Chapter 10

Training of neural networks

Up to this point, we have discussed the representation and approximation of certain function classes using neural networks. The second pillar of deep learning concerns the question of how to fit a neural network to given data, i.e., having fixed an architecture, how to find suitable weights and biases. This task amounts to minimizing a so-called **objective function** such as the empirical risk $\hat{\mathcal{R}}_S$ in (1.2.3). Throughout this chapter we denote the objective function by

$$f : \mathbb{R}^n \to \mathbb{R},$$

and interpret it as a function of all neural network weights and biases collected in a vector in $\mathbb{R}^n$. The goal is to (approximately) determine a **minimizer**, i.e., some $\boldsymbol{w}_* \in \mathbb{R}^n$ satisfying

$$f(\boldsymbol{w}_*) \leq f(\boldsymbol{w}) \qquad \text{for all } \boldsymbol{w} \in \mathbb{R}^n.$$

Standard approaches include, in particular, variants of (stochastic) gradient descent. These are the topic of this chapter, in which we present basic ideas and results in convex optimization using gradient-based methods.

10.1 Gradient descent

The general idea of gradient descent is to start with some $\boldsymbol{w}_0 \in \mathbb{R}^n$, and then apply sequential updates by moving in the direction of *steepest descent* of the objective function. Assume for the moment that $f \in C^2(\mathbb{R}^n)$, and denote the kth iterate by $\boldsymbol{w}_k$. Then

$$f(\boldsymbol{w}_k + \boldsymbol{v}) = f(\boldsymbol{w}_k) + \boldsymbol{v}^\top \nabla f(\boldsymbol{w}_k) + O(\|\boldsymbol{v}\|^2) \qquad \text{for } \|\boldsymbol{v}\|^2 \to 0. \tag{10.1.1}$$

This shows that the change in f around $\boldsymbol{w}_k$ is locally described by the gradient $\nabla f(\boldsymbol{w}_k)$. For small $\boldsymbol{v}$ the contribution of the second order term is negligible, and the direction $\boldsymbol{v}$ along which the decrease of the risk is maximized equals the negative gradient $-\nabla f(\boldsymbol{w}_k)$. Thus, $-\nabla f(\boldsymbol{w}_k)$ is also called the direction of steepest descent. This leads to an update of the form

$$\boldsymbol{w}_{k+1} := \boldsymbol{w}_k - h_k \nabla f(\boldsymbol{w}_k), \tag{10.1.2}$$

where $h_k > 0$ is referred to as the **step size** or **learning rate**. We refer to this iterative algorithm as **gradient descent**.

In practice tuning the learning rate can be a subtle issue as it should strike a balance between the following dissenting requirements:

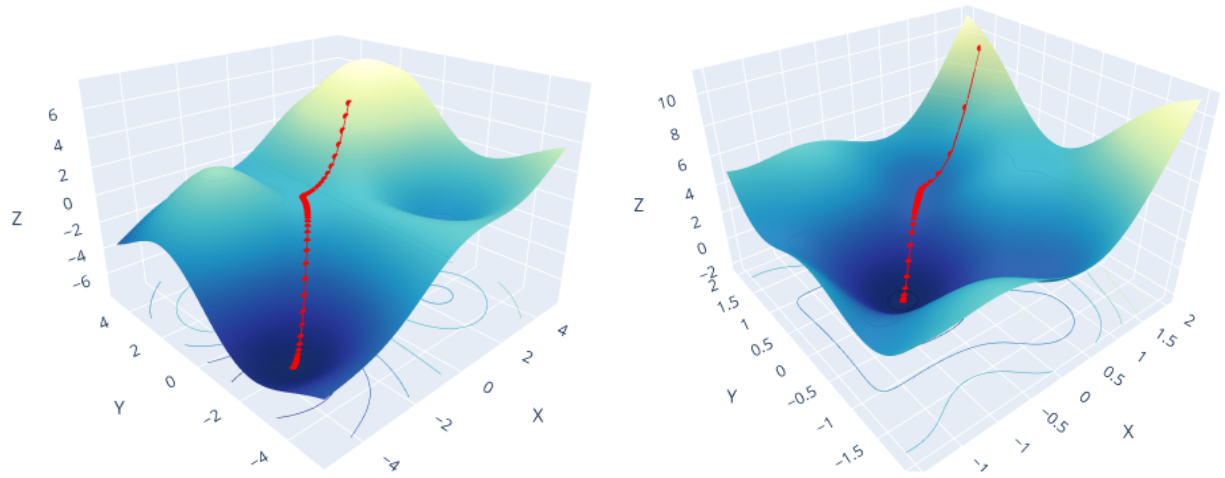

Figure 10.1: Two examples of gradient descent as defined in (10.1.2). The red points represent the $\boldsymbol{w}_k$.

(i) h_k needs to be sufficiently small so that with $\boldsymbol{v} = -h_k \nabla f(\boldsymbol{w}_k)$, the second-order term in (10.1.1) is not dominating. This ensures that the update (10.1.2) decreases the objective function.

(ii) h_k should be large enough to ensure significant decrease of the objective function, which facilitates faster convergence of the algorithm.

A learning rate that is too high might overshoot the minimum, while a rate that is too low results in slow convergence. Common strategies include, in particular, constant learning rates ($h_k = h$ for all $k \in \mathbb{N}_0$), learning rate schedules such as decaying learning rates ($h_k \searrow 0$ as $k \to \infty$), and adaptive methods. For adaptive methods the algorithm dynamically adjust h_k based on the values of $f(\boldsymbol{w}_j)$ or $\nabla f(\boldsymbol{w}_j)$ for $j \leq k$.

Remark 10.1. It is instructive to interpret (10.1.2) as an Euler discretization of the "gradient flow"

$$\boldsymbol{w}(0) = \boldsymbol{w}_0, \qquad \boldsymbol{w}'(t) = -\nabla f(\boldsymbol{w}(t)) \quad \text{for } t \in [0, \infty). \tag{10.1.3}$$

This ODE describes the movement of a particle $\boldsymbol{w}(t)$, whose velocity at time $t \geq 0$ equals $-\nabla f(\boldsymbol{w}(t))$—the vector of steepest descent. Note that

$$\frac{\mathrm{d} f(\boldsymbol{w}(t))}{\mathrm{d}t} = \big\langle \nabla f(\boldsymbol{w}(t)), \boldsymbol{w}'(t) \big\rangle = -\|\nabla f(\boldsymbol{w}(t))\|^2,$$

and thus the dynamics (10.1.3) necessarily decreases the value of the objective function along its path as long as $\nabla f(\boldsymbol{w}(t)) \neq 0$.

Throughout the rest of Section 10.1 we assume that $\boldsymbol{w}_0 \in \mathbb{R}^n$ is arbitrary, and the sequence $(\boldsymbol{w}_k)_{k \in \mathbb{N}_0}$ is generated by (10.1.2). We will analyze the convergence of this algorithm under suitable assumptions on f and the h_k. The proofs primarily follow the arguments in [157, Chapter 2]. We also refer to that book for a much more detailed discussion of gradient descent, and further reading on convex optimization.

10.1.1 L-smoothness

A key assumption to analyze convergence of (10.1.2) is Lipschitz continuity of ∇f.

Definition 10.2. Let $n \in \mathbb{N}$, and $L > 0$. The function $f : \mathbb{R}^n \to \mathbb{R}$ is called **L-smooth** if $f \in C^1(\mathbb{R}^n)$ and

$$\|\nabla f(\boldsymbol{w}) - \nabla f(\boldsymbol{v})\| \leq L\|\boldsymbol{w} - \boldsymbol{v}\| \qquad \text{for all } \boldsymbol{w}, \boldsymbol{v} \in \mathbb{R}^n.$$

For fixed $\boldsymbol{w}$, L-smoothness implies the linear growth bound

$$\|\nabla f(\boldsymbol{w} + \boldsymbol{v})\| \leq \|\nabla f(\boldsymbol{w})\| + L\|\boldsymbol{v}\|$$

for ∇f. Integrating the gradient along lines in $\mathbb{R}^n$ then shows that f is bounded from above by a quadratic function touching the graph of f at $\boldsymbol{w}$, as stated in the next lemma; also see Figure 10.2.

Lemma 10.3. *Let $n \in \mathbb{N}$ and $L > 0$. Let $f \colon \mathbb{R}^n \to \mathbb{R}$ be L-smooth. Then*

$$f(\boldsymbol{v}) \leq f(\boldsymbol{w}) + \langle \nabla f(\boldsymbol{w}), \boldsymbol{v} - \boldsymbol{w} \rangle + \frac{L}{2}\|\boldsymbol{w} - \boldsymbol{v}\|^2 \qquad \textit{for all } \boldsymbol{w}, \boldsymbol{v} \in \mathbb{R}^n. \tag{10.1.4}$$

Proof. We have for all $\boldsymbol{w}, \boldsymbol{v} \in \mathbb{R}^n$

$$\begin{aligned} f(\boldsymbol{v}) &= f(\boldsymbol{w}) + \int_0^1 \langle \nabla f(\boldsymbol{w} + t(\boldsymbol{v} - \boldsymbol{w})), \boldsymbol{v} - \boldsymbol{w} \rangle \, \mathrm{d}t \\ &= f(\boldsymbol{w}) + \langle \nabla f(\boldsymbol{w}), \boldsymbol{v} - \boldsymbol{w} \rangle + \int_0^1 \langle \nabla f(\boldsymbol{w} + t(\boldsymbol{v} - \boldsymbol{w})) - \nabla f(\boldsymbol{w}), \boldsymbol{v} - \boldsymbol{w} \rangle \, \mathrm{d}t. \end{aligned}$$

Thus

$$f(\boldsymbol{v}) - f(\boldsymbol{w}) - \langle \nabla f(\boldsymbol{w}), \boldsymbol{v} - \boldsymbol{w} \rangle \leq \int_0^1 L\|t(\boldsymbol{v} - \boldsymbol{w})\| \|\boldsymbol{v} - \boldsymbol{w}\| \, \mathrm{d}t = \frac{L}{2}\|\boldsymbol{v} - \boldsymbol{w}\|^2,$$

which shows (10.1.4). □

Remark 10.4. The argument in the proof of Lemma 10.3 also gives the lower bound

$$f(\boldsymbol{v}) \geq f(\boldsymbol{w}) + \langle \nabla f(\boldsymbol{w}), \boldsymbol{v} - \boldsymbol{w} \rangle - \frac{L}{2}\|\boldsymbol{w} - \boldsymbol{v}\|^2 \qquad \text{for all } \boldsymbol{w}, \boldsymbol{v} \in \mathbb{R}^n. \tag{10.1.5}$$

The previous lemma allows us to show a decay property for the gradient descent iterates. Specifically, the values of f necessarily decrease in each iteration as long as the step size h_k is small enough, and $\nabla f(\boldsymbol{w}_k) \neq 0$.

Lemma 10.5. *Let $n \in \mathbb{N}$ and $L > 0$. Let $f\colon \mathbb{R}^n \to \mathbb{R}$ be L-smooth. Further, let $(h_k)_{k=1}^{\infty}$ be positive numbers and let $(\boldsymbol{w}_k)_{k=0}^{\infty} \subseteq \mathbb{R}^n$ be defined by* (10.1.2)*.*

Then, for all $k \in \mathbb{N}$

$$f(\boldsymbol{w}_{k+1}) \leq f(\boldsymbol{w}_k) - \Big(h_k - \frac{Lh_k^2}{2}\Big)\|\nabla f(\boldsymbol{w}_k)\|^2. \tag{10.1.6}$$

Proof. Lemma 10.3 with $\boldsymbol{v} = \boldsymbol{w}_{k+1}$ and $\boldsymbol{w} = \boldsymbol{w}_k$ gives

$$f(\boldsymbol{w}_{k+1}) \leq f(\boldsymbol{w}_k) + \langle \nabla f(\boldsymbol{w}_k), -h_k \nabla f(\boldsymbol{w}_k)\rangle + \frac{L}{2}\|h_k \nabla f(\boldsymbol{w}_k)\|^2,$$

which corresponds to (10.1.6). □

Remark 10.6. The right-hand side in (10.1.6) is minimized for step size $h_k = 1/L$, in which case (10.1.6) reads

$$f(\boldsymbol{w}_{k+1}) \leq f(\boldsymbol{w}_k) - \frac{1}{2L}\|\nabla f(\boldsymbol{w}_k)\|^2.$$

Next, let us discuss the behavior of the gradients for constant step sizes.

Proposition 10.7. *Let $n \in \mathbb{N}$ and $L > 0$. Let $f\colon \mathbb{R}^n \to \mathbb{R}$ be L-smooth. Further, let $h_k = h \in (0, 2/L)$ for all $k \in \mathbb{N}$, and $(\boldsymbol{w}_k)_{k=0}^{\infty} \subseteq \mathbb{R}^n$ be defined by* (10.1.2)*.*

Then, for all $k \in \mathbb{N}$

$$\frac{1}{k+1}\sum_{j=0}^{k} \|\nabla f(\boldsymbol{w}_j)\|^2 \leq \frac{1}{k+1}\frac{2}{2h - Lh^2}(f(\boldsymbol{w}_0) - f(\boldsymbol{w}_{k+1})). \tag{10.1.7}$$

Proof. Set $c := h - (Lh^2)/2 = (2h - Lh^2)/2 > 0$. By (10.1.6) for $j \geq 0$

$$f(\boldsymbol{w}_j) - f(\boldsymbol{w}_{j+1}) \geq c\|\nabla f(\boldsymbol{w}_j)\|^2.$$

Hence

$$\sum_{j=0}^{k} \|\nabla f(\boldsymbol{w}_j)\|^2 \leq \frac{1}{c}\sum_{j=0}^{k} f(\boldsymbol{w}_j) - f(\boldsymbol{w}_{j+1}) = \frac{1}{c}\left(f(\boldsymbol{w}_0) - f(\boldsymbol{w}_{k+1})\right).$$

Dividing by $k+1$ concludes the proof. □

Suppose that f is bounded from below, i.e. $\inf_{\boldsymbol{w}\in\mathbb{R}^n} f(\boldsymbol{w}) > -\infty$. In this case, the right-hand side in (10.1.7) behaves like $O(k^{-1})$ as $k \to \infty$, and (10.1.7) implies

$$\min_{j=1,\dots,k} \|\nabla f(\boldsymbol{w}_j)\| = O(k^{-1/2}).$$

Thus, lower boundedness of the objective function together with L-smoothness already suffice to obtain some form of convergence of the gradients to 0. We emphasize that this does *not imply* convergence of $\boldsymbol{w}_k$ towards some $\boldsymbol{w}_*$ with $\nabla f(\boldsymbol{w}_*) = 0$ as the example $f(w) = \arctan(w)$, $w \in \mathbb{R}$, shows.

