# *An analysis of training and generalization errors in shallow and deep networks*

# An analysis of training and generalization errors in shallow and deep networks

H. N. Mhaskar* and T. Poggio †

**Abstract**

This paper is motivated by an open problem around deep networks, namely, the apparent absence of overfitting despite large over-parametrization which allows perfect fitting of the training data. In this paper, we analyze this phenomenon in the case of regression problems when each unit evaluates a periodic activation function. We argue that the minimal expected value of the square loss is inappropriate to measure the generalization error in approximation of compositional functions in order to take full advantage of the compositional structure. Instead, we measure the generalization error in the sense of maximum loss, and sometimes, as a pointwise error. We give estimates on exactly how many parameters ensure both zero training error as well as a good generalization error. We prove that a solution of a regularization problem is guaranteed to yield a good training error as well as a good generalization error and estimate how much error to expect at which test data.

**Keywords:** Deep learning, generalization error, interpolatory approximation

## 1 Introduction

The main problem of machine learning is the following. Given data $(\mathbf{x}, y)$ sampled from an unknown probability distribution $\mu$, the goal is to find a function $P$ that minimizes the generalization error $\mathbb{E}_\mu((y - P(\mathbf{x}))^2)$ among all functions in some function class that is thought to represent the prior information about the distribution. Since we do not know $\mu$, classical machine learning paradigm expresses the generalization error as a sum of three components: the variance, the approximation error, and the sampling error. The variance is a lower bound on the generalization error, and the estimation typically focuses on the other two errors. The sum of these two is given by $\mathbb{E}((f - P(\mathbf{x}))^2)$, where the expectation is with respect to the marginal distribution of $\mathbf{x}$ and the *target function* $f$ is the conditional expectation of $y$ given $\mathbf{x}$. If the marginal distribution of $\mathbf{x}$ is known, then the split between approximation and sampling errors is no longer necessary, and one can obtain estimates as well as constructions directly from characteristics of the training data (e.g., [11, 14, 18]). In the classical paradigm where this distribution is not known, the approximation error decreases as the number of parameters in $P$ increases to $\infty$. However, this makes the empirical risk minimization problem harder to solve; making it essential to choose the number of parameters in $P$ to balance the two errors. In turn, this explains a commonly observed phenomenon that if one achieves a zero empirical risk on the training data by over-parametrized model $P$, the test error is generally not good.

There are several recent papers that demonstrate that this phenomenon is often not observed (e.g., [10, 22, 25, 29, 2, 23]). There are a few recent papers that address this issue in the case of classification problems. Belkin, Hsu, and Mitra [1] analyze the "excess error" in least square fits by piecewise linear interpolants over that obtained by the optimal Bayes' classifier. In [23, 24], the question is analyzed from the perspective of the geometry of the error surface with respect to different loss functions near the local extrema. In particular, it is shown in [23] that substituting the ReLU activation function by a polynomial approximation exhibits the same behavior as the original network.

In this paper, we focus on regression problems. We wish to consider from the point of view of approximation theory the overfitting puzzle for universal approximation in a more intrinsic and theoretical manner, independent

*Institute of Mathematical Sciences, Claremont Graduate University, Claremont, CA 91711. The research of this author is supported in part by the Office of the Director of National Intelligence (ODNI), Intelligence Advanced Research Projects Activity (IARPA), via 2018-18032000002. email: hrushikesh.mhaskar@cgu.edu

†Center for Brains, Minds, and Machines, McGovern Institute for Brain Research, Massachusetts Institute of Technology, Cambridge, MA, 02139. The research of this author is supported by the Center for Brains, Minds and Machines (CBMM), funded by NSF STC award CCF-1231216. tp@mit.edu

of the training mechanism used. This paper does not seek to "explain" the overfitting phenomenon as observed. Such an explanation needs to involve not just the analysis of the approximation problem itself, but also an analysis of the training algorithms used for this purpose. It is obvious that a network (or any other model) using a number of parameters that is less than the number of training data points cannot in general produce a zero training error in the absence of some strong prior knowledge about the target function that generated the training data, no matter what training algorithm is used. Our goal is to study the fundamental problem of function approximation to examine what characteristics of the data and how much overparametrization will give **theoretical guarantees** that the generalization error can be controlled while achieving a zero training error. We will do this without any prior assumption about the target function apart from continuity or at most differentiability.

We wish to address the following issues about the phenomenon of zero/small training error and small test error for universal approximation:

1. What characteristics of the data govern a zero training error and a good generalization error?
2. How much over-parametrization will give a **mathematical guarantee** of the model to exhibit this behavior?
3. Propose a regularization scheme whose solution will guarantee a good (but not necessarily zero) training error while maintaining a good generalization error at the same time. The emphasis here is on an estimation of how much over-parametrization is necessary to get theoretical guarantees.
4. What bounds on the generalization error can be guaranteed by the solution of the (global) regularization scheme **at each point** in the test data (rather than a global error estimate), compared to the nearest neighbor in the training data?

In recent years, convolutional neural networks (CNNs) have achieved a revolution in machine learning. A good survey can be found in [12]. From a practical point of view, the central features of CNNs are locality and weight sharing. From a strictly mathematical point of view, convolution is a very special operation that requires a group structure on the data. According to the book [8], the CNNs "are a specialized kind of neural network for processing data that has a known, grid-like topology. Examples include time-series data, which can be thought of as a 1D grid taking samples at regular time intervals, and image data, which can be thought of as a 2D grid of pixels." For this kind of data, it is customary to treat it either as data on the whole real line/plane with zero-padding, or otherwise use a symmetrical extension to treat it as data on a circle/torus so that the standard group operations on these spaces can be used to define the operation of convolution.

In this paper, we will focus on function approximation on the torus. The most fundamental class for this purpose is the class of all trigonometric polynomials. Accordingly, we will study the problem of the lack of overfitting in the context of approximation by trigonometric polynomials. We will explain in Section 2.2 the theoretical relationship between trigonometric polynomials and neural/RBF networks in the periodic setting. In Section 3, we will show how the theorems about trigonometric approximation translate into theorems about shallow networks with arbitrary periodic activation functions.

In the study of approximation error in deep learning, it is observed in [20] that the compositional structure of the target function can be utilized effectively via a property called good propagation of error to obtain substantially better error bounds allowing us to overcome the curse of dimensionality observed in shallow networks. This allows us to "lift" many results on approximation by shallow networks to those on deep networks. However, there are a few barriers that prevent a straightforward extension.

One is that we do not know the functions evaluated at each node in the intervening layers; just the input/output relations between the data and output of the ultimate layers. There are some recent efforts in the direction of designing deep networks in some special applications using domain knowledge (e.g., [9, 19]). It is conceivable that this problem does not appear in these contexts.

The other problem is even more fundamental, requiring a change in the traditional notion of generalization error. For example, suppose we wish to approximate a function $f^* = f(f_1(\mathbf{x}_1), f_2(\mathbf{x}_2))$ by a network of the form $P^* = P(P_1(\mathbf{x}_1), P_2(\mathbf{x}_2))$, where $\mathbf{x}_1, \mathbf{x}_2 \in \mathbb{R}^d$. Without the compositional structure, $f^*$ has to be treated as a function of $2d$ variables. The compositional structure allows us to treat the approximation problem as a set of three approximation problems: approximating functions $f_1$, $f_2$ as functions of $d$ variables each, and a function $f$ of 2 variables.

How do we define generalization error? Defining it in terms of the original distribution of $((\mathbf{x}_1, \mathbf{x}_2), y)$ is not sensitive to the compositional structure. On the other hand, the input $(f_1, f_2)$ to the function $f$ is not the same (even may not have the same distribution) as the input $(P_1, P_2)$ to the approximation $P$.

Therefore, we measure in this paper the errors in the uniform norm rather than searching for appropriate $L^2$ norms suitable for the compositionality structure of the target function. Thus, we define the generalization error as

the maximal error between the target function and the model at all possible test points. One consequence is that the decomposition of the generalization error into three parts breaks down. Therefore, new ideas are required to achieve the approximation in terms of the training data alone. The approximation errors themselves are studied in [20, 15], but the techniques suggested there require the data points $\mathbf{x}_j$ to be sufficiently dense in the domain (Euclidean space, sphere, cube, etc.). When the data is not dense, it is clearly not expected to get a good approximation on the whole domain. However, if the data does become denser and denser on some subset of the domain, one can expect a good approximation at points close by to the training data. Thus, we will establish pointwise bounds for the generalization error obtained by a solution of a regularization scheme suggested for this purpose.

In Section 2, we develop some notation and provide some background information that has motivated our current paper. In Section 3, we state our theorems for the case of shallow networks. In Section 4, we state the analogues of the results in Section 3 in the case of deep networks. The proofs of the results in Sections 3 and 4 are given in Section 5.

# 2 Background

The purpose of this section is to explain the connection between trigonometric polynomials and neural networks (Section 2.2) as well as to explain a classical theorem which provides a motivation for our theorems in this paper (Section 2.3). In order to do so, we need first to develop some notation. This is done in Section 2.1.

## 2.1 Notation

Let $q \ge 1$ be an integer, $\mathbb{T}^q$ be the $q$ dimensional torus ($=\mathbb{R}^q/(2\pi\mathbb{Z})^q$). For $\mathbf{x}, \mathbf{y} \in \mathbb{T}^q$,

$$|\mathbf{x}-\mathbf{y}| = \max_{1\le i\le q} |(x_i - y_i)(\bmod 2\pi)|.$$

For a multi-integer $\mathbf{k}$, $|\mathbf{k}|_p$ is the $\ell^p$ norm of $\mathbf{k}$.

The space of all continuous functions $f : \mathbb{T}^q \to \mathbb{R}$, equipped with the supremum norm will be denoted by $C^*$ (or $C^*(\mathbb{T}^q)$ if we wish to emphasize the input dimension to the functions). The norm on $C^*(\mathbb{T}^q)$ will be denoted by $\|\cdot\|$ or $\|\cdot\|_q$ if it is important to identify the dimension. For $n > 0$, the space $\mathbb{H}_n^q$ of trignometric polynomials in $q$ variables with (spherical) degree $< n$ is defined by

$$\mathbb{H}_n^q = \mathsf{span}\{\exp(i\mathbf{k}\cdot\circ) : |\mathbf{k}|_2 < n\}.$$

The dimension of $\mathbb{H}_n^q$ is $\sim n^q$. If $f \in C^*(\mathbb{T}^q)$, then its Fourier coefficients are defined by

$$\hat{f}(\mathbf{k}) = \frac{1}{(2\pi)^q}\int_{\mathbb{T}^q} f(\mathbf{x})\exp(-i\mathbf{k}\cdot\mathbf{x})d\mathbf{x}, \qquad \mathbf{k}\in\mathbb{Z}^q, \tag{2.1}$$

and its degree of approximation from $\mathbb{H}_n^q$ is defined by

$$E_n(f) = E_n(q; f) := \inf_{T\in\mathbb{H}_n^q} \|f - T\|.$$

When our models are trigonometric polynomials in $\mathbb{H}_n^q$, the quantity $E_n(f)$ denotes the ideal generalization error for the target function $f$.

Let $h : \mathbb{R} \to [0, 1]$ be an infinitely differentiable, even function such that $h(t) = 1$ if $|t| \le 1/2$, $h(t) = 0$ if $|t| \ge 1$. We define

$$\Phi_N(\mathbf{x}) = \sum_{\mathbf{k}\in\mathbb{Z}^q} h\left(\frac{|\mathbf{k}|_2}{N}\right)\exp(i\mathbf{k}\cdot\mathbf{x}), \qquad \mathbf{x}\in\mathbb{T}^q,\ N > 0. \tag{2.2}$$

For $f \in C^*$, we define

$$\sigma_N(f)(\mathbf{x}) = \sum_{\mathbf{k}\in\mathbb{Z}^q} h\left(\frac{|\mathbf{k}|_2}{N}\right)\hat{f}(\mathbf{k})\exp(i\mathbf{k}\cdot\mathbf{x}), \qquad \mathbf{x}\in\mathbb{T}^q,\ N > 0. \tag{2.3}$$

We note that the sums in both (2.2) and (2.3) are finite sums, although they are written as infinite sums for convenience of notation.

For any finite subset $\mathcal{C} \subset \mathbb{T}^q$, we define its minimal separation by

$$\eta(\mathcal{C}) = \min_{\mathbf{x},\mathbf{y}\in\mathcal{C},\mathbf{x}\neq\mathbf{y}} |\mathbf{x}-\mathbf{y}|. \tag{2.4}$$

We assume a training data of the form $\mathcal{D} = \{(\mathbf{x}_j, y_j)\}_{j=1}^M$, where $\mathcal{C} = \{\mathbf{x}_j\}_{j=1}^M \subset \mathbb{T}^q$, and $y_j = f(\mathbf{x}_j) + \epsilon_j$ for some $f \in C^*$. We denote

$$\epsilon = \max_{1\le j\le M} |\epsilon_j|. \tag{2.5}$$

The quantity $\epsilon$ plays the role of variance in our theory in this paper. In regression problems, where numerical accuracy is expected in the supremum norm, the errors $\epsilon_j$ all need to be small. Thus, even though the quantity $\epsilon$ seems to increase with $M$, it remains small as $M \to \infty$. One example is when the values of $f$ are not computed exactly (as they rarely are), but only through a numerical computation (for example, $f$ is the ideal solution of a differential equation, but the data is obtained by solving this equation numerically at the grid points). The quantity $\epsilon$ then represents the maximal error in this numerical computation.

