Mathematics of Deep Learning

Lecture Notes

Professor
Joan Bruna Estrach

Scribes
James Bannon
Muge Chen
Ningyuan Huang
Hongyu Lu

Spring 2020

Contents

Part I

Geometric Stability and the Scattering Transform

Lecture 1

The Curse of Dimensionality

Introduction

This course is about some of the "mathematical mysteries" of deep learning. We will focus on "boring" deep learning in the supervised setting. The parallel curriculum covers deep learning in a reinforcement learning environment.

Supervised Learning Review

The main ingredients

- ▷ We need a **data space** $\mathcal{X}$, usually taken to be high-dimensional.
- ▷ A data distribution $\nu \in (\mathcal{X})$ where $(\mathcal{X})$ is the set of all probability measures on $\mathcal{X}$.
- ▷ A target function $f^* : \mathcal{X} \to \mathbb{R}$.
- ▷ A loss functional (sometimes called a risk functional) on the space $\doteq \{f :\to \mathbb{R}\}$. Typically we take this to be $(f) = x \sim \nu\ell[f(x), f^*(x)]$
- ▷ A model / hypothesis class , which we take to be a Banach space.[1]

Recall that for a space to be normed means that it is equipped with a function γ with the properties:

- ▷ $\gamma(f) \geq 0$,
- ▷ $\gamma(af) = a\gamma(f)$ for $a \in \mathbb{R}$,
- ▷ $\gamma(f + h) \leq \gamma(f) + \gamma(h)$.

We often think of the norm as giving some measure of complexity of the function f. As such we can look at the norm-balls $^{\delta} \doteq \{f \in: \gamma(f) \leq \delta\}$ as constraining the complexity. **The goal of supervised learning** is to predict f^* from finite samples of the form $\{x_p, f^*(x_p)\}_{p=1}^{P}$, $x_p \sim \nu$.
The typical way in which we approach this problem is through the empirical risk minimization (ERM) algorithm. To do this we pick a class of functions and pick the function that minimizes the average loss over our (training) data set, that is:

$$\hat{f} =_{f\in} \Big(\gamma(f) = \frac{1}{P} p = 1 P\ell\big[\hat{f}(x_p), f^*(x_p)\big]\Big). \tag{1.1}$$

The task of actually finding a better $\hat{f}$ (that generalises to the validation set) usually is done via structural risk minimization (SRM) (adding regularisation) which involves solving an optimization problem that is usually in one of three equivalent forms:

[1] A complete normed space.

Types	Complexity $\delta \uparrow$	Sample size L $\uparrow$
Approximation Error	$\downarrow$	same
Statistical Error	$\uparrow$	$\downarrow$
Optimization Error	?	?

▷ **Constraint form:**

$$\min_{f \in \delta} \gamma(f) \tag{C}$$

▷ **Penalty form:**

$$\min_{f \in} \gamma(f) + \lambda \gamma(f) \tag{P}$$

▷ **Interpolation form:**

$$\min_{f \in} \gamma(f) \quad \text{s.t.} \quad \gamma(f) = 0 \tag{I}$$

[Fundamental theorem of machine learning Bottou and Bousquet [2008]] Suppose I have an algorithm that will give me a function $\hat{f} \in^{\delta}$ such that $\gamma(\hat{f}) \leq \min_{f \in^{\delta}} \gamma(f) + \epsilon$ where $\epsilon > 0$ is a pre-chosen tolerance.

$$\begin{aligned}
(\hat{f}) - \inf_{f \in}(f) &= \inf_{f \in^{\delta}}(f) - \inf_{f \in}(f) + \inf_{f \in^{\delta}} \gamma(f) - \inf_{f \in^{\delta}}(f) + \gamma(\hat{f}) - (\hat{f}) + (\hat{f}) - \inf_{f \in^{\delta}} \gamma(f) \\
&= \inf_{f \in^{\delta}}(f) - \inf_{f \in}(f) \\
&+ \inf_{f \in^{\delta}} \gamma(f) - \inf_{f \in^{\delta}}(f) \\
&+ \gamma(\hat{f}) - \inf_{f \in^{\delta}} \gamma(f)
\end{aligned}$$

What this tells you is that how you do with your oracle function $\hat{f}$ in your constrained set, that is your error with respect to the best possible you can do, can be decomposed into three error sources:

▷ **Approximation Error:**

$$\inf_{f \in^{\delta}}(f) - \inf_{f \in}(f)$$

This tells you the penalty you pay for constraining your search to $^{\delta}$ rather than the full space .

▷ **Statistical Error/Random Fluctuation:**

$$\inf_{f \in^{\delta}} \gamma(f) - \inf_{f \in^{\delta}}(f)$$

which tells you how much the loss in your sample (which is random in the first term) contributes to error.

▷ **Optimization Error:**

$$\gamma(\hat{f}) - \inf_{f \in^{\delta}} \gamma(f)$$

which tells you how much of the error has to do with optimization.

In analyzing supervised learning settings/algorithms we have three challenges:

1. Design functional spaces with good approximation properties in high-dimensions.
2. Design efficient algorithms to solve equations (P), (I) and (C). Spoiler: these will all be gradient-based.
3. Can we control the statistical fluctuations?

1.1 The Curse of Dimensionality

We want to see how our three challenges are hard to work with due to the curse of dimensionality.

1.1.1 The Curse in Challenge 1: Statistical Fluctuations

Given observations $\{x_l, f(x_l)\}_{l=1,...,L}$, how many samples L are needed as a function of d (dimension), and the assumption of f?

1. Suppose f^* is linear: $f^* \in \{f : R^d \mapsto R, f(x) =< x, \theta >\}$. We need $L = d$ data points (under non-degenerate case).
2. Suppose f^* is "locally linear", namely f^* is Lipschitz. That is, $Lip(f^*)$ is the smallest β such that $|f^*(x) - f^*(y)| \leq \beta||x - y||, \forall x, y$. Given $F = \{f : R^d \mapsto R, Lip(f) \leq \infty\}$ is Banach (a complete normed vector space), we want to find $f \in F$ such that $\forall \epsilon \geq 0, ||f^* - f||_{L^2(R^d,V)} \leq \epsilon$ from L i.i.d samples. It turns out we need $L \sim \epsilon^{-d}$ samples. In other words, if we want to lower the error by half, we need 2^d more data points.
   - ▷ Upper bound: given $\{x_l, f(x_l)\}_{l=1,...,L}, Lip(f^*) = \beta$, find $\hat{f} = \arg\min_{f \in F}\{Lip(f), f(x_l) = f^*(x_l), l = 1, ..., L\}$
     Proof: note that $Lip(\hat{f}) \leq \beta$, let $\bar{x}$ be the nearest point of x,
     $$\begin{aligned} |\hat{f}(x) - f^*(x)| &\leq |\hat{f}(x) - \hat{f}(\bar{x})| + |\hat{f}(\bar{x}) - f^*(\bar{x})| + |f^*(\bar{x}) - f^*(x)| \\ &\leq 2\beta||x - \bar{x}|| \end{aligned}$$
     Thus, $L^{-\frac{1}{d}} \approx \epsilon \implies L \approx \epsilon^{-d}$
   - ▷ Lower bound: Maximum Discrepancy Lemma: given $\{x_l, f(x_l)\}_{l=1,...,L}, X = [0, 1]^d \in R^d, Lip(f^*) = \beta, L << 2^d, L = poly(d)$.
     $$\exists x, x', \sup_{f, Lip(f) \leq 1} |\frac{1}{L}\sum_{l=1}^{L} f(x_l) - \frac{1}{L}\sum_{l=1}^{L} f(x'_l)| \simeq L^{-\frac{1}{d}}$$
     Where x, x' represent two fresh data samples. More details can be found in Luxburg and Bousquet [2004]

1.1.2 The Curse in Challenge 2: Approximation

Suppose we're in the parametric setting. How does the approximation error change as number of dimensions change?
Let $\epsilon_n = dist(H_n; F) \doteq \sup_{f \in F} \inf_{h \in H_n} ||f - h||_{L_p}$, F represents functions with known regularity (e.g, Sobolev with order r (denote as W_p^r), $H_n = \{(\theta) : R^d \mapsto R, \dim(\theta) = n\}$ (e.g., a neural network with n parameters, $h(x) = \sum_{i=1}^{n} a_i \sigma(x_i, w + b_i)$)):

1. Universal Approximation Theorem Csáji et al. [2001]: $\epsilon_n \to 0$ as $n \to \infty$ (under mild assumptions on the activation function, and compact set R^n of input space)
2. Rate of Approximation Theorem Maiorov and Meir [1998]: for $F = W_p^r$, $\sup_{f \in F} \inf_{h \in H_n} ||f - h||_{L_p} \gtrapprox n^{-\frac{r}{d}}$. This means we need a function with r order of derivatives, which is extremely smooth as n increases.

We need exponentially many nodes

1.1.3 The Curse in Challenge 3: Optimization

Is it true that as $d \to \infty$ the algorithms for structural risk minimization in forms I, P, and C become harder to solve? It is not nearly as clear cut as the other two cases. In particular if the problem is convex then it is usually quite easy to solve. In the nonconvex case anything can happen.

1.2 Review of Feed-Forward Neural Networks

If $= (x_1, x_2, \ldots, x_n)$ is an input and we have a fully connected single hidden layer then the input layer is just the n nodes of the components of x and then hidden layer has an activation function σ then we can have the j th node output $\sigma(w_j^T + b_j)$. Stacking the w_j's into a matrix we can apply the function component-wise to $W + b$ where b is the vector of biases. Thus if we have a hidden layer of size L_1 then we get a new vector h^1 of dimension L_1 with

$$h_k^1 = \sigma(w_k^T + b_k)$$

or alternately $h^1 = \sigma(W + b)$ or

$$h^1 = \begin{pmatrix} \sigma(w_1^T + b_1) \\ \vdots \\ \sigma(w_{L_1}^T + b_{L_1}) \end{pmatrix}$$

Clearly W must be of dimension $n \times L_1$. For multiple layers we have multiple matrices $W^{(1)}, \ldots, W^{(N)}$. Then for the second layer we have the node-values as $W^{(2)}h^1 + b^1$. Unrolling gives

$$h^2 = W^{(2)}h_1 + b^2 = W^{(2)}(W^{(1)} + b) + b^1$$

which reveals the role of matrix multiplication in computing deep learning outputs. Let x be an input vector. We require an activation function σ

Lecture 2

Symmetries

In this lecture, we will discuss how to find "good" functional spaces that avoid the curse of dimensionality.

The classic notions of regularity are too generic. In the previous lecture, we looked at problems as functions between high-dimensional spaces. However, this is making the problem unnecessarily hard because, in reality, we are not interested in generic vectors of those spaces. We do not need to understand the function in full generality as long as we understand what we want to learn.

Consequently, we would like to go beyond the classic notions of regularity and make the learning problems more structured. In particular, we will focus on two concepts: symmetry and stability.

2.1 Motivation

Recall that the input space $\mathcal{X}$ we defined in the previous lecture is in some high-dimensional space $\mathbb{R}^d$, where $d \gg 1$. Fields that involve high-dimensional data include Computer Vision, Speech Recognition, Natural Language Processing, and Physics. In practice, however, the input spaces we deal with in these fields typically contain vectors that are functions themselves that map low-dimensional domains Ω to low-dimensional target spaces $\mathbb{R}^k$, i.e.

$$\mathcal{X} = \left\{x : \Omega \to \mathbb{R}^k\right\},$$

where k is small.

Example 2.1.1. In Computer Vision, we usually deal with images. We can have a function x that maps the position $u \in \Omega = [0,1]^2$ of a given pixel in an image to a vector $x(u) \in \mathbb{R}^3$ containing the RGB values of that pixel. In Finance or Natural Language Processing, if we are dealing with time series data, then $\Omega = \mathbb{R}$ (or $\Omega = \mathbb{Z}$ if the data is discritized). In Physics, if we are dealing with a physical system containing n particles, then we can encode the particles their space-time positions in $\mathbb{R}^4$ and their momentum also in $\mathbb{R}^4$. Therefore, we have $\Omega = (\mathbb{R}^4 \times \mathbb{R}^4)^N$.

Notice in the example above that Ω's typically have low-dimensional structures. We now define a more appropriate functional space for these structures.

2.2 Symmetry

Symmetries are known transformations of the domains that do not modify the functions we want to learn.

Definition 2.2.1 (Symmetry)**.** Let $\mathcal{F} := \{f : \mathcal{X} \to \mathbb{R}\}$ be a family of target tasks. A **symmetry** of $\mathcal{F}$ is an invertible transformation $T : \mathcal{X} \to \mathcal{X}$ such that

$$f(x) = f(Tx)$$

for all $f \in \mathcal{F}$ and $x \in \mathcal{X}$.

In physics, the *Noether's theorem* states the correspondence between symmetry and conservation law ("invariant"). One can think of the symmetry of a physical system as the quantity conserved as the input moves. On the other hand, the French mathematician Galois tried to understand the structure of solutions of polynomials by studying symmetries on polynomials (see *Galois theorem*).

However, it is challenging to find symmetries in a large high-dimensional space $\mathcal{X}$. We can instead try to exploit the low-dimensional structure of Ω. In particular, we would like to find out the relationships between symmetries in $\mathcal{X}$ and symmetries in Ω. We start by looking at symmetries in Ω since it is more approachable.

Example 2.2.2. We define the *translation* $T_\tau : \Omega \to \Omega$ on the domain Ω by

$$u \mapsto u - \tau,$$

where $u, \tau \in \Omega$.

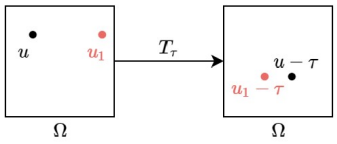

Figure 2.1: Translation on Ω (shown as 2-dimensional). The periodic boundary conditions are shown as the pink points.

Here, we omit the boundary effects and assume periodic boundary conditions. If we translate u around the boundary of Ω, then $u - \tau$ will appear on the other side of Ω (see Figure 2.1). Then, we can apply the translation on Ω to the input space $\mathcal{X} = \{x : \Omega \to \mathbb{R}\}$ and define the translation $T_\tau : \mathcal{X} \to \mathcal{X}$ on the input space $\mathcal{X}$ by

$$x \mapsto [u \mapsto x(u - \tau)].$$

In general, given any rigid transformation T on Ω defined by

$$u \mapsto Au + \tau,$$

where A is the orthogonal transformation, we can upgrade it to the rigid transformation $T : \mathcal{X} \to \mathcal{X}$ and therefore obtain symmetries on the input space $\mathcal{X}$.

Symmetric Group G under Composition

We note that the composition $T_1 \circ T_2$ of two symmetries T_1 and T_2 is also a symmetry. The set of symmetries T forms a group G under composition $\circ$. This group is not abelian since it involves matrix multiplications.

2.3 Invariant and Equivariant Representations

Definition 2.3.1 (Representation)**.** A **representation** is a mapping $\Phi : \mathcal{X} \to \mathcal{Z}$, where $\mathcal{X}$ is a high-dimensional input space and $\mathcal{Z}$ is a feature space.

We hope to find a good Φ for learning problems in deep learning. Once we find the good representation, we can then estimate a regression function $\widehat{f}$ with a linear or logistic model in $\mathcal{Z}$

$$f(x) = \begin{cases} \langle \theta, \phi(x) \rangle & \text{(regression)} \\ \sigma(\langle \theta, \phi(x) \rangle) & \text{(classification)} \end{cases}$$

We connect representation with symmetry as follows.

Definition 2.3.2 (Invariant)**.** A representation $\Phi : X \to Z$ is **invariant** with respect to a group G of transformations if

$$\Phi(Tx) = \Phi(x)$$

for all $x \in \mathcal{X}$.

Definition 2.3.3 (Equivariant)**.** A representation Φ is **equivariant** with respect to a group G of transformations if

$$\Phi(Tx) = \widetilde{T}\Phi(x),$$

where $\widetilde{T} \in \widetilde{G} \cong G$.

Note that here equivariance is equivalent of a commutation property. When Φ and T commute, we need to redefine the transformation from the feature space.

We see a consequence that follows from the definitions: if Φ is G-invariant, then $\widehat{f}$ is also G-invariant.

Example 2.3.4. Let $x(u) \in L^1(\mathbb{R})$. First, we define a representation (signal sum)

$$\Phi(x) = \int_{-\infty}^{\infty} x(u)\,\mathrm{d}u.$$

Then Φ is invariant to translation $T_\tau x(u) = x(u-\tau)$:

$$\Phi(T_\tau x) = \int_{-\infty}^{\infty} x(u-\tau)\,\mathrm{d}u = \int_{-\infty}^{\infty} x(u)\,\mathrm{d}u = \Phi(x).$$

Now, we define a representation of the *DC*

$$\widehat{\Phi}(x) = \lim_{N\to\infty} \underbrace{\frac{1}{2N}\int_{-N}^{N} x(u)\,\mathrm{d}u}_{\widehat{\Phi}_N(x)},$$

where we assume $|x(u)| \leq M$. We show that $\widehat{\Phi}$ is also invariant to translation as follows. First note that

$$\begin{aligned}
\lim_{N\to\infty} \widehat{\Phi}_N(x_\tau) &= \lim_{N\to\infty} \frac{1}{2N}\int_{-N}^{N} T_\tau x(u)\,\mathrm{d}u \\
&= \lim_{N\to\infty} \frac{1}{2N}\int_{-N}^{N} x(u-\tau)\,\mathrm{d}u \\
&= \lim_{N\to\infty} \frac{1}{2N}\int_{-N+\tau}^{N+\tau} x(u)\,\mathrm{d}u \\
&= \lim_{N\to\infty} \underbrace{\frac{1}{2N}\int_{-N}^{N} x(u)\,\mathrm{d}u}_{\Phi_N(x)} + \lim_{N\to\infty} \underbrace{\frac{1}{2N}\left[\int_{N}^{N+\tau} x(u)\,\mathrm{d}u - \int_{-N}^{-N+\tau} x(u)\,\mathrm{d}u\right]}_{|\Phi_N(x_\tau)-\Phi_N(x)|}
\end{aligned}$$

Since x is bounded, we have

$$\lim_{N\to\infty} \frac{1}{2N}\left[\int_{N}^{N+\tau} x(u)\,\mathrm{d}u - \int_{-N}^{-N+\tau} x(u)\,\mathrm{d}u\right] \leq \lim_{N\to\infty} \frac{1}{2N} 2\tau M = 0,$$

showing that the DC $\widehat{\Phi}$ is invariant to translation.

Note that Φ is not invariant to dilation, i.e.

$$x_s(u) \neq x(su).$$

Example 2.3.5. Let $x(u) \in L^1(\mathbb{R})$ the signal with compact support. We define a representation of *the center of the mass of* x

$$\overline{\Phi}(x) = \frac{\int u\,|x(u)|\,\mathrm{d}u}{\int |x(u)|\,\mathrm{d}u},$$

where u is the expected value. We show that $\overline{\Phi}$ is equivariant to translation as follows.

$$\overline{\Phi}(T_\tau x) = \frac{\int u\,|x(u-\tau)|\,\mathrm{d}u}{\int |x(u-\tau)|\,\mathrm{d}u} = \frac{\int (u+\tau)\,|x(u)|\,\mathrm{d}u}{\int |x(u)|\,\mathrm{d}u} = \overline{\Phi}(x) + \frac{\tau\int |x(u)|\,\mathrm{d}u}{\int |x(u)|\,\mathrm{d}u} = \overline{\Phi}(x) + \tau.$$

2.4 Building Group Invariant in a Systematic Way

To describe a systematic way of building abelian group invariant in a Hilbert space $\mathcal{X}$, we utilize a special abelian group.