10.1.2 Convexity

While L-smoothness entails some interesting properties of gradient descent, it does not have any direct implications on the existence or uniqueness of minimizers. To show convergence of $f(\boldsymbol{w}_k)$ towards $\min_{\boldsymbol{w}} f(\boldsymbol{w})$ for $k \to \infty$ (assuming this minimum exists), we will assume that f is a convex function.

Definition 10.8. Let $n \in \mathbb{N}$. A function $f : \mathbb{R}^n \to \mathbb{R}$ is called **convex** if and only if

$$f(\lambda \boldsymbol{w} + (1-\lambda)\boldsymbol{v}) \leq \lambda f(\boldsymbol{w}) + (1-\lambda) f(\boldsymbol{v}), \tag{10.1.8}$$

for all $\boldsymbol{w}$, $\boldsymbol{v} \in \mathbb{R}^n$, $\lambda \in (0,1)$.

Let $n \in \mathbb{N}$. If $f \in C^1(\mathbb{R}^n)$, then f is convex if and only if

$$f(\boldsymbol{w}) + \langle \nabla f(\boldsymbol{w}), \boldsymbol{v} - \boldsymbol{w} \rangle \leq f(\boldsymbol{v}) \qquad \text{for all } \boldsymbol{w}, \boldsymbol{v} \in \mathbb{R}^n, \tag{10.1.9}$$

as shown in Exercise 10.27. Thus, $f \in C^1(\mathbb{R}^n)$ is convex if and only if the graph of f lies above each of its tangents, see Figure 10.2.

For convex f, a minimizer neither needs to exist (e.g., $f(w) = w$ for $w \in \mathbb{R}$) nor be unique (e.g., $f(\boldsymbol{w}) = 0$ for $\boldsymbol{w} \in \mathbb{R}^n$). However, if $\boldsymbol{w}_*$ and $\boldsymbol{v}_*$ are two minimizers, then every convex combination $\lambda \boldsymbol{w}_* + (1-\lambda)\boldsymbol{v}_*$, $\lambda \in [0,1]$, is also a minimizer due to (10.1.8). Thus, the set of all minimizers is convex. In particular, a convex objective function has either zero, one, or infinitely many minimizers. Moreover, if $f \in C^1(\mathbb{R}^n)$ then $\nabla f(\boldsymbol{w}) = 0$ implies

$$f(\boldsymbol{w}) = f(\boldsymbol{w}) + \langle \nabla f(\boldsymbol{w}), \boldsymbol{v} - \boldsymbol{w} \rangle \leq f(\boldsymbol{v}) \qquad \text{for all } \boldsymbol{v} \in \mathbb{R}^n.$$

Thus, $\boldsymbol{w}$ is a minimizer of f if and only if $\nabla f(\boldsymbol{w}) = 0$.

By Lemma 10.5, smallness of the step sizes and L-smoothness suffice to show a decay property for the objective function f. Under the additional assumption of convexity, we also get a decay property for the distance of $\boldsymbol{w}_k$ to *any* minimizer $\boldsymbol{w}_*$.

Lemma 10.9. *Let $n \in \mathbb{N}$ and $L > 0$. Let $f \colon \mathbb{R}^n \to \mathbb{R}$ be L-smooth and convex. Further, let $h_k \in (0, 2/L)$ for all $k \in \mathbb{N}_0$, and $(\boldsymbol{w}_k)_{k=0}^{\infty} \subseteq \mathbb{R}^n$ be defined by (10.1.2). Suppose that $\boldsymbol{w}_*$ is a minimizer of f.*

Then, for all $k \in \mathbb{N}_0$

$$\|\boldsymbol{w}_{k+1} - \boldsymbol{w}_*\|^2 \leq \|\boldsymbol{w}_k - \boldsymbol{w}_*\|^2 - h_k \cdot \Big(\frac{2}{L} - h_k\Big) \|\nabla f(\boldsymbol{w}_k)\|^2.$$

To prove the lemma, we will require the following inequality [157, Theorem 2.1.5].

Lemma 10.10. *Let $n \in \mathbb{N}$ and $L > 0$. Let $f\colon \mathbb{R}^n \to \mathbb{R}$ be L-smooth and convex.*
Then,

$$\frac{1}{L}\|\nabla f(\boldsymbol{w}) - \nabla f(\boldsymbol{v})\|^2 \leq \langle \nabla f(\boldsymbol{w}) - \nabla f(\boldsymbol{v}), \boldsymbol{w} - \boldsymbol{v}\rangle \qquad \text{for all } \boldsymbol{w}, \boldsymbol{v} \in \mathbb{R}^n.$$

Proof. Fix $\boldsymbol{w} \in \mathbb{R}^n$ and set $\Psi(\boldsymbol{u}) := f(\boldsymbol{u}) - \langle \nabla f(\boldsymbol{w}), \boldsymbol{u}\rangle$ for all $\boldsymbol{u} \in \mathbb{R}^n$. Then $\nabla\Psi(\boldsymbol{u}) = \nabla f(\boldsymbol{u}) - \nabla f(\boldsymbol{w})$ and thus Ψ is L-smooth. Moreover, convexity of f, specifically (10.1.9), yields $\Psi(\boldsymbol{u}) \geq f(\boldsymbol{w}) - \langle \nabla f(\boldsymbol{w}), \boldsymbol{w}\rangle = \Psi(\boldsymbol{w})$ for all $\boldsymbol{u} \in \mathbb{R}^n$, and thus $\boldsymbol{w}$ is a minimizer of Ψ. Using (10.1.4) on Ψ we get for every $\boldsymbol{v} \in \mathbb{R}^n$

$$\begin{aligned}
\Psi(\boldsymbol{w}) &= \min_{\boldsymbol{u}\in\mathbb{R}^n} \Psi(\boldsymbol{u}) \leq \min_{\boldsymbol{u}\in\mathbb{R}^n} \Big(\Psi(\boldsymbol{v}) + \langle \nabla\Psi(\boldsymbol{v}), \boldsymbol{u} - \boldsymbol{v}\rangle + \frac{L}{2}\|\boldsymbol{u} - \boldsymbol{v}\|^2\Big) \\
&= \min_{t\geq 0} \Psi(\boldsymbol{v}) - t\|\nabla\Psi(\boldsymbol{v})\|^2 + t^2\frac{L}{2}\|\nabla\Psi(\boldsymbol{v})\|^2 \\
&= \Psi(\boldsymbol{v}) - \frac{1}{2L}\|\nabla\Psi(\boldsymbol{v})\|^2
\end{aligned}$$

since the minimum of $t \mapsto t^2 L/2 - t$ is attained at $t = L^{-1}$. This implies

$$f(\boldsymbol{w}) - f(\boldsymbol{v}) + \frac{1}{2L}\|\nabla f(\boldsymbol{w}) - \nabla f(\boldsymbol{v})\|^2 \leq \langle \nabla f(\boldsymbol{w}), \boldsymbol{w} - \boldsymbol{v}\rangle .$$

Adding the same inequality with the roles of $\boldsymbol{w}$ and $\boldsymbol{v}$ switched gives the result. □

of Lemma 10.9. It holds

$$\|\boldsymbol{w}_{k+1} - \boldsymbol{w}_*\|^2 = \|\boldsymbol{w}_k - \boldsymbol{w}_*\|^2 - 2h_k \langle \nabla f(\boldsymbol{w}_k), \boldsymbol{w}_k - \boldsymbol{w}_*\rangle + h_k^2\|\nabla f(\boldsymbol{w}_k)\|^2.$$

Since $\nabla f(\boldsymbol{w}_*) = 0$, Lemma 10.10 gives

$$-\langle \nabla f(\boldsymbol{w}_k), \boldsymbol{w}_k - \boldsymbol{w}_*\rangle \leq -\frac{1}{L}\|\nabla f(\boldsymbol{w}_k)\|^2$$

which implies the claim. □

These preparations allow us to show that for constant step size $h < 2/L$, we obtain convergence of $f(\boldsymbol{w}_k)$ towards $f(\boldsymbol{w}_*)$ with rate $O(k^{-1})$, as stated in the next theorem which corresponds to [157, Theorem 2.1.14].

Theorem 10.11. *Let $n \in \mathbb{N}$ and $L > 0$. Let $f\colon \mathbb{R}^n \to \mathbb{R}$ be L-smooth and convex. Further, let $h_k = h \in (0, 2/L)$ for all $k \in \mathbb{N}_0$, and let $(\boldsymbol{w}_k)_{k=0}^{\infty} \subseteq \mathbb{R}^n$ be defined by* (10.1.2)*. Suppose that $\boldsymbol{w}_*$ is a minimizer of f.*
Then, $f(\boldsymbol{w}_k) - f(\boldsymbol{w}_*) = O(k^{-1})$ for $k \to \infty$, and for the specific choice $h = 1/L$

$$f(\boldsymbol{w}_k) - f(\boldsymbol{w}_*) \leq \frac{2L}{4+k}\|\boldsymbol{w}_0 - \boldsymbol{w}_*\|^2 \qquad \text{for all } k \in \mathbb{N}_0. \tag{10.1.10}$$

Proof. The case $\boldsymbol{w}_0 = \boldsymbol{w}_*$ is trivial and throughout we assume $\boldsymbol{w}_0 \neq \boldsymbol{w}_*$.

Step 1. Let $j \in \mathbb{N}_0$. Using convexity (10.1.9)

$$f(\boldsymbol{w}_j) - f(\boldsymbol{w}_*) \leq -\langle \nabla f(\boldsymbol{w}_j), \boldsymbol{w}_* - \boldsymbol{w}_j \rangle \leq \|\nabla f(\boldsymbol{w}_j)\| \|\boldsymbol{w}_* - \boldsymbol{w}_j\|. \tag{10.1.11}$$

By Lemma 10.9 and since $\boldsymbol{w}_0 \neq \boldsymbol{w}_*$ it holds $\|\boldsymbol{w}_* - \boldsymbol{w}_j\| \leq \|\boldsymbol{w}_* - \boldsymbol{w}_0\| \neq 0$, so that we obtain a lower bound on the gradient

$$\|\nabla f(\boldsymbol{w}_j)\|^2 \geq \frac{(f(\boldsymbol{w}_j) - f(\boldsymbol{w}_*))^2}{\|\boldsymbol{w}_* - \boldsymbol{w}_0\|^2}.$$

Lemma 10.5 then yields

$$\begin{aligned} f(\boldsymbol{w}_{j+1}) - f(\boldsymbol{w}_*) &\leq f(\boldsymbol{w}_j) - f(\boldsymbol{w}_*) - \Big(h - \frac{Lh^2}{2}\Big)\|\nabla f(\boldsymbol{w}_j)\|^2 \\ &\leq f(\boldsymbol{w}_j) - f(\boldsymbol{w}_*) - \Big(h - \frac{Lh^2}{2}\Big)\frac{(f(\boldsymbol{w}_j) - f(\boldsymbol{w}_*))^2}{\|\boldsymbol{w}_0 - \boldsymbol{w}_*\|^2}. \end{aligned}$$

With $e_j := f(\boldsymbol{w}_j) - f(\boldsymbol{w}_*)$ and $\omega := (h - Lh^2/2)/\|\boldsymbol{w}_0 - \boldsymbol{w}_*\|^2$ this reads

$$e_{j+1} \leq e_j - \omega e_j^2 = e_j \cdot (1 - \omega e_j), \tag{10.1.12}$$

which is valid for all $j \in \mathbb{N}_0$.

Step 2. By L-smoothness (10.1.4) and $\nabla f(\boldsymbol{w}_*) = 0$ it holds

$$f(\boldsymbol{w}_0) - f(\boldsymbol{w}_*) \leq \frac{L}{2}\|\boldsymbol{w}_0 - \boldsymbol{w}_*\|^2, \tag{10.1.13}$$

which implies (10.1.10) for $k = 0$. It remains to show the bound for $k \in \mathbb{N}$.

Fix $k \in \mathbb{N}$. We may assume $e_k > 0$, since otherwise (10.1.10) is trivial. Then $e_j > 0$ for all $j = 0, \dots, k-1$ since $e_j = 0$ implies $e_i = 0$ for all $i > j$, contradicting $e_k = 0$. Moreover, $\omega e_j < 1$ for all $j = 0, \dots, k-1$, since $\omega e_j \geq 1$ implies $e_{j+1} \leq 0$ by (10.1.12), contradicting $e_{j+1} > 0$.

Using that $1/(1-x) \geq 1 + x$ for all $x \in [0, 1)$, (10.1.12) thus gives

$$\frac{1}{e_{j+1}} \geq \frac{1}{e_j}(1 + \omega e_j) = \frac{1}{e_j} + \omega \qquad \text{for all } j = 0, \dots, k-1.$$

Hence

$$\frac{1}{e_k} - \frac{1}{e_0} = \sum_{j=0}^{k-1} \Big(\frac{1}{e_{j+1}} - \frac{1}{e_j}\Big) \geq k\omega$$

and

$$f(\boldsymbol{w}_k) - f(\boldsymbol{w}_*) = e_k \leq \frac{1}{\frac{1}{e_0} + k\omega} = \frac{1}{\frac{1}{f(\boldsymbol{w}_0) - f(\boldsymbol{w}_*)} + k\frac{(h - Lh^2/2)}{\|\boldsymbol{w}_0 - \boldsymbol{w}_*\|^2}}.$$

Using (10.1.13) we get

$$f(\boldsymbol{w}_k) - f(\boldsymbol{w}_*) \leq \frac{\|\boldsymbol{w}_0 - \boldsymbol{w}_*\|^2}{\frac{2}{L} + kh \cdot (1 - \frac{Lh}{2})} = O(k^{-1}). \tag{10.1.14}$$

Finally, (10.1.10) follows by plugging in $h = 1/L$. □

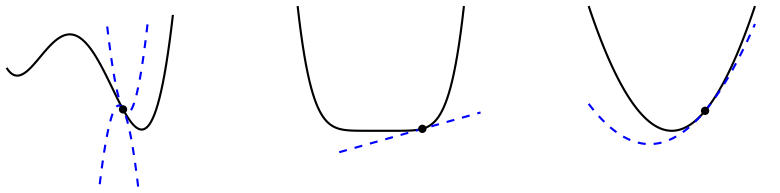

Figure 10.2: The graph of L-smooth functions lies between two quadratic functions at each point, see (10.1.4) and (10.1.5), the graph of convex functions lies above the tangent at each point, see (10.1.9), and the graph of μ-strongly convex functions lies above a quadratic function at each point, see (10.1.15).

Remark 10.12. The step size $h = 1/L$ is again such that the upper bound in (10.1.14) is minimized.

