Chapter 11
Deep Learning Optimization

11.1 Introduction

In Chap. 6, we discussed various optimization methods for deep neural network training. Although they are in various forms, these algorithms are basically gradient-based local update schemes. However, the biggest obstacle recognized by the entire community is that the loss surfaces of deep neural networks are extremely non-convex and not even smooth. This non-convexity and non-smoothness make the optimization unaffordable to analyze, and the main concern was whether popular gradient-based approaches might fall into local minimizers.

Surprisingly, the success of modern deep learning may be due to the remarkable effectiveness of gradient-based optimization methods despite its highly non-convex nature of the optimization problem. Extensive research has been carried out in recent years to provide a theoretical understanding of this phenomenon. In particular, several recent works [119–121] have noted the importance of the over-parameterization. In fact, it was shown that when hidden layers of a deep network have a large number of neurons compared to the number of training samples, the gradient descent or stochastic gradient converges to a global minimum with zero training errors. While these results are intriguing and provide important clues for understanding the geometry of deep learning optimization, it is still unclear why simple local search algorithms can be successful for deep neural network training.

Indeed, the area of deep learning optimization is a rapidly evolving area of intense research, and there are too many different approaches to cover in a single chapter. Rather than treating a variety of techniques in a disorganized way, this chapter explains two different lines of research just for food for thought: one is based on the geometric structure of the loss function and the other is based on the results of Lyapunov stability. Although the two approaches are closely related, they have different advantages and disadvantages. By explaining these two approaches, we can cover some of the key topics of research exploration such as optimization landscape [122–124], over-parameterization [119, 125–129], and neural tangent kernel (NTK)

J. C. Ye, *Geometry of Deep Learning*, Mathematics in Industry 37,
https://doi.org/10.1007/978-981-16-6046-7_11

[130–132] that have been used extensively to analyze the convergence properties of local deep learning search methods.

11.2 Problem Formulation

In Chap. 6, we pointed out that the basic optimization problem in neural network training can be formulated as

$$\min_{\boldsymbol{\theta}\in\mathbb{R}^n} \ell(\boldsymbol{\theta}), \tag{11.1}$$

where $\boldsymbol{\theta}$ refers to the network parameters and $\ell : \mathbb{R}^n \mapsto \mathbb{R}$ is the loss function. In the case of supervised learning with the mean square error (MSE) loss, the loss function is defined by

$$\ell(\boldsymbol{\theta}) := \frac{1}{2}\|\boldsymbol{y} - \boldsymbol{f}_{\boldsymbol{\theta}}(\boldsymbol{x})\|^2, \tag{11.2}$$

where $\boldsymbol{x}$, $\boldsymbol{y}$ denotes the pair of the network input and the label, and $\boldsymbol{f}_{\boldsymbol{\theta}}(\cdot)$ is a neural network parameterized by trainable parameters $\boldsymbol{\theta}$. For the case of an L-layer feed-forward neural network, the regression function $\boldsymbol{f}_{\Theta}(\boldsymbol{x})$ can be represented by

$$\boldsymbol{f}_{\boldsymbol{\theta}}(\boldsymbol{x}) := \left(\boldsymbol{\sigma} \circ \boldsymbol{g}^{(L)} \circ \boldsymbol{\sigma} \circ \boldsymbol{g}^{(L-1)} \cdots \circ \boldsymbol{g}^{(1)}\right)(\boldsymbol{x}), \tag{11.3}$$

where $\boldsymbol{\sigma}(\cdot)$ denotes the element-wise nonlinearity and

$$\boldsymbol{g}^{(l)} = \boldsymbol{W}^{(l)}\boldsymbol{o}^{(l-1)} + \boldsymbol{b}^{(l-1)}, \tag{11.4}$$

$$\boldsymbol{o}^{(l)} = \boldsymbol{\sigma}(\boldsymbol{g}^{(l)}), \tag{11.5}$$

$$\boldsymbol{o}^{(0)} = \boldsymbol{x}, \tag{11.6}$$

for $l = 1, \cdots, L$. Here, the number of the l-th layer hidden neurons, often referred to as the width, is denoted by $d^{(l)}$, so that $\boldsymbol{g}^{(l)}, \boldsymbol{o}^{(l)} \in \mathbb{R}^{d^{(l)}}$ and $\boldsymbol{W}^{(l)} \in \mathbb{R}^{d^{(l)}\times d^{(l-1)}}$.

The popular local search approaches using the gradient descent use the following update rule:

$$\boldsymbol{\theta}[k+1] = \boldsymbol{\theta}[k] - \eta_k \left.\frac{\partial \ell(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}}\right|_{\boldsymbol{\theta}=\boldsymbol{\theta}[k]}, \tag{11.7}$$

where η_k denotes the k-th iteration step size. In a differential equation form, the update rule can be represented by

$$\dot{\boldsymbol{\theta}}[t] = -\frac{\partial \ell(\boldsymbol{\theta}[k])}{\partial \boldsymbol{\theta}}, \tag{11.8}$$

where $\dot{\boldsymbol{\theta}}[t] = \partial \boldsymbol{\theta}[t]/\partial t$.

As previously explained, the optimization problem (11.1) is strongly non-convex, and it is known that the gradient-based local search schemes using (11.7) and (11.8) may get stuck in the local minima. Interestingly, many deep learning optimization algorithms appear to avoid the local minima and even result in zero training errors, indicating that the algorithms are reaching the global minima. In the following, we present two different approaches to explain this fascinating behavior of gradient descent approaches.

