Implicit Bias: Literature Overview

Felix Konrad

January 19, 2024

Abstract

The term "Implicit Bias" or "Implicit Regularization" of an optimization method refers to the phenomenon that among the many parameter configurations with minimal training error, the optimization procedure seems to favour a set of parameters satisfying additional properties, possibly improving performance of the trained model on unseen data.
In this document we aim to give an introduction to various optimization methods for neural networks and give an overview of implicit bias results thereof.
Note: This is a work in progress and thus not a complete overview of the literature.

Contents

1 Introduction

2 Optimization

2.1 Theory

2.1.1 The Karush-Kuhn-Tucker Theorem in $\mathbb{R}^d$

Theorem 2.1 (Karush-Kuhn-Tucker I, [Nes+18] Thm 3.1.26)**.** *Let*

$$f, g_i : \mathbb{R}^d \to \mathbb{R}, \qquad i = 1, \dots, n$$

be convex differentiable functions. Furthermore, suppose Slater's condition

$$\exists x \in \mathbb{R}^d : g_i(x) < 0, \qquad i = 1, \dots, n.$$

A point x_0 is optimal for the convex optimization problem

$$\begin{aligned} &\min \ f(x) \\ &s.t. \quad g_i(x) \leq 0, \qquad i = 1, \dots, n \end{aligned}$$

if and only if there exist non-negative $\lambda_i \in \mathbb{R}_{\geq 0}$ such that

$$\begin{aligned} \nabla f(x_0) - \sum_{i=1}^{n} \lambda_i \nabla g_i(x_0) &= 0 \\ \lambda_i \cdot g_i(x_0) &= 0 \qquad i = 1, \dots, n \end{aligned}$$

Theorem 2.2 (Karush-Kuhn-Tucker II, [Nes+18] Thm 3.1.27)**.** *Let $f : \mathbb{R}^d \to \mathbb{R}$ be a convex differentiable function with bounded level sets and let $A \in \mathbb{R}^{m \times d}$ be a full-row-rank matrix. Then $x_0 \in \mathbb{R}^d$ is optimal for the convex optimization problem*

$$\begin{aligned} &\min \ f(x) \\ &s.t. \ Ax = b \end{aligned}$$

if and only if $Ax = b$ and there exists $\mu \in \mathbb{R}^m$ with

$$\nabla f(x_0) - A^T \mu = 0$$

2.2 Algorithms

2.2.1 Gradient Descent

Theorem 2.3 (Convergence of Gradient Descent)**.** *Let $f : \mathbb{R}^d \to \mathbb{R}$ be convex and continuously differentiable with Lipschitz-continuous gradient, meaning for $x, y \in \mathbb{R}^d$ we have $\|\nabla f(x) - \nabla f(y)\| \leq L\|x - y\|$. Assume that there exists a minimum $f(x^*)$ of f. Choose a step size $\eta \in (0, \frac{1}{L}]$, then the iterates $(x_k)_{k\in\mathbb{N}}$ obtained by*

$$x_{k+1} := x_k - \eta \nabla f(x_k)$$

satisfy

$$f(x_k) - f(x^*) \leq \frac{\|x_0 - x^*\|^2}{2\eta k}$$

In particular, $(f(x_k))_{k\in\mathbb{N}}$ converges to the minimum $f(x^*)$.

2.2.2 Stochastic Gradient Descent

Since it is common to have a large amount of data points in many deep learning tasks, it can be too memory consuming to compute the loss gradient with respect to *all* data points simultaneously. The idea, or at least one of the ideas, of stochastic gradient descent (SGD) is to randomly select a data point at each iteration step for which the gradient computation is performed. More precisely, given a dataset $(x_1, y_1), \ldots, (x_N, y_N)$, a model f_θ and a corresponding loss

$$\mathcal{L}(\theta) = \frac{1}{N} \sum_{i=1}^{N} \ell(f_\theta(x_i), y_i),$$

in each step we sample $i_n \sim \mathrm{Uni}(\{1, \ldots, N\})$ and update

$$\theta_{n+1} = \theta_n - \eta_n \cdot \nabla_\theta \ell(f_{\theta_n}(x_{i_n}), y_{i_n}).$$

For this procedure, we are able to obtain similar convergence guarantees for SGD as we did for gradient descent (Theorem 2.3):

Theorem 2.4 (Convergence of SGD, [GG23] Thm 5.5)**.** *Let f be given by*

$$f(x) = \frac{1}{N} \sum_{i=1}^{N} f_i(x),$$

where $f_i : \mathbb{R}^d \to \mathbb{R}$ is convex and ∇f_i is L-Lipschitz. Assume a minimizer x^* of f exists. Let $x_0 \in \mathbb{R}^d$ and let $(x_n)_{n\in\mathbb{N}}$ be the iterates obtained by SGD with step sizes $(\eta_n)_{n\in\mathbb{N}}$. Then If $\eta_n = \eta < \frac{1}{2L}$, then for every $n \geq 1$ we have

$$\mathbb{E}\big[f(\bar{x}_n) - f(x^*)\big] \leq \frac{\|x_0 - x^*\|^2}{2\eta(1 - 2\eta L)n} + \frac{\eta}{1 - 2\eta L} \cdot \mathrm{Var}[\nabla f_i(x^*)],$$

where $\bar{x}_n = \frac{1}{n} \sum_{k=1}^{n} x_k$.