Definition 2.4.1 (One-parameter Unitary Group)**.** A **(strongly continuous) one-parameter unitary group** is a family of operators $\{\varphi_t : \mathcal{X} \to \mathcal{X}\}_{t\in\mathbb{R}}$ of $\mathcal{X}$ such that

1. $\varphi_{s+t} = \varphi_s\varphi_t$ for all $t, s \in \mathbb{R}$,
2. $\lim_{s\to t} \|\varphi_s - \varphi_t\| = 0$ for all $t, s \in \mathbb{R}$,
3. $\|\varphi_t x\| = \|x\|$ for all $x \in \mathcal{X}$ and $t \in \mathbb{R}$.

Note that the one-parameter unitary group is abelian because of the condition i). We connect one-parameter unitary groups to self-adjoint operators. This is due to Stone.

Definition 2.4.2 (Self-adjoint Operator)**.** A linear operator A on a finite-dimensional space V with inner product $\langle\cdot,\cdot\rangle$ is **self-adjoint** (**Hermitian**) if

$$\langle Au, w\rangle = \langle u, Aw\rangle$$

for all $u, w \in V$.

Theorem 2.4.3 (Stone's theorem)**.** *There is a one-to-one correspondence between self-adjoint operations and one-parameter unitary groups of $\mathcal{X}$.*

In particular, given a one-parameter unitary group $\{\varphi_t\}_{t\in\mathbb{R}}$, there exists a self-adjoint operator A such that for all $t \in \mathbb{R}$,

$$\varphi_t = e^{itA}.$$

(see more details in the original publication) Here, A is called the **infinitesimal generator** of the group. Furthermore, by saying A is self-adjoint we mean that there exists an orthogonal basis $\{v_k\}_k$ of $\mathcal{X}$ such that

$$Av_k = \lambda_k v_k.$$

where $\lambda_k \in \mathbb{R}$ are the eigenvalues. We can see that

$$\varphi_t v_k = \sum_{n=0}^{\infty} \frac{(it)^n}{n!} A^n v_k = \sum_{n=0}^{\infty} \frac{(it)^n}{n!} \lambda_k^n v_k = e^{it\lambda_k} v_k$$

for all k (see Borel Functional Calculus). In matrix form, we have

$$A = V\Lambda V^*.$$

Note that V is the "Fourier" transform of the unitary group.

Suppose we have $x \in \mathcal{X}$. To understand the effect of the transformation φ_t on x, first, we write x as a linear combination of basis vectors v_k

$$x = \sum_k \alpha_k v_k,$$

where $\alpha_k \in \mathbb{C}$. Then,

$$\varphi_t x = \sum_k \alpha_k \varphi_t v_k = \sum_k \alpha_k e^{it\lambda_k} v_k.$$

To visualize the transformation, consider $V \in \mathbb{C}^{d\times d}$, where $d = \dim \mathcal{X}$. Then, the transformation is merely the rotation of α_k.

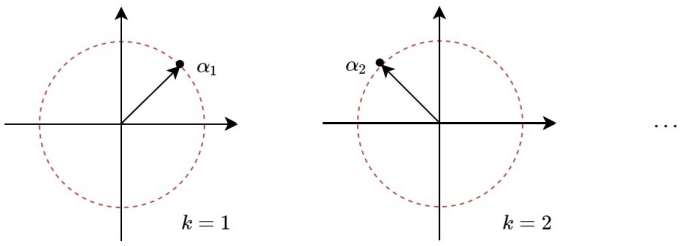

Figure 2.2: Visualization of the transformation φ_t on x

Example 2.4.4. Given the translation $\varphi_\tau x(u) = x(u-\tau)$. Recall that for $x \in L^2(\mathbb{R}^d)$, the Fourier transform of x is

$$\widehat{x}(\xi) = \int x(u) e^{-2\pi i \langle u, \xi \rangle} \, \mathrm{d}u.$$

Then, the Fourier transform of the translation is

$$\widehat{(\varphi_\tau x)}(\xi) = \int x(u-\tau) e^{-2\pi i \langle u, \xi \rangle} \, \mathrm{d}u = \int x(u) e^{-2\pi i \langle u+\tau, \xi \rangle} \, \mathrm{d}u = e^{-2\pi i \langle \tau, \xi \rangle} \widehat{x}(\xi).$$

We can now build Φ that is $G = \{\varphi_t\}_t$-invariant by defining

$$\Phi(x) = |V^* x| \, .$$

Claim. Φ is G-invariant.

Proof. Since

$$V^* x = (\alpha_1, \alpha_2, \ldots, \alpha_d),$$

we have

$$V^* \varphi_t x = (\alpha_1 e^{it\lambda_1}, \alpha_2 e^{it\lambda_2}, \ldots, \alpha_d e^{it\lambda_d}).$$

Therefore,

$$\Phi(\varphi_t x) = |V^* \varphi_t x| = \left(\left| \alpha_1 e^{it\lambda_1} \right|, \left| \alpha_2 e^{it\lambda_2} \right|, \ldots, \left| \alpha_d e^{it\lambda_d} \right| \right) = \left(|\alpha_1|, |\alpha_2|, \ldots, |\alpha_d| \right) = |V^* x| = \Phi(x)$$

for all t. □

Note that we do not need a deep representation for invariance to abelian structure since the definition of Φ indicates a linear transformation followed by an element-wise nonlinearity.

Shortcomings

Since $\Phi(x) = 0$ is also invariant, we need a "representational power" besides invariance to preserve more information. In other words, if $f^*(x) \neq f^*(x')$, we need $\Phi(x) \neq \Phi(x')$. In the definition of Φ above, we lost d degrees of freedom.

For instance, the Fourier transforms of a signal with Dirac distribution and a white noise with standard Gaussian distribution are both flat.

Lecture 3

Geometric Stability

In the last lecture, we learned to construct a (nonlinear) invariant representation for one-parameter unitary groups G

$$\Phi(x) = |V^* x|,$$

where V is the "Fourier transform" of G. Moreover, we can build such representation for more general groups as long as the abelian structure is preserved. In fact, we can extend to non-abelian structures using the *Peter-Weyl theorem* in group representation theory.

At the end of the last lecture, we described several possible limitations of such Φ.

- ▷ Symmetries are very low-dimensional. Are they sufficient to beat the curse of dimensionality?
- ▷ How robust is Φ to "quasi"-symmetries?
- ▷ How much information does Φ preserve?

We will justify these limitations in this lecture.

3.1 Deformation Models

Consider a function $x \in L^2(\mathbb{R}^d)$, where $d = 2$. In the previous lecture, we considered affine transformations $T : \mathbb{R}^d \to \mathbb{R}^d$, but there are very few of such transformations. In this lecture, we relax the condition and consider transformations that are not necessarily linear but still preserve enough structures.

In particular, we look at transformations $\varphi : \mathbb{R}^d \to \mathbb{R}^d$ that are invertible, smooth, and have smooth inverses. These transformations are **diffeomorphisms**. Note that diffeomorphisms also form a group, and it is a much bigger group than the group of rigid motions.

Suppose we have a mapping $\varphi(u) = u - \tau(u)$, where $\tau : \mathbb{R}^d \to \mathbb{R}^d$ is a **deformation**. To show that φ is a diffeomorphism, it is sufficient to show that

$$\|\nabla\tau\|_\infty = \sup_u \|\nabla\tau(u)\| < 1.$$

Signal Deformation

Consider $x_\tau \in L^2(\mathbb{R}^d)$, then we have

$$x_\tau(u) = x(u - \tau(u)) = x(\varphi(u)).$$

We can think of it as a form of noise. In terms of images, we typically add noise to the pixel values ($\widetilde{x}(u) = x(u) + n$) and the structures are preserved. Here, we instead add noise to the locations of the pixels (geometric noise).

3.2 Geometric Stability

Now, we introduce a notion to generalize regularity.

Definition 3.2.1 (Geometric Stability)**.** A function $f :\in L^2(\mathbb{R}^d) \to \mathbb{R}$ is **geometrically stable** with respect to a norm on τ if for all $x \in L^2(\mathbb{R}^d)$ and for all τ such that $\|\nabla\tau\|_\infty < 1$,

$$|f(x) - f(x_\tau)| \lesssim \|\tau\|.$$

Loosely speaking, the definition states that if we take an image x and apply a diffeomorphism x_τ, then the function f will be stable with respect to the deformation τ, i.e. the value of the function does not change much. More formally, given any x, we have a function F_x that takes any deformation τ and maps it to $f(x_\tau)$. Then, $F(0) = f(x)$ and $F(\tau) = f(x_\tau)$. We have $|F(\tau) - F(0)| \lesssim \|\tau\|$. In the definition, f depends on x. But now, we have a function F that depends on the deformation and we would like this function to have some regularity. Note that here F is *Lipschitz* with respect to $\|\tau\|$.

Global Geometric Stability

Next, we need to define a way to measure the deformation constant $\|\tau\|$. First note that $\tau \in \mathbb{R}^{d\times d}$. We look at the gradient $\nabla\tau$ of τ because we would like to see how much the domain is stretched (in Physics, the gradient of a velocity field measures elasticity of the field). Then, the **global geometry stability** is defined for τ to be

$$\|\tau\|_G = \sup_u |\nabla\tau(u)|,$$

where $u \in \Omega \in \mathbb{R}^d$.

Example 3.2.2. Given a translation $T : \mathbb{R}^d \to \mathbb{R}^d$ with $u \mapsto u - v$, the corresponding deformation is $\tau_0(u) = v$. Then, the deformation constant for translation is $\|\tau_0\| = 0$.

Stable Representation

Now we can define a stable representation similar to invariant representation we introduced in the last lecture.

Definition 3.2.3 (Stable Representation)**.** A mapping $\Phi : L^2(\mathbb{R}^d) \to \mathcal{F}$ is a **stable representation** if

$$\|\Phi(x_\tau) - \Phi(x)\| \lesssim \|\tau\|\|x\|$$

for all x and τ.

If we use Φ for classification or regression $\widehat{f}(x) = \langle\theta, \Phi(x)\rangle$, then the geometric stability at the level of the representation implies the geometric stability at the level of the predictors. That is, by *Cauchy-Schwarz inequality*, we have

$$\left|\widehat{f}(x) - \widehat{f}(x_\tau)\right| \le \|\theta\|\|\Phi(x) - \Phi(x_\tau)\|.$$

Local Geometric Stability

Instead of looking for notion of invariance that involves the whole group, We are sometimes more interested in the invariance locally. The **local geometric stability** is defined for τ to be

$$\|\tau\|_J = 2^{-J}\|\tau\|_\infty + \|\nabla\tau\|_\infty.$$

3.3 Geometric Stability of Fourier Invariants

In the last lecture, we discussed the Fourier invariant $\Phi(x) = |\widehat{x}|$. We know that the transformation x is invariant to translation. We want to know the behavior of the representation when x deviates from the translation group.

Consider the one-dimensional cases $x \in L^2(\mathbb{R})$ with $x(u) = e^{i\xi_0 k} h(u)$, where $h(u)$ is the lowpass window and ξ_0 is the central frequency. Given a deformation $x_\tau(u) = x((1+\epsilon)u)$, where $\epsilon \ll 1$. Then, the Fourier transform of x and x_τ are

$$\widehat{x}(\xi) = \widehat{h}(\xi - \xi_0)$$
$$\widehat{x}_\tau(\xi) = (1+\epsilon)^{-1}\widehat{x}((1+\epsilon)\xi) = (1+\epsilon)^{-1}\widehat{h}((1+\epsilon)^{-1}\xi - \xi_0).$$

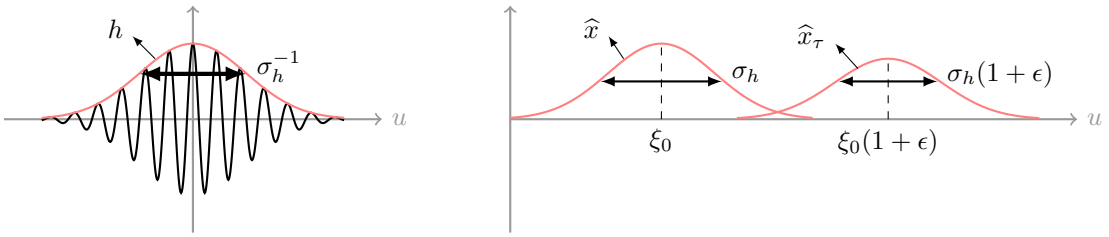

Figure 3.1: Left: transformation $x(u)$. Right: Fourier transform $\widehat{x}(u)$ and $\widehat{x}_\tau(u)$.

In terms of stability, recall that stability of the deformation is uniform over all x. If

$$(1+\epsilon)\xi_0 - \xi_0 = \epsilon\xi_0 \gg \sigma_n(2+\epsilon),$$

then

$$\|\Phi(x_\tau) - \Phi(x)\| = \||\widehat{x}_\tau| - |\widehat{x}|\| \sim \|x\|.$$

That is, if the gap between the central frequencies is much larger than the sum of the bandwidths, then $|\widehat{x}|$ and $|\widehat{x}_\tau|$ are disjoint. This means that high frequencies in this representation are unstable and $|\widehat{x}|$ loses information in phase correlations.

To process the information in high-frequency so that it becomes stable, we can build local invariants instead. Recall the unitary group $\varphi_v x$ of all transformations of the function x with $\|\varphi_v x\| = 1$ for all v. We want to have a representation that satisfy

$$\|\Phi(x) - \Phi(\varphi_v x)\| \le 2^{-J}\|v\| \quad \Longleftrightarrow \quad \frac{\|\Phi(x) - \Phi(\varphi_v x)\|}{\|v\|} \le 2^{-J}.$$

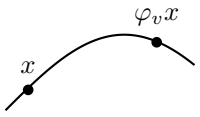

Figure 3.2: Mapping from x to $\varphi_v x$ with $\|\varphi_v x\| = 1$ for all v.

We can interpret it as some form of derivative with respect to v and the derivative needs to be small. We create a smooth function

$$\Phi(x) = 2^{-J}\int_G h(2^{-J}v)\varphi_v x \,\mathrm{d}\mu(v),$$

where μ is the Haar measure of G and $h \le 0$ with $\int_G h(v)\,\mathrm{d}\mu(v) = 1$. For translation group, we have

$$\Phi(x)(u) = 2^{-J}\int_{\mathbb{R}} h(2^{-J}v)x(u-v)\,\mathrm{d}v = (x * h_J)(u),$$

which corresponds to blurring by h_j. In fact, we have stability beyond pure translation.

Proposition 3.3.1. *$\Phi(x) = x * h_J$ satisfies*

$$\|\Phi(x) - \Phi(x_\tau)\| \leq C\|\tau\|_J$$

for all x with $\|x\| = 1$ and for all τ with $\|\nabla\tau\|_\infty \lesssim 1$.

Proof. First recalled that the *integral operator* $K : L^2(\mathbb{R}^d) \to L^2(\mathbb{R}^d)$ is defined as

$$(Kf)(u) = \int k(u, u') \cdot f(u') \, \mathrm{d}u.$$

We can find an upper bound for K using the *Schur lemma*

$$\|K\| \leq \max\left(\sup_u \int |k(u, u')| \, \mathrm{d}u', \sup_{u'} \int |k(u, u')| \, \mathrm{d}u\right).$$

All that is left is to find the kernel $k(u, u')$. First note that $x \mapsto \Phi(x) - \Phi(x_\tau)$ is linear. We have

$$\Phi(x) - \Phi(x_\tau) = Kx(u) = \int k(u, u')x(u') \, \mathrm{d}u'.$$

Now, consider the mapping $x \mapsto x * h_J$, where

$$(x + h_J)(u) = \int x(u')h_J(u - u') \, \mathrm{d}u'.$$

We can see that $k(u, u') = h(u - u')$. Recall that $x_\tau(u) = x(u - \tau(u))$. Then, by definition,

$$\Phi(x_\tau)(u) = \int x(\widetilde{u})h_j(u - \widetilde{u} + \beta(\widetilde{u})) \det(I - \nabla\beta(\widetilde{u})) \, \mathrm{d}\widetilde{u}.$$

We need to rewrite this integral so that it looks like the kernel using change of variables. Let $\widetilde{u} = u' - \tau(u')$. Then $u' = \widetilde{u} - \beta(\widetilde{u})$, where β is the inverse of τ. We have

$$\Phi(x_\tau)(u) = \int x(\widetilde{u})h_j(u - \widetilde{u} + \beta(\widetilde{u})) \det(I - \nabla\beta(\widetilde{u})) \, \mathrm{d}\widetilde{u}.$$

It follows that

$$k(u, u') = h_J(u - u') - h_J(u - u' + \beta(u')) \det(I - \nabla\beta(u')).$$

Note that $\nabla\beta(u')$ is small because it is controlled by the norm of the deformation, So the determinant is close to 1. We also note that $h_J(u, u') - h_J(u - u' + \beta(u'))$ is also small since h_J is smooth. Therefore,

$$h_J(u - u' + \beta(u')) = h_J(u - u') + \underbrace{\langle \nabla h_J(u - u'), \beta(u') \rangle}_{\simeq 2^{-J}\|\beta\|_\infty} + O(2^{-J} \parallel \beta n). \qquad \square$$

3.4 Other Stable Invariant Linear Operators

So far, we have discussed translation as a stable invariant linear operator. We would like to know if there are other invariant linear operators that are stable. Recall that a representation is invariant if for all v,

$$\Phi(x) = \Phi(\varphi_v x) \qquad \Longrightarrow \qquad \Phi(x) = \frac{1}{\mu(G)} \int_G \Phi(\varphi_v x) \, \mathrm{d}\mu(v).$$

In addition, Φ is linear if

$$\Phi(x) = \Phi\left(\frac{1}{\mu(G)} \int \varphi_v x \, \mathrm{d}\mu(v)\right).$$

We see that the only linear invariant that is stable is the local average. However, local average loses too much information. Therefore, we need to find an invariant representation which extracts information that complements average (wavelet) and process the information with a non-linearity (scattering transform).

Lecture 4

Scattering Transform

Recall

The only linear invariant to (local) translations is the (local) average, which is stable. But this is clearly insufficient for practice. Question: How to combine local average with appropriate non-linear operators? We need non-linear transformations.
Recall local average: $A_J x = x * \phi_J$, We can capture high frequency information with a high-pass filter: $B_J x = x - A_J x$. One way is to use Classic Harmonic Analysis (Fourier Transform) $x(u) = \sum_{k=0}^{\infty} \alpha_k e^{ik\pi u}$, but we have seen its downside. Can we do better?