It is important to note that while under the assumptions of Theorem 10.11 it holds $f(\boldsymbol{w}_k) \to f(\boldsymbol{w}_*)$, in general it is not true that $\boldsymbol{w}_k \to \boldsymbol{w}_*$ as $k \to \infty$. To show the convergence of the $\boldsymbol{w}_k$, we need to introduce stronger assumptions that guarantee the existence of a unique minimizer.

10.1.3 Strong convexity

To obtain faster convergence and guarantee the existence of unique minimizers, we next introduce the notion of strong convexity. As the terminology suggests, strong convexity implies convexity; specifically, while convexity requires f to be lower bounded by the linearization around each point, strongly convex functions are lower bounded by the linearization plus a positive quadratic term.

Definition 10.13. Let $n \in \mathbb{N}$ and $\mu > 0$. A function $f \in C^1(\mathbb{R}^n)$ is called **μ-strongly convex** if

$$f(\boldsymbol{v}) \geq f(\boldsymbol{w}) + \langle \nabla f(\boldsymbol{w}), \boldsymbol{v} - \boldsymbol{w} \rangle + \frac{\mu}{2}\|\boldsymbol{v} - \boldsymbol{w}\|^2 \qquad \text{for all } \boldsymbol{w}, \boldsymbol{v} \in \mathbb{R}^n. \tag{10.1.15}$$

Note that (10.1.15) is the opposite of the bound (10.1.4) implied by L-smoothness. We depict the three notions of L-smoothness, convexity, and μ-strong convexity in Figure 10.2.

Every μ-strongly convex function has a unique minimizer. To see this note first that (10.1.15) implies f to be lower bounded by a convex quadratic function, so that there exists at least one minimizer $\boldsymbol{w}_*$, and $\nabla f(\boldsymbol{w}_*) = 0$. By (10.1.15) we then have $f(\boldsymbol{v}) > f(\boldsymbol{w}_*)$ for every $\boldsymbol{v} \neq \boldsymbol{w}_*$.

The next theorem shows that the gradient descent iterates converge linearly towards the unique minimizer for L-smooth and μ-strongly convex functions. Recall that a sequence e_k is said to **converge linearly** to 0, if and only if there exist constants $C > 0$ and $c \in [0, 1)$ such that

$$e_k \leq Cc^k \qquad \text{for all } k \in \mathbb{N}_0.$$

The constant c is also referred to as the **rate of convergence**. Before giving the statement, we first note that comparing (10.1.4) and (10.1.15) it necessarily holds $L \geq \mu$ and therefore $\kappa := L/\mu \geq 1$. This term is known as the **condition number** of f. It crucially influences the rate of convergence.

Theorem 10.14. *Let $n \in \mathbb{N}$ and $L \geq \mu > 0$. Let $f\colon \mathbb{R}^n \to \mathbb{R}$ be L-smooth and μ-strongly convex. Further, let $h_k = h \in (0, 1/L]$ for all $k \in \mathbb{N}_0$, let $(\boldsymbol{w}_k)_{k=0}^{\infty} \subseteq \mathbb{R}^n$ be defined by (10.1.2), and let $\boldsymbol{w}_*$ be the unique minimizer of f.*

Then, $f(\boldsymbol{w}_k) \to f(\boldsymbol{w}_*)$ and $\boldsymbol{w}_k \to \boldsymbol{w}_*$ converge linearly for $k \to \infty$. For the specific choice $h = 1/L$

$$\|\boldsymbol{w}_k - \boldsymbol{w}_*\|^2 \leq \Big(1 - \frac{\mu}{L}\Big)^k \|\boldsymbol{w}_0 - \boldsymbol{w}_*\|^2 \tag{10.1.16a}$$

$$f(\boldsymbol{w}_k) - f(\boldsymbol{w}_*) \leq \frac{L}{2}\Big(1 - \frac{\mu}{L}\Big)^k \|\boldsymbol{w}_0 - \boldsymbol{w}_*\|^2. \tag{10.1.16b}$$

Proof. It suffices to show (10.1.16a) since (10.1.16b) follows directly by Lemma 10.3 and because $\nabla f(\boldsymbol{w}_*) = 0$. The case $k = 0$ is trivial, so let $k \in \mathbb{N}$.

Expanding $\boldsymbol{w}_k = \boldsymbol{w}_{k-1} - h\nabla f(\boldsymbol{w}_{k-1})$ and using μ-strong convexity (10.1.15)

$$\begin{aligned}\|\boldsymbol{w}_k - \boldsymbol{w}_*\|^2 &= \|\boldsymbol{w}_{k-1} - \boldsymbol{w}_*\|^2 - 2h\,\langle \nabla f(\boldsymbol{w}_{k-1}), \boldsymbol{w}_{k-1} - \boldsymbol{w}_*\rangle + h^2\|\nabla f(\boldsymbol{w}_{k-1})\|^2 \\ &\leq (1 - \mu h)\|\boldsymbol{w}_{k-1} - \boldsymbol{w}_*\|^2 - 2h \cdot (f(\boldsymbol{w}_{k-1}) - f(\boldsymbol{w}_*)) + h^2\|\nabla f(\boldsymbol{w}_{k-1})\|^2.\end{aligned}$$

Moreover, the descent property in Lemma 10.5 gives

$$\begin{aligned}&- 2h \cdot (f(\boldsymbol{w}_{k-1}) - f(\boldsymbol{w}_*)) + h^2\|\nabla f(\boldsymbol{w}_{k-1})\|^2 \\ &\quad\leq -2h \cdot (f(\boldsymbol{w}_{k-1}) - f(\boldsymbol{w}_*)) + \frac{h^2}{h \cdot (1 - Lh/2)}(f(\boldsymbol{w}_{k-1}) - f(\boldsymbol{w}_k)).\end{aligned} \tag{10.1.17}$$

The descent property also implies $f(\boldsymbol{w}_{k-1}) - f(\boldsymbol{w}_*) \geq f(\boldsymbol{w}_{k-1}) - f(\boldsymbol{w}_k)$. Thus the right-hand side of (10.1.17) is less or equal to zero as long as $2h \geq h/(1 - Lh/2)$, which is equivalent to $h \leq 1/L$. Hence

$$\|\boldsymbol{w}_k - \boldsymbol{w}_*\|^2 \leq (1 - \mu h)\|\boldsymbol{w}_{k-1} - \boldsymbol{w}_*\|^2 \leq \cdots \leq (1 - \mu h)^k \|\boldsymbol{w}_0 - \boldsymbol{w}_*\|^2.$$

This concludes the proof. □

Remark 10.15. With a more refined argument, see [157, Theorem 2.1.15], the constraint on the step size can be relaxed to $h \leq 2/(\mu + L)$. For $h = 2/(\mu + L)$ one then obtains (10.1.16) with $1 - \mu/L = 1 - \kappa^{-1}$ replaced by

$$\Big(\frac{L/\mu - 1}{L/\mu + 1}\Big)^2 = \Big(\frac{\kappa - 1}{\kappa + 1}\Big)^2 \in [0, 1). \tag{10.1.18}$$

We have

$$\Big(\frac{\kappa - 1}{\kappa + 1}\Big)^2 = 1 - 4\kappa^{-1} + O(\kappa^{-2})$$

as $\kappa \to \infty$. Thus, (10.1.18) gives a slightly better, but conceptually similar, rate of convergence than $1 - \kappa^{-1}$ shown in Theorem 10.14.

10.1.4 PL-inequality

Linear convergence for gradient descent can also be shown under a weaker assumption known as the Polyak-Łojasiewicz-inequality, or PL-inequality for short.

Lemma 10.16. *Let $n \in \mathbb{N}$ and $\mu > 0$. Let $f\colon \mathbb{R}^n \to \mathbb{R}$ be μ-strongly convex and denote its unique minimizer by $\boldsymbol{w}_*$. Then f satisfies the* **PL-inequality**

$$\mu \cdot (f(\boldsymbol{w}) - f(\boldsymbol{w}_*)) \leq \frac{1}{2}\|\nabla f(\boldsymbol{w})\|^2 \qquad \text{for all } \boldsymbol{w} \in \mathbb{R}^n. \tag{10.1.19}$$

Proof. By μ-strong convexity we have

$$f(\boldsymbol{v}) \geq f(\boldsymbol{w}) + \langle \nabla f(\boldsymbol{w}), \boldsymbol{v} - \boldsymbol{w}\rangle + \frac{\mu}{2}\|\boldsymbol{v} - \boldsymbol{w}\|^2 \qquad \text{for all } \boldsymbol{v}, \boldsymbol{w} \in \mathbb{R}^n. \tag{10.1.20}$$

The gradient of the right-hand side with respect to $\boldsymbol{v}$ equals $\nabla f(\boldsymbol{w}) + \mu \cdot (\boldsymbol{v} - \boldsymbol{w})$. This implies that the minimum of this expression is attained at $\boldsymbol{v} = \boldsymbol{w} - \nabla f(\boldsymbol{w})/\mu$. Minimizing both sides of (10.1.20) in $\boldsymbol{v}$ we thus find

$$f(\boldsymbol{w}_*) \geq f(\boldsymbol{w}) - \frac{1}{\mu}\|\nabla f(\boldsymbol{w})\|^2 + \frac{1}{2\mu}\|\nabla f(\boldsymbol{w})\|^2 = f(\boldsymbol{w}) - \frac{1}{2\mu}\|\nabla f(\boldsymbol{w})\|^2.$$

Rearranging the terms gives (10.1.19). □

As the lemma states, the PL-inequality is implied by strong convexity. Moreover, it is indeed weaker than strong convexity, and does not even imply convexity, see Exercise 10.28. The next theorem, which corresponds to [219, Theorem 1], gives a convergence result for L-smooth functions satisfying the PL-inequality. It therefore does *not require convexity*. The proof is left as an exercise. We only note that the PL-inequality bounds the distance to the minimal value of the objective function by the squared norm of the gradient. It is thus precisely the type of bound required to show convergence of gradient descent.

Theorem 10.17. *Let $n \in \mathbb{N}$ and $L > 0$. Let $f\colon \mathbb{R}^n \to \mathbb{R}$ be L-smooth. Further, let $h_k = 1/L$ for all $k \in \mathbb{N}_0$, and let $(\boldsymbol{w}_k)_{k=0}^{\infty} \subseteq \mathbb{R}^n$ be defined by* (10.1.2)*, and let $\boldsymbol{w}_*$ be a (not necessarily unique) minimizer of f, so that the PL-inequality* (10.1.19) *holds.*

Then, it holds for all $k \in \mathbb{N}_0$ that

$$f(\boldsymbol{w}_k) - f(\boldsymbol{w}_*) \leq \Big(1 - \frac{\mu}{L}\Big)^k (f(\boldsymbol{w}_0) - f(\boldsymbol{w}_*)).$$

10.2 Stochastic gradient descent (SGD)

We next discuss a stochastic variant of gradient descent. The idea, which originally goes back to Robbins and Monro [189], is to replace the gradient $\nabla f(\boldsymbol{w}_k)$ in (10.1.2) by a random variable that

we denote by $\boldsymbol{G}_k$. We interpret $\boldsymbol{G}_k$ as an approximation to $\nabla f(\boldsymbol{w}_k)$; specifically, throughout we will assume that (given $\boldsymbol{w}_k$) $\boldsymbol{G}_k$ is an unbiased estimator, i.e.

$$\mathbb{E}[\boldsymbol{G}_k|\boldsymbol{w}_k] = \nabla f(\boldsymbol{w}_k). \tag{10.2.1}$$

After choosing some initial value $\boldsymbol{w}_0 \in \mathbb{R}^n$, the update rule becomes

$$\boldsymbol{w}_{k+1} := \boldsymbol{w}_k - h_k \boldsymbol{G}_k, \tag{10.2.2}$$

where $h_k > 0$ denotes again the step size, and unlike in Section 10.1, we focus here on the case of h_k depending on k. The iteration (10.2.2) creates a Markov chain $(\boldsymbol{w}_0, \boldsymbol{w}_1, \dots)$, meaning that $\boldsymbol{w}_k$ is a random variable, and its state only depends[1] on $\boldsymbol{w}_{k-1}$. The main reason for replacing the actual gradient by an estimator, is not to improve the accuracy or convergence rate, but rather to *decrease the computational cost and storage requirements* of the algorithm. The underlying assumption is that $\boldsymbol{G}_{k-1}$ can be computed at a fraction of the cost required for the computation of $\nabla f(\boldsymbol{w}_{k-1})$. The next example illustrates this in the standard setting.

Example 10.18 (Empirical risk minimization)**.** Suppose we have some data $S := (\boldsymbol{x}_j, y_j)_{j=1}^m$, where $y_j \in \mathbb{R}$ is the label corresponding to the data point $\boldsymbol{x}_j \in \mathbb{R}^d$. Using the square loss, we wish to fit a neural network $\Phi(\cdot, \boldsymbol{w}) : \mathbb{R}^d \to \mathbb{R}$ depending on parameters (i.e. weights and biases) $\boldsymbol{w} \in \mathbb{R}^n$, such that the empirical risk

$$f(\boldsymbol{w}) := \hat{\mathcal{R}}_S(\boldsymbol{w}) = \frac{1}{2m} \sum_{j=1}^m (\Phi(\boldsymbol{x}_j, \boldsymbol{w}) - y_j)^2,$$

is minimized. Performing one step of gradient descent requires the computation of

$$\nabla f(\boldsymbol{w}) = \frac{1}{m} \sum_{j=1}^m (\Phi(\boldsymbol{x}_j, \boldsymbol{w}) - y_j) \nabla_{\boldsymbol{w}} \Phi(\boldsymbol{x}_j, \boldsymbol{w}), \tag{10.2.3}$$

and thus the computation of m gradients of the neural network Φ. For large m (in practice m can be in the millions or even larger), this computation might be infeasible. To decrease computational complexity, we replace the full gradient (10.2.3) by

$$\boldsymbol{G} := (\Phi(\boldsymbol{x}_j, \boldsymbol{w}) - y_j) \nabla_{\boldsymbol{w}} \Phi(\boldsymbol{x}_j, \boldsymbol{w})$$

where $j \sim \text{uniform}(1, \dots, m)$ is a random variable with uniform distribution on the discrete set $\{1, \dots, m\}$. Then

$$\mathbb{E}[\boldsymbol{G}] = \frac{1}{m} \sum_{j=1}^m (\Phi(\boldsymbol{x}_j, \boldsymbol{w}) - y_j) \nabla_{\boldsymbol{w}} \Phi(\boldsymbol{x}_j, \boldsymbol{w}) = \nabla f(\boldsymbol{w}),$$

but an evaluation of $\boldsymbol{G}$ merely requires the computation of a single gradient of the neural network. More general, one can choose a **mini-batch** size m_b (where $m_b \ll m$) and let $\boldsymbol{G} = \frac{1}{m_b} \sum_{j \in J} (\Phi(\boldsymbol{x}_j, \boldsymbol{w}) - y_j) \nabla_{\boldsymbol{w}} \Phi(\boldsymbol{x}_j, \boldsymbol{w})$, where J is a random subset of $\{1, \dots, m\}$ of cardinality m_b.

