Rahul Parhi and Robert D. Nowak

Deep Learning Meets Sparse Regularization

A signal processing perspective

Deep learning (DL) has been wildly successful in practice, and most of the state-of-the-art machine learning methods are based on neural networks (NNs). Lacking, however, is a rigorous mathematical theory that adequately explains the amazing performance of deep NNs (DNNs). In this article, we present a relatively new mathematical framework that provides the beginning of a deeper understanding of DL. This framework precisely characterizes the functional properties of NNs that are trained to fit to data. The key mathematical tools that support this framework include transform-domain sparse regularization, the Radon transform of computed tomography, and approximation theory, which are all techniques deeply rooted in signal processing. This framework explains the effect of weight decay regularization in NN training, use of skip connections and low-rank weight matrices in network architectures, role of sparsity in NNs, and explains why NNs can perform well in high-dimensional problems.

THIS ARTWORK WAS GENERATED USING DALL·E 2 WITH KEYWORDS FROM THE ARTICLE

Introduction

DL has revolutionized engineering and the sciences in the modern data age. The typical goal of DL is to predict an output $y \in \mathcal{Y}$ (e.g., a label or response) from an input $\boldsymbol{x} \in \mathcal{X}$ (e.g., a feature or example). An NN is "trained" to fit to a set of data consisting of the pairs $\{(\boldsymbol{x}_n, y_n)\}_{n=1}^{N}$ by finding a set of NN parameters $\boldsymbol{\theta}$ so that the NN mapping closely matches the data. The trained NN is a function, denoted by $f_{\boldsymbol{\theta}} : \mathcal{X} \to \mathcal{Y}$, that can be used to predict the output $y \in \mathcal{Y}$ of a new input $\boldsymbol{x} \in \mathcal{X}$. This paradigm is referred to as *supervised learning*, which is the focus of this article. The success of DL has spawned a burgeoning industry that is continually developing new applications, NN architectures, and training algorithms. This article reviews recent developments in the mathematics of DL, focused on the characterization of the kinds of functions learned by NNs fit to data. There are currently many competing theories that explain the success of DL. These developments are part of a wider body of theoretical work that can be crudely organized into three broad categories: 1) approximation theory with NNs, 2) the design and analysis of optimization ("training") algorithms for NNs, and 3) characterizations of the properties of trained NNs.

This article belongs to the latter category of research and investigates the functional properties (i.e., the regularity) of solutions to NN training problems with explicit, Tikhonov-type regularization. Although much of the success of DL in

Digital Object Identifier 10.1109/MSP.2023.3286988
Date of current version: 8 September 2023

practice comes from networks with highly structured architectures, it is hard to establish a rigorous and unified theory for such NNs used in practice. Therefore, we primarily focus on fully connected, feedforward NNs with the popular rectified linear unit (ReLU) activation function. This article introduces a mathematical framework that unifies a line of work from several authors over the last few years that sheds light on the nature and behavior of NN functions that are trained to a global minimizer with explicit regularization. The presented results are just one piece of the puzzle toward developing a mathematical theory of DL. The purpose of this article is, in particular, to provide a gentle introduction to this new mathematical framework, accessible to readers with a mathematical background in signals and systems and applied linear algebra. The framework is based on mathematical tools familiar to the signal processing community, including transform-domain sparse regularization, the Radon transform of computed tomography, and approximation theory. It is also related to well-known signal processing ideas such as wavelets, splines, and compressed sensing. This framework provides a new take on the following fundamental questions:

- What is the effect of regularization in DL?
- What kinds of functions do NNs learn?
- What is the role of NN activation functions?
- Why do NNs seemingly break the curse of dimensionality?

NNs and learning from data

The task of DL corresponds to learning the input–output mapping from a dataset in a hierarchical or multilayer manner. DNNs are complicated function mappings built from many smaller, simpler building blocks. The simplest building block of a DNN is an (artificial) neuron, inspired by the biological neurons of the brain [24]. A neuron is a function mapping $\mathbb{R}^d \to \mathbb{R}$ of the form $z \mapsto \sigma(\boldsymbol{w}^\top z - b)$, where $\boldsymbol{w} \in \mathbb{R}^d$ corresponds to the weights of the neuron and $b \in \mathbb{R}$ corresponds to the bias of the neuron. The function $\sigma : \mathbb{R} \to \mathbb{R}$ is referred to as the *activation function* of the neuron and controls the nonlinear response of the neuron. A neuron "activates" when the weighted combination of its input $\boldsymbol{x}$ exceeds a certain threshold, i.e., $\boldsymbol{w}^\top \boldsymbol{x} > b$. Therefore, typical activation functions such as the sigmoid, unit step function, or ReLU activate when their input exceeds zero, as seen in Figure 1.

A neuron is composed of a linear mapping followed by a nonlinearity. A popular form (or "architecture") of a DNN is a fully connected feedforward DNN, which is a cascade of alternating linear mappings and componentwise nonlinearities. A feedforward DNN $f_{\boldsymbol{\theta}}$ (parameterized by $\boldsymbol{\theta}$) can be represented as the function composition

$$f_{\boldsymbol{\theta}}(\boldsymbol{x}) = \boldsymbol{A}^{(L)} \circ \boldsymbol{\sigma} \circ \boldsymbol{A}^{(L-1)} \circ \cdots \boldsymbol{\sigma} \circ \boldsymbol{A}^{(1)}(\boldsymbol{x}) \qquad (1)$$

where for each $\ell = 1, \ldots, L$, the function $\boldsymbol{A}^{(\ell)}(\boldsymbol{z}) = \mathbf{W}^{(\ell)}\boldsymbol{z} - \boldsymbol{b}^{(\ell)}$ is an affine linear mapping with weight matrix $\mathbf{W}^{(\ell)}$ and bias vector $\boldsymbol{b}^{(\ell)}$. The functions $\boldsymbol{\sigma}$ that appear in the composition apply the activation function $\sigma : \mathbb{R} \to \mathbb{R}$ componentwise to the vector $\boldsymbol{A}^{(\ell)}(\boldsymbol{z})$. Although the activation function could change from neuron to neuron, in this article, we assume that the same activation function is used in the entire network. The parameters of this DNN are the weights and biases, i.e., $\boldsymbol{\theta} = \{(\mathbf{W}^{(\ell)}, \boldsymbol{b}^{(\ell)})\}_{\ell=1}^{L}$. Each mapping $\boldsymbol{A}^{(\ell)}$ corresponds to a layer of the DNN, and the number of mappings L is the depth of the DNN. The dimensions of the weight matrices $\mathbf{W}^{(\ell)}$ correspond to the number of neurons in each layer (i.e., the width of the layer). Alternative DNN architectures can be built with other simple building blocks, e.g., with convolutions and pooling/downsampling operations, which would correspond to deep convolutional NNs. DNN architectures are often depicted with diagrams, as in Figure 2.

Given a DNN $f_{\boldsymbol{\theta}}$ parameterized by $\boldsymbol{\theta} \in \Theta$ (of any architecture), the task of learning from the data $\{(\boldsymbol{x}_n, y_n)\}_{n=1}^{N}$ is formulated as the optimization problem

$$\min_{\boldsymbol{\theta} \in \Theta} \sum_{n=1}^{N} \mathcal{L}(y_n, f_{\boldsymbol{\theta}}(\boldsymbol{x}_n)) \qquad (2)$$

where $\mathcal{L}(\cdot, \cdot)$ is a loss function (squared error, logistic, hinge loss, and so on). For example, the squared error loss is given by $\mathcal{L}(y, z) = (y - z)^2$. A DNN is trained by solving this optimization problem, usually via some form of gradient descent. In typical scenarios, this optimization problem is ill-posed, so the problem is regularized either explicitly through the addition of a regularization term and/or implicitly by constraints on the network architecture or the behavior of gradient descent procedures [34]. A surprising phenomenon of gradient descent training algorithms for overparameterized NNs is that, among the many solutions that overfit the data, these algorithms select one that often generalizes well on new data, even without explicit regularization. This has led to researchers trying to understand the role of overparameterization and the effect of random initialization of NN parameters on the implicit bias of gradient-based training algorithms [8].

On the other hand, explicit regularization corresponds to solving an optimization problem of the form

$$\min_{\boldsymbol{\theta} \in \Theta} \sum_{n=1}^{N} \mathcal{L}(y_n, f_{\boldsymbol{\theta}}(\boldsymbol{x}_n)) + \lambda C(\boldsymbol{\theta}) \qquad (3)$$

where $C(\boldsymbol{\theta}) \geq 0$ for all $\boldsymbol{\theta} \in \Theta$. $C(\boldsymbol{\theta})$ is a regularizer, which measures the "size" (or "capacity") of the DNN parameterized by $\boldsymbol{\theta} \in \Theta$, and $\lambda > 0$ is an adjustable hyperparameter, which controls the tradeoff between the data-fitting term and the regularizer. DNNs are often trained using gradient descent

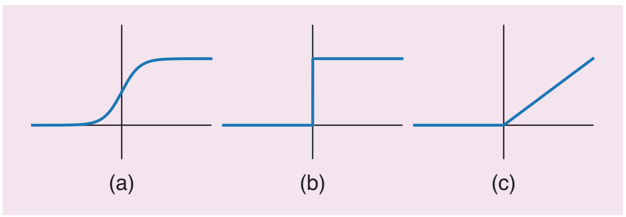


FIGURE 1. The typical activation functions found in NNs. (a) Sigmoid. (b) unit step. (c) ReLU.

algorithms with weight decay, which corresponds to solving the optimization problem

$$\min_{\boldsymbol{\theta} \in \Theta} \sum_{n=1}^{N} \mathcal{L}(y_n, f_{\boldsymbol{\theta}}(\boldsymbol{x}_n)) + \lambda C_{\mathrm{wd}}(\boldsymbol{\theta}) \tag{4}$$

where the weight decay regularizer $C_{\mathrm{wd}}(\boldsymbol{\theta})$ is the squared Euclidean-norm of all the network weights. Sometimes, the weight decay objective regularizes all the parameters, including biases, while sometimes it only regularizes the weights (so that the biases are unregularized). This article focuses on the variant of weight decay with unregularized biases.