4.1 Wavelets

4.1.1 Definition

A wavelet $\psi \in L^2(R^d)$ has at least one vanishing moment and is 'well-localized' both in space and in frequency. Formally:

$$\int (1 + |u|^k)\psi(u)du \leq \infty, \int (1 + |\xi|^k)|\hat{\psi}(u)|d\xi \leq \infty$$

One vanishing moment is defined as:

$$\int \psi(u)du = 0$$

$k^* + 1$ vanishing moment is defined as:

$$\int (u^k \psi(u)du = 0, \forall k \leq k^*$$

4.1.2 Wavelet Examples

1. Haar wavelet: $\psi_{j,k}(u) = \psi(2^j u - k)$. the simplest wavelet with one-vanishing property.

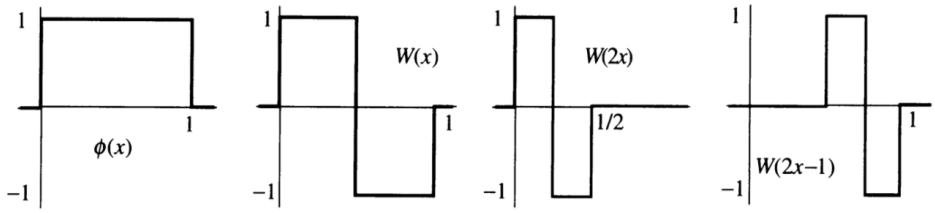

Figure 4.1: Haar wavelets, figure borrowed from Strang [1993]

2. Gabor wavelet: $\psi_{\sigma,\xi_0}(u) = \exp^{i\xi_o u} g_\sigma(u)$, where $g_\sigma(u) = \exp^{-u^2/2\sigma^2}$. this wavelet minimizes the uncertainty of information it carries (by minimizing the product of its standard deviation in time and frequency domain). However, its fourier transform is shifted and not centered at 0, so it doesn't fulfill one-vanishing property.

3. Morlet wavelet: $\tilde{\psi}_{\sigma,\xi_0}(u) = \psi_{\sigma,\xi_0}(u) - \alpha g_\sigma(u)$, where α is chosen such that $\hat{\tilde{\psi}}(0) = 0$. This remedies the one-vanishing problem in Gabor wavelet and qualifies as a real wavelet.

4.1.3 Littlewood-Paley Wavelet Decomposition

Fourier Transform: express the signal in terms of frequency;
Wavelet Transform: express the signal in terms of scale/resolution.
Why scales? This could beat the curse of dimensionality (e.g., signal has frequency N, which then only has scales of $\log N$)

Definition 4.1.1. A wavelet decomposition $\{\Psi_{j,\theta}\}_{j,\sigma}$ is obtained as $\Psi_{j,\theta}(u) = 2^{-2J}\Psi(R_\theta 2^{-J}u)$, where R_θ is a rotation by θ degrees, dilation on a scale of 2 and $u \in R^2$

Definition 4.1.2. A Littlewood-Paley wavelet decomposition of $x \in L^2(\Omega)$ is $Wx = \{x*\phi_J, x*\psi_{j,\theta}\}_{j\in\mathbb{Z}, j\leq J, \theta\in G}$, where G is the rotation group, ϕ_J represents the blurring effect (lowpass filter), ψ_j represents the wavelet decomposition (different scales).

Theorem 4.1.3 (Littlewood-Paley)**.** *If there exists* $0 \leq \delta \leq 1$ *such that* $\forall\xi$*,*

$$1 - \delta \leq |\hat{\phi}(2^J\xi)|^2 + \frac{1}{2}\sum_\lambda |\hat{\Psi}(2^j R_\theta \xi)|^2 \leq 1$$

then$\forall x \in L^2(\Omega)$*,*

$$(1-\delta)||x|| \leq ||Wx|| \leq ||x||$$

Note that W is a linear operator, whose Lipschiz bound is controlled by its singular values.

4.1.4 Translation Equivariance and Stability to Deformations

Summary:

$$Wx = \{x * \phi_J, x * \phi_{j,\theta}\}_{\lambda\in\Lambda_J}, ||Wx - W(x+n)|| \leq ||n||$$

Which means if you move in input space by a little, you should only move in feature space by less than the same amount. So W is well-behaved for additive noise, is it also stable to deformation?
Note that W is translation equivariant: $x_b(u) = x(u-b); Wx_b = (Wx)(b)$.
Recall that if $\phi_\tau x(u) = x(u - \tau(u))$ we saw that $||A_J x - A_J\phi_\tau x|| \leq ||x||||\tau||$

Theorem 4.1.4. *[Mallat, '10] There exists* $c > 0$ *s.t.* $\forall J > 0$ *and* $\forall\tau$ *with* $|\sigma_c|_\infty < \frac{1}{2}$*, then*

$$||W\phi_\tau - \phi_\tau W|| < ||\tau||$$

See Mallat [2012] for definition of $||\tau||$ but we can calculate this and show that it's mostly equivariant to deformations.

Proof. Key tool: Cotlar stein near-orthogonality lemma. If $\{T_j\}_j$ is a family of operators between $T_j : E \to E$ with a hilbert space E,

$$||\sum_j T_j|| \leq \sum_j ||T_j||$$

But this isn't enough! We have infinite scales so this bound is not sufficient. However, the T_j wavelets are near-orthogonal, so we can do better.
Idea: look at a single sub-band λ,

$$W_\lambda x = x * \Psi_\lambda$$

Define $[W_\lambda, \phi_\tau] = W_\lambda \phi_\tau - \phi_\tau W_\lambda$, which measures the commutation error (between the wavelet decomposition and the deformation). Its kernel is given by:

$$K_\lambda(u,v) = \Psi_\lambda(u + \tau^{-1}(u - \tau(u)) - v) - \Psi(\lambda(u + \tau^{-1}(v) - v) \det |I - \nabla \tau'|$$

Thus: $[W_\lambda, \phi_\tau]x(u) = \int K_\lambda(u,v)x(v)dv$
If the wavelet is high frequency (j small), then the function $\tau(u)$ is smooth; If the wavelet is low frequency (j large), then itself is smooth. Both cases we could bound the error.
Recall our goal is to characterize nonlinearty of M: $L^2(\Omega) \to L^2(\Omega)$ with the following properties:

- ▷ Preserve additive stability $||Mx - Mx'|| \leq ||x - x'||$
- ▷ Preserve geometric stability: commutes with deformations ϕ

Then [Bruna '12] shows pointwise nonlinearities are the only ones satisfing the above properties □

4.2 Scattering Transform

4.2.1 Definition

For each "path" $p = \lambda_1, \ldots, \lambda_m)$ with $\lambda_i \in \Lambda_J$,
the scattering transform of $x \in L^2(\Omega)$ is $S_J[p]x := |||x * \phi_{\lambda_1} * \ldots * x * \phi_{\lambda_m}||| * \phi_J$

4.2.2 Geometric Stability

We have M (the nonlinearity), W (the wavelet transform) and A (the lowpass filter).
Our representations are $A, AWM, AWMWM, AWMWMWM$....
W is "nearly" commutative, and M is commutative, so we pay a cost every time we do a WM transformation.
Thus - we can upper bound by something like Mc with cost c.

Lecture 5

Beyond Scattering and Euclidean Data

Introduction

Last Week

▷ We saw the basic building blocks of scattering representations.

1. A smoothing operator A_J
2. A wavelet decomposition W_J
3. A nonlinear activation function M.

The scattering representation

$$S = \{A_J, A_J MW, A_J MW_J MW_J, \ldots\}$$

provides

◇ Stability to additive noise
◇ Stability to geometric noise

Today

We address a few remaining questions:

1. What is the information content in S?
2. Can we extend this beyond the translation group?
3. Can we go from scattering to convolutional neural networks (CNNs)?

Definition 5.0.1 (Scattering Metric)**.** For $x \in L^2(\Omega)$ We define the scattering metric as

$$||S(x)||^2 = \sum_p ||S[p]||^2_{L^2(\Omega)}$$

Where $p = (\lambda_1, \ldots, \lambda_m), \lambda_i \in \Lambda_j$ is indexity the scattering path.

Theorem 5.0.2. *Assuming $\{A_J, W_J\}$ satisfy the Littlewood-Paley condition (see last week) then the scattering metric is non-expansive, that is,*

$$||Sx - S\widetilde{x}|| \leq ||x - \widetilde{x}||.$$

Moreover, if ϕ are unitary wavelets, then

$$||Sx|| = ||x||.$$

Proof.

$$||A_J x - A_j \widetilde{x}||^2 + \sum_\lambda |||x * \Psi_\lambda - \widetilde{x} * \Psi_\lambda|||^2 \leq ||x - \widetilde{x}||^2$$

Let $x_\lambda = x * \Psi_\lambda$ and $\widetilde{x}_\lambda = \widetilde{x} * \Psi_\lambda$. We have

$$||A_J x_\lambda - A_J \widetilde{x}_\lambda||^2 + \sum_{\lambda'} |||x_\lambda * \Psi_{\lambda'}| - |\widetilde{x}_\lambda * \Psi_{\lambda'}|||^2 \leq ||x_\lambda - \widetilde{x} + \lambda.$$

Observing that

$$||A_J X||^2 + \sum_\lambda ||x * \Psi_\lambda||^2 = \sum_{|p| \leq m} ||S[p]x||^2 + \sum_{|p|=m} ||U[p]x||^2,$$

where $||U[p] = W_{\lambda_p} M W_{\lambda_{p-1}} \dots M W_{\lambda_1}$. This implies that we need

$$\lim_{m \to \infty} \sum_{|p|=m} ||U[p]x||^2 = 0,$$

which follows from propagation to lower frequency. □

Energy is eventually absorbed into the coefficients and the representation captures the scale and orientation interactions. However, due to the tree-like architecture of the scattering transform we don't recombine features.

Join vs. Separable Group Invariance

Let $G = G_1 \times G_2$ be the semi-direct product of two subgroups with G_1 a normal subgroup of G. Suppose more generally we can factor (in a not-necessarily Abelian way) an element g of G such that

$$g = g_1, \dots, g_k; g_i \in G_i$$

We can leverage factorizations by looking at invariances and equivariances in each subgroup.
Suppose ϕ_1 is G_1 invariant and G_2 equivariant and ϕ_2 is G_2 invariant. Let $\phi = \phi_2 \circ \phi_q$. Then ϕ satisfies

$$\begin{aligned}
\phi(x_g) &= \phi(x_{g_1 g_2}) \\
&= \phi_2 \phi_1(\phi(x_{g_1 g_2}) \\
&= \phi_2 \phi_1(x_{g_2}) \text{ b.c } \phi_1 \text{ is } G_1 \text{ invariant} \\
&= \phi_2(\phi_1(x)_{g_2}) \text{ b.c } \phi_2 \text{ is } G_2 \text{ equivariant} \\
&= \phi_2 \phi_1 x = \phi(x)
\end{aligned}$$

So we can build larger invariants by composing simple equivariants and invariants. Note that just because the group can be factored the group action on the data cannot necessarily be factored similarly.
As a result we want to construct a joint scattering representation.
Suppose G is a compact group. We can define the group convolution as

$$x *_G h(g) = \int_G x_g h * g^{-1} \, \mathrm{d}\mu(g),$$

where μ is the Haar measure. Group convolutions are the only linear operators which are equivariant with respect to G.

Example 5.0.3. If $x(u)$ for $u \in \Omega \subset \mathbb{R}^2$, then the first step is to construct the wavelet decomposition over the translation group $W_j = \{\Psi_{j,\theta} : j \in \mathbb{Z}, \theta \in SO(2)\}$ then $U(x)(p_j) = |x * \Psi_{j,\theta}|(u)$ with $p_j = (u, j, \theta)$ is equivariant with respect to G.

Moving to CNNs: we can write a CNN layer as

$$x_{\lambda'}^{\ell+1} = \rho \left[\sum_\lambda \Psi_\lambda x^\ell \theta_{\lambda,\lambda'} \right],$$

where $\theta_{\lambda,\lambda'}$ is a trainable filter. The deformation stability we had can be transferred to CNNs if we have

Euclidean to Non-Euclidean Stability

We have, so far, considered a fixed domain Ω having global symmetries models as group transformations. Many applications lack this structure, e.g. graphs. Suppose Ω now has a generic metric structure. We can think of the diffeomorphisms now as being changes in the metric.

$$\langle x, x' \rangle = \int x(u)x'(u)x(u-\tau(u))x'(u-\tau(u))d\mu(u)$$
$$\langle x, x' \rangle_\tau = \int x(u-\tau(u))x'(u-\tau(u))\mu(du) = \int x(u)x(u')\underbrace{|I-\nabla\phi(u)|}_{\text{new metric}}\mu(du)$$

If we define a distance between metric spaces d then our new definition of geometric stability is

$$||\phi_x - \phi_{x'}|| \leq d(X, X')$$

For discrete metric spaces we represent them as graphs and permutations replace translations. The rest of the machinery (wavelets, learnable filters) translates to graphs in the form of graph neural networks (GNNs).

Part II

Optimization and Geometry in DL

The main goals of the part of the class:

1. Develop tools to analyze gradient descent on generic high-dimensional learning problems.
    - ▷ We will study convex, non-convex, and stochastic optimization.
2. Develop functional space view of neural networks.
    - ▷ We will discuss Reproducing Kernel Hilbert Spaces (RKHS) and measure spaces.

Lecture 6

A Primer on Convex Optimization

From this lecture we are going to start a new part of the class: Optimization and Geometry in deep learning. Basically we are going to try to develop two main frameworks to two main ideas:

- ▷ Developing tools to analyse gradient descent on generic high-dimensional learning problems (convex, non-convex, stochastic optimization)
- ▷ Functional space view of neural network (RKHS: reproducing kernel Hilbert space and Measure space)

In this lecture, we will introduce convex optimization and discuss some of the classic results.

Recall in the first lecture that the empirical risk minimization, a prototypical supervised learning problem, is of the form

$$\min_{x \in \mathcal{X}} f(x), \tag{6.1}$$

where f denotes a generic loss. Here $\mathcal{X}$ high-dimensional. For simplicity, we suppose $\mathcal{X} \in \mathbb{R}^d$, where $d \gg 1$ and x are parameters being optimized
The worst complexity case of Equation (6.1), under mild assumption that f is Lipschitz, is exponential in the dimension. This is because if we want to find the minimum of the function and we know very little about of the function, the only thing we can do is to "grid" the domain. By "gridding the domain" we mean to evaluate all the points in the domain and find the minimum of these points. This does not always work. As a result, we want to overcome this curse of dimensionality. We need to utilize the assumptions on the function that we can use to break the curse of dimensionality.

6.1 Convexity

The first fundamental source of structure comes from convexity.

Definition 6.1.1 (Convexity)**.** A function $f : \mathbb{R}^d \to \mathbb{R}$ is **convex** if for all $x, y \in \mathbb{R}^d$ and all $t \in [0, 1]$,

$$f(tx + (1-t)y) \leq tf(x) + (1-t)(y).$$

An important property of convexity is that if f s differentiable, then

$$\frac{f(tx + (1-t)y) - f(y)}{t} \leq f(x) - f(y),$$

and as $t \to 0$, we have

$$\langle \nabla f(y), x - y \rangle \leq f(x) - f(y) \tag{6.2}$$

This means that f stays above its linear approximation at y

$$f_l(x) = f(y) + \langle \nabla f(y), x - y \rangle.$$

Remark 6.1.1. We can extend the notion of Equation (6.2) to non-differentiable functions using sub-gradients.

6.2 Gradient Descent for Smooth Convex Functions

First, we give a definition of smooth functions.

Definition 6.2.1 (Smooth Function)**.** A continuously differentiable function f is β**-smooth** if ∇f is β-Lipschitz, i.e.

$$\|\nabla f(x) - \nabla f(y)\| \leq \beta \|x - y\|.$$

In particular, if f is twice differentiable, then

$$\nabla^2 f \preceq \beta \cdot Id,$$

meaning that all singular values of $\nabla^2 f$ are below β.

Definition 6.2.2 (Gradient Descent Algorithm)**.** Let η be the step size. Then, the **gradient descent** is defined by

$$x_{t+1} = x_t - \eta \nabla f(x_t).$$

where $t = 0, 1, \ldots$.

We are interested in the convergence and the speed of convergence of the gradient descent algorithm. To analyze these, we first find out what happens as $\eta \to 0$. We introduce the following approach.

We claim that $x_t = X(t \cdot \eta)$, where η is the time variable and $X(s)$ is a smooth, continuous curve with $s \in \mathbb{R}^+$.

Figure 6.1: Gradient Descent. As $\eta \to 0$, we postulate a smooth curve $X(s)$ (right).

Let $s = \eta \cdot t$. We can see the gradient descent as a time-discretization of an underlying process, where the sample rate is given by the step size. Now, we have an analogy between $X(s)$ and x_t. We can rewrite the gradient descent algorithm as as

$$\frac{X(s+\eta) - X(s)}{\eta} = -\nabla f(X(s)).$$

As $\eta \to 0$, we have $\dot{X}(s) = -\nabla f(X(s))$, the g*radient flow* ordinary differential equation (ODE). We see that the gradient descent algorithm is a discretization of the gradient flow ODE.

Remark 6.2.1. The discretization we use here is the forward-Euler (explicit) method, which yields the gradient descent algorithm. Another common discretization is the backward-Euler (implicit) method, which yields the proximal gradient algorithm.

Example 6.2.3. Given $f(x) = \frac{1}{2} x^\top H x$, H is positive definite. $H \succ \alpha \cdot Id (\alpha > 0)$.
It's easy to see $\min_x f(x) = 0$ at $x = 0$.
We have

$$\dot{X}(s) = -\nabla f(X(s)) = -H \cdot X(s)$$

By basic ODE integration we can see

$$X(s) = e^{-sH} X(0)$$

Given that H is positive definite bounded by α, we have:

$$\|X(s)\| \leq \|X(0)\| e^{-s \cdot \alpha}$$

Therefore, the gradient flow converges at linear rate.

Example 6.2.4. Given f is smooth and convex. Again, we are interested how fast does the gradient flow converge.
We can see gradient will converge at $x^* \in \arg\min_x f(x)$.
To see how fast does it converge, first let us define

$$\mathcal{L}(s) := s \cdot [f(X(s)) - f(x^*)] + \frac{1}{2}\|X(s) - x^*\|^2$$

And the derivative of $\mathcal{L}$ is:

$$\begin{aligned}\dot{\mathcal{L}}(s) :&= \frac{d}{ds}\mathcal{L}(s) \\ &= f(X(s)) - f(x^*) + s \cdot \langle \nabla f(X(s)), -\nabla f(X(s)) \rangle - \langle \nabla f(X(s)), X(s) - x^* \rangle \\ &= -s\|\nabla f(X(s))\|^2 + f(X(s)) - f(x^*) + \langle x^* - \nabla f(X(s)), X(s) \rangle\end{aligned}$$

Obviously, $-s\|\nabla f(X(s))\|^2$ and $f(X(s)) - f(x^*)$ is always non-positive, and we have $\langle x^* - \nabla f(X(s)), X(s) \rangle$ is also always non-positive by convexity.
Hence, all these three terms are always non-positive, we can see

$$s \cdot [f(X(s)) - f(x^*)] \leq \mathcal{L}(s) \leq \mathcal{L}(0) = \frac{1}{2}\|X(s) - x^*\|^2$$

.
Which indicates that

$$f(X(s)) - f(x^*) \leq \frac{\|X(0) - x^*\|^2}{2s}$$

Remark 6.2.2. $\mathcal{L}$ here is a "Lyapunov function".