**The constant convention**
*The symbols $c, c_1, \cdots$ will denote generic positive constants, depending on such fixed parameters of the problem as $q$, $h$, $\mathcal{G}$, and $S$ (to be introduced later), etc. and other quantities explicitly indicated, but their values may be different at different occurrences, even within a single formula. The notation $A \sim B$ means that $c_1 A \le B \le c_2 A$.*

## 2.2 Neural networks and trigonometric polynomials

The material in this section is based on [16]. For reasons that will become clear shortly, the term *activation function* in this paper will mean $\phi \in C^*$ such that $\hat{\phi}(1) \neq 0$. We note that a trigonometric polynomial is itself a neural network with the activation function $t \mapsto \cos t$. There is a close connection between trigonometric polynomials and networks with other activation functions. Let $\phi \in C^*(\mathbb{T})$ and $\hat{\phi}(1) \neq 0$. Then for $\mathbf{k} \in \mathbb{Z}^q$ and $\mathbf{x} \in \mathbb{T}^q$, it is not difficult to verify that

$$\exp(i\mathbf{k}\cdot\mathbf{x}) = \frac{1}{2\pi\hat{\phi}(1)} \int_{\mathbb{T}} \phi(t) \exp\left(i(\mathbf{k}\cdot\mathbf{x} - t)\right) dt = \frac{1}{2\pi\hat{\phi}(1)} \int_{\mathbb{T}} \phi(\mathbf{k}\cdot\mathbf{x} - t) \exp(it) dt.$$

Discretizing the integral expression above, it can be shown (cf. [13, Proposition 4.2.1]) that for any integer $N \ge 1$, $\mathbf{k} \in \mathbb{Z}^q$,

$$\left\| \exp(i\mathbf{k}\cdot(\circ)) - \frac{1}{(2N+1)\hat{\phi}(1)} \sum_{j=0}^{2N} \exp\left(\frac{2\pi ij}{2N+1}\right) \phi\left(\mathbf{k}\cdot(\circ) - \frac{2\pi j}{2N+1}\right) \right\| \le \frac{4}{|\hat{\phi}(1)|} E_N(1;\phi). \tag{2.6}$$

In particular, if $T$ is a trigonometric polynomial, let

$$\mathbb{G}_N(\phi, T)(\mathbf{x}) = \frac{1}{(2N+1)\hat{\phi}(1)} \sum_{j=0}^{2N} \exp\left(\frac{2\pi ij}{2N+1}\right) \left( \sum_{\mathbf{k}\in\mathbb{Z}^q} \hat{T}(\mathbf{k}) \phi\left(\mathbf{k}\cdot(\circ) - \frac{2\pi j}{2N+1}\right)\right), \tag{2.7}$$

where it is understood that $\hat{T}(\mathbf{k}) = 0$ if $|\mathbf{k}|_2$ exceeds the degree $n$ of $T$. The number of neurons involved in the network $\mathbb{G}_N(\phi, T)$ is $\sim Nn^q$. The estimate (2.6) leads to

$$\|T - \mathbb{G}_N(\phi, T)\| \le \frac{4}{|\hat{\phi}(1)|} E_N(1;\phi) \sum_{\mathbf{k}\in\mathbb{Z}^q} |\hat{T}(\mathbf{k})|. \tag{2.8}$$

Thus, a trigonometric polynomial can be approximated by a neural network with activation function $\phi$ and the error bounds can be obtained by keeping track of the degree of approximation of the target function by trigonometric polynomials, the magnitude of its Fourier coefficients and the bound in (2.8) (cf. [16] for some examples). This leads to the following proposition, which will be proved in Section 5.

**Proposition 2.1** *Let $\phi \in C^*(\mathbb{T})$, $\hat{\phi}(1) \neq 0$, $f \in C^*$. Then for $n, N \geq 1$, we have*

$$\begin{aligned}\|f - \mathbb{G}_N(\phi, \sigma_n(f))\| &= \left\| f - \frac{1}{(2N+1)\hat{\phi}(1)} \sum_{j=0}^{2N} \exp\left(\frac{2\pi i j}{2N+1}\right) \left( \sum_{|\mathbf{k}|_2 < n} h\left(\frac{|\mathbf{k}|_2}{n}\right) \hat{f}(\mathbf{k}) \phi\left(\mathbf{k} \cdot (\circ) - \frac{2\pi j}{2N+1}\right)\right)\right\| \\ &\leq c(\phi) \left\{ E_{n/2}(q; f) + n^{q/2} E_N(1; \phi) \|f\| \right\}. \end{aligned} \tag{2.9}$$

**Example 2.1** For example, we consider the smooth ReLU function $t \mapsto \log(1 + e^t) = t_+ + \mathcal{O}(e^{-|t|})$. Then the function $\psi(t) = \log\left(\frac{(1+e^{t+\pi})(1+e^{t-\pi})}{(1+e^t)^2}\right)$ is integrable on $\mathbb{R}$. The periodization

$$\phi(t) = \sum_{j \in \mathbb{Z}} \psi(t + 2\pi j), \qquad t \in \mathbb{R}, \tag{2.10}$$

is an analytic function on $\mathbb{T}$. So, Bernstein approximation theorem [28, Theorem 5.4.2] shows that there exists $\rho_1 < 1$ with $E_N(1; \phi) \leq \rho_1^N$ for all $N \geq c(\phi)$. In Proposition 2.1, if $f$ satisfies $E_n(q; f) = \mathcal{O}(n^{-\gamma})$ for some $\gamma$, then we may choose $N \sim \log n$ to get a network with $\mathcal{O}(n^q \log n)$ neurons to obtain an estimate $\mathcal{O}(n^{-\gamma})$ on the right hand side of (2.9). ■

This idea is generalized to many other settings, and algorithms are known to find the approximation to the target function using the training data, **without assuming any prior on the target function** (see, e.g. [13] for an early construction). However, formulating the problem directly as a minimization of the supremum norm error between the function and the neural network model may not work. The theory implies certain relationships between the coefficients and the weights and thresholds.

Conversely, one can approximate $\phi$ by trigonometric polynomials. Therefore, if one can obtain or assume some bounds on the coefficients of a neural network with $\phi$ as the activation function, then these bounds can be translated to bounds on the degree of approximation by trigonometric polynomials. This part is hard to do on the torus with neural networks, but has been done in a far more general setting with kernel based approximation [14], where Mercer expansions satisfying certain technical conditions are known.

In view of this close relationship between general neural networks and trigonometric polynomials (i.e., networks with activation function $t \mapsto \cos t$), we will focus in this paper on trigonometric polynomials, and demonstrate in Section 3 how these results are translated to those with other neural networks.

## 2.3 Interpolatory approximation

In the language of classical approximation theory, the problem of achieving a zero training error is the problem of interpolation. In the context of trigonometric polynomials, for any data of the form $\{(\theta_j, y_j)\}_{j=0}^{2n}$, $y_j \in \mathbb{R}$, $\theta_j \in \mathbb{T}$, and $\theta_j \neq \theta_k$ if $j \neq k$, there exists $T \in \mathbb{H}_n^1$ such that $T(\theta_j) = y_j$ for $j = 0, \cdots, 2n$ [30, Chapter X, Theorem 1.2]. Thus, it is easy to obtain a zero training error with a minimal number of free parameters. As we argued in the introduction, the test error in this context should be measured in terms of uniform approximation to the target function. A well known theorem attributed in [21] to Faber and Bernstein states that for **any** system of interpolation nodes, there exists a function $f \in C^*(\mathbb{T})$ for which the sequence of interpolatory trigonometric polynomials (with minimal degree as above) does not converge uniformly to $f$.

In 1943, Erdős [6] initiated (in the context of algebraic polynomials) a study of the question whether one can get a convergent sequence of interpolatory polynomials if one allows a polynomial of higher than minimal degree. A positive answer was given by Szabados in [27]. Although the answer is given in terms of algebraic polynomials, Szabados remarks that the same is clearly true for trigonometric polynomials as well. An explicit statement to this effect is the following [17, Theorem 3.1(a)].

**Theorem 2.1** *Let $\theta_1, \ldots, \theta_N$ be distinct points in $[-\pi, \pi]$, $\theta_{N+1} := \theta_1 + 2\pi$, $\alpha > 0$, and*

$$\min_{1 \leq k \leq N} |\theta_{k+1} - \theta_k| =: \eta.$$

*Then for $f \in C^*(\mathbb{T}^1)$, there exists a trigonometric polynomial $T$ of degree at most $(1 + 2/\alpha)(\pi/\eta)$ such that $f(\theta_j) = T(\theta_j)$, $1 \leq j \leq N$, and*

$$\|f - T\| \leq (2 + \alpha) E_m(1; f) \tag{2.11}$$

*where $m = (1 + 2/\alpha)(\pi/\eta)$.*

**Remark 2.1** A curious feature of Theorem 2.1 is that one obtains the bound (2.11) on the generalization on the entire torus $\mathbb{T}^1$ without requiring that the training data $\mathcal{C}$ be "dense" in $\mathbb{T}^1$.

# 3 Shallow networks

Our first theorem in this section shows the connection between the structural properties of the training data and the construction of a trigonometric polynomial $T_N^{\#}(\mathcal{D})$ that can interpolate the noisy data (i.e., achieve a zero training error), as well as achieve a good generalization error in the sense defined in Section 1.

Before stating our main results, we first discuss two straightforward ideas. The first of these is to construct a trigonometric interpolatory polynomial $\mathcal{I}(\mathcal{D})$ of minimal degree. The other is to use a larger degree $N$, and solve the system of equations (cf. (2.2))

$$\sum_{j=1}^{M} a_j \Phi_N(\mathbf{x}_\ell - \mathbf{x}_j) = y_\ell, \qquad \ell = 1, \cdots, M. \tag{3.1}$$

For a sufficiently large value of $N$, one can show that this sytem of equations has a unique and stable solution. We denote the corresponding polynomial by

$$\mathcal{L}_N(\mathcal{D})(\mathbf{x}) = \sum_{j=1}^{M} a_j \Phi_N(\mathbf{x} - \mathbf{x}_j). \tag{3.2}$$

We examine these constructions using a univariate example. We consider $f(x) = |\cos x|$, and consider only the case when exact samples of $f$ are known at 128 points. We note that for $x \in \mathbb{T}$,

$$f(x) = \frac{2}{\pi}\left\{1 - \sum_{k=1}^{\infty} \frac{(-1)^k}{4k^2-1}\cos(2kx)\right\}.$$

Therefore, the sequence $\{(k^2+1)^{s/2}\hat{f}(k)\}$ represents a sequence of Fourier coefficients of a continuous function for all $s < 1$, but not if $s = 1$ (and of course, not for any $s > 1$). In particular, there is no "optimal" class with which to use the results of [3] for this function.

**Example 3.1** We consider $M = 128$, $x_j = -\pi + 2\pi j/M$, $j = 0, \cdots, 127$, and denote the corresponding data by $\mathcal{D}_1$. We will refer to this choice as the **dense case**. Figure 1 shows the errors at 1024 points on $[-\pi, \pi]$ for the operators $\mathcal{I}(\mathcal{D}_1) \in \mathbb{H}^1_{64}$ and $\mathcal{L}_N(\mathcal{D}_1) \in \mathbb{H}^1_{128}$ with $N = 128$. It is noted that the collocation matrix for computing $\mathcal{I}(\mathcal{D}_1)$ is very ill-conditioned, but the matrix for computing $\mathcal{L}_N(\mathcal{D}_1)$ is well-conditioned. Of course, in this very special example, one can compute $\mathcal{I}(\mathcal{D}_1)$ explicitly without having to solve a system of equations, but the intent of this example is to demonstrate the need to have methods more sophisticated than a straightforward interpolation. ■

The situation is quite different when the data is not dense on $[-\pi, \pi)$.