What is the effect of regularization in DL?

Weight decay is a common form of regularization for DNNs. On the surface, it appears to simply be the familiar Tikhonov or "ridge" regularization. In standard linear models, it is well known that this sort of regularization tends to reduce the size of the weights but does not produce sparse weights. However, when this regularization is used in conjunction with NNs, the results are strikingly different. Regularizing the sum of squared weights turns out to be equivalent to regularization with a type of ℓ^1-norm regularization on the network weights, leading to sparse solutions in which the weights of many neurons are zero [47]. This is due to the key property that the most commonly used activation functions in DNNs are homogeneous. A function $\sigma(t)$ is said to be homogeneous (of degree 1) if $\sigma(\gamma t) = \gamma\sigma(t)$ for any $\gamma > 0$. The most common NN activation function, the ReLU, is homogeneous as well as the leaky ReLU, linear activation, and pooling/downsampling units. This homogeneity leads to the following theorem, referred to as the *neural balance theorem* (*NBT*) [(47), Th.1] (see "Neural Balance Theorem").

The proof of this theorem boils down to the simple observation that for any homogeneous unit with input weights $\boldsymbol{w}$ and output weights $\boldsymbol{v}$, we can scale the input weight by $\gamma > 0$ and the output weight by $1/\gamma$ without changing the function mapping. For example, consider the single neuron $\boldsymbol{z} \mapsto \boldsymbol{v}\sigma(\boldsymbol{w}^\top \boldsymbol{z} - b)$ with homogeneous activation function σ, as depicted in Figure 2(b). In the case of a DNN, as in (1), $\boldsymbol{w}$ corresponds to a row of a weight matrix in the affine mapping of a layer, $\boldsymbol{v}$ corresponds to a column of the weight matrix in the subsequent layer, and b corresponds to an entry in the bias vector. It is immediate that $(\boldsymbol{v}/\gamma)\sigma((\gamma\boldsymbol{w})^\top \boldsymbol{z} - \gamma b) = \boldsymbol{v}\sigma(\boldsymbol{w}^\top \boldsymbol{z} - b)$. By noting that the biases are unregularized, the theorem follows by noticing that $\min_{\gamma > 0} \|\gamma \boldsymbol{w}\|_2^2 + \|\boldsymbol{v}/\gamma\|_2^2$ occurs when

Neural Balance Theorem

Let $f_{\boldsymbol{\theta}}$ be a deep neural network (DNN) of any architecture parameterized by $\boldsymbol{\theta} \in \Theta$, which solves the DNN training problem with weight decay in (4). Then, the weights satisfy the following balance constraint: if $\boldsymbol{w}$ and $\boldsymbol{v}$ denote the input and output weights of any homogeneous unit in the DNN, respectively, then $\|\boldsymbol{w}\|_2 = \|\boldsymbol{v}\|_2$.

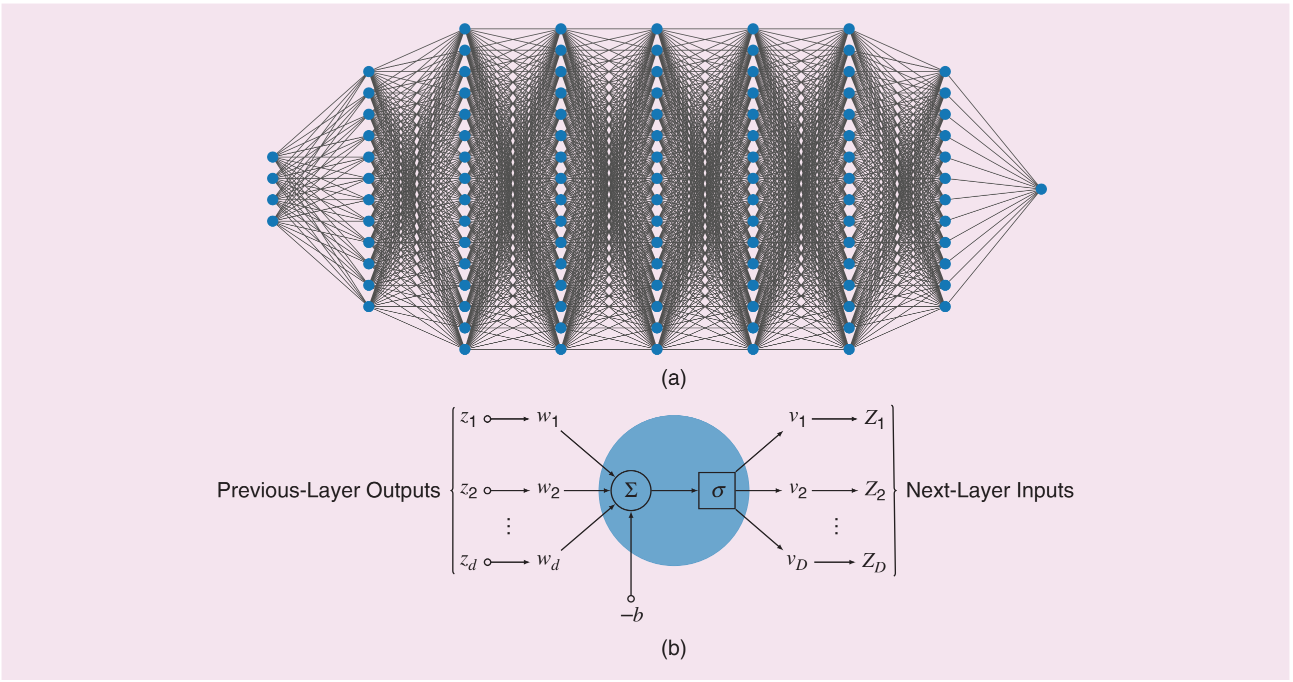


FIGURE 2. An example depiction of a DNN and its components. (a) A feedforward DNN architecture where the nodes represent the neurons and the edges represent the weights. (b) A single neuron from the DNN in (a) mapping an input $\boldsymbol{z} \in \mathbb{R}^d$ to an output $\boldsymbol{Z} \in \mathbb{R}^D$ via $\boldsymbol{Z} = \boldsymbol{v}\sigma(\boldsymbol{w}^\top \boldsymbol{z} - b)$.

$\gamma = \sqrt{\|\boldsymbol{v}\|_2/\|\boldsymbol{w}\|_2}$, which implies that the minimum squared Euclidean-norm solution must satisfy the property that the input and output weights, $\boldsymbol{w}$ and $\boldsymbol{v}$, respectively, are balanced.

The balancing effect of the NBT has a striking effect on solutions to the weight decay objective, particularly a sparsity-promoting effect akin to least absolute shrinkage and selection operator regularization.

The secret sparsity of weight decay

The balancing effect of the NBT has a striking effect on solutions to the weight decay objective, particularly a sparsity-promoting effect akin to least absolute shrinkage and selection operator (LASSO) regularization [41]. As an illustrative example, consider a shallow ($L = 2$), feedforward NN mapping $\mathbb{R}^d \to \mathbb{R}^D$ with a homogeneous activation function (e.g., the ReLU) and K neurons. In this case, the NN is given by

$$f_{\boldsymbol{\theta}}(\boldsymbol{x}) = \sum_{k=1}^{K} \boldsymbol{v}_k \sigma(\boldsymbol{w}_k^{\top}\boldsymbol{x} - b_k). \tag{5}$$

Here, the weight decay regularizer is of the form $(1/2)\Sigma_{k=1}^{K}\|\boldsymbol{v}_k\|_2^2 + \|\boldsymbol{w}_k\|_2^2$, where $\boldsymbol{w}_k$ and $\boldsymbol{v}_k$ are the input and output weights of the kth neuron, respectively. By the NBT, this is equivalent to using the regularizer $\Sigma_{k=1}^{K}\|\boldsymbol{v}_k\|_2\|\boldsymbol{w}_k\|_2$. Due to homogeneity of the activation function, we can assume, without loss of generality, that $\|\boldsymbol{w}_k\|_2 = 1$ by "absorbing" the magnitude of the input weight $\boldsymbol{w}_k$ into the output weight $\boldsymbol{v}_k$. Therefore, by constraining the input weights to be unit norm, the training problem can then be reformulated using the regularizer $\Sigma_{k=1}^{K}\|\boldsymbol{v}_k\|_2$ [47]. Remarkably, this is exactly the well-known group LASSO regularizer [48], which favors solutions with few active neuron connections (i.e., solutions typically have many $\boldsymbol{v}_k$ exactly equal to zero), although the overall training objective remains nonconvex. We also note that there is a line of work that has reformulated the nonconvex training problem as a convex group LASSO problem [32].

More generally, consider the feedforward DNN architecture in (1) with a homogeneous activation function, and consider training the DNN with weight decay only on the network weights. An application of the NBT shows that the weight decay problem is equivalent to the regularized DNN training problem with the regularizer

$$C(\boldsymbol{\theta}) = \frac{1}{2}\sum_{k=1}^{K^{(1)}}\|\boldsymbol{w}_k^{(1)}\|_2^2 + \frac{1}{2}\sum_{k=1}^{K^{(L)}}\|\boldsymbol{v}_k^{(L)}\|_2^2 + \sum_{\ell=1}^{L}\sum_{k=1}^{K^{(\ell)}}\|\boldsymbol{w}_k^{(\ell)}\|_2\|\boldsymbol{v}_k^{(\ell)}\|_2 \tag{6}$$

where $K^{(\ell)}$ denotes the number of neurons in layer ℓ, $\boldsymbol{w}_k^{(\ell)}$ denotes the input weights to the kth neuron in layer ℓ, and $\boldsymbol{v}_k^{(\ell)}$ denotes the output weights to the kth neuron in layer ℓ (see [47, eq. (2)]). The solutions based on this regularizer will also be sparse due to the two norms that appear in the last term in (6) being not squared, akin to the group LASSO regularizer. In particular, this regularizer can be viewed as a mixed $\ell^{2,1}$-norm on the weight matrices. Moreover, increasing the regularization parameter λ, will increase the number of weights that are zero in the solution. Therefore, training the DNN with weight decay favors sparse solutions, where sparsity is quantified via the number of active neuron connections. An early version of this result appeared in 1998 [16], although it did not become well known until it was rediscovered in 2015 [25].