Two take away so far :

- ▷ We identify a $\frac{1}{S}$ rate of convergence in the convex case.
- ▷ There is no curse of dimension, in fact, rate is dimension free!.

6.3 From Continuous to Discrete

The results we have seen so far are merely continuous time "intuition". Fortunately, we have almost the same results for the discrete condition.
Theorem 6.3.1 tells us that gradient descenting will also converge to optima at linear rate.

Theorem 6.3.1. *Assume f is convex and β-smooth in m^d.*
Then the gradient descent algorithm with step-size $\eta = \beta^{-1}$ satisfies $f(x_t) - f(x^*) \leq \frac{2\beta\|x_1 - x^*\|^2}{t^{-1}}$

Proof. First we need to prove 2 "structural" lemma to prove Theorem 6.3.1.

Lemma 6.3.2. *Assume f is β-smooth, then $\forall x, y$, we have*

$$|f(x) - f(y) - \langle \nabla f(y), x - y \rangle| \leq \frac{\beta}{2}\|x - y\|^2$$

To prove Lemma 6.3.2, first we use the fundamental theorem of calculus:

$$f(x) - f(y) = \int_0^1 \langle \nabla f(y + t(x - y), x - y \rangle dt$$

So

$$
\begin{aligned}
&|f(x) - f(y) - \langle \nabla f(y), x - y \rangle| = |\int_0^1 \langle \nabla f(y + t(x-y) - \nabla f(y), x - y \rangle dt| \\
&\leq \int_0^1 \|x - y\| \cdot \|\nabla f(y + t(x-y) - \nabla f(y)\| dt \text{ (by Cauchy Schwarz inequality)} \\
&\leq \int_0^1 \|x - y\|^2 \cdot t \cdot \beta dt \text{ (by } \beta\text{-smooth)} \\
&\leq \frac{\beta}{2} \|x - y\|^2
\end{aligned}
$$

Combine lemma 1 with convexity of f, we get:

$$0 \leq f(y) - f(x) - \langle \nabla f(x), y - x \rangle \leq \frac{\beta}{2} \|x - y\|^2$$

Assigning $y = x - \beta^{-1} \nabla f(x)$ (intuition: $(x = x_t, y = x_t + 1)$ in gradient descent).
Then we can easily see that

$$f(x - \beta^{-1} \nabla f(x)) - f(x) + \langle \nabla f(x), \beta^{-1} \nabla f(x) \rangle \leq \frac{\beta}{2} \cdot \beta^{-2} \|\nabla f(x)\|^2$$

After simplification we get:

$$f(x - \beta^{-1} \nabla f(x)) - f(x) + \beta^{-1} \|\nabla f(x)\|^2 \leq \frac{\beta^{-1}}{2} \|\nabla f(x)\|^2$$

Hence

$$f(x - \beta^{-1} \nabla f(x)) - f(x) \leq -\frac{\beta^{-1}}{2} \|\nabla f(x)\|^2$$

Since $x_{t+1} = x - \beta^{-1} \nabla f(x)$:

$$f(x_{t+1}) - f(x_t) \leq -\frac{\beta^{-1}}{2} \|\nabla f(x)\|^2$$

The result we achieved so far is called "sharpness": the progress in the loss is controlled by the site/norm of the gradient.

Lemma 6.3.3. *Assume f is convex and β-smooth. Then*

$$f(x) - f(y) \leq \langle \nabla f(x), x - y \rangle - \frac{1}{2\beta} \|\nabla f(x) - \nabla f(y)\|^2$$

The result from Lemma 6.3.3 is also called co-coercivity.
We are not going to prove Lemma 6.3.3 in this class. The proof can be found as lemma 3.5 in Bubeck at page 39.
Now we can prove Theorem 6.3.1, first using the sharpness result:

$$f(x_{s+1}) - f(x_s) \leq -\frac{1}{2\beta} \|\nabla f(x_s)\|^2, s \in \mathbb{N}$$

Denote $\delta_s = f(x_s) - f(x^*)$ (optimality gap), we have

$$\delta_{s+1} \leq \delta_s - \frac{1}{2\beta} \|\nabla f(x_s)\|^2 \tag{6.3}$$

Using convexity of f, by definition we got:

$$\delta_s \leq \langle \nabla f(x_s), x_s - x^* \rangle$$

And from the property of dot product, we can see that:

$$\delta_s \leq \langle \nabla f(x_s), x_s - x^* \rangle \leq \|\nabla f(x_s)\| \cdot \|x_s - x^*\| \tag{6.4}$$

Assume for now that $\|x_s - x^*\|$ is non-increasing (we are going to prove it later), then we can got the below inequality from Equation (6.4):

$$\|\nabla f(x_s)\| \geq \frac{\delta_s}{\|x_s - x^*\|} \geq \frac{\delta_s}{\|x_1 - x^*\|}$$

From Equation (6.3), this implies:

$$\delta_{s+1} \leq \delta_s - \frac{\delta_s^2}{2\beta\|x_1 - x^*\|^2} \tag{6.5}$$

Denote $w = \frac{1}{2\beta\|x_1 - x^*\|^2}$, we can rewrite 6.5 as

$$w \cdot \delta_s^2 + \delta_{s+1} \leq \delta_s$$

And then multiply by $\frac{1}{\delta_s \delta_{s+1}}$, we have:

$$w \cdot \frac{\delta_s}{\delta_{s+1}} + \frac{1}{\delta_s} \leq \frac{1}{\delta_{s+1}}$$

From $\delta_s \geq \delta_{s+1}$ (which is obvious from Equation (6.5)) we now have

$$\frac{1}{\delta_{s+1}} - \frac{1}{\delta_s} \geq w$$

Thus we can see:

$$\sum_{s=0}^{t} \left(\frac{1}{\delta_{s+1}} - \frac{1}{\delta_s} \right) \geq (t-1) \cdot w$$

which indicates $\frac{1}{\delta_t} \geq (t-1) \cdot w$, hence

$$\delta_t \leq \frac{1}{(t-1) \cdot w} = \frac{2\beta\|x_1 - x^*\|^2}{t-1} \tag{6.6}$$

We can see that Equation (6.6) is exactly the same as Theorem 6.3.1.
So we have proved the theorem with the assumption that $\|x_s - x^*\|$ is non-increasing. Now it remains to show that's true.
Recall Lemma 6.3.3:

$$f(x) - f(y) \leq \langle \nabla f(x), x - y \rangle - \frac{1}{2\beta}\|\nabla f(x) - \nabla f(y)\|^2$$

Apply twice by swapping x and y, now we have:

$$\frac{1}{2\beta}\|\nabla f(x) - \nabla f(y)\|^2 \leq \langle \nabla f(x), x - y \rangle + f(y) - f(x)$$

$$\frac{1}{2\beta}\|\nabla f(x) - \nabla f(y)\|^2 \leq \langle \nabla f(y), y - x \rangle + f(x) - f(y)$$

By summing them up we get:

$$\frac{1}{\beta}\|\nabla f(x) - \nabla f(y)\|^2 \leq \langle \nabla f(x) - \nabla f(y), x - y \rangle$$

Then

$$\begin{aligned}\|x_{s+1} - x^*\|^2 &= \left\|x_s - x^* - \beta^{-1}\nabla f(x_s)\right\|^2 \\ &= \|x_s - x^*\|^2 - \frac{2}{\beta}\langle \nabla f(x_s), \nabla(x^*), x_s - x^*\rangle + \frac{1}{\beta^2}\|\nabla f(x_s) - \nabla f(x^*)\|^2 \\ &\leq \|x_s - x^*\|^2 - \frac{2}{\beta}\|f(x_s\|^2 + \frac{1}{\beta^2}\|\nabla f(x_s)\|^2 \\ &\leq \|x_s - x_x^*\|^2\end{aligned}$$

Therefore the assumption $\|x_s - x^*\|$ is non-increasing must be true.

□

Remark 6.3.1. Here are takeaways so far:

- ▷ The rate we proved in this lecture so far is not optimal in the class of convex, smooth functions, we will learn more in the next lecture.
- ▷ Rate is dimension-free!! To get error ϵ, we need $\Theta(\frac{\beta}{\epsilon})$ calls to "the oracle" (the oracle complexity).
- ▷ Gradient flow "intuition" is correct.
- ▷ Discrete time is harder!
- ▷ General theme: Extending continuous time behavior to discrete time is <u>not</u> straight-forward. (It's a big topic in PDE/ODE: discretization of "integrators")

6.4 Strong Convexity

To achieve better result with rate of convergence, we need something beyond smoothness. We are going to discuss strong convexity here.

Definition 6.4.1 (Strong Convexity)**.** A continuously differentiable function f is α strongly convex if

$$f(x) - f(y) \leq \langle \nabla f(x), x - y\rangle - \frac{\alpha}{2}\|x - y\|^2$$

In particular, if f is twice differentiable, then

$$\nabla^2 f(x) \preceq \alpha I$$

α is the "curvature" of this family of functions.
At any x, consider two quadratic forms:

$$q_x^-(y) = f(x) + \langle \nabla f(x), y - x\rangle + \frac{\alpha}{2}\|x - y\|^2$$

$$q_x^+(y) = f(x) + \langle \nabla f(x), y - x\rangle + \frac{\beta}{2}\|x - y\|^2$$

And if f is α-strong convex and β-smooth, we have

$$q_x^-(y) \leq f(y) \leq q_x^+(y)$$

Denote $\kappa := \frac{\beta}{\alpha}(\geq 1)$ as the condition number of f.
We need another structural lemma to show better result with rate of convergence:

Lemma 6.4.2. *f is β-smooth and α-strong convex, then*

$$\langle \nabla f(x) - \nabla f(y), x - y\rangle \geq \frac{\alpha\beta}{\alpha + \beta}\|x - y\|^2 + \frac{1}{\alpha + \beta}\|\nabla f(x) - \nabla f(y)\|^2$$

with $\alpha = 0$, this gives us the same result of Lemma 6.3.3
Again we are not going to proof Lemma 6.4.2 in this lecture, here is some hint for the proof:
If f is β-smooth and α-strong convex, then $\phi(x) = f(x) - \frac{\alpha}{2}\|x\|^2$ is convex and $(\beta - \alpha)$-smooth. Then we can apply Lemma 6.3.3 to ϕ to prove this lemma.
By using Lemma 6.4.2, we can prove the following theorem:

Theorem 6.4.3. *if f is β-smooth and α-strong convex, then gradient descent with $\eta = \frac{2}{\alpha+\beta}$ gives*

$$f(x_t) - f(x^*) \leq \frac{\beta}{2} exp\left(-\frac{4k}{\kappa + 1}\right)\|x_1 - x^*\|^2$$

Remark 6.4.1. Without assumption of strong convexity, to reach error ϵ with gradient descent, we need $\Theta(1/\epsilon)$ iterations.
With strong convexity, we need $\Theta(\kappa \cdot \log(1/\epsilon))$ oracle complexity to achieve the same error rate, which is far better than the previous result.

Lecture 7

Convex Optimization and Stochastic Optimization

Recall

From convex optimization using a single algorithm, Gradient Descent (GD), we obtained dimension-free convergence rates. This shows that the geometry of the function plays an essential role: the rate is $O(\frac{1}{\epsilon})$ in no curvature case; $O(\log \frac{1}{\epsilon})$ in curvature case (strong convexity). Note that both results use the same algorithm.
In today's class, we will investigate:

1. Is GD optimal in such classes? If not, how to upgrade it?

2. Stochastic Optimization

7.1 Oracle Complexity Lower Bounds

We want to understand whether GD is optimal amongst a reasonable class of algorithms. Let point X_{t+1} satisfies

$$x_{t+1} \in \text{span}(\nabla f(x_1), \cdots, \nabla f(x_t)) \tag{$\star$}$$

Theorem 7.1.1 (Smooth Case,[Nemirovsky; Nesterov, 2013]). *There exists a convex β-smooth function such that any black box algorithm satisfying (11.1), we have* $\min_{s\le t} f(x_s) - f(x^*) \ge \frac{3}{32}\beta\frac{||x_1-x^*||^2}{(t+1)^2}$

Remarks: There is a gap between this lower bound ($\frac{1}{t^2}$) and the rate of GD ($\frac{1}{t}$). The "worst-case" example is in fact a quadratic function (Dirichlet energy)

Theorem 7.1.2 (Smooth + Strongly Convex Case,[Nemirovsky; Nesterov, 2013]). *There exists a β-smooth and α-strongly convex function f such that for any t, $f(x_t) - f(x^*) \ge \frac{\alpha}{2}(\frac{\sqrt{K}-1}{\sqrt{K}+1})^{2(t-1)}||x_1 - x^*||^2$, where $K = \frac{\beta}{\alpha}$ is the condition number of f.*

Remarks: There is also a gap between this lower bound ($\frac{\sqrt{K}-1}{\sqrt{K}+1} \approx \exp(\frac{-4(t-1)}{\sqrt{K}})$ and the rate of GD ($\exp(\frac{-4(t-1)}{K})$).

Question: How to close these gaps?

7.1.1 Quadratic case

Let's first consider the quadratic case: $f(x) = \frac{1}{2}(x - x^*)^\top H(x - x^*)$, H is symmetric and $\alpha \leq H \leq \beta$ (f is β-smooth and α-strongly convex). GD update rule becomes:

$$\begin{aligned} x_(t+1) &= x_t - \eta \nabla f(x_t) \\ &= x_t - \eta H(x_t - x^*) \\ &= [-\eta H]x_t + \eta H x^* \\ &= [-\eta H]x_t + \eta b \end{aligned}$$

Suppose that $x_0 = 0, x_{t+1} = \eta(\sum_{k \leq t}[-\eta H]^k)b, \alpha \leq H \leq \beta$

$$\implies (1 - \eta\beta) \leq -\eta H \leq (1 - \eta\alpha)$$

If $\eta \leq \beta^{-1}$, then $|| - \eta H|| \leq 1$,

$$\implies (\eta H)^{-1} = [-(-\eta H)]^{-1} = \sum_{k \geq 0}[-\eta H]^k$$

Let $H_t := \sum_{k=0}^{t}[-\eta H]^k$, then

$$\begin{aligned} x_{t+1} &= \eta H_t b = \eta[(\eta H)^{-1} + H_t - (\eta H)^{-1}]b \\ &= x^* + (H_t - (\eta H)^{-1})\eta b \end{aligned}$$

So

$$\begin{aligned} ||x_{t+1} - x^*|| &\leq ||H_t - (\eta H)^{-1}|| \\ &= O(|| - \eta H)^t||) \\ &= O((1 - \frac{\alpha}{\beta})^t) \\ &= O((1 - K^{-1})^t) \end{aligned}$$

Can we do better than GD? Note the error comes from approximating the inverse matrix $A = (\eta H)^{-1}$ with a finite power series $\sum_{k=0}^{t}[-\eta H]^k$, which is a polynomial of degree t in A. Let q_t be a polynomial of degree t:

$$\min_{q_t} ||A^{-1} - q_t(A)|| \simeq \min_{q_t} || - Aq_t(A)||$$

, by multiplying A. Denote $p_t(A) = -Aq_t(A)$. Note that p_t is a polynomial of degree $\leq t+1, p_t(0) = 1, p_t(A)$ commutes with A, thus λ is an eigenvalue of $A \iff p_t(\lambda) = 1 - \lambda q_t(\lambda)$ is an eigenvalue of $p_t(A)$. Since eigenvalues of A are in $[\alpha, \beta]$, the goal is thus:

$$\min_{x \in (\alpha, \beta)} |p_t(x)| \, , s.t \;\; p_t(0) = 1$$

Lemma 7.1.3 (Chebyshev Polynomials)**.** *There is a polynomial p_t of degree $t = O(\sqrt{(\beta/\alpha)\log(\frac{1}{\epsilon})}$ such that $p_t(0) = 1$ and $|p_t(x)| \leq \epsilon \; \forall x \in (\alpha, \beta)$*

So q_t are Chebyshev Polynomials, which can be computed recursively from q_{t-1}, q_{t-2}. Can we extend such "Chebyshev" scheme beyond quadratic functions?

7.1.2 Nesterov Accelerated Gradient Descent (NAGD)

[left=]align x$_{k+1} = y_k - \eta \nabla f(y_k)$
$y_{k+1} = x_{k+1} + \frac{k}{k+3}(x_{k+1} - x_k)$

Theorem 7.1.4 ([Nesterov, 1983])**.** *Let $\eta = \beta^{-1}$, for f is convex and β-smooth, $f(x_t) - f(x^*) \leq \frac{2\beta||x_0 - x^*||^2}{t^2}$*

A slight modification of the scheme leads to:

$$f(x_t) - f(x^*) \leq \beta(1 - \sqrt{\frac{\alpha}{\beta}})^t ||x_0 - x^*||^2$$

And we obtain a better convergence rate $(1 - \sqrt{K^{-1}})^t$ in NAGD than vanilla GD $(1 - K^{-1})^t$. NAGD is related to Momentum, Heavy-ball methods. Can we obtain a continuous time interpretation of NAGD $(\eta \to 0)$?
Recall for Gradient Descent (GD): $x_t = x_t - \eta \nabla f(x_t)$, its Gradient Flow (GF) is given by: $x(t) = -\nabla f(x(t))$

Theorem 7.1.5 ([Su et al., 2014]). *$\ddot{x} + \frac{3}{t}\dot{x} + \nabla f(x(t)) = 0$. This second-order ODE has unique solution $x(t)$ in $C^2((0, \infty) \times \mathbb{R}^d)$. Moreover, as $y \to 0$, Nesterov's scheme converges to this ODE:*

$$\forall T \geq 0, \lim_{y \to 0} \sup_{k \leq T/\sqrt{\eta}} ||x_k - x(k\sqrt{\eta})|| = 0$$

Simple consequence: we show easily that $\frac{1}{t^2}$ rate, $f(x(t)) - f(x^*) = .0(\frac{1}{t^2})$. Consider the following Lyapunov function:

$$\mathcal{L}(t) = t^2[f(x(t)) - f(x^*)] + 2||x + \frac{t}{2}\dot{x} - x^*||^2$$

Fact: $\frac{\partial}{\partial t}\mathcal{L}(t) \leq 0$ (apply definition of convexity of f).

$$t^2(f(x(t)) - f(x^*)) \leq \mathcal{L}(t) \leq \mathcal{L}(0) \implies f(x(t)) - f(x^*) \leq \frac{\mathcal{L}(0)}{t^2}$$

Remarks:

1. Many different interpretations of acceleration: symplectic methods, geometric descent, etc.
2. In Machine Learning (ML), f is not always convex (and never convex in Deep Learning!). On the other hand, f in ML has a very special form: f is an average, as $f(x)$ is the empirical risk $Ł(\theta) = \mathbb{E}_{(x,y)}[\ell(x; \theta), y)]$. How to leverage such structure?