Remark. Theorem 5.5 of [GG23] also includes a convergence Theorem for variable step sizes chosen as $\eta_n := \eta/\sqrt{n+1}$ with $\eta < (2L)^{-1}$.

This Theorem only tells us that $f(\bar{x}_n) \to f(x^*)$. It is not immediately clear whether $f(x_n) \to f(x^*)$ as well and whether the iterates $(x_n)_{n\in\mathbb{N}}$ converge to some minimizer (not necessarily x^*, as there may be multiple minimizers).
To guarantee convergence of the iterates $(x_n)_{n\in\mathbb{N}}$ we need stronger assumptions:

Theorem 2.5.

2.2.3 Mirror Descent

To motivate the Mirror Descent Algorithm, note that the iteration performed in Gradient Descent can be interpreted in the following sense: Let $\eta > 0$ be the step size and let f be the differentiable function we are trying to minimize. Given a current point x_k we find x_{k+1} by minimizing:

$$x_{k+1} := \arg\min_x \left\{ f(x_k) + \langle \nabla f(x_k), x - x_k \rangle + \frac{1}{2\eta}\|x - x_k\|^2 \right\}.$$

The idea is that for x near x_k, the first-order approximation of f given by $f(x_k) + \langle \nabla f(x_k), x - x_k \rangle$ is quite accurate; Thus, instead of minimizing f (which was our original goal and is not doable in closed form) we minimize the local approximation. The addition of an error term $\frac{1}{2\eta}\|x - x_k\|^2$ prevents us from deviating from x_k too much. (Without this quadratic error term the minimization problem would also be ill-posed)
We can solve this minimization problem explicitly: Taking gradients yields

$$0 \overset{!}{=} \nabla_x \left\{ f(x_k) + \langle \nabla f(x_k), x - x_k \rangle + \frac{1}{2\eta}\|x - x_k\|^2 \right\} = \nabla f(x_k) + \frac{1}{\eta}(x - x_k)$$

which rearranges to

$$x = x_k - \eta \nabla f(x_k),$$

which is exactly the iteration performed in Gradient Descent.
Mirror Descent now simply replaces the norm-penalty $\frac{1}{2\eta}\|x - x_k\|^2$ by a *Bregman Divergence*:

Definition 2.1 (Bregman Divergence)**.** Let $\Omega \subseteq \mathbb{R}^d$ be a closed convex set. Let $\psi : \Omega \to \mathbb{R}$ be a strictly convex differentiable function. The Bregman Divergence is defined as

$$D_\psi(x, y) := \psi(x) - \psi(y) - \langle \nabla\psi(y), x - y \rangle, \qquad x, y \in \Omega.$$

A few properties immediately follow from Definition 2.1:

- Convexity of ψ implies $D_\psi \geq 0$
- *Strict* convexity of ψ implies $D_\psi(x, y) = 0$ if and only if $x = y$

Now we follow the same approach as before: Given a current point x_k we minimize

$$x_{k+1} := \arg\min_x \left\{ f(x_k) + \langle \nabla f(x_k), x - x_k \rangle + \frac{1}{\eta} D_\psi(x, x_k) \right\}.$$

Taking gradients yields

$$\begin{aligned} 0 &\stackrel{!}{=} \nabla_x \left\{ f(x_k) + \langle \nabla f(x_k), x - x_k \rangle + \frac{1}{\eta} D_\psi(x, x_k) \right\} \\ &= \nabla f(x_k) + \frac{1}{\eta} \Big(\nabla \psi(x) - \nabla \psi(x_k) \Big) \end{aligned}$$

Since ψ is *strictly* convex and differentiable, its gradient is invertible and thus

$$x_{k+1} = (\nabla \psi)^{-1} \big[\nabla \psi(x_k) - \eta \nabla f(x_k) \big]$$

Example 2.1 (Examples of Bregman Divergences)**.** Here we list a few important examples of Bregman Divergences:

1. $\Omega = \mathbb{R}^d$ and $\psi(x) = \frac{1}{2}\|x\|^2$ gives $D_\psi(x, y) = \frac{1}{2}\|x - y\|^2$ and Mirror Descent corresponds to standard Gradient Descent
2. $\Omega = \{x \in \mathbb{R}^d : x_i \geq 0 \text{ and } \sum_{i=1}^d x_i = 1\}$ and $\psi(x) = \sum_{i=1}^d x_i \cdot \log x_i$ gives the *Kullback-Leibler divergence* D_ψ. Elements in Ω are interpreted as probability measures on a set with d elements.