[1] More precisely, given $\boldsymbol{w}_{k-1}$, the state of $\boldsymbol{w}_k$ is conditionally independent of $\boldsymbol{w}_1, \dots, \boldsymbol{w}_{k-2}$. See Appendix A.3.3.

Remark 10.19. In practice, the following variant is also common: Let $m_b k = m$ for m_b, k, $m \in \mathbb{N}$, i.e. the number of data points m is a k-fold multiple of the mini-batch size m_b. In each **epoch**, first a random partition $\dot{\bigcup}_{i=1}^{k} J_i = \{1, \dots, m\}$ is determined. Then for each $i = 1, \dots, k$, the weights are updated with the gradient estimate

$$\frac{1}{m_b} \sum_{j \in J_i} \Phi(\boldsymbol{x}_j - y_j, \boldsymbol{w}) \nabla_{\boldsymbol{w}} \Phi(\boldsymbol{x}_j, \boldsymbol{w}).$$

Hence, in one epoch (which corresponds to k updates of the neural network weights), the algorithm sweeps through the whole dataset.

SGD can be analyzed in various settings. To give the general idea, we concentrate on the case of L-smooth and μ-strongly convex objective functions. Let us start by looking at a property akin to the (descent) Lemma 10.5. Using Lemma 10.3

$$f(\boldsymbol{w}_{k+1}) \leq f(\boldsymbol{w}_k) - h_k \langle \nabla f(\boldsymbol{w}_k), \boldsymbol{G}_k \rangle + h_k^2 \frac{L}{2} \|\boldsymbol{G}_k\|^2.$$

In contrast to gradient descent, we cannot say anything about the sign of the term in the middle of the right-hand side. Thus, (10.2.2) need not necessarily decrease the value of the objective function in every step. The key insight is that *in expectation* the value is still decreased under certain assumptions, namely

$$\begin{aligned}
\mathbb{E}[f(\boldsymbol{w}_{k+1}) | \boldsymbol{w}_k] &\leq f(\boldsymbol{w}_k) - h_k \mathbb{E}[\langle \nabla f(\boldsymbol{w}_k), \boldsymbol{G}_k \rangle | \boldsymbol{w}_k] + h_k^2 \frac{L}{2} \mathbb{E}\left[\|\boldsymbol{G}_k\|^2 \middle| \boldsymbol{w}_k \right] \\
&= f(\boldsymbol{w}_k) - h_k \|\nabla f(\boldsymbol{w}_k)\|^2 + h_k^2 \frac{L}{2} \mathbb{E}\left[\|\boldsymbol{G}_k\|^2 \middle| \boldsymbol{w}_k \right] \\
&= f(\boldsymbol{w}_k) - h_k \left(\|\nabla f(\boldsymbol{w}_k)\|^2 - h_k \frac{L}{2} \mathbb{E}[\|\boldsymbol{G}_k\|^2 | \boldsymbol{w}_k] \right)
\end{aligned}$$

where we used (10.2.1).

Assuming, for some fixed $\gamma > 0$, the uniform bound

$$\mathbb{E}[\|\boldsymbol{G}_k\|^2 | \boldsymbol{w}_k] \leq \gamma$$

and that $\|\nabla f(\boldsymbol{w}_k)\| > 0$ (which is true unless $\boldsymbol{w}_k$ is the minimizer), upon choosing

$$0 < h_k < \frac{2 \|\nabla f(\boldsymbol{w}_k)\|^2}{L \gamma},$$

the expectation of the objective function decreases. Since $\nabla f(\boldsymbol{w}_k)$ tends to 0 as we approach the minimum, this also indicates that we should choose step sizes h_k that tend to 0 for $k \to \infty$. For our analysis we will work with the specific choice

$$h_k := \frac{1}{\mu} \frac{(k+1)^2 - k^2}{(k+1)^2} \quad \text{for all} \quad k \in \mathbb{N}_0, \tag{10.2.4}$$

as, e.g., in [75]. Note that

$$h_k = \frac{2k+1}{\mu(k+1)^2} = \frac{2}{\mu(k+1)} + O(k^{-2}) = O(k^{-1}).$$

Since $\boldsymbol{w}_k$ is a random variable by construction, a convergence statement can only be stochastic, e.g., in expectation or with high probability. We concentrate here on the former, but emphasize that also the latter can be shown.

Theorem 10.20. *Let $n \in \mathbb{N}$ and L, μ, $\gamma > 0$. Let $f\colon \mathbb{R}^n \to \mathbb{R}$ be L-smooth and μ-strongly convex. Let $(h_k)_{k=0}^{\infty}$ satisfy (10.2.4) and let $(\boldsymbol{G}_k)_{k=0}^{\infty}$, $(\boldsymbol{w}_k)_{k=0}^{\infty}$ be sequences of random variables satisfying (10.2.1) and (10.2.2). Assume that $\mathbb{E}[\|\boldsymbol{G}_k\|^2|\boldsymbol{w}_k] \le \gamma$ for all $k \in \mathbb{N}_0$.*

Then

$$\mathbb{E}[\|\boldsymbol{w}_k - \boldsymbol{w}_*\|^2] \le \frac{4\gamma}{\mu^2 k} = O(k^{-1}),$$
$$\mathbb{E}[f(\boldsymbol{w}_k)] - f(\boldsymbol{w}_*) \le \frac{4L\gamma}{2\mu^2 k} = O(k^{-1})$$

for $k \to \infty$.

Proof. We proceed similar as in the proof of Theorem 10.14. It holds for $k \ge 1$

$$\begin{aligned}
&\mathbb{E}[\|\boldsymbol{w}_k - \boldsymbol{w}_*\|^2|\boldsymbol{w}_{k-1}] \\
&\quad = \|\boldsymbol{w}_{k-1} - \boldsymbol{w}_*\|^2 - 2h_{k-1}\mathbb{E}[\langle \boldsymbol{G}_{k-1}, \boldsymbol{w}_{k-1} - \boldsymbol{w}_*\rangle\,|\boldsymbol{w}_{k-1}] + h_{k-1}^2\mathbb{E}[\|\boldsymbol{G}_{k-1}\|^2|\boldsymbol{w}_{k-1}] \\
&\quad = \|\boldsymbol{w}_{k-1} - \boldsymbol{w}_*\|^2 - 2h_{k-1}\langle \nabla f(\boldsymbol{w}_{k-1}), \boldsymbol{w}_{k-1} - \boldsymbol{w}_*\rangle + h_{k-1}^2\mathbb{E}[\|\boldsymbol{G}_{k-1}\|^2|\boldsymbol{w}_{k-1}].
\end{aligned}$$

By μ-strong convexity (10.1.15)

$$\begin{aligned}
-2h_{k-1}\langle \nabla f(\boldsymbol{w}_{k-1}), \boldsymbol{w}_{k-1} - \boldsymbol{w}_*\rangle &\le -\mu h_{k-1}\|\boldsymbol{w}_{k-1} - \boldsymbol{w}_*\|^2 - 2h_{k-1}\cdot(f(\boldsymbol{w}_{k-1}) - f(\boldsymbol{w}_*)) \\
&\le -\mu h_{k-1}\|\boldsymbol{w}_{k-1} - \boldsymbol{w}_*\|^2.
\end{aligned}$$

Thus

$$\mathbb{E}[\|\boldsymbol{w}_k - \boldsymbol{w}_*\|^2|\boldsymbol{w}_{k-1}] \le (1 - \mu h_{k-1})\|\boldsymbol{w}_{k-1} - \boldsymbol{w}_*\|^2 + h_{k-1}^2\gamma.$$

Using the Markov property, we have

$$\mathbb{E}[\|\boldsymbol{w}_k - \boldsymbol{w}_*\|^2|\boldsymbol{w}_{k-1}, \boldsymbol{w}_{k-2}] = \mathbb{E}[\|\boldsymbol{w}_k - \boldsymbol{w}_*\|^2|\boldsymbol{w}_{k-1}]$$

so that

$$\mathbb{E}[\|\boldsymbol{w}_k - \boldsymbol{w}_*\|^2|\boldsymbol{w}_{k-1}] \le (1 - \mu h_{k-1})\mathbb{E}[\|\boldsymbol{w}_{k-1} - \boldsymbol{w}_*\|^2|\boldsymbol{w}_{k-2}] + h_{k-1}^2\gamma.$$

With $e_0 := \|\boldsymbol{w}_0 - \boldsymbol{w}_*\|^2$ and $e_k := \mathbb{E}[\|\boldsymbol{w}_k - \boldsymbol{w}_*\|^2|\boldsymbol{w}_{k-1}]$ for $k \ge 1$ we have found

$$\begin{aligned}
e_k &\le (1 - \mu h_{k-1})e_{k-1} + h_{k-1}^2\gamma \\
&\le (1 - \mu h_{k-1})((1 - \mu h_{k-2})e_{k-2} + h_{k-2}^2\gamma) + h_{k-1}^2\gamma \\
&\le \cdots \le e_0\prod_{j=0}^{k-1}(1 - \mu h_j) + \gamma\sum_{j=0}^{k-1} h_j^2\prod_{i=j+1}^{k-1}(1 - \mu h_i).
\end{aligned}$$

By choice of h_i

$$\prod_{i=j}^{k-1}(1-\mu h_i) = \prod_{i=j}^{k-1}\frac{i^2}{(i+1)^2} = \frac{j^2}{k^2}$$

and thus

$$\begin{aligned} e_k &\le \frac{\gamma}{\mu^2}\sum_{j=0}^{k-1}\left(\frac{(j+1)^2-j^2}{(j+1)^2}\right)^2\frac{(j+1)^2}{k^2} \\ &\le \frac{\gamma}{\mu^2}\frac{1}{k^2}\sum_{j=0}^{k-1}\underbrace{\frac{(2j+1)^2}{(j+1)^2}}_{\le 4} \\ &\le \frac{\gamma}{\mu^2}\frac{4k}{k^2} \\ &\le \frac{4\gamma}{\mu^2 k}. \end{aligned}$$

Since $\mathbb{E}[\|\boldsymbol{w}_k - \boldsymbol{w}_*\|^2]$ is the expectation of $\mathbb{E}[\|\boldsymbol{w}_k - \boldsymbol{w}_*\|^2|\boldsymbol{w}_{k-1}]$ with respect to the random variable $\boldsymbol{w}_{k-1}$, and $e_0/k^2 + 4\gamma/(\mu^2 k)$ is a constant independent of $\boldsymbol{w}_{k-1}$, we obtain

$$\mathbb{E}[\|\boldsymbol{w}_k - \boldsymbol{w}_*\|^2] \le \frac{4\gamma}{\mu^2 k}.$$

Finally, using L-smoothness

$$f(\boldsymbol{w}_k) - f(\boldsymbol{w}_*) \le \langle \nabla f(\boldsymbol{w}_*), \boldsymbol{w}_k - \boldsymbol{w}_*\rangle + \frac{L}{2}\|\boldsymbol{w}_k - \boldsymbol{w}_*\|^2 = \frac{L}{2}\|\boldsymbol{w}_k - \boldsymbol{w}_*\|^2,$$

and taking the expectation concludes the proof. □

The specific choice of h_k in (10.2.4) simplifies the calculations in the proof, but it is not necessary in order for the asymptotic convergence to hold. One can show similar convergence results with $h_k = c_1/(c_2 + k)$ under certain assumptions on c_1, c_2, e.g. [23, Theorem 4.7].

10.3 Backpropagation

We now explain how to apply gradient-based methods to the training of neural networks. Let $\Phi \in \mathcal{N}_{d_0}^{d_{L+1}}(\sigma; L, n)$ (see Definition 3.6) and assume that the activation function satisfies $\sigma \in C^1(\mathbb{R})$. As earlier, we denote the neural network parameters by

$$\boldsymbol{w} = ((\boldsymbol{W}^{(0)}, \boldsymbol{b}^{(0)}), \dots, (\boldsymbol{W}^{(L)}, \boldsymbol{b}^{(L)})) \tag{10.3.1}$$

with weight matrices $\boldsymbol{W}^{(\ell)} \in \mathbb{R}^{d_{\ell+1}\times d_\ell}$ and bias vectors $\boldsymbol{b}^{(\ell)} \in \mathbb{R}^{d_{\ell+1}}$. Additionally, we fix a differentiable loss function $\mathcal{L} : \mathbb{R}^{d_{L+1}} \times \mathbb{R}^{d_{L+1}} \to \mathbb{R}$, e.g., $\mathcal{L}(\boldsymbol{w}, \tilde{\boldsymbol{w}}) = \|\boldsymbol{w} - \tilde{\boldsymbol{w}}\|^2/2$, and assume given data $(\boldsymbol{x}_j, \boldsymbol{y}_j)_{j=1}^m \subseteq \mathbb{R}^{d_0} \times \mathbb{R}^{d_{L+1}}$. The goal is to minimize an empirical risk of the form

$$f(\boldsymbol{w}) := \frac{1}{m}\sum_{j=1}^m \mathcal{L}(\Phi(\boldsymbol{x}_j, \boldsymbol{w}), \boldsymbol{y}_j)$$

as a function of the neural network parameters $\boldsymbol{w}$. An application of the gradient step (10.1.2) to update the parameters requires the computation of

$$\nabla f(\boldsymbol{w}) = \frac{1}{m}\sum_{j=1}^{m} \nabla_{\boldsymbol{w}} \mathcal{L}(\Phi(\boldsymbol{x}_j, \boldsymbol{w}), \boldsymbol{y}_j).$$

For stochastic methods, as explained in Example 10.18, we only compute the average over a (random) subbatch of the dataset. In either case, we need an algorithm to determine $\nabla_{\boldsymbol{w}} \mathcal{L}(\Phi(\boldsymbol{x}, \boldsymbol{w}), \boldsymbol{y})$, i.e. the gradients

$$\nabla_{\boldsymbol{b}^{(\ell)}} \mathcal{L}(\Phi(\boldsymbol{x}, \boldsymbol{w}), \boldsymbol{y}) \in \mathbb{R}^{d_{\ell+1}}, \quad \nabla_{\boldsymbol{W}^{(\ell)}} \mathcal{L}(\Phi(\boldsymbol{x}, \boldsymbol{w}), \boldsymbol{y}) \in \mathbb{R}^{d_{\ell+1} \times d_\ell} \tag{10.3.2}$$

for all $\ell = 0, \dots, L$.