11.3 Polyak–Łojasiewicz-Type Convergence Analysis

The loss function ℓ is said to be strongly convex (SC) if

$$\ell(\boldsymbol{\theta}') \geq \ell(\boldsymbol{\theta}) + \langle \nabla \ell(\boldsymbol{\theta}), \boldsymbol{\theta}' - \boldsymbol{\theta} \rangle + \frac{\mu}{2}\|\boldsymbol{\theta}' - \boldsymbol{\theta}\|^2, \quad \forall \boldsymbol{\theta}, \boldsymbol{\theta}'. \tag{11.9}$$

It is known that if ℓ is SC, then gradient descent achieves a global linear convergence rate for this problem [133]. Note that SC in (11.9) is a stronger condition than the convexity in Proposition 1.1, which is given as

$$\ell(\boldsymbol{\theta}') \geq \ell(\boldsymbol{\theta}) + \langle \nabla \ell(\boldsymbol{\theta}), \boldsymbol{\theta}' - \boldsymbol{\theta} \rangle, \quad \forall \boldsymbol{\theta}, \boldsymbol{\theta}'. \tag{11.10}$$

Our starting point is the observation that the convex analysis mentioned above is not the right approach to analyzing a deep neural network. The non-convexity is essential for the analysis. This situation has motivated a variety of alternatives to the convexity to prove the convergence. One of the oldest of these conditions is the error bounds (EB) of Luo and Tseng [134], but other conditions have been recent considered, which include essential strong convexity (ESC) [135], weak strong convexity (WSC) [136], and the restricted secant inequality (RSI) [137]. See their specific forms of conditions in Table 11.1. On the other hand, there is a much older condition called the Polyak–Łojasiewicz (PL) condition, which was originally introduced by Polyak [138] and found to be a special case of the inequality of Łojasiewicz [139]. Specifically, we will say that a function satisfies the PL inequality if the following holds for some $\mu > 0$:

$$\frac{1}{2}\|\nabla \ell(\boldsymbol{\theta})\|^2 \geq \mu(\ell(\boldsymbol{\theta}) - \ell^*), \quad \forall \boldsymbol{\theta}. \tag{11.11}$$

Table 11.1 Examples of conditions for gradient descent (GD) convergence. All of these definitions involve some constant $\mu > 0$ (which may not be the same across conditions). $\boldsymbol{\theta}_p$ denotes the projection of $\boldsymbol{\theta}$ onto the solution set $\mathcal{X}^*$, and ℓ^* refers to the minimum cost

Name	Conditions	
Strong convexity (SC)	$\ell(\boldsymbol{\theta}') \geq \ell(\boldsymbol{\theta}) + \langle \nabla \ell(\boldsymbol{\theta}), \boldsymbol{\theta}' - \boldsymbol{\theta} \rangle + \frac{\mu}{2} \|\boldsymbol{\theta}' - \boldsymbol{\theta}\|^2,$	$\forall \boldsymbol{\theta}, \boldsymbol{\theta}'$
Essential strong convexity (ESC)	$\ell(\boldsymbol{\theta}') \geq \ell(\boldsymbol{\theta}) + \langle \nabla \ell(\boldsymbol{\theta}), \boldsymbol{\theta}' - \boldsymbol{\theta} \rangle + \frac{\mu}{2} \|\boldsymbol{\theta}' - \boldsymbol{\theta}\|^2,$	$\forall \boldsymbol{\theta}, \boldsymbol{\theta}' \; s.t. \, \boldsymbol{\theta}_p = \boldsymbol{\theta}'_p$
Weak strong convexity (WSC)	$\ell^* \geq \ell(\boldsymbol{\theta}) + \langle \nabla \ell(\boldsymbol{\theta}), \boldsymbol{\theta}_p - \boldsymbol{\theta} \rangle + \frac{\mu}{2} \|\boldsymbol{\theta}_p - \boldsymbol{\theta}\|^2,$	$\forall \boldsymbol{\theta}$
Restricted secant inequality (RSI)	$\langle \nabla \ell(\boldsymbol{\theta}), \boldsymbol{\theta} - \boldsymbol{\theta}_p \rangle \geq \mu \|\boldsymbol{\theta}_p - \boldsymbol{\theta}\|^2,$	$\forall \boldsymbol{\theta}$
Error bound (EB)	$\|\nabla \ell(\boldsymbol{\theta})\| \geq \mu \|\boldsymbol{\theta}_p - \boldsymbol{\theta}\|^2,$	$\forall \boldsymbol{\theta}$
Polyak–Łojasiewicz (PL)	$\frac{1}{2} \|\nabla \ell(\boldsymbol{\theta})\|^2 \geq \mu(\ell(\boldsymbol{\theta}) - \ell^*),$	$\forall \boldsymbol{\theta}$

Note that this inequality implies that every stationary point is a global minimum. But unlike SC, it does not imply that there is a unique solution. We will revisit this issue later.

Similar to other conditions in Table 11.1, PL is a sufficient condition for gradient descent to achieve a linear convergence rate [122]. In fact, PL is the mildest condition among them. Specifically, the following relationship between the conditions holds [122]:

$$(SC) \to (ESC) \to (WSC) \to (RSI) \to (EB) \equiv (PL),$$

if ℓ have a Lipschitz continuous gradient, i.e. there exists $L > 0$ such that

$$\|\nabla\ell(\boldsymbol{\theta}) - \nabla\ell(\boldsymbol{\theta}')\| \leq L\|\boldsymbol{\theta} - \boldsymbol{\theta}'\|, \quad \forall\boldsymbol{\theta}, \boldsymbol{\theta}'. \tag{11.12}$$

In the following, we provide a convergence proof of the gradient descent method using the PL condition, which turns out to be an important tool for non-convex deep learning optimization problems.