3 Wide Neural Networks

We are interested in the asymptotic behaviour of neural networks during training as their width goes to ∞. To achieve a reasonable limit, we need to suitably normalize the weights of the network or the output of the network. To get an idea of what might be a good normalization, consider a neural network with just one hidden layer with n nodes. Notice that if $h \in \mathbb{R}^n$ is the output of the hidden layer and $w^L =\in \mathbb{R}^{1\times d_{L-1}}$ is the weight matrix of the output layer, then the final output of the neural network f_θ will be given by

$$f_\theta(x) = w^T h = \sum_{i=1}^{n} w_i h_i.$$

Since in practice the weights w are randomly initialized, this quantity is just a sum of random variables. Now there are two well-known methods of probability theory to describe the asymptotic behaviour of sums of random variables:

1. Central Limit Theorem: Normalize the sum by $n^{-1/2}$, get a normal distribution in the limit, given that the random variables are more-or-less independent and identically distributed with finite second moments.

2. Law of Large Numbers: Normalize the sum by n^{-1}, get a deterministic limit, given that the random variables are more-or-less independent and identically distributed, with finite first moments.

We then have two choices on how to normalize: Normalize the output, i.e. compute $n^{-1/2} f_\theta(x)$ or $n^{-1} f_\theta(x)$, or normalize the parameter distribution, i.e. instead of sampling $w_i \sim \mathcal{N}(0,1)$ sample $w_i \sim \mathcal{N}(0, n^{-1/2})$ then compute $f_\theta(x)$ as usual. While these two methods will induce the same distribution on f_θ upon initialization, the dynamics under gradient descent will be different.

Combining these possibilities gives us the following initialization procedures:

1. **Standard Parametrization**: Sample the weights $W^l \sim \frac{1}{d_l}\mathcal{W}$, where $\mathcal{W}$ is a pre-determined distribution, then define the neural network as

$$\begin{aligned} h^0 &= x \\ z^{l+1} &= W^l h^l + b^l \\ h^{l+1} &= \sigma(z^{l+1}) \end{aligned}$$

   for $W^l \in \mathbb{R}^{d_{l+1}\times d_l}, b^l \in \mathbb{R}^{d_{l+1}}$. We always assume the activation function σ to be Lipschitz.

2. **NTK Parametrization**, [JGH18]: Sample the weights $w \sim \mathcal{W}$ indepen-

dently, then define the neural network as

$$\begin{aligned} h^0 &= x \\ z^{l+1} &= \frac{1}{\sqrt{n_l}} W^l h^l + \beta b^l \\ h^{l+1} &= \sigma(z^{l+1}) \end{aligned}$$

Note that we are only normalizing the weights W, but not the bias terms b. The factor $\beta \in \mathbb{R}$ is there to tune the influence of the bias terms.

3. **Mean-Field Parametrization:** ...

3.1 Neural Tangent Kernel

(Throughout this section, we will assume the NTK parametrization for our neural network f_θ.)
Neural networks f_θ are, in general, *non-linear* functions of their parameters θ. This makes training difficult, as the loss landscape may be non-convex. As a consequence, the path of gradient descent may be very difficult to determine. Nevertheless, in the infinite-width limit the dynamics of gradient descent can be very well-approximated by a *kernel gradient descent*.
Let ℓ be a loss function and let

$$\mathcal{L}(\theta) := \frac{1}{N} \sum_{i=1}^{N} \ell(f_\theta(x^{(i)}), y^{(i)}),$$

where $f_\theta : \mathbb{R}^{n_0} \to \mathbb{R}^{n_L}$ is a neural network with L layers with parameters $\theta \in \mathbb{R}^p$. Suppose we initialize θ_0 and update the parameters via gradient flow:

$$\dot{\theta}_t = -\nabla_\theta \mathcal{L}(\theta_t) = \frac{1}{N} \sum_{i=1}^{N} \nabla_f \ell(f_{\theta_t}(x^{(i)}), y^{(i)}) \cdot \nabla_\theta f_{\theta_t}(x^{(i)}),$$

then the corresponding neural network f_{θ_t} evolves as

$$\begin{aligned} \dot{f}_{\theta_t}(x) &= \nabla_\theta f_{\theta_t}(x) \cdot \dot{\theta}_t \\ &= -\frac{1}{N} \sum_{i=1}^{N} \underbrace{\nabla_\theta f_{\theta_t}(x) \nabla_\theta f_{\theta_t}(x^{(i)})^T}_{=:\Theta(\theta_t)(x, x^{(i)})} \nabla_f \ell(f_{\theta_t}(x^{(i)}), y^{(i)}) \end{aligned}$$

The matrix Θ is called the *Neural Tangent Kernel*. The name "kernel" is justified by the fact that since $\Theta \in \mathbb{R}^{n_L \times n_L}$ is positive semi-definite, we can view Θ as a kernel $(x, y) \mapsto \Theta(x, y) := x^T \Theta y$.
As this heuristic derivation shows, if we update our parameters via gradient flow, then the evolution of the neural network f_{θ_t} is described by the neural tangent kernel Θ. However, Θ is still dependent on the parameters θ. As we will see later, in the infinite-width limit the neural tangent kernel will converge (in probability) to a fixed kernel. In this case, the evolution of the "infinite-width-network" is nothing but a *kernel gradient descent*.