The backpropagation algorithm [196] provides an *efficient* way to do so. To explain it, for fixed input $\boldsymbol{x} \in \mathbb{R}^{d_0}$ introduce the notation

$$\bar{\boldsymbol{x}}^{(1)} := \boldsymbol{W}^{(0)}\boldsymbol{x} + \boldsymbol{b}^{(0)} \tag{10.3.3a}$$

$$\bar{\boldsymbol{x}}^{(\ell+1)} := \boldsymbol{W}^{(\ell)}\sigma(\bar{\boldsymbol{x}}^{(\ell)}) + \boldsymbol{b}^{(\ell)} \qquad \text{for } \ell \in \{1, \dots, L\}, \tag{10.3.3b}$$

where the application of $\sigma : \mathbb{R} \to \mathbb{R}$ to a vector is, as always, understood componentwise. With the notation of Definition 2.1, $\boldsymbol{x}^{(\ell)} = \sigma(\bar{\boldsymbol{x}}^{(\ell)}) \in \mathbb{R}^{d_\ell}$ for $\ell = 1, \dots, L$ and $\bar{\boldsymbol{x}}^{(L+1)} = \boldsymbol{x}^{(L+1)} = \Phi(\boldsymbol{x}, \boldsymbol{w}) \in \mathbb{R}^{d_{L+1}}$ is the output of the neural network. Therefore, the $\bar{\boldsymbol{x}}^{(\ell)}$ for $\ell = 1, \dots, L$ are sometimes also referred to as the *preactivations*.

In the following, we additionally fix $\boldsymbol{y} \in \mathbb{R}^{d_{L+1}}$ and write

$$\mathcal{L} := \mathcal{L}(\Phi(\boldsymbol{x}, \boldsymbol{w}), \boldsymbol{y}) = \mathcal{L}(\bar{\boldsymbol{x}}^{(L+1)}, \boldsymbol{y}).$$

Note that $\bar{\boldsymbol{x}}^{(k)}$ depends on $(\boldsymbol{W}^{(\ell)}, \boldsymbol{b}^{(\ell)})$ only if $k > \ell$. Since $\bar{\boldsymbol{x}}^{(\ell+1)}$ is a function of $\bar{\boldsymbol{x}}^{(\ell)}$ for each ℓ, by repeated application of the chain rule

$$\frac{\partial \mathcal{L}}{\partial W_{ij}^{(\ell)}} = \underbrace{\frac{\partial \mathcal{L}}{\partial \bar{\boldsymbol{x}}^{(L+1)}}}_{\in \mathbb{R}^{1 \times d_{L+1}}} \underbrace{\frac{\partial \bar{\boldsymbol{x}}^{(L+1)}}{\partial \bar{\boldsymbol{x}}^{(L)}}}_{\in \mathbb{R}^{d_{L+1} \times d_L}} \cdots \underbrace{\frac{\partial \bar{\boldsymbol{x}}^{(\ell+2)}}{\partial \bar{\boldsymbol{x}}^{(\ell+1)}}}_{\in \mathbb{R}^{d_{\ell+2} \times d_{\ell+1}}} \underbrace{\frac{\partial \bar{\boldsymbol{x}}^{(\ell+1)}}{\partial W_{ij}^{(\ell)}}}_{\in \mathbb{R}^{d_{\ell+1} \times 1}}. \tag{10.3.4}$$

An analogous calculation holds for $\partial \mathcal{L}/\partial b_j^{(\ell)}$. Since all terms in (10.3.4) are easy to compute (see (10.3.3)), in principle we could use this formula to determine the gradients in (10.3.2). To avoid unnecessary computations, the main idea of backpropagation is to introduce

$$\boldsymbol{\alpha}^{(\ell)} := \nabla_{\bar{\boldsymbol{x}}^{(\ell)}} \mathcal{L} \in \mathbb{R}^{d_\ell} \qquad \text{for all } \ell = 1, \dots, L+1$$

and observe that

$$\frac{\partial \mathcal{L}}{\partial W_{ij}^{(\ell)}} = (\boldsymbol{\alpha}^{(\ell+1)})^\top \frac{\partial \bar{\boldsymbol{x}}^{(\ell+1)}}{\partial W_{ij}^{(\ell)}}.$$

As the following lemma shows, the $\boldsymbol{\alpha}^{(\ell)}$ can be computed recursively for $\ell = L+1, \dots, 1$. This explains the name "backpropagation". In the following, $\odot$ denotes the componentwise (Hadamard) product, i.e. $\boldsymbol{a} \odot \boldsymbol{b} = (a_i b_i)_{i=1}^d$ for every $\boldsymbol{a}$, $\boldsymbol{b} \in \mathbb{R}^d$.

Lemma 10.21. *It holds*

$$\boldsymbol{\alpha}^{(L+1)} = \nabla_{\bar{\boldsymbol{x}}^{(L+1)}} \mathcal{L}(\bar{\boldsymbol{x}}^{(L+1)}, \boldsymbol{y}) \tag{10.3.5}$$

and

$$\boldsymbol{\alpha}^{(\ell)} = \sigma'(\bar{\boldsymbol{x}}^{(\ell)}) \odot (\boldsymbol{W}^{(\ell)})^\top \boldsymbol{\alpha}^{(\ell+1)} \qquad \text{for all } \ell = L, \dots, 1.$$

Proof. Equation (10.3.5) holds by definition. For $\ell \in \{1, \dots, L\}$ by the chain rule

$$\boldsymbol{\alpha}^{(\ell)} = \frac{\partial \mathcal{L}}{\partial \bar{\boldsymbol{x}}^{(\ell)}} = \underbrace{\Big(\frac{\partial \bar{\boldsymbol{x}}^{(\ell+1)}}{\partial \bar{\boldsymbol{x}}^{(\ell)}}\Big)^\top}_{\in \mathbb{R}^{d_\ell \times d_{\ell+1}}} \underbrace{\frac{\partial \mathcal{L}}{\partial \bar{\boldsymbol{x}}^{(\ell+1)}}}_{\in \mathbb{R}^{d_{\ell+1} \times 1}} = \Big(\frac{\partial \bar{\boldsymbol{x}}^{(\ell+1)}}{\partial \bar{\boldsymbol{x}}^{(\ell)}}\Big)^\top \boldsymbol{\alpha}^{(\ell+1)}.$$

By (10.3.3b) for $i \in \{1, \dots, d_{\ell+1}\}$, $j \in \{1, \dots, d_\ell\}$

$$\Big(\frac{\partial \bar{\boldsymbol{x}}^{(\ell+1)}}{\partial \bar{\boldsymbol{x}}^{(\ell)}}\Big)_{ij} = \frac{\partial \bar{x}_i^{(\ell+1)}}{\partial \bar{x}_j^{(\ell)}} = W_{ij}^{(\ell)} \sigma'(\bar{x}_j^{(\ell)}).$$

Thus the claim follows. □

Putting everything together, we obtain explicit formulas for (10.3.2).

Proposition 10.22. *It holds*

$$\nabla_{\boldsymbol{b}^{(\ell)}} \mathcal{L} = \boldsymbol{\alpha}^{(\ell+1)} \in \mathbb{R}^{d_{\ell+1}} \qquad \text{for } \ell = 0, \dots, L$$

and

$$\nabla_{\boldsymbol{W}^{(0)}} \mathcal{L} = \boldsymbol{\alpha}^{(1)} \boldsymbol{x}^\top \in \mathbb{R}^{d_1 \times d_0}$$

and

$$\nabla_{\boldsymbol{W}^{(\ell)}} \mathcal{L} = \boldsymbol{\alpha}^{(\ell+1)} \sigma(\bar{\boldsymbol{x}}^{(\ell)})^\top \in \mathbb{R}^{d_{\ell+1} \times d_\ell} \qquad \text{for } \ell = 1, \dots, L.$$

Proof. By (10.3.3a) for i, $k \in \{1, \dots, d_1\}$, and $j \in \{1, \dots, d_0\}$

$$\frac{\partial \bar{x}_k^{(1)}}{\partial b_i^{(0)}} = \delta_{ki} \qquad \text{and} \qquad \frac{\partial \bar{x}_k^{(1)}}{\partial W_{ij}^{(0)}} = \delta_{ki} x_j,$$

and by (10.3.3b) for $\ell \in \{1, \dots, L\}$ and i, $k \in \{1, \dots, d_{\ell+1}\}$, and $j \in \{1, \dots, d_\ell\}$

$$\frac{\partial \bar{x}_k^{(\ell+1)}}{\partial b_i^{(\ell)}} = \delta_{ki} \qquad \text{and} \qquad \frac{\partial \bar{x}_k^{(\ell+1)}}{\partial W_{ij}^{(\ell)}} = \delta_{ki} \sigma(\bar{x}_j^{(\ell)}).$$

Thus, with $\boldsymbol{e}_i = (\delta_{ki})_{k=1}^{d_{\ell+1}}$

$$\frac{\partial \mathcal{L}}{\partial b_i^{(\ell)}} = \Big(\frac{\partial \bar{\boldsymbol{x}}^{(\ell+1)}}{\partial b_i^{(\ell)}}\Big)^\top \frac{\partial \mathcal{L}}{\partial \bar{\boldsymbol{x}}^{(\ell+1)}} = \boldsymbol{e}_i^\top \boldsymbol{\alpha}^{(\ell+1)} = \alpha_i^{(\ell+1)} \qquad \text{for } \ell \in \{0, \dots, L\}$$

and similarly

$$\frac{\partial \mathcal{L}}{\partial W_{ij}^{(0)}} = \Big(\frac{\partial \bar{\boldsymbol{x}}^{(1)}}{\partial W_{ij}^{(0)}}\Big)^\top \boldsymbol{\alpha}^{(1)} = \bar{x}_j^{(0)} \boldsymbol{e}_i^\top \boldsymbol{\alpha}^{(1)} = \bar{x}_j^{(0)} \alpha_i^{(1)}$$

and

$$\frac{\partial \mathcal{L}}{\partial W_{ij}^{(\ell)}} = \sigma(\bar{x}_j^{(\ell)}) \alpha_i^{(\ell+1)} \qquad \text{for } \ell \in \{1, \dots, L\}.$$

This concludes the proof. □

Lemma 10.21 and Proposition 10.22 motivate Algorithm 1, in which a forward pass computing $\bar{\boldsymbol{x}}^{(\ell)}$, $\ell = 1, \dots, L+1$, is followed by a backward pass to determine the $\boldsymbol{\alpha}^{(\ell)}$, $\ell = L+1, \dots, 1$, and the gradients of $\mathcal{L}$ with respect to the neural network parameters. This shows how to use gradient-based optimizers from the previous sections for the training of neural networks.

Two important remarks are in order. First, the objective function associated to neural networks is typically not convex as a function of the neural network weights and biases. Thus, the analysis of the previous sections will in general not be directly applicable. It may still give some insight about the convergence behavior locally around the minimizer however. Second, to derive the back-propagation algorithm we assumed the activation function to be continuously differentiable, which does not hold for ReLU. Using the concept of subgradients, gradient-based algorithms and their analysis may be generalized to some extent to also accommodate non-differentiable loss functions, see Exercises 10.31–10.33.

10.4 Acceleration

Acceleration is an important tool for the training of neural networks [220]. The idea was first introduced by Polyak in 1964 under the name "heavy ball method" [178]. It is inspired by the dynamics of a heavy ball rolling down the valley of the loss landscape. Since then other types of acceleration have been proposed and analyzed, with Nesterov acceleration being the most prominent example [158]. In this section, we first give some intuition by discussing the heavy ball method for a simple quadratic loss. Afterwards we turn to Nesterov acceleration and give a convergence proof for L-smooth and μ-strongly convex objective functions that improves upon the bounds obtained for gradient descent.

10.4.1 Heavy ball method

We proceed similar as in [69, 179, 181] to motivate the idea. Consider the quadratic objective function in two dimensions

$$f(\boldsymbol{w}) := \frac{1}{2} \boldsymbol{w}^\top \boldsymbol{D} \boldsymbol{w} \qquad \text{where} \qquad \boldsymbol{D} = \begin{pmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{pmatrix} \tag{10.4.1}$$

Algorithm 1 Backpropagation

Input: Network input $\boldsymbol{x}$, target output $\boldsymbol{y}$, neural network parameters $((\boldsymbol{W}^{(0)}, \boldsymbol{b}^{(0)}), \dots, (\boldsymbol{W}^{(L)}, \boldsymbol{b}^{(L)}))$
Output: Gradients of the loss function $\mathcal{L}$ with respect to neural network parameters

Forward pass
$\bar{\boldsymbol{x}}^{(1)} \leftarrow \boldsymbol{W}^{(0)}\boldsymbol{x} + \boldsymbol{b}^{(0)}$
for $\ell = 1, \dots, L$ **do**
  $\bar{\boldsymbol{x}}^{(\ell+1)} \leftarrow \boldsymbol{W}^{(\ell)}\sigma(\bar{\boldsymbol{x}}^{(\ell)}) + \boldsymbol{b}^{(\ell)}$
end for

Backward pass
$\boldsymbol{\alpha}^{(L+1)} \leftarrow \nabla_{\bar{\boldsymbol{x}}^{(L+1)}}\mathcal{L}(\bar{\boldsymbol{x}}^{(L+1)}, \boldsymbol{y})$
for $\ell = L, \dots, 1$ **do**
  $\nabla_{\boldsymbol{b}^{(\ell)}}\mathcal{L} \leftarrow \boldsymbol{\alpha}^{(\ell+1)}$
  $\nabla_{\boldsymbol{W}^{(\ell)}}\mathcal{L} \leftarrow \boldsymbol{\alpha}^{(\ell+1)}\sigma(\bar{\boldsymbol{x}}^{(\ell)})^\top$
  $\boldsymbol{\alpha}^{(\ell)} \leftarrow \sigma'(\bar{\boldsymbol{x}}^{(\ell)}) \odot (\boldsymbol{W}^{(\ell)})^\top\boldsymbol{\alpha}^{(\ell+1)}$
end for
$\nabla_{\boldsymbol{b}^{(0)}}\mathcal{L} \leftarrow \boldsymbol{\alpha}^{(1)}$
$\nabla_{\boldsymbol{W}^{(0)}}\mathcal{L} \leftarrow \boldsymbol{\alpha}^{(1)}\boldsymbol{x}^\top$

with $\lambda_1 \geq \lambda_2 > 0$. Clearly, f has a unique minimizer at $\boldsymbol{w}_* = \boldsymbol{0} \in \mathbb{R}^2$. Starting at some $\boldsymbol{w}_0 \in \mathbb{R}^2$, gradient descent with constant step size $h > 0$ computes the iterates

$$\boldsymbol{w}_{k+1} = \boldsymbol{w}_k - h\boldsymbol{D}\boldsymbol{w}_k = \begin{pmatrix} 1 - h\lambda_1 & 0 \\ 0 & 1 - h\lambda_2 \end{pmatrix} \boldsymbol{w}_k = \begin{pmatrix} (1 - h\lambda_1)^{k+1} & 0 \\ 0 & (1 - h\lambda_2)^{k+1} \end{pmatrix} \boldsymbol{w}_0.$$

The method converges for arbitrary initialization $\boldsymbol{w}_0$ if and only if

$$|1 - h\lambda_1| < 1 \qquad \text{and} \qquad |1 - h\lambda_2| < 1.$$

The optimal step size balancing out the speed of convergence in both coordinates is

$$h_* = \operatorname{argmin}_{h>0} \max\{|1 - h\lambda_1|, |1 - h\lambda_2|\} = \frac{2}{\lambda_1 + \lambda_2}. \tag{10.4.2}$$

With $\kappa = \lambda_1/\lambda_2$ we then obtain the convergence rate

$$|1 - h_*\lambda_1| = |1 - h_*\lambda_2| = \frac{\lambda_1 - \lambda_2}{\lambda_1 + \lambda_2} = \frac{\kappa - 1}{\kappa + 1} \in [0, 1). \tag{10.4.3}$$

If $\lambda_1 \gg \lambda_2$, this term is close to 1, and thus the convergence will be slow. This is consistent with our analysis for strongly convex objective functions; by Exercise 10.34 the condition number of f equals $\kappa = \lambda_1/\lambda_2 \gg 1$. Hence, the upper bounds in Theorem 10.14 and Remark 10.15 converge only slowly. Similar considerations hold for general quadratic objective functions in $\mathbb{R}^n$ such as

$$\tilde{f}(\boldsymbol{w}) = \frac{1}{2}\boldsymbol{w}^\top\boldsymbol{A}\boldsymbol{w} + \boldsymbol{b}^\top\boldsymbol{w} + c \tag{10.4.4}$$

with $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ symmetric positive definite, $\boldsymbol{b} \in \mathbb{R}^n$ and $c \in \mathbb{R}$, see Exercise 10.35.