**Example 3.2** We consider $M = 128$, $x_j = \pi/4 + \pi j/(2M)$, $j = 0, \cdots, 127$, and denote the corresponding data by $\mathcal{D}_2$. We will refer to this choice as the **non-dense case**. Figure 2 shows the errors at 1024 points on $[-\pi, \pi]$ for the operators $\mathcal{I}(\mathcal{D}_2)$ and $\mathcal{L}_N(\mathcal{D}_2)$ with $N = 128$, and $N = 256$. As before, the collocation matrix for computing $\mathcal{I}(\mathcal{D}_2)$ is very ill-conditioned, but the matrix for computing $\mathcal{L}_N(\mathcal{D}_2)$ is well-conditioned. However, a comparison between the middle and right figure of Figure 2 shows the critical importance of choosing a degree higher than the minimal possible, commensurate with the minimal separation among the interpolation nodes. Also, we note in the right figure that the polynomial $\mathcal{L}_{256}(\mathcal{D}_2)$ is highly localized, so that the errors on the interval $[\pi/4, \pi/2)$ are small, but those away from this interval are not; in fact, the polynomial is close to 0 away from this interval. ■

As remarked in Remark 2.1, it is not possible to achieve a bound analogous to (2.11) unless the training data is sufficiently dense on $\mathbb{T}^q$ or unless more information about the target function is used in addition to its values at the training data. Therefore, let us now assume that the Fourier coefficients $\hat{f}(\mathbf{k})$ are known for all $\mathbf{k}$ with $|\mathbf{k}|_2 < n$

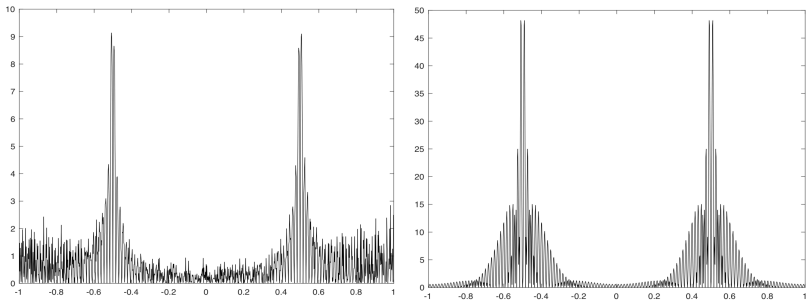

Figure 1: With $x_j = -\pi + 2\pi j/M$, $j = 0, \cdots, 127$, the differences at 1024 equidistant points of $[-\pi, \pi)$ between the true values $|\cos x|$ and the interpolatory polynomial of minimal degree, $\mathcal{I}(\mathcal{D}_1)(x)$ (left), and interpolation obtained by a higher degree localized kernel $\mathcal{L}_{128}(\mathcal{D}_1)(x)$ (right). The errors are magnified 1000 times and the points on the $x$-axis are multiples of $\pi$. The errors are similar near $\pm\pi/2$, but decrease rapidly to 0 away from these in the figure on the right.

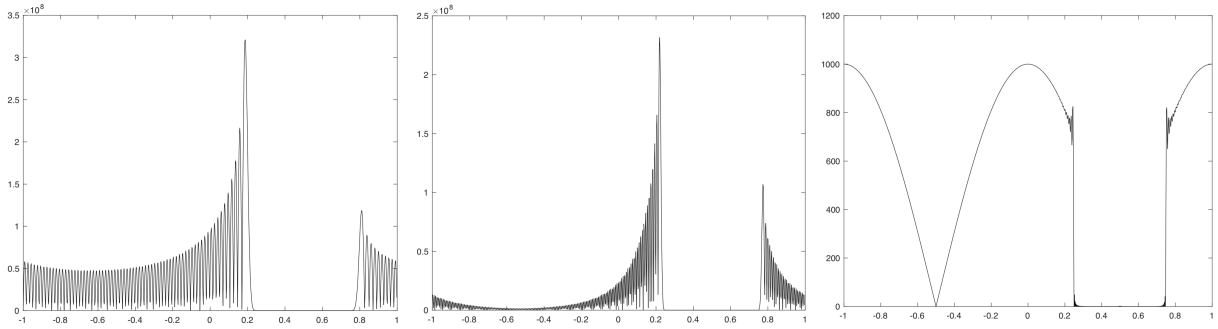

Figure 2: With $x_j = \pi/4 + \pi j/(256)$, $j = 0, \cdots, 127$, the differences at 1024 equidistant points of $[-\pi, \pi)$ between the true values $|\cos x|$ and the interpolatory polynomial of minimal degree, $\mathcal{I}(\mathcal{D}_2)(x)$ (left), interpolation obtained by a higher degree localized kernel $\mathcal{L}_{128}(\mathcal{D}_2)(x)$ (middle), and $\mathcal{L}_{256}(\mathcal{D}_2)(x)$ (right). The errors are magnified 1000 times and the points on the $x$-axis are multiples of $\pi$.

for some $n > 0$. With this information, we can construct $\sigma_n(f)$ using (2.3) with $n$ in place of $N$. For a sufficiently large value of $N$, we may solve a system of equations

$$\sum_{j=1}^{M} a_j \Phi_N(\mathbf{x}_\ell - \mathbf{x}_j) = y_\ell - \sigma_n(f)(\mathbf{x}_\ell), \qquad \ell = 1, \cdots, M. \tag{3.3}$$

We then define

$$\mathcal{T}^{\#}_{n,N}(\mathcal{D})(\mathbf{x}) = \sigma_n(f)(\mathbf{x}) + \sum_{j=1}^{M} a_j \Phi_N(\mathbf{x} - \mathbf{x}_j). \tag{3.4}$$

**Example 3.3** We continue the examples with the target functions as in Examples 3.1 and 3.2. We estimate $\hat{f}(k)$ using a 128 point discrete Fourier transform. In the dense case, this is done using only the training data. In the non-dense case, the coefficients is the additional information we need, apart from the training data itself. In the calculation of the operators $\mathcal{T}^{\#}_{n,N}$, we use $n = 128$, $N = 256$. The resulting errors are reported in Figure 3. We note that in the non-dense case, the errors are uniformly small on the entire interval $[-\pi, \pi)$, in contrast to the errors in the right-most figure in Figure 2. Also, the errors near the interpolation nodes are smaller than near the other singularity of the target function at $-\pi/2$. In the non-dense case, we might as well use a larger number of Fourier coefficients to make $n = N$. We did this using 1024 point discrete Fourier transform, and $n = N = 256$. The results are shown in Figure 3 as well. ■

Theorem 3.1 below is a generalization of Theorem 2.1, showing that the operators $\mathcal{T}^{\#}_N = \mathcal{T}^{\#}_{N,N}$ yield the requisite approximation. Our proof is much simpler than that of Theorem 2.1 in either [17] or [27].

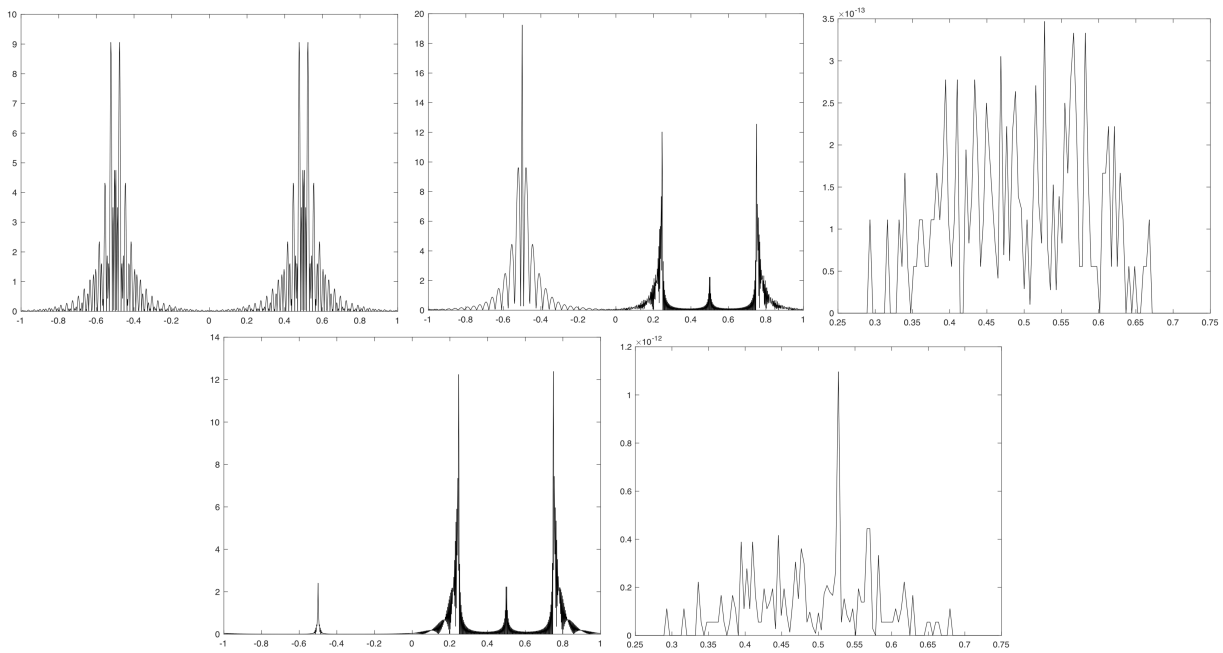

Figure 3: The errors at points of $[-\pi, \pi)$ using $\mathcal{T}^{\#}_{128,256}(\mathcal{D}_1)$ on the upper left, $\mathcal{T}^{\#}_{128,256}(\mathcal{D}_2)$ in the upper middle. The upper right figure is the error for $\mathcal{T}^{\#}_{128,256}(\mathcal{D}_2)$ at the interpolation nodes, to verify that the operator does the interpolation correctly. The bottom left and bottom right figures are analogous to the upper middle and upper right figures with $\mathcal{T}^{\#}_{256,256}(\mathcal{D}_2)$. All errors are maginfied 1000 times, and the $x$ axis has multiples of $\pi$.

**Theorem 3.1** *There exists $B > 0$ with the following property: for $f \in C^*$ and $N \geq B\eta(\mathcal{C})^{-1}$, the trigonometric polynomial $T^{\#}_N(\mathcal{D}) = \mathcal{T}^{\#}_{N,N}(\mathcal{D}) \in \mathbb{H}^q_N$ in (3.4) is well defined, and satisfies*

$$T^{\#}_N(\mathcal{D})(\mathbf{x}_j) = y_j, \qquad j = 1, \cdots, M, \qquad \textit{(Zero training error)} \tag{3.5}$$

*and*

$$\|f - T^{\#}_N(\mathcal{D})\| \leq c\left\{\epsilon + E_{N/2}(f)\right\}. \qquad \textit{(Good generalization error).} \tag{3.6}$$

In view of the observations in Section 2.2, the proof of the above theorem can be modified with neural network approximations at each stage to obtain the following version, where we overload the notation a bit for simplicity.

Suppose the following system of equations has a solution for some $\tilde{N}$ and $N$ (cf. (2.7)):

$$\sum_{j=1}^{M} a_j \mathbb{G}_{\tilde{N}}(\phi, \Phi_N)(\mathbf{x}_\ell - \mathbf{x}_j) = y_\ell - \mathbb{G}_{\tilde{N}}(\phi, \sigma_N(f))(\mathbf{x}_\ell), \qquad \ell = 1, \cdots, M. \tag{3.7}$$

We then define

$$\mathbb{G}^{\#}_{\tilde{N},N}(\mathcal{D})(\mathbf{x}) = \mathbb{G}_{\tilde{N}}(\phi, \sigma_N(f))(\mathbf{x}) + \sum_{j=1}^{M} a_j \mathbb{G}_{\tilde{N}}(\phi, \Phi_N)(\mathbf{x} - \mathbf{x}_j). \tag{3.8}$$

**Theorem 3.2** *There exist $B, \alpha^* > 0$ with the following property. Let $f \in C^*$, $\phi \in C^*(\mathbb{T})$, $\hat{\phi}(1) \neq 0$, $N \geq B\eta(\mathcal{C})^{-1}$. Let $\tilde{N}$ be such that*

$$\|\Phi_N - \mathbb{G}_{\tilde{N}}(\phi, \Phi_N)\| \leq \alpha^*. \tag{3.9}$$

*Then the network $\mathbb{G}^{\#}_N(\mathcal{D}) = \mathbb{G}^{\#}_{\tilde{N},N}(\mathcal{D})$ in (3.8) is well defined and satisfies*

$$\mathbb{G}^{\#}_N(\mathcal{D})(\mathbf{x}_j) = y_j, \qquad j = 1, \cdots, M, \qquad \textit{(Zero training error)} \tag{3.10}$$

*and*

$$\|f - \mathbb{G}^{\#}_N(\mathcal{D})\| \leq c\left\{\epsilon + E_{N/2}(f) + N^{q/2} E_{\tilde{N}}(1; \phi)\|f\|\right\}. \qquad \textit{(Good generalization error).} \tag{3.11}$$

**Remark 3.1** A volume comparison argument shows that the number $M$ of data points satisfies $M \leq c\eta(\mathcal{C})^{-q}$. Thus, Theorems 3.1 and 3.2 show that for a right configuration of the training data, a good generalization error as well as zero training error can be achieved by choosing the number of parameters proportional to the number of data points. It is demonstrated in [2] that for RBF approximation, this phenomenon seems to hold in many applications with the number of parameters exactly equal to the number of data points.

**Remark 3.2** In practice, the training data is high dimensional and sparse; i.e., $\eta(\mathcal{C})$ is large. The requirement that $N \geq B\eta(\mathcal{C})^{-1}$ is therefore satisfied with moderate degrees $N$.