Definition 3.1 (Neural Tangent Kernel)**.** Let f_θ be a neural network. We define the neural tangent kernel $\Theta(\theta) : \mathbb{R}^{n_0} \times \mathbb{R}^{n_0} \to \mathbb{R}^{n_L \times n_L}$ of f_θ as

$$\Theta(\theta)(x, x') = \nabla_\theta f_\theta(x) \nabla_\theta f_\theta(x')^T.$$

Understanding the convergence of the neural tangent kernel as the width goes to ∞ will help us understand gradient flow (and gradient descent) for wide neural networks. First, we take a look at what happens to the neural network and the neural tangent kernel during initialization in the infinite-width limit:

Theorem 3.1 (Infinite-Width Neural Networks are Gaussian Processes I, [JGH18], Prop. 1)**.** *As $n_1, \dots, n_{L-1} \to \infty$, the functions $f_\theta^{(i)}$ for $i = 1, \dots, n_L$ converge in distribution to i.i.d. centered Gaussian processes with covariance $\Sigma^{(L)}$, where $\Sigma^{(L)}$ is defined as*

$$\begin{aligned}\Sigma^{(1)}(x, x') &= \frac{1}{n_0} x^T x' + \beta^2 \\ \Sigma^{(l+1)}(x, x') &= \mathbb{E}_{f \sim \mathcal{N}(0, \Sigma^{(l)})}\big[\sigma(f(x))\sigma(f(x'))\big] + \beta^2\end{aligned}$$

where $f \sim \mathcal{N}(0, \Sigma)$ means f is a Gaussian process with covariance function Σ.

We get a similar convergence result for the neural tangent kernel:

Theorem 3.2 (Convergence of the NTK, [JGH18] Thm. 1)**.** *As $n_1, \dots, n_{L-1} \to \infty$, the NTK at initialization Θ converges in probability to a deterministic limiting kernel Θ_∞.*

Both of the preceding Theorems deal with the initialization. What happens during training? Will the output of f_θ remain Gaussian? How will the NTK Θ_∞ change?
It turns out that during training, the NTK will remain constant. From this fact it will follow that f_θ will indeed remain a Gaussian process whose mean and covariance functions will evolve over time.
More precisely, suppose we train our neural network via gradient flow, meaning $\dot{\theta}(t) = -\nabla_\theta \mathcal{L}(\theta(t))$. Let f_t be the corresponding neural network and $\Theta(t)(x, x') = \nabla_\theta f_t(x) \nabla_\theta f_t(x')^T$ be its neural tangent kernel. Then we have the following result:

Theorem 3.3 (NTK Remains Constant During Training, [JGH18] Thm. 2)**.** *Let the activation function σ be Lipschitz and twice differentiable with bounded second derivative. For any $T > 0$ such that*

$$\int_0^T \mathcal{L}(\theta_t) dt$$

stays stochastically bounded as $n_1, \dots, n_{L-1} \to \infty$ we have

$$\Theta(t) \to \Theta_\infty, \qquad \textit{uniformly for } t \in [0, T] \textit{ as } n_1, \dots, n_{L-1} \to \infty$$

Example:
Consider the mean-squared-error $\mathcal{L}(\theta) = \frac{1}{N}\sum_{i=1}^{N} \|f_\theta(x_i) - y_i\|_2^2$. The NTK dynamics in the infinite-width limit then read as

$$\dot{f}_t(x) = -\frac{1}{N}\sum_{i=1}^{N} \Theta_\infty(x, x_i)\big(f_t(x_i) - y_i\big),$$

which, since Θ_∞ is deterministic and constant during training, is a linear differential equation. We can thus explicitly solve this ODE to obtain

$$f_t(\mathcal{X}) = \mathcal{Y} + \exp(-t\Theta_\infty)\big(f_0(\mathcal{X}) - \mathcal{Y}\big).$$

Thus, *if* $\exp(-t\Theta_\infty)$ converges to 0 as $t \to \infty$, gradient flow converges to a function with 0 training loss. The convergence of $\exp(-t\Theta_\infty)$ is highly dependent on whether Θ_∞ is strictly positive definite, meaning whether $\lambda_{\min}(\Theta_\infty) \geq c > 0$ everywhere for some c.
Furthermore, in this example, the output functions f_t remain Gaussian at all times t.

3.2 "Wide Neural Networks of Any Depth Evolve as Linear Models Under Gradient Descent"

If we stick with the NTK parametrization, [Lee+19] have shown that the training of a wide neural network f of fixed depth can be well-approximated by the training of its linearization f^{lin}. Recall that if f is a neural network with parameter initialization $\theta_0 \in \mathbb{R}^p$, its linearization is defined by

$$f^{lin}(x;\theta) := f(x;\theta) + \nabla_\theta f(x;\theta)\big(\theta - \theta_0\big).$$

The outline of this section is as follows:

1. First we state the main result of [Lee+19], establishing the connection between wide networks and their linearization.
2. Then we determine closed form solutions for the training trajectories of the linearized model f^{lin}.
3. Finally, we determine the asymptotic behaviour of the linearized model as the width of the network goes to infinity.