What kinds of functions do NNs learn?

The sparsity-promoting effect of weight decay arising from the NBT in network architectures with homogeneous activation functions has several consequences on the properties of trained NNs. In this section, we focus on the popular ReLU activation function $\rho(t) = \max\{0, t\}$. The imposed sparsity not only promotes sparsity in the sense of the number of active neuron connections but also promotes a kind of transform-domain sparsity. This transform-domain sparsity suggests the inclusion of skip connections and low-rank weight matrices in network architectures.

Shallow NNs

In the univariate case, a shallow feedforward ReLU NN with K neurons is realized by the mapping

$$f_{\boldsymbol{\theta}}(x) = \sum_{k=1}^{K} v_k \rho(w_k x - b_k). \tag{7}$$

Training this NN with weight decay corresponds to the solving the optimization problem

$$\min_{\boldsymbol{\theta}\in\Theta}\sum_{n=1}^{N}\mathcal{L}(y_n, f_{\boldsymbol{\theta}}(x_n)) + \frac{\lambda}{2}\sum_{k=1}^{K}|v_k|^2 + |w_k|^2. \tag{8}$$

From the "The Secret Sparsity of Weight Decay" section, we saw that the NBT implies that this problem is equivalent to

$$\min_{\boldsymbol{\theta}\in\Theta}\sum_{n=1}^{N}\mathcal{L}(y_n, f_{\boldsymbol{\theta}}(x_n)) + \lambda\sum_{k=1}^{K}|v_k||w_k|. \tag{9}$$

As illustrated in "Rectified Linear Unit Sparsity in the Second Derivative Domain," we see that (9) is actually regularizing the integral of the second derivative of the NN, which can be viewed as a measure of sparsity in the second derivative domain. The integral in (S6) must be understood in the distributional sense because the Dirac impulse is not a function, but a generalized function or distribution. To make this precise, let $g_\varepsilon(x) = e^{-x^2/2\varepsilon}/\sqrt{2\pi\varepsilon}$ denote the Gaussian density with variance $\varepsilon > 0$. As is well known in signal processing, g_ε converges to the Dirac impulse as $\varepsilon \to 0$. Using this idea, given a distribution f, define the norm

$$\|f\|_{\mathcal{M}} := \sup_{\varepsilon>0} \|f * g_\varepsilon\|_{L^1} = \sup_{\varepsilon>0} \int_{-\infty}^{\infty} \left| \int_{-\infty}^{\infty} f(x) g_\varepsilon(y-x) \mathrm{d}x \right| \mathrm{d}y. \tag{10}$$

This definition provides an explicit construction, via the convolution with a Gaussian, of a sequence of smooth functions that converge to *f*, where the supremum acts as the limit. For example, if $f(x) = g(x) + \Sigma_{k=1}^{K} v_k \delta(x - t_k)$, where g is an absolutely integrable function, then $\|f\|_{\mathcal{M}} = \|g\|_{L^1} + \Sigma_{k=1}^{K} |v_k| = \|g\|_{L^1} + \|\boldsymbol{v}\|_1$, with $\|\boldsymbol{v}\|_1 = \Sigma_{k=1}^{K} |v_k|$. It is in this sense that (S6) holds, i.e., $\|\mathrm{D}^2 f_{\boldsymbol{\theta}}\|_{\mathcal{M}} = \Sigma_{k=1}^{K} |v_k||w_k|$. In particular, the $\mathcal{M}$-norm is precisely the continuous-domain analog of the sparsity-promoting discrete ℓ^1-norm. Therefore, we see that training an NN with weight decay, as in (8), prefers solutions with sparse second derivatives.

It turns out that the connection between sparsity in the second derivative domain and NNs is even tighter. Let $\mathrm{BV}^2(\mathbb{R})$ denote the space of functions mapping $\mathbb{R} \to \mathbb{R}$ such that $\|\mathrm{D}^2 f\|_{\mathcal{M}}$ is finite. This is the space of functions of second-order *bounded variation* and the quantity $\|\mathrm{D}^2 f\|_{\mathcal{M}}$ is the second-order total variation (TV) of *f*. Note that the classical notion of TV, often used in signal denoising problems [35], is $\mathrm{TV}(f) := \|\mathrm{D} f\|_{\mathcal{M}}$, and so the second-order TV of *f* can be viewed as the TV of the derivative of $f: \|\mathrm{D}^2 f\|_{\mathcal{M}} = \mathrm{TV}(\mathrm{D} f)$.

It is well known from spline theory [14], [23], [44] that functions that fit data and have minimal second-order TV are continuous piecewise linear (CPwL) functions. As the ReLU is a CPwL function, ReLU NNs are CPwL functions [3]. In fact, under mild assumptions on the loss function, the solution set to the optimization problem

$$\min_{f \in \mathrm{BV}^2(\mathbb{R})} \sum_{n=1}^{N} \mathcal{L}(y_n, f(x_n)) + \lambda \|\mathrm{D}^2 f\|_{\mathcal{M}} \tag{11}$$

is completely characterized by NNs of the form

$$f_{\boldsymbol{\theta}}(x) = \sum_{k=1}^{K} v_k \rho(w_k x - b_k) + c_1 x + c_0 \tag{12}$$

where the number of neurons is strictly less than the number of data ($K < N$) in the sense that the solution set to (11) is

Rectified Linear Unit Sparsity in the Second Derivative Domain

Given a rectified linear unit neuron $r(x) = \rho(wx - b)$, its first derivative, $\mathrm{D}\, r(\boldsymbol{x})$, is

$$\begin{aligned} \mathrm{D}\, r(x) &= \mathrm{D}\, \rho(wx - b) \\ &= wu(wx - b) \end{aligned} \tag{S1}$$

where u is the unit step function [Figure 1(b)]. Therefore, its second derivative, $\mathrm{D}^2 r(x)$, is

$$\begin{aligned} \mathrm{D}^2 r(x) &= \mathrm{D}\, wu(wx - b) \\ &= w^2 \delta(wx - b). \end{aligned} \tag{S2}$$

By the scaling property of the Dirac impulse [S1, Problem 1.38(a)]

$$\delta(\gamma x) = \frac{1}{|\gamma|} \delta(x) \tag{S3}$$

we have

$$\begin{aligned} \mathrm{D}^2 r(x) &= \frac{w^2}{|w|} \delta\left(x - \frac{b}{w}\right) \\ &= |w| \delta\left(x - \frac{b}{w}\right). \end{aligned} \tag{S4}$$

The second derivative of the neural network (7) is then

$$\mathrm{D}^2 f_{\boldsymbol{\theta}}(x) = \sum_{k=1}^{K} v_k |w_k| \delta\left(x - \frac{b_k}{w_k}\right). \tag{S5}$$

Therefore,

$$\int_{-\infty}^{\infty} |\mathrm{D}^2 f_{\boldsymbol{\theta}}(x)| \mathrm{d}x = \sum_{k=1}^{K} |v_k||w_k|. \tag{S6}$$

Reference

[S1] A. V. Oppenheim, A. S. Willsky, and S. H. Nawab, *Signals and Systems* (Prentice-Hall Signal Processing Series). Englewood Cliffs, NJ, USA: Prentice-Hall, 1997.

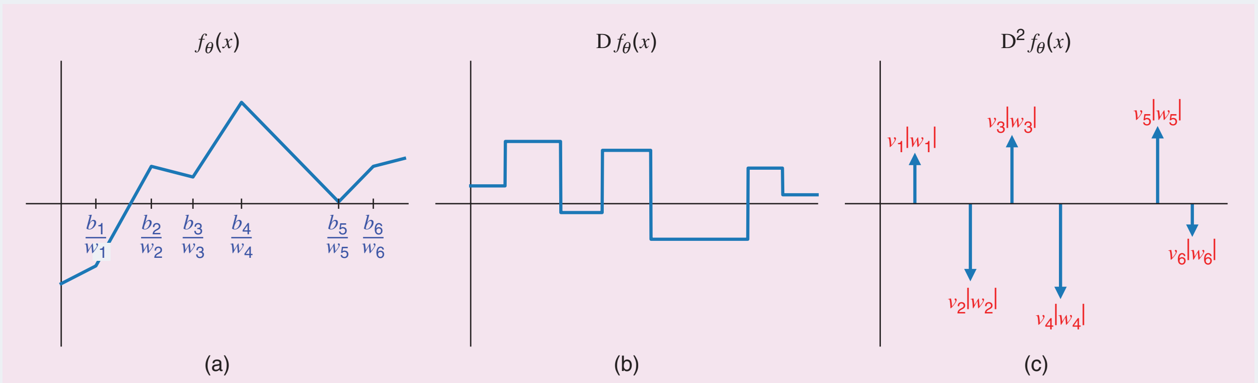


FIGURE S1. An illustration of the sparsity in the second derivative domain of a univariate, shallow feedforward neural network with six neurons.

a closed convex set whose extreme points take the form of (12) with $K < N$ [9], [28], [37]. In NN parlance, the $c_1 x + c_0$ term is a skip connection [17]. This term is an affine function that naturally arises because the second-order TV of an affine function is zero, and so the regularizer places no penalty for the inclusion of this term.