As demonstrated before, one cannot construct $T_N^{\#}(\mathcal{D})$ based only on the training data $\mathcal{C}$, unless $\mathcal{C}$ is sufficiently dense on $\mathbb{T}^q$. We have already discussed an effort in the form of the operators $\mathcal{L}_N(\mathcal{D})$ in Example 3.2. An even simpler approach of just considering

$$\frac{1}{\Phi_N(0)} \sum_{k=1}^{M} y_k \Phi_N(\circ - \mathbf{x}_k)$$

yields similar bounds on the generalization error as those obtained by $\mathcal{L}_N(\mathcal{D})$. Of course, the training error for this simple construction is not 0, but the localization properties of $\Phi_N$ ensure that it is small. In general, if we anticipate a scenario where the training data sets become increasingly dense on some compact subset $K \subset \mathbb{T}^q$, then we cannot expect convergence of trigonometric polynomials that interpolate a noisy data, where the noise level does not decrease as well.

Another approach, described in [3], is to minimize a high order Sobolev norm of the trigonometric polynomial subject to the interpolatory conditions. This approach has been used to great advantage for a numerical solution of some notoriously hard partial differential equations in 2 or 3 dimensions. However, the calculations are very ill-conditioned and require very carefully designed algorithms.

We describe a softer regularization scheme that does not require high order Sobolev norms, and yields both good training and generalization errors. The generalization error is given point-wise, and is commensurate with the estimates given in Theorem 3.1 when there is no noise.

The space $W^* = W^*(\mathbb{T}^q)$ consists of all continuously differentiable functions $f \in C^*$. We define

$$\|f\|_{W^*} = \|f\|_{W^*(\mathbb{T}^q)} = \sum_{j=1}^{q} \|D_j f\|. \tag{3.12}$$

For $n > 0$ and $T \in \mathbb{H}_n^q$, let

$$R_n(T) = \max_{1 \leq j \leq M} |y_j - T(\mathbf{x}_j)| + \frac{1}{n}\|T\|_{W^*}. \tag{3.13}$$

**Theorem 3.3** *Let $f \in W^*$, $B$ be as in Theorem 3.1, $N \geq B\eta(\mathcal{C})^{-1}$, and $T^*(\mathcal{D}) = \arg\min_{T \in \mathbb{H}_N^q} R_N(T)$. Then*

$$\max_{1 \leq j \leq M} |y_j - \mathcal{T}^*(\mathcal{D})(\mathbf{x}_j)| \leq \min_{T \in \mathbb{H}_N^q} R_N(T) \leq c\left\{\epsilon + \frac{1}{N}\|f\|_{W^*}\right\}. \qquad \textit{(Good training error)} \tag{3.14}$$

*Let $\mathbf{x} \in \mathbb{T}^q$, and $\delta = \min_{1 \leq j \leq M} |\mathbf{x} - \mathbf{x}_j|$. Then*

$$|f(\mathbf{x}) - T^*(\mathcal{D})(\mathbf{x})| \leq c(1 + N\delta)\left\{\epsilon + \frac{1}{N}\|f\|_{W^*}\right\}. \qquad \textit{(Good generalization error).} \tag{3.15}$$

**Remark 3.3** Theorem 3.3 makes no assumption on the target function except for differentiability. This is in contrast to usual machine learning theory, where one has to assume that the target function belongs to a reproducing kernel Hilbert space, with the kernel prescribed by the learning algorithm.

**Remark 3.4** The estimate (3.15) shows that if $\mathbf{x}$ is very close to the training data so that $N\delta < 1$, then the generalization error at $\mathbf{x}$ satisfies the same upper bound (3.14) that holds for $R_N(T^*)$. As remarked earlier in Remark 3.2, we have greater liberty in choosing $N$ when the data is sparse. To take advantage of this fact, let $\mathbf{x}$ be such that $N\delta \geq 1$. Then (3.15) can be reformulated in the form

$$|f(\mathbf{x}) - T^*(\mathcal{D})(\mathbf{x})| \leq c\delta\left\{N\epsilon + \|f\|_{W^*}\right\}. \tag{3.16}$$

The term $\delta\|f\|_{W^*}$ is clearly a customary bound from numerical analysis in view of the mean value theorem. If $\epsilon \leq \eta(\mathcal{C})\|f\|_{W^*}$, it is possible to choose $N \sim \eta(\mathcal{C})^{-1}$ so that error bound is $c\delta\|f\|_{W^*}$.

**Remark 3.5** It is well known (cf. [30, Chapter X, Theorem 7.28] for the univariate case) that for any $T \in \mathbb{H}_N^q$,

$$\|T\| \sim \max_{|\mathfrak{m}|_\infty \le 3N-1} \left| T\left(\frac{2\pi\mathfrak{m}}{3N}\right)\right|. \tag{3.17}$$

Therefore,

$$R_N(T) \sim \max_{1\le j\le M} |y_\ell - \sum_{\mathbf{k}} \hat{T}(\mathbf{k}) \exp(i\mathbf{k}\cdot\mathbf{x}_j)| + \frac{1}{N}\sum_{j=1}^{q} \max_{|\mathfrak{m}|_\infty \le 3N-1} \left|\sum_{\mathbf{k}} k_j \hat{T}(\mathbf{k}) \exp\left(2\pi i \frac{\mathbf{k}\cdot\mathfrak{m}}{3N}\right)\right|.$$

If we replace $R_N(T)$ by the expression on the right hand side of the above equation, we get an optimization problem directly in terms of the coefficients $\hat{T}(\mathbf{k})$. The theoretical results are not affected except for the actual values of the constants involved.

**Remark 3.6** In this remark, let $T^* = \mathcal{T}^*(\mathcal{D})$. In view of (2.8), the estimates (3.14) and (3.15) imply for any activation function $\phi$,

$$\max_{1\le j\le M} |y_j - \mathbb{G}_{\tilde{N}}(\phi, T^*)(\mathbf{x}_j)| \le c\left\{\epsilon + \frac{1}{N}\|f\|_{W^*}\right\} + \frac{4E_{\tilde{N}}(1;\phi)}{\hat{\phi}(1)} \sum_{\mathbf{k}} |\widehat{T^*}(\mathbf{k})|,$$

and

$$|f(\mathbf{x}) - \mathbb{G}_{\tilde{N}}(\phi, T^*)(\mathbf{x})| \le c(1+N\delta)\left\{\epsilon + \frac{1}{N}\|f\|_{W^*}\right\} + \frac{4E_{\tilde{N}}(1;\phi)}{\hat{\phi}(1)} \sum_{\mathbf{k}} |\widehat{T^*}(\mathbf{k})|$$

respectively. In particular, for the activation function obtained from the smooth ReLU function, the extra error terms decrease exponentially rapidly with $\tilde{N}$. It is therefore tempting to set up the regularization functional (3.13) directly with neural networks with free coefficients, weights, and thresholds to be determined by a suitable optimization technique. However, the estimate (2.8) implies a strong connection between these parameters for the network approximating a trigonometric polynomial. Therefore, the solution to such a direct approach with neural networks is not guaranteed to give the right training and testing errors.

# 4 Deep networks

The following discussion regarding the terminology for deep networks, including Figure 4, is based on the discussion in [20], and elaborates upon the same.

A commonly used definition of a deep network is the following. Let $\phi : \mathbb{R} \to \mathbb{R}$ be an activation function; applied to a vector $\mathbf{x} = (x_1, \cdots, x_q)$, $\phi(\mathbf{x}) = (\phi(x_1), \cdots, \phi(x_q))$. Let $L \ge 1$ be an integer, for $\ell = 0, \cdots, L$, let $q_\ell \ge 1$ be an integer ($q_0 = q$), $T_\ell : \mathbb{R}^{q_\ell} \to \mathbb{R}^{q_{\ell+1}}$ be an affine transform, where $q_{L+1} = 1$. A deep network with $L$ hidden layers is defined as the compositional function

$$\mathbf{x} \mapsto T_L(\phi(T_{L-1}(\phi(T_{L-2}\cdots\phi(T_0(\mathbf{x}))\cdots). \tag{4.1}$$

This definition has several shortcomings. First, a function may have more than one compositional representation, as we will demonstrate shortly, so that the affine transforms and $L$ are not determined uniquely by the function itself. Second, this notion does not capture the connection between the nature of the target function and its approximation. Third, the affine transforms $T_\ell$ define a special directed acyclic graph (DAG). It is cumbersome to describe notions of weight sharing, convolutions, sparsity, skipping of layers, etc. in terms of these transforms. Therefore, we have proposed in [20], to describe a deep network more generally as a directed acyclic graph (DAG) architecture.

Let $\mathcal{G}$ be a DAG, with the set of nodes $V \cup \mathbf{S}$, where $\mathbf{S}$ is the set of source nodes, and $V$ that of non-source nodes. A $\mathcal{G}$-function is defined as follows. Each of the in-edges to each node in $V \cup \mathbf{S}$ represents an input real variable. For each node $v \in V \cup \mathbf{S}$, we denote its in-degree by $d(v)$. A node $v \in V \cup \mathbf{S}$ itself represents the evaluation of a real valued function $f_v$ of the $d(v)$ inputs. The out-edges fan out the result of this evaluation. Each of the source nodes obtains an input from some Euclidean space. Other nodes can also obtain such an input, but by introducing dummy nodes, it is convenient to assume that only the source nodes obtain an input from the Euclidean space.

The notion of the level (or height) of a node is defined as follows. The level of a source node is 0; it represents a shallow network. The level of $v \in V$ is the length of the longest path from the nodes in $\mathbf{S}$ to $v$. The level of $v$ will be denoted by $H(v)$.

In [20], we have argued that deep networks display better approximation properties than shallow networks because they can take advantage of a compositional structure in the target function which shallow networks cannot. However, compositionality is the property of an expression for the function; not an intrinsic property of a function itself. A simple example in the univariate case is the constant function $f(x) \equiv 2$, $x \in [0,1]$, that can also be expressed as a compositional function

$$f(x) = (x+1)\cosh\left(\log\left(\frac{2+\sqrt{3-2x-x^2}}{x+1}\right)\right), \qquad x \in [0,1].$$

It is not clear whether two different DAG structures can give rise to the same function. Even if we assume a certain DAG, it is not clear that the choice of the constituent functions is uniquely determined for a given function on $\mathbb{R}^q$. For example, one can write

$$\cos^4 x = ((\cos x)^2)^2 = (\cos^2 x)^2 = (1/4)(1+\cos(2x))^2.$$

The second expression above has the structure $h_1(h_2(h_3(x)))$, and the other two have the structures $g_1(g_2(x))$ or $f_1(f_2(x))$, both representing the same DAG but with different constituent functions. Thus, the question of whether a given multivariate function is in fact compositional cannot be answered. Of course, for a given DAG, it is possible to use inverse/implicit function theorem (in theory) in some cases to decide whether a family of functions are compositional according to the given DAG.

For our mathematical analysis, we therefore find it convenient to think of a $\mathcal{G}$-function as a set of functions $f = \{f_v : \mathbb{R}^{d(v)} \to \mathbb{R}\}_{v \in V \cup \mathbf{S}}$, rather than a single function on $\mathbb{R}^q$. For example, the DAG $\mathcal{G}$ in Figure 4 ([20]) represents the compositional function

$$\begin{aligned} f^*(x_1,\cdots,x_9) &= h_{19}(h_{17}(h_{13}(h_{10}(x_1,x_2,x_3),h_{11}(x_4,x_5)),\\ &\qquad h_{14}(h_{10},h_{11}),h_{16}(h_12(x6,x_7,x_8,x_9)),h_{18}(h_{15}(h_{11},h_{12}),h_{16})). \end{aligned} \tag{4.2}$$

The $\mathcal{G}$-function is $\{h_{10},\cdots,h_{19}=f^*\}$. The individual functions $f_v$ will be called *constituent functions*.

We assume that there is only one sink node, $v^*$ (or $v^*(\mathcal{G})$) whose output is denoted by $f^*$ (the *target function*). Technically, there are two functions involved here: one is the final output as a function of all the inputs to all source nodes, the other is the final output as a function of the inputs to the node $v^*$. We will use the symbol $f^*$ to denote both with comments on which meaning is intended when we feel that it may not be clear from the context. A similar convention is followed with respect to each of the constituent functions as well. For example, in the DAG of Figure 4, the function $h_{14}$ can be thought of both as a function of two variables, namely the outputs of $h_{10}$ and $h_{11}$ as well as a function of five variables $x_1,\cdots,x_5$. In particular, if each constituent function is a neural network, $h_{14}$ is a shallow network receiving two inputs.

