tending to zero.

Comparison with gradient descent. For online gradient descent $\theta_{t+1} = \theta_t - \alpha g_t$ with a constant scalar step size, let $D_2 > 0$ bound $\max_{1 \leq t \leq T} \|\theta_t - \theta^*\|_2$. Writing $S_T := \sum_{t=1}^T \|g_t\|_2^2$, we can use the regret upper bound

$$R_T(\theta^*) \leq \frac{D_2^2}{\alpha} + \frac{\alpha}{2} S_T.$$

For $S_T > 0$, the choice minimizing this upper bound is

$$\alpha = \frac{\sqrt{2} D_2}{\sqrt{S_T}} = \frac{\sqrt{2} D_2}{(\sum_{t=1}^T \|g_t\|_2^2)^{1/2}}.$$

This is a choice in hindsight, since S_T depends on the gradients observed along the gradient-descent trajectory. Substitution gives the comparison bound [3, Eq. (6)]

$$R_T(\theta^*) \leq \sqrt{2} D_2 \left(\sum_{t=1}^T \|g_t\|_2^2 \right)^{1/2}. \quad (15)$$

The AdaGrad update. For the exact bound below, set $\epsilon = 0$ in (11). Starting from any $\theta_1 \in \mathbb{R}^d$ and $s_0 = 0$, use

$$s_t = s_{t-1} + g_t^2, \quad \theta_{t+1,i} = \theta_{t,i} - \frac{\alpha g_{t,i}}{\sqrt{s_{t,i}}}. \quad (16)$$

If $s_{t,i} = 0$, then $g_{t,i} = 0$, and we leave that coordinate unchanged; throughout this subsection, ratios with this zero denominator are defined to be zero. The following bound uses a coordinatewise distance bound D_∞ , where $\|u\|_\infty := \max_i |u_i|$.

Theorem 3.2 (Duchi, Hazan, and Singer [3, Corollary 1]). *In the setting above, let $D_\infty > 0$ and run (16) with $\alpha = D_\infty / \sqrt{2}$. For every $\theta^* \in \mathbb{R}^d$ satisfying $\max_{1 \leq t \leq T} \|\theta_t - \theta^*\|_\infty \leq D_\infty$, we have*

$$R_T(\theta^*) \leq \sqrt{2d} D_\infty \left(\inf_{\substack{q \in (0, \infty)^d \\ \sum_{i=1}^d q_i \leq d}} \sum_{t=1}^T \sum_{i=1}^d \frac{g_{t,i}^2}{q_i} \right)^{1/2} = \sqrt{2} D_\infty \sum_{i=1}^d \sqrt{s_{T,i}}. \quad (17)$$

What adaptivity buys. The quantity $\sum_i g_{t,i}^2/q_i$ is the squared gradient norm weighted by $\text{diag}(q)^{-1}$. The infimum chooses a diagonal scaling in hindsight, subject to $\sum_i q_i \leq d$. AdaGrad attains the bound without knowing future gradients. Taking $q = \mathbf{1}$ gives the simpler bound $R_T(\theta^*) \leq \sqrt{2d} D_\infty(\sum_t \|g_t\|_2^2)^{1/2}$. If the gradients are supported on a fixed set of k coordinates, Cauchy–Schwarz instead gives $\sum_i \sqrt{s_{T,i}} \leq \sqrt{k}(\sum_t \|g_t\|_2^2)^{1/2}$. Thus the adaptive bound can replace the factor $\sqrt{d}$ by $\sqrt{k}$. In particular, bounded gradients give an $O(\sqrt{T})$ regret bound, or an $O(T^{-1/2})$ upper bound on average regret.

4 Hessian structure and transformer optimization

The Hessian describes how curvature varies with direction. This lets us identify a limitation of using the same learning rate for every parameter.

Why look at the Hessian? For twice continuously differentiable F , write $H(\theta) := \nabla^2 F(\theta)$. Suppose the Hessian is ρ -Lipschitz in operator norm near θ : $\|H(u) - H(v)\|_{\text{op}} \leq \rho \|u - v\|_2$, where $\|A\|_{\text{op}} := \sup_{\|w\|_2=1} \|Aw\|_2$. Then, for a displacement s staying in this neighborhood,

$$F(\theta + s) = F(\theta) + \langle \nabla F(\theta), s \rangle + \frac{1}{2} s^\top H(\theta) s + \mathcal{R}_3(s), \quad |\mathcal{R}_3(s)| \leq \frac{\rho}{6} \|s\|_2^3. \quad (18)$$

For a convex quadratic with minimizer θ^* ,

$$Q(\theta) = Q(\theta^*) + \frac{1}{2}(\theta - \theta^*)^\top H(\theta - \theta^*), \quad H \succeq 0,$$

gradient descent gives $e_{t+1} = (I - \alpha H)e_t$, where $e_t := \theta_t - \theta^*$. Along an eigenvector of H with eigenvalue $\lambda > 0$, the error is multiplied by $1 - \alpha\lambda$. All positive-curvature directions contract exactly when

$$0 < \alpha < \frac{2}{\lambda_{\max}(H)} \quad \text{if } \lambda_{\max}(H) > 0. \quad (19)$$

Zero-eigenvalue directions are flat. In particular, the conservative choice $\alpha \leq 1/\lambda_{\max}(H)$ satisfies the smooth descent bound.