Theorem 3.4 (Wide NNs Evolve as Linear Models, [Lee+19] Thm 2.1)**.**

3.3 The Mean-Field-Regime

4 Implicit Bias

In this section, we will review selected papers on the topic of implicit bias.

4.1 General Results for (Stochastic) Gradient Descent

4.1.1 "Implicit Gradient Regularization"

This section is based on the paper [BD20].

Question: How does the path of gradient descent $\theta_{n+1} = \theta_n - \eta \nabla_\theta \mathcal{L}(\theta)$ differ from the path of gradient flow $\dot{\theta}_t = -\nabla_\theta \mathcal{L}(\theta)$? Is there an implicit bias introduced by this discretization? Does it depend on η?

The main result of this paper answers this question: The iterates of gradient descent are near the path of a gradient flow for a modified loss function $\widetilde{\mathcal{L}} = \mathcal{L} + \eta \mathcal{R}$, where $\mathcal{R}$ is a regularizing term.

Theorem 4.1 (Implicit Gradient Regularization, [BD20] Thm 3.1)**.** *Let $E : \mathbb{R}^p \to \mathbb{R}$ be a sufficiently differentiable function. Let $(\theta_n)_{n\in\mathbb{N}}$ be the sequence obtained by gradient descent, meaning $\theta_{n+1} = \theta_n - \eta \nabla \theta E(\theta_n)$. Consider the gradient flow*

$$\dot{\theta}_t = -\nabla_\theta \widetilde{E}(\theta_t)$$

with

$$\widetilde{E}(\theta) := E(\theta) + \lambda \underbrace{\frac{1}{p} \|\nabla_\theta E(\theta)\|_2^2}_{=: R_{IG(\theta)}}$$

and let $\widetilde{\theta}(t)$ be its solution starting at $\widetilde{\theta}(0) = \theta_{n-1}$. Then the local error $\|\theta_n - \widetilde{\theta}(\eta)\|$ is of order $\mathcal{O}(\eta^3)$.
Note that the error would be $\mathcal{O}(\eta^2)$ if we would consider the gradient flow $\dot{\theta}_t = -\nabla_\theta E(\theta_t)$.

Proof. For simplicity of notation, let $f(\theta) = -\nabla_\theta E(\theta)$. The proof follows the approach of finding a modified function

$$\widetilde{f}(\theta) = f(\theta) + \eta f_1(\theta) + \eta^2 f_2(\theta) + \dots$$

such that the iterates of gradient descent for $\dot{\theta} = f(\theta)$ lie on the path of the solution of the gradient flow $\widetilde{\theta}_t' = \widetilde{f}(\widetilde{\theta})$.
Let $\alpha = \eta/n$ and consider an Euler discretization $(\hat{\theta}_n)$ of $\theta_t' = \widetilde{f}(\theta_t)$ with n steps and step size α, then

$$\begin{aligned}
\hat{\theta}_{t+n} &= \hat{\theta}_t + n\alpha \widetilde{f}(\theta_t) + \alpha \widetilde{f}(\theta_{t+1}) + \dots \\
&= \hat{\theta}_t + \alpha \widetilde{f}(\theta_t) + \alpha \widetilde{f}\big(\theta_t + \alpha \widetilde{f}(\theta_t)\big) + \dots \\
\text{(Taylor exp. of } \widetilde{f}) &= \theta_t + n\alpha \widetilde{f}(\theta_t) + \frac{n(n-1)}{2} \alpha^2 \nabla \widetilde{f}(\theta_t) \widetilde{f}(\theta_t) + \mathcal{O}(n^3 \alpha^3)
\end{aligned}$$

Now as $n \to \infty$ the Euler discretization converges to the solution of the gradient flow $\theta' = \widetilde{f}(\theta)$ and thus

$$\underbrace{\theta(t+\eta)}_{\text{GF for } \widetilde{f}} = \theta_t + \eta \widetilde{f}(\theta_t) + \frac{1}{2}\eta^2 \nabla \widetilde{f}(\theta_t)\widetilde{f}(\theta_t) + \mathcal{O}(\eta^3)$$

$$(\text{Def. of } \widetilde{f}) = \underbrace{\theta_t + \eta f(\theta_t)}_{\text{iterate of GD for } f} + \eta^2 \left(f_1(\theta_t) + \frac{1}{2}\nabla f(\theta_t) f(\theta_t) \right) + \mathcal{O}(\eta^3).$$

Thus if we want the iterates of gradient descent for f to stay on the trajectory of the gradient flow of $\widetilde{f}$, we need that $f_1 = -\frac{1}{2}(\nabla f) \cdot f$.
Since $f = -\nabla E$ we obtain

$$\nabla_\theta \widetilde{E}(\theta) = \nabla_\theta E(\theta) - \eta \frac{1}{2} \nabla_\theta^2 E(\theta) \nabla_\theta E(\theta) = \nabla \left(E(\theta) - \eta \frac{1}{4} \|\nabla E(\theta)\|_2^2 \right).$$

Since the remainder was of order $\mathcal{O}(\eta^3)$, we get the desired result. □

Informally, this theorem tells us that it is the *discretization* that induces an implicit bias (regularization by $\|\nabla \mathcal{L}\|$). This implicit bias is thus not present in the continuous-time setting of gradient flows.