In this paper, we are interested only in the case where each of the inputs to each of the source nodes is in $\mathbb{T}$ rather than $\mathbb{R}$. Although this is no longer true for the non-source nodes, it is possible to accomplish this in the case when each of the constituent functions is continuous, as follows. We observe first that there is a one-to-one correspondence between functions on $[-1,1]^d$ and functions on $\mathbb{T}^d$ that are even in each variable, given by

$$F^\circ(\theta_1,\cdots,\theta_d) = F(\cos\theta_1,\cdots,\cos\theta_d), \qquad (\theta_1,\cdots,\theta_d) \in \mathbb{T}^d.$$

Let $f_u$ be one of the constituent functions. With a re-normalization, we may assume that the range of $f_u$ is a subset of $[-1,1]$. If $u_1,\cdots,u_{d(v)}$ are children of $v$ (i.e., if there is an edge from each $u_1,\cdots,u_{d(v)}$ to $v$), then $f_v$ can be seen as a function on $[-1,1]^d$, from which one can construct an even function $f_v^\circ$ on $\mathbb{T}^d$ as just explained : informally, think of $f_v$ as an even function of the points $(\pm\arccos(f_{u_1}),\cdots,\pm\arccos(f_{u_{d(v)}})) \in \mathbb{T}^d$. Rather than complicating our notations, we will therefore assume in this paper that the domain of each constituent function is a torus, and the range is a subset of $\mathbb{T}$. In particular, we may assume that every constituent function $f_v \in C^*(\mathbb{T}^{d(v)})$.

We adopt the convention that for any function class $\mathbb{X}(\mathbb{T}^d)$, the class $\mathcal{G}$-$\mathbb{X}$ denotes the set of $\mathcal{G}$-functions $f = \{f_v\}_{v\in V}$, where each constituent function $f_v \in \mathbb{X}(\mathbb{T}^{d(v)})$. We define

$$\|f\|_{\mathbb{X},\mathcal{G}} = \sum_{v\in V\cup\mathbf{S}} \|f_v\|_{\mathbb{X}(\mathbb{T}^{d(v)})}. \tag{4.3}$$

Thus, for example, $\mathcal{G}$-$W^*$ is the set of all $\mathcal{G}$-functions $f = \{f_v\}_{v\in V}$, where each constituent function $f_v \in \mathbb{X}(\mathbb{T}^{d(v)})$, and we write

$$\|f\|_{W^*,\mathcal{G}} = \sum_{v\in V\cup\mathbf{S}} \|f_v\|_{W^*(\mathbb{T}^{d(v)})}.$$

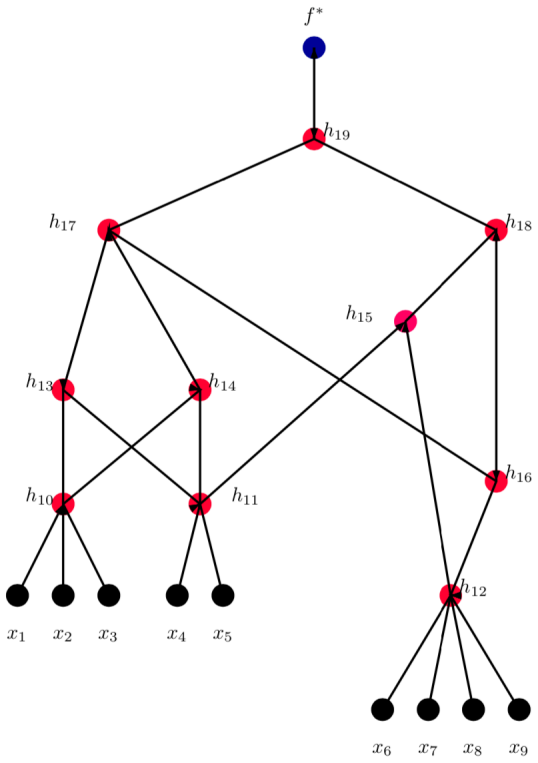


Figure 4: An example of a $\mathcal{G}$-function ($f^*$ given in (4.2)). The vertices of the DAG $\mathcal{G}$ are denoted by red dots. The black dots represent the input to the various nodes as indicated by the in–edges of the red nodes, and the blue dot indicates the output value of the $\mathcal{G}$-function, $f^*$ in this example.

For a vector $\mathbf{N} = (N_v)$, the symbol $\mathcal{G}$-$\mathbb{H}_{\mathbf{N}}$ denotes the set of $\mathcal{G}$-functions $\{T_v\}_{v\in V}$, where each $T_v \in \mathbb{H}_{N_v}^{d(v)}$, and for a $\mathcal{G}$-function $f = \{f_v\}_{v\in V}$,

$$E_{\mathbf{N},\mathcal{G}}(f) = \sum_{v\in V\cup\mathbf{S}} E_{N_v}(d(v); f_v). \tag{4.4}$$

To make precise the various inputs to the constituent functions, we introduce some conventions. Let $\mathbf{S} = \{v_1, \cdots, v_s\}$, $q = \sum_{j=1}^{s} d(v_j)$. The input $\mathbf{x}$ can be viewed as a vector in $\mathbb{R}^q$, but also as an element of $\prod_{j=1}^{s} \mathbb{R}^{d(v_j)}$; i.e., $\mathbf{x} = ((\mathbf{x})_{v_1}, \cdots, (\mathbf{x})_{v_s})$ so that $(\mathbf{x})_{v_j} \in \mathbb{R}^{d(v_j)}$. We note that in making this statement, we have tacitly introduced some possible dummy variables here: for example, the function with the compositional structure

$$F(x_1, x_2, x_3) = f(f_1(x_1, x_2), f_2(x_2, x_3))$$

is viewed as a function of 4 variables

$$\tilde{F}(x_1, \cdots, x_4) = f(f_1(x_1, x_2), f_2(x_4, x_3))$$

although $F$ is the restriction of $\tilde{F}$ to the hyper-plane $x_2 = x_4$. In our opinion, it is a very deep question to figure out what domains the functions in practice are defined on, and to some extent, manifold learning addresses this issue. Much of the approximation theory literature is however limited to approximation on cubes, spheres, and other known domains, torus in the present paper. From a purely mathematical point of view, this is justified by an appeal to Stein's extension theorem [26, Chapter VI]. Therefore, making this tacit introduction of dummy variables allows us to simplify notation and results without any theoretical loss of generality.

Correspondingly, we define the sets

$$\mathcal{C}_{v_j} = \{(\mathbf{x})_{v_j} : \mathbf{x} \in \mathcal{C}\}, \qquad j = 1, \cdots, s.$$

Thus, $\mathcal{C}_{v_j}$ is the training data "seen" by the source node $v_j$. This notion is extended recursively to other nodes of $\mathcal{G}$. Let $v$ not be a source node, $u_1, \cdots, u_{d(v)}$ be the children of $v$, $\mathcal{C}_{u_1}, \cdots, \mathcal{C}_{u_{d(v)}}$ be the training data seen by these nodes. Thus, for any $\mathbf{x} \in \mathcal{C}$, the components given by $(\mathbf{x})_{u_j}$ are seen by $u_j$, and

$$\mathcal{C}_v = \left\{(f_{u_1}((\mathbf{x})_{u_1}), \cdots, f_{u_{d(v)}}((\mathbf{x})_{u_{d(v)}})) : \mathbf{x} \in \mathcal{C}\right\}.$$

For example, in the DAG of Figure 4, the children of $h_{14}$ are $h_{10}$ and $h_{11}$. For each $\mathbf{x} \in \mathbb{R}^9$, $h_{10}$ sees the components $(x_1, x_2, x_3)$, while $h_{11}$ sees the components $(x_4, x_5)$. We have

$$\begin{aligned} \mathcal{C}_{h_{10}} = \{((\mathbf{x})_1, (\mathbf{x})_2, (\mathbf{x})_3) : \mathbf{x} \in \mathcal{C}\}, \quad \mathcal{C}_{h_{11}} = \{((\mathbf{x})_4, (\mathbf{x})_5) : \mathbf{x} \in \mathcal{C}\}, \\ \mathcal{C}_{h_{14}} = \{((h_{10}((\mathbf{x})_1, (\mathbf{x})_2, (\mathbf{x})_3), h_{11}((\mathbf{x})_4, (\mathbf{x})_5)) : \mathbf{x} \in \mathcal{C}\} \end{aligned}$$

In this section, we state our theorems only for networks with activation function $t \mapsto \cos t$; i.e., networks that evaluate trigonometric polynomials. The transition to networks with other activation functions is obtained by approximating these by trigonometric polynomials using (2.8) as in Section 3. This only adds to an additional complication in the notation without adding anything new conceptually.

The analogue of Theorem 3.1 is the following.

**Theorem 4.1** *Let $\mathcal{G}$ be a DAG with sink node $v^*$. There exists $C = C(\mathcal{G}) > 0$ with the following property: Let $f = \{f_v\}_{v \in V \cup \mathbf{S}}$ be a $\mathcal{G}$-function such that each of the constituent functions $\{f_v\}_{v \in V}$ is Lipschitz continuous with Lipschitz constant $\le L$. If $N_v \ge C(\mathcal{G})\eta(\mathcal{C}_v)^{-1}$, for each $v \in V$, and $\mathbf{N} = (N_v)$, the $\mathcal{G}$-function $T^{\#}_{\mathbf{N}}(\mathcal{D}) \in \mathcal{G}\text{-}\mathbb{H}_{\mathbf{N}}$ defined in (5.36) satisfies*

$$(T^{\#}_{\mathbf{N}}(\mathcal{D}))_{v^*}(\mathbf{x}_j) = y_j, \qquad j = 1, \cdots, M, \qquad \textit{(Zero training error)} \tag{4.5}$$

*and*

$$\|f - T^{\#}_{\mathbf{N}}(\mathcal{D})\|_{C^*,\mathcal{G}} \le c(\mathcal{G}, L)\left\{\epsilon + E_{\mathbf{N}/2,\mathcal{G}}(f)\right\}. \qquad \textit{(Good generalization error).} \tag{4.6}$$

We recall that the polynomials in Theorem 3.1 and hence, in Theorem 4.1 are constructed using the Fourier coefficients of the various functions involved. One could use the polynomials $\mathcal{L}_N(\mathcal{D})$ defined in (3.2) instead to get a construction based only on the values of the various functions computed using the training data. However, since we do not know the constituent functions, this construction is not as constructive as with shallow networks. Besides, as with shallow networks, such a construction does not yield a uniform error bound analogous to (4.6).

In contrast, Theorem 3.3 can be extended in a purely constructive manner, so as to yield a good training error and to keep the gradient of the resulting approximation under control so that pointwise generalization error bounds can be obtained. Even though one does not know the constituent functions, one can construct a DAG trigonometric polynomial knowing the DAG structure and the training data alone.

For $\mathbf{n} = (n_v)$ and $T \in \mathcal{G}\text{-}\mathbb{H}_{\mathbf{n}}$, let

$$R_{\mathbf{n},\mathcal{G}}(T) = \max_{1 \le j \le M} |y_j - T_{v^*}(\mathbf{x}_j)| + \sum_{v \in V} \frac{1}{n_v} \|T_v\|_{W^*(\mathbb{T}^{d(v)})}. \tag{4.7}$$

In this definition, it is understood that $T_{v^*}$ is thought of as a function of the $q$-dimensional vector $\mathbf{x}$, but is computed using the all the constituent functions in $T$ using DAG structure prescribed by $\mathcal{G}$.

**Theorem 4.2** *Let $\mathcal{G}$ be a DAG with sink node $v^*$. Let $f = \{f_v\}_{v \in V \cup \mathbf{S}} \in \mathcal{G}\text{-}W^*$, and $\max_{v \in V \cup \mathbf{S}} \|f_v\|_{W^*(\mathbb{T}^{d(v)})} \le L$. There exists $C(\mathcal{G}) > 0$ with the following property: If $N_v \ge C(\mathcal{G})\eta(\mathcal{C}_v)^{-1}$, for each $v \in V$, and $\mathbf{N} = (N_v)$,*

$$\min_{T \in \mathcal{G}\text{-}\mathbb{H}_{\mathbf{N}}} R_{\mathbf{N},\mathcal{G}}(T) \le c(\mathcal{G}, L)\left\{\epsilon + \sum_{v \in V} \frac{1}{N_v} \|f_v\|_{W^*(\mathbb{T}^{d(v)})}\right\}. \qquad \textit{(Good training error)} \tag{4.8}$$

*Let $\mathbf{x} \in \mathbb{T}^q$, $\delta = \min_{1 \le j \le M} |\mathbf{x} - \mathbf{x}_j|$, $N = \max_{v \in V} N_v$. Then with*

$$T^*(\mathcal{D}) = \underset{\mathcal{G}\text{-}\mathbb{H}_{\mathbf{N}}}{\textit{arg min}}\, R_{\mathbf{N},\mathcal{G}}(T),$$

*we have*

$$|f_{v^*}(\mathbf{x}) - (T^*(\mathcal{D}))_{v^*}(\mathbf{x})| \le c(\mathcal{G}, L)\left\{\epsilon + \sum_{v \in V} \frac{1}{N_v} \|f_v\|_{W^*(\mathbb{T}^{d(v)})}\right\} + c(\mathcal{G}, L)\,(1 + N\epsilon)^{H(v^*)}\,\delta. \qquad \textit{(Good generalization error).} \tag{4.9}$$

**Remark 4.1** The theorems in this section indicate that the superiority of deep learning over shallow learning comes from two factors. One is that the compositional structure of the target function allows us to study the problem in a cascade of low dimensional problems. The other is that this structure might allow us to sparsify the training data as we move up the cascade; noting that the minimal separation is defined only among the distinct points in a set.