7.2 Stochastic Approximation

As before, we consider minimisation of f defined in $\mathbb{R}^d$ with respect to θ. However, we no longer have access to $\nabla f(\theta)$ directly. Instead, we only have access to unbiased estimates of $\nabla f(\theta)$. For examples:

$$\begin{aligned} f_n &: \text{loss at a single datapoint} \\ f_n(\theta) &= \ell(\phi(x_n; \theta), y_n) \\ f(\theta) &= \mathbb{E}_{(x,y)\sim V}[\ell(\phi(x; \theta), y)], \text{where } E \text{ is either the empirical loss or the population loss} \end{aligned}$$

7.2.1 Stochastic Approximation

General Setting: Find the zeroes of a function, $h : \mathbb{R}^d \to \mathbb{R}^d$ from random observations at certain points. If $h = \nabla f$, we recover stochastic approximation.
[Robbins Munro Scheme (SGD)]

$$\theta_n = \theta_{n-1} - \gamma_n(h(\theta_n) + \epsilon_n)$$

Questions: What are the conditions for convergence? And what is the convergence rate?

- ▷ On γ_n learning rate
- ▷ On ϵ_n noise term
- ▷ on the geometry of h

Motivating Example: Estimating the mean of a sample recursively. Start from $\theta_0 = 0$, then we get data $x_n \in \mathbb{R}^d$, and consider $\theta_n = \theta_{n-1} - \gamma_n(\theta_{n-1} - x_n) = \gamma_n x_n + (1-\gamma_n)\theta_{n-1}$.

- ▷ If we set $\gamma_n = \frac{1}{n}$, then $\theta_n = \frac{1}{n}\sum_{k=1}^n x_k$;
- ▷ If we set $\gamma_n = \frac{2}{n+1}$, then $\theta_n = \frac{2}{n(n+1)}\sum_{k=1}^n k x_k$;
- ▷ If x_n are i.i.d with $\mathbb{E}(x_n) = x, \mathbb{E}[||x_n - x||^2] = \sigma^2$, then:

$$\begin{aligned}\theta_n - x &= (1-\gamma_n)(\theta_{n-1} - x) + \gamma_n(x_n - x)\\ &= \underbrace{\prod_{k=1}^n (1-\gamma_k)(\theta_0 - x)}_{\text{deterministic}} + \underbrace{\sum_{i=1}^n \prod_{k=i+1}^n (1-\gamma_k)\gamma_i(x_i - x)}_{\text{sum of i.i.d variables}}\end{aligned}$$

$$\implies \mathbb{E}[||\theta_n - x||^2] = \underbrace{\prod_{k=1}^n (1-\gamma_k)^2 ||\theta_0 - x||^2}_{\text{initial condition}} + \underbrace{\sigma^2 \sum_{i=1}^n \gamma_i^2 \prod_{k=i+1}^n (1-\gamma_k)^2}_{\text{robustness to noise}}$$

So the error has two contributions:

- ◇ Initial conditions: $\prod_{k=1}^n (1-\gamma_k)^2 \to 0$ as $n \to \infty$. We want to forget initial conditions with an aggressive learning rate. If $\gamma_n \to 0, \log\prod_{k=1}^n (1-\gamma_k) \approx -\sum_{k=1}^n \gamma_k$, so $\sum_{k=1}^n \gamma_k$ should diverge as $n \to \infty$.
- ◇ Robustness to noise in the gradients: $\sum_{i=1}^n \gamma_i^2 \prod_{k=i+1}^n (1-\gamma_k)^2 \to 0$ as $n \to \infty$. We want small noise.

Consider the following example (for initial condition convergence): $\gamma_n = Cn^{-\alpha}$.

- ◇ $\alpha = 1 : \sum_{k=1}^n \gamma_k \approx \log n + C + O(n^{-1})$ $\surd$
- ◇ $\alpha > 1 : \sum_{k=1}^n \gamma_k = C + O(n^{1-\alpha})$ $\times$
- ◇ $\alpha \in (0,1) : \sum_{k=1}^n \gamma_k = Cn^{1-\alpha} + O(1)$ $\surd$

Lecture 8

Non-Convex Optimization

In the previous lecture, we showed that the Nesterov acceleration achieves the lower bound on the class of convex functions (with and without curvature). We also discussed stochastic optimization and showed that tunable parameter (learning rate schedule) controlled a "bias-variance" tradeoff, where the forgetting of initial condition is the bias and noise in the gradient corresponds to variance.

In this lecture, we continue to investigate SGD and move to the study of GD and SGD on non-convex objectives.

Estimating the Mean of a Sample: Continued

At the end of the last lecture, we estimated the mean of the distribution where samples are drawn from using SGD:

$$\mathbb{E}\|\theta_n - x\|^2 = \underbrace{\prod_{k\le n}(1-\gamma_k)^2\|\boldsymbol{\theta_0} - x\|^2}_{\text{initial condition}} + \underbrace{\sigma^2\sum_{i\le n}\gamma_i^2\prod_{k>i}(1-\gamma_k)^2}_{\text{robustness to noise}}.$$

In order to forget the initial conditions, the learning rate has to be sufficiently strong. That is, $\sum_{k\le n}\gamma_k$ should diverge with n. An example of such a learning rate is when $\gamma_n = C\cdot n^{-\alpha}$, where $\alpha\in[0,1]$.

We also need to ensure that the noise does not blow up. Assume that γ_n is non-increasing and $\gamma_n \le \frac{1}{\mu}$, where $\mu < 2$. We can approximate

$$\sum_{i\le n}\gamma_i^2\prod_{k>i}(1-\gamma_k)^2 \approx \sum_{i\le n}\gamma_i^2\prod_{k>i}(1-2\gamma_k)$$

since γ_k is small, the dominant part is the linear part. We can also find an upper bound for the RHS, i.e.

$$\sum_{i=1}^{n}\gamma_i^2\prod_{k>i}(1-2\gamma_k) \le \sum_{i=1}^{n}\gamma_i^2\prod_{k>i}(1-\mu\gamma_k)$$

Now, we break the sum on the RHS into two parts. Let $m \le n$. Then

$$\begin{aligned}
\sum_{i=1}^{n}\gamma_i^2\prod_{k>i}(1-\mu\gamma_k) &= \sum_{i=1}^{m}\gamma_i^2\prod_{k>i}(1-\mu\gamma_k) + \sum_{i=m+1}^{n}\gamma_i^2\prod_{k>i}(1-\mu\gamma_k)\\
&\le \prod_{i=m+1}^{n}(1-\mu\gamma_i)\sum_{k=1}^{n}\gamma_i^2 + \gamma_m\sum_{i=m+1}^{n}\gamma_i\prod_{k>i}(1-\mu\gamma_i)\\
&\le \underbrace{\exp\left(-\mu\sum_{i=m+1}^{n}\gamma_i\right)\sum_{k=1}^{n}\gamma_k^2}_{*} + \gamma_m\sum_{i=m+1}^{n}\gamma_i\prod_{k>i}(1-\mu\gamma_k) \qquad (* : \text{by concavity})
\end{aligned}$$

$$
\begin{aligned}
&\leq \exp\left(-\mu \sum_{i=m+1}^{n} \gamma_i\right) \sum_{k=1}^{n} \gamma_k^2 + \frac{\gamma_m}{\mu} \sum_{i=m+1}^{n} \left[\prod_{k>i}(1-\mu\gamma_k) - \prod_{k>i-1}(1-\mu\gamma_k)\right] \\
&\leq \exp\left(-\mu \sum_{i=m+1}^{n} \gamma_i\right) \sum_{k=1}^{n} \gamma_k^2 + \frac{\gamma_m}{\mu} \left[1 - \prod_{k=m+1}^{n}(1-\mu\gamma_k)\right] \\
&\leq \exp\left(-\mu \sum_{i=m+1}^{n} \gamma_i\right) \sum_{k=1}^{n} \gamma_k^2 + \frac{\gamma_m}{\mu}
\end{aligned}
$$

To make the noise term goes to 0, we require γ_n to go to 0 as $n \to \infty$. In particular, we have $\gamma_n = \frac{c}{n}$. Then, the noise term converges in $O\left(\frac{1}{n}\right)$ and the forgetting of the initial condition converges in $O\left(\frac{1}{n^{2c}}\right)$.

8.1 General Behavior of SGD

Recall that the vanilla SGD (Robbins-Munro) is given by

$$\theta_{n+1} = \theta_n - \gamma_n g(\theta_n, \xi_n),$$

where $g(\theta_n, \xi_n)$ is an estimate of $\nabla f(\theta_n)$. The canonical choice of the learning rate is $\gamma_n = \frac{c}{n}$, as explained above. We are interested in the asymptotic behavior of SGD when f is a strongly convex function.

Assume f is such that

$$\theta^* = \arg\min_{\theta} f(\theta),$$

where

$$f(\theta) \approx (\theta - \theta^*)^\top \underbrace{\nabla^2 f(\theta^*)}_{A} (\theta - \theta^*)$$

for all θ close to θ^*. Since f is strongly convex, A is positive definite. The asymptotic behavior of the SGD is shown in the following theorem.

Theorem 8.1.1 (Asymptotic Normality; [Fabian, 1968])**.** *θ_n is asymptotically Gaussian distributed as $n \to \infty$. That is,*

$$\mathbb{E}[\theta_n] \to \theta^*$$

and

$$\mathbb{E}[(\theta_n - \theta^*)(\theta_n - \theta^*)^\top] \approx n^{-2CA}(\theta_n - \theta^*)(\theta_n - \theta^*)^\top + \left(\frac{1}{n} C^2 \frac{1}{2CA - I} \Sigma\right),$$

where Σ is the covariance (noise) of the stochastic gradients.

For $(\theta_n - \theta^*)(\theta_n - \theta^*)^\top$ to go to 0, we need $2C\lambda_{\min}(A) \geq 1$. If C is too small, then we do not have convergence. On the other hand, if C is too large, then we have a large variance. We see that the vanilla SGD, even for very simple problems, is brittle. That is, it is highly dependent on the conditioning of the problem.

To overcome such brittleness, a simple solution is by taking average. Consider the averaged iterate

$$\overline{\theta}_n = \frac{1}{n} \sum_{k=1}^{n} \theta_k,$$

where θ_k are the iterates from vanilla SGD. we can compute $\overline{\theta}_n$ recursively as

$$\overline{\theta}_n = \left(1 - \frac{1}{n}\right) \overline{\theta}_{n-1} + \frac{1}{n} \theta_n.$$

Theorem 8.1.2. *Suppose $\theta_n \to \theta^*$ with some rate $\|\theta_n - \theta^*\| \le \alpha_n$. Then $\overline{\theta}_n \to \theta^*$ with rate*

$$\left\|\overline{\theta}_n - \theta^*\right\| \le \overline{\alpha}_n \le \frac{1}{n}\sum_{k=1}^{n} \alpha_n$$

whenever the upper bound is well-defined.

Proof.

$$\left\|\overline{\theta}_n - \theta^*\right\| = \left\|\frac{1}{n}\sum_{k=1}^{n}(\theta_k - \theta^*)\right\| \le \frac{1}{n}\sum_{k=1}^{n}\|\theta_k - \theta^*\| \le \frac{1}{n}\sum_{k=1}^{n}\alpha_k.$$

We know that $\alpha_n \to 0$. Consider N^* such that $\alpha_k \le \frac{\epsilon}{2}$ for all $k > N^*$. Then, for any $\epsilon > 0$, there exists $N = CN^*$, where $C = \frac{2\alpha_0}{\epsilon}$, such that

$$\frac{1}{N}\sum_{k=1}^{N}\alpha_k \le \epsilon.$$ □

Moreover, one can show that if $\sum_k \alpha_k < \infty$, then the rate $\overline{\alpha}_n$ is always $\frac{1}{n}$, independent of α_k. We lose convergence speed but we gain robustness.

In convex stochastic optimization, averaged SGD gives the best rate of convergence. There are several known minimized rates of convergence under varying assumptions. See Nemisovski '80s, Arganal '12, Bach '17, and Carmon '19. Typical results are as follows. For α-strongly convex cases, the rate of convergence is $O((\alpha)^{01})$. For non-strongly convex case, the rate of convergence is $O(n^{-\frac{1}{2}})$. Both cases are attained with averaged SGD. We see that in some convex smooth problems, $\gamma_n \sim n^{-1/2}$ gives robustness and adaptivity.

Now, we wonder what happens when we have constant step size. First, we set up the least squares

$$f(\theta) = \frac{1}{2}\mathbb{E}_{(X,Y)\sim\nu}|Y - \Phi(\theta; X)|^2$$

If we have SGD with constant rage γ, then

$$\theta_{n+1} = \theta_n - \gamma\nabla\Phi(\theta_n, x_n)(\Phi(\theta_n, x_n) - y_n),$$

where $(x_n, y_n) \sim \nu$ i.i.d. We see that $\theta_{n+1}|\theta_n \sim Q(\theta)$, which is independent of n. Note that a sample from Q is obtained as $\widetilde{\theta} = q((x, y)))$, where $(x, y) \sim \nu$ and

$$q(x, y) = \theta_n\gamma\nabla\Phi(\theta_n, x)(\Phi(\theta_n, x) - y).$$

We can see it as a homogeneous Markov chain since γ is independent of n. Therefore, under appropriate assumptions, this Markov chain converges to a stationary distribution Π_γ not to a minimizer of f. To obtain the mean of this distribution, we average the Markov chain using the *Ergodic theorem*, which states that the temporal average equals to the expectation with respect to the stationary distribution. We see that this Markov chain can be analyzed in simple setups ($\Phi(\theta, x) = \langle\theta, \Psi(x)\rangle$).

8.2 From Convex to Non-convex Optimization

In the convex case, we saw that GD and SGD are extremely efficient with lots of positive results. We wonder what we can hope to achieve when f is no longer convex. We know that generic non-convex optimization is NP-hard.

First-Order Critical Point

If we run a gradient descent algorithm on a non-convex function, we want to find a minimizer efficiently.

Definition 8.2.1 (Critical Point)**.** An ϵ-approximate first-order **critical point** of f is θ such that $\|\nabla f(\theta)\| \leq \epsilon$.

Here we note that if f is convex, then the first-order critical point is a global minimum.

One immediate result is given as follows.

Theorem 8.2.2. *Let f be β-smooth, i.e. $\|\nabla f(x) - \nabla f(y)\| \leq \beta\|x - y\|$, and bounded below. Then the gradient descent with step size $\gamma = \beta^{-1}$ requires $\beta \cdot \frac{f(\theta_0) - f^*}{\epsilon^2}$ iterations, where $f(\theta) \geq f^*$ for all θ, to reach an epsilon first-order critical point.*

Proof. (Proof of a simplified argument in continuous time.)
Recalled that the gradient flow is given by $\dot{\theta}(t) = -\nabla f(\theta(t))$, where f is bounded below. Let $h(t) = f(\theta(t))$, then

$$\underbrace{f(\theta(T))}_{h(T)} - \underbrace{f(\theta(0))}_{h(0)} = \int_0^T h'(t)\,\mathrm{d}t = \int_0^T \langle \nabla f(\theta(t)) \rangle\,\mathrm{d}t = -\int_0^T \|\nabla f(\theta(t))\|^2\,\mathrm{d}t.$$

This implies that

$$\int_0^T \|\nabla f(\theta(t))\|^2\,\mathrm{d}t \leq f(\theta(0)) - f^* \leq K,$$

where K is some bound. Therefore, $\|\nabla f(\theta(t))\|^2 = O(t^{-1})$ and $\|\nabla f(\theta(t))\| = O(t^{-1/2})$. □

Note that there is no curse of dimensionality here and so far this concerns the first-order stationary points.

Stochastic Non-convex Case

The stochastic non-convex case has essentially the same behavior.

Theorem 8.2.3. *Consider a stochastic gradient descent with decreasing step size γ_k, $\sum_k \gamma = \infty$, and $\sum_k \gamma_k^2 < \infty$. Let $A_n := \sum_{k=1}^n \gamma_k$. Then*

$$\mathbb{E}\left[\frac{1}{A_n}\sum_{k=1}^n \gamma_k \|\nabla f(\theta_k)\|^2\right] \to 0$$

as $n \to 0$.

Classification of Critical Points

So far, we studied how GD and SGD can find stationary points in non-convex objectives. Let $\mathcal{G} = \{\theta; f(\theta) = \min_{\theta'} f(\theta')\}$ be the global minimizer and $\mathcal{F} = \{\theta; \nabla f(\theta) = 0\}$ be the critical points. In the convex case, $\mathcal{G} = \mathcal{F}$. But in general, we only have $\mathcal{G} \subseteq \mathcal{F}$. We would like to find a set of solutions with an efficient algorithm that is much smaller than $\mathcal{F}$.

Definition 8.2.4 (Strict Saddle Point)**.** θ^* is a **strict saddle** if $\nabla f(\theta^*) = 0$ and $\lambda_{min}(\nabla^2 f(\theta^*)) < 0$.

We can now split $\mathcal{F}$ in to two disjoint sets $\mathcal{F}_M$ and $\mathcal{F}_S$, where $\mathcal{F}_M = \left\{\theta : \nabla f(\theta) = 0, \lambda_{\min}(\nabla^2 f(\theta)) \geq 0\right\}$ and $\mathbb{F}_S$ is the set of strict saddle points. We want to see if $\mathcal{F}_M$ is the set of local minimum M of f. Here, $\theta \in M$ if there exist $\epsilon < 0$ such that for all $x \in B(\theta; \epsilon)$, we have $f(x) \geq f(\theta)$. We know that $M \subseteq \mathcal{F}_m$. However, $M \not\supseteq \mathcal{F}_M$. To see this, note that the definitions of $\mathcal{F}$'s only look at the second-order approximations of the function whereas where the definition of the local minimizer does not care about the order of approximation. We have $G \subseteq M \subseteq \mathcal{F}_M \subseteq \mathcal{F}$.

Stable Critical Points

Now that we know every critical point is an equilibrium of GD . We want to find those that are stable. The intuition is that strict saddles $\mathcal{F}_S$ are unstable.

Example 8.2.5. Consider the quadratic case $f(\theta) = \frac{1}{2}\theta^* H\theta$, where $H = \operatorname{diag}(\lambda_1, \ldots, \lambda_d)$ with $\lambda_1, \ldots, \lambda_s > 0$ and $\lambda_{s+1}, \ldots, \lambda_d < 0$. Then, there is a single critical point $\theta^* = 0$. It is a strict saddle since $\nabla^2 f(\theta^*) = H$. If we initialize gradient descent at θ_0, we have

$$\theta_{k+1} = \theta_k - \gamma \nabla f(\theta_k) = \sum_{i=1}^{d} (1 - \gamma\lambda_i)^{k+1} \theta_{0,i} e_i,$$

where $\{e_1, \ldots, ed\}$ is the canonical basis of $\mathbb{R}^d$.
Assume $\gamma < \frac{1}{\lambda_{\max}}$. Then $\|\theta_k\| \to 0$ if $\theta_0 \in \operatorname{span}\{e_1, \ldots, e_s\} = \mathcal{S}$. We note that $\dim \mathcal{S} = s < d$. If θ_0 is initialized from some distribution with continuous density, then gradient descent escapes the saddle point with probability 1.

