Deep learning theory (DRAFT[1])

Matus Telgarsky

September 25, 2023

[1]

This is an in-progress heavy rewrite. Somehow, google is indexing it. The full previous version is at `https://mjt.cs.illinois.edu/dlt/index.pdf`, and will eventually become subsumed.

Contents

Chapter 1

Introduction

In practice, a family of functions $\mathcal{F}$ is called *neural/deep network* or *neural/deep network architecture* if it has the following characteristics.

1. $\mathcal{F}$ **is parameterized by** p **real numbers:** $\mathcal{F} := \{x \mapsto F(x; w) : w \in \mathbb{R}^p\}$ for some fixed function F and fixed parameter dimension p and input space $x \in \mathcal{X}$;
2. **Bounded computation:** the number of elementary operations on real numbers needed to compute $\mathcal{F}$ is uniformly bounded over x and w.
3. **Efficient hardware implementation:** standard choices for the construction of F co-evolve with hardware developments.
4. **Convenient programming interface:** libraries so simple that one hour suffices, with no prior library or even ML experience.
5. **Amenable to gradient descent:** a single good parameter choice $w \in \mathbb{R}^p$ can be found easily via a first-order descent method.

Items 1 and 2 are usually treated as the standard definition after some restrictions to "elementary operations". These conditions are satisfied when F is defined by a directed acyclic graph with a single source corresponding to input x, a single sink corresponding to the output, and graph nodes perform a bounded amount of computation parametrized by $w \in \mathbb{R}^p$; this graph-based perspective motivates the word "network". Equivalently, code written in a standard library such as `pytorch` meets these definitions if there are no loops, no unbounded recursion, or equivalents.

It should also be noted that Items 1 and 2 are not sufficient to characterize deep networks, as they are also satisfied by polynomial classifiers, SVMs, and many other choices. This suggests the importance of further items, specifically Items 3 to 5, which aim to capture what makes deep networks different and successful. This text will not address Items 3 and 4, but will Item 5 as a central concern.

The magic of deep networks is that if we have some data and a corresponding performance measure $\widehat{\mathcal{R}}(w)$ of our selected parameters $w \in \mathbb{R}^p$, then by further tuning w with a simple procedure such as gradient descent (Item 5 above), we can induce $\mathcal{R}(w)$ to also be small, where $\mathcal{R}$ corresponds to our performance on data we have not seen. Here are two examples.

Example 1.1 (Prompt-based image generation)**.** The popular software DALL-E 2 takes an english sentence (the $x \in \mathcal{X}$), passes it through a complicated $x \mapsto F(x; w)$ where $w \in \mathbb{R}^p$ with $p \approx 2^{32}$, and outputs a 1024×1024 image with 3 color channels (Ramesh et al., 2022). Even though there are almost 10^9 training examples, this seemingly large number is dwarfed by the magnitude of joint input and output spaces: we have a vanishingly sparse cover of all reasonable input/output pairs. But of even greater concern, it is not clear how to even define $\widehat{\mathcal{R}}$ and $\mathcal{R}$, let alone minimize $\widehat{\mathcal{R}}$ and show that $\mathcal{R}$ is also small, and moreover analyze the full procedure. ◇

Example 1.2 (Protein folding)**.** AlphaFold uses similarly complicated F and huge $w \in \mathbb{R}^p$ to convert the amino acid sequence of a protein (the $x \in \mathcal{X}$) into a three-dimensional description of the protein (i.e., how the protein is "folded") (Jumper et al., 2021). In this case, $\widehat{\mathcal{R}}$ and $\mathcal{R}$ can be defined in a variety of reasonable ways, but are very costly: sometimes it requires a large amount of equipment and scientific expertise to produce a few accurate input/output pairs. Due to this expense, the set of labeled examples is not only small, it is *biased* (e.g., since analyzing certain proteins seems more beneficial to humans), meaning it is not a uniform sampling of the collection of proteins across all species. Despite this, AlphaFold happily outputs folding information for all amino acid sequences it is fed, the shock being its accuracy on $\mathcal{R}$ (which can then be checked) despite the seen and unseen data having different structure. ◇

1.1 Scope of this book

The tools in this book are from being able to analyze **????**; the focus is summarized as follows.

1. This book will primarily analyze the simple *feed-forward* architectures in Definition 1.3. A few other architectures will be discussed in *[todo 1/93]*.
2. The focus is on arguing why $\mathcal{R}$ can be made small, via a decomposition of $\mathcal{R}$ which has small $\widehat{\mathcal{R}}$ as a subproblem, given below in eq. (1.4).
3. The goal is to present mathematical material is tools with short proofs and flexible usage, and hopefully to de-emphasize specific dogma. This also means some results are omitted simply because the author could not produce a short proof.
4. Wherever possible, bounds are presented with data- and algorithm-sensitive quantities.
5. Bridging old and new: deep networks have been investigated many times in waves, each time bringing new ideas, but often they are not connected across waves. An explicit example is *[todo 2/93]*.

Standard feedforward architectures are as follows.

Definition 1.3 (Feedforward networks.)**.** A *feedforward architecture* has $w = ((W_1, \dots, W_L), (b_1, \dots, b_L))$, a tuple of matrices and vectors, the *weights* $(W_i)_{i=1}^L$ and *biases* $(b_i)_{i=1}^L$, and computes

$$F(x; w) := \sigma_L(W_L \sigma_{L-1}(W_{L-1} \cdots \sigma_1(W_1 x + b_1) \cdots + b_{L-1}) + b_L),$$

where $(\sigma_i)_{i=1}^L$ are fixed nonlinear functions (also called *activations* or *transfer functions*). A common example is for each σ_i to apply a single function coordinate-wise, popular examples being the ReLU $z \mapsto \max\{0, z\}$ and the sigmoid $z \mapsto 1/(1 + \exp(-z))$. The gates can also be multi-variate, another common choice being the *softmax mapping* $v \mapsto \exp(v)/\sum_j \exp(v_j)$, where the numerator is applied coordinate-wise, but the denominator needs information from each input.
A typical simplification is to remove the biases, and also consider only two layers, written as

$$x \mapsto a^\intercal \sigma(Vx),$$

where $a \in \mathbb{R}^m$ and $V \in \mathbb{R}^{m \times d}$. ◇

There is significant focus in these notes on this two-layer setup because the higher-layer constructions often have strictly worse analysis (at present), in contrast with practice.

The decomposition of $\mathcal{R}$ is now given as follows. The starting point is that we would like to minimize $\mathcal{R}$, over data we have not seen, but only have access to $\widehat{\mathcal{R}}$, which uses data we have seen. For convenience define a mapping $\widehat{f} := F(\cdot; w)$, where $w \in \mathbb{R}^p$ is chosen via some algorithm aiming to minimize $\widehat{\mathcal{R}}$ (for example, gradient descent). To get a better handle on $\mathcal{R}$, let $\overline{f} := F(\cdot; \bar{w})$ denote an ideal mapping which does very well on $\mathcal{R}$ (and thus we can only handle $\overline{f}$ mathematically, it is not something our algorithms can access). Using $\overline{f}$, we can decomposose $\mathcal{R}(\widehat{f})$ as

$$\begin{aligned}
\mathcal{R}(\widehat{f}) &= \mathcal{R}(\widehat{f}) - \widehat{\mathcal{R}}(\widehat{f}) && (\textit{generalization}, \text{ Part III}) \\
&\quad + \widehat{\mathcal{R}}(\widehat{f}) - \widehat{\mathcal{R}}(\overline{f}) && (\textit{optimization}, \text{ Part II}) \\
&\quad + \widehat{\mathcal{R}}(\overline{f}) - \mathcal{R}(\overline{f}) && (\textit{generalization}, \text{ Part III}) \\
&\quad + \mathcal{R}(\overline{f}) && (\textit{approximation}, \text{ Part I}).
\end{aligned} \tag{1.4}$$

The main content of this book is organized around this decomposition; the idea is that we can show $\mathcal{R}(\widehat{f})$ is small by showing the four terms in the right hand side are all small. Unfortunately, there is evidence that this classical decomposition is in fact unable to analyze deep learning; as follows is concrete obstruction.

Remark 1.5 (Standard setup and *interpolation*)**.** A standard refinment to this setup is *statistical learning theory*: suppose future and past data are drawn IID from a common distribution, whereby $\widehat{\mathcal{R}}(\widehat{f}) \to \mathcal{R}(\widehat{f})$ and $\widehat{\mathcal{R}}(\overline{f}) \to \mathcal{R}(\overline{f})$ as $n \to \infty$ under a variety of regularity conditions, as discussed in Part III.
By contrast, as revealed empirically (Neyshabur et al., 2014; Zhang et al., 2017), it is often the case that $\widehat{\mathcal{R}}(\widehat{f}) = 0 \ll \mathcal{R}(\overline{f}) \approx \mathcal{R}(\widehat{f})$; ostensibly this is outside the purview

of eq. (1.4), which seems to need $\widehat{\mathcal{R}} \approx \mathcal{R}$ in order to be effective. This setting is sometimes called *interpolation* and has attracted considerable recent attention (Belkin et al., 2018). ◇

Another standard criticism is that the various components of eq. (1.4) or simply that concerns are too compartmentalized, and a tight analysis is impossible. *[todo 3/93]*

To circumvent these issues, the present perspective again focuses on data- and algorithimic-sensitve complexity terms, and on treating bounds as tools. For instance, as will be seen in *[todo 4/93]*

1.2 Status and purpose of deep learning theory

Bad news. The practical successes in **????** were not only produced without mathematical understanding of their components, they moreover were evolutions of a sequence of prior architectures and training practice which in turn were also produced without mathematical understanding. As such, is mathematical understanding necessary, and is there any way for this understanding to stop falling further and further behind?

One perspective is that theory must fundamentally change in its approach; this is already evidence by the main volume of works being *phenomenological*: identifying an empirical phenomenon and then producing mathematical analysis capturing some aspect, possibly after simplification, and not attempt to analyze the full end-to-end training in generality (i.e., showing $\mathcal{R}(\widehat{f})$ is small under practical conditions). This is in fact the perspective of Remark 1.5, which to date has only produced mathematical analyses for linear predictors with simplified (usually Gaussian) data. This approach too is without pitfalls, since for instance the applied community has stopped the practice of achiving $\widehat{\mathcal{R}}(\widehat{f}) \approx 0$ at all costs, and now is again achieving something closer to $\widehat{\mathcal{R}}(\widehat{f}) \approx \mathcal{R}(\widehat{f})$.

Good news? With the preceding in mind, why study deep learning theory?

1. Deep networks are deployed throughout the world now, and therefore it is essential — for safety and sanity — to understand them and their weaknesses. Mathematical analysis — even of the phenomological sort — can help with this, even without being a complete characterization.

2. Analyzing deep learning requires new mathematical insights, and this insights can feed back into other areas of machine learning and mathematics. *[todo 5/93]*

3. Math is fun.

Part I

Approximation

Chapter 2

Constructive approximation with shallow networks

Let's start by trying to make the definition of approximation precise. As in the introduction (and in particular eq. (1.4)), our goal is to select a predictor $\widehat{f} \in \mathcal{F}$ so that $\mathcal{R}(f)$ is small, where $\mathcal{R}$ measures performance on data we have not seen, whereas we only have access to $\widehat{\mathcal{R}}$, a performance criterion on data we have seen. Equation (1.4) decomposes $\mathcal{R}(f)$ into separate concerns, where optimization and generalization ensure that $\widehat{f}$ satisfies $\mathcal{R}(f) \approx \mathcal{R}(\overline{f})$ for some good choice $\overline{f} \in \mathcal{F}$ — for instance, e.g., $\mathcal{R}(f) \approx \inf_{f \in \mathcal{F}} \mathcal{R}(f) \approx \mathcal{R}(\bar{f})$ — whereas the goal of approximation is to argue that $\inf_{f \in \mathcal{F}} \mathcal{R}(f)$ can be made small.

What is $\mathcal{R}$? There are many ways to proceed when trying to make this problem precise; let's consider two natural choices.

1. **Perform well for some fixed future $\mathcal{R}$.** One possible goal is to show that certain network architectures perform well for certain specific well-structured choices of $\mathcal{R}$. For instance, suppose that future examples (x, y) are sampled from some distribution ν (not necessarily related to the distributiong of seen data!), and that performance on individual examples is measured with a nonnegative pairwise function ℓ, meaning the individual *loss* or *error* is $\ell(f(x), y)$, and overall $\mathcal{R}(f) = \mathbb{E}_{(x,y)\sim\nu} \ell(f(x), y)$. The goal in this setting would be to produce refined theorems that are as adapted to ℓ and ν as possible.

2. **Perform well given mere guidelines for the structure of $\mathcal{R}$.** Suppose that we only know a little bit about $\mathcal{R}$, for instance it is again an expectation, but we know nothing about the distribution, and also no specifics about the loss ℓ, but instead that it is merely Lipschitz (or satisfies some other basic regularity). In this situation, we instead ask for $\inf_{f \in \mathcal{F}} \mathcal{R}(f)$ to be close to some other $\inf_{g \in \mathcal{G}} \mathcal{R}(g)$, for instance if $\mathcal{G}$ denotes all continuous functions. As developed in the exercises *[todo 6/93]*, one can prove a variety of theorems of the form

$$\sup_{g \in \mathcal{G}} \inf_{f \in \mathcal{F}} \|f - g\| \approx 0 \qquad \Longrightarrow \qquad \inf_{f \in \mathcal{F}} \mathcal{R}(f) \approx \inf_{g \in \mathcal{G}} \mathcal{R}(g)$$

   for a variety of compatible regularity conditions on $\mathcal{F}$, $\mathcal{G}$, $\|\cdot\|$, and $\mathcal{R}$.

Historically, the community mainly focused on guarantees of the second type. Unfortunately, these guarantees intrinsically scale exponentially with dimension (von Luxburg and Bousquet, 2004), which makes them completely ineffective at capturing the good properties of deep networks, which shine in high-dimension settings. *[todo 7/93]*

What is $\mathcal{F}$? In this chapter and the subsequent one, we will restrict to feedforward networks and standard activations. But even beyond this, there are many important ways to restrict $\mathcal{F}$, with major consequences on how the approximation component fits together with optimization and generalization (and achieves the promised goal of an overall tight analysis).

1. **Models reached by gradient descent** (or some other standard training method). These methods do not consider every possible network of a fixed architecture, they consider a very complicated subset. Unfortunately, the community lacks refined understanding of this subset, though a few key properties are starting to emerge; e.g., it is possible it can be captured via a complicated norm centered around initialization.

2. **Models of low norm**, the aforementioned surrogate for "reached by gradient descent". Here there are already many questions, based on sensitivity to initialization, how to balance the norms of different layers, etc.

3. **All models of some fixed architecture**, meaning the weights can be arbitrary. This is the classical setup, and we'll cover it in parts of this chapter, but it can often seem loose or insensitive to data, and was a key part of the criticisms against the general learning-theoretic approach (Zhang et al., 2017).

2.1 Folklore elementary approximations

As a warm-up, we will establish two approximation results, one in $\mathbb{R}$ and one in $\mathbb{R}^d$, both of which are impractical but come with simple, intuitive proofs.

Proposition 2.1. *Suppose $g : \mathbb{R} \to \mathbb{R}$ is ρ-Lipschitz. For any $\epsilon > 0$, there exists a 2-layer network f with $\lceil \frac{\rho}{\epsilon} \rceil$ threshold nodes $z \mapsto \mathbf{1}[z \geq 0]$ so that $\sup_{x \in [0,1]} |f(x) - g(x)| \leq \epsilon$.*

The proof is intuitive: grid the space and stack bricks *[todo 8/93]*.

Proof. Define width $m := \lceil \frac{\rho}{\epsilon} \rceil$, biases and $b_i := (i-1)\epsilon/\rho$ for $i \in \{1, \dots, m\}$, outer weights

$$a_1 = g(b_1), \qquad a_i = g(b_i) - g(b_{i-1}),$$

and a 2-layer network $f(x) := \sum_{i=1}^{m} a_i \mathbf{1}[x_i \geq b_i]$. Then for any $x \in [0, 1]$, letting k be the

largest index so that $b_k \leq x$, then f is constant along $[b_k, x]$, and

$$\begin{aligned}
|g(x) - f(x)| &\leq |g(x) - g(b_k)| + |g(b_k) - f(b_k)| + |f(b_k) - f(x)| \\
&\leq \rho|x - b_k| + \left| g(b_k) - \sum_{i=0}^{k} a_i \right| + 0 \\
&\leq \rho(\epsilon/\rho) + \left| g(b_k) - g(b_0) - \sum_{i=1}^{k} (g(b_i) - g(b_{i-1})) \right| \\
&= \epsilon.
\end{aligned}$$

□

Remark 2.2. This construction has already lost something special and important: the gridding is *non-adaptive*, and in particular pays for flat regions. This is a weakness of the proof, and not inherent to neural network approximation. Notably, polynomial approximaton *does* pay for flat regions, and for instance approximating the absolute value requires $\mathcal{O}(1/\epsilon)$ degree polynomials but just two ReLUs *[todo 9/93]*. ◇

Now let's handle the multivariate case. We will replicate the univariate approach: we will increment function values when the target function changes. In the univariate case, we could "localize" function modifications, but in the multivariate case by default we will modify an entire halfspace at once. To get around this, we use an additional layer.

Theorem 2.3. *Let ρ-Lipschitz $g : \mathbb{R}^d \to \mathbb{R}$ and $\epsilon > 0$ be given. Then there exists a 3-layer network f with $\Omega\left(\left\lceil \rho/\epsilon \right\rceil^d\right)$ ReLU nodes so that $\int_{[0,1]^d} |f(x) - g(x)| dx \leq 2\epsilon$.*

Before giving the proof, a few remarks are in order.

Remark 2.4.

- This proof suffers explicitly from the *curse of dimension* (exponential dependence on d, which also appears in the aforementioned lower bounds (Luxburg and Bousquet 2004)). Note CIFAR has $d = 3072$; not only is this dependence catastrophic, but it makes the construction irrelevant in practice, where deep networks seem to shine particularly when data has high dimension. arbitrary continuous functions, and makes this irrelevant in practice.

- The construction uses three layers and not two. While a later proof will use only two, three layers entail interesting consequences. In particular, this construction has an inner construction with indicators on rectangles, and uses $4d + 1$ ReLUs in this step, but repeating this with one fewer layer seems to require exponentiall many ReLUs *[todo 10/93]*.

- The construction has not only large cardinality, but also large weight norm.