The intuition behind this result is due to the fact that the second derivative of a CPwL function is an impulse train and therefore exhibits extreme sparsity in the second derivative domain, as illustrated in Figure S1. Therefore, the optimization problem (11) will favor sparse CPwL functions that always admit a representation, as in (12). In signal processing parlance, "signals" that are sparse in some transform domain are said to have a finite rate of innovation [45]. Here, the involved transform is the second derivative operator, and the innovation is the impulse train that arises after applying the second derivative operator to a CPwL function.

Consider the optimization over the NN parameter space Θ_K of networks, as in (12), with fixed-width $K \geq N$. From the derivation in "Rectified Linear Unit Sparsity in the Second Derivative Domain," we have $\{f_{\boldsymbol{\theta}} : \boldsymbol{\theta} \in \Theta_K\} \subset \mathrm{BV}^2(\mathbb{R})$. Furthermore, from the earlier discussion, we know that there always exists an optimal solution to (11) that takes the form of (12) with $K < N$, i.e., there always exists a solution to (11) in $\{f_{\boldsymbol{\theta}} : \boldsymbol{\theta} \in \Theta_K\}$. Therefore, from the equivalence of (8) and (9), we see that training a sufficiently wide $(K \geq N)$ NN with a skip connection (12) and weight decay (8) results in a solution to the optimization problem (11) over the function space $\mathrm{BV}^2(\mathbb{R})$. Although this result may seem obvious in hindsight, it is remarkable because it says that the kinds of functions that NNs trained with weight decay (to a global minimizer) are exactly optimal functions in $\mathrm{BV}^2(\mathbb{R})$. Moreover, this result sheds light on the role of overparameterization as this correspondence hinges on the network being critically parameterized or overparameterized (because $K \geq N$).

It is well known from spline theory that functions that fit data and have minimal second-order TV are continuous piecewise linear functions.

In the multivariate case, a shallow feedforward NN has the form

$$f_{\boldsymbol{\theta}}(\boldsymbol{x}) = \sum_{k=1}^{K} v_k \rho(\boldsymbol{w}_k^\top \boldsymbol{x} - b_k). \tag{13}$$

The key property that connects the univariate case and $\mathrm{BV}^2(\mathbb{R})$ is that ReLU neurons are sparsified by the second derivative operator, as in (S4). A similar analysis can be carried out in the multivariate case by finding an operator that is the sparsifying transform of the multivariate ReLU neuron $r(\boldsymbol{x}) = \rho(\boldsymbol{w}^\top \boldsymbol{x} - b)$. The sparsifying transform was proposed in 2020 in the seminal work of Ongie et al. [26] and hinges on the Radon transform that arises in tomographic imaging. Connections between the Radon transform and neural networks have been known since at least the 1990s, gaining popularity due to the proposal of ridgelets [6], and early versions of the sparsifying transform for neurons were studied as early as 1997 [19]. A summary of the Radon transform appears in "The Radon Transform and Fourier Slice Theorem." The sparsifying transform for multivariate ReLU neurons is based on a result regarding the (filtered) Radon transform, which appears in "Filtered Radon Transform of a Neuron With Unit-Norm Input Weights."

The filter K in (S13) is exactly the backprojection filter that arises in the filtered backprojection algorithm in tomographic image reconstruction and acts as a high-pass filter (or ramp filter) to correct the attenuation of high frequencies from the Radon transform. The intuition behind this is that the Radon transform integrates a function along hyperplanes. In the

The Radon Transform and Fourier Slice Theorem

The Radon transform, first studied by Radon in 1917 [S2], of a function mapping $\mathbb{R}^d \to \mathbb{R}$ is specified by the integral transform

$$\mathscr{R}\{f\}(\boldsymbol{\alpha}, t) = \int_{\mathbb{R}^d} f(\boldsymbol{x}) \delta(\boldsymbol{\alpha}^\top \boldsymbol{x} - t) \,\mathrm{d}\boldsymbol{x} \tag{S7}$$

where δ is the univariate Dirac impulse, $\boldsymbol{\alpha} \in \mathbb{S}^{d-1} = \{\boldsymbol{u} \in \mathbb{R}^d : \|\boldsymbol{u}\|_2 = 1\}$ is a unit vector, and $t \in \mathbb{R}$ is a scalar. The Radon transform of f at $(\boldsymbol{\alpha}, t)$ is the integral of f along the hyperplane $\{\boldsymbol{x} \in \mathbb{R}^d : \boldsymbol{\alpha}^\top \boldsymbol{x} = t\}$.

The Radon transform is tightly linked with the Fourier transform, specified by

$$\hat{f}(\boldsymbol{\omega}) = \int_{\mathbb{R}^d} f(\boldsymbol{x}) e^{-\mathrm{j}\boldsymbol{\omega}^\top \boldsymbol{x}} \,\mathrm{d}\boldsymbol{x} \tag{S8}$$

where $\mathrm{j}^2 = -1$. Indeed,

$$\begin{aligned}
&\widehat{\mathscr{R}\{f\}}(\boldsymbol{\alpha}, \omega) \\
&= \int_{\mathbb{R}} \left(\int_{\mathbb{R}^d} f(\boldsymbol{x}) \delta(\boldsymbol{\alpha}^\top \boldsymbol{x} - t) \,\mathrm{d}\boldsymbol{x} \right) e^{-\mathrm{j}\omega t} \,\mathrm{d}t \\
&= \int_{\mathbb{R}^d} f(\boldsymbol{x}) \left(\int_{\mathbb{R}} \delta(\boldsymbol{\alpha}^\top \boldsymbol{x} - t) e^{-\mathrm{j}\omega t} \,\mathrm{d}t \right) \mathrm{d}\boldsymbol{x} \\
&= \int_{\mathbb{R}^d} f(\boldsymbol{x}) e^{-\mathrm{j}(\omega \boldsymbol{\alpha})^\top \boldsymbol{x}} \,\mathrm{d}\boldsymbol{x} = \hat{f}(\omega \boldsymbol{\alpha}).
\end{aligned} \tag{S9}$$

This result is known as the *Fourier slice theorem*. It states that the univariate Fourier transform in the Radon domain corresponds to a slice of the Fourier transform in the spatial domain.

Reference

[S2] J. Radon, "Über die bestimmung von funktionen durch ihre integralwerte längs gewisser mannigfaltigkeiten," *Berichte Verhandlungen Sächsische Akademie Wissenschaften*, vol. 69, pp. 262–277, 1917.

univariate case, the magnitude of the frequency response of an integrator behaves as $1/|\omega|$ and therefore attenuates high frequencies. The magnitude of the frequency response of integration along a hyperplane therefore behaves as $1/|\omega|^{d-1}$ as hyperplanes are of dimension $(d-1)$. Note that the even projector that appears in (S17) is due to the fact that the Radon-domain variables $(\boldsymbol{\alpha},t)$ and $(-\boldsymbol{\alpha},-t)$ parameterize the same hyperplane.

From the derivation in "Filtered Radon Transform of a Neuron With Unit-Norm Input Weights," we immediately see that the sparsifying transform of the multivariate ReLU neuron $r(\boldsymbol{x})=\rho(\boldsymbol{w}^\top\boldsymbol{x}-b)$ with $(\boldsymbol{w},b)\in\mathbb{S}^{d-1}\times\mathbb{R}$ is the $\mathrm{D}_t^2\mathrm{K}\mathscr{R}$ operator, where $\mathrm{D}_t^2=\partial^2/\partial t^2$ denotes the second-order partial derivative with respect to t. We have

$$\mathrm{D}_t^2\mathrm{K}\mathscr{R}\{r\}(\boldsymbol{\alpha},t)=\delta_{\mathscr{R}}((\boldsymbol{\alpha},t)-(\boldsymbol{w},b)) \qquad (14)$$

where $\delta_{\mathscr{R}}(z-z_0):=\mathrm{P}_{\mathrm{even}}\{\delta(z-z_0)\}=(\delta(z-z_0)+\delta(z+z_0))/2$ is the even symmetrization of the Dirac impulse that arises due to the even symmetry of the Radon domain. From the homogeneity of the ReLU activation, applying this sparsifying transform to the (unconstrained) neuron $r(\boldsymbol{x})=\rho(\boldsymbol{w}^\top\boldsymbol{x}-b)$ with $(\boldsymbol{w},b)\in\mathbb{R}^d\times\mathbb{R}$ yields

$$\mathrm{D}_t^2\mathrm{K}\mathscr{R}\{r\}(\boldsymbol{\alpha},t)=\|\boldsymbol{w}\|_2\delta_{\mathscr{R}}((\boldsymbol{\alpha},t)-(\tilde{\boldsymbol{w}},\tilde{b})) \qquad (15)$$

where $\tilde{\boldsymbol{w}}=\boldsymbol{w}/\|\boldsymbol{w}\|_2$ and $\tilde{b}=b/\|\boldsymbol{w}\|_2$. This is analogous to how D^2 is the sparsifying transform for univariate neurons, as in (S4). The sparsifying operator is simply the second derivative in the filtered Radon domain. The key idea is that the (filtered) Radon transform allows us to extract the (univariate) activation function from the multivariate neuron and apply the univariate sparsifying transform in the t variable. Figure 3 is a cartoon diagram that depicts the sparsifying transform of a ReLU neuron.