Theorem 11.1 (Karimi et al. [122]) *Consider problem* (11.1)*, where ℓ has an L-Lipschitz continuous gradient, a non-empty solution set, and satisfies the PL inequality* (11.11)*. Then the gradient method with a step-size of* $1/L$*:*

$$\boldsymbol{\theta}[k+1] = \boldsymbol{\theta}[k] - \frac{1}{L}\nabla\ell(\boldsymbol{\theta}[k]) \tag{11.13}$$

has a global convergence rate

$$\ell(\boldsymbol{\theta}[k]) - \ell^* \leq \left(1 - \frac{\mu}{L}\right)^k \left(\ell(\boldsymbol{\theta}[0]) - \ell^*\right).$$

Proof Using Lemma 11.1 (see next section), L-Lipschitz continuous gradient of the loss function ℓ implies that the function

$$g(\boldsymbol{\theta}) = \frac{L}{2}\|\boldsymbol{\theta}\|^2 - \ell(\boldsymbol{\theta})$$

is convex. Thus, the first-order equivalence of convexity in Proposition 1.1 leads to the following:

$$\begin{aligned}\frac{L}{2}\|\boldsymbol{\theta}'\|^2 - \ell(\boldsymbol{\theta}') &\geq \frac{L}{2}\|\boldsymbol{\theta}\|^2 - \ell(\boldsymbol{\theta}) + \langle\boldsymbol{\theta}' - \boldsymbol{\theta}, L\boldsymbol{\theta} - \nabla\ell(\boldsymbol{\theta})\rangle \\ &= -\frac{L}{2}\|\boldsymbol{\theta}\|^2 - \ell(\boldsymbol{\theta}) + L\langle\boldsymbol{\theta}', \boldsymbol{\theta}\rangle - \langle\boldsymbol{\theta}' - \boldsymbol{\theta}, \nabla\ell(\boldsymbol{\theta})\rangle.\end{aligned}$$

By arranging terms, we have

$$\ell(\boldsymbol{\theta}') \le \ell(\boldsymbol{\theta}) + \langle \nabla \ell(\boldsymbol{\theta}), \boldsymbol{\theta}' - \boldsymbol{\theta} \rangle + \frac{L}{2}\|\boldsymbol{\theta}' - \boldsymbol{\theta}\|^2, \quad \forall \boldsymbol{\theta}, \boldsymbol{\theta}'.$$

By setting $\boldsymbol{\theta}' = \boldsymbol{\theta}[k+1]$ and $\boldsymbol{\theta} = \boldsymbol{\theta}[k]$ and using the update rule (11.13), we have

$$\ell(\boldsymbol{\theta}[k+1]) - \ell(\boldsymbol{\theta}[k]) \le -\frac{1}{2L}\|\nabla \ell(\boldsymbol{\theta}[k])\|^2. \tag{11.14}$$

Using the PL inequality (11.11), we get

$$\ell(\boldsymbol{\theta}[k+1]) - \ell(\boldsymbol{\theta}[k]) \le -\frac{\mu}{L}\left(\ell(\boldsymbol{\theta}[k]) - \ell^*\right).$$

Rearranging and subtracting ℓ^* from both sides gives us

$$\ell(\boldsymbol{\theta}[k+1]) - \ell^* \le \left(1 - \frac{\mu}{L}\right)\left(\ell(\boldsymbol{\theta}[k]) - \ell^*\right).$$

Applying this inequality recursively gives the result. □

The beauty of this proof is that we can replace the long and complicated proofs from other conditions with simpler proofs based on the PL inequality [122].

11.3.1 Loss Landscape and Over-Parameterization

In Theorem 11.1, we use the two conditions for the loss function: (1) ℓ satisfies the PL condition and (2) the gradient of ℓ is Lipschitz continuous. Although these conditions are much weaker than the convexity of the loss function, they still impose the geometric constraint for the loss function, which deserves further discussion.

Lemma 11.1 *If the gradient of $\ell(\boldsymbol{\theta})$ satisfies the L-Lipschitz condition in (11.12), then the transformed function $g : \mathbb{R}^n \mapsto \mathbb{R}$ defined by*

$$g(\boldsymbol{\theta}) := \frac{L}{2}\boldsymbol{\theta}^\top \boldsymbol{\theta} - \ell(\boldsymbol{\theta}) \tag{11.15}$$

is convex.

Proof Using the Cauchy–Schwarz inequality, (11.12) implies

$$\langle \nabla \ell(\boldsymbol{\theta}) - \nabla \ell(\boldsymbol{\theta}'), \boldsymbol{\theta} - \boldsymbol{\theta}' \rangle \le L\|\boldsymbol{\theta} - \boldsymbol{\theta}'\|^2, \quad \forall \boldsymbol{\theta}, \boldsymbol{\theta}'.$$

This is equivalent to the following condition:

$$\langle \boldsymbol{\theta}' - \boldsymbol{\theta}, \nabla g(\boldsymbol{\theta}') - \nabla g(\boldsymbol{\theta}) \rangle \geq 0, \quad \forall \boldsymbol{\theta}, \boldsymbol{\theta}', \tag{11.16}$$

where

$$g(\boldsymbol{\theta}) = \frac{L}{2}\|\boldsymbol{\theta}\|^2 - \ell(\boldsymbol{\theta}).$$

Thus, using the monotonicity of gradient equivalence in Proposition 1.1, we can show that $g(\boldsymbol{\theta})$ is convex. □

Lemma 11.1 implies that although ℓ is not convex, its transformed function by (11.15) can be convex. Figure 11.1a shows an example of such case. Another important geometric consideration for the loss landscape comes from the PL condition. More specifically, the PL condition in (11.11) implies that every stationary point is a global minimizer, although the global minimizers may not be unique, as shown in Fig. 11.1b,c. While the PL inequality does not imply convexity of ℓ, it does imply the weaker condition of *invexity* [122]. A function is invex if it is differentiable and there exists a vector-valued function η such that for any $\boldsymbol{\theta}$ and $\boldsymbol{\theta}'$ in $\mathbb{R}^n$, the following inequality holds:

$$\ell(\boldsymbol{\theta}') \geq \ell(\boldsymbol{\theta}) + \langle \nabla \ell(\boldsymbol{\theta}), \eta(\boldsymbol{\theta}, \boldsymbol{\theta}') \rangle. \tag{11.17}$$

A convex function is a special case of invex functions since (11.17) holds when we set $\eta(\boldsymbol{\theta}, \boldsymbol{\theta}') = \boldsymbol{\theta}' - \boldsymbol{\theta}$. It was shown that a smooth ℓ is invex if and only if every stationary point of ℓ is a global minimum [140]. As the PL condition implies that every stationary point is a global minimizer, a function satisfying PL is an invex function. The inclusion relationship between convex, invex, and PL functions is illustrated in Fig. 11.2.