- The approximation is only on average (the L_1 distance), whereas what we want, in particular to approximate various distributions, is a supremum or uniform norm, as in *[todo 11/93]*.
- Recall the discussion of flat regions following the proof of **??**; unfortunately, in this multivariate case, it is not clear how to make an adaptive construction. *[todo 12/93]*

◇

Proof (Proof of Theorem 2.3)**.** *[todo 13/93]* Pick $k := \lceil \rho/(\epsilon\sqrt{d}) \rceil$ and $m := k^d$, and let $\mathcal{P} = (R_1, \dots, R_m)$ be a partition of $[0,1)^d$ into half-open cubes of side length $1/k$; for concreteness, suppose cube R_j has corners $u_j \in \mathbb{R}^d$ and $v_j \in \mathbb{R}^d$, meaning R_j is a product of d intervals of the form $\times_{i=1}^d [u_{j,i}, v_{j,i})$. Define $a_j := g(u_j)$, and consider the piecewise-constant function $h : \mathbb{R}^d \to \mathbb{R}$ defined as

$$h(x) := \sum_{j=1}^m a_j \mathbb{1}[x \in R_j];$$

by construction, for any $x \in [0,1)^d$, letting R_s denote the unique partition element with $x \in R_s$,

$$|h(x) - g(x)| = |a_s - g(x)| = |g(u_s) - g(x)| \le \rho \|u_s - x\| \le \epsilon.$$

As such, the proof is complete if for each R_j, we can construct a 3-layer ReLU network f_j with

$$\int_{[0,1)^d} \left| f_j(x) - \mathbb{1}[x \in R_j] \right| \mathrm{d}x \le \tau := \frac{\epsilon}{\sum_j |a_s|},$$

since the choice $f(x) := \sum_j a_j f_j(x)$ is also a 3-layer ReLU network, and satisfies

$$\begin{aligned}
\int_{[0,1)^d} |f(x) - g(x)| \,\mathrm{d}x &\le \int_{[0,1)^d} |f(x) - h(x)| \,\mathrm{d}x + \int_{[0,1)^d} |h(x) - g(x)| \,\mathrm{d}x \\
&\le \int_{[0,1)^d} \left| \sum_j a_j (f_j(x) - \mathbb{1}[x \in R_j]) \right| \mathrm{d}x + \epsilon \\
&\le \sum_j |a_j| \int_{[0,1)^d} \left| f_j(x) - \mathbb{1}[x \in R_j] \right| \mathrm{d}x + \epsilon \\
&\le \sum_j |a_j| \tau + \epsilon \le 2\epsilon.
\end{aligned}$$

(If $\sum_j |a_j| = 0$, then the constant function network $f(x) = 0\sigma(0^\intercal x)$ suffices.) As such, the remainder of the proof will show how to construct $(f_j)_{j=1}^m$.
Fix any j, and corresponding f_j and $R_j = [u_j, v_j)$; since j is fixed, the rest of the proof will drop j for convenience when unambiguous. Let $\gamma > 0$ be arbitrary, and for each

$i \in \{1, \dots, d\}$, define

$$f_{\gamma,i}(z) := \sigma\left(\frac{z-(u_i-\gamma)}{\gamma}\right) - \sigma\left(\frac{z-u_i}{\gamma}\right) - \sigma\left(\frac{z-v_i}{\gamma}\right) + \sigma\left(\frac{z-(v_i+\gamma)}{\gamma}\right)$$
$$\in \begin{cases} \{1\} & z \in [u_i, v_i], \\ \{0\} & z \notin [a_j - \gamma, b_j + \gamma], \\ [0,1] & \text{otherwise}, \end{cases}$$

and additionally

$$f_\gamma(x) := \sigma\left(\sum_j f_{\gamma,i}(x_i) - (d-1)\right).$$

(Note that a second hidden layer is crucial in this construction, it is not clear how to proceed without it, certainly with only $\mathcal{O}(d)$ nodes. Later proofs can use only a single hidden layer, but they are not constructive, and need $\mathcal{O}(d)$ nodes.) Note that $f_\gamma \approx \mathbb{1}_{R_j}$ as desired, specifically

$$f_\gamma(x) = \begin{cases} 1 & x \in R_j, \\ 0 & x \notin \times_i [u_i - \gamma, v_i + \gamma], \\ [0,1] & \text{otherwise}, \end{cases}$$

from which it follows that

$$\begin{aligned} \int_{[0,1)^d} \left|f_\gamma(x) - \mathbb{1}_{R_j}(x)\right| \mathrm{d}x &= \int_{R_j} |f_\gamma - \mathbf{1}_{R_j}| + \int_{\times_i [u_i-\gamma, v_i+\gamma] \setminus R_j} |f_\gamma - \mathbb{1}_{R_j}| + \int_{\mathbb{R}^d \setminus \times_i [u_i-\gamma, v_i+\gamma]} |f_\gamma - \mathbb{1}_{R_j}| \\ &\leq 0 + \prod_{i=1}^d (v_i - u_i + 2\gamma) - \prod_{i=1}^d (v_i - u_i) + 0 \\ &\leq \mathcal{O}(\gamma), \end{aligned}$$

which means we can ensure $\|\mathbb{1}_{R_j} - f_\gamma\|_1 \leq \frac{\epsilon}{\sum_i |\alpha_i|}$ by choosing sufficiently small γ, which completes the proof. □

2.2 Universal approximation with two layers

Theorem 2.3 had a few weaknesses: it used average distance (L_1 and not supremum/uniform norm), a specific activation, and three layers. This section will present a classical *universal approximation theorem* which resolves all issues.

Recall that the proof of Theorem 2.3 constructed localized bumps via the product

$$x \mapsto \prod_{i=1}^d \mathbb{1}[x \geq u_i] \cdot \mathbb{1}[x < v_i];$$

as such, it seems that multiplication is a useful operation. The proof scheme here, based

on an idea from (Hornik et al., 1989), will invoke the *Stone-Weierstrass* theorem, which establishes that polynomial-like classes of functions are universal approximators. *[todo 14/93]*

Theorem 2.5 (Universal approximation)**.** *Suppose $\sigma : \mathbb{R} \to \mathbb{R}$ is continuous and not a polynomial. Then for any continuous function $g : \mathbb{R}^d \to \mathbb{R}$ and any ϵ, there exists a 2-layer biased network $f : \mathbb{R}^d \to \mathbb{R}$ using σ nodes with $|f(x) - g(x)| \leq \epsilon$ for all $x \in [0,1]^d$.*

Before discussing the proof, a variety of remarks are in order.

Remark 2.6.

- **Necessity.** We can only approximate along a compact set: for instance, we need infinitely many ReLUs to approximate $r \mapsto \sin(r)$ uniformly over $\mathbb{R}$. We need σ to be not a polynomial: if it is a polynomial of some fixed degree k, then $x \mapsto a^\intercal \sigma(Vx+b)$ is also a k-degree polynomial, which is inadequate (e.g., it can't uniformly approximate $k+1$ degree polynomials). We need two layers: if we have only $x \mapsto \sigma(v^\intercal x + b)$ and σ is a ReLU, then we can not approximate a function which is not monotone along v.
- **The name "universal approximation".** This goes back to our discussion at the start of the chapter: as in exercises *[todo 15/93]*, by approximating continuous functions uniformally, we can ensure $\inf_{f \in \mathcal{F}} \mathcal{R}(f)$ is small for a wide variety of definitions of $\mathcal{R}$.

◇

The proof will proceed in two stages: first we will quickly check the claim for exponential activations, and then reduce other activations to exponentials.

Lemma 2.7. *Given any continuous $g : \mathbb{R}^d \to \mathbb{R}$ and $\epsilon > 0$, there exists a 2-layer network $f : \mathbb{R}^d \to \mathbb{R}$ with $\sigma(r) = \exp(r)$ so that $|f(x) - g(x)| \leq \epsilon$ for all $x \in [0,1]^d$.*

Proof. As mentioned above, the proof proceeds via opaque invocation of a heavyweight tool: the Stone-Weierstrass theorem (Folland, 1999, Theorem 4.45). To this end, define our function class $\mathcal{F}$ as

$$\mathcal{F} := \left\{ x \mapsto a^\intercal \exp(Vx) : m \geq 0, a \in \mathbb{R}^m, V \in \mathbb{R}^{m \times d} \right\}.$$

To apply Stone-Weierstrass, it suffices to check four conditions, which completes the proof.

1. Every $f \in \mathcal{F}$ is continuous: this is direct since exp is continuous, the linear mappings are continuous, and composition preserves continuity.
2. For every $x \in [0,1]^d$, there exists $f \in \mathcal{F}$ with $f(x) \neq 0$: for this one, it suffices to pick $\left(x \mapsto \exp(0^\intercal x) = 1\right) \in \mathcal{F}$.
3. $\mathcal{F}$ separates points, meaning for every $x \neq x'$, there exists $f \in \mathcal{F}$ with $f(x) \neq f(x')$; for this it suffices to define

$$f(z) := \exp(\langle z - x', x - x' \rangle) = \exp(\langle -x', x - x' \rangle) \exp(\langle z, x - x' \rangle) \in \mathcal{F},$$

which satisfies $f(x) = 1 \neq \exp(\|x - x'\|^2) = f(x')$.

4. $\mathcal{F}$ is closed under vector space operations and product: for this let $b, c \in \mathbb{R}$ and $f(x) = a^\intercal \exp(Vx) \in \mathcal{F}$ and $h(x) = u^\intercal \exp(Wx)$ be given, and note

$$ba^\intercal \exp(Vx) + cu^\intercal \exp(Wx) = \begin{bmatrix} ba \\ cu \end{bmatrix} \exp\left(\begin{bmatrix} V \\ W \end{bmatrix} x \right) \in \mathcal{F},$$

whereas for multiplication

$$\begin{aligned} \left(a^\intercal \exp(Vx)\right)\left(u^\intercal \exp(Wx)\right) &= \left(\sum_{j=1}^m a_j \exp(v_j^\intercal x)\right)\left(\sum_{i=1}^m u_i \exp(w_i^\intercal x)\right) \\ &= \sum_{j=1}^m \sum_{i=1}^n a_j u_i \exp\left((v_j + w_i)^\intercal x\right) \in \mathcal{F}. \end{aligned}$$

□

Proof (Proof of Theorem 2.5)**.** The proof proceeds in two steps.

1. Thanks to Lemma 2.7, there exists $h(x) := a^\intercal \exp(Vx)$ such that $|h(x) - g(x)| \leq \epsilon/2$ for every $x \in [0,1]^d$.

2. Use *[todo 16/93]* to obtain $p(r) := \sum_i u_i \sigma(w_i x + b_i)$ with $|p(r) - \exp(r)| \leq \epsilon/(1 + 2\sum_j |a_j|)$ for $|r| \leq \max_j \|v_j\|_1$. Then

$$f(x) := \sum_{j=1}^m a_j p(v_j^\intercal x) = \sum_{j=1}^m \sum_{i=1}^n a_j u_i \sigma(w_i v_j^\intercal x + b_i)$$

is a 2-layer biased network with σ activations, and satisfies for any $x \in [0,1]^d$

$$|f(x) - h(x)| \leq \sum_{j=1}^m |a_j| \cdot \left|\exp(v_j^\intercal x) - p(v_j^\intercal x)\right| \leq \frac{\epsilon}{2}.$$

Combining the two steps, $|f(x) - g(x)| \leq |f(x) - h(x)| + |h(x) - g(x)| \leq \epsilon$. □

2.3 Infinite-width networks, Fourier transforms, and the Barron norm

This section studies infinite-width representations.

- They have become popular again recently, and thus should be taught.

- They typically allow the approximated function to be written *with equality*, further helping alleviate and study looseness in the approximation approaches.
- They can be converted to finite-width networks via sampling (cf. Section 2.4).
- They sometimes exhibit slight data adaptivity.

As a warm-up, let's produce the infinite-width analog to Proposition 2.1

Proposition 2.8. *Suppose $g : \mathbb{R} \to \mathbb{R}$ is continuously differentiable, and $g(0) = 0$. If $x \in [0, 1]$, then $g(x) = \int_0^1 g'(b)\mathbb{1}[x \geq b]\,\mathrm{d}b$.*

Proof. By FTC and $g(0) = 0$ and $x \in [0, 1]$,

$$g(x) = g(0) + \int_0^x g'(b)\mathrm{d}b = 0 + \int_0^1 \mathbf{1}[x \geq b]g'(b)\,\mathrm{d}b.$$

□

Let's compare this closely to the grid-based univariate approximation bound from Proposition 2.1.

- It may seem Proposition 2.8 has a much shorter proof, but it invokes FTC, and in fact, the construction of the Riemann integral is similar to the gridding in Proposition 2.1, so they are in fact nearly the same proof.
- As will be argued shortly, the "complexity measure" corresponding to Proposition 2.8 is $\left(\int_0^1 |g'(b)|\,\mathrm{d}b\right)^2/\epsilon$, and corresponds to ρ/ϵ, the number of nodes from Proposition 2.1. The key difference between these two is that the integral *is* sensitive to flat regions, it only pays for the variation of the function. *[todo 17/93]*
- Here is a quick calculation on how to do the sampling. Define a normalization constant $Z := \int_0^1 |g'(b)|\,\mathrm{d}b$, and note $|g'(b)|/Z$ defines a probability density over $[0, 1]$. Sampling b_j from this distribution and defining $a_j := Z\mathrm{sgn}(g'(b_j))/m$, we can define a network

  $$f(x) := \sum_j a_j \mathbb{1}[x \geq b_j],$$

  which is an unbiased estimate of g:

  $$\mathbb{E}f(x) = \sum_{j=1}^m \mathbb{E}_{b_j} a_j \mathbb{1}[x \geq b_j] = m\int_0^1 \frac{Z\mathrm{sgn}(g'(b))}{m}\mathbb{1}[x \geq b]\frac{|g'(b)|}{Z}\,\mathrm{d}b = \int_0^1 g'(b)\mathbb{1}[x \geq b] = g(x).$$

  Using the tools in Section 2.4 (cf. *[todo 18/93]*) gives the sampling estimate.

Now let's handle the multivariate case, which we'll do with Fourier transforms. As with Proposition 2.8, our goal is to rewrite the network with equality, and give an estimate of its mass (the integral of the absolute value of its weights).

Since this construction will be the only place in these notes where complex numbers and Fourier transforms appear, the following lemma captures all properties that will be used.

Lemma 2.9 (Basic Fourier and complex properties (Folland, 1999))**.** *Throughout, let $g : \mathbb{R}^d \to \mathbb{R}$ with $\int_{\mathbb{R}^d} |g| < \infty$ be given (henceforth "g is integrable"), and let $|\cdot|$ denote the absolute value of a complex number, meaning $|b + ic| = \sqrt{b^2 + c^2}$, and define the Fourier transform $\widetilde{g} : \mathbb{R}^d \to \mathbb{C}$ of g as*

$$\widetilde{g}(w) = \int_{\mathbb{R}^d} \exp(-2\pi i w^\intercal x) g(x) \, \mathrm{d}x.$$

1. *(Inversion.) If $\int_{\mathbb{R}^d} |\widetilde{g}(w)| \, \mathrm{d}w < \infty$, then*

$$g(x) = \int_{\mathbb{R}^d} \exp(2\pi i w^\intercal x) \widetilde{g}(w) \, \mathrm{d}w.$$

2. *(Derivatives.) Given w, then $2\pi \|w\| \cdot |\widetilde{g}(w)| = \|\widetilde{\nabla g}\|$.*

3. *(Euler formula.) If $r \in \mathbb{R}$, then $\exp(ir) = \cos(r) + i \sin(r)$.*

4. *(Polar decomposition.) Given integrable $h : \mathbb{R}^d \to \mathbb{C}$, there exists an integrable function $\theta_g : \mathbb{R}^d \to \mathbb{C}$ with $|\theta_g| \leq 1$ and $g(x) = |g(x)| \exp(2\pi i \theta_g(x))$ almost everywhere.*

5. *(Real parts and integration.) Let $\mathrm{Re}(b + ic) = b$ denote the real part of a complex number. Then for a complex-valued function $h : \mathbb{R}^d \to \mathbb{C}$ which is integrable,*

$$\mathrm{Re}\left[\int_{\mathbb{R}^d} h(x) \, \mathrm{d}x\right] = \int_{\mathbb{R}^d} \mathrm{Re}\left[h(x)\right] \mathrm{d}x.$$

Rather than giving the statement and continuing with its proof, it will be proved first. To start, consider the Fourier inversion formula from Lemma 2.9: if g and $\widetilde{g}$ are integrable, then

$$g(x) = \int \exp(2\pi i w^\intercal x) \widetilde{g}(w) \, \mathrm{d}w;$$

this is *already* an infinite width network, albeit using non-standard, complex activations. Our approach will simply be to rewrite these complex activations with thresholds (meaning $\sigma(z) := \mathbb{1}[z \geq 0]$).

1. **Removing complex numbers.** Since g is real-valued, we can make the integral

real-valued, and use the polar decomposition of $\widetilde{g}$ to isolate all complex terms:

$$\begin{aligned} g(x) &= \operatorname{Re}\left[g(x)\right] \\ &= \operatorname{Re}\left[\int \exp(2\pi i w^\mathsf{T} x)\widetilde{g}(w)\,\mathrm{d}w\right] \\ &= \int \operatorname{Re}\left[\exp(2\pi i w^\mathsf{T} x)|\widetilde{g}(w)|\exp\left(2\pi i\theta_{\widetilde{g}}(w)\right)\right]\mathrm{d}w \\ &= \int \operatorname{Re}\left[\exp\left(2\pi i(w^\mathsf{T} x + \theta_{\widetilde{g}}(w)\right)|\widetilde{g}(w)|\right]\mathrm{d}w \\ &= \int |\widetilde{g}(w)|\operatorname{Re}\left[\cos(2\pi(w^\mathsf{T} x + \theta_{\widetilde{g}}(w)) + i\sin(2\pi(w^\mathsf{T} x + \theta_{\widetilde{g}}(w))|\widetilde{g}(w)|\right]\mathrm{d}w \\ &= \int |\widetilde{g}(w)|\cos\left((2\pi(w^\mathsf{T} x + \theta_{\widetilde{g}}(w)\right)\mathrm{d}w, \end{aligned} \tag{2.10}$$

which is an infinite-width network with real activations and weights, but still using a nonstandard activation, cos.

2. **Introducing thresholds.** Rewriting cos is now a univariate approximation question, which we can handle as we did before with FTC in Proposition 2.8, albeit with some extra effort since the integration domain is not necessarily nonnegative. Focusing on the integrand within eq. (2.10),

$$\begin{aligned} &\cos\left((2\pi(w^\mathsf{T} x + \theta_{\widetilde{g}}(w)\right) - \cos\left(2\pi\theta_{\widetilde{g}}(w)\right) \\ &\quad = -2\pi\int_0^{w^\mathsf{T} x} \sin\left(2\pi(b + \theta_{\widetilde{g}}(w))\right)\mathrm{d}b \\ &\quad = -2\pi\int_0^{\|w\|} \sin\left(2\pi(b + \theta_{\widetilde{g}}(w))\right)\mathbb{1}[w^\mathsf{T} x \geq b]\,\mathrm{d}b \\ &\qquad + 2\pi\int_{-\|w\|}^{0} \sin\left(2\pi(b + \theta_{\widetilde{g}}(w))\right)\mathbb{1}[w^\mathsf{T} x \leq b]\,\mathrm{d}b \\ &\quad = 2\pi\int_0^{\|w\|} \left[\sin\left(2\pi(-b + \theta_{\widetilde{g}}(-w))\right) - \sin\left(2\pi(b + \theta_{\widetilde{g}}(w))\right)\right]\mathbb{1}[w^\mathsf{T} x \geq b]\,\mathrm{d}b. \end{aligned} \tag{2.11}$$

We are done: combining the removal of complex numbers from eq. (2.10) with the replacement of cos with threshold activations in σ gives a way to rewrite g as an infinite-width threshold activation network, summarized as follows.

Theorem 2.12. *Suppose $g, \widetilde{g} \in L_1$ and $g(0) = 0$, and define a parameter density*

$$q(w,b) := 2\pi|\widetilde{g}(w)|\left(\sin\left(2\pi(-b + \theta_{\widetilde{g}}(-w)\right) - \sin\left(2\pi(b + \theta_{\widetilde{g}}(w))\right)\right)\mathbb{1}[0 \leq b \leq \|w\|].$$

Then

$$g(x) = \iint q(w,b)\mathbb{1}[w^\mathsf{T} x \geq b]\,\mathrm{d}b\,\mathrm{d}w.$$

and moreover $\iint |q(w,b)|\,\mathrm{d}b\,\mathrm{d}w \le 2\int \|\widetilde{\nabla} g\|\,\mathrm{d}w$.

Remark 2.13. *[todo 19/93]*
[todo 20/93] ◇

Proof. Let $g, \widetilde{g}, q$ be as in the statement. The key equality $g(x) = \iint q(w,b)\mathbb{1}[w^\mathsf{T}x \ge b]\,\mathrm{d}b\,\mathrm{d}w$ is simply the combination of eqs. (2.10) and (2.11) after unpacking the definition of q. Lastly, to calculate the mass of q, using Lemma 2.9 and $|\sin| \le 1$,

$$\begin{aligned}\iint |q(w,b)|\,\mathrm{d}b\,\mathrm{d}w &\le 2\pi \int\int_0^{\|w\|} 2|\widetilde{g}(w)|\,\mathrm{d}b\,\mathrm{d}w \\ &= 4\pi \int \|w\| \cdot |\widetilde{g}(w)|\,\mathrm{d}w \\ &= 2\int \|\widetilde{\nabla} g\|\,\mathrm{d}w.\end{aligned}$$

□

To close, here are a few estimates for $\int \|\widetilde{\nabla} g(w)\|\,\mathrm{d}w$.