Parameter blocks. Partition θ into K groups $\theta^{(1)}, \dots, \theta^{(K)}$, and write $H_{k\ell} := \partial^2 F / (\partial \theta^{(k)} \partial \theta^{(\ell)})$. A group might correspond to the parameters of a single layer, or a query/key matrix. If cross-group terms of the Hessian are small, $H \approx \text{diag}(H_{11}, \dots, H_{KK})$. For an exactly block-diagonal positive semidefinite Hessian,

$$\lambda_{\max}(H) = \max_{k \in \{1, \dots, K\}} \lambda_{\max}(H_{kk}).$$

Consequently, a common learning rate must accommodate the largest block curvature. As a simple example, write $\theta = (a, b)$ and take $Q(\theta) = \frac{1}{2}(Ma^2 + mb^2)$ with $M \gg m > 0$. This quadratic has two scalar blocks. Using $\alpha = 1/M$ eliminates the a error in one step, but multiplies the b error by $1 - m/M$ at every step. Separate learning rates $1/M$ and $1/m$ would eliminate both errors in one step. This illustrates why coordinate-dependent rates can help.

Why might the Hessian be nearly block diagonal? Consider a scalar-output network with one hidden layer,

$$h_\theta(u) = \sum_{i=1}^q a_i \varphi(w_i^\top u), \quad \ell(h, y) = \log(1 + e^{-yh}), \quad y \in \{-1, 1\},$$

where $u \in \mathbb{R}^p$ is the input, $w_i \in \mathbb{R}^p$ are hidden-neuron weights, $a_i \in \mathbb{R}$ are output weights, and φ is twice differentiable. Write $p_y := (1 + e^{-y h_\theta(u)})^{-1}$ for the predicted probability of the correct label. Then $\ell''(h_\theta(u), y) = p_y(1 - p_y)$. Treating w_i and w_j as distinct parameter blocks, the chain rule gives, for $i \neq j$,

$$\frac{\partial^2 \ell(h_\theta(u), y)}{\partial w_i \partial w_j} = p_y(1 - p_y) a_i a_j \varphi'(w_i^\top u) \varphi'(w_j^\top u) u u^\top. \quad (20)$$

The network output has no mixed derivative between these blocks, so only the displayed term remains. As training makes p_y close to 1, $p_y(1 - p_y) \rightarrow 0$, provided the other factors stay controlled. Averaging over examples then gives small cross-block entries.

Block heterogeneity in transformers. The *spectrum* of a block H_{kk} is its collection of eigenvalues; the curves in fig. 1 approximate their frequency distributions. These spectra differ substantially for BERT, whereas the sampled VGG16 blocks have similar spectra. This *block heterogeneity* motivates giving different parameter groups different learning rates; it is evidence for one explanation of Adam’s advantage [4].

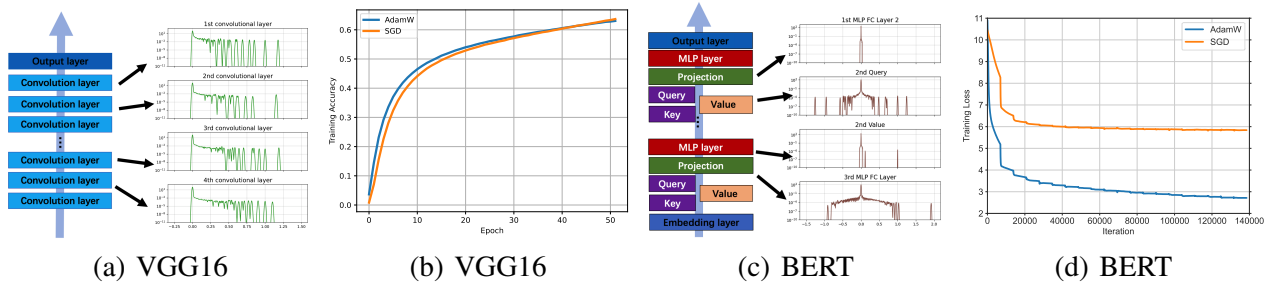


Figure 1: Reproduced from [4, Figure 3]. Four sampled block spectra at initialization and optimizer comparisons for VGG16 and BERT. Spectral curves use a logarithmic density axis and normalization by their tenth-largest eigenvalues.

References

- [1] Diederik P. Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization. ICLR, 2015. <https://arxiv.org/abs/1412.6980>.
- [2] Sashank J. Reddi, Satyen Kale, and Sanjiv Kumar. On the Convergence of Adam and Beyond. ICLR, 2018. <https://arxiv.org/abs/1904.09237>.
- [3] 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>.
- [4] Yushun Zhang, Congliang Chen, Tian Ding, Ziniu Li, Ruoyu Sun, and Zhi-Quan Luo. Why Transformers Need Adam: A Hessian Perspective. NeurIPS, 2024. <https://arxiv.org/abs/2402.16788>.

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