The loss landscape, where every stationary point is a global minimizer, implies that that there are no spurious local minimizers. This is often called the *benign* optimization landscape. Finding the conditions for a benign optimization landscape

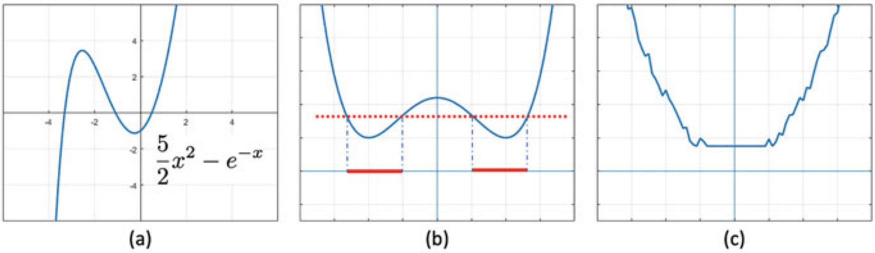

Fig. 11.1 Loss landscape for the function $\ell(x)$ with (**a**) (11.15) is convex, and (**b**, **c**) PL conditions

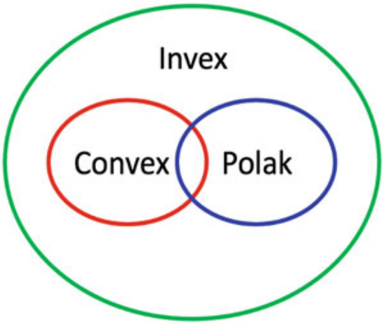

Fig. 11.2 Inclusion relationship between invex, convex and PL-type functions

of neural networks was an important theoretical interest of the theorists in machine learning. Originally observed by Kawaguch [141], Lu and Kawaguchi [142] and Zhou and Liang [143] have proven that the loss surfaces of linear neural networks, whose activation functions are all linear functions, do not have any spurious local minima under some conditions and all local minima are equally good.

Unfortunately, this good property no longer stands when the activations are nonlinear. Zhou and Liang [143] show that ReLU neural networks with one hidden layer have spurious local minima. Yun et al. [144] prove that ReLU neural networks with one hidden layer have infinitely many spurious local minima when the outputs are one-dimensional.

These somewhat negative results were surprising and seemed to contradict the empirical success of optimization in neural networks. Indeed, it was later shown that if the activation functions are continuous, and the loss functions are convex and differentiable, over-parameterized fully-connected deep neural networks do not have any spurious local minima [145].

The reason for the benign optimization landscape for an over-parameterized neural network was analyzed by examining the geometry of the global minimum. Nguyen [123] discovered that the global minima are interconnected and concentrated on a unique valley if the neural networks are sufficiently over-parameterized. Similar results were obtained by Liu et al. [124]. In fact, they found that the set of solutions of an over-parameterized system is generically a manifold of positive dimensions, with the Hessian matrices of the loss function being positive semidefinite but not positive definite. Such a landscape is incompatible with convexity unless the set of solutions is a linear manifold. However, the linear manifold with zero curvature of the curve of global minima is unlikely to occur due to the essential non-convexity of the underlying optimization problem. Hence, gradient type algorithms can converge to any of the global minimum, although the exact point of the convergence depends on a specific optimization algorithm. This *implicit bias* of an optimization algorithm is another important theoretical topic in deep learning, which will be covered in a later chapter. In contrast, an under-parameterized landscape generally has several isolated local minima with a positive definite Hessian of the loss, the function being locally convex. This is illustrated in Fig. 11.3.

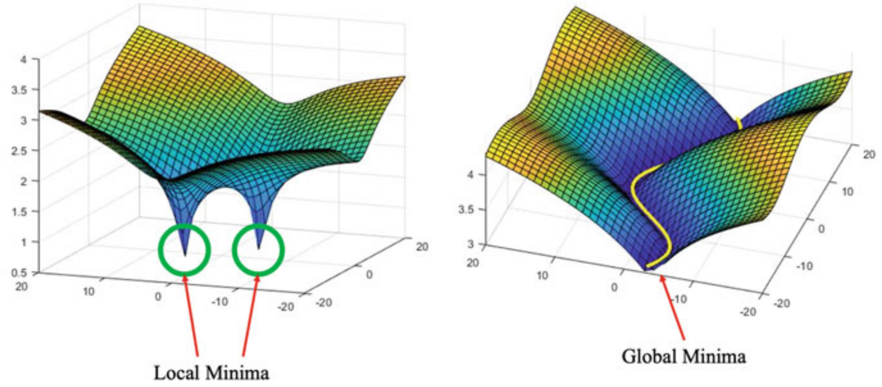

Fig. 11.3 Loss landscapes of (**a**) under-parameterized models and (**b**) over-parameterized models

11.4 Lyapunov-Type Convergence Analysis

Now let us introduce a different type of convergence analysis with a different mathematical flavor. In contrast to the methods discussed above, the analysis of the global loss landscape is not required here. Rather, a local loss geometry along the solution trajectory is the key to this analysis.

In fact, this type of convergence analysis is based on Lyapunov stability analysis [146] for the solution dynamics described by (11.8). Specifically, for a given nonlinear system,

$$\dot{\boldsymbol{\theta}}[t] = \boldsymbol{g}(\boldsymbol{\theta}[t]), \tag{11.18}$$

the Lyapunov stability analysis is concerned with checking whether the solution trajectory $\boldsymbol{\theta}[t]$ converges to zero as $t \to \infty$. To provide a general solution for this, we first define the Lyapunov function $V(z)$, which satisfies the following properties:

Definition 11.1 A function $V : \mathbb{R}^n \mapsto \mathbb{R}$ is positive definite (PD) if

- $V(z) \geq 0$ for all z.
- $V(z) = 0$ if and only if $z = \mathbf{0}$.
- All sublevel sets of V are bounded.