- **Gaussians.** Using standard Fourier transform calculations (Folland, 1999, e.g., Proposition 8.24)

$$g(x) = (2\pi\sigma^2)^{d/2}\exp(-\frac{\|x\|^2}{2\sigma^2}) \qquad \Longrightarrow \qquad \widetilde{g}(w) = \exp(-2\pi^2\sigma^2\|w\|^2),$$

  meaning tg is an unnormalized Gaussian with variance $(4\pi^2\sigma^2)^{-1}$. Using normalization $Z := (2\pi\sigma^2)^{-d/2}$ and Holder gives

$$\begin{aligned}\int \|w\| \cdot |\widetilde{g}(w)|\,\mathrm{d}w &= Z\int Z^{-1}\|w\| \cdot |\widetilde{g}(w)|\,\mathrm{d}w \\ &\le Z\left(\int Z^{-1}\|w\|^2|\widetilde{g}(w)|\,\mathrm{d}w\right)^{1/2} \\ &= Z\left(\frac{d}{4\pi^2\sigma^2}\right)^{1/2} = \frac{\sqrt{d}}{\sqrt{2\pi}(2\pi\sigma^2)^{(d+1)/2}}.\end{aligned}$$

  Consequently, if $2\pi\sigma^2 \ge 1$, then $\int \left\|\widetilde{\nabla} g(w)\right\|\mathrm{d}w = \mathcal{O}(\sqrt{d})$. On the other hand, general radial functions have exponential $\|\widetilde{\nabla} g(w)\|$ (Barron, 1993, Comment IX.9); this is circumvented here since $\|x\| \le 1$ and hence the Gaussian is quite flat.

- Further brief example $\int \left\|\widehat{\nabla} f(w)\right\|\mathrm{d}w$ calculations:
  - A few more from (Barron, 1993, Section IX): radial functions (IX.9), compositions with polynomials (IX.12) and analytic functions (IX.13), functions with $\mathcal{O}(d)$ bounded derivatives (IX.15).

- Barron also gives a lower bound for a specific set of functions which is exponential in dimension.
- Further comments on Barron's constructions can be found in (Lee et al., 2017).
- General continuous functions can fail to satisfy $\int \left\| \widetilde{\nabla} g(w) \right\| \mathrm{d}w < \infty$, but we can first convolve them with Gaussians and sample the resulting nearby function; this approach, along with a Barron theorem using ReLUs, can be found in (Ji et al., 2020). *[todo 21/93]*

2.4 Sampling from infinite-width networks

To close this chapter, this section gives a rather technical approach to sampling a finite-width network from an infinite-width one. Though Section 2.3 made the task sound like sampling from a continuous density, in general the densities will not be continuous; for instance, if we do not require $g(0)$, then we should account for $g(0)$ within the weight distribution. Lastly, as a further technical point, the tools in this section will not yield uniform norms, but rather squared L_2 norms, but similar techniques can also yield uniform norm guarantees *[todo 22/93]* *[todo 23/93]*

Now we will show how to obtain a finite-width representation from an infinite-width representation. Coarsely, given a representation $\int \sigma(w^\mathsf{T} x) g(w) \mathrm{d}w$, we can form an estimate

$$\sum_{j=1}^{m} s_j \tilde{\sigma}(w_j^\mathsf{T} x), \qquad \text{where } s_j \in \pm 1,\ \tilde{\sigma}(z) = \sigma(z) \int |g(w)| \mathrm{d}w,$$

by sampling $w_j \sim |g(w)| / \int |g(w)| \mathrm{d}w$, and letting $s_j := \operatorname{sgn}(g(w_j))$, meaning the sign corresponding to whether w fell in a negative or positive region of g. In expectation, this estimate is equal to the original function.

Here we will give a more general construction where the integral is not necessarily over the Lebesgue measure, which is useful when it has discrete parts and low-dimensional sets. This section will follow the same approach as (**?**), namely using Maurey's sampling method Lemma 2.16, which gives an L_2 error; it is possible to use these techniques to obtain an L_∞ error via the "co-VC dimension technique" (Gurvits and Koiran, 1995), but this is not pursued here.

To build this up, first let us formally define these infinite-width networks and their mass.

Definition 2.14. n *infinite-width shallow network* is characterized by a *signed measure* ν over weight vectors in $\mathbb{R}^p$:

$$x \mapsto \int \sigma(w^\mathsf{T} x) \mathrm{d}\nu(w).$$

The *mass* of ν is the total positive and negative weight mass assigned by ν: $|\nu|(\mathbb{R}^p) = \nu_-(\mathbb{R}^p) + \nu_+(\mathbb{R}^p)$. ◇

Remark 2.15. We can connect this to the initial discussion of $\int \sigma(w^\intercal x) g(w) \mathrm{d}w$ by defining a signed measure ν via $\mathrm{d}\nu = g$, and the mass is once again $|\nu|(\mathbb{R}^p) = \int |g(w)| \mathrm{d}w$, and the positive and negative parts ν_- and ν_+ are simply the regions where g is respectively negative (or just non-positive) and positive.
In the case of general measures, a decomposition into ν_- and ν_+ is guaranteed to exist (Jordan decomposition, Folland 1999), and is unique up to null sets.
The notation here uses $\mathbb{R}^p$ not $\mathbb{R}^d$ since we might bake in biases and other feature mappings. ◇

To develop sampling bounds, first we give the classical general Maurey sampling technique, which is stated as sampling in Hilbert spaces.

Suppose $X = \mathbb{E}\, V$, where r.v. V is supported on a set S. A natural way to "simplify" X is to instead consider $\hat{X} := \frac{1}{k}\sum_{i=1}^k V_i$, where $(V_1, \ldots, V_k)$ are sampled iid. We want to argue $\hat{X} \approx X$; since we're in a Hilbert space, we'll try to make the Hilbert norm $\|X - \hat{X}\|$ small.

[todo 24/93]

Lemma 2.16 (Maurey)**.** *Let $X = \mathbb{E}\, V$ be given, with V supported on S, and let $(V_1, \ldots, V_k)$ be iid draws from the same distribution. Then*

$$\mathop{\mathbb{E}}_{V_1,\ldots,V_k} \left\| X - \frac{1}{k}\sum_i V_i \right\|^2 \leq \frac{\mathbb{E}\,\|V\|^2}{k} \leq \frac{\sup_{U\in S}\|U\|^2}{k},$$

and moreover there exist $(U_1, \ldots, U_k)$ in S so that

$$\left\| X - \frac{1}{k}\sum_i U_i \right\|^2 \leq \mathop{\mathbb{E}}_{V_1,\ldots,V_k} \left\| X - \frac{1}{k}\sum_i V_i \right\|^2.$$

After proving this, we'll get a corollary for sampling from networks. This lemma is widely applicable; e.g., we'll use it for generalization too. It was first used used for neural networks by (Barron, 1993) and (Jones, 1992), and attributed to Maurey by (Pisier, 1980).

Proof. Let $(V_1, \dots, V_k)$ be IID as stated. Then

$$\begin{aligned}
&\mathop{\mathbb{E}}_{V_1,\dots,V_k} \left\| X - \frac{1}{k}\sum_i V_i \right\|^2 \\
&= \mathop{\mathbb{E}}_{V_1,\dots,V_k} \left\| \frac{1}{k}\sum_i (V_i - X) \right\|^2 \\
&= \mathop{\mathbb{E}}_{V_1,\dots,V_k} \frac{1}{k^2} \left[\sum_i \|V_i - X\|^2 + \sum_{i\neq j} \langle V_i - X, V_j - X \rangle \right] \\
&= \mathop{\mathbb{E}}_{V} \frac{1}{k} \|V - X\|^2 \\
&= \mathop{\mathbb{E}}_{V} \frac{1}{k} \left(\|V\|^2 - \|X\|^2 \right) \\
&\leq \mathop{\mathbb{E}}_{V} \frac{1}{k} \|V\|^2 \leq \sup_{U\in S} \frac{1}{k} \|U\|^2 .
\end{aligned}$$

To conclude, there must exist $(U_1, \dots, U_k)$ in S so that $\left\|X - k^{-1}\sum_i U_i\right\|^2 \leq \mathbb{E}_{V_1,\dots,V_k} \left\|X - k^{-1}\sum_i V_i\right\|^2$. ("Probabilistic method".) □

Now let's apply this to infinite-width networks in the generality of Definition 2.14. We have two issues to resolve.

- **Issue 1:** what is the appropriate Hilbert space?
  - **Answer:** We'll use $\langle f, g \rangle = \int f(x)g(x)\mathrm{d}P(x)$ for some probability measure P on x, so $\|f\|^2_{L_2(P)} = \int f(x)^2 \mathrm{d}P(x)$.
- **Issue 2:** our "distribution" on weights is not a probability!
  - **Example:** consider $x \in [0,1]$ and $\sin(2\pi x) = \int_0^1 \mathbf{1}[x \geq b] 2\pi \cos(2\pi b)\mathrm{d}b$. There are two issues: $\int_0^1 |2\pi\cos(2\pi b)|\mathrm{d}b \neq 1$, and $\cos(2\pi b)$ takes on negative and positive values.
  - **Answer:** we'll correct this in detail shortly, but here is a sketch; recall also the discussion in Definition 2.14 of splitting a measure into positive and negative parts. First, we introduce a fake parameter $s \in \{\pm 1\}$ and multiply $\mathbf{1}[x \geq b]$ with it, simulating positive and negative weights with only positive weights; now our distribution is on pairs (s, b). Secondly, we'll normalize everything by $\int_0^1 |2\pi\cos(2\pi b)|\mathrm{d}b$.

Let's write a generalized shallow network as $x \mapsto \int g(x; w)\mathrm{d}\mu(w)$, where μ is a nonzero signed measure over some abstract parameter space $\mathbb{R}^p$. E.g., $w = (a, b, v)$ and $g(x; w) = a\sigma(v^\mathsf{T} x + b)$.

- Decompose $\mu = \mu_+ - \mu_-$ into nonnegative measures $\mu_\pm$ with disjoint support (this is the *Jordan decomposition* (Folland, 1999), which was mentioned in Definition 2.14.

- For nonnegative measures, define total mass $\|\mu_\pm\|_1 = \mu_\pm(\mathbb{R}^p)$, and otherwise $\|\mu\|_1 = \|\mu_+\|_1 + \|\mu_-\|_1$.

- Define $\widetilde{\mu}$ to sample $s \in \{\pm 1\}$ with $\Pr[s = +1] = \frac{\|\mu_+\|_1}{\|\mu\|_1}$, and then sample $g \sim \frac{\mu_s}{\|\mu_s\|_1} =: \tilde{\mu}_s$, and output $\tilde{g}(\cdot; w, s) = s\|\mu\|_1 g(\cdot; w)$.

This sampling procedure has the correct mean:

$$\begin{aligned}
\int g(x;w)\mathrm{d}\mu(w) &= \int g(x;w)\mathrm{d}\mu_+(w) - \int g(x;w)\mathrm{d}\mu_-(w) \\
&= \|\mu_+\|_1 \mathop{\mathbb{E}}_{\tilde{\mu}_+} g(x;w) - \|\mu_-\|_1 \mathop{\mathbb{E}}_{\tilde{\mu}_-} g(x;w) \\
&= \|\mu\|_1 \left[\Pr_{\widetilde{\mu}}[s=+1] \mathop{\mathbb{E}}_{\tilde{\mu}_+} g(x;w) - \Pr_{\widetilde{\mu}}[s=-1] \mathop{\mathbb{E}}_{\tilde{\mu}_-} g(x;w) \right] = \mathop{\mathbb{E}}_{\widetilde{\mu}} \tilde{g}(x;w,s).
\end{aligned}$$

Lemma 2.17 (Maurey for signed measures)**.** *Let μ denote a nonzero signed measure supported on $S \subseteq \mathbb{R}^p$, and write $g(x) := \int g(x;w)d\mu(w)$. Let $(\tilde{w}_1, \ldots, \tilde{w}_k)$ be IID draws from the corresponding $\widetilde{\mu}$, and let P be a probability measure on x. Then*

$$\begin{aligned}
\mathop{\mathbb{E}}_{\tilde{w}_1,\ldots,\tilde{w}_k} \left\| g - \frac{1}{k}\sum_i \tilde{g}(\cdot;\tilde{w}_i) \right\|_{L_2(P)}^2 &\leq \frac{\mathbb{E}\,\|\tilde{g}(\cdot;\tilde{w})\|_{L_2(P)}^2}{k} \\
&\leq \frac{\|\mu\|_1^2 \sup_{w\in S} \|g(\cdot;w)\|_{L_2(P)}^2}{k},
\end{aligned}$$

and moreover there exist $(w_1, \ldots, w_k)$ in S and $s \in \{\pm 1\}^m$ with

$$\left\| g - \frac{1}{k}\sum_i \tilde{g}(\cdot;w_i,s_i) \right\|_{L_2(P)}^2 \leq \mathop{\mathbb{E}}_{\tilde{w}_1,\ldots,\tilde{w}_k} \left\| g - \frac{1}{k}\sum_i \tilde{g}(\cdot;\tilde{w}_i) \right\|_{L_2(P)}^2.$$

Proof. By the mean calculation we did earlier, $g = \mathbb{E}_{\widetilde{\mu}} \|\mu\| s g_w = \mathbb{E}_{\widetilde{\mu}} \tilde{g}$, so by the regular

Maurey applied to $\widetilde{\mu}$ and Hilbert space $L_2(P)$ (i.e., writing $V := \tilde{g}$ and $g = \mathbb{E}\, V$),

$$\begin{aligned}\mathop{\mathbb{E}}_{\tilde{w}_1,\ldots,\tilde{w}_k} \left\| g - \frac{1}{k}\sum_i \tilde{g}(\cdot;\tilde{w}_i) \right\|_{L_2(P)}^2 &\leq \frac{\mathbb{E}\,\|\tilde{g}(\cdot;\tilde{w})\|_{L_2(P)}^2}{k} \\ &\leq \frac{\sup_{s\in\{\pm 1\}}\sup_{w\in\mathcal{W}}\left\{\|\mu\|_1 s g(\cdot;w)\right\}_{L_2(P)}^2}{k}. \\ &\leq \frac{\|\mu\|_1^2 \sup_{w\in S}\|g(\cdot;w)\|_{L_2(P)}^2}{k},\end{aligned}$$

and the existence of the fixed (w_i, s_i) is also from Maurey. □

Example 2.18 (Various infinite-width sampling bounds)**.** 1. Suppose $x \in [0,1]$ and f is differentiable. Using our old univariate calculation,

$$f(x) - f(0) = \int_0^1 \mathbf{1}[x \geq b] f'(b)\mathrm{d}b.$$

Let μ denote $f'(b)\mathrm{d}b$; then a sample $((b_i, s_i))_{i=1}^k$ from $\widetilde{\mu}$ satisfies

$$\begin{aligned}\left\| f(\cdot) - f(0) - \frac{\|\mu\|_1}{k}\sum_i s_i \mathbf{1}[\cdot \geq b_i] \right\|_{L_2(P)}^2 &\leq \frac{\|\mu\|_1^2 \sup_{b\in[0,1]}\|\mathbf{1}[\cdot \geq b]\|_{L_2(P)}^2}{k} \\ &= \frac{1}{k}\left(\int_0^1 |f'(b)|\mathrm{d}b\right)^2.\end{aligned}$$

2. Now consider the Fourier representation via Barron's theorem:

$$\begin{aligned}f(x) - f(0) = &-2\pi \int\!\!\int_0^{\|w\|} \mathbf{1}[w^\mathsf{T} x - b \geq 0]\left[\sin(2\pi b + 2\pi\theta(w))|\hat{f}(w)|\right]\mathrm{d}b\mathrm{d}w \\ &+ 2\pi \int\!\!\int_{-\|w\|}^0 \mathbf{1}[-w^\mathsf{T} x + b \geq 0]\left[\sin(2\pi b + 2\pi\theta(w))|\hat{f}(w)|\right]\mathrm{d}b\mathrm{d}w,\end{aligned}$$

and also our calculation that the corresponding measure μ on thresholds has $\|\mu\|_1 \leq 2\|\widehat{\nabla f}(w)\|$. Then Maurey's lemma implies that there exist $((w_i, b_i, s_i))_{i=1}^m$ such that, for any probability measure P support on $\|x\| \leq 1$,

$$\begin{aligned}\left\| f(\cdot) - f(0) - \frac{\|\mu\|_1}{k}\sum_i s_i \mathbf{1}[\langle w_i, \cdot\rangle \geq b_i] \right\|_{L_2(P)}^2 &\leq \frac{\|\mu\|_1^2 \sup_{w,b}\|\mathbf{1}[\langle w, \cdot\rangle \geq b]\|_{L_2(P)}^2}{k} \\ &\leq \frac{4\|\widehat{\nabla f}(w)\|^2}{k}.\end{aligned}$$

◇

2.5 Bibliographic notes

[todo 25/93]
- *[todo 26/93]*
- *[todo 27/93]*
- *[todo 28/93]*
- *[todo 29/93]*
- *[todo 30/93]*
- *[todo 31/93]*
- *[todo 32/93]*

2.6 Exercises

2.6.1 Problems

[todo 33/93]
- *[todo 34/93]*
- *[todo 35/93]*
- *[todo 36/93]*

2.6.2 Research questions

Research question 2.1 (Data adaptivity)**.** [todo 37/93]
The results of this chapter typically need a network size scaling exponentially with dimension, and for a few of the upper bounds, there is a matching lower bound (e.g., when approximating continuous functions (von Luxburg and Bousquet, 2004). By contrast, deep networks seem to flourish (especially when compared to other methods) in settings where the dimension is in the thousands or millions. This research question is about building an approximation theory which addresses this gap; it has both an applied component and a theoretical component. Here are a variety of uncoordinated remarks.

- *Firstly, there is an applied and theoretical component since, due to the lower bounds, necessarily the approximated functions must be restricted in some way which as a subset contains networks similar to those appearing in practice.*
- *It may seem that moving from a worst-case derivative bound (the Lipschitz constant) in Proposition 2.1 to an average-case derivative bound (bounded variation) in Proposition 2.8 constitutes a nice appearance of adaptivity; unfortunately, the issue is obscured since this case is univariate. For instance, a generalization of this allows one to consider functions whose various higher-order partial derivatives satisfy some norm bound; unfortunately, this setting is insufficient to remove exponential dependence on dimension (see for instance (Yarotsky, 2016) and things which cite it).*

- *Noting that approximating the indicator on a rectangle already potentially requires exponential width with two layers (Eldan and Shamir, 2015) but polynomial width with three layers, perhaps increasing depth (or changing the architecture in other ways) is crucial; unfortunately, that is the approach in the previous bullet and it is not enough, as the choice of target function class is still critical.*

- *There exist fixed small (even discrete) classes of functions (for instance, k-sparse parity, as discussed later in [todo 38/93]), for which it is already interesting and difficult to produce approximation bounds, and moreover these bounds suffice to further imply sample complexity bounds. While more modest than the goals of this section, perhaps they constitute a better starting point.*

- *Another approach is to be sensitive to the behavior of gradient descent, as in the next chapter.*

Chapter 3

Initialization and overparameterization

In Chapter 2, it was shown that various architectures can approximate continuous functions, where the approximation complexity was always related to the number of nodes which were needed (often exponential). The number of nodes is a poor way to measure approximation difficulty; for instance, the goal of approximating a single ReLU by another ReLU may seem silly from an approximation perspective, however it is an active area of research in optimization (**??**), with a variety of negative results (**?**).

What did the perspective in Chapter 2 miss? Though it is still too early to say definitively, it seems that standard gradient-based optimization methods prefer functions which satisfy two competing concerns:

- the chosen network has small norm;
- the chosen network is close to initialization.