The story is now analogous to the univariate case. Indeed, by the NBT, training the NN in (13) with weight decay is equivalent to solving the optimization problem

$$\min_{\boldsymbol{\theta}\in\Theta}\sum_{n=1}^{N}\mathcal{L}(y_n,f_{\boldsymbol{\theta}}(\boldsymbol{x}_n))+\lambda\sum_{k=1}^{K}|v_k|\|\boldsymbol{w}_k\|_2. \qquad (16)$$

From (15), we see that $\|\mathrm{D}_t^2\mathrm{K}\mathscr{R}f_{\boldsymbol{\theta}}\|_{\mathcal{M}}=\Sigma_{k=1}^{K}|v_k|\|\boldsymbol{w}_k\|_2$, so training the NN (13) with weight decay prefers solutions with sparse second derivatives in the filtered Radon domain. This measure of sparsity can be viewed as the second-order TV in the (filtered) Radon domain. Let $\mathscr{R}\mathrm{BV}^2(\mathbb{R}^d)$ denote the space of functions on $\mathbb{R}^d$ of second-order bounded variation in the (filtered) Radon domain (i.e., the second-order TV

Filtered Radon Transform of a Neuron With Unit-Norm Input Weights

First, consider the neuron $r(\boldsymbol{x})=\sigma(\boldsymbol{w}^\top\boldsymbol{x})$ with $\boldsymbol{w}=\boldsymbol{e}_1=(1,0,\ldots,0)$ (the first canonical unit vector). In this case, $r(\boldsymbol{x})=\sigma(x_1)$. By noticing that this function can be written as a tensor product, the Fourier transform is given by the following product:

$$\hat{r}(\boldsymbol{\omega})=\hat{\sigma}(\omega_1)\prod_{k=2}^{d}2\pi\delta(\omega_k). \qquad (\mathrm{S10})$$

By the Fourier slice theorem

$$\widehat{\mathscr{R}\{r\}}(\boldsymbol{\alpha},\omega)=\hat{\sigma}(\omega\alpha_1)\prod_{k=2}^{d}2\pi\delta(\omega\alpha_k). \qquad (\mathrm{S11})$$

By the scaling property of the Dirac impulse (S3), the quantity in (S11) equals

$$=\hat{\sigma}(\omega\alpha_1)\frac{(2\pi)^{d-1}}{|\omega|^{d-1}}\delta(\alpha_2,\ldots,\alpha_d). \qquad (\mathrm{S12})$$

If we define the filter via the frequency response

$$\widehat{\mathrm{K}f}(\omega)=\frac{|\omega|^{d-1}}{2(2\pi)^{d-1}}\hat{f}(\omega) \qquad (\mathrm{S13})$$

we find

$$\widehat{\mathrm{K}\mathscr{R}\{r\}}(\boldsymbol{\alpha},\omega)=\frac{\hat{\sigma}(\omega\alpha_1)}{2}\delta(\alpha_2,\ldots,\alpha_d). \qquad (\mathrm{S14})$$

Taking the inverse Fourier transform,

$$\begin{aligned}&\mathrm{K}\mathscr{R}\{r\}(\boldsymbol{\alpha},t)\\&=\frac{1}{2|\alpha_1|}\sigma\left(\frac{t}{\alpha_1}\right)\delta(\alpha_2,\ldots,\alpha_d)\\&=\frac{\sigma(t)\delta(\alpha_1-1)+\sigma(-t)\delta(\alpha_1+1)}{2}\delta(\alpha_2,\ldots,\alpha_d)\\&=\frac{\sigma(t)\delta(\boldsymbol{\sigma}-\boldsymbol{e}_1)+\sigma(-t)\delta(\boldsymbol{\alpha}+\boldsymbol{e}_1)}{2}\\&=:\mathrm{P}_{\mathrm{even}}\{\sigma(t)\delta(\boldsymbol{\alpha}-\boldsymbol{e}_1)\}\end{aligned} \qquad (\mathrm{S15})$$

where $\mathrm{P}_{\mathrm{even}}$ is the projector, which extracts the even part of its input [in terms of the variables $(\boldsymbol{\alpha},t)$]. The second line holds by the dilation property of the Fourier transform [S1, eq. (4.34)].

$$\frac{1}{|\gamma|}f\left(\frac{t}{\gamma}\right)\xleftrightarrow{\mathscr{F}}\hat{f}(\gamma\omega). \qquad (\mathrm{S16})$$

As $\boldsymbol{\alpha}\in\mathbb{S}^{d-1}$, the third line holds by observing that when $\alpha_1=\pm1,\alpha_2,\ldots,\alpha_d=0$, the second line is $\sigma(\pm t)/2$ multiplied by an impulse, and when $\alpha_1\neq\pm1$, the second line is zero, which is exactly the third line. By the rotation properties of the Fourier transform, we have the following result for the neuron $r(\boldsymbol{x})=\sigma(\boldsymbol{w}^\top\boldsymbol{x})$:

$$\mathrm{K}\mathscr{R}\{r\}(\boldsymbol{\alpha},t)=\mathrm{P}_{\mathrm{even}}\{\sigma(t)\delta(\boldsymbol{\alpha}-\boldsymbol{w})\} \qquad (\mathrm{S17})$$

where $\mathbf{w}\in\mathbb{S}^{d-1}$.

in the (filtered) Radon domain is finite). A key result regarding $\mathscr{R}\mathrm{BV}^2(\mathbb{R}^d)$ is the following representer theorem for NNs, first proven in [29]. Under mild assumptions on the loss function, the solution set to the optimization problem

$$\min_{f \in \mathscr{R}\mathrm{BV}^2(\mathbb{R}^d)} \sum_{n=1}^{N} \mathcal{L}(y_n, f(\boldsymbol{x}_n)) + \lambda \| \mathrm{D}_t^2 \mathrm{K} \mathscr{R} f \|_{\mathcal{M}} \tag{17}$$

is completely characterized by NNs of the form

$$f_{\boldsymbol{\theta}}(x) = \sum_{k=1}^{K} v_k \rho(\boldsymbol{w}_k^\top \boldsymbol{x} - b_k) + \boldsymbol{c}^\top \boldsymbol{x} + c_0 \tag{18}$$

where the number of neurons is strictly less than the number of data ($K < N$) in the sense that the solution set to (17) is a closed convex set whose extreme points take the form of (18) with $K < N$ (see [5], [27], and [43] for further refinements of this result). Common loss functions such as the squared-error satisfy the mild assumptions. The skip connection $\boldsymbol{c}^\top \boldsymbol{x} + c_0$ arises because the null space of the sparsifying transform is the space of affine functions. Therefore, by the same argument presented in the univariate case, sufficiently wide ($K \geq N$) NNs [as in (18)] trained with weight decay to global minimizers are exactly optimal functions in $\mathscr{R}\mathrm{BV}^2(\mathbb{R}^d)$.

DNNs

The machinery is straightforward to extend to the case of DNNs. The key idea is to consider fitting data using compositions of $\mathscr{R}\mathrm{BV}^2$ functions. It is shown in [30] and [39] that under mild assumptions on the loss function, a solution to the optimization problem

$$\min_{f^{(1)},\dots,f^{(L)}} \sum_{n=1}^{N} \mathcal{L}(y_n, f^{(L)} \circ \cdots \circ f^{(1)}(\boldsymbol{x}_n)) + \lambda \sum_{\ell=1}^{L} \sum_{i=1}^{d_\ell} \| \mathrm{D}_t^2 \mathrm{K} \mathscr{R} f_i^{(\ell)} \|_{\mathcal{M}} \tag{19}$$

has the form of a DNN, as in (1), where d_ℓ are the intermediary dimensions in the function compositions, which satisfy the following properties:

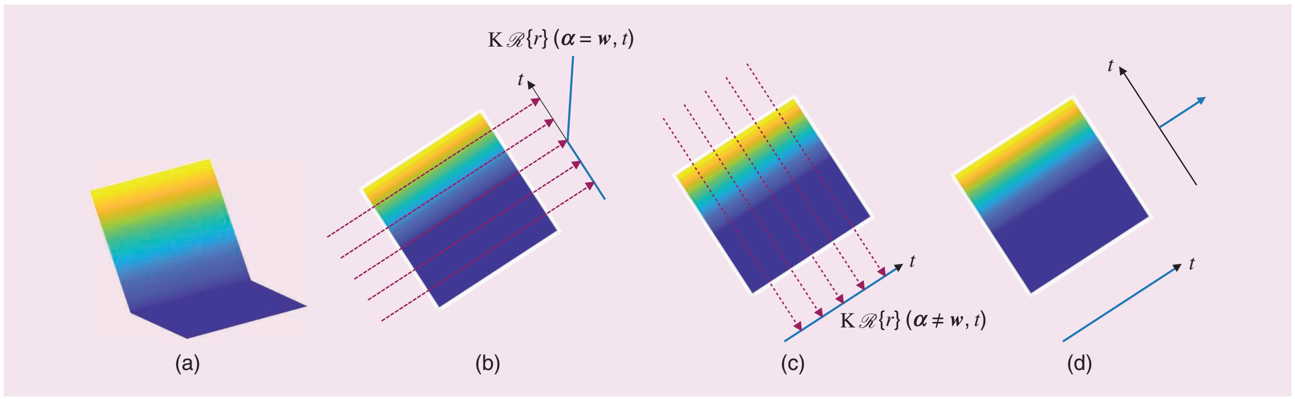
- The number of layers is $L + 1$.
- The solution is sparse in the sense of having few active neuron connections (the widths of the layers are bounded by N^2).
- The solution has skip connections in all layers.
- The architecture has linear bottlenecks, which force the weight matrices to be low rank.

Such an architecture is illustrated in Figure 4. The result shows that ReLU DNNs with skip connections and linear bottlenecks trained with a variant of weight decay [30, Remark 4.7] are optimal solutions to fitting data using compositions of $\mathscr{R}\mathrm{BV}^2$ functions. Linear bottlenecks may be written as factorized (low-rank) weight matrices of the form $\mathbf{W}^{(\ell)} = \mathbf{U}^{(\ell)}\mathbf{V}^{(\ell)}$. These bottleneck layers correspond to layers with linear activation functions ($\sigma(t) = t$). They arise naturally due to the compositions of functions that arise in (19). The number of neurons in each bottleneck layers is bounded by d_ℓ. The incorporation of linear bottlenecks of this form in DNNs have been shown to speed up learning [1] and increase the accuracy [15], robustness [36], and computational efficiency [46] of DNNs.