# 5 Proofs

## 5.1 Preliminary results on trigonometric approximation

We recall first some essential properties of the kernels and operators defined in (2.2) and (2.3). Since the proofs of these facts are scattered among several of our publications, we will sketch the proof in some detail for the sake of completion.

**Proposition 5.1** *Let $S > q$ be an integer. For $N \geq 1$,*

$$|\Phi_N(\mathbf{x})| \leq \frac{cN^q}{\max(1, (N|\mathbf{x}|)^S)}, \qquad \mathbf{x} \in \mathbb{T}^q, \tag{5.1}$$

*and*

$$|\Phi_N(\mathbf{0})| \geq cN^q. \tag{5.2}$$

PROOF. Our proof summarizes that of [3, Theorem 6.1]. We consider the function $H(\mathbf{t}) = h(|\mathbf{t}|_2)$, $\mathbf{t} \in \mathbb{R}^q$. Since $0 \leq H(\mathbf{t}) \leq 1$ for all $\mathbf{t} \in \mathbb{R}^q$, and $H(\mathbf{t}) = 1$ for $|\mathbf{t}|_2 \leq 1/2$, it is clear that

$$|\Phi_N(\mathbf{x})| \leq \Phi_N(\mathbf{0}) \sim N^q. \tag{5.3}$$

In particular, this proves (5.2). Since $h$ is constant in a neighborhood of $\mathbf{0}$, it is easy to see that $H$ is $S$ times continuously differentiable as well, so that its Fourier transform satisfies

$$|\hat{H}(\mathbf{u})| \leq \frac{c(H)}{|\mathbf{u}|^S}, \qquad \mathbf{u} \neq \mathbf{0}. \tag{5.4}$$

Hence, the Poisson summation formula yields

$$\Phi_N(\mathbf{x}) = N^q \sum_{\mathbf{j} \in \mathbb{Z}^q} \hat{H}(N(\mathbf{x} + 2\mathbf{j}\pi)), \qquad \mathbf{x} \in \mathbb{T}^q. \tag{5.5}$$

For $\mathbf{x} \neq 0$, $\mathbf{j} \neq \mathbf{0}$,

$$|\mathbf{x} + 2\mathbf{j}\pi| \geq 2|\mathbf{j}|_\infty \pi - |\mathbf{x}| \geq (2|\mathbf{j}|_\infty - 1)|\mathbf{x}|.$$

Therefore, (5.4) and (5.5) show that for $\mathbf{x} \neq \mathbf{0}$,

$$|\Phi_N(\mathbf{x})| \leq N^q \left\{ |\hat{H}(N\mathbf{x})| + \sum_{\mathbf{j} \in \mathbb{Z}^q, \mathbf{j} \neq \mathbf{0}} |\hat{H}(N(\mathbf{x} + 2\mathbf{j}\pi))| \right\} \leq c(H) N^q \left\{ \frac{1}{(N|\mathbf{x}|)^S} + \sum_{\mathbf{j} \in \mathbb{Z}^q, \mathbf{j} \neq \mathbf{0}} \frac{1}{((2|\mathbf{j}|_\infty - 1)N|\mathbf{x}|)^S} \right\}.$$

Since $S > q$, the infinite series converges. Therefore,

$$|\Phi_N(\mathbf{x})| \leq c \frac{N^q}{(N|\mathbf{x}|)^S}, \qquad \mathbf{x} \neq \mathbf{0}. \tag{5.6}$$

Together with (5.3), this leads to (5.1). ■

**Proposition 5.2** *Let $S > q$ be an integer.*
*(a) There exists a constant $B > 0$ with the following property: If $\mathcal{C} = \{\mathbf{x}_1, \cdots, \mathbf{x}_M\}$, and $N \geq B\eta(\mathcal{C})^{-1}$ then for $\mathbf{x} \in \mathbb{T}^q$,*

$$\sum_{j=1}^{M} |\Phi_N(\mathbf{x} - \mathbf{x}_j)| \leq cN^q. \tag{5.7}$$

*and*

$$\sum_{j:|\mathbf{x}_j-\mathbf{x}|\ge\eta(\mathcal{C})} |\Phi_N(\mathbf{x}-\mathbf{x}_j)| \le (1/2)\Phi_N(\mathbf{0}) = (1/2)\Phi_N(\mathbf{x}-\mathbf{x}) \le cN^q. \tag{5.8}$$

*(b) If $T \in \mathbb{H}^q_{N/2}$ then $\sigma_N(T) = T$. Further,*

$$\|\sigma_N(f)\| \le c\|f\|, \qquad f \in C^*. \tag{5.9}$$

*Consequently,*

$$E_N(f) \le \|f - \sigma_N(f)\| \le cE_{N/2}(f), \qquad f \in C^*. \tag{5.10}$$

The two parts of the proposition are proved along the same lines. It is convenient to prove the following lemma (cf. [14, Proposition 5.1]), which we will use once with $\nu$ being the measure that associates the mass 1 with each $\mathbf{x}_j$ for part (a), and once with $\nu$ being the Lebesgue measure $\mu^*$ on $\mathbb{T}^q$ for part (b).

**Lemma 5.1** *Let $\nu$ be a positive measure on $\mathbb{T}^q$, $d \ge 0$, and for some $A_\nu > 0$,*

$$\nu(\mathbb{B}(\mathbf{x}, r)) \le A_\nu(r+d)^q, \qquad r > 0,\ \mathbf{x} \in \mathbb{T}^q. \tag{5.11}$$

*Then for $r \ge 1/N$,*

$$\int_{\mathbb{T}^q\setminus\mathbb{B}(\mathbf{x},r)} |\Phi_N(\mathbf{x}-\mathbf{u})|d\nu(\mathbf{u}) \le cA_\nu(Nr)^{q-S}(1+d/r)^q, \tag{5.12}$$

*where $c$ is a positive constant depending only on $q$, $S$, and $h$ but not on $r$, $d$, or $\nu$.*

Proof. In this proof, we assume by re-normalization that $A_\nu = 1$. Also, we write $\mathbb{A}_k = \mathbb{B}(\mathbf{x}, 2^{k+1}r) \setminus \mathbb{B}(\mathbf{x}, 2^k r)$. Then (5.11) shows that $\nu(\mathbb{A}_k) \le (2^{k+1}r + d)^q \le 2^q 2^{kq}(r+d)^q$. Using (5.6), we conclude that

$$\int_{\mathbb{T}^q\setminus\mathbb{B}(\mathbf{x},r)} |\Phi_N(\mathbf{x}-\mathbf{u})|d\nu(\mathbf{u}) \le \sum_{k=0}^{\infty}\int_{\mathbb{A}_k} |\Phi_N(\mathbf{x}-\mathbf{u})|d\nu(\mathbf{u}) \le c2^q(Nr)^{q-S}(1+d/r)^q \sum_{k=0}^{\infty} 2^{(q-S)k}.$$

■

**Proof of Proposition 5.2.**

To prove part (a), let $\nu$ be a measure that associates the mass 1 with each $\mathbf{x}_j$, $j = 1, \cdots, M$, and let $\eta = \eta(\mathcal{C})$. We claim that $\nu$ satisfies (5.11) with $d = \eta$ and $A_\nu = c\eta^{-q}$ for some $c > 0$ depending only on $q$. Fix $\mathbf{x}$ and $r$. If $r < \eta/2$, then $\mathbb{B}(\mathbf{x}, r) \cap \mathcal{C}$ can contain at most 1 point. Therefore the claim is satisfied trivially. Let $r \ge \eta/2$, and $\mathbb{B}(\mathbf{x}, r) \cap \mathcal{C} = \{\mathbf{x}_1, \cdots, \mathbf{x}_J\}$. We observe that $\mu^*(\mathbb{B}(\mathbf{y}, s)) = cs^q$ for all $\mathbf{y} \in \mathbb{T}^q$ and $s \in (0, \pi)$. Since the balls $\mathbb{B}(\mathbf{x}_j, \eta/3)$ are disjoint, we have

$$\nu(\mathbb{B}(\mathbf{x}, r)) = |J| = c\eta^{-q}\sum_{j=1}^{J} \mu^*(\mathbb{B}(\mathbf{x}_j, \eta/3)) = c\eta^{-q}\mu^*\left(\cup_{j=1}^{J}\mathbb{B}(\mathbf{x}_j, \eta/3)\right) \le c\eta^{-q}\mu^*(\mathbb{B}(\mathbf{x}, r+\eta/3)) \le c_1\eta^{-q}(r+\eta)^q.$$

This proves the claim. Thus, we may use (5.12) with $r = d = \eta$, $A_\nu = c\eta^{-q}$ to obtain for $N \ge \eta^{-1}$ that

$$\sum_{j:|\mathbf{x}-\mathbf{x}_j|\ge\eta} |\Phi_N(\mathbf{x}-\mathbf{x}_j)| = \int_{\mathbb{T}^q\setminus\mathbb{B}(\mathbf{x},\eta)} |\Phi_N(\mathbf{x}-\mathbf{u})|d\nu(\mathbf{u}) \le c\eta^{-q}(N\eta)^{q-S} = cN^q(N\eta)^{-S} \le c\Phi_N(\mathbf{0})(N\eta)^{-S}.$$

We choose $B > 0$ such that if $N\eta \ge B$, $c(N\eta)^{-S} \le 1/2$. This proves (5.8). Further, since any ball $\mathbb{B}(\mathbf{x}, \eta)$ contains at most $2^q$ points of $\mathcal{C}$, (5.8) and (5.3) lead to (5.7).

To prove part (b), we use Lemma 5.1 with $\mu^*$ in place of $\nu$. Clearly, (5.11) is satisfied with $d = 0$, and $A_{\mu^*} = c$. Therefore, using (5.3) and (5.12) (with $r = 1/N$) leads to

$$\int_{\mathbb{T}^q} |\Phi_N(\mathbf{x}-\mathbf{u})|d\mu^*(\mathbf{u}) = \int_{\mathbb{B}(\mathbf{x},1/N)} |\Phi_N(\mathbf{x}-\mathbf{u})|d\mu^*(\mathbf{u}) + \int_{\mathbb{T}^q\setminus\mathbb{B}(\mathbf{x},1/N)} |\Phi_N(\mathbf{x}-\mathbf{u})|d\mu^*(\mathbf{u}) \le c.$$

It is now easy to deduce (5.9). If $T \in \mathbb{H}^q_{N/2}$, then $\hat{T}(\mathbf{k}) = 0$ if $|\mathbf{k}|_2 \ge N/2$, while $h(t) = 1$ if $|t| \le 1/2$. It follows from the definition (2.3) that $\sigma_N(T) = T$. For any $f \in C^*$ and $T \in \mathbb{H}^q_{N/2}$, (5.9) leads to

$$E_N(f) \le \|f - \sigma_N(f)\| = \|f - T - \sigma_N(f-T)\| \le c\|f-T\|.$$

This implies (5.10). ■

**Proposition 5.3** *Let $\mathcal{C}$, $B$, and $N$ be as in Proposition 5.2, $\Psi \in C^*$, and $\mathbf{B}$ be the matrix $[\Psi(\mathbf{x}_j - \mathbf{x}_k)]_{j,k=1}^M$. There exists $\alpha^* > 0$ such that if*

$$\|\Phi_N - \Psi\| \le \alpha^*, \tag{5.13}$$

*then the matrix $\mathbf{B}$ is invertible, and $\|\mathbf{B}^{-1}\|_{\ell^\infty \to \ell^\infty} \le cN^{-q}$. In particular, let $\mathbf{b} \in \mathbb{R}^M$. Then there exists (uniquely) $\mathbf{a} \in \mathbb{R}^M$ such that*

$$\sum_{j=1}^M a_j \Psi(\mathbf{x}_\ell - \mathbf{x}_j) = b_\ell, \qquad \ell = 1, \cdots, M, \tag{5.14}$$

*and*

$$\max_{1\le j\le M} |a_j| \le cN^{-q} \max_{1\le \ell\le M} |b_\ell|. \tag{5.15}$$

Proof. When $\Psi = \Phi_N$, the proposition follows from (5.2), (5.8), and standard facts from linear algebra, (cf. [14, Proposition 6.1]). In this proof, the matrix norm $\|\cdot\|$ will refer to the norm $\|\cdot\|_{\ell^\infty \to \ell^\infty}$, and we write $\eta = \eta(\mathcal{C})$. Denoting the matrix $[\Phi_N(\mathbf{x}_j - \mathbf{x}_k)]_{j,k=1}^M$ by $\mathbf{A}$, we have observed that $\|\mathbf{A}^{-1}\| \le cN^{-q}$. Also, it is easy to see that $M \le c\eta^{-q}$. Therefore, recalling that $N\eta \ge B$,

$$\begin{aligned}\|\mathbf{A} - \mathbf{B}\| &= \max_j \sum_{k=1}^M |\Phi_N(\mathbf{x}_j - \mathbf{x}_k) - \Psi(\mathbf{x}_j - \mathbf{x}_k)| \le M\|\Phi_N - \Psi\| \le c\eta^{-q}\|\Phi_N - \Psi\| \\ &\le cB^{-q}N^q\|\Phi_N - \Psi\| \le \frac{cB^{-q}}{\|\mathbf{A}^{-1}\|}\|\Phi_N - \Psi\|.\end{aligned}$$

We now choose $\alpha^*$ so that (5.13) implies

$$\|\mathbf{A} - \mathbf{B}\| \le \frac{1}{2\|\mathbf{A}^{-1}\|}.$$

A perturbation theorem from linear algebra [7, Theorem 2.3.4] then shows that

$$\|\mathbf{A}^{-1} - \mathbf{B}^{-1}\| \le \|\mathbf{A}^{-1}\| \le cN^{-q}.$$

This shows that $\|\mathbf{B}^{-1}\| \le cN^{-q}$. ■

We also need some results about approximation of a function and its derivatives. We recall a theorem from [5, Theorem 1°]. Per Section 2.1, the notation $\|\circ\|_1$ indicates the univariate supremum norm.