Connection to the Neural Tangent Kernel:
As noted in [BD20] Appendix A.3 the regularization term R_{IG} can be expressed in terms of the neural tangent kernel.

Proposition 4.1. Consider a model $f_\theta : \mathbb{R}^d \to \mathbb{R}^n$ with parameters $\theta \in \mathbb{R}^p$. Let

$$E(\theta) := \sum_{i=1}^{N} \|f_\theta(x_i) - y_i\|_2^2.$$

Then the modified loss $\widetilde{E}$ can be rewritten as

$$\widetilde{E}(\theta) = E(\theta) + \eta \underbrace{\sum_{i,j=1}^{N} \epsilon_i(\theta) K_\theta(x_i, x_j) \epsilon_j(\theta)}_{R_{IG}}$$

where $\epsilon_k(\theta) := f_\theta(x_k) - y_k$ is the k-th residual and K_θ is the neural tangent kernel

$$K_\theta(x_i, x_j) = \nabla f_\theta(x_i) \nabla_\theta f_\theta(x_j)^T.$$

4.1.2 "On the Origin of Implicit Regularization in Stochastic Gradient Descent"

This section is based on the paper [Smi+20].

4.2 Linear Models

This section will deal with the implicit bias for linear models. The big advantage of starting with linear models is that, unlike deep neural networks, their loss landscape will be convex (given that the loss function is convex). This will make a theoretical analysis of gradient descent much easier.
Despite the difference of linear models and deep neural networks, we will be able to use the implicit bias results of linear models to derive implicit bias results for neural networks by using the fact that in the NTK regime, training wide neural networks is similar to training linear models (see the results of Section 3.2).

4.2.1 Implicit Bias of Gradient Descent for Linear Regression

In this section we are going to consider linear models $f_\omega : \mathbb{R}^d \to \mathbb{R}; f(x) := \omega^T x$ where $\omega \in \mathbb{R}^d$ denotes the parameters of the model. The loss function is given by the mean squared error

$$\ell(f_\omega, (x^{(i)}, y^{(i)})_{i=1,\dots,N}) = \sum_{i=1}^{N} \left[f_\omega(x^{(i)}) - y^{(i)}\right]^2.$$

Throughout this, we will assume a fixed dataset $x^{(1)}, \dots, x^{(N)} \in \mathbb{R}^d; y^{(1)}, \dots, y^{(N)} \in \mathbb{R}$ and we will abbreviate $\ell(\omega) := \ell(f_\omega, (x^{(i)}, y^{(i)})_{i=1,\dots,N})$.
A classical result regarding Gradient Descent for such models is that Gradient Descent (with suitable step size) not only converges to a global minimum, but in addition the iterates of Gradient Descent converge to a global minimum *closest to the initial parameters* $\omega^{(0)}$. We make this precise in the following Theorem:

Theorem 4.2 (Implicit Bias of Gradient Descent for Linear Models)**.** *Let* $X^T = (x^{(1)}, \dots, x^{(N)}) \in \mathbb{R}^{d \times N}, y = (y^{(1)}, \dots, y^{(N)}) \in \mathbb{R}^N$ *be any realizable dataset with* $d \geq N$*. Let* $\omega^{(0)} \in \mathbb{R}^d$ *and define*

$$\omega^{(k+1)} := \omega^{(k)} - \eta \nabla \ell(\omega^{(k)}).$$

Then for small enough $\eta > 0$ *the sequence* $(\omega^{(k)})_{k \in \mathbb{N}}$ *converges to a global minimum* ω^* *solving the following optimization problem*

$$\begin{aligned} \min_{\omega} \quad & \|\omega - \omega^{(0)}\|^2 \\ s.t. \quad & X\omega = y \end{aligned} \tag{MIN}$$

Proof. The proof is quite elegant and reveals a strategy that will be used repeatedly when proving "Implicit Bias" results; hence we will include it here:
Note that by the Karush-Kuhn-Tucker Theorem (Theorem 2.2) we can characterize an optimal solution ω^* to the optimization problem (MIN): A point $\omega^* \in \mathbb{R}^d$ is optimal for (MIN) if and only if it satisfies:

$$X\omega^* = y, \qquad \exists \lambda \in \mathbb{R}^N : (\omega^* - \omega^{(0)}) + X^T \lambda = 0.$$

Now we need to prove two things:

1. The iterates $(\omega^{(k)})_{k\in\mathbb{N}}$ converge to a point ω^* with 0 loss, i.e. $X\omega^* = y$: Note that f_ω is linear in ω and thus $\ell(\omega)$ is strictly convex and differentiable with Lipschitz gradient $\nabla_\omega \ell$. Hence, by Theorem 2.3 the convergence of Gradient Descent to a global minimum follows. Since $d \geq N$, there exists an exact interpolation of the data points, meaning $\ell(\omega^*) = 0$.