What is the role of NN activation functions?

The primary focus of the article so far has been the ReLU activation function $\rho(t) = \max\{0, t\}$. Many of the previously discussed ideas can be extended to a broad class of activation functions. The property of the ReLU exploited so far has been that it is sparsified by the second derivative operator in the sense that $\mathrm{D}_t^2 \rho = \delta$. Indeed, we can define a broad class of neural function spaces akin to $\mathscr{R}\mathrm{BV}^2(\mathbb{R}^d)$ by defining spaces characterized by different sparsifying transforms matched to an activation function. This entails replacing D_t^2 in (14) with a generic sparsifying transform H. Table 1 (adapted from [42]) provides examples of common activation functions that fall into this framework, where each sparsifying transform H is defined by its frequency response $\hat{H}(\omega)$. For the ReLU, we have $\mathrm{H} = \mathrm{D}_t^2$, and so $\hat{H}(\omega) = (\,j\omega)^2 = -\omega^2$.

Therefore, many of the previously discussed results can thus be directly extended to a broad class of activation functions, including the classical sigmoid and arctan activation



$\mathrm{K}\mathscr{R}\{r\}\,(\boldsymbol{\alpha} = \boldsymbol{w}, t)$
t
t
$\mathrm{K}\mathscr{R}\{r\}\,(\boldsymbol{\alpha} \neq \boldsymbol{w}, t)$
t
t
(a)
(b)
(c)
(d)

FIGURE 3. A cartoon diagram illustrating the sparsifying transform of the ReLU neuron $r(\boldsymbol{x}) = \rho(\boldsymbol{w}^\top \boldsymbol{x} - b)$ with $(\boldsymbol{w}, b) \in \mathbb{S}^{d-1} \times \mathbb{R}$. The heatmap is a top-down view of the ReLU neuron depicted in (a). (a) The surface plot of $r(\boldsymbol{x})$. (b) The filtered Radon transform when $\boldsymbol{\alpha} = \boldsymbol{w}$. (c) The filtered Radon transform when $\boldsymbol{\alpha} \neq \boldsymbol{w}$. (d) Sparsifying transform $\mathrm{D}_t^2 \mathrm{K} \mathscr{R}\{r\}$.

functions. We remark that the sparsity-promoting effect of weight decay hinges on the homogeneity of the activation function in the DNN. Although the ReLU and truncated power activation functions in Table 1 are homogeneous, the other activation functions are not. This provides evidence that one should prefer homogeneous activation functions like the ReLU to exploit the tight connections between weight decay and sparsity. Although the sparsity-promoting effect of weight decay does not apply to the nonhomogeneous activation functions, statements akin to (14) do hold by considering neurons with input weights constrained to be unit-norm. Therefore, these sparsifying transforms uncover the innovations of finite-width NNs with unit-norm input weights. Therefore, by only considering neurons with unit-norm input weights, the key results that characterize the solution sets to the optimization problems akin to (17) and (19) hold, providing insight into the kinds of functions favored by DNNs using these activation functions.

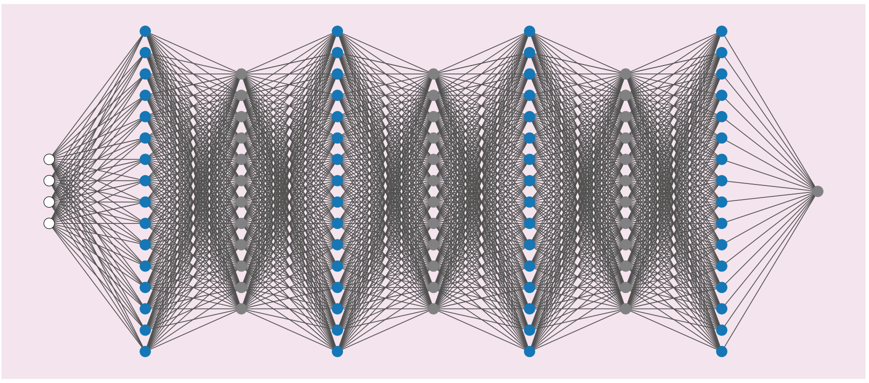


FIGURE 4. A feedforward DNN architecture with linear bottlenecks. The blue nodes represent ReLU neurons, gray nodes represent linear neurons, and white nodes depict the DNN inputs. As the linear layers are narrower than the ReLU ones, this architecture is referred to as a *DNN with linear bottlenecks*.

Why do NNs seemingly break the curse of dimensionality?

In 1993, Barron published his seminal paper [4] on the ability of NNs with sigmoid activation functions to approximate a wide variety of multivariate functions. Remarkably, he showed that NNs can approximate functions that satisfy certain decay conditions on their Fourier transforms at a rate that is completely independent of the input dimension of the functions. This property has led to many people heralding his work as "breaking the curse of dimensionality." Today, the function spaces he studied are often referred to as the *spectral Barron spaces*. It turns out that this remarkable approximation property of NNs is due to sparsity.

To explain this phenomenon, we first recall a problem that "suffers the curse of dimensionality." A classical problem in signal processing is reconstructing a signal from its samples. Shannon's sampling theorem asserts that sampling a band-limited signal on a regular grid at a rate faster than the Nyquist rate guarantees that the sinc interpolator perfectly reconstructs the signal. As the sinc function and its shifts form an orthobasis for the space of band-limited signals, the energy of the signal (squared L^2-norm) corresponds to the squared (discrete) ℓ^2-norm of its samples. Multivariate versions of the sampling theorem are similar and assert that sampling multivariate, band-limited signals on a sufficiently fine regular grid guarantees perfect reconstruction with (multivariate) sinc interpolation. It is easy to see that the grid size (and therefore the number of samples) grows exponentially with the dimension of the signal. This shows that the sampling and reconstruction of band-limited signals suffers the curse of dimensionality. The fundamental reason for this is that the energy or "size" of a band-limited signal corresponds to the ℓ^2-norm of the signal's expansion coefficients in the sinc basis.

It turns out that there is a stark difference if we instead measure the "size" of a function by the more restrictive ℓ^1-norm instead of the ℓ^2-norm, an idea popularized by wavelets and compressed sensing. Let $\mathcal{D} = \{\psi\}_{\psi \in \mathcal{D}}$ be a dictionary of atoms (e.g., sinc functions, wavelets, neurons, and so on). Consider the problem of approximating a multivariate function mapping $\mathbb{R}^d \to \mathbb{R}$ that admits a decomposition $f(\boldsymbol{x}) = \sum_{k=1}^{\infty} v_k \psi_k(\boldsymbol{x})$, where $\psi_k \in \mathcal{D}$ and the expansion coefficients satisfy $\sum_{k=1}^{\infty} |v_k| = \|\boldsymbol{v}\|_{\ell^1} < \infty$. It turns out that there exists an approximant constructed with K terms from the dictionary $\mathcal{D}$ whose L^2-approximation error $\|f - f_K\|_{L^2}$ decays at a rate completely independent of the input dimension d.

Here, we illustrate the argument when $\mathcal{D} = \{\psi_k\}_{k=1}^{\infty}$ is an orthonormal basis (e.g., multivariate Haar wavelets). Given a function $f : \mathbb{R}^d \to \mathbb{R}$ that admits a decomposition $f(\boldsymbol{x}) = \sum_{k=1}^{\infty} v_k \psi_k(\boldsymbol{x})$ such that $\|\boldsymbol{v}\|_{\ell^1} < \infty$, we can construct an approximant f_K by a simple thresholding procedure that keeps the K largest coefficients of f and sets all other coefficients to zero. If we let $|v_{(1)}| \geq |v_{(2)}| \geq \cdots$ denote the coefficients of f sorted in nonincreasing magnitude, then the squared approximation error is bounded as

$$\|f - f_K\|_{L^2}^2 = \left\| \sum_{k>K} v_{(k)} \psi_{(k)} \right\|_{L^2}^2 = \sum_{k>K} |v_{(k)}|^2 \tag{20}$$

Table 1. Common activation functions.

Activation Function	$\sigma(t)$	Frequency Response of Sparsifying Transform: $\hat{H}(\omega)$
ReLU	$\max\{0, t\}$	$-\omega^2$
Truncated power	$\dfrac{\max\{0,t\}^k}{k!}, k \in \mathbb{N}$	$(\mathrm{j}\omega)^{k+1}$
Sigmoid	$\dfrac{1}{1+e^{-t}} - \dfrac{1}{2}$	$\dfrac{\mathrm{j}}{\pi}\sinh(\pi\omega)$
arctan	$\dfrac{\arctan(t)}{\pi}$	$\mathrm{j}\omega e^{\lvert\omega\rvert}$
Exponential	$\dfrac{e^{-\lvert t\rvert}}{2}$	$1+\omega^2$

where the last equality follows by exploiting the orthonormality of the $\{\psi_k\}_{k=1}^{\infty}$. Finally, as the original sequence of coefficients $\boldsymbol{v} = (v_1, v_2, \ldots)$ is absolutely summable, $|v_{(k)}|$ must decay strictly faster than $1/k$ for $k > K$ (because the tail of the harmonic series $\Sigma_{k>K} \frac{1}{k}$ diverges). Putting this together with (20), the L^2-approximation error $\| f - f_K \|_{L^2}$ must decay as $K^{-1/2}$, completely independent of the input dimension d. For a more precise treatment of this argument, we refer the reader to [22, Th. 9.10]. These kinds of thresholding procedures, particularly with wavelet bases [12], revolutionized signal and image processing and were the foundations of compressed sensing [7], [11].