**Theorem 5.1** *Let $q = 1$, $f \in W^*(\mathbb{T})$, $n \ge 1$ be an integer, $E > 0$, and $T \in \mathbb{H}_n^1$ satisfy*

$$\|f - T\|_1 \le E. \tag{5.16}$$

*Then*

$$\|f' - T'\|_1 \le c\left\{nE + E_n(f')\right\}. \tag{5.17}$$

In the multivariate case, we take the derivatives one variable at a time to deduce the following corollary of Theorem 5.1.

**Corollary 5.1** *Let $f \in W^*$, $N \ge 1$ be an integer, $E > 0$, and $T \in \mathbb{H}_N^q$ satisfy*

$$\|f - T\| \le E. \tag{5.18}$$

*Then*

$$\|f - T\|_{W^*} \le c\left\{NE + \sum_{j=1}^q E_N(D_j f)\right\}. \tag{5.19}$$

*In particular,*

$$\|T\|_{W^*} \le cN\left\{E + \frac{1}{N}\|f\|_{W^*}\right\}. \tag{5.20}$$

We note also the direct theorem of trigonometric approximation [28, Section 5.3].

**Proposition 5.4** *If $f \in W^*$ then*

$$E_N(f) \le \frac{c}{N}\|f\|_{W^*}. \tag{5.21}$$

## 5.2 Proof of Proposition 2.1.

Using (2.8) and Schwarz inequality, we obtain that

$$\begin{aligned}\|\sigma_n(f) - \mathbb{G}_N(\phi, \sigma_n(f))\| &\le c(\phi) E_N(1;\phi) \sum_{\mathbf{k}\in\mathbb{Z}^q} \left| h\left(\frac{|\mathbf{k}|_2}{n}\right) \hat{f}(\mathbf{k}) \right| \\ &\le c(\phi) E_N(1;\phi) \left\{ \sum_{\mathbf{k}\in\mathbb{Z}^q} \left( h\left(\frac{|\mathbf{k}|_2}{n}\right)\right)^2 \right\}^{1/2} \left\{ \sum_{\mathbf{k}\in\mathbb{Z}^q} |\hat{f}(\mathbf{k})|^2 \right\}^{1/2}. \end{aligned} \tag{5.22}$$

Since $h(t) = 0$ if $t \ge 1$, and $0 \le h(t) \le 1$ for all $t$,

$$\sum_{\mathbf{k}\in\mathbb{Z}^q} \left( h\left(\frac{|\mathbf{k}|_2}{n}\right)\right)^2 \le cn^q.$$

In view of Bessel inequality, (5.22) now implies

$$\|\sigma_n(f) - \mathbb{G}_N(\phi, \sigma_n(f))\| \le c(\phi) E_N(1;\phi) n^{q/2} \left( \int_{\mathbb{T}^q} |f(\mathbf{x})|^2 d\mathbf{x} \right)^{1/2} \le c(\phi) E_N(1;\phi) n^{q/2} \|f\|.$$

Together with (5.10), this leads to (2.9). ■

## 5.3 Proof of the theorems in Section 3.

**Proof of Theorem 3.1.**

In this proof, let $B$ be as in Proposition 5.2(a). $N \ge B\eta(\mathcal{C})^{-1}$, and for $\ell = 1, \cdots, M$, $z_\ell = y_\ell - \sigma_N(f)(\mathbf{x}_\ell)$. Proposition 5.3 then guarantees that there exist $a_j \in \mathbb{R}$ such that

$$\sum_{j=1}^M a_j \Phi_N(\mathbf{x}_\ell - \mathbf{x}_j) = z_\ell = y_\ell - \sigma_N(f)(\mathbf{x}_\ell), \qquad \ell = 1, \cdots, M, \tag{5.23}$$

and (cf. (5.15) and (5.10))

$$\begin{aligned}\max_{1\le j\le M} |a_j| &\le cN^{-q} \max_{1\le \ell\le M} |z_\ell| = cN^{-q} \max_{1\le \ell\le M} |y_\ell - \sigma_N(f)(\mathbf{x}_\ell)| \\ &\le cN^{-q} \left\{ \max_{1\le \ell\le M} |y_\ell - f(\mathbf{x}_\ell)| + \max_{1\le \ell\le M} |f(\mathbf{x}_\ell) - \sigma_N(f)(\mathbf{x}_\ell)| \right\} \\ &\le cN^{-q} \left\{ \epsilon + E_{N/2}(f) \right\}. \end{aligned} \tag{5.24}$$

We now recall that

$$\mathcal{T}_N^{\#}(\mathcal{D})(\mathbf{x}) = \sigma_N(f)(\mathbf{x}) + \sum_{j=1}^M a_j \Phi_N(\mathbf{x} - \mathbf{x}_j), \qquad \mathbf{x} \in \mathbb{T}^q. \tag{5.25}$$

Clearly, $\mathcal{T}_N^{\#}(\mathcal{D}) \in \mathbb{H}_N^q$, and $\mathcal{T}_N^{\#}(\mathcal{D})(\mathbf{x}_\ell) = y_\ell$, $\ell = 1, \cdots, M$. This proves (3.5).

Moreover, for every $\mathbf{x} \in \mathbb{T}^q$, (5.25) and (5.10) lead to

$$\begin{aligned}|f(\mathbf{x}) - \mathcal{T}_N^{\#}(\mathcal{D})(\mathbf{x})| &\le |f(\mathbf{x}) - \sigma_N(f)(\mathbf{x})| + \left\{ \max_{1\le j\le M} |a_j| \right\} \sum_{j=1}^M |\Phi_N(\mathbf{x} - \mathbf{x}_j)| \\ &\le c \left\{ E_{N/2}(f) + cN^{-q} \left\{ \epsilon + E_{N/2}(f) \right\} \sum_{j=1}^M |\Phi_N(\mathbf{x} - \mathbf{x}_j)| \right\}. \end{aligned} \tag{5.26}$$

In view of (5.7), this leads to (3.6). ■

**Proof of Theorem 3.2.**

This proof is very similar to that of Theorem 3.1. The condition (3.9) and Proposition 5.3 ensure that the system of equations (3.7) has a unique solution satisfying (5.15). The same argument as in (5.26) then works with the operators as in Theorem 3.2 replacing those in Theorem 3.1; we use (2.9) in place of (5.10). ■

**Proof of Theorem 3.3.**

In this proof, let $\mathcal{T}^\# = T_N^\#(\mathcal{D})$ be as in Theorem 3.1, and

$$E = \epsilon + E_{N/2}(f). \tag{5.27}$$

In view of (3.6), we may use Corollary 5.1 to deduce that

$$\|f - \mathcal{T}^\#\|_{W^*} \le c\left\{NE + \sum_{j=1}^{q} E_N(D_j f)\right\} \le cN\left\{E + \frac{1}{N}\|f\|_{W^*}\right\}. \tag{5.28}$$

and in particular,

$$\|\mathcal{T}^\#\|_{W^*} \le cN\left\{E + \frac{1}{N}\|f\|_{W^*}\right\}. \tag{5.29}$$

Using (3.5), (5.27) and (5.29), we obtain:

$$\begin{aligned}\min_{T\in\mathbb{H}_N} R_N(T) &\le R_N(\mathcal{T}^\#) \le \max_{1\le j\le M} |y_j - \mathcal{T}^\#(\mathbf{x}_j)| + \frac{1}{N}\|\mathcal{T}^\#\|_{W^*} \\ &= \frac{1}{N}\|\mathcal{T}^\#\|_{W^*} \le c\left\{E + \frac{1}{N}\|f\|_{W^*}\right\}. \end{aligned}\tag{5.30}$$

In view of Proposition 5.4,

$$E \le c\left\{\epsilon + \frac{1}{N}\|f\|_{W^*}\right\}.$$

Together with (5.30), this proves (3.14).

Next, let $\mathbf{x} \in \mathbb{T}^q$, and $\delta = \min_{1\le j\le M} |\mathbf{x} - \mathbf{x}_j| = |\mathbf{x} - \mathbf{x}_\ell|$. For brevity, we write

$$\tilde{E} = \epsilon + \frac{1}{N}\|f\|_{W^*}.$$

Using (3.14) and Corollary 5.1, we deduce that

$$\begin{aligned}|f(\mathbf{x}) - T^*(\mathbf{x})| &\le |f(\mathbf{x}) - f(\mathbf{x}_\ell)| + |f(\mathbf{x}_\ell) - T^*(\mathbf{x}_\ell)| + |T^*(\mathbf{x}_\ell) - T^*(\mathbf{x})| \\ &\le |\mathbf{x} - \mathbf{x}_\ell|\|f\|_{W^*} + c\tilde{E} + |\mathbf{x} - \mathbf{x}_\ell|\|T^*\|_{W^*} \le N\delta\frac{1}{N}\|f\|_{W^*} + c\tilde{E} + N\delta\tilde{E} \\ &\le c(1 + N\delta)\tilde{E}. \end{aligned}\tag{5.31}$$

This proves (3.15). ■

**Remark 5.1** If $\delta = \min_{1\le j\le M} |\mathbf{x} - \mathbf{x}_j| = |\mathbf{x} - \mathbf{x}_\ell|$, then we have

$$\begin{aligned}&\left|f(\mathbf{x}) - \frac{1}{\Phi_N(0)}\sum_{k=1}^{M} y_k \Phi_N(\mathbf{x} - \mathbf{x}_k)\right| \\ &\le |f(\mathbf{x}) - f(\mathbf{x}_\ell)| + \left|\frac{1}{\Phi_N(0)}\sum_{k=1}^{M}(y_k - f(\mathbf{x}_k))\Phi_N(\mathbf{x} - \mathbf{x}_k)\right| + \left|f(\mathbf{x}_\ell) - \frac{1}{\Phi_N(0)}\sum_{k=1}^{M} f(\mathbf{x}_k)\Phi_N(\mathbf{x} - \mathbf{x}_k)\right| \\ &\le \delta\|f\|_{W^*} + \epsilon + \frac{1}{\Phi_N(0)}|f(\mathbf{x}_\ell)(\Phi_N(\mathbf{x}_\ell - \mathbf{x}_\ell) - \Phi_N(\mathbf{x} - \mathbf{x}_\ell))| + \frac{1}{\Phi_N(0)}\sum_{k\ne\ell}|f(\mathbf{x}_k)||\Phi_N(\mathbf{x} - \mathbf{x}_k)| \\ &\le \delta\|f\|_{W^*} + c\epsilon + cN\delta\|f\| + \frac{1}{\Phi_N(0)}\sum_{k\ne\ell}|f(\mathbf{x}_k)||\Phi_N(\mathbf{x} - \mathbf{x}_k)|. \end{aligned}$$

Hence, if $2\pi \geq \delta \geq 1/N$, Lemma 5.1 used with $\nu$ as in the proof of Proposition 5.2(a), $d = \eta(\mathcal{C})$, $r = \delta$ shows that

$$\frac{1}{\Phi_N(0)} \sum_{k\neq\ell} |f(\mathbf{x}_k)||\Phi_N(\mathbf{x}-\mathbf{x}_k)| \leq c(N\delta)^{-S}(\delta+\eta(\mathcal{C}))^q \|f\| \leq c(N\delta)^{-S}\|f\|.$$

Therefore, if $1/N \leq \delta \leq 2\pi$,

$$\left| f(\mathbf{x}) - \frac{1}{\Phi_N(0)} \sum_{k=1}^{M} y_k \Phi_N(\mathbf{x}-\mathbf{x}_k) \right| \leq c\{\delta\|f\|_{W^*} + \epsilon + cN\delta\|f\|\}. \tag{5.32}$$

A similar bound can be proved with the polynomial $\mathcal{L}_N(\mathcal{D})$. ■

## 5.4 Proofs of the theorems in Section 4.

The induction argument given in the proof of Theorem 4.1 allows us to "lift" results about shallow networks to deep networks. We will refer to this argument as *good propagation of error*.