The relationship of these two objectives is delicate: the first norm is implicitly measured against the origin, whereas the segment is measured against random initialization, which is large and not fully cleared out in standard initialization and training setups. Further differences between these two will be discussed in **??**.

The purpose of this chapter is to investigate the second point above, meaning properties of networks near initialization, an area which has seen immense research activity over the past few years. While the analyses in this section are generally falsified in practice — that is, the closeness needed far exceeds what is observed in experiments — still this perspective captures many phenomena which were missing in classical analyses , and also predict other interesting behaviors, for instance the benefits of large width (that is, *overparameterization*). The central new object introduced in this chapter is F_0, the Taylor expansion of the network around initialization, defined as follows.

Definition 3.1. Given a prediction mapping $x \mapsto F(x; w)$ and *initial parameters* w_0, define the Taylor approximation at initialization as

$$F_0(x; w) := F(x; w_0) + \left\langle \bar{\partial}_w F(x; w_0), w - w_0 \right\rangle,$$

where $\bar{\partial}$ is the *minimum norm element of the Clarke differential*, and at this point in the text can be treated as a gradient. In this chapter, the choice of derivative at 0 for the ReLU will make no difference. *[todo 39/93]* ◇

Conventions on the initial point w_0 will be discussed shortly. First, with F_0 defined, the organization of this chapter is as follows.

- Section 3.1 first shows if norm is fixed and width is increased, then networks becomes closer and closer to their Taylor expansion F_0 at initialization.
- Since networks become close to their Taylor expansions, what can these Taylor expansions approximate? Section 3.2 shows that holding Frobenius norm fixed and increasing width is enough to approximate all continuous functions, and moreover that this suggests the study of a corresponding family of infinite-width networks.
- Rather than studying these Taylor expansions themselves, Section 3.3 studies pairwise inner products of the Taylor expansions on individual data points, what is normally called the *gram matrix of a kernel*, and gives rise to the *neural tangent kernel*.
- Lastly, **??** will briefly give some estimates for the norm in a more refined way than Section 3.2.

Remark 3.2 (Choice of initialization)**.** The theoretical literature uses many conventions, the most standard being $a_j \sim \text{Discrete}(\pm 1/\sqrt{m})$, a discrete uniform distribution on -1 and $+1$, and $v_j \sim \mathcal{N}(0, \mathcal{I}_d/d)$, a continuous multivariate Gaussian with independent coordinates and per-coordinate variance $1/d$; by contrast, `pytorch` defaults to $a_j \sim \text{Uniform}([-1/\sqrt{m}, +1/\sqrt{m}])$, meaning the continuous uniform distribution over the interval $[-1/\sqrt{m}, 1/\sqrt{m}]$, and v_j has independent coordinates with $v_{j,i} \sim \text{Uniform}([-1/\sqrt{d}, +1/\sqrt{d}])$. While the statistics of these two options are of the same order, and while both appear interchangeably in many aspects of probability theorem *[todo 40/93]*, they are not the same.
In this chapter, we will typically use $a_j \sim \text{Discrete}(\pm 1)$ and $v_j \sim \mathcal{N}(0, \mathcal{I}_d)$.
[todo 41/93] ◇

Throughout this chapter, the networks will have only two layers, and moreover only the first layer V will vary; the first choice is since, as mentioned, the corresponding bounds only degrade with more layers, and the second is so that the prediction mapping F is still nonlinear in the parameters. *[todo 42/93]*

Remark 3.3 (Random feature models)**.** *[todo 43/93]* ◇

A side story throughout this chapter will be the choice of *scaling*, meaning how to choose the scale of the various layers, and the consequence of this choice. This topic is somewhat hard to follow in the literature, as different choices are introduced with complicated consequences on the setting, and the reasons are rarely given. This chapter will only introduce scaling in Section 3.2, and attempt to justify the given choice.
[todo 44/93]
[todo 45/93]

3.1 Near initialization means near Taylor expansion

[todo 46/93]

This section shows that the quality of approximation given by the Taylor expansion degrades smoothly with the Frobenius norm to initialization. A key point is that this degradation never has a factor $\sqrt{m}$, which will be essential to the scaling selection in Section 3.2.

As a warm-up, consider a network with smooth activations σ.

Proposition 3.4. *If $\sigma : \mathbb{R} \to \mathbb{R}$ is β-smooth (meaning $|\sigma'(b) - \sigma'(c)| \le \beta|b-c|$), and $\|x\|_2 \le 1$, then for any parameters $V, V_0 \in \mathbb{R}^{m\times d}$,*

$$\left|F(x;(a_0,V))) - F_0(x;(a_0,V))\right| \le \frac{\beta\|a\|_\infty}{2}\|V - V_0\|_{\mathrm{F}}^2 .$$

Proof. By β-smoothness, for any r, s,

$$\left|\sigma(r) - \sigma(s) - \sigma'(s)(r-s)\right| = \left|\int_r^s \big(\sigma'(t) - \sigma'(s)\big)\,\mathrm{d}t\right| \le \frac{\beta(r-s)^2}{2}.$$

Therefore

$$\begin{aligned}
&\left|F(x;V) - F(x;V_0) - \left\langle \nabla F(x;V_0), V - V_0\right\rangle\right| \\
&\qquad \le \sum_j |a_j| \cdot \left|\sigma(v_j^\intercal x) - \sigma(v_{0,j}^\intercal x) - \sigma'(v_{0,j}^\intercal x)x^\intercal(v_j - v_{0,j})\right| \\
&\qquad \le \|a\|_\infty \sum_j \frac{\beta(v_j^\intercal x - v_{0,j}^\intercal x)^2}{2} \\
&\qquad \le \frac{\beta\|a\|_\infty}{2}\|V - V_0\|^2.
\end{aligned}$$

□

As mentioned, a key property in Proposition 3.4 is that it scales only with $\|V - V_0\|^2$, and not with m. Notice that if we try to brute-force a similar argument for the ReLU, we get a bad dependence on m.

Remark 3.5 (Incorrect ReLU brute-forcing)**.** Let's see how badly things go awry if we try to brute-force the proof, even in the simplifying situation that $W = V_0$. By similar reasoning to the earlier ReLU simplification, *[todo 47/93]*

$$\begin{aligned}
\left|F(x;V) - F_0(x;V)\right| &= \left|\left\langle \bar\partial F(x;V), V\right\rangle - \left\langle \bar\partial F(x;V_0), V\right\rangle\right| \\
&= \left|\sum_j a_j \left(\mathbb{1}[v_j^\intercal x \ge 0] - \mathbb{1}[v_{0,j}^\intercal x \ge 0]\right) v_j^\intercal x\right|. \qquad (3.6)
\end{aligned}$$

A direct brute-forcing with no sensitivity to random initialization gives

$$\left|F(x;V) - F_0(x;V)\right| \le \|a\|_\infty \sum_j \|v_j\| \le \|a\|_\infty \sqrt{m}\|V\|_{\mathrm{F}}.$$

We can try to save a bit by using the randomness of $(a_j)_{j=1}^m$, but since Proposition 3.7 is claimed to hold for every $\|W - V_0\|_{\mathrm{F}} \le B$, the argument might be complicated. Our eventual proof will only use randomness of V_0. ◇

Now we show how a careful study of $\mathbb{1}[v_j^\intercal x \ge 0] - \mathbb{1}[v_{0,j}^\intercal x \ge 0]$ via concentration can get us a bound closer to Proposition 3.4. This bound is stated with an extra degree of freedom, namely two matrices, a form we will use later with optimization.

Proposition 3.7. *For any radius $B \ge 0$, for any fixed $x \in \mathbb{R}^d$ with $\|x\| \le 1$, with probability at least $1-\delta$ over the draw of V_0, for any $W, V \in \mathbb{R}^{m\times d}$ with $\|W - V_0\|_{\mathrm{F}} \le B$ and $\|V - V_0\|_{\mathrm{F}} \le B$, then*

$$\left|F(x;V) - \left(F(x;W) + \left\langle \bar\partial_W F(x;W), V - W\right\rangle\right)\right| \le \|a\|_\infty m^{1/3}\left(4B^{4/3} + 2B\ln(1/\delta)^{1/4}\right).$$

Before giving the proof, a few interesting differences with Proposition 3.4 are worth mentioning. First, there is a dependence on m, namely $m^{1/3}$; this is not great, but sufficient for the scaling discussion in Section 3.2, where anything below $\sqrt{m}$ suffices. Secondly, note that Proposition 3.7 will make crucial use of the Gaussian random initialization of V_0, whereas probability made no appearance in Proposition 3.4.

Proceeding with the proof, the first step is a convenient concentration inequality.

Lemma 3.8. *For any $\tau > 0$ and $x \in \mathbb{R}^d$ with $\|x\| > 0$, with probability at least $1-\delta$ over $(v_j)_{j=1}^m$ with $v \sim \mathcal{N}(0, \mathcal{I}_d)$,*

$$\sum_{j=1}^m \mathbb{1}\left[|v_j^\intercal x| \le \tau\|x\|\right] \le m\tau + \sqrt{\frac{m}{2}\ln\frac{1}{\delta}}.$$

Proof. For any row j, define an indicator random variable

$$P_j := \mathbb{1}[|v_j^\intercal x| \le \tau\|x\|].$$

By rotational invariance, P_j is equivalent in distribution to $Q_j := \mathbb{1}[|g_j| \le \tau]$, where $g_j \sim \mathcal{N}(0,1)$, which by the form of the Gaussian density gives

$$\Pr[P_j = 1] = \Pr[Q_j = 1] = \int_{-\tau}^{+\tau} \frac{1}{\sqrt{2\pi}} e^{-g^2/2}\,\mathrm{d}g \le \frac{2\tau}{\sqrt{2\pi}} \le \tau.$$

By Hoeffding's inequality, with probability at least $1-\delta$,

$$\sum_{j=1}^{m} P_j \le m\Pr[P_1 = 1] + \sqrt{\frac{m}{2}\ln\frac{1}{\delta}} \le m\tau + \sqrt{\frac{m}{2}\ln\frac{1}{\delta}}.$$

□

Proof (Proof of Proposition 3.7)**.** Fix $x \in \mathbb{R}^d$. If $\|x\| = 0$, then for any $W \in \mathbb{R}^d$, $f(x;W) = 0 = f_0(x;W)$, and the proof is complete; henceforth consider the case $\|x\| > 0$. The proof idea is roughly as follows. The Gaussian initialization of V_0 concentrates around a rather large shell, and this implies $|v_{0,j}^\intercal x|$ is large with reasonably high probability. If $\|W - V_0\|_{\mathrm{F}}$ is not too large, then $\|w_j - v_{0,j}\|$ must be small for most coordinates; this means that $w_j^\intercal x$ and $v_{0,j}^\intercal x$ must have the same sign for most j, which controls the sum which was unclear in the earlier brute-forcing in eq. (3.6).
Proceeding in detail, fix a parameter $\tau > 0$ which will be optimized shortly. Let $\|W - V_0\| \le B$ amd $\|V - V_0\| \le B$ be given, and define the sets

$$\begin{aligned}
S_1 &:= \left\{ j \in [m] : |v_{0,j}^\intercal x| \le \tau\|x\| \right\}, \\
S_2 &:= \left\{ j \in [m] : \|w_j - v_{0,j}\| \ge \tau \right\}, \\
S_3 &:= \left\{ j \in [m] : \|v_j - v_{0,j}\| \ge \tau \right\}, \\
S &:= S_1 \cup S_2 \cup S_3.
\end{aligned}$$

By Lemma 3.8, with probability at least $1-\delta$,

$$|S_1| \le \tau m + \sqrt{m\ln(1/\delta)}.$$

On the other hand,

$$B^2 \ge \|W - V_0\|^2 \ge \sum_{j \in S_2} \|w_j - v_{0,j}\|^2 \ge |S_2|\tau^2,$$

meaning $|S_2| \le B^2/\tau^2$, and similarly $|S_3| \le B^2/\tau^2$. For any $j \notin S$, if $w_j^\intercal x > 0$, then

$$v_{0,j}^\intercal x \ge w_j^\intercal x - \|w_j - w_{0,j}\| \cdot \|x\| > \|x\|(\tau - \tau) = 0,$$

meaning $\mathbb{1}[w_j^\intercal x \ge 0] = \mathbb{1}[w_{0,j}^\intercal x \ge 0]$; the case that $j \notin S$ and $w_j^\intercal x < 0$ is analogous, as are inequalities for $v_j^\intercal x > 0$ and $v_j^\intercal x < 0$. Together,

$$\begin{aligned}
|S| &\le \tau m + \sqrt{m\ln(1/\delta)} + \frac{2B^2}{\tau^2}, \\
j \notin S &\Longrightarrow \mathbb{1}[w_j^\intercal x \ge 0] = \mathbb{1}[w_{0,j}^\intercal x \ge 0] = \mathbb{1}[v_j^\intercal x \ge 0].
\end{aligned} \tag{3.9}$$

Lastly, we can finally choose τ to balance terms in $|S|$: picking $\tau := B^{2/3}/m^{1/3}$ gives

$$|S| \le (Bm)^{2/3} + \sqrt{m\ln(1/\delta)} + 2(Bm)^{2/3} \le m^{2/3}\left(3B^{2/3} + \sqrt{\ln(1/\delta)}\right).$$

Now that we can control the set S (in particular, via eq. (3.9)), we can proceed essentially as in eq. (3.6). The derivation will also critically use an interesting pieces of algebra: if $\mathbb{1}[w_j^\intercal x \geq 0] \neq \mathbb{1}[v_j^\intercal x \geq 0]$, then $w_j^\intercal x$ and $v_j^\intercal x$ have different signs, and therefor $|v_j^\intercal x| \leq |v_j^\intercal x - w_j^\intercal x|$. Combining all these pieces,

$$\begin{aligned}
&\left|F(x;V) - \left(F(x;W) + \left\langle \bar{\partial}_W F(x;W), V - W\right\rangle\right)\right| \\
&\qquad = \left|\left\langle \bar{\partial}_W F(x;V) - \bar{\partial}_W F(x;W), V\right\rangle\right| \\
&\qquad = \left|\sum_j a_j \left(\mathbb{1}[w_j^\intercal x \geq 0] - \mathbb{1}[v_j^\intercal x \geq 0]\right) v_j^\intercal x\right| \\
&\qquad \leq \sum_j |a_j| \cdot \left|\mathbb{1}[w_j^\intercal x \geq 0] - \mathbb{1}[v_j^\intercal x \geq 0]\right| \cdot \left|w_j^\intercal x - v_j^\intercal x\right| \\
&\qquad \leq \|a\|_\infty \sum_j \left|\mathbb{1}[w_j^\intercal x \geq 0] - \mathbb{1}[v_j^\intercal x \geq 0]\right| \cdot \|w_j - v_j\| \\
&\qquad \leq \|a\|_\infty \sum_{j \in S} \|w_j - v_j\| \\
&\qquad \leq \|a\|_\infty \|W - V\|_{\mathrm{F}} \sqrt{|S|} \\
&\qquad \leq 2\|a\|_\infty B m^{1/3} \sqrt{3B^{2/3} + \sqrt{\ln(1/\delta)}} \\
&\qquad \leq \|a\|_\infty m^{1/3} \left(4B^{4/3} + 2B \ln(1/\delta)^{1/4}\right).
\end{aligned}$$

□

3.2 Scaling and universal approximation near initialization

[todo 48/93]

[todo 49/93]

[todo 50/93]

[todo 51/93]

[todo 52/93]

Research question 3.1. [todo 53/93]

The previous section showed that $F \approx F_0$ near initialization, but left open any characterization of F_0 itself; in particular, since F_0 is near *random* initialization, is F_0 essentially dominated by this randomness, and effectively a random function?

This section will show that in fact F_0 is a universal approximator, and moreover, that this universal approximation property comes from a *signal to noise ratio* that arises with large width. This signal to noise property will give rise to a choice of scaling in the networks, and taking the most reasonable choice will lead to the standard scaling. Moreover, this choice of

scaling allows for a natural infinite-width limiting object as $m \to \infty$.

To start, the signal to noise property is effectively a consequence of Frobenius norm geometry, described as follows. Suppose we pick a single pair $(a, v) \in \mathbb{R} \times \mathbb{R}^d$ with $|a| = 1$ and $\|v\| = 1$ and copy it m times; the corresponding network has magnitude m in some directions, since

$$F(v; (a, V)) = \sum_{j=1}^{m} a\sigma(v^\intercal v) = ma\sigma(\|v\|^2).$$

By contrast, this is unlikely to occur with random initialization: as provided rigorously below in Lemma 3.15 (but also as a direct consequence of standard Gaussian concentration, as in *[todo 54/93]*), $|F(x; w_0)| \leq 4\sqrt{md \ln(m/\delta)}$ with probability at least $1 - \delta$ for every $\|x\| \leq 1$. As such, planting many copies of certain directions means we can, with small Frobenius norm, easily dominate the noise and adjust the predictor into anything we want, all while staying close to initialization. This is formalized in the following statement.

Theorem 3.10. [todo 55/93] *Let a reference network $g(x) := \sum_{i=1}^{k} \alpha_i \sigma(\beta_i^\intercal x)$ be given with $\|\beta_i\| = 1$ for all i, along with a signal-to-noise parameter τ satisfying $\tau < \min_{i \neq j} \|\beta_i - \beta_j\|/2$. Then, with probability at least $1 - \delta$ over the draw of V_0 with $m \geq 4^{d+2} \ln(1/\delta)/\tau^{2d-2}$, there exists a choice of parameters $V \in \mathbb{R}^{m \times d}$ satisfying*

$$\max_j \|v_j - v_{0,j}\| \leq \frac{2^{d+1}\|\alpha\|_\infty}{\tau^{d-1}\sqrt{m}}, \qquad \text{and} \qquad \|V - V_0\| \leq \frac{2^{d+1}\|\alpha\|_2}{\tau^{d-1}},$$

so that for any $\|x\| \leq 1$, then $F(x; V) = F_0(x; V)$ and

$$\left| F(x; V) - \frac{2^{d+1}\sqrt{m}}{\tau^{d-1}} g(x) \right| \leq \sqrt{m} \left(16d \ln(m/\delta) + \|\alpha\|_1 \right).$$

For sake of discussion, the proofs are deferred to the end of the section.

In words, Theorem 3.10 says we can start with the random initialization parameters w_0 and some other network g, and spread g across many different nodes in $F(\cdot; w_0)$, obtaining another set of parameters w which are close in Frobenius norm to w_0, but now F is close to a rescaled copy of g.

Consider instead F by $\rho/\sqrt{m}$ where $\rho := \tau^{d-1}/2^{d+1}$, whereby theorem 3.10 becomes

$$\left| \frac{\rho}{\sqrt{m}} F(x; V) - g(x) \right| \leq \frac{\tau^{d-1}}{2^{d+1}} \left(16d \ln(m/\delta) + \|\alpha\|_1 \right),$$

meaning g and the rescaling $\rho F/\sqrt{m}$ are now close. Furthermore, since $\tau > 0$ is arbitrary, taking $\tau \to 0$ makes the right hand side of the preceding inequality equal to zero; while this also means the Frobenius norm in Lemma 3.15 will explode, on the other hand, τ can be taken to 0 slower than, say, $1/m^{1/6}$, in which case the the moral "close to initialization" yardstick provided by Proposition 3.7 is satisfied.

Summarizing, this discussion justifies scaling the network F by $\rho/\sqrt{m}$, which is standard practice, the $1/\sqrt{m}$ typically being absorbed in the initialization of a_j, and ρ being a common temperature parameter.