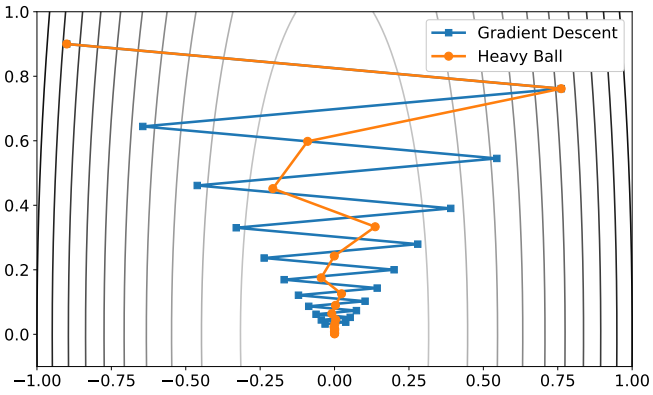

Figure 10.3: 20 steps of gradient descent and the heavy ball method on the objective function (10.4.1) with $\lambda_1 = 12 \gg 1 = \lambda_2$, step size $h = \alpha = h_*$ as in (10.4.2), and $\beta = 1/3$.

Remark 10.23. Interpreting (10.4.4) as a second-order Taylor expansion of some objective function $\tilde{f}$ around its minimizer $\boldsymbol{w}_*$, we note that the above described effects also occur for general objective functions with ill-conditioned Hessians at the minimizer.

Figure 10.3 gives further insight into the poor performance of gradient descent for (10.4.1) with $\lambda_1 \gg \lambda_2$. The loss-landscape looks like a ravine (the derivative is much larger in one direction than the other), and away from the floor, ∇f mainly points to the opposite side. Therefore the iterates oscillate back and forth in the first coordinate, and make little progress in the direction of the valley along the second coordinate axis. To address this problem, the heavy ball method introduces a "momentum" term which can mitigate this effect to some extent. The idea is, to choose the update not just according to the gradient at the current location, but to add information from the previous steps. After initializing $\boldsymbol{w}_0$ and, e.g., $\boldsymbol{w}_1 = \boldsymbol{w}_0 - \alpha \nabla f(\boldsymbol{w}_0)$, let for $k \in \mathbb{N}$

$$\boldsymbol{w}_{k+1} = \boldsymbol{w}_k - \alpha \nabla f(\boldsymbol{w}_k) + \beta(\boldsymbol{w}_k - \boldsymbol{w}_{k-1}). \tag{10.4.5}$$

This is known as Polyak's heavy ball method [178]. Here $\alpha > 0$ and $\beta \in (0, 1)$ are hyperparameters (that could also depend on k) and in practice need to be carefully tuned to balance the strength of the gradient and the momentum term. Iteratively expanding (10.4.5) with the given initialization, observe that for $k \geq 0$

$$\boldsymbol{w}_{k+1} = \boldsymbol{w}_k - \alpha \left(\sum_{j=0}^{k} \beta^j \nabla f(\boldsymbol{w}_{k-j}) \right). \tag{10.4.6}$$

Thus, $\boldsymbol{w}_k$ is updated using an *exponentially weighted average* of all past gradients. Choosing the momentum parameter β in the interval $(0, 1)$ ensures that the influence of previous gradients on the update decays exponentially. The concrete value of β determines the balance between the impact of recent and past gradients.

Intuitively, this (exponentially weighted) linear combination of the past gradients averages out some of the oscillation observed for gradient descent in Figure 10.3 in the first coordinate, and thus "smoothes" the path. The partial derivative in the second coordinate, along which the objective

function is very flat, does not change much from one iterate to the next. Thus, its proportion in the update is strengthened through the addition of momentum. This is observed in Figure 10.3.

As mentioned earlier, the heavy ball method can be interpreted as a discretization of the dynamics of a ball rolling down the valley of the loss landscape. If the ball has positive mass, i.e. is "heavy", its momentum prevents the ball from bouncing back and forth too strongly. The following remark further elucidates this connection.

Remark 10.24. As pointed out, e.g., in [179, 181], for suitable choices of α and β, (10.4.5) can be interpreted as a discretization of the second-order ODE

$$m\boldsymbol{w}''(t) = -\nabla f(\boldsymbol{w}(t)) - r\boldsymbol{w}'(t). \tag{10.4.7}$$

This equation describes the movement of a point mass m under influence of the force field $-\nabla f(\boldsymbol{w}(t))$; the term $-\boldsymbol{w}'(t)$, which points in the negative direction of the current velocity, corresponds to friction, and $r > 0$ is the friction coefficient. The discretization

$$m\frac{\boldsymbol{w}_{k+1} - 2\boldsymbol{w}_k + \boldsymbol{w}_{k-1}}{h^2} = -\nabla f(\boldsymbol{w}_k) - \frac{\boldsymbol{w}_{k+1} - \boldsymbol{w}_k}{h}$$

then leads to

$$\boldsymbol{w}_{k+1} = \boldsymbol{w}_k - \underbrace{\frac{h^2}{m - rh}}_{=\alpha}\nabla f(\boldsymbol{w}_k) + \underbrace{\frac{m}{m - rh}}_{=\beta}(\boldsymbol{w}_k - \boldsymbol{w}_{k-1}), \tag{10.4.8}$$

and thus to (10.4.5), [181].

Letting $m = 0$ in (10.4.8), we recover the gradient descent update (10.1.2). Hence, the positive mass corresponds to the momentum term. Similarly, letting $m = 0$ in the continuous dynamics (10.4.7), we obtain the gradient flow (10.1.3). The key difference between these equations is that $-\nabla f(\boldsymbol{w}(t))$ represents the *velocity* of $\boldsymbol{w}(t)$ in (10.1.3), whereas in (10.4.7), up to the friction term, it corresponds to an *acceleration*.

Let us sketch an argument to show that (10.4.5) improves the convergence over plain gradient descent for the objective function (10.4.1). Denoting $\boldsymbol{w}_k = (w_{k,1}, w_{k,2})^\top \in \mathbb{R}^2$, we obtain from (10.4.5) and the definition of f in (10.4.1)

$$\begin{pmatrix} w_{k+1,j} \\ w_{k,j} \end{pmatrix} = \begin{pmatrix} 1 + \beta - \alpha\lambda_j & -\beta \\ 1 & 0 \end{pmatrix} \begin{pmatrix} w_{k,j} \\ w_{k-1,j} \end{pmatrix} \tag{10.4.9}$$

for $j \in \{1, 2\}$ and $k \geq 1$. The smaller the modulus of the eigenvalues of the matrix in (10.4.9), the faster the convergence towards the minimizer $w_{*,j} = 0 \in \mathbb{R}$ for arbitrary initialization. Hence, the goal is to choose $\alpha > 0$ and $\beta \in (0, 1)$ such that the maximal modulus of the eigenvalues of the matrix for $j \in \{1, 2\}$ is possibly small. We omit the details of this calculation (also see [179, 163, 69]), but mention that this is obtained for

$$\alpha = \Big(\frac{2}{\sqrt{\lambda_1} + \sqrt{\lambda_2}}\Big)^2 \qquad \text{and} \qquad \beta = \Big(\frac{\sqrt{\lambda_1} - \sqrt{\lambda_2}}{\sqrt{\lambda_1} + \sqrt{\lambda_2}}\Big)^2.$$

With these choices, the modulus of the maximal eigenvalue is bounded by

$$\sqrt{\beta} = \frac{\sqrt{\kappa} - 1}{\sqrt{\kappa} + 1} \in [0, 1),$$

where again $\kappa = \lambda_1/\lambda_2$. Due to (10.4.9), this expression gives a rate of convergence for (10.4.5). Contrary to gradient descent, see (10.4.3), for this problem the heavy ball method achieves a convergence rate that only depends on the *square root* of the condition number κ. This explains the improved performance observed in Figure 10.3.

10.4.2 Nesterov acceleration

Nesterov's accelerated gradient method (NAG) [158, 157], is a refinement of the heavy ball method. After initializing $\boldsymbol{v}_0$, $\boldsymbol{w}_0 \in \mathbb{R}^n$, the update is formulated as the two-step process

$$\boldsymbol{v}_{k+1} = \boldsymbol{w}_k - \alpha \nabla f(\boldsymbol{w}_k) \tag{10.4.10a}$$

$$\boldsymbol{w}_{k+1} = \boldsymbol{v}_{k+1} + \beta(\boldsymbol{v}_{k+1} - \boldsymbol{v}_k), \tag{10.4.10b}$$

where again $\alpha > 0$ and $\beta \in (0, 1)$ are hyperparameters. Substituting the second line into the first we get

$$\boldsymbol{v}_{k+1} = \boldsymbol{v}_k - \alpha \nabla f(\boldsymbol{w}_k) + \beta(\boldsymbol{v}_k - \boldsymbol{v}_{k-1}).$$

Comparing with the heavy ball method (10.4.5), the key difference is that the gradient is not evaluated at the current position $\boldsymbol{v}_k$, but instead at the point $\boldsymbol{w}_k = \boldsymbol{v}_k + \beta(\boldsymbol{v}_k - \boldsymbol{v}_{k-1})$, which can be interpreted as an estimate of the position at the next iteration.

We next discuss the convergence for L-smooth and μ-strongly convex objective functions f. It turns out, that these conditions are not sufficient in order for the heavy ball method (10.4.5) to converge, and one can construct counterexamples [131]. This is in contrast to NAG, as the next theorem shows. To give the analysis, it is convenient to first rewrite (10.4.10) as a three sequence update: Let $\tau = \sqrt{\mu/L}$, $\alpha = 1/L$, and $\beta = (1-\tau)/(1+\tau)$. After initializing $\boldsymbol{w}_0$, $\boldsymbol{v}_0 \in \mathbb{R}^n$, (10.4.10) can also be written as $\boldsymbol{u}_0 = ((1+\tau)\boldsymbol{w}_0 - \boldsymbol{v}_0)/\tau$ and for $k \in \mathbb{N}_0$

$$\boldsymbol{w}_k = \frac{\tau}{1+\tau}\boldsymbol{u}_k + \frac{1}{1+\tau}\boldsymbol{v}_k \tag{10.4.11a}$$

$$\boldsymbol{v}_{k+1} = \boldsymbol{w}_k - \frac{1}{L}\nabla f(\boldsymbol{w}_k) \tag{10.4.11b}$$

$$\boldsymbol{u}_{k+1} = \boldsymbol{u}_k + \tau \cdot (\boldsymbol{w}_k - \boldsymbol{u}_k) - \frac{\tau}{\mu}\nabla f(\boldsymbol{w}_k), \tag{10.4.11c}$$

see Exercise 10.36.

The proof of the following theorem proceeds along the lines of [230, 240].

Theorem 10.25. *Let $n \in \mathbb{N}$ and L, $\mu > 0$. Let $f\colon \mathbb{R}^n \to \mathbb{R}$ be L-smooth and μ-strongly convex. Further, let $\boldsymbol{v}_0$, $\boldsymbol{w}_0 \in \mathbb{R}^n$ and let $\tau = \sqrt{\mu/L}$. Let $(\boldsymbol{w}_k, \boldsymbol{v}_{k+1}, \boldsymbol{u}_{k+1})_{k=0}^{\infty} \subseteq \mathbb{R}^n$ be defined by* (10.4.11a)*, and let $\boldsymbol{w}_*$ be the unique minimizer of f.*

Then, for all $k \in \mathbb{N}_0$, it holds that

$$\|\boldsymbol{u}_k - \boldsymbol{w}_*\|^2 \leq \frac{2}{\mu}\Big(1 - \sqrt{\frac{\mu}{L}}\Big)^k \Big(f(\boldsymbol{v}_0) - f(\boldsymbol{w}_*) + \frac{\mu}{2}\|\boldsymbol{u}_0 - \boldsymbol{w}_*\|^2\Big), \tag{10.4.12a}$$

$$f(\boldsymbol{v}_k) - f(\boldsymbol{w}_*) \leq \Big(1 - \sqrt{\frac{\mu}{L}}\Big)^k \Big(f(\boldsymbol{v}_0) - f(\boldsymbol{w}_*) + \frac{\mu}{2}\|\boldsymbol{u}_0 - \boldsymbol{w}_*\|^2\Big). \tag{10.4.12b}$$

Proof. Define

$$e_k := f(\boldsymbol{v}_k) - f(\boldsymbol{w}_*) + \frac{\mu}{2}\|\boldsymbol{u}_k - \boldsymbol{w}_*\|^2. \tag{10.4.13}$$

To show (10.4.12), it suffices to prove with $c = 1 - \tau$ that $e_{k+1} \leq c e_k$ for all $k \in \mathbb{N}_0$.