By a more sophisticated argument, a similar phenomenon occurs when the orthonormal basis is replaced with an essentially arbitrary dictionary of atoms. The result for general atoms is based on a probabilistic technique presented by Pisier in 1981 at the Functional Analysis Seminar at École Polytechnique, Palaiseau, France, crediting the idea to Maurey [33]. An implication of Maurey's technique is that, given a function that is an ℓ^1 combination of bounded atoms from a dictionary, there exists a K-term approximant that admits a dimension-free approximation rate that decays as $K^{-1/2}$. Motivated by discussions with Jones [18] on his work on greedy approximation, which provides a deterministic algorithm to find the approximant that admits the dimension-free rate, Barron used the technique of Maurey to prove his dimension-free approximation rates with sigmoidal NNs. This abstract approximation result is now referred to as the *Maurey–Jones–Barron lemma*.

The Maurey–Jones–Barron lemma can be applied to any function space where the functions are ℓ^1 combinations of bounded atoms.

In particular, the Maurey–Jones–Barron lemma can be applied to any function space where the functions are ℓ^1 combinations of bounded atoms. Such spaces are sometimes called *variation spaces* [2], [20]. Recall from the "What Kinds of Functions Do NNs Learn?" and "What Is the Role of NN Activation Functions" sections that the $\mathrm{H}\,\mathrm{K}\,\mathscr{R}$ operator sparsifies neurons of the form $\sigma(\boldsymbol{w}^\top \boldsymbol{x} - b)$, where $(\boldsymbol{w}, b) \in \mathbb{S}^{d-1} \times \mathbb{R}$ and σ are matched to H. This implies that the space of functions $f : \mathbb{R}^d \to \mathbb{R}$ such that $\| \mathrm{H}\,\mathrm{K}\,\mathscr{R} f \|_{\mathcal{M}} < \infty$ can be viewed as a variation space, where the dictionary corresponds to the neurons $\{\sigma(\boldsymbol{w}^\top \boldsymbol{x} - b)\}_{(\boldsymbol{w},b) \in \mathbb{S}^{d-1} \times \mathbb{R}}$. Therefore, given $f : \mathbb{R}^d \to \mathbb{R}$ such that $\| \mathrm{H}\,\mathrm{K}\,\mathscr{R} f \|_{\mathcal{M}} < \infty$, there exists a K-term approximant f_K that takes the form of a shallow NN with K neurons such that the L^2-approximation error decays as $K^{-1/2}$. These techniques have been studied and extended in great detail [40] and have been extended to the setting of DNNs [13] by considering compositional function spaces akin to the compositional space introduced in the "DNNs" section.

Combining these dimension-free approximation rates with the sparsity-promoting effect of weight decay regularization for ReLU NNs has a striking effect on the learning problem. Suppose that we train a shallow ReLU NN with weight decay on data generated from the noisy samples $y_n = f(\boldsymbol{x}_n) + \varepsilon_n, n = 1, \ldots, N$, of $f \in \mathscr{R}\mathrm{BV}^2(\mathbb{R}^d)$, where $\boldsymbol{x}_n$ are independent identically distributed (i.i.d.) uniform random variables on some bounded domain $\Omega \subset \mathbb{R}^d$, and ε_n are i.i.d. Gaussian random variables. Let f_N denote this trained NN. Then, it has been shown [31] that the mean integrated squared error (MISE) $\mathbb{E} \| f - f_N \|_{L^2(\Omega)}^2$ decays at a rate of $N^{-1/2}$, independent of the input dimension d. Moreover, this result also shows that the generalization error of the trained NN on a new example $\boldsymbol{x}$ generated uniformly at random on Ω is also immune to the curse of dimensionality. Furthermore, these ideas have been studied in the context of DNNs [38], proving dimension-free MISE rates for estimating Hölder functions (that exhibit low-dimensional structure) with ReLU DNNs.

Mixed variation and low-dimensional structure

The national meeting of the American Mathematical Society in 2000 was held to discuss the mathematical challenges of the 21st century. Here, Donoho [10] gave a lecture titled "High-Dimensional Data Analysis: The Curses and Blessings of Dimensionality." In this lecture, he coined the term *mixed variation* to refer to the kinds of functions that lie in variation spaces, citing the spectral Barron spaces as an example. Variation spaces are different from classical multivariate function spaces in that they favor functions that have weak variation in multiple directions (very smooth functions) as well as functions that have very strong variation in one or a few directions (very rough functions). These spaces also disfavor functions with strong variation in multiple directions. It is this fact that makes them quite "small" compared to classical multivariate function spaces, giving rise to their dimension-free approximation and MISE rates. Examples of functions with different kinds of variation are illustrated in Figure 5. The prototypical examples of functions that lie in mixed variation spaces can be thought of as superpositions of few neurons with different directions or superpositions of many neurons (even continuously many) in only a few directions.

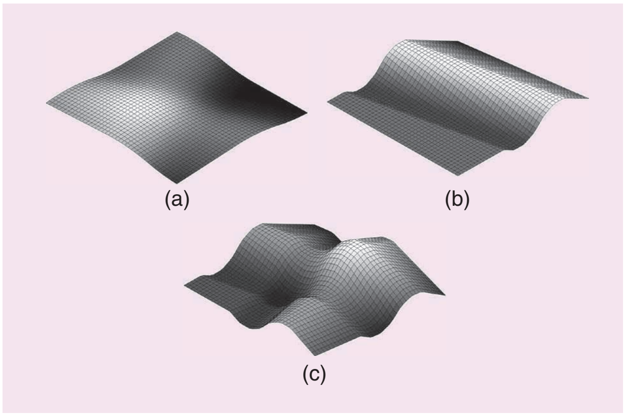


FIGURE 5. Examples of functions exhibiting different kinds of variation. (a) Weak variation in multiple directions. (b) Strong variation in one direction. (c) Strong variation in multiple directions.

To interpret the idea of mixed variation in the context of modern data analysis and DL, we turn our attention to Figure 5(b). In this figure, the function has strong variation, but only in a single direction. In other words, this function has a low-dimensional structure. It has been observed by a number of authors that DNNs are able to automatically adapt to the low-dimensional structure that often arises in natural data. This is possible because the input weights can be trained to adjust orientation of each neuron. The dimension-independent approximation rate quantifies the power of this tunability. This explains why DNNs are good at learning functions with a low-dimensional structure. In particular, the function in Figure 5(c) has strong variation in all directions, so no method can overcome the curse of dimensionality in this sort of situation. On the other hand, in Figure 5(a), the function has weak variation in all directions, and Figure 5(b) has strong variation in only one direction, so these are functions for which NNs will overcome the curse. For Figure 5(b), the sparsity-promoting effect of weight decay promotes DNN solutions with neurons oriented in the direction of variation (i.e., it automatically learns the low-dimensional structure).

A better understanding of compositional function spaces could provide new insights into the benefits of depth in NNs.

Takeaway messages and future research directions

In this article, we presented a mathematical framework to understand DNNs from first principles, through the lens of sparsity and sparse regularization. Using familiar mathematical tools from signal processing, we provided an explanation for the sparsity-promoting effect of the common regularization scheme of weight decay in NN training, the use of skip connections and low-rank weight matrices in network architectures, and why NNs seemingly break the curse of dimensionality. This framework provides the mathematical setting for many future research directions.

The framework suggests the possibility of new neural training algorithms. The equivalence of solutions using weight decay regularization and the regularization in (6) leads to the use of proximal gradient methods akin to iterative soft-thresholding algorithms to train DNNs. This avenue has already begun to be explored. The preliminary results in [47] have shown that proximal gradient training algorithms for DNNs perform as well as and often better (particularly when labels are corrupted) than standard gradient-based training with weight decay, while simultaneously producing sparser networks.

The large body of work dating back to 1989 [21] on NN pruning has shown empirically that large NNs can be compressed or sparsified to a fraction of their size while still maintaining their predictive performance. The connection between weight decay and sparsity-promoting regularizers, like in (6), suggests new approaches to pruning. For example, one could apply proximal gradient algorithms to derive sparse approximations to large pretrained NNs [39]. There are many open questions in this direction, both experimental and theoretical, including applying these algorithms to other DNN architectures and deriving convergence results for these algorithms.

The framework in this article also shows that trained ReLU DNNs are compositions of $\mathscr{R}\mathrm{BV}^2$ functions. As we have seen in this article, at this point in time, we have a clear and nearly complete understanding of $\mathscr{R}\mathrm{BV}^2(\mathbb{R}^d)$. In particular, the $\mathscr{R}\mathrm{BV}^2$ space favors functions that are smooth in most or all directions, which explains why NNs seemingly break the curse of dimensionality. Less is clear and understood about the compositions of $\mathscr{R}\mathrm{BV}^2$ functions (which characterize DNNs). A better understanding of compositional function spaces could provide new insights into the benefits of depth in NNs. This in turn could lead to new guidelines for designing NN architectures and training algorithms.

Acknowledgment

The authors would like to thank Rich Baraniuk, Misha Belkin, Çağ̆atay Candan, Ron DeVore, Kangwook Lee, Greg Ongie, Dimitris Papailiopoulos, Tomaso Poggio, Lorenzo Rosasco, Joe Shenouda, Jonathan Siegel, Ryan Tibshirani, Michael Unser, Becca Willett, Stephen Wright, Liu Yang, and Jifan Zhang for many insightful discussions on the topics presented in this article.

Rahul Parhi was supported in part by the National Science Foundation (NSF) Graduate Research Fellowship Program under Grant DGE-1747503 and the European Research Council's Project FunLearn under Grant 101020573. Robert Nowak was supported in part by NSF Grants DMS-2134140 and DMS-2023239, U.S. Office of Naval Research MURI Grant N00014-20-1-2787, Air Force Office of Scientific Research/Air Force Research Laboratory Grant FA9550-18-1-0166, and the Keith and Jane Nosbusch Professorship.