The Lyapunov function V has an analogy to the potential function of classical dynamics, and $-\dot{V}$ can be considered the associated generalized dissipation function. Furthermore, if we set $z := \boldsymbol{\theta}[t]$ to analyze the nonlinear dynamic system in (11.18), then $\dot{V} : z \in \mathbb{R}^n \mapsto \mathbb{R}$ is computed by

$$\dot{V}(z) = \left(\frac{\partial V}{\partial z}\right)^{\top} \dot{z} = \left(\frac{\partial V}{\partial z}\right)^{\top} \boldsymbol{g}(z). \tag{11.19}$$

The following Lyapunov global asymptotic stability theorem is one of the keys to the stability analysis of dynamic systems:

Theorem 11.2 (Lyapunov Global Asymptotic Stability [146]) *Suppose there is a function V such that 1) V is positive definite, and 2) $\dot{V}(\boldsymbol{z}) < 0$ for all $\boldsymbol{z} \neq \boldsymbol{0}$ and $\dot{V}(\boldsymbol{0}) = \boldsymbol{0}$. Then, every trajectory $\boldsymbol{\theta}[t]$ of $\dot{\boldsymbol{\theta}} = \boldsymbol{g}(\boldsymbol{\theta})$ converges to zero as $t \to \infty$. (i.e., the system is globally asymptotically stable).*

Example: 1-D Differential Equation
Consider the following ordinary differential equation:

$$\dot{\theta} = -\theta.$$

We can easily show that the system is globally asymptotically stable since the solution is $\theta[t] = C\exp(-t)$ for some constant C, and $\theta[t] \to 0$ as $t \to \infty$. Now, we want to prove this using Theorem 11.2 without ever solving the differential equation. First, choose a Lyapunov function

$$V(z) = \frac{z^2}{2},$$

where $z = \theta[t]$. We can easily show that $V(z)$ is positive definite. Furthermore, we have

$$\dot{V} = z\dot{z} = -(\theta[t])^2 < 0, \quad \forall\theta[t] \neq 0.$$

Therefore, using Theorem 11.2 we can show that $\theta[t]$ converges to zero as $t \to \infty$.

One of the beauties of Lyapunov stability analysis is that we do not need an explicit knowledge of the loss landscape to prove convergence. Instead, we just need to know the local dynamics along the solution path. To understand this claim, here we apply Lyapunov analysis to the convergence analysis of our gradient descent dynamics:

$$\dot{\boldsymbol{\theta}}[t] = -\frac{\partial \ell}{\partial \boldsymbol{\theta}}(\boldsymbol{\theta}[t]).$$

For the MSE loss, this leads to

$$\dot{\boldsymbol{\theta}}[t] = -\frac{\partial \boldsymbol{f}_{\boldsymbol{\theta}[t]}(\boldsymbol{x})}{\partial \boldsymbol{\theta}}\left(\boldsymbol{y} - \boldsymbol{f}_{\boldsymbol{\theta}[t]}(\boldsymbol{x})\right). \tag{11.20}$$

Now let

$$\boldsymbol{e}[t] := \boldsymbol{f}_{\boldsymbol{\theta}[t]}(\boldsymbol{x}) - \boldsymbol{y}\ ,$$

and consider the following positive definite Lyapunov function

$$V(\boldsymbol{z}) = \frac{1}{2}\boldsymbol{z}^\top \boldsymbol{z},$$

where $\boldsymbol{z} = \boldsymbol{e}[t]$. Then, we have

$$\dot{V}(\boldsymbol{z}) = \left(\frac{\partial V}{\partial \boldsymbol{z}}\right)^\top \dot{\boldsymbol{z}} = \boldsymbol{z}^\top \dot{\boldsymbol{z}}. \tag{11.21}$$

Using the chain rule, we have

$$\dot{\boldsymbol{z}} = \dot{\boldsymbol{e}}[t] = \left(\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{\theta}}\right)^\top \dot{\boldsymbol{\theta}}[t] = -\boldsymbol{K}_t \boldsymbol{e}[t],$$

where

$$\boldsymbol{K}_t = \boldsymbol{K}_{\boldsymbol{\theta}[t]} := \left(\frac{\partial \boldsymbol{f}_{\boldsymbol{\theta}}}{\partial \boldsymbol{\theta}}\right)^\top \frac{\partial \boldsymbol{f}_{\boldsymbol{\theta}}}{\partial \boldsymbol{\theta}}\Bigg|_{\boldsymbol{\theta}=\boldsymbol{\theta}[t]} \tag{11.22}$$

is often called the *neural tangent kernel (NTK)* [130–132]. By plugging this into (11.21), we have

$$\dot{V} = -\eta \boldsymbol{e}[t]^\top \boldsymbol{K}_t \boldsymbol{e}[t]. \tag{11.23}$$

Accordingly, if the NTK is positive definite for all t, then $\dot{V}(\boldsymbol{z}) < 0$. Therefore, $\boldsymbol{e}[t] \to \boldsymbol{0}$ so that $\boldsymbol{f}(\boldsymbol{\theta}[t]) \to \boldsymbol{y}$ as $t \to \infty$. This proves the convergence of gradient descent approach.

11.4.1 The Neural Tangent Kernel (NTK)

In the previous discussion we showed that the Lyapunov analysis only requires a positive-definiteness of the NTK along the solution trajectory. While this is a great advantage over PL-type analysis, which requires knowledge of the global loss landscape, the NTK is a function of time, so it is important to obtain the conditions for the positive-definiteness of NTK along the solution trajectory.

To understand this, here we are interested in deriving the explicit form of the NTK to understand the convergence behavior of the gradient descent methods.