We start with the last term in (10.4.13). By (10.4.11c)

$$\begin{aligned}
&\frac{\mu}{2}\|\boldsymbol{u}_{k+1} - \boldsymbol{w}_*\|^2 - \frac{\mu}{2}\|\boldsymbol{u}_k - \boldsymbol{w}_*\|^2 \\
&\quad = \frac{\mu}{2}\|\boldsymbol{u}_{k+1} - \boldsymbol{u}_k + \boldsymbol{u}_k - \boldsymbol{w}_*\|^2 - \frac{\mu}{2}\|\boldsymbol{u}_k - \boldsymbol{w}_*\|^2 \\
&\quad = \frac{\mu}{2}\|\boldsymbol{u}_{k+1} - \boldsymbol{u}_k\|^2 + \frac{\mu}{2} \cdot \left(2 \left\langle \tau \cdot (\boldsymbol{w}_k - \boldsymbol{u}_k) - \frac{\tau}{\mu}\nabla f(\boldsymbol{w}_k), \boldsymbol{u}_k - \boldsymbol{w}_* \right\rangle \right) \\
&\quad = \frac{\mu}{2}\|\boldsymbol{u}_{k+1} - \boldsymbol{u}_k\|^2 + \tau \left\langle \nabla f(\boldsymbol{w}_k), \boldsymbol{w}_* - \boldsymbol{u}_k \right\rangle - \tau\mu \left\langle \boldsymbol{w}_k - \boldsymbol{u}_k, \boldsymbol{w}_* - \boldsymbol{u}_k \right\rangle .
\end{aligned} \tag{10.4.14}$$

From (10.4.11a) we have $\tau \boldsymbol{u}_k = (1+\tau)\boldsymbol{w}_k - \boldsymbol{v}_k$ so that

$$\tau \cdot (\boldsymbol{w}_k - \boldsymbol{u}_k) = \tau \boldsymbol{w}_k - (1+\tau)\boldsymbol{w}_k + \boldsymbol{v}_k = \boldsymbol{v}_k - \boldsymbol{w}_k \tag{10.4.15}$$

and using μ-strong convexity (10.1.15), we get

$$\begin{aligned}
\tau \left\langle \nabla f(\boldsymbol{w}_k), \boldsymbol{w}_* - \boldsymbol{u}_k \right\rangle &= \tau \left\langle \nabla f(\boldsymbol{w}_k), \boldsymbol{w}_k - \boldsymbol{u}_k \right\rangle + \tau \left\langle \nabla f(\boldsymbol{w}_k), \boldsymbol{w}_* - \boldsymbol{w}_k \right\rangle \\
&\leq \left\langle \nabla f(\boldsymbol{w}_k), \boldsymbol{v}_k - \boldsymbol{w}_k \right\rangle - \tau \cdot (f(\boldsymbol{w}_k) - f(\boldsymbol{w}_*)) - \frac{\tau\mu}{2}\|\boldsymbol{w}_k - \boldsymbol{w}_*\|^2.
\end{aligned}$$

Moreover,

$$\begin{aligned}
&-\frac{\tau\mu}{2}\|\boldsymbol{w}_k - \boldsymbol{w}_*\|^2 - \tau\mu \left\langle \boldsymbol{w}_k - \boldsymbol{u}_k, \boldsymbol{w}_* - \boldsymbol{u}_k \right\rangle \\
&\qquad = -\frac{\tau\mu}{2}\Big(\|\boldsymbol{w}_k - \boldsymbol{w}_*\|^2 - 2\left\langle \boldsymbol{w}_k - \boldsymbol{u}_k, \boldsymbol{w}_k - \boldsymbol{w}_* \right\rangle + 2\left\langle \boldsymbol{w}_k - \boldsymbol{u}_k, \boldsymbol{w}_k - \boldsymbol{u}_k \right\rangle \Big) \\
&\qquad = -\frac{\tau\mu}{2}(\|\boldsymbol{u}_k - \boldsymbol{w}_*\|^2 + \|\boldsymbol{w}_k - \boldsymbol{u}_k\|^2).
\end{aligned}$$

Thus, (10.4.14) is bounded by

$$\begin{aligned}
&\frac{\mu}{2}\|\boldsymbol{u}_{k+1} - \boldsymbol{u}_k\|^2 + \langle \nabla f(\boldsymbol{w}_k), \boldsymbol{v}_k - \boldsymbol{w}_k \rangle - \tau \cdot (f(\boldsymbol{w}_k) - f(\boldsymbol{w}_*)) \\
&- \frac{\tau\mu}{2}\|\boldsymbol{u}_k - \boldsymbol{w}_*\|^2 - \frac{\tau\mu}{2}\|\boldsymbol{w}_k - \boldsymbol{u}_k\|^2
\end{aligned}$$

which gives with $c = 1 - \tau$

$$\begin{aligned}
\frac{\mu}{2}\|\boldsymbol{u}_{k+1} - \boldsymbol{w}_*\|^2 &\leq c\frac{\mu}{2}\|\boldsymbol{u}_k - \boldsymbol{w}_*\|^2 + \frac{\mu}{2}\|\boldsymbol{u}_{k+1} - \boldsymbol{u}_k\|^2 \\
&\quad + \langle \nabla f(\boldsymbol{w}_k), \boldsymbol{v}_k - \boldsymbol{w}_k \rangle - \tau \cdot (f(\boldsymbol{w}_k) - f(\boldsymbol{w}_*)) - \frac{\tau\mu}{2}\|\boldsymbol{w}_k - \boldsymbol{u}_k\|^2.
\end{aligned} \tag{10.4.16}$$

To bound the first term in (10.4.13), we use L-smoothness (10.1.4) and (10.4.11b)

$$f(\boldsymbol{v}_{k+1}) - f(\boldsymbol{w}_k) \leq \langle \nabla f(\boldsymbol{w}_k), \boldsymbol{v}_{k+1} - \boldsymbol{w}_k \rangle + \frac{L}{2}\|\boldsymbol{v}_{k+1} - \boldsymbol{w}_k\|^2 = -\frac{1}{2L}\|\nabla f(\boldsymbol{w}_k)\|^2,$$

so that

$$f(\boldsymbol{v}_{k+1}) - f(\boldsymbol{w}_*) - \tau \cdot (f(\boldsymbol{w}_k) - f(\boldsymbol{w}_*)) \leq (1-\tau)(f(\boldsymbol{w}_k) - f(\boldsymbol{w}_*)) - \frac{1}{2L}\|\nabla f(\boldsymbol{w}_k)\|^2$$
$$= c \cdot (f(\boldsymbol{v}_k) - f(\boldsymbol{w}_*)) + c \cdot (f(\boldsymbol{w}_k) - f(\boldsymbol{v}_k)) - \frac{1}{2L}\|\nabla f(\boldsymbol{w}_k)\|^2. \tag{10.4.17}$$

Now, (10.4.16) and (10.4.17) imply

$$e_{k+1} \leq ce_k + c \cdot (f(\boldsymbol{w}_k) - f(\boldsymbol{v}_k)) - \frac{1}{2L}\|\nabla f(\boldsymbol{w}_k)\|^2 + \frac{\mu}{2}\|\boldsymbol{u}_{k+1} - \boldsymbol{u}_k\|^2$$
$$+ \langle \nabla f(\boldsymbol{w}_k), \boldsymbol{v}_k - \boldsymbol{w}_k \rangle - \frac{\tau\mu}{2}\|\boldsymbol{w}_k - \boldsymbol{u}_k\|^2.$$

Since we wish to bound e_{k+1} by ce_k, we now show that all terms except ce_k on the right-hand side of the inequality above sum up to a non-positive value. By (10.4.11c) and (10.4.15)

$$\frac{\mu}{2}\|\boldsymbol{u}_{k+1} - \boldsymbol{u}_k\|^2 = \frac{\mu}{2}\|\boldsymbol{v}_k - \boldsymbol{w}_k\|^2 - \tau \langle \nabla f(\boldsymbol{w}_k), \boldsymbol{v}_k - \boldsymbol{w}_k \rangle + \frac{\tau^2}{2\mu}\|\nabla f(\boldsymbol{w}_k)\|^2.$$

Moreover, using μ-strong convexity

$$\langle \nabla f(\boldsymbol{w}_k), \boldsymbol{v}_k - \boldsymbol{w}_k \rangle$$
$$\leq \tau \langle \nabla f(\boldsymbol{w}_k), \boldsymbol{v}_k - \boldsymbol{w}_k \rangle + (1-\tau)\Big(f(\boldsymbol{v}_k) - f(\boldsymbol{w}_k) - \frac{\mu}{2}\|\boldsymbol{v}_k - \boldsymbol{w}_k\|^2\Big).$$

Thus, we arrive at

$$e_{k+1} \leq ce_k + c \cdot (f(\boldsymbol{w}_k) - f(\boldsymbol{v}_k)) - \frac{1}{2L}\|\nabla f(\boldsymbol{w}_k)\|^2 + \frac{\mu}{2}\|\boldsymbol{v}_k - \boldsymbol{w}_k\|^2$$
$$- \tau \langle \nabla f(\boldsymbol{w}_k), \boldsymbol{v}_k - \boldsymbol{w}_k \rangle + \frac{\tau^2}{2\mu}\|\nabla f(\boldsymbol{w}_k)\|^2 + \tau \langle \nabla f(\boldsymbol{w}_k), \boldsymbol{v}_k - \boldsymbol{w}_k \rangle$$
$$+ c \cdot (f(\boldsymbol{v}_k) - f(\boldsymbol{w}_k)) - c\frac{\mu}{2}\|\boldsymbol{v}_k - \boldsymbol{w}_k\|^2 - \frac{\tau\mu}{2}\|\boldsymbol{w}_k - \boldsymbol{u}_k\|^2$$
$$= ce_k + \Big(\frac{\tau^2}{2\mu} - \frac{1}{2L}\Big)\|\nabla f(\boldsymbol{w}_k)\|^2 + \frac{\mu}{2}\Big(\tau - \frac{1}{\tau}\Big)\|\boldsymbol{w}_k - \boldsymbol{v}_k\|^2$$
$$\leq ce_k$$

where we used once more (10.4.15), and the fact that $\tau^2/(2\mu) - 1/(2L) = 0$ and $\tau - 1/\tau \leq 0$ since $\tau = \sqrt{\mu/L} \in (0, 1]$. □

Comparing the result for gradient descent (10.1.16) with NAG (10.4.12), the improvement lies in the convergence rate, which is $1-\kappa^{-1}$ for gradient descent (also see Remark 10.15), and $1-\kappa^{-1/2}$ for NAG, where $\kappa = L/\mu$. In contrast to gradient descent, for NAG the convergence depends only on the *square root* of the condition number κ. For ill-conditioned problems where κ is large, we therefore expect much better performance for accelerated methods.

Finally, we mention that NAG also achieves faster convergence in the case of L-smooth and convex objective functions. While the error decays like $O(k^{-1})$ for gradient descent, see Theorem 10.11, for NAG one obtains convergence $O(k^{-2})$, see [158, 156, 240].

10.5 Other methods

In recent years, a multitude of first order (gradient descent) methods has been proposed and studied for the training of neural networks. They typically employ (a subset) of the three critical strategies: mini-batches, acceleration, and adaptive step sizes. The concept of mini-batches and acceleration have been covered in the previous sections, and we will touch upon adaptive learning rates in the present one. Specifically, we present three algorithms—AdaGrad, RMSProp, and Adam—which have been among the most influential in the field, and serve to explore the main ideas. An intuitive overview of first order methods can also be found in [192], which discusses additional variants that are omitted here. Moreover, in practice, various other techniques and heuristics such as batch normalization, gradient clipping, data augmentation, regularization and dropout, early stopping, specific weight initializations etc. are used. We do not discuss them here, and refer to [22] or to [66, Chapter 11] for a practitioners guide.

After initializing $\boldsymbol{m}_0 = \mathbf{0} \in \mathbb{R}^n$, $\boldsymbol{v}_0 = \mathbf{0} \in \mathbb{R}^n$, and $\boldsymbol{w}_0 \in \mathbb{R}^n$, all methods discussed below are special cases of the update

$$\boldsymbol{m}_{k+1} = \beta_1 \boldsymbol{m}_k + \beta_2 \nabla f(\boldsymbol{w}_k) \tag{10.5.1a}$$

$$\boldsymbol{v}_{k+1} = \gamma_1 \boldsymbol{v}_k + \gamma_2 \nabla f(\boldsymbol{w}_k) \odot \nabla f(\boldsymbol{w}_k) \tag{10.5.1b}$$

$$\boldsymbol{w}_{k+1} = \boldsymbol{w}_k - \alpha_k \boldsymbol{m}_{k+1} \oslash \sqrt{\boldsymbol{v}_{k+1} + \varepsilon} \tag{10.5.1c}$$

for $k \in \mathbb{N}_0$, and certain hyperparameters α_k, β_1, β_2, γ_1, γ_2, and ε. Here $\odot$ and $\oslash$ denote the componentwise multiplication and division, respectively, and $\sqrt{\boldsymbol{v}_{k+1} + \varepsilon}$ is understood as the vector $(\sqrt{v_{k+1,i} + \varepsilon})_i$. We will give some default values for those hyperparameters in the following, but mention that careful problem dependent tuning can significantly improve the performance. Equation (10.5.1a) corresponds to heavy ball momentum if $\beta_1 > 0$. If $\beta_1 = 0$, then $\boldsymbol{m}_{k+1}$ is simply a multiple of the current gradient. Equation (10.5.1b) defines a weight vector $\boldsymbol{v}_{k+1}$ that is used to set the componentwise learning rate in the update of the parameter in (10.5.1c). These type of methods are often applied using mini-batches, see Section 10.2. For simplicity we present them with the full gradients.

10.5.1 AdaGrad

In Section 10.2 we argued, that for stochastic methods the learning rate should decrease in order to get convergence. The choice of how to decrease the learning rate can significantly impact performance. AdaGrad [56], which stands for adaptive gradient algorithm, provides a method to dynamically adjust learning rates during optimization. Moreover, it does so by using individual learning rates for each component.

AdaGrad correspond to (10.5.1) with

$$\beta_1 = 0, \qquad \gamma_1 = \beta_2 = \gamma_2 = 1, \qquad \alpha_k = \alpha \quad \text{for all } k \in \mathbb{N}_0.$$

This leaves the hyperparameters $\varepsilon > 0$ and $\alpha > 0$. The constant $\varepsilon > 0$ is chosen small but positive to avoid division by zero in (10.5.1c). Possible default values are $\alpha = 0.01$ and $\varepsilon = 10^{-8}$. The AdaGrad update then reads

$$\begin{aligned} \boldsymbol{v}_{k+1} &= \boldsymbol{v}_k + \nabla f(\boldsymbol{w}_k) \odot \nabla f(\boldsymbol{w}_k) \\ \boldsymbol{w}_{k+1} &= \boldsymbol{w}_k - \alpha \nabla f(\boldsymbol{w}_k) \oslash \sqrt{\boldsymbol{v}_{k+1} + \varepsilon}. \end{aligned}$$

Due to

$$\boldsymbol{v}_{k+1} = \sum_{j=0}^{k} \nabla f(\boldsymbol{w}_j) \odot \nabla f(\boldsymbol{w}_j), \tag{10.5.2}$$

the algorithm scales the gradient $\nabla f(\boldsymbol{w}_k)$ in the update component-wise by the inverse square root of the sum over all past squared gradients plus ε. Note that the scaling factor $(v_{k+1,i} + \varepsilon)^{-1/2}$ for component i will be large, if the previous gradients for that component were small, and vice versa. In the words of the authors of [56]: "our procedures give frequently occurring features very low learning rates and infrequent features high learning rates."