Remark 3.11. *[todo 56/93]* ◇

Remark 3.12 (Universal approximation)**.** Note that Theorem 3.10 implies F_0 is also a universal approximator: given a continuous function $h : \mathbb{R}^d \to \mathbb{R}$, use the techniques of Chapter 2 to obtain an approximating network g, and use Theorem 3.10 to embed it near initialization. While Theorem 3.10 introduces an exponential dependence on dimension, this dependence was already present due to the techniques in Chapter 2. Additionally, although the results of Chapter 2 were presented as though non-algorithmic, thanks to this remark, they in fact can directly relate to behavior of gradient descent near initialization. ◇

This choice of scaling has another consequence: it allows us to ensure F has a well-defined limit as $m \to \infty$. To construct this limit, consider the following

The limiting object will be formalized as follows. Let $\mathcal{T} : \mathbb{R}^d \to \mathbb{R}^d$ be a *transport mapping* of the weights at initialization: specifically, given a Gaussian vector $v_{0,j} \sim \mathcal{N}(0, \mathcal{I}_d)$, we construct our new weight v_j via $\mathcal{T}(v_{0,j})$ as

$$v_j := v_{0,j} + \frac{a_j \rho}{\sqrt{m}} \mathcal{T}(v_{0,j}). \tag{3.13}$$

This mapping $\mathcal{T}$ is thus an unambiguous way to define network weights $(v_j)_{j=1}^m$ given *any* initial random weights $(v_{0,j})_{j=1}^m$. Alternatively, $\mathcal{T}$ itself can be used to define an infinite-width network:

$$F_\infty(x; \mathcal{T}) := \int \left\langle \mathcal{T}(v), \bar{\partial}_v \sigma(v^\mathsf{T} x) \right\rangle \mathrm{d}\mathcal{N}(v);$$

while this looks complicated it is similar to the definition of F_0, and in fact they are the same as $m \to \infty$ with a bit more work.

Specifically, recall in the discussion after Theorem 3.10 that for the approximation error to go to 0, the scaling term ρ must also go to zero with m. The following equivalence between F_∞ and F_0 thus works with a sequence of temperatures $(\rho_m)_{m \geq 1}$.

Theorem 3.14. *Let transport mapping $\mathcal{T} : \mathbb{R}^d \to \mathbb{R}^d$, activation σ, and temperature sequence $(\rho_m)_{m \geq 1}$ be given satisfying the following properties:*

1. *$B := \sup_{v \in \mathbb{R}^d} |\mathcal{T}(v)| < \infty$;*
2. *F_0 satisfies a sub-exponential tail inequality, meaning there exists $c \geq 0$ so that for any $\|x\| \leq 1$, with probability at least $1 - \delta$, then $|F_0(x)| \leq c\sqrt{m} \ln(m/\delta)$;*
3. *there exists $C > 0$ so that $\rho_m \leq C / \ln(1+m)^2$.*

Then, for any $\|x\| \leq 1$, with probability 1 over the random sampling of $(v_{0,j})_{j \geq 1}$ and $(a_j)_{j \geq 1}$, letting $V^{(m)}$ denote the matrix obtained by stacking rows $(v_j^\mathsf{T})_{j=1}^m$ where v_j is given by eq. (3.13)*, then*

$$\lim_{m \to \infty} \frac{\rho}{\sqrt{m}} F_0(x; V^{(m)}) = F_\infty(x; \mathcal{T}).$$

The conditions of the bound are rather permissive; e.g., ρ need only decay inverse logarithmically, and the sub-gaussianity condition on σ is met for all standard activations (e.g., for the ReLU and for sigmoids). *[todo 57/93]*

Summarizing the material of this section: due to a signal-to-noise phenomenon, F_0 is close to any target function; rescaling the resulting bound gives the standard notion of scale; this notion of scale also leads to a limiting *infinite-width* network.

The rest of the section provides the deferred proofs. First, the claim that the initial prediction mapping (pure noise) has small magnitude (in particular, sublinear in m).

[todo 58/93]

Lemma 3.15. *Consider w_0 at initialization.* [todo 59/93]

1. *For any fixed $x \in \mathbb{R}^d$, with probability at least $1-\delta$, then $|F(x;w_0)| \leq 4\|x\|\sqrt{m}\ln(2m/\delta)$.*
2. *With probability at least $1-2\delta$, for any $\|x\| \leq 1$, then $|F(x;w_0)| \leq 32d\sqrt{m}\ln(m/\delta)$.*

[todo 60/93]

Remark 3.16. *[todo 61/93]* ◇

Proof. Throughout this proof, write $w = w_0$ for convenience.

1. Define $f_j := a_j\sigma(w_j^\intercal x)$, whereby $F(x;w) = \sum_j f_j$, and moreover each f_j is equivalent in distribution (via rotational invariance) to $q_j := a_j\sigma(g_j)\|x\|$, where $g_j \sim \mathcal{N}(0,1)$. By standard Gaussian concentration and a union bound, with probability at least $1-\delta/2$, then $\max_j g_j \leq 1+\sqrt{2\ln(2m/\delta)}$. Conditionining away this event, q_j is a zero-mean random variable with range $\|x\| \cdot [-1-\sqrt{2\ln(m/\delta)}, +1+\sqrt{2\ln(2m/\delta)}]$, whereby Hoeffding's inequality grants, with probability at least $1-\delta/2$,

$$\sum_j q_j \leq \|x\|\sqrt{2m(1+\sqrt{2\ln(2m/\delta)})^2\ln(2/\delta)} \leq 4\|x\|\sqrt{m}\ln(2m/\delta).$$

   Unioning the two failure events together gives the final bound. *[todo 62/93]*

2. Let $\epsilon > 0$ be a parameter which is optimized at the end of the proof. Let S_ϵ denote a discretization of $S := \{x \in \mathbb{R}^d : \|x\| \leq 1\}$, meaning for any $\|x\| \leq 1$, there exists $x_\epsilon \in S_\epsilon$ with $\|x - x_\epsilon\| \leq \epsilon$; as a lazy estimate, $|S_\epsilon| \leq (2/\epsilon)^d$. By a union bound over the preceding part, with probability at least $1-\delta_0$, then $\max_{x\in S_\epsilon}|F(x;w)| \leq 4\sqrt{m}\ln(2m|S_\epsilon|/\delta_0)$.

   Next, by standard Gaussian concentration and a union bound, with probability at least $1-\delta_1$, then $\max_j \|v_j\| \leq 1+\sqrt{2\ln(m/\delta_1)}$. Together, with probablity at least

$1 - \delta_1 + \delta_0$, for any $\|x\| \le 1$, choosing $x_\epsilon \in S_\epsilon$ with $\|x - x_\epsilon\| \le \epsilon$, then

$$\begin{aligned}
\big|F(x; w_0)\big| &\le \big|F(x_\epsilon; w)\big| + \big|F(x; w) - F(x_\epsilon; w)\big| \\
&\le 4\sqrt{m}\ln(2m|S_\epsilon|/\delta_0) + \sum_j \Big|\sigma(v_j^{\mathsf{T}} x) - \sigma(v_j^{\mathsf{T}} x_\epsilon)\Big| \\
&\le 4\sqrt{m}\ln(2m|S_\epsilon|/\delta_0) + m\|x - x_\epsilon\| \max_j \|v_j\| \\
&\le 4\sqrt{m}\ln(2m|S_\epsilon|/\delta_0) + m\epsilon\left(1 + \sqrt{2\ln(m/\delta_1)}\right).
\end{aligned}$$

The claim now follows by choosing $\epsilon := 1/m$ and $\delta_0 := \delta_1 := \delta$ and combining terms.

□

Now comes the proof of Theorem 3.10. As mentioned above, the method of proof is effectively to split up g and add it as tiny increments to the weights of w_0, whereby, despite a small change in Frobenius norm, the lining-up of these increments causes a signal-to-noise phenomenon. The proof is a little cautious, and only introduces changes in weights which are very close to those of the target network; this caution is what introduces the exponential dependence on dimension.

Proof (Proof of Theorem 3.10)**.** The method of proof is to argue that thanks to this large width, we can pick set of nodes U_i clustered around each β_i, meaning $\|\beta_i - \tilde{v}_j\| \le \tau$ where $\tilde{v}_j := v_{0,j}/\|v_{0,j}\|$ (the denominator is positive almost surely); if we then slightly amplify the norm of all these good nodes, it is easy to satisfy all conditions, and the main work is in arguing that these sets U_i are quite large.

The first step is to union together and discard a few probability events on V_0.

1. First, thanks to Lemma 3.15, with probability at least $1 - 2\delta$, then $|F(x; w_0)| \le 16d\sqrt{m}\ln(m/\delta)$ for all $\|x\| \le 1$.

2. For each $i \in \{1, \dots, k\}$, let $S_i \subset \{1, \dots, m\}$ denote the subset of weights close to β_i, with the signs of a_j and α_i matching, and the norm of $\|v_j\|$ is not too small: specifically,

$$S_i := \left\{j \in \{1, \dots, m\} \ : \ \|\beta_i - \tilde{v}_j\| \le \tau,\ \operatorname{sgn}(a_j) = \operatorname{sgn}(\alpha_i)\right\}.$$

   By standard cap estimates for spheres (Ball, 1997, Lemma 2.3), the probability of any j satisfying $\|\beta_i - v_j/\|v_j\|\| \le \tau$ is at least $(\tau/2)^{d-1}/2$, and the probability of $\operatorname{sgn}(a_j) = \operatorname{sgn}(\alpha_i)$ is $1/2$. Together, defining $\tau_d := \tau^{d-1}/2^{d+2}$, then

$$\mathbb{E}|S_i| = \sum_{j=1}^m \Pr[j \in S_i] \ge \frac{m\tau^{d-1}}{2^{d+1}} = 2m\tau_d,$$

   and by Hoeffding's inequality, a union bound, and the lower bound on m, it holds with probability at least $1 - \delta$ that

$$\min_i |S_i| \ge 2m\tau_d - \sqrt{m\ln(1/\delta)} \ge m\tau_d.$$

   Lastly, note that each $(S_i)_{i=1}^k$ are disjoint since $\min_{i \ne j} \|\beta_i - \beta_j\| > 2\tau$.

Henceforth discard the preceding failure events (and the null event that $\min_j \|v_j\| = 0$), for convenience define

$$c := \frac{1}{\tau\tau_d\sqrt{m}}, \qquad \text{and} \qquad c_0 := c\tau_d m = \frac{\sqrt{m}}{\tau},$$

and construct the network as follows. For each S_i, let U_i denote the first $m\tau_d$ vectors in S_i (which is well-defined since $|S_i| \geq m\tau_d$ as above), and define

$$v_j := \begin{cases} v_{0,j} + \frac{c|\alpha_i|v_{0,j}}{\|v_{0,j}\|} & \exists i \centerdot v_{0,j} \in U_i, \\ v_{0,j} & \text{otherwise.} \end{cases}$$

By construction, $\|v_j - v_{0,j}\| \leq c\|\alpha\|_\infty$ and

$$\|V - V_0\|^2 = \sum_{i=1}^{k} \sum_{j \in U_i} c^2\alpha_i^2 = m\tau_d c^2 \|\alpha\|_2^2 = \frac{\|\alpha\|_2^2}{\tau^2\tau_d}.$$

Furthermore, for any $x \in \mathbb{R}^d$, since v_j and $v_{0,j}$ have the same direction, setting $\tilde{v}_j := v_{0,j}/\|v_{0,j}\|$ for convenience,

$$\begin{aligned} F(x;(a_0,V)) &= \sum_j a_j \sigma(v_j^\mathsf{T} x) = \sum_j a_j \sigma(v_{0,j}^\mathsf{T} x) + \sum_j a_j \sum_i \mathbb{1}[j \in U_i]|\alpha_i| c\sigma(\tilde{v}_j^\mathsf{T} x), \\ &= F(x;w_0) + c\sum_i \alpha_i \sum_{j \in U_i} \sigma(\tilde{v}_j^\mathsf{T} x), \end{aligned}$$

whereby

$$\begin{aligned} \left|F(x;(a_0,V)) - c_0 g(x)\right| &\leq \left|F(x;w_0)\right| + \left|c_0 g(x) - c\sum_i \alpha_i \sum_{j \in U_i} \sigma(\tilde{v}_j^\mathsf{T} x)\right| \\ &\leq 16d\sqrt{m}\ln(m/\delta) + c\sum_i |\alpha_i| \sum_{j \in U_i} \left|\sigma(\beta_i^\mathsf{T} x) - \sigma(\tilde{v}_j^\mathsf{T} x)\right| \\ &\leq 16d\sqrt{m}\ln(m/\delta) + c\tau\tau_d m\|\alpha\|_1 \\ &= \sqrt{m}\left(16d\ln(m/\delta) + \|\alpha\|_1\right). \end{aligned}$$

Furthermore, for any x,

$$\left\langle \bar{\partial} F(x;V_0), V - V_0 \right\rangle = \sum_{i=1}^{k} \sum_{j \in U_i} \left\langle a_j x \sigma'(v_{0,j}^\mathsf{T} x), c|\alpha_i|\tilde{v}_{0,j} \right\rangle = c\sum_{i=1}^{k} \alpha_i \sum_{j \in U_i} \sigma(\tilde{v}_j^\mathsf{T} x),$$

and therefore $F(x;V) = F_0(x;V_0)$. □

Lastly, the proof of the limit property $F_0 \xrightarrow{m\to\infty} F_\infty$. While at first it may seem the strong law of large numbers suffices, the permissive conditions on ρ_m necessitate a more technical

proof.

Proof (Proof of Theorem 3.14)**.** Note

$$\begin{aligned}\frac{\rho_m}{\sqrt{m}} F_0(x; V^{(m)}) &= \frac{\rho_m}{\sqrt{m}} \sum_j a_j \left(\sigma(v_{0,j}^\intercal x) + \sigma'(v_{0,j}^\intercal x) \left\langle x, v_j^{(m)} - v_{0,j} \right\rangle \right) \\ &= \frac{\rho_m}{\sqrt{m}} \sum_j a_j \sigma(v_{0,j}^\intercal x) + \frac{1}{m} \sum_j \mathcal{T}(v_{0,j})^\intercal x \sigma'(v_{0,j}^\intercal x),\end{aligned}$$

and consider both terms separately. The second is easier: it is in the form of a standard law of large numbers, and is equal to its expected value $\mathbb{E}_v \mathcal{T}(v)^\intercal x \sigma'(v^\intercal x)$ almost surely. For the first term, fix any $\epsilon > 0$, and for each m define the event

$$E_m := \left[\left| \frac{\rho_m}{\sqrt{m}} \sum_{j=1}^m a_j \sigma(v_{0,j}^\intercal x) \right| \geq \epsilon \right].$$

[todo 63/93] By the conditions on ρ_m, it follows that if $m \geq m_0 := \exp(6C/\epsilon)$, then $\Pr[E_m] \leq 1/(1+m)^2$, since with probability at least $1 - 1/(1+m)^2$,

$$\left| \frac{\rho_m}{\sqrt{m}} F(x; V_0^{(m)}) \right| \leq \rho_m \ln(m(1+m)^2) \leq \frac{3C \ln(1+m)}{\ln(1+m)^2} \leq \epsilon.$$

Therefore

$$\sum_{m=1}^\infty \Pr(E_m) \leq m_0 + \sum_{m > m_0} \frac{2}{(1+m)^2} \leq m_0 + \frac{2\pi^2}{6} < \infty,$$

which by the Borel-Cantelli lemma implies

$$\limsup_{m \to \infty} \left| \frac{\rho_m}{\sqrt{m}} \sum_{j=1}^m a_j \sigma(v_{0,j}^\intercal x) \right| \leq \epsilon.$$

Since $\epsilon > 0$ was arbitrary, the proof is complete. *[todo 64/93]* □

3.3 The neural tangent kernel

[todo 65/93]

[todo 66/93]

The word "kernel" is used in many places, but one is as an abstraction of inner products; e.g., we can replace $x^\intercal x'$ with a function $k(x, x') = x^\intercal x'$; in certain special circumstances, given a function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ satisfying conditions, we can reverse engineer a Hilbert space induced by k.

Let's see how this naturally arises in our setup. Let's take $F \approx F_0$ to hard and sim-

ply predict with $F_0(x;w) = F(x;w_0) + \left\langle \bar{\partial} F(x;w_0), w - w_0 \right\rangle$. Here are a few elementary observations.

1. Our prediction mapping is *affine* in w; note that it is still nonlinear in x for general choices of w_0 (though this fails if $w_0 = 0$!).

2. Since we've said that we like low norm predictors, we may as well force $w - w_0 \in \operatorname{span}(\{\bar{\partial} F(x_1;w_0), \dots, F(x_n;w_0)\})$. We can achieve this by picking $v \in \mathbb{R}^n$, and writing

$$w := w_0 + \bar{\partial} F(X;w_0)^\intercal v, \qquad \text{where } \bar{\partial} F(X;w_0) = \begin{bmatrix} \leftarrow & \bar{\partial} F(x_1;w_0)^\intercal & \rightarrow \\ & \vdots & \\ \leftarrow & \bar{\partial} F(x_m;w_0)^\intercal & \rightarrow \end{bmatrix} \in \mathbb{R}^{n \times p}.$$

   As such, a prediction time, we compute

$$F_0(x;w) = F(x;w_0) + \left\langle \bar{\partial} F(x;w_0), w - w_0 \right\rangle = F(x;w_0) + \sum_{i=1}^n v_i \bar{\partial} F(x;w_0)^\intercal \bar{\partial} F(x_i;w_0),$$

   meaning we only rely upon inner products, and can replace the last term with

$$k_m(x, x_i) := \bar{\partial} F(x;w_0)^\intercal \bar{\partial} F(x_i;w_0),$$

   the *neural tangent kernel*. *[todo 67/93]*

3. Even more explicitly, consider trying to achieve low error in a standard least squares regression setup: we wish to pick $w \in \mathbb{R}^p$ to minimize

$$\widehat{\mathcal{R}}_0(w) = \sum_{i=1}^n \frac{1}{2} \left(F_0(x_i;w) - y_i \right)^2 = \sum_{i=1}^n \frac{1}{2} \left(\left\langle \bar{\partial} F(x_i;w_0), w - w_0 \right\rangle - (y_i - F(x_i;w_0)) \right)^2.$$

   The standard ordinary least squares solution is

$$w_{\text{ols}} := \left[\bar{\partial} F(X;w_0)^\intercal \bar{\partial} F(X;w_0) \right]^+ \bar{\partial} F(X;w_0) \left(y - F(X;w_0) \right).$$

   To perform well here, it seems natural to invoke the standard theory of least squares, e.g., requiring the *kernel gram matrix* $\bar{\partial} F(X;w_0)^\intercal \bar{\partial} F(X;w_0)$, and this motivates the appearance of these eigenvalues in a variety of works.

Remark 3.17. *[todo 68/93]* ◇

Remark 3.18. *[todo 69/93]* ◇

[todo 70/93]
[todo 71/93]
[todo 72/93]

To make things concrete, let's resume our consideration of shallow networks with only the inner layer trained. The first consideration will be the limiting kernel, which will have a simple form with the ReLU. To start, given two data points x and x', consider

$$\lim_{m\to\infty} \frac{\rho^2}{m} \bar{\partial} F(x; w_0)^\intercal \bar{\partial} F(x; w_0) = \rho^2 \lim_{m\to\infty} \frac{1}{m} \sum_j x^\intercal x' \sigma'(v_{0,j}^\intercal x) \sigma'(v_{0,j}^\intercal x'),$$

where our scaling choice from the previous section was again crucial in controlling the limit: we can simply apply the SLLN once more.