2. The iterates $(\omega^{(k)})_{k\in\mathbb{N}}$ satisfy the second KKT condition: Note that

$$\omega^{(k)} - \omega^{(0)} = \eta \sum_{l=0}^{k-1} \nabla \ell(\omega_l) = 2\eta \sum_{l=0}^{k-1} X^T (X\omega^{(l)} - y)$$

and hence for $\lambda = -2\eta \sum_{l=0}^{k-1} (X\omega^{(l)} - y)$ we see that the second KKT condition is satisfied. By continuity, the limiting parameters $\omega^* = \lim_{k\to\infty} \omega^{(k)}$ also satisfies this condition.

Combining these two claims gives the stated result. □

4.2.2 "Characterizing Implicit Bias in Terms of Optimization Geometry"

This section is based on the paper [Gun+18].
In Section 2.2.6 we have seen how mirror descent can be seen as a generalization of gradient descent. In the previous section we have characterized the implicit bias of gradient descent for least squares regression. Now [Gun+18] generalize this theorem to mirror descent for *losses with a unique finite root*:

Definition 4.1 (Losses with a unique finite root)**.** We say ℓ is a loss function with a unique finite root if for all y and any sequence $(y_n)_{n\in\mathbb{N}}$ we have $\ell(y_n, y) \to \inf_{\hat{y}} \ell(\hat{y}, y)$ if and only if $y_n \to y$.

The mean squared error loss function is a special case of a loss function with a unique finite root, as any L_p-loss ($p \geq 1$) or the Huber loss. However, common loss functions like the logistic loss of the exponential loss are not within this class. Such loss functions will be dealt with later in this chapter.
We now come to the main theorem regarding mirror descent for losses with a unique finite root, which is to be seen as a generalization of Theorem 4.2:

Theorem 4.3 (Implicit Bias of Mirror Descent for Losses with a unique finite root, [Gun+18] Thm 1)**.** *Let ℓ be a loss function with unique finite root, $(x_n, y_n)_{n=1,\dots,N}$ be a realizable dataset. Let ψ be a strictly convex potential. Consider the linear model $f_\omega(x) = \langle \omega, x\rangle$. Assume that the iterates $(\omega_n)_{n\in\mathbb{N}}$ obtained by mirror descent starting at ω_0 with step sizes $(\eta_n)_{n\in\mathbb{N}}$ satisfy*

$$\lim_{n\to\infty} \omega_n = \omega_\infty \text{ exsits and satisfies } \ell(f_{\omega_\infty}(x_n), y_n) = 0 \forall n.$$

Then ω_∞ is the solution to the following optimization problem:

$$\begin{aligned} \min_{\omega\in\mathbb{R}^d} \ & D_\psi(\omega, \omega_0) \\ s.t. \ & f_\omega(x_n) = y_n, \ \ n = 1, \dots, N \end{aligned}$$

In particular, if we let ω_0 be a minimizer of ψ s.t. $\nabla\psi(\omega_0) = 0$, then ω_∞ is again a minimizer of ψ with $\langle \omega_\infty, x_n\rangle = y_n$.

Proof. The proof is completely analogous to the proof of Theorem 4.2. Note that since ψ is convex, the function $\omega \mapsto D_\psi(\omega, \omega_0)$ is convex as well and hence the minimization problem given above is a convex minimization problem. Applying the Karush-Kuhn-Tucker Theorem (Theorem 2.2) tells us that ω_∞ is a minimizer if and only if

$$\langle \omega_\infty, x_n\rangle = y_n \text{ for all } n \text{ and } \exists \lambda \in \mathbb{R}^N : \nabla\psi(\omega_\infty) - \nabla\psi(\omega_0) - \sum_{n=1}^{N} \lambda_n x_n = 0.$$

By assumption, the iterates of mirror descent converge to a point ω_∞ with 0 loss, hence the first condition is satisfied.
To verify the second condition, we will show that every iterate ω_n of mirror descent satisfies this condition, hence by convergence of $\omega_n \to \omega_\infty$ and by continuity of the condition it follows that ω_∞ satisfies the second condition as well. Recall that ω_{n+1} satisfies

$$\nabla\psi(\omega_n) = \nabla\psi(\omega_{n-1}) - \eta_n \nabla\mathcal{L}(\omega_{n-1})$$

and thus by a telescopic sum

$$\nabla\psi(\omega_n) - \nabla\psi(\omega_0) = -\sum_{k=1}^{n} \eta_k \nabla\mathcal{L}(\omega_k) = -\sum_{k=1}^{n} \eta_k \sum_{i=1}^{N} \ell'(\omega_k^T x_i, y_i) \cdot x_i \in \operatorname{span}(x_1, \dots, x_N),$$

where ℓ' denotes the derivative of ℓ in the first component. Thus the second KKT condition holds for every ω_n. □

Remark. Note that unlike Theorem 4.2, in Theorem 4.3 we *assume* the convergence of mirror descent for given step sizes $(\eta_n)_{n\in\mathbb{N}}$, but no convergence guarantee is given for any sequence of step sizes. Hence the assumption of Theorem 4.3 are stronger than those of Theorem 4.2.