Lecture 9

Towards Tractable Non-convex Optimization: Overparametrized Neural Networks

Previously, we saw that in stochastic optimization, vanilla SGD requires a lot of care at adjusting the learning rate (balance between forgetting initial conditions and control of the noise), where averaging comes to the rescue. For non-convex optimization, there are dimension-free rastes of convergence to find critical points, and a hint that we could replace critical points by local minima (illustrated by a simple quadratic example). In this lecture, we will discuss more on saddle points and the influence of noise. We will then investigate Neural Network optimization in the overparametrization regime.

9.1 Stable Manifolds

Assume that f is twice differentiable: a discrete-time optimization scheme is a mapping $\phi : \mathbb{R}^d \to \mathbb{R}^d$. Take Gradient Descent as an example: $\varphi(\theta) = \theta - \eta \nabla f(\theta)$.
After k iterations from the initial θ_0:

$$\theta_k = \underbrace{\varphi \cdots \varphi}_{\text{k}} \theta_0 = \varphi^k(\theta_0)$$

Definition 9.1.1 (Global Stable Set)**.** Given $X^* = \{\text{strict saddles of } f\}$, the global stable set of X^* is $S_\varphi = \{\theta_0; \lim_{k\to\infty} \varphi^k \theta_0 \in X^*\}$. Namely, all initial conditions that will drive φ towards X^*.

If θ^* is a critical point (strict saddle), the local attractive set of θ^* is given by $\text{span}\{e; e^\top \nabla^2 f(\theta^*) e > 0\}$, where e denotes the eigenvector. If this local attractive set has zero measure, then with probability 1 an initialization nearby θ^* will escape the neighborhood. The question is - how to go from local to global stability? The key idea is to use differential geometry.

Definition 9.1.2. Given a diffeomorphism $\varphi : \mathbb{R}^d \to \mathbb{R}^d$, the unstable fixed points of φ are $A^*_\varphi = \{\theta; \varphi(\theta) = \theta; ||D\varphi(\theta)|| > 1\}$, where D denotes the Jacobian.

Theorem 9.1.3 (Stable Manifold Theorem [Smale, 1967])**.** *Let φ be a diffeomorphism, $\varphi : X \to X$ and $\det(D\varphi(\theta)) \neq 0 \forall \theta$. Then the set of initial conditions that converge to A^*_φ has zero measure;*

$$\mu(\{\theta_0; \lim_{k\to\infty} \varphi^k(\theta_0) \in A^*_\varphi\}) = 0$$

.

Corollary 9.1.4. *If $X^* \subseteq A^*_\varphi$, then $\mu(S_\varphi) = 0$*

Theorem 9.1.5 (GD avoids strict saddles [Lee et al., 2017]). *Assume f is β-smooth and $\gamma \leq \beta^{-1}$. Then, if θ^* is a strict saddle of f,*

$$\Pr_{\theta_0}(\lim_k \theta_k = \theta^*) = 0$$

Proof. First note that every strict saddle of f is an unstable fixed point of φ. Recall:

$$\varphi(\theta) = \theta - \gamma \nabla f(\theta), D\varphi(\theta) = -\gamma \nabla^2 f(\theta)$$

Since θ^* is a strict saddle:

$$\implies \lambda_{\min}(\nabla^2 f(\theta^*)) < 0 \implies \lambda_{\max}(D\varphi(\theta^*)) > 1 \implies \theta^* \in A^*_\varphi$$

Then we have $\det(D\varphi(\theta)) \neq 0 \; \forall \theta$. If $\gamma \leq \beta^{-1}$, then $\lambda_{\min}(D\varphi(\theta)) \geq 0 \; \forall \theta$.
It remains to show that φ is a diffeomorphism: φ is injective and surjective.

Injective: Suppose there exist x, y such that

$$\underbrace{\varphi(x)}_{x - \gamma \nabla f(x)} = \underbrace{\varphi(y)}_{y - \gamma \nabla f(y)}$$

$\implies x - y = \gamma(\nabla f(x) - \nabla f(y)) \implies ||x - y|| = \gamma ||\nabla f(x) - \nabla f(y)|| \leq \gamma\beta ||x - y||$, since f is β-smooth.
But $\gamma\beta < 1 \implies x = y$.

Surjective: Show $\forall y \;\; \exists z \;\; s.t \;\; y = \varphi(z)$. Define proximal operator:

$$\phi(y) := \arg\min_x \underbrace{\frac{1}{2}||x - y||^2 - \gamma f(x)}_{F_y(x)}$$

Since $\gamma < \beta^{-1} \implies F_y$ is strongly convex w.r.t $x \implies F_y$ has a unique minimum $(= \phi(y))$.

$$\begin{aligned} \nabla_x F_y(\phi(y)) = 0 &\implies \phi(y) - y - \gamma \nabla f((\phi(y)) = 0 \\ &\implies y = \phi(y) - \gamma \nabla (f(\phi(y)) = \varphi \phi y \end{aligned}$$

□

Remark 9.1.1. Does this result imply that GD converges to local minima? Not in general. We need a couple of extra properties:

- ▷ A set of strict saddles is countable and discrete
- ▷ $\lim_k \theta_k$ exists (which is guaranteed in the convex case, but no longer true in the non-convex case). Two sufficient conditions for guarantee:
  - ◇ Isolated critical points and compact level sets.
  - ◇ Local "sharpness" condition: $\exists m, a, \epsilon$ such that:

$$||\nabla f(\theta)|| \geq m|f(\theta) - f(\theta^*)|^a \;\; \text{for } \theta \in \mathcal{N}(\theta^*, \epsilon)$$

So far, we have shown randomness in initial conditions is enough to escape saddle points, and to find local minima. What is the benefit of adding extra noise (e.g. SGD) in the dynamics?

Definition 9.1.6. An ϵ-second-order stationary point of f is θ^* such that $||\nabla f(\theta^*)|| \leq \epsilon$ and $\lambda_{\min}(\nabla^2 f(\theta^*)) \geq -\sqrt{\epsilon\rho}$, with $\rho =$ Lipschitz constant of $\nabla^2 f$.

Noisy Gradient Descent (NGD) is given by:

$$\theta_{k+1} = \theta_k - \gamma(\nabla f(\theta_k) + \epsilon_k)$$

, where ϵ_k is the noise with appropriate variance.

Theorem 9.1.7 ([Ge et al., 2015]). *If f satisfies the strict saddle conditions, then noisy GD converges to ϵ-second-order stationary point in polynomial time.*

Theorem 9.1.8 ([Jin et al., 2017]). *NGD finds ϵ-second-order stationary point with $\tilde{O}(\frac{\beta(f(\theta_0)-f^*)}{\epsilon^2})$ iterations.*

Remark 9.1.2. The benefits of noise:

- ▷ Adding noise is sufficient to escape saddles
- ▷ Up to log factors, this rate matches GD rate to find first-order stationary points
- ▷ Remarkably, noise in the dynamics is also necessary to achieve such matching rate. Even with uniform initialisation, one can construct smooth functions for which GD requires exponential time to converge [Du et al., 2017].

So far, we see how to efficiently find local minima in high dimensional landscapes using SGD. We also note that adding noise to the dynamics is not necessary for convergence, but it can speed things up (provably). But how good is the local minima? Which non-convex landscapes have only "good local minima"?

Generic case: Random functions - do they have complex landscape?

Example 9.1.9 (Spherical Spin Glass)**.** A random polynomial over the unit sphere.

$$\underbrace{\Pr(x_1, \cdots, x_d)}_{x^*} = \sum_{i=1}^{J} \underbrace{w_{i_1,\cdots,i_s}}_{\text{random i.i.d Gaussian}} x_{i_1} \cdots x_{i_s}$$

Using tools from random matrix theory, we can count expected number of critical points at every energy level (λ), and every "index" (the ratio between positive and negative eigenvalues). Conclusion is that such landscapes are complex, with exponentially many bad local minima [Auffinger et al., 2013].

Note that common machine learning losses are not random polynomials!

Positive case:

- ▷ Matrix completion, matrix factorization.

$$\min_{U,W} \ell(X, U^\top w) + R(U, W)$$

  These are landscapes without bad local minima.

- ▷ Neural Networks with linear activations.

$$\phi(x;\theta) = \underbrace{W_k \cdots W_1}_{\bar{W}} x,\ \theta = \{W_1, \cdots, W_k\}$$

9.2 Overparametrized Neural Network

We see positive results of linear neural networks (NN). How about non-linear NN? There are negative results with the so-called planted models (teacher-student framework).
Let $\phi(x;\theta)$ be an arbitrary models (e.g., NN). A planted model consider $x \sim \mu$ (data distribution), $y|x = \phi(x;\theta^*)$ for certain θ^* (the teacher). Then:

$$L(\theta) = \mathbb{E}_{(x,y)}[\ell(\underbrace{\phi(x;\theta)}_{\text{student}}, \underbrace{\phi(x;\theta^*)}_{\text{teacher}})]$$

We know $\min_\theta L(\theta) = 0$ when $\theta = \theta^*$. Note that student network can be made wider or deeper than the teacher network. [Safran and Shamir, 2016] showed that for single hidden layer `ReLU`, and arbitrary choice

of θ^*, $L(\theta)$ has bad local minima for certain widths. [Venturi et al., 2018] extended the above results for more general activation functions and general width. Yet, in practice we can successfully optimize deep models! The main insights is **overparametrization**, namely to increase the size of the student network. A topological interpretation is given as follows: consider the sublevel sets $\Omega_\lambda = \{\theta; L(\theta) \leq \lambda\}$. Call the connected components of Ω_λ "valleys". Any descent algorithm stays in valleys. As illustrated below, the spurious valley and non-spurious valley are disconnected in $\mathbb{R}^2$. The overparametrization of NN adds another dimension that might help connect these two vallies, which allows us to escape from the spurious to non-spurious valley. Note that while no spurious valley is a sufficient condition for success, it is not necessary to get high-probability statements.

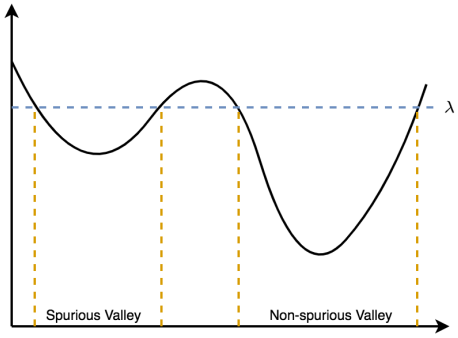

Figure 9.1: Spurious Valley versus Non-spurious Valley

Questions:

- ▷ What are the statistical consequences? Are we paying a price for having a nicer landscape in terms of generalization?
- ▷ Among these solutions, which ones are prefered by GD (SGD)?

We shall focus on shallow NN and address these questions in next lecture.

Lecture 10

Approximation with Shallow Neural Networks

Introduction

Last Time

- ▷ We looked at how gradient descent and stochastic gradient descent (GD/SGD) behaved in non-convex landscapes.
  - ◇ Specifically showed that adding noise efficiently finds local minima with convergence rate that is independent of dimension. That is, we saw through the stable manifold theory that saddle points were not a problem.
  - ◇ In practice we saw both positive and negative results. We saw different examples of problems where there were no bad local minima, problems with the exact opposite property.

This Time

- ▷ Wrap up non-convex optimization and noise
- ▷ Move on to shallow neural networks.
  - ◇ Focus on single hidden layers
  - ◇ Specifically look at their approximation properties
  - ◇ This means we need to look at kernel spaces.

Wrapping Up Non-Convex

We saw that adding a small amount of noise to gradient descent was helpful. The point of adding noise was to try to avoid saddle points. How far can we push this?
Suppose $F(\theta)$ is a generic non-convex function for $\theta \in \mathbb{R}^d$ with d being high dimensional. Taking inspiration from physics, we tree F as energy/Hamiltonian, and try to understand a probability distribution in some configuration space. Using an **inverse temperature parameter** β we can create a probability measure (the **Gibbs** or **Boltzman** measure).

$$P_\beta(\theta) = \frac{e^{-\beta F(\theta)}}{Z} \tag{10.1}$$

As $\beta \to \infty$, P_β concentrates around the minimizer(s) of F, that is

$$P_\beta(\theta) > 0 \iff \theta \in F(\theta)$$

which suggests an optimization scheme:

1. Let β grow large
2. Sample from P_β
3. In the limit you will only sample minimizers

This raises the question: how do we sample from P_β?
One answer is Markov-Chain Monte Carlo(MCMC). If we view P_β as the stationary distribution of a markov chain, then sample from it.
One scheme is **Langevin Diffusion**. Which can be done in discrete or continuous time.

$$\theta_{k+1} \leftarrow \underbrace{\theta_k - \eta \nabla F(\theta_k)}_{\text{Standard Langevin}} + \underbrace{\xi_k \sqrt{2\eta\beta^{-1}}}_{\text{noise term}}$$

where $\xi_k \sim \mathcal{N}(0, I)$. Note that the standard Langevin diffusion looks exactly like gradient descent. It can be shown that the stochastic process $\{\theta_k\}_{k=1}^{\infty}$ satisfies the markov condition and that the stationary distribution corresponds to P_β.
Wait! This is an algorithm that gives global convergence for general non-convex functions. This was supposed to be hard! The deception is that the convergence rate of this chain is very bad. In fact: using functional inequalities (log-sobolev, Poincare),we can show that

$$D_{KL}(\mu_t || P_\beta) \leq e^{-2t/c}$$

Where c is called the spectral gap which is cursed by dimensionality. Intuitively, what this is saying is if I search somewhat randomly over θ values but also following the gradient of the function, it's possibly I will get lucky and land on a global minimum. The problem is I may be waiting for an exponentially long time.

Shallow Neural Networks

Let $\sigma : \mathbb{R} \to \mathbb{R}$ be a lipschitz activation function (e.g. RELU, Sigmoid, tanh) and $x \in \mathbb{R}^d$, then define

$$f_k(x, \theta) = \sum_{k=1}^{K} \underbrace{c_k \sigma(\langle x, a_k \rangle + b_k)}_{g_k(\langle a_k, x \rangle)}$$

where c_k are the output weights, a_k the input weights and b_k the bias. We take a vector x, project it, and then apply any nonlinearity you want. In analysis these get called **ridge functions**. These are building blocks to approximate functions in high dimensions.
We define $H_\sigma = \{f_k, \theta = \{a_k, b_k, c_k\}, k \in \mathbb{N}\}$ to be the collection of all single hidden layer neural networks.
Question: How expressive is this set? **Answer:** Answers to this question take the form of universal approximation theorems.
More formally, can H_σ represent continuous functions $f : \mathbb{R}^d \to \mathbb{R}$? Even more formally we want to show that: given a metric ρ on $\mathcal{F} = \{f : \mathbb{R}^d \to \mathbb{R} | f \text{ continuous}\}$, and any $\epsilon > 0$ there exists some function $\tilde{f} \in H_\sigma$ such that $\rho(\tilde{f}, f) < \epsilon$. This is the same as asking if H_σ is **dense** in the space of continuous functions.
Intuitively, let's think about boolean functions on a hypercube $\{0, 1\}^{\tilde{d}}$ with $\tilde{d} \approx d$. Let $\Omega = \{\omega : f(\omega) = 1\}$ and let

$$\tilde{f}(x) = \cup_{\omega \in \Omega} \{w : f(\omega) == x\}$$

. We could approximate this with a neural network $e_\omega(x) = \text{sigmoid}(\langle x, \omega \rangle - (\tilde{d} - 1)$

$$x = \omega \implies \langle x, \omega \rangle = \tilde{d}$$
$$x \neq \omega \implies \langle x, \omega \rangle < \tilde{d}$$

We get our neural network by writing

$$\tilde{f} = \text{sigmoid}(\sum_{\omega \in \Omega} e_\omega(x) - (|\Omega| - 1))$$

This approximation is just look up which requires a huge number of neurons.
There are many works that solve this Petrusev, Hromik, Cysbenko, Pinkus, Leshno], Meir.
[Universal Approximation Theorem (UAT)] Let σ be a continuous, non-polynomial activation function. Then H_σ is dense in the class of continuous function $C(\mathbb{R}^d, \mathbb{R})$ under the uniform compact converge (just tells us which metric we use).

Proof. A sketch:

1. Let $f \in C(\mathbb{R}^d, \mathbb{R})$ and consider its restriction to a compact set Ω. Then we can look at the fourier series
$$\hat{f}(\xi) = \langle f, \exp(i\langle x, \xi \rangle) \rangle_{L^2(\Omega)}$$
and so $f(x) = \sum_\xi \hat{f}(\xi) \exp(i\langle x, \xi \rangle)$. Since f is continuous, its fourier series decays as the frequency $|\xi|$ grows. So $f(x)$ can be approximated by looking at the frequencies that dominate
$$\sum_{\xi : |\xi| \leq R} \hat{f}(\xi) \exp(i\langle x, \xi \rangle)$$
Now f has been expressed as a linear combination of ridge functions (Fourier ridges)

2. Now we approximate the Fourier ridges. But this is essentially a 1d approximation problem! Given $h : \mathbb{R} \to \mathbb{R}$ and $g : \mathbb{R} \to \mathbb{R}$ then we can approximate h with translated versions of g? yes if $h(t) = e^{it}$ and $g(t)$ is not a polynomial.

□

Not that UAT is NOT quantitative. It could be hiding the curse of dimensionality.
Why not a polynomial? If I have a linear combination of polynomials of degree $\leq L$ then it's still a polynomial of degree $\leq L$. No matter what I do I keep adding functions but don't keep adding to my approximation power.

Reproducing Kernel Hilbert Spaces (RKHS)

This is a particular class of function spaces. Recall that fourier transforms were an isometry in $L^2(\mathbb{R}^d)$. A natural generalization is to consider a symmetric kernel κ that goes from $\mathbb{R}^d \to \mathbb{R}^d$ such that for all finite sequences $x_1, \ldots, x_n$ and $\alpha_i \in \mathbb{R}$ we have

$$\sum_{i,j} \alpha_i \alpha_j k(x_i, x_j) \geq 0$$

Think of this as a generalization of a positive semidefinite (psd) matrix. We can define kernels naturally as

$$k(x, x') = \langle \phi(x), \phi(x') \rangle$$

for arbitrary mapping $\phi : \mathbb{R}^d \to \mathcal{F}$ with $\mathcal{F}$ a hilbert space.
[Aronszajn] k is a psd kernel iff there exists a hilbert space $\mathcal{F}$ and a mapping $\phi : \mathbb{R}^d \to \mathcal{F}$ such that $k(x, x') = \langle \phi(x), \phi(x') \rangle_{\mathcal{F}}$ Sometimes this $\mathcal{F}$ is called the **feature space** associated with k.
Examples:

- $k(x, x') = \langle x, x' \rangle$; $\phi(x) = x$
- $k(x, x') = (1 + \langle x, x' \rangle)^d$; $\phi(x)$ are monomials of x

▷ Gaussian/RBF kernel. $k(x, x') = \exp(-\alpha||x - x'||^2)$; ϕ is infinite dimensional.