Theorem 3.19. *Fix any measurable selection σ' of $\bar{\partial}\sigma$. Then, for any inputs x and x',*

$$\begin{aligned} k_\infty(x, x') &= \lim_{m\to\infty} k_m(x, x') \\ &= \lim_{m\to\infty} \left\langle \frac{\rho}{\sqrt{m}} \bar{\partial} F(x; w_0), \frac{\rho}{\sqrt{m}} \bar{\partial} F(x'; w_0) \right\rangle \\ &= \rho^2 \int \left\langle x, x' \right\rangle \sigma'(v^\intercal x) \sigma'(v^\intercal x') \, \mathrm{d}\mathcal{N}(v) \qquad \text{almost surely.} \end{aligned}$$

Proof. For each $v_{0,j} \sim \mathcal{N}(0, \mathcal{I}_d)$, define a scalar random variable $z_j := \rho^2 \left\langle x, x' \right\rangle \sigma'(v_{0,j}^\intercal x) \sigma'(v_{0,j}^\intercal x')$, whereby

$$\mathbb{E} z_j = \rho^2 \int \left\langle x, x' \right\rangle \sigma'(v_{0,j}^\intercal x) \sigma'(v_{0,j}^\intercal x') \, \mathrm{d}\mathcal{N}(v_{0,j}) = k_\infty(x, x').$$

As such, by the strong law of large numbers, almost surely

$$\lim_{m\to\infty} k_m(x, x') = \lim_{m\to\infty} \frac{1}{m} \sum_j z_j = k_\infty(x, x').$$

□

Closing the loop with the previous section, another way to k_∞ is

$$k_\infty(x, x') = \int \left\langle x\sigma'(v^\intercal x), x'\sigma'(v^\intercal x') \right\rangle \mathrm{d}\mathcal{N}(v),$$

where the feature mapping is the same as the one used in F_∞ (for instance, as in Theorem 3.14. Similarly to the preceding discussion, if we only care about some finite training set $(x_i)_{i=1}^n$, then we may as well pick $\mathcal{T} \in \operatorname{span}\left((v \mapsto x_i \sigma'(v^\intercal x_i))_{i=1}^n\right)$, meaning selecting some $\alpha \in \mathbb{R}^n$ and then defining

$$\mathcal{T}(v) := \sum_i \alpha_i x_i \sigma'(v^\intercal x_i),$$

and thereby

$$\begin{aligned} F_\infty(x_j;\mathcal{T}) &= \int \left\langle \mathcal{T}(v), x_j\sigma'(v^\intercal x_j)\right\rangle \mathrm{d}\mathcal{N}(v) \\ &= \int \sum_i \alpha_i \left\langle x_i\sigma'(v^\intercal x_i), x_j\sigma'(v^\intercal x_j)\right\rangle \mathrm{d}\mathcal{N}(v) \\ &= \sum_i \alpha_i k_\infty(x_i,x_j). \end{aligned} \tag{3.20}$$

Remark 3.21. *[todo 73/93]* ◇

Remark 3.22. Once again let's revisit the issue of the choice of $\sigma'(0)$. In k_∞, this is a measure zero set, so once again the choice does not matter. ◇

[todo 74/93]

An attractive property of k_∞ is that its explicit form is often both easy to compute and has a simple expression.

Proposition 3.23. *Consider temperature $\rho = 1$, the ReLU $\sigma(z) := \max\{0, z\}$, and any x, x' with $\|x\| = 1 = \|x'\|$. Then*

$$k_\infty(x,x') = \left\langle x, x'\right\rangle \mathbb{E}_{v\sim\mathcal{N}} \mathbb{1}[v^\intercal x \geq 0]\cdot \mathbb{1}[w^\intercal x \geq 0] = \left\langle x, x'\right\rangle \left(\frac{\pi - \arccos(\langle x, x'\rangle)}{2\pi}\right).$$

Remark 3.24. One way to relax $\|x\| = 1$ is to start with $\|x\| \leq 1$ and work with $(x^\intercal, \sqrt{1-\|x\|^2}) \in \mathbb{R}^{d+1}$; the convenience of $\|x\| = 1$ and this padding trick are common in the literature. As in the results of the previous section, however, this trick is not necessary. ◇

Proof. To start, by definition of k_∞,

$$k_\infty(x,x') = \left\langle x, x'\right\rangle \mathbb{E}_{v\sim\mathcal{N}} \sigma'(v^\intercal x)\sigma'(v^\intercal x') = \left\langle x, x'\right\rangle \mathbb{E}_{v\sim\mathcal{N}} \mathbb{1}[v^\intercal x \geq 0]\cdot \mathbb{1}[v^\intercal x' \geq 0],$$

where the second equality used the fact that the two sets $\{v : v^\intercal x = 0\}$ and $\{v : v^\intercal x' = 0\}$ are $\mathcal{N}$-null.

Next note that this expression does not depend on $\|v\|$, meaning we can replace v by $\|v\|$ (throwing out the $\mathcal{N}$-null event $\|v\| = 0$), and consider $z \sim \mathcal{U}$, the uniform probability distribution on the surface of the sphere:

$$k_\infty(x,x') = \left\langle x, x'\right\rangle \mathbb{E}_{z\sim\mathcal{U}} \mathbb{1}[z^\intercal x \geq 0]\cdot \mathbb{1}[z^\intercal x' \geq 0].$$

Note that if $x = x'$, then $k_\infty(x,x') = 1/2$ and the proof is complete, thus consider the case $x \neq x'$.

To simplify further, it seems that all that should matter is the plane spanned by $\{x, x'\}$, explicitly, by rotational invariance of $\mathcal{N}$ (e.g., by substituting v with Mv where M is the fixed rotation matrix whose first column is x, second column is $(I - xx^\intercal)x'/\|(I - xx^\intercal)x'\|$, and the remaining are arbitrary but orthogonal), then

$$\begin{aligned} k_\infty(x, x') &= \left\langle x, x' \right\rangle \mathbb{E}_{v\sim\mathcal{N}} \mathbb{1}[v^\intercal M x \geq 0] \cdot \mathbb{1}[v^\intercal M x' \geq 0] \\ &= \left\langle x, x' \right\rangle \mathbb{E}_{v\sim\mathcal{N}} \mathbb{1}[v_1 \geq 0] \cdot \mathbb{1}[v_1 \left\langle x, x' \right\rangle + v_2 \sqrt{1 - \left\langle x, x' \right\rangle^2} \geq 0]. \end{aligned}$$

We can now consider this geometrically: v is sampled uniformly on the circle in $\mathbb{R}^2$, we have one data point at $(1, 0)$, and another at $(\left\langle x, x' \right\rangle, \sqrt{1 - \left\langle x, x' \right\rangle^2})$, and we'd like to know the probablity that v has positive inner product with both. Letting $\theta = \arccos(\left\langle x, x' \right\rangle)$ denote the angle between the two points, the region of the circle which we can fall within has arc length $\pi - \theta$, and thus probability mass $(\pi - \theta)/(2\pi)$, completing the proof. *[todo 75/93]* □

To close are a few observations. Firstly, consider the multi-layer case; here we have parameters $w = (W_L, \ldots, W_1)$, and we can organize $\bar{\partial}_w F(x; w_0)$ by layers as $(\bar{\partial}_{W_i} F(x_0; w_0))_{i=1}^L$, whereby

$$\begin{aligned} \left\langle \bar{\partial}_w F(x; w_0), \bar{\partial}_w F(x'; w_0) \right\rangle &= \left\langle \left(\bar{\partial}_{W_i} F(x; w_0)\right)_{i=1}^L, \left(\bar{\partial}_{W_i} F(x'; w_0)\right)_{i=1}^L \right\rangle \\ &= \sum_{i=1}^L \left\langle \bar{\partial}_{W_i} F(x; w_0), \bar{\partial}_{W_i} F(x'; w_0) \right\rangle. \end{aligned}$$

On the one hand, this is a nicely clean expression, which decomposes over layers. On the other hand, this highlights that this perspective near initialization is perhaps insufficiently sensitive to the benefits of composing layers together; indeed, there is evidence that the kernel view exhibits no great strenghtening in representation as depth increases (Bietti and Bach, 2020).

Lastly, to close with another tangential comment, rather than indirectly proving F_∞ is a universal approximator via taking $m \to \infty$ within the signal-to-noise bound Theorem 3.10 and then applying universal approximation of finite-width networks from Chapter 2, we can directly establish universal approximation properties of the infinite-width ReLU kernel from Proposition 3.23.

Proceeding in detail, define a subset of interest $\mathcal{X} \subseteq \mathbb{R}^d$ as

$$\mathcal{X} := \left\{ x \in \mathbb{R}^d : \|x\| = 1, x_d = 1/\sqrt{2} \right\},$$

which corresponds to baking in padding as in Remark 3.24. Additionally, define a family of predictors corresponding to F_∞ written in terms of kernels as in eq. (3.20), namely

$$\mathcal{H} := \left(x \mapsto \sum_{j=1}^m \alpha_j k(x, x_j) \ : \ m \geq 0, \alpha_j \in \mathbb{R}, x_j \in \mathcal{X} \right).$$

We now show $\mathcal{H}$ is a universal approximator.

Proposition 3.25. *$\mathcal{H}$ is a* universal approximator *over $\mathcal{X}$: for every continuous $g : \mathbb{R}^d \to \mathbb{R}$ and every $\epsilon > 0$, there exists $h \in \mathcal{H}$ with $\sup_{x\in\mathcal{X}} |g(x) - h(x)| \leq \epsilon$.*

Remark 3.26. Following on the padding comments in Remark 3.24, the use of bias and padding here is simply to reduce more quickly to existing lemmas for universal approximation; a direct proof should also be fairly easy. ◇

Proof. Consider the set $U := \{u \in \mathbb{R}^{d-1} : \|u\|^2 \leq 1/2\}$, and the kernel function

$$k(u, u') := f(u^\intercal u'), \qquad f(z) := \frac{(z + 1/2)}{2} - \frac{(z + 1/2)\arccos(z + 1/2)}{2\pi}.$$

We will show that this kernel is a universal approximator over U, which means it is also a universal approximator on its boundary $\{u \in \mathbb{R}^{d-1} : \|u\|^2 = 1/2\}$, and thus the kernel

$$(x, x') \mapsto \frac{x^\intercal x'}{\pi} - \frac{x^\intercal x' \arccos(x^\intercal x')}{2\pi}$$

is a universal approximator over $\mathcal{X}$.
Going back to the original claim, first note that arccos has the Maclaurin series

$$\arccos(z) = \frac{\pi}{2} - \sum_{k\geq 0} \frac{(2k)!}{2^{2k}(k!)^2}\left(\frac{z^{2k+1}}{2k+1}\right),$$

which is convergent for $z \in [-1, +1]$. From here, it can be checked that f has a Maclaurin series where every term is not only nonzero, but positive (adding the bias ensured this). This suffices to ensure that k is a universal approximator (Steinwart and Christmann, 2008, Corollary 4.57). □

3.4 Bibliographic notes

[todo 76/93]

3.5 Exercises

3.5.1 Research questions

Research question 3.2. *Improve Proposition 3.7 to reduce or entirely drop $m^{1/3}$, perhaps via careful use of a second layer.* [todo 77/93] [todo 78/93]

[todo 79/93]

Chapter 4

Benefits of other architectures

This section is unfortunately incomplete, and for now will only include boiled-down information needed for lecture. The sections are as follows:

- Section 4.1 will define and provide some approximation properties of the *triangle mapping* Δ; this construction is the basis for all results giving the power of many-layered networks (see bibliographic remarks in **??** for details, in particular a different separation technique between depths 2 and 3).
- Section 4.2 formally proves a few separation guarantees, showing in a strong sense that the iterated triangle map is easily-approximable with a dep network, and hard to approximate byu a shallow network, even with exponential width.

Results I will eventualyl add: Sobolev space approximation, other architectures, other settings, other types of layers, and other ways of measuring benefits (e.g., norms in function spaces).

4.1 Multi-layer benefits via the triangle mapping ΔBenefits of depth

[todo 80/93]

Consider the Δ function:

$$\Delta(x) = 2\sigma_{\mathrm{r}}(x) - 4\sigma_{\mathrm{r}}(x-1/2) + 2\sigma_{\mathrm{r}}(x-1) = \begin{cases} 2x & x \in [0, 1/2), \\ 2-2x & x \in [1/2, 1), \\ 0 & \text{otherwise.} \end{cases}$$

How does Δ look? And how about $\Delta^2 := \Delta \circ \Delta$? And Δ^3? *[todo 81/93]*

The pattern is that Δ^L has 2^{L-1} copies of it self, uniformly shrunk down. In a sense, complexity has increased exponentially as a function of the the number of nodes and layers (both $\mathcal{O}(L)$). Later, it will matter that we not only have many copies, but that they are identical (giving uniform spacing). There are a few ways to capture this "fractal" or "exponential" power, as follows is one way which we will use later.

Proposition 4.1 (Fractal property of Δ)**.** *Let $\langle x \rangle := x - \lfloor x \rfloor$ denote the fractional part of $x \in \mathbb{R}$. Then*

$$\Delta^L(x) = \Delta(\left\langle 2^{L-1}x \right\rangle) = \Delta(2^{L-1}x - \lfloor 2^{L-1}x \rfloor).$$

Proof. The proof proceeds by induction on $L = i$.
For the base case $i = 1$, if $x \in [0, 1)$ then directly

$$\Delta^1(x) = \Delta(x) = \Delta(\langle x \rangle) = \Delta(\left\langle 2^0 x \right\rangle),$$

whereas $x = 1$ means $\Delta^1(x) = \Delta(0) = \Delta(\left\langle 2^0 x \right\rangle)$.
For the inductive step, consider Δ^{i+1}. The proof can proceed by peeling individual Δ from the left or from the right; the choice here is to peel from the right. Consider two cases.

- If $x \in [0, 1/2]$,

$$\Delta^{i+1}(x) = \Delta^i(\Delta(x)) = \Delta^i(2x) = \Delta(\left\langle 2^{i-1}2x \right\rangle) = \Delta(\left\langle 2^i x \right\rangle).$$

- If $x \in (1/2, 1]$, now additionally using a reflection property of Δ (namely $\Delta(z) = \Delta(1 - z)$ for $z \in [0, 1]$),

$$\begin{aligned}\Delta^{i+1}(x) &= \Delta^i(\Delta(x)) = \Delta^i(2 - 2x) \\ &= \Delta^{i-1}(\Delta(2 - 2x)) = \Delta^{i-1}(\Delta(1 - (2 - 2x))) = \Delta^i(2x - 1) \\ &= \Delta(\left\langle 2^i x - 2^{i-1} \right\rangle) = \Delta(\left\langle 2^i x \right\rangle).\end{aligned}$$

  (If $i = 1$, use $\Delta^{1-1}(x) = x$.)

□

We've established that Δ^L has exponentially many copies of itself. But does this yield anything useful? Here are a few applications, one of which we will investigate in detail for the rest of the section.

1. As in Theorem 4.2, we can use Δ to approximate $x \mapsto x^2$ with to accuracy $\epsilon > 0$ with $\ln(1/\epsilon)$ nodes, whereas the shallow approaches of Chapter 2 required $\mathcal{O}(1/\epsilon)$ nodes. As will be discussed there, this also implies efficient approximation of polynomials and smooth functions.

2. We can efficiently write the parity function on the hypercube in $d = 2^L$ dimensions: given $x \in \{\pm 1\}^d$, then $\prod_{i=1}^d x_i = \Delta^{L-1}\left(\frac{d+\sum_i x_i}{2d}\right)$.

3. ("Viral fractal property".) If $f : [0, 1] \to \mathbb{R}$ is symmetric about $1/2$ (meaning $f(x) = f(1 - x)$ for $x \in [0, 1]$)m then $f \circ \Delta^L$ also creates 2^L copies of f; as such, the fractal property from Proposition 4.1 can be inherited by other functions!

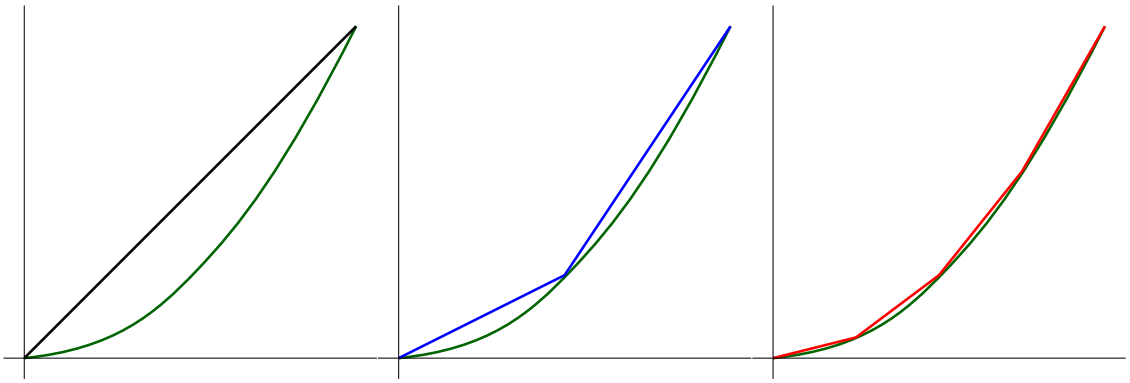

Figure 4.1: Affine interpolation of x^2 at different scales.

Figure 4.2: The difference $h_{i+1} - h_i$.

4. We can use Δ^L to extract bits: this is direct from Proposition 4.1, and used in a variety of works.

The rest of the section now turns to motivating and proving Theorem 4.2, namely the ability to efficiently approximate squaring, which will also give us multiplication via the *polarization identity* $xy = \frac{1}{2}\left((x+y)^2 - x^2 - y^2\right)$.

To start, let's recall one of the perspectives on univariate approximation from Chapter 2: using an exact integral remainder form of Taylor's theorem, we can write x^2 as an infinite-width network:

$$x^2 = \int_0^\infty 2\sigma(x-b)\,\mathrm{d}b;$$

in particular, we need to place ReLU nodes *uniformly*. While univariate approximation could not take advantage of this, this uniformity will allow us to use the "copying" structure of Δ^L as in Proposition 4.1.

Now let's proceed in detail, indeed with a brute-force approach which will happen to work: namely, we will simply produce a uniform affine interpolation of x^2, and then show we can write it efficiently using Δ^L. First define a set of interpolation points S_i at resolution i as

$$S_i := \left\{ \frac{0}{2^i}, \frac{1}{2^i}, \ldots, \frac{2^i}{2^i} \right\},$$

and let h_i be the affine interpolation of x^2 along S_i, meaning $h_i(x) = x^2$ for $x \in S_i$, with affine interpolation otherwise, as in Figure 4.1.

To write h_i in terms of Δ^i, we will consider differences $h_{i+1} - h_i$, and complete the construction with the telescoping sum

$$h_i(x) = h_0(x) + \sum_{j<i} \left(h_{j+1}(x) - h_j(x)\right) = x + \sum_{j<i} \left(h_{j+1}(x) - h_j(x)\right).$$

In detail, since $S_i \subset S_{i+1}$, we know that $h_{i+1}(x) = h_i(x)$ for $x \in S_i \cap S_{i+1} = S_i$, therefore the interesting case is to choose any integer $k\{0, 1, \ldots, 2^i - 1\}$ and consider $x = (2k+1)/2^{i+1} \in S_{i+1} \setminus S_i$: in this situation,

$$\begin{aligned} h_i(x) - h_{i+1}(x) &= \frac{1}{2}\left[h_i(x - 2^{-i-1}) + h_i(x + 2^{-i-1})\right] - h_{i+1}(x) \\ &= \frac{1}{2}\left[\left(\frac{2k}{2^{i+1}}\right)^2 + \left(\frac{2k+2}{2^{i+1}}\right)^2\right] - \left(\frac{2k+1}{2^{i+1}}\right)^2 \\ &= \frac{1}{4^{i+1}}\left[2k^2 + \left(2k^2 + 4k + 2\right) - \left(4k^2 + 4k - 1\right)\right] \\ &= \frac{1}{4^{i+1}}. \end{aligned}$$

Remarkably, this error does not depend on k, meaning it follows exactly the same fractal structure we get with Δ^{i+1} (cf. Figure 4.2), and therefore for any $x \in S_{i+1}$ this ddifference across the two cases can be written compactly as

$$h_i(x) - h_{i+1}(x) = \frac{1}{4^{k+1}}\Delta^{i+1}(x).$$

Moreover, since h_{i+1} and h_i are affine interpolants on a refining grid, then $h_i - h_{i+1}$ is itself an affine interpolation between the points of S_{i+1}, and therefore the preceding equality holds for $x \in [0,1] \setminus S_{i+1}$ as well. Concluding by telescoping,

$$h_i(x) = x + \sum_{j<i}\left(h_{j+1}(x) - h_j(x)\right) = x - \sum_{j<i}\frac{\Delta^{j+1}(x)}{4^{j+1}}.$$

Summarizing this construction and analyzing a few more of its properties gives the following.