Using the backpropagation in Chap. 6, we can obtain the weight update as follows:

$$\begin{aligned}\frac{\partial \boldsymbol{f}_{\boldsymbol{\theta}}}{\partial \mathrm{VEC}(\boldsymbol{W}^{(l)})} &= \frac{\partial \boldsymbol{g}_n^{(l)}}{\partial \mathrm{VEC}(\boldsymbol{W}^{(l)})} \frac{\partial \boldsymbol{o}_n^{(l)}}{\partial \boldsymbol{g}_n^{(l)}} \frac{\partial \boldsymbol{g}_n^{(l+1)}}{\partial \boldsymbol{o}_n^{(l)}} \cdots \frac{\partial \boldsymbol{o}_n^{(L)}}{\partial \boldsymbol{g}_n^{(L)}} \\ &= (\boldsymbol{o}^{(l)} \otimes \boldsymbol{I}_{d^{(l)}}) \boldsymbol{\Lambda}_n^{(l)} \boldsymbol{W}^{(l+1)\top} \boldsymbol{\Lambda}_n^{(l+1)} \boldsymbol{W}^{(l+2)\top} \cdots \boldsymbol{W}^{(L)\top} \boldsymbol{\Lambda}_n^{(L)}.\end{aligned}$$

Similarly, we have

$$\begin{aligned}\frac{\partial \boldsymbol{f}_{\boldsymbol{\theta}}}{\partial \boldsymbol{b}^{(l)}} &= \frac{\partial \boldsymbol{g}_n^{(l)}}{\partial \boldsymbol{b}^{(l)}} \frac{\partial \boldsymbol{o}_n^{(l)}}{\partial \boldsymbol{g}_n^{(l)}} \frac{\partial \boldsymbol{g}_n^{(l+1)}}{\partial \boldsymbol{o}_n^{(l)}} \cdots \frac{\partial \boldsymbol{o}_n^{(L)}}{\partial \boldsymbol{g}_n^{(L)}} \\ &= \boldsymbol{\Lambda}_n^{(l)} \boldsymbol{W}^{(l+1)\top} \boldsymbol{\Lambda}_n^{(l+1)} \boldsymbol{W}^{(l+2)\top} \cdots \boldsymbol{W}^{(L)\top} \boldsymbol{\Lambda}_n^{(L)}.\end{aligned}$$

Therefore, the NTK can be computed by

$$\begin{aligned}\boldsymbol{K}_t^{(L)} &:= \left(\frac{\partial \boldsymbol{f}_{\boldsymbol{\theta}}}{\partial \boldsymbol{\theta}}\right)^{\top} \frac{\partial \boldsymbol{f}_{\boldsymbol{\theta}}}{\partial \boldsymbol{\theta}}\bigg|_{\boldsymbol{\theta}=\boldsymbol{\theta}[t]} \\ &= \sum_{l=1}^{L} \left(\frac{\partial \boldsymbol{f}_{\boldsymbol{\theta}}}{\partial \mathrm{VEC}(\boldsymbol{W}^{(l)})}\right)^{\top} \frac{\partial \boldsymbol{f}_{\boldsymbol{\theta}}}{\partial \mathrm{VEC}(\boldsymbol{W}^{(l)})} + \left(\frac{\partial \boldsymbol{f}_{\boldsymbol{\theta}}}{\partial \boldsymbol{b}^{(l)}}\right)^{\top} \frac{\partial \boldsymbol{f}_{\boldsymbol{\theta}}}{\partial \boldsymbol{b}^{(l)}} \\ &= \sum_{l=1}^{L} (\|\boldsymbol{o}^{(l)}[t]\|^2 + 1) \boldsymbol{M}^{(l)}[t],\end{aligned}$$

where

$$\boldsymbol{M}^{(l)}[t] = \boldsymbol{\Lambda}^{(L)} \boldsymbol{W}^{(L)}[t] \cdots \boldsymbol{W}^{(l+1)}[t] \boldsymbol{\Lambda}^{(l)} \boldsymbol{\Lambda}^{(l)} \boldsymbol{W}^{(l+1)\top}[t] \cdots \boldsymbol{W}^{(L)\top}[t] \boldsymbol{\Lambda}^{(L)}. \tag{11.24}$$

Therefore, the positive definiteness of the NTK comes from the properties of $\boldsymbol{M}^{(l)}[t]$. In particular, if $\boldsymbol{M}^{(l)}[t]$ is positive definite for any l, the resulting NTK is positive definite. Moreover, the positive-definiteness of $\boldsymbol{M}^{(l)}[t]$ can be readily shown if the following sensitivity matrix is full row ranked:

$$\boldsymbol{S}^{(l)} := \boldsymbol{\Lambda}^{(L)} \boldsymbol{W}^{(L)}[t] \cdots \boldsymbol{W}^{(l+1)}[t] \boldsymbol{\Lambda}^{(l)}.$$

11.4.2 NTK at Infinite Width Limit

Although we derived the explicit form of the NTK using backpropagation, still the component matrix in (11.24) is difficult to analyze due to the stochastic nature of the weights and ReLU activation patterns.

To address this problem, the authors in [130] calculated the NTK at the infinite width limit and showed that it satisfies the positive definiteness. Specifically, they considered the following normalized form of the neural network update:

$$\boldsymbol{o}_n^{(0)} = \boldsymbol{x}, \tag{11.25}$$

$$\boldsymbol{g}^{(l)} = \frac{1}{\sqrt{d^{(l)}}}\boldsymbol{W}^{(l)}\boldsymbol{o}_n^{(l-1)} + \beta\boldsymbol{b}^{(l-1)}, \tag{11.26}$$

$$\boldsymbol{o}^{(l)} = \boldsymbol{\sigma}(\boldsymbol{g}^{(l)}), \tag{11.27}$$

for $l = 1, \cdots, L$, and $d^{(l)}$ denotes the width of the l-th layer. Furthermore, they considered what is sometimes called LeCun initialization, taking $W_{ij}^{(l)} \sim \mathcal{N}\left(0, \frac{1}{d^{(l)}}\right)$ and $b_j^{(l)} \sim \mathcal{N}(0, 1)$. Then, the following asymptotic form of the NTK can be obtained.