If K_1, K_2 are kernels then $\alpha K_1 + \beta K_2$ is a kernel if $\alpha, \beta \geq 0$. Also $(k_1, k_2)(x, x') = k_1(x, x')k_2(x, x')$.
If we define $\mathcal{F}$ as a functional space over some feature space X then f can be seen as $\phi(x) : X \to \mathbb{R}$ $\Phi(x) = k(x, \cdot)$, $x' \mapsto k(x, x')$.
That is, if I only give you similarity, then given any input x' the only possible features are similarities to other inputs x.
We have two key properties.

1. Function evaluation: for any $f \in \mathcal{F}$ we can write $f(x) = \langle f, \phi(x) \rangle$
2. If $f = \phi(x')$ then $\phi(x')(x) = \langle \phi(x), \phi(x') \rangle = k(x, x')$. This is the **reproducing** property.

In other words: to compute dot product, I don't need the features themselves, only $k(x, x')$. As a consequence: learning in RKHS is computationally "efficient."
A good way to think about this is the representer theorem below. The work of Kimeldorf & Wahba and Smola & Scholkoff.
Suppose we have data $x_1, \ldots, x_n \in \mathcal{X}$ and labels $y_1, \ldots, y_n$, and a kernel k for some RKHS. We want to do

$$\min_{f \in \mathcal{F}} \sum_{i=1}^{n} \ell(y_i, f(x_i)) + \lambda ||f||_{\mathcal{F}}^2 \tag{*}$$

[Representer Theorem] The minimizer of equation (*) is of the form

$$\tilde{f}(x) = \sum_i \alpha_i \phi(x_i) = \sum_i \alpha_i k(x, x_i)$$

Note that ℓ doesn't need to be convex!

Proof. Write $f \in \mathcal{F}$ as the sum of two functions $f_d + V$ with $f_d \in \{\phi(x_i)\}$ and v in its orthogonal complement. So $f_d = \sum_i \alpha_i \phi(x_i)$ and V is in the orthogonal complement of that span. Then by the reproducing property

$$\begin{aligned} f(x_j) &= \langle f, \phi(x_j) \rangle = \langle f_d + v, \phi(x_j) \rangle \\ &= \langle f_d, \phi(x_j) \rangle \end{aligned}$$

By the pythagorean theorem we have $||f||^2 = ||f_d||^2 + ||v||^2$ so any minimizer must have $v = 0$ meaning it is of the desired form. □

Next week: continue link between RKHS and neural networks and see the limits of RKHS

Lecture 11

Kernels and Single Hidden-Layer Networks: Lazy Training

Introduction

Previously, we discussed universal approximation for single hidden layer NN and Hilbert Structure using kernels (RKHS). Today, we will show the limitations of kernels in terms of approximation, and learning dynamics in a single hidden layer NN (the lazy regime).

11.1 Kernels and Single Hidden Layer Neural Nets

Recall that kernels are dot-products in a generic feature space. Adding kernels together forms a kernel. So the questionis how to construct kernels from NN? Consider a neuron, $\varphi(x, v),\ x \in \mathbb{R}^d,\ v \in V$. For example:

$$\varphi(x, v) = \sigma(\langle x, v \rangle))$$

, where σ could be `Sigmoid, ReLU`, etc.

$$\begin{aligned} \varphi_v : \mathbb{R}^d &\to \mathbb{R} \\ x &\to \varphi(x, v) \end{aligned}$$

The associated "single neuron" kernel is:

$$K_v(x, x') = \varphi_v(x)\varphi_v(x') \text{ for fixed } v$$

Combining kernels for all possible v results in the kernel:

$$K(x, x') = \int_V k_v(x, x') d\tau(v)$$

where $\tau(v)$ represents the probability distribution over parameter space V.

Proposition 11.1.1 ([Bach, 2017])**.** *Let*

$$F = \left\{ f; f(x) = \int_V g(v)\varphi(x, v)\tau(dv) \text{ with } g \text{ square-integrable w.r.t } \tau;\ ||g||^2_{L^2(V,\tau)} := \int |g(v)|^2 \tau(dv) < \infty \right\}$$

Then F is an RKHS, with kernel given by

$$K(x, x') = \int \varphi(x, v)\varphi(x', v)\tau(dv)$$

For $f \in F$,

$$||f|| = \inf \left\{ ||g||_{L^2(V,\tau)} \text{ s.t } f(x) = \int g\varphi\tau(dv) \right\}$$

Intuitively if $f \leftrightarrow x \in \mathbb{R}^n$, then $||f|| \leftrightarrow \langle x, Kx\rangle = x^\top Kx$.

Remark 11.1.1. This kernel K is an average of rank-1 kernels.

$$K(x, x') = \mathbb{E}_{v\sim\tau}[\varphi(x, v)\varphi(x', v)]$$

There kernels are well-suited to stochastic (or Monte-Carlo) approximiation.

$$K(x, x') \approx \frac{1}{m}\sum_{i=1}^{m} \varphi(x, v_i)\varphi(x', v_i), \text{ where } v_i \sim \tau \text{ i.i.d}$$

Question: What is kernel regression in this setting? Recall from Representor Theorem that our predictors have the form

$$\begin{aligned}\hat{f}(x) &= \sum_{i=1}^{n} \alpha_i K(x_i, x) \\ &= \sum_{i=1}^{n} \alpha_i \frac{1}{m}\sum_{j=1}^{m} \varphi(x_i, v_j)\varphi(x, v_j) \\ &= \sum_{j=1}^{m} \beta_j \varphi_{v_j}(x)\end{aligned}$$

, where β_j is adjusted in the kernel regression step while v_j is fixed.

Answers: Recall that $\varphi_x(x) = \sigma(\langle x, v\rangle)$:

- ▷ First-layer weights are fixed (not learned!). Second-layer weights (β_j) are adjusted.
- ▷ This is Random Feature Learning proposed in [Rahimi and Recht, 2008].
- ▷ As $m \to \infty, K_m \to K$.

Question: How good are these spaces for approximation? And what is the consequence in terms of learning (generalisation?)
Let us suppose $\varphi(x, v)$ is homogenous w.r.t the parameter v, $\varphi(x, \alpha v) = \alpha^h \varphi(x, v), (\alpha > 0)$. For example, `ReLU` is homogenous while `tanh` is not. Without loss of generality, we can assume that parameter space V is compact (e.g., the unit sphere). For now, let's fix τ to be uniform measure on the sphere. The previous space F now becomes:

$$G_2 = \left\{ f : S^d \to \mathbb{R}; f(x) = \int g(v)\sigma(\langle x, v\rangle)dv, \ \int |g(v)|^2 dv < \infty \right\}$$

G_2 is an RKHS, and its kernel can be computed explicitly. Rather than fixing the distribution over parameters, we can also adjust it doing training. What is the associated functional space?

$$G_1 = \left\{ f : s^d \to \mathbb{R}; F(X) = \int \sigma(\langle x, v\rangle)\mu(dv) \text{ s.t. } \mu \text{ is a Radon measure in } S^d \right\}$$

The total variation norm is given by:

$$||\mu||_{TV} := \sup_{|X|\leq 1} \int X(t)\mu(dt)$$

Fact: G_1 is no longer a Hilbert space, but it is a Banach space.

$$||f||_{G1} := \inf \left\{ ||\mu||_{TV};\ f(x) = \int \sigma(\langle x, v\rangle)\mu(dv) \right\}$$

Question: What is the relationship between G_1 and G_2?
For simplicity, consider a measure μ with density $\mu(dv) = \rho(v)dv$. In that case, $||\mu||_{TV} = \int |\rho(v)|dv$.
From Jensen Inequality:

$$\mathbb{E}|X| \leq (\mathbb{E}|X|^2)^{\frac{1}{2}} \implies ||f||_{G1} \leq ||f||_{G2} \implies G_2 \subseteq G_1$$

Key Questions:

1. Which functions belong to these spaces?
2. Approximation with finite number of neurons?

Let's start with Question 2.

Proposition 11.1.2. *Assume* $\sup_{v \in S^d} ||\varphi_V||^2_{L^2(S^d,V)} = B < \infty$. *Given* $f \in G_1$, *we can find an m-term approximation,* $f_m = \sum_{j=1}^m \alpha_j \varphi_{v_j}$ *such that:*

$$||f - f_m||^2_{L^2(S^d,V)} \leq \frac{B||f||^2_{G_1}}{m}$$

, which gives us a statistical rate.

Proof. Let $f(x) = \int \varphi(x,v)\mu(dv)$, $||\mu||_{TV} = ||f||_{G_1}$. Consider

$$q(v) = \frac{|\mu|(v)}{||\mu||_{TV}} \tag{11.1}$$

We have: $q \geq 0$; $\int q(dv) = \frac{\int |\mu|(dv)}{||\mu||_{TV}} = \frac{||\mu||_{TV}}{||\mu||_{TV}} = 1$. By re-aranging (11.1):

$$\mu(v) = \text{sign}(\mu(v))||\mu||_{TV} q(v)$$

Plugging in $f(x)$:

$$\begin{aligned} f(x) &= \int \underbrace{\varphi(x,v)\text{sign}(\mu(v))||\mu||_{TV}}_{\bar{\varphi}(x,v)} q(dv) \\ &= \mathbb{E}_{v\sim q}[\bar{\varphi}(x,v)] \end{aligned}$$

Let $f_m(x) = \frac{1}{m}\sum_{i=1}^m \bar{\varphi}(x,v_i)$, where $v_i \overset{\text{i.i.d}}{\sim} q$.

$$\begin{aligned} \mathbb{E}_V||f_m - f||^2_{L^2(S^d,V)} &= \mathbb{E}_V\mathbb{E}_x|f_m(x) - f(x)|^2 \\ &= \mathbb{E}_x\mathbb{E}_V|\frac{1}{m}\sum_{i=1}^m \bar{\varphi}(x,v_i) - \mathbb{E}_{v\sim q}[\bar{\varphi}(x,v)]|^2 \\ &\leq \frac{1}{m}\mathbb{E}_x\mathbb{E}_V|\bar{\varphi}(x,v)|^2 \text{ (upper bound by 2nd moment)} \\ &\leq \frac{||f||^2_{G_1}B}{m} \end{aligned}$$

If in expectation over $v_1, \cdots, v_m$ we have the error $\frac{||f||^2_{G_1}B}{m}$, then there must exists "lucky draws" below the expectation (Markov Inequality). □

Corollary 11.1.3. *Same is true for* $f \in G_2$, *since* $||f||_{G_1} \leq ||f||_{G_2}$

Remark 11.1.2. What we seen so far:

▷ Curse of dimensionality? It can only come from $||f||_{G_1}, ||f||_{G_2}$.

- ▷ These are known as Monte-Carlo approximations. Other finite-neuron approximations are possible using deterministic approaches (e.g., Frank-Wolfe Algorithm - essentially the same rate).
- ▷ These can be easily extended to $\mathbb{R}^d$ as opposed to S^d.

Back to Question 1: Which functions belong to there spaces?
Functions $f : S^d \to \mathbb{R}$ with at least $\frac{d+3}{2}$ bounded derivatives (by C) belong to G_2 (and thus to G_1), with $||f||_{G_2} \propto C$

Proposition 11.1.4 ([Bach, 2017]). *Let $f : S^d \to \mathbb{R}$ with Lipschitz(f) $\leq \eta$. Then $\exists h \in G_2$ with $||h||_{G_2} \leq \delta$ and*

$$\sup_{z \in S^d} |h(z) - f(z)| \lessapprox C(d)\eta(\frac{\delta}{\eta})^{-\frac{1}{1+(\frac{d-1}{2})}} \approx -\frac{1}{d}$$

Note that this rate of approximation is cursed by dimensionality!
Question: What is the main advantage of G_1 over G_2?
Answer: Consider $f(x) = \sigma(\langle x, \theta^* \rangle)$, a single neuron (ridge function).

- ▷ $f \in G_2$? No, because we need regularities in all direction. The norm will blow up in high dimension.
- ▷ $f \in G_1$? Yes. Since this is a more "relaxed" space, we can "redirect" all the mass at θ^* to get $||f||_{G_1}$ independent of dimension.

So far, we have seen the space G_2 is non-adaptive (depends on the "prior" over the parameter space τ). The space G_1 can adapt to low-dimensional features. This shows the essence of representation learning is to adjust the features to benefit efficient approximation. To summarize:

- ▷ Kernel learning $\leftrightarrow$ only learn the last layer $\leftrightarrow$ learn on a RKHS
- ▷ Shallow feature learning $\leftrightarrow$ learn in G_1
- ▷ Deep feature learning $\leftrightarrow$ currently under study (basically unknown).

Remark 11.1.3. Risk of learning functions with large (even exponentially large) norms? In order to control generalization error, we need to learn with small complexity, as generalization error in a RKHS $\approx \frac{||f||_{\text{RKHS}}}{\sqrt{n}}$.

11.2 Optimisation of non-linear function

We have discussed the functional view of single hidden layer NN. What is the link between optimizing in parameter space versus in function space?

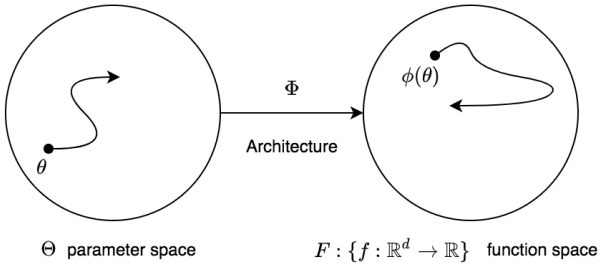

Figure 11.1: Link between parameter and function space

Examples of function and its parameter space can be given as:

$$\phi(\theta) : \mathbb{R}^d \to \mathbb{R}$$
$$x \to \phi(\theta; x) = W_k \rho(W_{k-1}\rho \cdots Wx)$$
$$\Theta : \{\text{measures of } \mu\}. \ \phi(\mu)(x) = \int \varphi(x, \theta)\mu(d\theta)$$

We will continue next week to further discuss how to optimize in the kernel regime G_2 and G_1 respectively.

Lecture 12

From Lazy to Active Learning in Shallow Neural Networks

Previously, we saw that kernels could be associated with neural networks in which the first layer weights are frozen. We also saw that despite enjoying universal approximation, kernels in general cannot adapt to high-dimensional functions with a low-dimensional structure, which is the curse of dimensionality. And lastly, we discussed the total variation spaces.

In this lecture, we continue to discuss shallow neural networks. We discuss the dynamics in shallow models from "lazy" to "active", where "lazy" is associated with kernel and "active" is associated with total variation.

12.1 Optimization of Non-linear Functions

Consider a mapping Φ from a space Θ of parameters θ, where training happens, to the space $\mathcal{F}$ of functions , where prediction happens. For instance, in a neural network, we have a mapping $\Theta \to (\mathbb{R}^d \to \mathbb{R})$ with $\theta = (w_1, \dots, w_L) \mapsto (x \mapsto \Phi(\theta, x) = w_L \sigma w_{L-1} \cdots w_1 x)$.

Recall the empirical risk function $L : \mathcal{F} \to \mathbb{R}$ bounded below by 0 where learning happens. An example of such function is $L(f) = \|f - f^*\|^2$. In practice, we are interested in the risk in terms of the parameter space Θ, i.e. $E(\theta) = L(\Phi(\theta))$. The best case scenario is when L is convex and Φ is linear because they imply that E is also convex. An example is when $\mathcal{F}$ is an RKHS

$$\Phi(\theta) = \sum_{i=1}^{n} \theta_i k(x_i, \cdot)$$

from representer theorem.

Now, we are interested in the case when Φ is not linear. We start with the gradient descent using fixed step size $\eta > 0$

$$\theta_{k+1} = \theta_k - \eta \nabla_\theta \underbrace{L(\Phi(\theta_k))}_{E(\theta_k)}.$$

We will also try to understand the gradient descent in continuous time, i.e. the gradient flow

$$\dot{\theta}(t) = -\nabla_\theta L(\Phi(\theta(t))).$$

Denote $\Phi_t = \Phi(\theta(t))$. Consider the dynamics in the function space

$$\frac{\mathrm{d}}{\mathrm{d}t}\Phi_t = D\Phi(\theta(t))^\top \frac{\mathrm{d}}{\mathrm{d}t}\theta(t) = D\Phi(\theta(t))^\top D\Phi(\theta(t))(\delta L)(\Phi(t)) = -K_t(\delta L)(\Phi(t)),$$

where δL is the functional derivative and K is the tangent kernel

$$(K_t f)(x) = \int k_t(x, x') f(x') \, \mathrm{d}\nu(x')$$

where $k_t(x, x') = \langle \nabla_\theta \Phi(\theta(t), x), \nabla_\theta \Phi(\theta(t), x') \rangle$. In general, we have a parametrization of a function that is not linear, K_t looks at linearization of the parametrization at a point.

Example 12.1.1. We compute the kernel for a large shallow neural network

$$\Phi(\theta; x) = \frac{1}{\sqrt{m}} \sum_{i=1}^{m} g(\theta_i, x),$$

where $\theta_i \sim \mu_0$ i.i.d (μ_0 is an initial distribution) and $g(\theta, x) = c\sigma(\langle x, w \rangle)$. We have

$$K_0(x, x') = \frac{1}{m} \sum_{i=1}^{m} \left\langle \nabla_\theta g(\theta, x)\big|_{\theta=\theta_i}, \nabla_\theta g(\theta_i, x')\big|_{\theta=\theta_i} \right\rangle,$$

which converges to

$$\mathbb{E}_{\theta \sim \mu_0}[\langle \nabla_\theta g(\theta, x), \nabla_\theta g(\theta_i, x') \rangle] = \overline{K}(x, x')$$

as $m \to \infty$. This is called the *neural tangent kernel*

Theorem 12.1.2. *For any $T > 0$ as $m \to \infty$, uniformly in $[0, T]$ (and with uniform compact topology), we have*

$$K_t(x, x') \to \overline{K}(x, x')$$

for all $t \in (0, t)$. In particular,

$$\frac{\mathrm{d}}{\mathrm{d}t} \Phi(\theta(t)) \cong -\overline{K}(\Phi(\theta(t)) - f^*).$$

Note that this also holds for deep neural network. The key point here is that in this scaling limit ($1/\sqrt{n}$), we are "de facto" using a linear learning model. We are interested in understanding why a non-linear model (neural networks) behaves as a linear model (kernels).

12.2 Lazy Dynamics

Recall that $\Phi(\theta)$ is a differential map $\Theta \to \mathcal{F}$, where θ_0 is the initial point. Consider the linearized tangent model

$$\overline{\Phi}(\theta) = \Phi(\theta_0) + \langle \theta - \theta_0, \nabla_\theta \Phi(\theta_0) \rangle.$$

We wonder when is learning under these two models similar.