Remark 10.26. A benefit of the componentwise scaling can be observed for the ill-conditioned objective function in (10.4.1). Since in this case $\nabla f(\boldsymbol{w}_j) = (\lambda_1 w_{j,1}, \lambda_2 w_{j,2})^\top$ for each $j = 1, \dots, k$, setting $\varepsilon = 0$ AdaGrad performs the update

$$\boldsymbol{w}_{k+1} = \boldsymbol{w}_k - \alpha \begin{pmatrix} w_{k,1}(\sum_{j=0}^{k} w_{j,1}^2)^{-1/2} \\ w_{k,2}(\sum_{j=0}^{k} w_{j,1}^2)^{-1/2} \end{pmatrix}.$$

Note how the λ_1 and λ_2 factors in the update have vanished due to the division by $\sqrt{\boldsymbol{v}_{k+1}}$. This makes the method invariant to a componentwise rescaling of the gradient, and results in a more direct path towards the minimizer.

10.5.2 RMSProp

The sum of past squared gradients can increase rapidly, leading to a significant reduction in learning rates when training neural networks with AdaGrad. This often results in slow convergence, see for example [241]. RMSProp [89] seeks to rectify this by adjusting the learning rates using an exponentially weighted average of past gradients.

RMSProp corresponds to (10.5.1) with

$$\beta_1 = 0, \qquad \beta_2 = 1, \qquad \gamma_2 = 1 - \gamma_1 \in (0, 1), \qquad \alpha_k = \alpha \quad \text{for all } k \in \mathbb{N}_0,$$

effectively leaving the hyperparameters $\varepsilon > 0$, $\gamma_1 \in (0, 1)$ and $\alpha > 0$. Typically, recommended default values are $\varepsilon = 10^{-8}$, $\alpha = 0.01$ and $\gamma_1 = 0.9$. The algorithm is given through

$$\boldsymbol{v}_{k+1} = \gamma_1 \boldsymbol{v}_k + (1 - \gamma_1) \nabla f(\boldsymbol{w}_k) \odot \nabla f(\boldsymbol{w}_k) \tag{10.5.3a}$$

$$\boldsymbol{w}_{k+1} = \boldsymbol{w}_k - \alpha \nabla f(\boldsymbol{w}_k) \oslash \sqrt{\boldsymbol{v}_{k+1} + \varepsilon}. \tag{10.5.3b}$$

Note that

$$\boldsymbol{v}_{k+1} = (1 - \gamma_1) \sum_{j=0}^{k} \gamma_1^j \nabla f(\boldsymbol{w}_{k-j}) \odot \nabla f(\boldsymbol{w}_{k-j}),$$

so that, contrary to AdaGrad (10.5.2), the influence of gradient $\nabla f(\boldsymbol{w}_{k-j})$ on the weight $\boldsymbol{v}_{k+1}$ decays exponentially in j.

10.5.3 Adam

Adam [114], short for adaptive moment estimation, combines adaptive learning rates based on exponentially weighted averages as in RMSProp, with heavy ball momentum. Contrary to AdaGrad an RMSProp it thus uses a value $\beta_1 > 0$.

More precisely, Adam corresponds to (10.5.1) with

$$\beta_2 = 1 - \beta_1 \in (0,1), \qquad \gamma_2 = 1 - \gamma_1 \in (0,1), \qquad \alpha_k = \alpha \frac{\sqrt{1-\gamma_1^{k+1}}}{1-\beta_1^{k+1}}$$

for all $k \in \mathbb{N}_0$, for some $\alpha > 0$. The default values for the remaining parameters recommended in [114] are $\varepsilon = 10^{-8}$, $\alpha = 0.001$, $\beta_1 = 0.9$ and $\gamma_1 = 0.999$. The update can be formulated as

$$\boldsymbol{m}_{k+1} = \beta_1 \boldsymbol{m}_k + (1-\beta_1)\nabla f(\boldsymbol{w}_k), \qquad \hat{\boldsymbol{m}}_{k+1} = \frac{\boldsymbol{m}_{k+1}}{1-\beta_1^{k+1}} \tag{10.5.4a}$$

$$\boldsymbol{v}_{k+1} = \gamma_1 \boldsymbol{v}_k + (1-\gamma_1)\nabla f(\boldsymbol{w}_k) \odot \nabla f(\boldsymbol{w}_k), \qquad \hat{\boldsymbol{v}}_{k+1} = \frac{\boldsymbol{v}_{k+1}}{1-\gamma_1^{k+1}} \tag{10.5.4b}$$

$$\boldsymbol{w}_{k+1} = \boldsymbol{w}_k - \alpha \hat{\boldsymbol{m}}_{k+1} \oslash \sqrt{\hat{\boldsymbol{v}}_{k+1} + \varepsilon}. \tag{10.5.4c}$$

Note that $\boldsymbol{m}_{k+1}$ equals

$$\boldsymbol{m}_{k+1} = (1-\beta_1)\sum_{j=0}^{k} \beta_1^j \nabla f(\boldsymbol{w}_{k-j})$$

and thus correspond to heavy ball style momentum with momentum parameter $\beta = \beta_1$, see (10.4.6). The normalized version $\hat{\boldsymbol{m}}_{k+1}$ is introduced to account for the bias towards $\mathbf{0}$, stemming from the initialization $\boldsymbol{m}_0 = \mathbf{0}$. The weight-vector $\boldsymbol{v}_{k+1}$ in (10.5.4b) is analogous to the exponentially weighted average of RMSProp in (10.5.3a), and the normalization again serves to counter the bias from $\boldsymbol{v}_0 = \mathbf{0}$.

It should be noted that there exist examples of convex functions for which Adam does *not converge to a minimizer*, see [188]. The authors of [188] propose a modification termed AMSGrad, which avoids this issue and their analysis also applies to RMSProp. Nonetheless, Adam remains a highly popular and successful algorithm for the training of neural networks. We also mention that the proof of convergence in the stochastic setting requires k-dependent decreasing learning rates such as $\alpha = O(k^{-1/2})$ in (10.5.3b) and (10.5.4c).

Bibliography and further reading

Section 10.1 on gradient descent is based on standard textbooks such as [20, 25, 161] and especially [157]. These are also good references for further reading on convex optimization. In particular Theorem 10.11 and the Lemmas leading up to it closely follow Nesterov's arguments in [157]. Convergence proofs under the PL inequality can be found in [112]. Stochastic gradient descent discussed in Section 10.2 originally dates back to Robbins and Monro [189]. The first non-asymptotic convergence analysis for strongly convex objective functions was given in [152]. The proof presented here is similar to [75] and in particular uses their choice of step size. A good overview of proofs for (stochastic) gradient descent algorithms together with detailed references can be found in [64], and

for a textbook specifically on stochastic optimization also see [124]. The backpropagation algorithm discussed in Section 10.3 was first introduced by Rumelhart, Hinton and Williams in [196]; for a more detailed discussion, see for instance [83]. The heavy ball method in Section 10.4 goes back to Polyak [178]. To motivate the algorithm we proceed similar as in [69, 179, 181]. For the analysis of Nesterov acceleration [158], we follow the Lyapunov type proofs given in [230, 240]. Finally, for Section 10.5 on other algorithms, we refer to the original works that introduced AdaGrad [56], RMSProp [89] and Adam [114]. A good overview of gradient descent methods popular for deep learning can be found in [192]. Regarding the analysis of RMSProp and Adam, we refer to [188] which gave an example of a convex function for which Adam does not converge, and provide a provably convergent modification of the algorithm. Convergence proofs (for variations of) AdaGrad and Adam can also be found in [48].

For a general discussion and analysis of optimization algorithms in machine learning see [23]. Details on implementations in Python can for example be found in [66], and for recommendations and tricks regarding the implementation we also refer to [22, 127].

Exercises

Exercise 10.27. Let $f \in C^1(\mathbb{R}^n)$. Show that f is convex in the sense of Definition 10.8 if and only if

$$f(\boldsymbol{w}) + \langle \nabla f(\boldsymbol{w}), \boldsymbol{v} - \boldsymbol{w} \rangle \leq f(\boldsymbol{v}) \qquad \text{for all } \boldsymbol{w}, \boldsymbol{v} \in \mathbb{R}^n.$$

Exercise 10.28. Find a function $f : \mathbb{R} \to \mathbb{R}$ that is L-smooth, satisfies the PL-inequality (10.1.19) for some $\mu > 0$, has a unique minimizer $w_* \in \mathbb{R}$, but is not convex and thus also not strongly convex.

Exercise 10.29. Prove Theorem 10.17, i.e. show that L-smoothness and the PL-inequality (10.1.19) yield linear convergence of $f(\boldsymbol{w}_k) \to f(\boldsymbol{w}_*)$ as $k \to \infty$.

Definition 10.30. For convex $f : \mathbb{R}^n \to \mathbb{R}$, $\boldsymbol{g} \in \mathbb{R}^n$ is called a **subgradient** (or subdifferential) of f at $\boldsymbol{v}$ if and only if

$$f(\boldsymbol{w}) \geq f(\boldsymbol{v}) + \langle \boldsymbol{g}, \boldsymbol{w} - \boldsymbol{v} \rangle \qquad \text{for all } \boldsymbol{w} \in \mathbb{R}^n. \tag{10.5.5}$$

The set of all subgradients of f at $\boldsymbol{v}$ is denoted by $\partial f(\boldsymbol{v})$.

A subgradient always exists, i.e. $\partial f(\boldsymbol{v})$ is necessarily nonempty. This statement is also known under the name "Hyperplane separation theorem". Subgradients generalize the notion of gradients for convex functions, since for any convex continuously differentiable f, (10.5.5) is satisfied with $\boldsymbol{g} = \nabla f(\boldsymbol{v})$.

Exercise 10.31. Let $f : \mathbb{R}^n \to \mathbb{R}$ be convex and $\mathrm{Lip}(f) \leq L$. Show that for any $\boldsymbol{g} \in \partial f(\boldsymbol{v})$ holds $\|\boldsymbol{g}\| \leq L$.

Exercise 10.32. Let $f : \mathbb{R}^n \to \mathbb{R}$ be convex, $\mathrm{Lip}(f) \leq L$ and suppose that $\boldsymbol{w}_*$ is a minimizer of f. Fix $\boldsymbol{w}_0 \in \mathbb{R}^d$, and for $k \in \mathbb{N}_0$ define the **subgradient descent** update

$$\boldsymbol{w}_{k+1} := \boldsymbol{w}_k - h_k \boldsymbol{g}_k,$$

where $\boldsymbol{g}_k$ is an arbitrary fixed element of $\partial f(\boldsymbol{w}_k)$. Show that

$$\min_{i \leq k} f(\boldsymbol{w}_i) - f(\boldsymbol{w}_*) \leq \frac{\|\boldsymbol{w}_0 - \boldsymbol{w}_*\|^2 + L^2 \sum_{i=1}^{k} h_i^2}{2 \sum_{i=1}^{k} h_i}.$$

Hint: Start by recursively expanding $\|\boldsymbol{w}_k - \boldsymbol{w}_*\|^2 = \cdots$, and then apply the property of the subgradient.

Exercise 10.33. Consider the setting of Exercise 10.32. Determine step sizes $h_1, \ldots, h_k$ (which may depend on k, i.e. $h_{k,1}, \ldots, h_{k,k}$) such that for any arbitrarily small $\delta > 0$

$$\min_{i \leq k} f(\boldsymbol{w}_i) - f(\boldsymbol{w}_*) = O(k^{-1/2+\delta}) \qquad \text{as } k \to \infty.$$

Exercise 10.34. Let $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ be symmetric positive semidefinite, $\boldsymbol{b} \in \mathbb{R}^n$ and $c \in \mathbb{R}$. Denote the eigenvalues of $\boldsymbol{A}$ by $\lambda_1 \geq \cdots \geq \lambda_n \geq 0$. Show that the objective function

$$f(\boldsymbol{w}) := \frac{1}{2}\boldsymbol{w}^\top \boldsymbol{A}\boldsymbol{w} + \boldsymbol{b}^\top \boldsymbol{w} + c \tag{10.5.6}$$

is convex and λ_1-smooth. Moreover, if $\lambda_n > 0$, then f is λ_n-strongly convex. Show that these values are optimal in the sense that f is neither L-smooth nor μ-strongly convex if $L < \lambda_1$ and $\mu > \lambda_n$.

Hint: Note that L-smoothness and μ-strong convexity are invariant under shifts and the addition of constants. That is, for every $\alpha \in \mathbb{R}$ and $\boldsymbol{\beta} \in \mathbb{R}^n$, $\tilde{f}(\boldsymbol{w}) := \alpha + f(\boldsymbol{w}+\boldsymbol{\beta})$ is L-smooth or μ-strongly convex if and only if f is. It thus suffices to consider $\boldsymbol{w}^\top \boldsymbol{A}\boldsymbol{w}/2$.

Exercise 10.35. Let f be as in Exercise 10.34. Show that gradient descent converges for arbitrary initialization $\boldsymbol{w}_0 \in \mathbb{R}^n$, if and only if

$$\max_{j=1,\ldots,n} |1 - h\lambda_j| < 1.$$

Show that $\operatorname{argmin}_{h>0} \max_{j=1,\ldots,n} |1 - h\lambda_j| = 2/(\lambda_1 + \lambda_n)$ and conclude that the convergence will be slow if f is ill-conditioned, i.e. if $\lambda_1/\lambda_n \gg 1$.

Hint: Assume first that $\boldsymbol{b} = \boldsymbol{0} \in \mathbb{R}^n$ and $c = 0 \in \mathbb{R}$ in (10.5.6), and use the singular value decomposition $\boldsymbol{A} = \boldsymbol{U}^\top \operatorname{diag}(\lambda_1, \ldots, \lambda_n)\boldsymbol{U}$.

Exercise 10.36. Show that (10.4.10) can equivalently be written as (10.4.11) with $\tau = \sqrt{\mu/L}$, $\alpha = 1/L$, $\beta = (1-\tau)/(1+\tau)$ and the initialization $\boldsymbol{u}_0 = ((1+\tau)\boldsymbol{w}_0 - \boldsymbol{v}_0)/\tau$.