**Proof of Theorem 4.1.**
We observe that each node in $V \cup \mathbf{S}$ can be thought of as the sink node of an appropriate sub-DAG $\mathcal{G}_v$ of $\mathcal{G}$. Moreover, in the definitions such as (4.3), $\|f\|_{\mathbb{X},\mathcal{G}} \sim \sum_{v\in V\cup\mathbf{S}} \|f_v\|_{\mathbb{X},\mathcal{G}_v}$, and similarly for (4.4). So, the statement of this theorem should be true also for each such sub-DAG. Let $C = \max_{v\in V\cup\mathbf{S}} B(d(v))$ where $B(d(v))$ is the constant introduced in Proposition 5.2 applied with $d(v)$ in place of $q$.

Theorem 4.1 applied to a vertex in $\mathbf{S}$ is the same as Theorem 3.1.

Suppose the theorem is proved for every node of level up to $\ell$ for some $\ell \geq 0$, $v$ be a node at level $\ell+1$, and $u_1, \cdots, u_{d(v)}$ be its children. The sub-DAG with the outputs of these children as the input to the sink node $v$ is of course a shallow network, to which we may apply Theorem 3.1. Since $N_v \geq C\eta(\mathcal{C}_v)^{-1}$ , Theorem 3.1 implies that there is $T_v^{\#} \in \mathbb{H}_{N_v}^{d(v)}$ satisfying[1]

$$T_v^{\#}(\mathbf{x}) = f_v(\mathbf{x}), \qquad \mathbf{x} \in \mathcal{C}_v, \qquad \|f_v - T_v^{\#}\|_{d(v)} \leq cE_{N_v/2}(d(v); f_v). \tag{5.35}$$

Our induction hypothesis shows that the analogues of the estimates (5.35) and (5.33) are true for each of the children $u_1, \cdots, u_{d(v)}$. Necessarily, $v \in V$, and so, each $f_v$ is Lipschitz continuous with Lipschitz constant $\leq L$. Using triangle inequality, we deduce that

$$\begin{aligned}
&|f_v(f_{u_1}(\mathbf{x}_{u_1}), \cdots, f_{u_{d(v)}}(\mathbf{x}_{u_{d(v)}})) - T_v^{\#}(T_{u_1}^{\#}(\mathbf{x}_{u_1}), \cdots, T_{u_{d(v)}}^{\#}(\mathbf{x}_{u_{d(v)}}))| \\
&\quad\leq |f_v(T_{u_1}^{\#}(\mathbf{x}_{u_1}), \cdots, T_{u_{d(v)}}^{\#}(\mathbf{x}_{u_{d(v)}})) - T_v^{\#}(T_{u_1}^{\#}(\mathbf{x}_{u_1}), \cdots, T_{u_{d(v)}}^{\#}(\mathbf{x}_{u_{d(v)}}))| \\
&\qquad\quad +|f_v(f_{u_1}(\mathbf{x}_{u_1}), \cdots, f_{u_{d(v)}}(\mathbf{x}_{u_{d(v)}})) - f_v(T_{u_1}^{\#}(\mathbf{x}_{u_1}), \cdots, T_{u_{d(v)}}^{\#}(\mathbf{x}_{u_{d(v)}}))| \\
&\quad\leq \sup_{\mathbf{y}\in\mathbb{T}^{d(v)}} |f_v(\mathbf{y}) - T_v^{\#}(\mathbf{y})| + L \left\| (f_{u_1}(\mathbf{x}_{u_1}), \cdots, f_{u_{d(v)}}(\mathbf{x}_{u_{d(v)}})) - (T_{u_1}^{\#}(\mathbf{x}_{u_1}), \cdots, T_{u_{d(v)}}^{\#}(\mathbf{x}_{u_{d(v)}})) \right\|_1 \\
&\quad\leq \|f_v - T_v^{\#}\|_{d(v)} + L\sum_{j=1}^{d(v)} \|f_{u_j} - T_{u_j}^{\#}\|_{C^*,\mathcal{G}_{u_j}}.
\end{aligned}$$

In view of our dual interpretation of the constituent functions as functions of their immediate input as well as the input seen by these functions and the induction hypothesis, this shows that Theorem 4.1 is valid for the sub-DAG with $v$ as the sink node. We define

$$T_{\mathbf{N}}^{\#}(\mathcal{D}) = \{T_v^{\#}\} \in \mathcal{G} - \mathbb{H}_{\mathbf{N}}. \tag{5.36}$$

[1]Here, if $v = v^*$ is the sink node of the original DAG, the statement needs to be modified as follows: there is $T_{v^*}^{\#} \in \mathbb{H}_N^{d(v^*)}$ that satisfies (thought of as a function of all the inputs to the network)

$$T_{v^*}^{\#}(\mathbf{x}_j) = T_{v^*}^{\#}((\mathbf{x}_j)_{v^*}) = y_j, \qquad j = 1, \cdots, M, \tag{5.33}$$

and (thought of as a function on $\mathbb{T}^{d(v^*)}$)

$$\|f_{v^*} - T_{v^*}^{\#}\|_{d(v^*)} \leq c\left\{\epsilon + E_{N_{v^*}/2}(d(v^*); f_{v^*})\right\}. \tag{5.34}$$

The equation (5.33) is the same as (4.5). The estimate (4.6) follows by induction as just explained. ■

**Proof of Theorem 4.2.**

This proof is essentially the same as that of Theorem 3.3. We point out some considerations required in the details which are different. Since each $f_v \in W^*(\mathbb{T}^{d(v)})$, the simultaneous approximation theorem Corollary 5.1 holds for each $f_v$. We may assume that the perturbation exists only at the sink node, and the rest of the data is exact. Finally, we use the good propagation of error to obtain (4.8). The proof of (4.9) involves an argument similar to (5.31). The middle term on the first line of that chain of inequalities gives the first term on the right hand side in (4.9). The other terms can be estimated as in (5.31), except that the chain rule of differentiation needs to be used several times to get to the level of the source nodes. This is facilitated by our extra assumption that $\max_{v\in V} \|f_v\|_{W^*(\mathbb{T}^{(d(v))}} \leq L$. ■

# 6 Conclusions and open problems

We have explored the puzzle that deep networks (and sometimes also shallow ones) do not exhibit over-fitting even though the number of parameters is very large and the training error is reduced to zero. We have initiated a rigorous study of this phenomenon from the point of view of function approximation, giving estimates on how many parameters are needed to exhibit a zero or good training error, which is also compatible with the generalization error. Our estimates are given in terms of the data characteristics and the smoothness of the target function.

One obvious problem is to reduce the number of parameters to be trained in the regularization functional (3.13). Expressing the trigonometric polynomials as linear combinations of the translates $\Phi_N(\circ - 2\pi\mathbf{j}/N)$ ensures that the resulting solution will have coefficients that are small away from the training data. It is therefore reasonable to take only some of these translates that are close to the training data. However, it is not clear that the theory will work with the space defined by the span of only those translates which are kept. Also, there may be some numerical instability problems with this basis.

A deeper and wider area of theoretical investigation is the following. Theorems 3.3 and 4.2 suggest that the approximation error given by the trigonometric polynomials constructed there gives an estimate on the support of the marginal distribution of the training and test data. In [4], the approximation errors of the convergent bounded interpolatory polynomials constructed in [3] were used for texture detection and segmentation of images. What would be the analogues for understanding the nature of the data using the approximation errors obtained here?

# References


[1] M. Belkin, D. J. Hsu, and P. Mitra. Overfitting or perfect fitting? risk bounds for classification and regression rules that interpolate. In *Advances in Neural Information Processing Systems*, pages 2300–2311, 2018.

[2] M. Belkin, S. Ma, and S. Mandal. To understand deep learning we need to understand kernel learning. In *Proceedings of the 35th International Conference on Machine Learning (PMLR)*, pages 80:541–549, 2018.

[3] S. Chandrasekaran, K. R. Jayaraman, and H. N. Mhaskar. Minimum Sobolev norm interpolation with trigonometric polynomials on the torus. *Journal of Computational Physics*, 249:96–112, 2013.

[4] S. Chandrasekaran, K. R. Jayaraman, J. Moffitt, H. N. Mhaskar, and S. Pauli. Minimum Sobolev norm schemes and applications in image processing. In *IS&T/SPIE Electronic Imaging*, pages 753507–753507. International Society for Optics and Photonics, 2010.

[5] J. Czipszer and G. Freud. Sur l'approximation d'une fonction périodique et de ses dérivées successires par un polynome trigonométrique et par ses dérivées successives. *Acta Math.,(Sweden)*, 99:33–51, 1958.

[6] P. Erdős. On some convergence properties of the interpolation polynomials. *Annals of Mathematics*, pages 330–337, 1943.

[7] G. H. Golub and C. F. Van Loan. *Matrix computations*, volume 3. JHU Press, 2012.

[8] I. Goodfellow, Y. Bengio, A. Courville, and Y. Bengio. *Deep learning*, volume 1. MIT press Cambridge, 2016.

[9] Y. Han, L. Sunwoo, and J. C. Ye. k-space deep learning for accelerated MRI. *IEEE transactions on medical imaging*, 2019.

[10] M. Hardt, B. Recht, and Y. Singer. Train faster, generalize better: Stability of stochastic gradient descent. In *ICML'16 Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, pages 1225–1234, 2016.

[11] Q. T. Le Gia and H. N. Mhaskar. Localized linear polynomial operators and quadrature formulas on the sphere. *SIAM Journal on Numerical Analysis*, 47(1):440–466, 2009.

[12] Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.

[13] H. N. Mhaskar. Approximation theory and neural networks. In *Wavelet Analysis and Applications, Proceedings of the international workshop in Delhi*, pages 247–289, 1999.

[14] H. N. Mhaskar. Eignets for function approximation on manifolds. *Applied and Computational Harmonic Analysis*, 29(1):63–87, 2010.

[15] H. N. Mhaskar. Function approximation with zonal function networks with activation functions analogous to the rectified linear unit functions. *Journal of Complexity*, 51:1–19, April 2019.

[16] H. N. Mhaskar and C. A. Micchelli. Degree of approximation by neural and translation networks with a single hidden layer. *Advances in Applied Mathematics*, 16(2):151–183, 1995.

[17] H. N. Mhaskar, F. J. Narcowich, N. Sivakumar, and J. D. Ward. Approximation with interpolatory constraints. *Proceedings of the American Mathematical Society*, 130(5):1355–1364, 2002.

[18] H. N. Mhaskar, P. Nevai, and E. Shvarts. Applications of classical approximation theory to periodic basis function networks and computational harmonic analysis. *Bulletin of Mathematical Sciences*, 3(3):485–549, 2013.

[19] H. N. Mhaskar, S. V. Pereverzyev, and M. D. van der Walt. A deep learning approach to diabetic blood glucose prediction. *Frontiers in Applied Mathematics and Statistics*, 3:14, 2017.

[20] H. N. Mhaskar and T. Poggio. Deep vs. shallow networks: An approximation theory perspective. *Analysis and Applications*, 14(06):829–848, 2016.

[21] I. P. Natanson. *Constructive function theory*. Ungar, 1964.

[22] B. Neyshabur, S. Bhojanapalli, D. McAllester, and N. Srebro. Exploring generalization in deep learning. In *Advances in Neural Information Processing Systems*, pages 5949–5958, 2017.

[23] T. Poggio, K. Kawaguchi, Q. Liao, B. Miranda, L. Rosasco, X. Boix, J. Hidary, and H. Mhaskar. Theory of deep learning III: explaining the non-overfitting puzzle. *arXiv preprint arXiv:1801.00173*, 2017.

[24] T. Poggio, Q. Liao, B. Miranda, A. Banburski, X. Boix, and J. Hidary. Theory IIIb: Generalization in deep networks. *arXiv preprint arXiv:1806.11379*, 2018.

[25] J. Sokolić, R. Giryes, G. Sapiro, and M. R. Rodrigues. Robust large margin deep neural networks. *IEEE Transactions on Signal Processing*, 65(16):4265–4280, 2016.

[26] E. M. Stein. *Singular integrals and differentiability properties of functions (PMS-30)*, volume 30. Princeton university press, 2016.

[27] J. Szabados. On some convergent interpolatory polynomials. *Fourier Analysis and Approximation Theory, Coll. Math. Soc. János Bolyai*, 19:805–815, 1978.

[28] A. F. Timan. *Theory of Approximation of Functions of a Real Variable: International Series of Monographs on Pure and Applied Mathematics*, volume 34. Elsevier, 2014.

[29] C. Zhang, Q. Liao, A. Rakhlin, B. Miranda, N. Golowich, and T. Poggio. Musings on deep learning: Properties of SGD. *CBMM Memo*, 67, 2017.

[30] A. Zygmund. *Trigonometric series*, volume 1. Cambridge University Press, 2002.