4.2.3 "The Implicit Bias of Gradient Descent on Separable Data"

This section is based on [Sou+18], which deals with the implicit bias of gradient descent for linear support vector machines. In particular, given a dataset $x_1, \dots, x_N \in \mathbb{R}^d$ with labels $y_1, \dots, y_N \in \{-1, +1\}$ we consider the homogeneous linear model $f_\omega(x) := \omega^T x$ (no bias term). The goal of SVMs is to find ω such that $\operatorname{sgn}(w^T x_i) = y_i$ for all i. The hard-margin formulation of this problem reads

$$\begin{aligned} \min_{\omega} \ & \frac{1}{2}\|\omega\|_2^2 \\ s.t. \ & y_i \cdot \omega^T x_i \geq 1, \quad i = 1, \dots, N \end{aligned} \tag{H-SVM}$$

The main result of [Sou+18] states that for linearly separable datasets and a specific class of loss-functions (β-smooth, decreasing, with an exponential tail) the iterates of gradient descent $(\omega_n)_{n\in\mathbb{N}}$ roughly behave like $\omega^* \cdot \log n$, where ω^* is the solution to the hard-margin SVM formulation (H-SVM). In particular their limiting direction is given by that of ω^*, meaning

$$\lim_{n\to\infty} \frac{\omega_n}{\|\omega_n\|} = \frac{\omega^*}{\|\omega^*\|}.$$

Note that we cannot hope for the iterates of gradient descent to converge in this case, since given some parameter $\hat{\omega}$ which perfectly classifies the data, we can often further decrease the training loss (e.g. when using the exponential loss) by simply scaling $\hat{\omega}$ by a positive scalar $\alpha > 1$. However, the decision boundary of this linear model only depends on the *direction* of ω, not on its magnitude. Hence analyzing the convergence of the direction is a reasonable alternative.
We now give the relevant definitions, precisely state the main result and give an outline for its proof:

Definition 4.2 ([Sou+18] Def. 2)**.** A function $f(u)$ has a "tight exponential tail", if there exist positive constants c, a, μ_+, μ_-, L, l, such that

$$\begin{aligned} \forall u > L : & f(u) \leq c(1 + \exp(-\mu_+ u))e^{-au} \\ \forall u > l : & f(u) \geq c(1 - \exp(-\mu_- u))e^{-au} \end{aligned}$$

Both the exponential loss $\ell(u) := \exp(-u)$ as well as the logistic loss $\ell(u) := \log(1 + \exp(-u))$ have tight exponential tails.

Theorem 4.4 (Implicit Bias of GD for linear models with exponential loss, [Sou+18], Thm 3)**.** *Assume our dataset is linearly separable. Let the loss function ℓ satisfy:*

1. *ℓ is decreasing.*
2. *ℓ is β-smooth, meaning ℓ' is β-Lipschitz.*
3. *ℓ has a tight exponential tail.*

Then if $\eta < 2\beta^{-1}\sigma_{\max}^{-2}(X)$, we have that for any starting point ω_0, the gradient descent iterates $(\omega_n)_{n\in\mathbb{N}}$ satisfy

$$\omega_n = \omega^* \log n + \rho(n),$$

where ω^* is the solution to (H-SVM) and ρ grows at most as $\|\rho(n)\| = \mathcal{O}(\log\log n)$. In particular,

$$\lim_{n\to\infty} \frac{\omega_n}{\|\omega_n\|} = \frac{\omega^*}{\|\omega^*\|}.$$

Proof. We only sketch the idea of the proof here. □

4.2.4 "The implicit bias of gradient descent on nonseparable data"

4.3 Neural Networks

4.3.1 Linear Neural Networks

References

[Gun+18] Suriya Gunasekar et al. "Characterizing implicit bias in terms of optimization geometry". In: *International Conference on Machine Learning*. PMLR. 2018, pp. 1832–1841.

[JGH18] Arthur Jacot, Franck Gabriel, and Clément Hongler. "Neural tangent kernel: Convergence and generalization in neural networks". In: *Advances in neural information processing systems* 31 (2018).

[Nes+18] Yurii Nesterov et al. *Lectures on convex optimization*. Vol. 137. Springer, 2018.

[Sou+18] Daniel Soudry et al. "The implicit bias of gradient descent on separable data". In: *The Journal of Machine Learning Research* 19.1 (2018), pp. 2822–2878.

[Lee+19] Jaehoon Lee et al. "Wide neural networks of any depth evolve as linear models under gradient descent". In: *Advances in neural information processing systems* 32 (2019).

[BD20] David Barrett and Benoit Dherin. "Implicit Gradient Regularization". In: *International Conference on Learning Representations*. 2020.

[Smi+20] Samuel L Smith et al. "On the Origin of Implicit Regularization in Stochastic Gradient Descent". In: *International Conference on Learning Representations*. 2020.

[GG23] Guillaume Garrigos and Robert M Gower. "Handbook of convergence theorems for (stochastic) gradient methods". In: *arXiv preprint arXiv:2301.11235* (2023).