Theorem 11.3 (Jacot et al. [130]) *For a network of depth L at initialization, with a Lipschitz nonlinearity σ, and in the limit as the layers width $d^{(1)} \cdots , d^{(L-1)} \to \infty$, the neural tangent kernel $\boldsymbol{K}^{(L)}$ converges in probability to a deterministic limiting kernel:*

$$\boldsymbol{K}^{(L)} \to \kappa_\infty^{(L)} \otimes \boldsymbol{I}_{d_L}. \tag{11.28}$$

Here, the scalar kernel $\kappa_\infty^{(L)} : \mathbb{R}^{d^{(0)} \times d^{(0)}} \mapsto \mathbb{R}$ is defined recursively by

$$\kappa_\infty^{(1)}(\boldsymbol{x}, \boldsymbol{x}') = \frac{1}{d^{(0)}}\boldsymbol{x}^\top \boldsymbol{x}' + \beta^2, \tag{11.29}$$

$$\kappa_\infty^{(l+1)}(\boldsymbol{x}, \boldsymbol{x}') = \kappa_\infty^{(l)}(\boldsymbol{x}, \boldsymbol{x}')\dot{\nu}^{(l+1)}(\boldsymbol{x}, \boldsymbol{x}') + \nu^{(l+1)}(\boldsymbol{x}, \boldsymbol{x}'), \tag{11.30}$$

where

$$\nu^{(l+1)}(\boldsymbol{x}, \boldsymbol{x}') = E_g\left[\sigma(g(\boldsymbol{x}))\sigma(g(\boldsymbol{x}'))\right] + \beta^2, \tag{11.31}$$

$$\dot{\nu}^{(l+1)}(\boldsymbol{x}, \boldsymbol{x}') = E_g\left[\dot{\sigma}(g(\boldsymbol{x}))\dot{\sigma}(g(\boldsymbol{x}'))\right], \tag{11.32}$$

where the expectation is with respect to a centered Gaussian process g of covariance $\nu^{(l)}$, and where $\dot{\sigma}$ denotes the derivative of σ.

Note that the symptotic form of the NTK is positive definite since $\kappa_\infty^{(L)} > 0$. Therefore, the gradient descent using the infinite width NTK converges to the global minima. Again, we can clearly see the benefit of the over-parameterization in terms of large network width.

11.4.3 NTK for General Loss Function

Now, we are interested in extending the example above to the general loss function with multiple training data sets. For a given training data set $\{\boldsymbol{x}_n\}_{n=1}^N$, the gradient dynamics in (11.7) can be extended to

$$\dot{\boldsymbol{\theta}} = -\sum_{n=1}^N \frac{\partial \ell(\boldsymbol{f}_{\boldsymbol{\theta}}(\boldsymbol{x}_n))}{\partial \boldsymbol{\theta}} = -\sum_{n=1}^N \frac{\partial \boldsymbol{f}_{\boldsymbol{\theta}}(\boldsymbol{x}_n)}{\partial \boldsymbol{\theta}} \frac{\partial \ell(\boldsymbol{x}_n)}{\partial \boldsymbol{f}_{\boldsymbol{\theta}}(\boldsymbol{x}_n)},$$

where $\ell(\boldsymbol{x}_n) := \ell(\boldsymbol{f}(\boldsymbol{x}_n))$ with a slight abuse of notation. This leads to

$$\begin{aligned}
\dot{\boldsymbol{f}}_{\boldsymbol{\theta}}(\boldsymbol{x}_m) &= \left(\frac{\partial \boldsymbol{f}_{\boldsymbol{\theta}}(\boldsymbol{x}_m)}{\partial \boldsymbol{\theta}}\right)^\top \dot{\boldsymbol{\theta}} \\
&= -\sum_{n=1}^N \left(\frac{\partial \boldsymbol{f}_{\boldsymbol{\theta}}(\boldsymbol{x}_m)}{\partial \boldsymbol{\theta}}\right)^\top \frac{\partial \boldsymbol{f}_{\boldsymbol{\theta}}(\boldsymbol{x}_n)}{\partial \boldsymbol{\theta}} \frac{\partial \ell(\boldsymbol{x}_n)}{\partial \boldsymbol{f}_{\boldsymbol{\theta}}(\boldsymbol{x}_n)} \\
&= -\sum_{n=1}^N \boldsymbol{K}_t(\boldsymbol{x}_m, \boldsymbol{x}_n) \frac{\partial \ell(\boldsymbol{x}_n)}{\partial \boldsymbol{f}_{\boldsymbol{\theta}}(\boldsymbol{x}_n)},
\end{aligned}$$

where $\boldsymbol{K}_t(\boldsymbol{x}_m, \boldsymbol{x}_n)$ denotes the (m, n)-th block NTK defined by

$$\boldsymbol{K}_t(\boldsymbol{x}_m, \boldsymbol{x}_n) := \left(\frac{\partial \boldsymbol{f}_{\boldsymbol{\theta}}(\boldsymbol{x}_m)}{\partial \boldsymbol{\theta}}\right)^\top \left.\frac{\partial \boldsymbol{f}_{\boldsymbol{\theta}}(\boldsymbol{x}_n)}{\partial \boldsymbol{\theta}}\right|_{\boldsymbol{\theta}=\boldsymbol{\theta}[t]}.$$

Now, consider the following Lyapunov function candidate:

$$V(\boldsymbol{z}) = \sum_{m=1}^N \ell(\boldsymbol{f}_{\boldsymbol{\theta}}(\boldsymbol{x}_m)) = \sum_{m=1}^N \ell(\boldsymbol{z}_m + \boldsymbol{f}_m^*),$$

where

$$\boldsymbol{z} = \begin{bmatrix} \boldsymbol{z}_1 \\ \boldsymbol{z}_2 \\ \vdots \\ \boldsymbol{z}_N \end{bmatrix} = \begin{bmatrix} \boldsymbol{f}_{\boldsymbol{\theta}}(\boldsymbol{x}_1) - \boldsymbol{f}^*(\boldsymbol{x}_1) \\ \boldsymbol{f}_{\boldsymbol{\theta}}(\boldsymbol{x}_2) - \boldsymbol{f}^*(\boldsymbol{x}_2) \\ \vdots \\ \boldsymbol{f}_{\boldsymbol{\theta}}(\boldsymbol{x}_N) - \boldsymbol{f}^*(\boldsymbol{x}_N) \end{bmatrix},$$