Theorem 4.2. *Let h_i denote the piecewise-affine interpolation of x^2 along S_i.*

1. *h_i can be written as a ReLU network consisting of $2i$ layers and $3i$ nodes using "skip connections", or a pure ReLU network with $2i$ layers and $5i$ nodes.*
2. *$\sup_{x\in[0,1]} |h_i(x) - x^2| \leq 4^{-i-1}$.*

To interpret the statement, we can say that to achieve a desired accuracy ϵ, it suffices to use $\mathcal{O}(\ln(1/\epsilon))$ layers and nodes. We have not yet formally stated that this is impossible with a shallow network, we only have a bad $\mathcal{O}(1/\epsilon)$ *upper bound* from Chapter 2; a formal separation will be given in Section 4.2.

Proof. 1. Since $h_i = x - \sum_{j=1}^{i} \frac{\Delta^j}{4^j}$ and since Δ^j requires 3 nodes and 2 layers for each new power, a worst case construction would need $2i$ layers and $3\sum_{j\leq i} j = \mathcal{O}(i^2)$ nodes, but we can reuse individual Δ elements across the powers, and thus need only $3i$, though the network has "skip connections" (in the ResNet sense); alternatively we can replace the skip connections with a single extra node per layer which accumulates the output, or rather after layer j outputs h_j, which suffices since $h_{j+1} - h_j = \Delta^{j+1}/4^{j+1}$.

[todo 82/93]

2. Fix i, let any $x \in [0,1]$ be given, and choose $k \in \{0, 1, \ldots, 2^i - 1\}$ and $\tau \in [0,1]$ so that $x = (k+\tau)/2^i$. Then the error between x^2 and $h_i(x)$ is bounded above by

$$\begin{aligned}
|x^2 - h_i(x)| &= \left| \left(\frac{k+\tau}{2^i}\right)^2 - (1-\tau)\left(\frac{k}{2^i}\right)^2 - \tau\left(\frac{k+1}{2^i}\right)^2 \right| \\
&= \frac{1}{4^i}\left|k^2 + 2k\tau + \tau^2 - (1-\tau)k^2 - \tau(k^2 + 2k + 1)\right| \\
&= \frac{1}{4^i}\left|t^2 - \tau\right| \\
&\leq \frac{1}{4^{i+1}}.
\end{aligned}$$

[todo 83/93]

□

To conclude this section, we note as above that approximate squaring implies approximate products; this fact will be strengthened in **??** to approximate functions with well-behaved derivatives by approximating their Taylor expansions.

Corollary 4.3. *Given any i, there exists $g : \mathbb{R}^2 \to \mathbb{R}$ written as a ReLU network with $16i$ nodes and $3i$ layers so that, for any $(x, x') \in [0,1]^2$,*

$$|g_i(x, x') - xx'| \leq \frac{1}{4^i},$$

and moreover $g(x, x') = 0$ if $x = 0$ or $x' = 0$.

Proof. It suffices to choose

$$g_i(x, x') = \frac{1}{2}\left(4h_i((x + x')/2) - h_i(x) - h_i(x')\right),$$

, which can be written as a ReLU network with $16i$ nodes and $6i$ layers (by running three networks for h_i in parallel, combining their outputs, and using the size estimates from Theorem 4.2). By construction, since $h_i(x) = 0$ when $x = 0$ and $h_i(x') = 0$ when $x' = 0$, then $g_i(x, x') = 0$ when either $x = 0$ or $x' = 0$ (or both). Moreover, via the polarization identity, the error in the general case can be upper bounded via the error estimate for h_i from Theorem 4.2 as

$$\begin{aligned}
\left|g_i(x, x') - xx'\right| &= \frac{1}{2}\left|4h_i((x + x')/2) - h_i(x) - h_i(x') - \left(4((x + x')/2)^2 - x^2 - (x')^2\right)\right| \\
&\leq \frac{1}{2}\left(4\left|h_i((x + x')/2) - ((x + x')/2)^2\right| + \left|h_i(x) - x^2\right| + \left|h_i(x') - (x')^2\right|\right) \\
&\leq \frac{1}{2}\left(\frac{4}{4^{i+1}} + \frac{1}{4^{i+1}} + \frac{1}{4^{i+1}}\right) \\
&\leq \frac{1}{4^i},
\end{aligned}$$

where dividing by two in $h_i((x+x')/2)$ was necessary to invoke the error estimate for h_i, which only holds over $[0,1]$, not $[0,2]$. □

4.2 Depth separations

This section will establish two different *depth separation* theorems using essentially the same proof technique (ReLU affine piece counting).

- **??** shows that Δ^{L^2+2} is hard to approximate *up to constant accuracy* by shallow networks of subexponential depth.
- Theorem 4.6 shows that x^2 is hard to approximate *up to* $\epsilon > 0$ accuracy by networks of depth much lower than $\ln(1/\epsilon)$.

The reason the second statement has a much weaker goal ($\epsilon > 0$ accuracy and not a constant) is because the approximated functoin x^2 is much simpler; e.g., it is 2-Lipschitz and monotone over $[0,1]$, whereas Δ^{L^2+2} is 2^{L^2+2}-Lipschitz and changes monotonicity $2^{L^2+2}-1$ times.

Theorem 4.4. *Let $L \geq 2$ be given. Then $f = \Delta^{L^2+2}$ can be written as a ReLU network with $3L^2+6$ nodes and $2L^2+4$ layers, but any ReLU network g with $\leq 2^L$ nodes and $\leq L$ layers can not approximate it:*

$$\int_{[0,1]} \big|f(x) - g(x)\big| \, \mathrm{d}x \geq \frac{1}{32}.$$

Remark 4.5 (Why L_1 metric?)**.** Previously, we used L_2 and L_∞ to state good upper bounds on approximation; for bad approximation, we want to argue there is a large region where we fail, not just a few points, and that's why we use an L_1 norm.
To be able to argue that such a large region exists, we don't just need the hard function $f = \Delta^{L^2+2}$ to have many regions, we need them to be regularly spaced, and not bunch up. In particular, if we replaced Δ with the similar function $4x(1-x)$, then this proof would need to replace $\frac{1}{32}$ with something decreasing with L. All of these considerations are much simpler if we use the L_∞ norm. ◇

[todo 84/93]
[todo 85/93]

Theorem 4.6. *Any ReLU network f with $\leq L$ layers and $\leq N$ nodes satisfies*

$$\int_{[0,1]} (f(x) - x^2)^2 dx \geq \frac{1}{5760(2N/L)^{4L}}.$$

[todo 86/93]

Remark 4.7 (Explicit $\ln(1/\epsilon)$ conversion)**.** *[todo 87/93]*
for apples-to-apples, use quadratic depth sep? so say $L < \sqrt{\ln(1/\epsilon)}$? Then want to pick upper bond on N satisfying

$$\frac{1}{5760(2N/L)^{4L}} > \epsilon,$$

equvalently

$$\ln 5760(2N/L)^{4L} \leq \ln \frac{1}{\epsilon},$$

and thus

$$\ln 5760(2N/L)^{4L} = \ln 5760 + 4L \ln 2N - 4L \ln L \leq \ln 5760 + 4\sqrt{\ln \frac{1}{\epsilon}} \ln 2N - 4L \ln L,$$

So that means if $N \leq \frac{1}{2\epsilon^{1/\sqrt{\ln(1/\epsilon)}}}$, then

$$\sqrt{\ln \frac{1}{\epsilon}} \ln 2N \leq \sqrt{\ln \frac{1}{\epsilon}} \frac{\ln(1/\epsilon)}{\sqrt{\ln(1/\epsilon)}} = \ln(1/\epsilon).$$

This sort of means N is exponential in $\ln(1/\epsilon)$, sort of. Need to do this carefully to make it sound compelling.

◇

[todo 88/93]

Proof plan for Theorem 5.1 ((Telgarsky 2015, 2016)):

1. (Shallow networks have low complexity.) First we will upper bound the number of oscillations in ReLU networks. The key part of the story is that oscillations will grow polynomially in width, but *exponentially* in depth. [mjt☹: give explicit lemma ref]

2. (There exists a *regular*, high complexity deep network.) Then we will show there exists a function, realized by a slightly deeper network, which has many oscillations, which are moreover *regularly spaced*. The need for regular spacing will be clear at the end of the proof. We have already handled this part of the proof: the hard function is Δ^{L^2+2}.

3. Lastly, we will use a region-counting argument to combine the preceding two facts to prove the theorem. This step would be easy for the L_∞ norm, and takes a bit more effort for the L_1 norm.

Proceeding with the proof, first we want to argue that shallow networks have low complexity. Our notion of complexity is simply the number of affine pieces.

Definition 4.8 (Affine complexity)**.** For any univariate function $f : \mathbb{R} \to \mathbb{R}$, let $N_A(f)$ denote the number of affine pieces of f: the minimum cardinality (or ∞) of a partition of $\mathbb{R}$ so that f is affine when restricted to each piece. ◇

Lemma 4.9. *Let $f : \mathbb{R} \to \mathbb{R}$ be a ReLU network with L layers of widths $(m_1, \ldots, m_L)$ with $m = \sum_i m_i$.*

- *Let $g : \mathbb{R} \to \mathbb{R}$ denote the output of some node in layer i as a function of the input. Then the number of affine pieces $N_A(g)$ satisfies*

$$N_A(g) \leq 2^i \prod_{j<i} m_j.$$

- $N_A(f) \leq \left(\frac{2m}{L}\right)^L$.

Remark 4.10. Working with the ReLU really simplifies this reasoning! ◇

Our proof will proceed by induction, using the following combination rules for piecewise affine functions.

Lemma 4.11. *Let functions $f, g, (g_1, \ldots, g_k)$, and scalars $(a_1, \ldots, a_k, b)$ be given.*

1. $N_A(f + g) \leq N_A(f) + N_A(g)$.
2. $N_A(\sum_i a_i g_i + b) \leq \sum_i N_A(g_i)$.
3. $N_A(f \circ g) \leq N_A(f) \cdot N_A(g)$.
4. $N_A\left(x \mapsto f(\sum_i a_i g_i(x) + b)\right) \leq N_A(f) \sum_i N_A(g_i)$.

This immediately hints a "power of composition": we increase the "complexity" multiplicatively rather than additively!

Proof (Proof of Lemma 4.11)**.**

1. Draw f and g, with vertical bars at the right boundaries of affine pieces. There are $\leq N_A(f) + N_A(g) - 1$ distinct bars, and $f + g$ is affine between each adjacent pair of bars.
2. $N_A(a_i g_i) \leq N_A(g_i)$ (equality if $a_i \neq 0$), thus induction with the preceding gives $N_A(\sum_i a_i g_i) = \sum_i N_A(g_i)$, and N_A doesn't change with addition of constants.
3. Let $P_A(g)$ denote the pieces of g, and fix some $U \in P_A(g)$; g is a fixed affine function along U. U is an interval, and consider the pieces of $f_{|g(U)}$; for each $T \in P_A(f_{|g(U)})$,

f is affine, thus $f \circ g$ is affine (along $U \cap g_{|U}^{-1}(T)$), and the total number of pieces is

$$\sum_{U \in P_A(g)} N_A(f_{|g(U)}) \leq \sum_{U \in P_A(g)} N_A(f) \leq N_A(g) \cdot N_A(f).$$

4. Combine the preceding two.

□

Remark 5.9 The composition rule is hard to make tight: the image of each piece of g must hit all intervals of f! This is part of the motivation for the function Δ, which essentially meets this bound with every composition.

Proof (Proof of Lemma 4.9)**.** To prove the second from the first, $N_A(f) \leq 2^L \prod_{j \leq L} m_j$,

$$\prod_{j \leq L} m_j = \exp \sum_{j \leq L} \ln m_j = \exp \frac{1}{L} \sum_{j \leq L} L \ln m_j \leq \exp L \ln \frac{1}{L} \sum_{j \leq L} m_j = \left(\frac{m}{L}\right)^L.$$

For the first, proceed by induction on layers. Base case: layer 0 mapping the data with identity, thus $N_A(g) = 1$. For the inductive step, given g in layer $i+1$ which takes $(g_1, \ldots, g_{m_i})$ from the previous layer as input,

$$\begin{aligned} N_A(g) = N_A(\sigma(b + \sum_j a_j g_j)) &\leq 2 \sum_{j=1}^{m_i} N_A(g_j) \\ &\leq 2 \sum_{j=1}^{m_i} 2^i \prod_{k<i} m_k = 2^{i+1} m_i \cdot \prod_{k<i} m_k. \end{aligned}$$

□

This completes part 1 of our proof plan, upper bounding the number of affine pieces polynomially in width and exponentially in depth.

The second part of the proof was to argue that Δ^L gives a high complexity, regular function: we already provided this in Proposition 5.1, which showed that Δ^L gives exactly 2^{L-1} copies of Δ, each shrunken uniformly by a factor of 2^{L-1}.

The third part is a counting argument which ensures the preceding two imply the claimed separation in L_1 distance; details are as folllows.

Proof (Proof of **??**)**.** The proof proceeds by "counting triangles".

- Draw the line $x \mapsto 1/2$ (as in the figure). The "triangles" are formed by seeing how this line intersects $f = \Delta^{L^2+2}$. There are 2^{L^2+1} copies of Δ, which means $2^{L^2+2} - 1$ (half-)triangles since we get two (half-)triangles for each Δ but one is lost on the boundary of $[0, 1]$. Each (half-)triangle has area $\frac{1}{4} \cdot \frac{1}{2^{L^2+2}} = 2^{-L^2-4}$.
- We will keep track of when g passes above and below this line; when it is above, we will count the triangles below; when it is above, we'll count the triangles below.

Summing the area of these triangles forms a lower bound on $\int_{[0,1]} |f-g|$.

- Using the earlier lemma, g has $N_A(g) \le (2 \cdot 2^L/L)^L \le 2^{L^2}$.
- For each piece, we shouldn't count the triangles at its right endpoint, or if it crosses the line, and we also need to divide by two since we're only counting triangles on one side; together

$$\begin{aligned}\int_{[0,1]} |f-g| &\ge [\text{number surviving triangles}] \cdot [\text{area of triangle}] \\ &\ge \frac{1}{2}\left[2^{L^2+2} - 1 - 2 \cdot 2^{L^2}\right] \cdot \left[2^{-L^2-4}\right] \\ &= \frac{1}{2}\left[2^{L^2+1} - 1\right] \cdot \left[2^{-L^2-4}\right] \\ &\ge \frac{1}{32}.\end{aligned}$$

□

Proof (Proof of Theorem 4.6)**.** By a bound from last lecture, $N_A(f) \le (2N/L)^L$. Using a symbolic package to differentiate, for any interval $[a,b]$,

$$\min_{(c,d)} \int_{[a,b]} (x^2 - (cx+d))^2 \mathrm{d}x = \frac{(b-a)^5}{180}.$$

Let S index the subintervals of length at least $1/(2N)$ with $N := N_A(f)$, and restrict attention to $[0,1]$. Then

$$\sum_{[a,b]\in S} (b-a) = 1 - \sum_{[a,b]\notin S} (b-a) \ge 1 - N/(2N) = 1/2.$$

Consequently,

$$\begin{aligned}\int_{[0,1]} (x^2 - f(x))^2 \mathrm{d}x &= \sum_{[a,b]\in P_A(f)} \int_{[a,b]\cap[0,1]} (x^2 - f(x))^2 \mathrm{d}x \\ &\ge \sum_{[a,b]\in S} \frac{(b-a)^5}{180} \\ &\ge \sum_{[a,b]\in S} \frac{(b-a)}{2880N^4} \ge \frac{1}{5760N^4}.\end{aligned}$$

□

4.2.1 Sobolev balls

Here we will continue and give a version of Yarotsky's main consequence to the approximation of x^2: approximating functions with many bounded derivatives (by approximating their Taylor expansions), formally an approximation result against a Sobolev ball in function space.

Remark 5.14 ***(bibliographic notes)*** This is an active area of work; in addition to the original work by (Yarotsky 2016), it's also worth highlighting the re-proof by (Schmidt-Hieber 2017), which then gives an interesting regression consequence. There are many other works in many directions, for instance adjusting the function class to lessen the (still bad) dependence on dimension (Montanelli, Yang, and Du 2020). These approaches all work with polynomials, but it's not clear this accurately reflects approximation power of ReLU networks (Telgarsky 2017).

Theorem 5.4 Suppose $g : \mathbb{R}^d \to \mathbb{R}$ satisfies $g(x) \in [0,1]$ and all partial derivatives of all orders up to r are at most M. Then there exists a ReLU network with $\mathcal{O}(k(r+d))$ layers and $\mathcal{O}((kd + d^2 + r^2 d^r + krd^r)s^d)$ nodes such that

$$\left|f(x) - g(x)\right| \leq Mrd^r \left(s^{-r} + 4d2^d \cdot 4^{-k}\right) + 3d2^d \cdot 4^{-k}. \qquad \forall x \in [0,1]^d.$$

[mjt☹: This isn't quite right; yarotsky claims a width $c(d,r)/\epsilon^{d/r} \ln(1/\epsilon)$ suffices for error ϵ; need to check what I missed.]

Remark 5.15 ***(not quite right)*** Matus note from Matus to Matus: Yarotsky gets width $c(d,r)\ln(1/\epsilon)/\epsilon^{d/r}$ and mine is worse, need to track down the discrepancy.

The proof consists of the following pieces:

1. Functions in Sobolev space are locally well-approximated by their Taylor expansions; therefore we will expand the approximation of x^2 to give approximation of general monomials in Lemma 5.4.

2. These Taylor approximations really only work locally. Therefore we need a nice way to switch between different Taylor expansions in different parts of $[0,1]^d$. This leads to the construction of a *partition of unity*, and is one of the other very interesting ideas in (Yarotsky 2016) (in addition to the construction of x^2; this is done below in Lemma 5.5.

First we use squaring to obtain multiplication.

Lemma 5.3 For any integers k, l, there exists a ReLU network $\text{prod}_{k,l} : \mathbb{R}^l \to \mathbb{R}$ which requires $\mathcal{O}(kl)$ layers and $\mathcal{O}(kl + l^2)$ nodes such that for any $x \in [0,1]^l$,

$$\left|\text{prod}_{k,l}(x) - \prod_{j=1}^{l} x_j\right| \leq l \cdot 4^{-k},$$

and $\text{prod}_{k,l}(x) \in [0,1]$, and $\text{prod}_{k,l}(x) = 0$ if any x_j is 0.