Let $E(\theta) = L(\Phi(\theta))$ and assume that θ_0 is such that $E(\theta_0) > 0$. Recall that a gradient step is $\theta_1 = \theta_0 - \eta \nabla E(\theta_0)$. The relative change in the objective is

$$\Delta E = \frac{E(\theta_0) - E(\theta_1)}{E(\theta_0)} \approx \frac{\langle \nabla E(\theta_0), \theta_1 - \theta_0 \rangle}{E(\theta_0)} = \eta \frac{\langle \nabla E(\theta_0), \nabla E(\theta_0) \rangle}{E(\theta_0)}.$$

The relative change in tangent model is

$$\Delta(D\Phi) = \frac{\|D\Phi(\theta_1) - D\Phi(\theta_0)\|}{\|D\Phi(\theta_0)\|} \approx \frac{\eta \left\| D^2 \Phi(\theta_0) \right\| \|\nabla E(\theta_0)\|}{\|D\Phi(\theta_0)\|}.$$

Definition 12.2.1 (Lazy Regime)**.** A **lazy regime** is where $\Delta(D\Phi) \ll \Delta E$.

We see that

$$\frac{\left\|D^2\Phi(\theta_0)\right\|\|\nabla E(\theta_0)\|}{\|D\Phi(\theta_0)\|} \ll \frac{\|\nabla E(\theta_0)\|^2}{E(\theta_0)} \quad \Longrightarrow \quad \frac{\left\|D^2\Phi(\theta_0)\right\|}{\|D\Phi(\theta_0)\|} \ll \frac{\|\nabla E(\theta_0)\|}{E(\theta_0)}. \tag{12.1}$$

When $L(f) = \|f - f^*\|^2$, then $E(\theta) = \|\Phi(\theta) - f^*\|^2$. Then

$$\|\nabla E(\theta_0)\| = \|D\Phi(\theta)(\Phi(\theta) - f^*)\| \approx \|D\Phi(\theta_0)\|\|\Phi(\theta) - f^*\|.$$

As a result, we obtain Equation (12.1) and is equivalent to

$$\|\Phi(\theta) - f^*\| \frac{\left\|D^2\Phi(\theta)\right\|}{\|D\Phi(\theta)\|^2} \ll 1.$$

We refer to the LHS as $\kappa_\Phi(\theta)$, which is the relative scale of Φ at θ. This is associated with the curvature of the mapping $\Omega \to \mathcal{F}$. One interesting result is given as follows.

Theorem 12.2.2. *Assume that Φ and $D\Phi$ are Lipschitz around θ_0. Let $\theta(t)$ be the non-linear gradient flow and $\overline{\theta}(t)$ be the linearized gradient flow. Then, for $t \leq C_\Phi$, it holds that*

$$\frac{\left\|\Phi(\theta(t)) - \overline{\Phi}(\overline{\theta}(t))\right\|}{\|\Phi(\theta_0) - f^*\|} \lesssim t^2 \kappa_\Phi(\theta_0).$$

This theorem states that if we have a non-linear model that is almost flat, then the two trajectories we get (following either the true trajectory or the linear trajectory) are going to be close to each other. (Note that the theorem above is in finite time. The infinite-time version can be seen in the original paper under more restrictive assumptions.) This result is a very general yet insightful way to think about learning. If we have a differentiable model with non-linearity of the parameterization that has a small relative scale, then we should directly train with the respective kernel model that is much faster.

Now we are interested in knowing when does lazy regime happen because we want to avoid it. We give three examples below.

Example 12.2.3. Consider a scaled model $\Phi_\alpha = \alpha\Phi(\theta)$, where $\alpha > 0 \in \mathbb{R}$. We look at the relative scale

$$\kappa_{\Phi_\alpha}(\theta_0) = \|\Phi_\alpha(\theta_0) - f^*\| \frac{\left\|D^2\Phi_\alpha(\theta_0)\right\|}{\|D\Phi_\alpha(\theta_0)\|^2} = \frac{1}{\alpha}\|\alpha\Phi(\theta_0) - f^*\| \frac{\left\|D^2\Phi(\theta_0)\right\|}{\|D\Phi(\theta_0)\|^2}.$$

In particular, if $\Phi(\theta_0) = 0$ (), then

$$\kappa_{\Phi_\alpha}(\theta_0) = \frac{1}{\alpha}\kappa_\Phi(\theta_0),$$

which approaches to 0 as α increases. This makes that we can make the model more linear by increasing the scale of initialization.

Example 12.2.4. Consider a single hidden layer neural network

$$\Phi_m(\theta) = \alpha(m) \sum_{i=1}^{m} g(\theta_i),$$

where $\theta_i \sim \mu$ i.i.d. and $\theta \in \mathbb{R}^d$, such that $\mathbb{E}_{\theta\sim\mu}[g(\theta)] = 0$. We compute the relative scale. First,

$$\mathbb{E}\|\Phi_m(\theta)\|^2 = m \cdot \alpha(m)^2 \mathbb{E}\|g(\theta)\|^2.$$

The Jacobian of $\Phi_m(\theta)$ is

$$D\Phi_m(\theta) = \alpha(m) \begin{bmatrix} Dg(\theta_1) & Dg(\theta_2) & \cdots & Dg(\theta_m) \end{bmatrix}$$

Then

$$\frac{D\Phi_m(\theta)D\Phi_m(\theta)^\top}{\alpha(m)^2 \cdot m} = \frac{1}{m}\sum_{i=1}^m Dg(\theta_i)Dg(\theta_i)^\top$$

which approaches to $\mathbb{E}Dg(\theta)Dg(\theta)^\top$ as m increases. Therefore,

$$\mathbb{E}\|D\Phi_m(\theta)\|^2 = \mathbb{E}\left\|D\Phi_m(\theta)\cdot D\Phi_m(\theta)^\top\right\| \sim \alpha(m)^2 m\mathbb{E}\|Dg(\theta)\|^2.$$

The Hessian is

$$\left\|D^2\Phi_m(\theta)\right\| = \sup_{u\in\mathbb{R}^{d\times m},\|u\|\le 1} \alpha(m)\sum_{i=1}^m u_i^\top D^2 g(\theta_i)u_i \le \alpha(m)\sup_{\theta_i}\left\|D^2 g(\theta_i)\right\| \le \alpha(m)\mathrm{Lip}(Dg).$$

Since $\|\Phi_m(\theta) - f^*\| \le \|\Phi_m(\theta)\| + \|f^*\|$ by triangle inequality, we have

$$\kappa_{\Phi_m}(\theta) \le (A + \sqrt{m}\alpha(m)\cdot B)\frac{\alpha(m)C}{m\alpha(m)^2 D} \lesssim \frac{C_1}{\sqrt{m}} + \frac{C_2}{m\alpha(m)}.$$

We see that $\kappa_{\Phi_m}(\theta)$ approaches to 0 if $m\alpha(m)$ approaches ∞ as m approaches ∞. Recall that in NTK, $\alpha(m) = 1/\sqrt{m}$. We are in the lazy regime. It is good in the sense that the model has global convergence of critical loss in the regime. It is bad because kernel is not very good at learning in high-dimensional space. Next, we would like to know what happens when $\alpha(m) = 1/m$.

12.3 The Mean-Field Formulation of Shallow Neural Networks

Consider the mapping

$$\Phi(\theta; x) = \frac{1}{m}\sum_{i=1}^m g(\theta_i; x)$$

where $g(\theta; x) = c\sigma(\langle x, w\rangle + b)$ and $\theta \in \mathbb{R}\times\mathbb{R}^d\times\mathbb{R} := \mathcal{D}$. Consider now the probability measure over $\mathcal{D}$

$$\mu_m = \frac{1}{m}\sum_{i=1}^m \delta_{\theta_i},$$

where $\mu_m \in P(\mathcal{D})$, the space of probability measures over $\mathcal{D}$.

We want to know if we can recover δ given μ_m. Recall that if $\mu(\theta) = \delta_{\theta_i^*}$, then for all test function X on $\mathcal{D}$,

$$\int X(\theta)\mu\,\mathrm{d}\theta = X(\theta^*)$$

We have

$$\Phi(\theta; x) = \frac{1}{m}\sum_{i=1}^m g(\theta_i; x) = \int_{\mathcal{D}} g(\theta; x)\mu_m\,\mathrm{d}\theta$$

Instead of thinking in terms of $\mathcal{D}^{\otimes m} \to \mathcal{F}$, we can think in terms of $P(\mathcal{D}) \to \mathcal{F}$.

$$\frac{1}{m}\sum_{i=1}^m g(\theta_i; x) = \phi(\theta; x) = \Phi(\mu_m; x) = \int g(\theta; x)\mu_m\,\mathrm{d}\theta$$

We go from a very non-linear Φ to a Φ that is linear with respect to μ. Also, we go from a Euclidean space $\mathcal{D}^{\otimes m}$ to $P(D)$ that will not behave as an Euclidean space.

Lecture 13

Learning in Variation Spaces and Beyond

Topic today:
Non-linear learning in shallow Neural networks using measure space; particle interaction mean-field perspective
concluding remarks (how to go beyond the shallow learning)

13.1 Shallow NN as particle-interaction Systems

Definition 13.1.1 (Shallow NN model)**.**

$$\phi(x;\theta_1,\cdots,\theta_m)=\frac{1}{m}\sum_{i=1}^{m}g(x;\theta_i)$$

$$g(x;\theta)=c\cdot\sigma(\langle x,a\rangle+b),\theta=(a,b,c)\in\mathbb{R}^d\times\mathbb{R}\times\mathbb{R}=\mathcal{D}$$

Consider least-square regression.

$$\min_{\theta_1,\cdots,\theta_m\in\mathcal{D}}\frac{1}{2}\|\phi(\theta_1,\cdots,\theta_m)-f^*\|_2^2,\ \text{with}\ \|f\|_2^2=\mathbb{E}_{x\sim\nu}[|f(x)|^2]$$

We can think the expectation as either the empirical loss or the population loss
We can define $E(\theta_1,\cdots,\theta_m):=\frac{1}{2}\|\phi(\theta_1,\cdots,\theta_m)-f^*\|_2^2$ as the loss function.
By developing the square,

$$\begin{aligned}E(\theta_1,\cdots,\theta_m)&=\frac{1}{2}\|f^*\|_2^2-\langle f^*,\phi(\theta_1,\cdots,\theta_m)\rangle+\frac{1}{2}\|\phi\|_2^2\\&\equiv-\frac{1}{m}\sum_{i=1}^{m}\langle f^*,g(\theta_i)\rangle+\frac{1}{2m^2}\sum_{i,j=1}^{m}\langle g(\theta_i),g(\theta_j)\rangle\ \text{(note: we omit}\ \|f^*\|\ \text{since it's a constant)}\end{aligned}$$

$$\langle f,g\rangle:=\mathbb{E}_{x\sim\nu}f(x)\cdot g(x)$$

Lets denote

$$F(\theta):=\langle f^*,g(\theta)\rangle=\mathbb{E}_{x\sim\nu}[f^*(x)\cdot g(\theta;x)]$$

$F(\theta)$ represents how a neuron $g(\theta)$ correlates with the target function f^* over the data.

$$K(\theta,\theta')=\langle g(\theta),g(\theta')\rangle$$

The energy becomes

$$E(\theta_1, \cdots, \theta_m) = -\frac{1}{m}\sum_{i=1}^{m} F(\theta_i) + \frac{1}{2m^2}\sum_{i,j=1}^{m} K(\theta_i, \theta_j)$$

So far we just rewrite the loss function without doing anything useful.
Now we are going to do Ausatz
We can associate a neuron $\theta_i \in \mathcal{D}$ with a particle in space $\mathcal{D}$, so now we have m particle in the parameter space $\mathcal{D}$ and an energy function $E(\theta_1, \cdots, \theta_m)$ of a system of m interacting particles.
physic analog:
$\theta_1, \cdots, \theta_m$: position of particles in $\mathcal{D}$.
F: external potential
K: interaction potential
Gradient flow (up to scaling) with respect to θ gives us the dynamics

$$\dot{\theta}_i(t) = -m\nabla_{\theta_i} E(\theta_1, \cdots, \theta_m) = -\nabla F(\theta_i) + \frac{1}{m}\sum_{j=1}^{m} \nabla K(\theta_i, \theta_j)$$

This a Lagrangian Description of the system. However it's complicated (even "hopeless") to analyze at finite m from a deterministic perspective.
Let's consider instead an Eulerian perspective:
associate these position with a sum of probability measures over $\mathcal{D}$

$$\overrightarrow{\theta} = (\theta_1, \cdots, \theta_m) \in \mathcal{D}^{\otimes m}$$

with measure

$$\mu^{(m)} = \frac{1}{m}\sum_{i=1}^{m} \delta_{\theta_i} \in \mathcal{P}(\mathcal{D})$$

our model

$$\phi(\overrightarrow{\theta}; x) = \frac{1}{m}\sum_{i=1}^{m} g(\theta_i; x)$$

now becomes

$$\phi(\overrightarrow{\theta}; x) = \int_{\mathcal{D}} g(\theta; x)\mu^{(m)}(d\theta)$$

(It's a consequence of the fact that if $\mu = \delta_{\theta^*}$, then $\int \mathcal{X}(\theta)\mu(d\theta) = \mathcal{X}(\theta^*)$)
Now we can see that our function ϕ is linear with μ!
It follows that the loss function is now

$$\begin{aligned}\mathcal{E}[\mu] &= \frac{1}{2}\left\| \int g(\theta; x)\mu(d\theta) - f^* \right\|^2 \\ &\equiv -\int_{\mathcal{D}} F(\theta)\mu(d\theta) + \frac{1}{2}\int\int_{\mathcal{D}^2} K(\theta, theta')\mu(d\theta)\mu(d\theta')\end{aligned}$$

And we can see our loss is quadratic and convex w.r.t. mu.
Does that mean we are done?
No! $\mathcal{E}[\mu]$ is convex with respect to mixtures:
if we have 2 probability distribution $\mu_0, \mu_1 \in \mathcal{P}(\mathcal{D})$
A mixture μ_t can be define as $\mu_t = t \cdot \mu_1 + (1-t)\mu_0,\ t \in (0,1)$
And the convexity gives us

$$\mathcal{E}[\mu_t] \leq t\mathcal{E}[\mu_1] + (1-t)\mathcal{E}[\mu_0], \forall t \in (0,1), \forall \mu_0, \mu_1 \in \mathcal{P}(\mathcal{D})$$

But our dynamics do not "fit" with this metric structure!
question: What is the relationship between gradient descent and Metric?
Answer (very quick): proximal view point of gradient descent.
Classic gradient descent step in an Euclidean space:

$$\begin{aligned}\theta_{t+1} &= \theta_t - \eta_t \nabla \delta E(\theta_t) \\ &= \arg\min_{\theta} \left\{ \underbrace{E(\theta_t) + \langle \nabla E(\theta(t), \theta - \theta_t \rangle}_{\text{linear approximation of E at } \theta_t} + \underbrace{\frac{1}{2\eta_t} \|\theta - \theta_t\|^2}_{\text{proximity term}} \right\}\end{aligned}$$

We only trust the linear approximation of E around θ_t
Here we measure proximity with L2 metric. It turns out this is not always appropriate, even not well-defined!!
The generalization of Gradient Descent to other proximal metrics is given by Mirror Descent (NemiovskiYudin'83)

$$x_{t+1} = \arg\min_{x \in \mathcal{X}} \left\{ E(x) + \frac{1}{\eta_t} \underbrace{\mathcal{D}(x;_t)}_{\text{Bregmen Divergence}} \right\}$$

$\mathcal{X}$ can be non-Euclidean.

Bibliography

A. Auffinger, G. B. Arous, and J. Černỳ. Random matrices and complexity of spin glasses. Communications on Pure and Applied Mathematics, 66(2):165–201, 2013.

F. Bach. Breaking the curse of dimensionality with convex neural networks. The Journal of Machine Learning Research, 18(1):629–681, 2017.

L. Bottou and O. Bousquet. The tradeoffs of large scale learning. In Advances in neural information processing systems, pages 161–168, 2008.

S. Bubeck. Convex optimization: Algorithms and complexity. page 39. URL http://arxiv.org/abs/1405.4980. cite arxiv:1405.4980Comment: A previous version of the manuscript was titled "Theory of Convex Optimization for Machine Learning".

B. C. Csáji et al. Approximation with artificial neural networks. Faculty of Sciences, Etvs Lornd University, Hungary, 24(48):7, 2001.

S. S. Du, C. Jin, J. D. Lee, M. I. Jordan, A. Singh, and B. Poczos. Gradient descent can take exponential time to escape saddle points. In Advances in neural information processing systems, pages 1067–1077, 2017.

V. Fabian. On asymptotic normality in stochastic approximation. Ann. Math. Statist., 39(4):1327–1332, 08 1968. doi: 10.1214/aoms/1177698258. URL https://doi.org/10.1214/aoms/1177698258.

R. Ge, F. Huang, C. Jin, and Y. Yuan. Escaping from saddle points—online stochastic gradient for tensor decomposition. In Conference on Learning Theory, pages 797–842, 2015.

C. Jin, R. Ge, P. Netrapalli, S. M. Kakade, and M. I. Jordan. How to escape saddle points efficiently. In Proceedings of the 34th International Conference on Machine Learning-Volume 70, pages 1724–1732. JMLR. org, 2017.

J. D. Lee, I. Panageas, G. Piliouras, M. Simchowitz, M. I. Jordan, and B. Recht. First-order methods almost always avoid saddle points, 2017.

U. v. Luxburg and O. Bousquet. Distance-based classification with lipschitz functions. Journal of Machine Learning Research, 5(Jun):669–695, 2004.

V. Maiorov and R. S. Meir. Approximation bounds for smooth functions in c (r/sup d/) by neural and mixture networks. IEEE Transactions on Neural Networks, 9(5):969–978, 1998.

S. Mallat. Group invariant scattering. Communications on Pure and Applied Mathematics, 65(10):1331–1398, 2012.

A. Nemirovsky. Problem complexity and method efficiency in optimization.

Y. Nesterov. Introductory lectures on convex optimization: A basic course, volume 87. Springer Science & Business Media, 2013.

Y. E. Nesterov. A method for solving the convex programming problem with convergence rate o (1/kˆ 2). In Dokl. akad. nauk Sssr, volume 269, pages 543–547, 1983.

A. Rahimi and B. Recht. Random features for large-scale kernel machines. In Advances in neural information processing systems, pages 1177–1184, 2008.

I. Safran and O. Shamir. On the quality of the initial basin in overspecified neural networks. In International Conference on Machine Learning, pages 774–782, 2016.

S. Smale. Differentiable dynamical systems. Bulletin of the American mathematical Society, 73(6):747–817, 1967.

G. Strang. Wavelet transforms versus fourier transforms. Bulletin of the American Mathematical Society, 28(2):288–305, 1993.

W. Su, S. Boyd, and E. Candes. A differential equation for modeling nesterov's accelerated gradient method: Theory and insights. In Advances in Neural Information Processing Systems, pages 2510–2518, 2014.

L. Venturi, A. S. Bandeira, and J. Bruna. Spurious valleys in two-layer neural network optimization landscapes, 2018.