and $\boldsymbol{f}^*(\boldsymbol{x}_m)$ refers to $\boldsymbol{f}_{\boldsymbol{\theta}^*}(\boldsymbol{x}_m)$ with $\boldsymbol{\theta}^*$ being the global minimizer. We further assume that the loss function satisfies the property that

$$\forall n, \quad \ell(\boldsymbol{f}_{\boldsymbol{\theta}}(\boldsymbol{x}_n)) > 0, \quad \text{if } \boldsymbol{f}_{\boldsymbol{\theta}}(\boldsymbol{x}_n) \neq \boldsymbol{f}_n^* \ , \quad \ell(\boldsymbol{f}_n *) = 0,$$

so that $V(\boldsymbol{z})$ is a positive definite function. Under this assumption, we have

$$\begin{aligned}
\dot{V}(\boldsymbol{z}) &= \sum_{m=1}^{N} \left(\frac{\partial \ell(\boldsymbol{f}_{\boldsymbol{\theta}}(\boldsymbol{x}_m))}{\partial \boldsymbol{z}_m} \right)^{\top} \dot{\boldsymbol{z}}_m = \sum_{m=1}^{N} \left(\frac{\partial \ell(\boldsymbol{x}_m)}{\partial \boldsymbol{f}_{\boldsymbol{\theta}}(\boldsymbol{x}_m)} \right)^{\top} \dot{\boldsymbol{f}}_{\boldsymbol{\theta}}(\boldsymbol{x}_m) \Bigg|_{\boldsymbol{\theta}=\boldsymbol{\theta}[t]} \\
&= -\sum_{m=1}^{N} \sum_{n=1}^{N} \left(\frac{\partial \ell(\boldsymbol{f}_{\boldsymbol{\theta}}(\boldsymbol{x}_m))}{\partial \boldsymbol{f}_{\boldsymbol{\theta}}(\boldsymbol{x}_m)} \right)^{\top} \boldsymbol{K}_t(\boldsymbol{x}_m, \boldsymbol{x}_n) \frac{\partial \ell(\boldsymbol{f}_{\boldsymbol{\theta}}(\boldsymbol{x}_n))}{\partial \boldsymbol{f}_{\boldsymbol{\theta}}(\boldsymbol{x}_n)} \Bigg|_{\boldsymbol{\theta}=\boldsymbol{\theta}[t]} \\
&= -\boldsymbol{e}[t]^{\top} \mathcal{K}[t] \boldsymbol{e}[t],
\end{aligned}$$

where

$$\boldsymbol{e}[t] = \begin{bmatrix} \frac{\partial \ell(\boldsymbol{f}_{\boldsymbol{\theta}}(\boldsymbol{x}_1))}{\partial \boldsymbol{f}_{\boldsymbol{\theta}}(\boldsymbol{x}_1)} \\ \vdots \\ \frac{\partial \ell(\boldsymbol{f}_{\boldsymbol{\theta}}(\boldsymbol{x}_N))}{\partial \boldsymbol{f}_{\boldsymbol{\theta}}(\boldsymbol{x}_N)} \end{bmatrix}_{\boldsymbol{\theta}=\boldsymbol{\theta}[t]}, \quad \mathcal{K}[t] = \begin{bmatrix} \boldsymbol{K}_t(\boldsymbol{x}_1, \boldsymbol{x}_1) & \cdots & \boldsymbol{K}_t(\boldsymbol{x}_1, \boldsymbol{x}_N) \\ \vdots & \ddots & \vdots \\ \boldsymbol{K}_t(\boldsymbol{x}_N, \boldsymbol{x}_1) & \cdots & \boldsymbol{K}_t(\boldsymbol{x}_N, \boldsymbol{x}_N). \end{bmatrix}$$

Therefore, if the NTK $\mathcal{K}[t]$ is positive definite for all t, then Lyapunov stability theory guarantees that the gradient dynamics converge to the global minima.

11.5 Exercises

1. Show that a smooth $\ell(\boldsymbol{\theta})$ is invex if and only if every stationary point of $\ell(\boldsymbol{\theta})$ is a global minimum.
2. Show that a convex function is invex.
3. Let $a > 0$. Show that $V(x, y) = x^2 + 2y^2$ is a Lyapunov function for the system

$$\dot{x} = ay^2 - x \ , \ \dot{y} = -y - ax^2.$$

4. Show that $V(x, y) = \ln(1 + x^2) + y^2$ is a Lyapunov function for the system

$$\dot{x} = x(y - 1) \ , \ \dot{y} = -\frac{x^2}{1 + x^2}.$$

5. Consider a two-layer fully connected network $f_\Theta : \mathbb{R}^2 \to \mathbb{R}^2$ with ReLU nonlinearity, as shown in Fig. 10.10.

(a) Suppose the weight matrices and biases are given by

$$\boldsymbol{W}^{(0)} = \begin{bmatrix} 2 & -1 \\ 1 & 1 \end{bmatrix}, \quad \boldsymbol{b}^{(0)} = \begin{bmatrix} 1 \\ -1 \end{bmatrix}$$

$$\boldsymbol{W}^{(1)} = \begin{bmatrix} 1 & 2 \\ -1 & 1 \end{bmatrix}, \quad \boldsymbol{b}^{(1)} = \begin{bmatrix} -9 \\ -2 \end{bmatrix}.$$

Given the corresponding input space partition in Fig. 10.11, compute the neural tangent kernel for each partition. Are they positive definite?

(b) In problem (a), suppose that the second layer weight and bias are changed to

$$\boldsymbol{W}^{(1)} = \begin{bmatrix} 1 & 2 \\ 0 & 1 \end{bmatrix}, \quad \boldsymbol{b}^{(1)} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}.$$

Given the corresponding input space partition, compute the neural tangent kernel for each partition. Are they positive definite?