Proof. The proof first handles the case $l = 2$ directly, and uses $l - 1$ copies of $\text{prod}_{k,2}$ for the general case.
As such, for $(a, b) \in \mathbb{R}^2$, define

$$\text{prod}_{k,2}(a, b) := \frac{1}{2}\left(4h_k((a + b)/2) - h_k(a) - h_k(b)\right).$$

The size of this network follows from the size of h_k given in Theorem 5.2 (roughly following (Yarotsky 2016)), and $\text{prod}_{k,2}(a, b) = 0$ when either argument is 0 since $h_k(0) = 0$. For the approximation guarantee, since every argument to each h_k is within $[0, 1]$, then Theorem 5.2 (roughly following (Yarotsky 2016)) holds, and using the polarization identity to rewrite $a \cdot b$ gives

$$\begin{aligned}
2|\text{prod}_{k,l}(a, b) - ab| &= 2|\text{prod}_{k,l}(a, b) - \frac{1}{2}((a + b)^2 - a^2 - b^2)| \\
&\leq 4|h_k((a + b)/2) - ((a + b)/2)^2| + |h_k(a) - a^2| + |h_k(b) - b^2| \\
&\leq 4 \cdot 4^{-k-1} + 4^{-k-1} + 4^{-k-1} \leq 2 \cdot 4^{-k}.
\end{aligned}$$

Now consider the case $\text{prod}_{k,i}$ for $i > 2$: this network is defined via

$$\text{prod}_{k,i}(x_1, \ldots, x_i) := \text{prod}_{k,2}(\text{prod}_{k,i-1}(x_1, \ldots, x_{i-1}), x_i).$$

It is now shown by induction that this network has $\mathcal{O}(ki + i^2)$ nodes and $\mathcal{O}(ki)$ layers, that it evaluates to 0 when any argument is zero, and lastly satisfies the error guarantee

$$\left|\text{prod}_{k,i}(x_{1:i}) - \prod_{j=1}^{i} x_j\right| \leq i4^{-k}.$$

The base case $i = 2$ uses the explicit $\text{prod}_{k,2}$ network and gaurantees above, thus consider $i > 2$. The network embeds $\text{prod}_{k,i-1}$ and another copy of $\text{prod}_{k,2}$ as subnetworks, but additionally must pass the input x_i forward, thus requires $\mathcal{O}(ki)$ layers and $\mathcal{O}(ki+i^2)$ nodes, and evaluates to 0 if any argument is 0 by the guarantees on $\text{prod}_{k,2}$ and the inductive hypothesis. For the error estimate,

$$\begin{aligned}
\left|\text{prod}_{k,i}(x_1, \ldots, x_i) - \prod_{j=1}^{i} x_j\right| &\leq \left|\text{prod}_{k,2}(\text{prod}_{k,i-1}(x_1, \ldots, x_{i-1}), x_i) - x_i\text{prod}_{k,i-1}(x_1, \ldots, x_{i-1})\right| \\
&\quad + \left|x_i\text{prod}_{k,i-1}(x_1, \ldots, x_{i-1}) - x_i \prod_{j=1}^{i-1} x_j\right| \\
&\leq 4^{-k} + |x_i| \cdot \left|\text{prod}_{k,i-1}(x_1, \ldots, x_{i-1}) - \prod_{j=1}^{i-1} x_j\right| \\
&\leq 4^{-k} + |x_i| \cdot \left((i - 1)4^{-k}\right) \leq i4^{-k}.
\end{aligned}$$

From multiplication we get monomials.

Lemma 5.4 Let degree r and input dimension d be given, and let N denote the number of monomials of degree at most r. Then there exists a ReLU network $\mathrm{mono}_{k,r} : \mathbb{R}^d \to \mathbb{R}^N$ with $\mathcal{O}(kr)$ layers and $\mathcal{O}(d^r(kr + r^2))$ nodes so that for any vector of exponents $\vec{\alpha}$ corresponding to a monomial of degree at most r, meaning $\vec{\alpha} \geq 0$, $\sum_i \alpha_i \leq r$, and $x^{\vec{\alpha}} := \prod_{i=1}^d x_i^{\alpha_i}$, then the output coordinate of $\mathrm{mono}_{k,r}$ corresponding to $\vec{\alpha}$, written $\mathrm{mono}_{k,r}(x)_{\vec{\alpha}}$ for convenience, satisfies

$$\left|\mathrm{mono}_{k,r}(x)_{\vec{\alpha}} - x^{\vec{\alpha}}\right| \leq r4^{-k} \qquad \forall x \in [0,1]^d.$$

Proof. $\mathrm{mono}_{k,r}$ consists of N parallel networks, one for each monomial. As such, given any $\vec{\alpha}$ of degree $q \leq r$, to define coordinate $\vec{\alpha}$ of $\mathrm{mono}_{k,r}$, first rewrite α as a vector $v \in \{1, \ldots, d\}^q$, whereby

$$x^{\vec{\alpha}} := \prod_{i=1}^{q} x_{v_i}.$$

Define

$$\mathrm{mono}_{k,r}(x)_{\vec{\alpha}} := \mathrm{prod}_{k,q}(x_{v_1}, \ldots, x_{v_q}),$$

whereby the error estimate follows from Lemma 5.3, and the size estimate follows by multiplying the size estimate from Lemma 5.3 by N, and noting $N \leq d^r$.

Next we construct the approximate partition of unity.

Lemma 5.5 For any $s \geq 1$, let $\mathrm{part}_{k,s} : \mathbb{R}^d \to \mathbb{R}^{(s+1)^d}$ denote an approximate partition of unity implemented by a ReLU network, detailed as follows.

1. For any vector $v \in S := \{0, 1/s, \ldots, s/s\}^d$, there is a corresponding coordinate $\mathrm{part}_{k,s}(\cdot)_v$, and this coordinate is only supported locally around v, meaning concretely that $\mathrm{part}_{k,s}(x)_v$ is zero for $x \notin \prod_{j=1}^d [v_j - 1/s, v_j + 1/s]$.
2. For any $x \in [0,1]^d$, $|\sum_{v \in S} \mathrm{part}_{k,s}(x)_v - 1| \leq d2^d 4^{-k}$.
3. $\mathrm{part}_{k,s}$ can be implemented by a ReLU network with $\mathcal{O}(kd)$ layers and $\mathcal{O}((kd + d^2)s^d)$ nodes.

Proof. Set $N := (s+1)^d$, and let S be any enumeration of the vectors in the grid $\{0, 1/s, \ldots, s/s\}^d$. Define first a univariate bump function

$$h(a) := \sigma(sa + 1) - 2\sigma(sa) + \sigma(sa - 1) = \begin{cases} 1 + sa & a \in [-1/s, 0), \\ 1 - sa & a \in [0, 1/s] \\ 0 & \text{o.w.}. \end{cases}$$

For any $v \in S$, define

$$f_v(x) := \mathrm{prod}_{k,d}(h(x_1 - v_1), \ldots, h(x_d - v_d)).$$

By Lemma 5.3,

$$\sup_{x\in[0,1]^d} |f_v(x) - \prod_{j=1}^{d} h(x_j - v_j)| \le d4^{-k}.$$

Each coordinate of the output of $\text{part}_{k,s}$ corresponds to some $v \in S$; in particular, define

$$\text{part}_{k,s}(x)_v := f_v(x).$$

As such, by the definition of f_v, and Lemma 5.3, and since $|S| \le (s+1)^d$, then $\text{part}_{k,s}$ can be written with kd layers and $\mathcal{O}((kd+d^2)s^d)$ nodes. The local support claim for $\text{part}_{k,s}(\cdot)_v$ follows by construction. For the claim of approximate partition of unity, using $U \subseteq S$ to denote the local set of coordinates corresponding to nonzero coordinates of $\text{part}_{k,s}$ (which has $|U| \le 2^d$ by the local support claim),

$$\begin{aligned}
|\sum_{v\in S} \text{part}_{k,s}(x)_v - 1| &= |\sum_{v\in U} (\text{part}_{k,s}(x)_v - \prod_{j=1}^{d} h(x_j - v_j) + \prod_{j=1}^{d} h(x_j - v_j) - 1| \\
&\le \sum_{v\in U} |\text{part}_{k,s}(x)_v - \prod_{j=1}^{d} h(x_j - v_j)| + |\sum_{v\in U} \prod_{j=1}^{d} h(x_j - v_j) - 1| \\
&\le 2^d d4^{-k} + |\sum_{v\in U} \prod_{j=1}^{d} h(x_j - v_j) - 1|.
\end{aligned}$$

It turns out the last term of the sum is 0, which completes the proof: letting u denote the lexicographically smallest element in U (i.e., the "bottom left corner"),

$$\begin{aligned}
|\sum_{v\in U} \prod_{j=1}^{d} h(x_j - v_j) - 1| &= |\sum_{w\in\{0,1/s\}^d} \prod_{j=1}^{d} h((x-u+w)_j) - 1| \\
&= |\prod_{j=1}^{d} \sum_{w_j\in\{0,1/s\}} h((x-u+w)_j) - 1| \\
&= |\prod_{j=1}^{d} (h(x_j - u_j) + h(x_j - u_j + 1/s)) - 1|,
\end{aligned}$$

which is 0 because $z := x - u \in [0, 1/s]^d$ by construction, and using the case analysis of h gives

$$h(z_j) + h(z_j + 1/s) = (1 + sz_j) + (1 - s(z_j + 1/s)) = 1$$

as desired.

Finally we are in shape to prove Theorem 5.4.

Proof of Theorem 5.4. The ReLU network for f will combine $\text{part}_{k,s}$ from Lemma 5.5 with $\text{mono}_{k,r}$ from Lemma 5.4 via approximate multiplication, meaning $\text{prod}_{k,2}$ from

Lemma 5.3.
In detail, let the grid $S := \{0, 1/s, \dots, s/s\}^d$ be given as in the statement of Lemma 5.5. For each $v \in S$, let $p_v : \mathbb{R}^d \to \mathbb{R}$ denote the Taylor expansion of degree r at v; by a standard form of the Taylor error, for any $x \in [0,1]^d$ with $\|x - v\|_\infty \leq 1/s$,

$$|p_v(x) - g(x)| \leq \frac{Md^r}{r!}\|v - x\|_\infty^r \leq \frac{Md^r}{r!s^r}.$$

Next, let w_v denote the Taylor coefficients forming p_v, and define $f_v : \mathbb{R}^d \to \mathbb{R}$ as $x \mapsto w_v^{\mathrm{T}}\mathrm{mono}_{k,r}(x - v)$, meaning approximate p_v by taking the linear combination with weights w_v of the approximate monomials in $x \mapsto \mathrm{mono}_{k,r}(x - v)$. By Lemma 5.4, since there are at most d^r terms, the error is at most

$$|f_v(x) - p_v(x)| = |\sum_{\vec{\alpha}}(w_v)_{\vec{\alpha}}(\mathrm{mono}_{k,r}(x - v)_{\vec{\alpha}} - (x - v)^{\vec{\alpha}})| \leq \sum_{\vec{\alpha}}|(w_v)_{\vec{\alpha}}|r4^{-k} \leq Mrd^r4^{-k}.$$

[mjt☹: just realized a small issue that negative inputs might occur; can do some shifts or reflections or whatever to fix.]
The final network is now obtained by using $\mathrm{prod}_{k,2}$ to approximately multiply each approximate Taylor expansion f_v by the corresponding locally-supported approximate partition of unity element $\mathrm{part}_{k,s}(x)_v$; in particular, define

$$f(x) := \sum_{v \in S}\mathrm{prod}_{k,2}(f_v(x), \mathrm{part}_{k,s}(x)_v).$$

Then, using the above properties and the fact that the partition of unity is locally supported, letting $U \subseteq S$ denote the set of at most 2^d active elements,

$$\begin{aligned}
\big|f(x) - g(x)\big| &\leq \left|\sum_{v\in S}\mathrm{prod}_{k,2}(f_v(x), \mathrm{part}_{k,s}(x)_v) - \sum_{v\in S} f_v(x)\mathrm{part}_{k,s}(x)_v\right| \\
&+ \left|\sum_{v\in S} f_v(x)\mathrm{part}_{k,s}(x)_v - \sum_{v\in S} p_v(x)\mathrm{part}_{k,s}(x)_v\right| \\
&+ \left|\sum_{v\in S} p_v(x)\mathrm{part}_{k,s}(x)_v - \sum_{v\in S} g(x)\mathrm{part}_{k,s}(x)_v\right| \\
&+ \left|\sum_{v\in S} g(x)\mathrm{part}_{k,s}(x)_v - g(x)\right| \\
&\leq 2|U|4^{-k} + Mrd^r4^{-k}(1 + d2^d4^{-k}) + \frac{Md^r}{r!s^r}(1 + d2^d4^{-k}) + |f(x)|d2^d4^{-k} \\
&\leq Mrd^r\left(s^{-r} + 4d2^d \cdot 4^{-k}\right) + 3d2^d \cdot 4^{-k}.
\end{aligned}$$

[mjt☹: The input to $\mathrm{prod}_{k,2}$ can exceed 1. for a maximally lazy fix, I should just clip its input.]

4.3 Bibliographic notes

[todo 89/93]

From squaring we can get many other things (still with $\mathcal{O}(\ln(1/\epsilon))$ depth and size.

- Multiplication (via "polarization"):

$$(x, y) \mapsto xy = \frac{1}{2}\left((x+y)^2 - x^2 - y^2\right).$$

- Multiplications gives polynomials.
- $\frac{1}{x}$ and rational functions (Telgarsky 2017).
- Functions with "nice Taylor expansions" (Sobolev spaces) (Yarotsky 2016); though now we'll need size bigger than $\ln \frac{1}{\epsilon}$:
  - First we approximate each function locally with a polynomial.
  - We multiply each local polynomial by a bump ((Yarotsky 2016) calls the family of bumps a "partition of unity").
  - This was also reproved and connected to statistics questions by (Schmidt-Hieber 2017).

Remark 5.5 ***(bibliographic notes)*** Theorem 5.1 ((Telgarsky 2015, 2016)) was the earliest proof showing that a deep network can not be approximated by a reasonably-sized shallow network, however prior work showed a separation for *exact* representation of deep *sum-product networks* as compared with shallow ones (Bengio and Delalleau 2011). A sum-product network has nodes which compute affine transformations or multiplications, and thus a multi-layer sum-product network is a polynomial, and this result, while interesting, does not imply a ReLU separation.
As above, step 1 of the proof upper bounds the total possible number of affine pieces in a univariate network of some depth and width, and step 2 constructs a deep function which roughly meets this bound. Step 1 can be generalized to the multivariate case, with reasoning similar to the VC-dimension bounds in +(**sec:vc?**). A version of step 2 appeared in prior work but for the multivariate case, specifically giving a multivariate-input network with exponentially many affine pieces, using a similar construction (Montúfar et al. 2014). A version of step 2 also appeared previous as a step in a proof that recurrent networks are Turing complete, specifically a step used to perform digit extraction (Siegelmann and Sontag 1994, fig. 3).

It is natural and important to wonder if this exponential increase is realized in practice. Preliminary work reveals that, at least near initialization, the effective number of pieces is much smaller (Hanin and Rolnick 2019).

Remark 4.12 (Other depth separations)**.** • Our construction was univariate. Over $\mathbb{R}^d$, there exist ReLU networks with poly(d) notes in 2 hidden layers which

can not be approximated by 1-hidden-layer networks unless they have $\geq 2^d$ nodes (Eldan and Shamir 2015).

- The 2-hidden-layer function is approximately radial; we also mentioned that these functions are difficult in the Fourier material; the quantity $\int \|w\| \cdot |\hat{f}(w)| \mathrm{d}w$ is generally exponential in dimension for radial functions.
- The proof by (Eldan and Shamir 2015) is very intricate; if one adds the condition that weights have subexponential size, then a clean proof is known (Daniely 2017).
- Other variants of this problem are open; indeed, there is recent evidence that separating constant depth separations is hard, in the sense of reducing to certain complexity theoretic questions (Vardi and Shamir 2020).

◇

- A variety of works consider connections to tensor approximation and sum product networks (Cohen and Shashua 2016; Cohen, Sharir, and Shashua 2016).
- Next we will discuss the approximation of x^2.
- Last one can be beefed up to a lower bound against strongly convex functions. *[todo 90/93]*

From squaring we can get many other things (still with $\mathcal{O}(\ln(1/\epsilon))$ depth and size.

- Multiplication (via "polarization"):

$$(x, y) \mapsto xy = \frac{1}{2}\left((x+y)^2 - x^2 - y^2\right).$$

- Multiplications gives polynomials.
- $\frac{1}{x}$ and rational functions (Telgarsky 2017).
- Functions with "nice Taylor expansions" (Sobolev spaces) (Yarotsky 2016); though now we'll need size bigger than $\ln \frac{1}{\epsilon}$:
  - First we approximate each function locally with a polynomial.
  - We multiply each local polynomial by a bump ((Yarotsky 2016) calls the family of bumps a "partition of unity").
  - This was also reproved and connected to statistics questions by (Schmidt-Hieber 2017).

Theorem 5.3 ***(sketch, from (Yarotsky 2016; Schmidt-Hieber 2017))*** Suppose $f : \mathbb{R}^d \to \mathbb{R}$ has all coordinates of all partial derivatives of order up to r within $[-1, +1]$ and let $\epsilon > 0$ be given. Then there exists a $\widetilde{\mathcal{O}}(\ln(1/\epsilon)$ layer and $\widetilde{\mathcal{O}}(\epsilon^{-d/r})$ width network so that

$$\sup_{x \in [0,1]^d} |f(x) - g(x)| \leq \epsilon.$$

[mjt☹: gross and vague, i should clean]

Remark 5.13 There are many papers following up on these; e.g., crawl the citation graph outwards from (Yarotsky 2016).

4.4 Bibliographic notes

can say essentially three categories for separations: multi-layer, 2-to-3, and polynomial.

4.5 Exercises

4.5.1 Research questions

Research question 4.1. *signal-to-noise in multi-layer case*

Research question 4.2. *norm-based repr results*

Research question 4.3. *other arch*

Research question 4.4. *characterize multi-layer multi-variate easy-to-repr fucntions.*

Research question 4.5. *proper depth sep (L vs $L + 1$)*

Part II

Optimization

Chapter 5

Optimization near initialization

Chapter 6

Strongly convex NTK

Chapter 7

Margins and feature learning

Part III

Generalization

Part IV

Other topics

Chapter 8

Distribution modeling

Bibliography

Keith M. Ball. An elementary introduction to modern convex geometry. In *Flavors of Geometry*, Mathematical Sciences Research Institute Publications, pages 1–58. Cambridge University Press, 1997.

Andrew R. Barron. Universal approximation bounds for superpositions of a sigmoidal function. *IEEE Transactions on Information Theory*, 39(3):930–945, May 1993.

Mikhail Belkin, Daniel J Hsu, and Partha Mitra. Overfitting or perfect fitting? risk bounds for classification and regression rules that interpolate. *Advances in neural information processing systems*, 31, 2018.

Alberto Bietti and Francis Bach. Deep equals shallow for relu networks in kernel regimes. 2020. `arXiv:2009.14397 [stat.ML]`.

Ronen Eldan and Ohad Shamir. The power of depth for feedforward neural networks. 2015. `arXiv:1512.03965 [cs.LG]`.

Gerald B. Folland. *Real analysis: modern techniques and their applications*. Wiley Interscience, 2 edition, 1999.

Leonid Gurvits and Pascal Koiran. Approximation and learning of convex superpositions. In Paul Vitányi, editor, *Computational Learning Theory*, pages 222–236. Springer, 1995.

K. Hornik, M. Stinchcombe, and H. White. Multilayer feedforward networks are universal approximators. *Neural Networks*, 2(5):359–366, july 1989.

Ziwei Ji, Matus Telgarsky, and Ruicheng Xian. Neural tangent kernels, transportation mappings, and universal approximation. In *ICLR*, 2020.

Lee K. Jones. A simple lemma on greedy approximation in hilbert space and convergence rates for projection pursuit regression and neural network training. *The Annals of Statistics*, 20(1):608–613, 03 1992.

John Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Žídek, Anna Potapenko, et al. Highly accurate protein structure prediction with alphafold. *Nature*, 596(7873):583–589, 2021.

Holden Lee, Rong Ge, Tengyu Ma, Andrej Risteski, and Sanjeev Arora. On the ability of neural nets to express distributions. In *COLT*, 2017.

Behnam Neyshabur, Ryota Tomioka, and Nathan Srebro. In search of the real inductive bias: On the role of implicit regularization in deep learning. `arXiv:1412.6614 [cs.LG]`, 2014.

Gilles Pisier. Remarques sur un résultat non publié de b. maurey. *Séminaire Analyse fonctionnelle (dit)*, pages 1–12, 1980.

Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, 2022.

Ingo Steinwart and Andreas Christmann. *Support Vector Machines*. Springer, 1 edition, 2008.

Ulrike von Luxburg and Olivier Bousquet. Distance-based classification with lipschitz functions. *Journal of Machine Learning Research*, 2004.

Dmitry Yarotsky. Error bounds for approximations with deep relu networks. 2016. `arXiv:1610.01145 [cs.LG]`.

Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. *ICLR*, 2017.

Part V

Appendix

Appendix A

Technical background

A.1 Convexity

A.2 Miscellaneous inequalities

A.3 Probability

[todo 91/93]

A.4 Clarke differentials

[todo 92/93]
[todo 93/93]

A.5 Mirror descent and the perceptron algorithm