Authors

Rahul Parhi (rahul.parhi@epfl.ch) received his B.S. degree in mathematics and his B.S. degree in computer science from the University of Minnesota, Twin Cities in 2018, and his M.S. and Ph.D. degrees in electrical engineering from the University of Wisconsin–Madison in 2019 and 2022, respectively. During his Ph.D. studies, he was supported by a National Science Foundation Graduate Research Fellowship. Currently, he is a postdoctoral researcher with the Biomedical Imaging Group, École Polytechnique Fédérale de Lausanne, 1015 Lausanne, Switzerland. His research interests include applications of functional and harmonic analysis to problems in signal processing and data science. He is a Member of IEEE.

Robert D. Nowak (rdnowak@wisc.edu) received his Ph.D. degree in electrical engineering from the University of Wisconsin–Madison, Madison, WI 53706 USA, where he is the Grace Wahba Professor of Data Science and Keith and

Jane Morgan Nosbusch Professor of Electrical and Computer Engineering. He serves as a section editor of *SIAM Journal on Mathematics of Data Science* and a senior editor of *IEEE Journal on Selected Areas in Information Theory*. With research focusing on signal processing, machine learning, optimization, and statistics, his work on sparse signal recovery and compressed sensing has won several awards. He is a Fellow of IEEE.

References

[1] L. J. Ba and R. Caruana, "Do deep nets really need to be deep?" in *Proc. 27th Int. Conf. Neural Inf. Process. Syst.*, 2014, vol. 2, pp. 2654–2662.

[2] F. Bach, "Breaking the curse of dimensionality with convex neural networks," *J. Mach. Learn. Res.*, vol. 18, no. 1, pp. 629–681, Jan. 2017.

[3] R. Balestriero and R. G. Baraniuk, "Mad max: Affine spline insights into deep learning," *Proc. IEEE*, vol. 109, no. 5, pp. 704–727, May 2021, doi: 10.1109/JPROC.2020.3042100.

[4] A. R. Barron, "Universal approximation bounds for superpositions of a sigmoidal function," *IEEE Trans. Inf. Theory*, vol. 39, no. 3, pp. 930–945, May 1993, doi: 10.1109/18.256500.

[5] F. Bartolucci, E. De Vito, L. Rosasco, and S. Vigogna, "Understanding neural networks with reproducing kernel Banach spaces," *Appl. Comput. Harmon. Anal.*, vol. 62, pp. 194–236, Jan. 2023, doi: 10.1016/j.acha.2022.08.006.

[6] E. J. Candès, "Ridgelets: Theory and applications," Ph.D. dissertation, Stanford Univ., Stanford, CA, USA, 1998.

[7] E. J. Candès, J. Romberg, and T. Tao, "Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information," *IEEE Trans. Inf. Theory*, vol. 52, no. 2, pp. 489–509, Feb. 2006, doi: 10.1109/TIT.2005.862083.

[8] L. Chizat and F. Bach, "Implicit bias of gradient descent for wide two-layer neural networks trained with the logistic loss," in *Proc. 33rd Conf. Learn. Theory (PMLR)*, 2020, pp. 1305–1338.

[9] T. Debarre, Q. Denoyelle, M. Unser, and J. Fageot, "Sparsest piecewise-linear regression of one-dimensional data," *J. Comput. Appl. Math.*, vol. 406, May 2022, Art. no. 114044, doi: 10.1016/j.cam.2021.114044.

[10] D. L. Donoho, "High-dimensional data analysis: The curses and blessings of dimensionality," *AMS Math Challenges Lectures*, vol. 1, p. 32, Aug. 2000.

[11] D. L. Donoho, "Compressed sensing," *IEEE Trans. Inf. Theory*, vol. 52, no. 4, pp. 1289–1306, Apr. 2006, doi: 10.1109/TIT.2006.871582.

[12] D. L. Donoho and I. M. Johnstone, "Minimax estimation via wavelet shrinkage," *Ann. Statist.*, vol. 26, no. 3, pp. 879–921, Jun. 1998, doi: 10.1214/aos/1024691081.

[13] E. Weinan and S. Wojtowytsch, "On the Banach spaces associated with multi-layer ReLU networks: Function representation, approximation theory and gradient descent dynamics," *CSIAM Trans. Appl. Math.*, vol. 1, no. 3, pp. 387–440, 2020, doi: 10.4208/csiam-am.20-211.

[14] S. D. Fisher and J. W. Jerome, "Spline solutions to L^1 extremal problems in one and several variables," *J. Approximation Theory*, vol. 13, no. 1, pp. 73–83, Jan. 1975, doi: 10.1016/0021-9045(75)90016-7.

[15] A. Golubeva, B. Neyshabur, and G. Gur-Ari, "Are wider nets better given the same number of parameters?" in *Proc. Int. Conf. Learn. Representations*, 2021.

[16] Y. Grandvalet, "Least absolute shrinkage is equivalent to quadratic penalization," in *Proc. Int. Conf. Artif. Neural Netw.*, London, U.K.: Springer-Verlag, 1998, pp. 201–206, doi: 10.1007/978-1-4471-1599-1_27.

[17] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit. (CVPR)*, 2016, pp. 770–778.

[18] L. K. Jones, "A simple lemma on greedy approximation in Hilbert space and convergence rates for projection pursuit regression and neural network training," *Ann. Statist.*, vol. 20, no. 1, pp. 608–613, Mar. 1992, doi: 10.1214/aos/1176348546.

[19] V. Kůrková, P. C. Kainen, and V. Kreinovich, "Estimates of the number of hidden units and variation with respect to half-spaces," *Neural Netw.*, vol. 10, no. 6, pp. 1061–1068, Aug. 1997, doi: 10.1016/S0893-6080(97)00028-2.

[20] V. Kůrková and M. Sanguineti, "Bounds on rates of variable-basis and neural-network approximation," *IEEE Trans. Inf. Theory*, vol. 47, no. 6, pp. 2659–2665, Sep. 2001, doi: 10.1109/18.945285.

[21] Y. LeCun, J. Denker, and S. Solla, "Optimal brain damage," in *Proc. 2nd Int. Conf. Neural Inf. Process. Syst. (NIPS)*, 1989, pp. 598–605.

[22] S. Mallat, *A Wavelet Tour of Signal Processing*, 3rd ed. Amsterdam, The Netherlands: Elsevier/Academic Press, 2009.

[23] E. Mammen and S. van de Geer, "Locally adaptive regression splines," *Ann. Statist.*, vol. 25, no. 1, pp. 387–413, Feb. 1997, doi: 10.1214/aos/1034276635.

[24] W. S. McCulloch and W. Pitts, "A logical calculus of the ideas immanent in nervous activity," *Bull. Math. Biophys.*, vol. 5, no. 4, pp. 115–133, Dec. 1943, doi: 10.1007/BF02478259.

[25] B. Neyshabur, R. Tomioka, and N. Srebro, "In search of the real inductive bias: On the role of implicit regularization in deep learning," in *Proc. Int. Conf. Learn. Representations (Workshop)*, 2015.

[26] G. Ongie, R. Willett, D. Soudry, and N. Srebro, "A function space view of bounded norm infinite width ReLU nets: The multivariate case," in *Proc. Int. Conf. Learn. Representations*, 2020.

[27] R. Parhi, "On ridge splines, neural networks, and variational problems in Radon-domain BV spaces," Ph.D. dissertation, The Univ. of Wisconsin–Madison, Madison, WI, USA, 2022.

[28] R. Parhi and R. D. Nowak, "The role of neural network activation functions," *IEEE Signal Process. Lett.*, vol. 27, pp. 1779–1783, Sep. 2020, doi: 10.1109/LSP.2020.3027517.

[29] R. Parhi and R. D. Nowak, "Banach space representer theorems for neural networks and ridge splines," *J. Mach. Learn. Res.*, vol. 22, no. 1, pp. 1960–1999, Jan. 2021.

[30] R. Parhi and R. D. Nowak, "What kinds of functions do deep neural networks learn? Insights from variational spline theory," *SIAM J. Math. Data Sci.*, vol. 4, no. 2, pp. 464–489, 2022, doi: 10.1137/21M1418642.

[31] R. Parhi and R. D. Nowak, "Near-minimax optimal estimation with shallow ReLU neural networks," *IEEE Trans. Inf. Theory*, vol. 69, no. 2, pp. 1125–1140, Feb. 2023, doi: 10.1109/TIT.2022.3208653.

[32] M. Pilanci and T. Ergen, "Neural networks are convex regularizers: Exact polynomial-time convex optimization formulations for two-layer networks," in *Proc. 37th Int. Conf. Mach. Learn. (PMLR)*, 2020, pp. 7695–7705.

[33] G. Pisier, "Remarques sur un résultat non publié de B. Maurey," in *Proc. Séminaire d'Anal. Fonctionnelle (dit "Maurey-Schwartz")*, Apr. 1981, pp. 1–12.

[34] T. Poggio, A. Banburski, and Q. Liao, "Theoretical issues in deep networks," *Proc. Nat. Acad. Sci. USA*, vol. 117, no. 48, pp. 30,039–30,045, Dec. 2020, doi: 10.1073/pnas.1907369117.

[35] L. I. Rudin, S. Osher, and E. Fatemi, "Nonlinear total variation based noise removal algorithms," *Phys. D, Nonlinear Phenomena*, vol. 60, nos. 1–4, pp. 259–268, Nov. 1992, doi: 10.1016/0167-2789(92)90242-F.

[36] A. Sanyal, P. H. Torr, and P. K. Dokania, "Stable rank normalization for improved generalization in neural networks and GANs," in *Proc. Int. Conf. Learn. Representations*, 2019.

[37] P. Savarese, I. Evron, D. Soudry, and N. Srebro, "How do infinite width bounded norm networks look in function space?" in *Proc. 32nd Conf. Learn. Theory (PMLR)*, 2019, pp. 2667–2690.

[38] J. Schmidt-Hieber, "Nonparametric regression using deep neural networks with ReLU activation function," *Ann. Statist.*, vol. 48, no. 4, pp. 1875–1897, Aug. 2020, doi: 10.1214/19-AOS1875.

[39] J. Shenouda, R. Parhi, K. Lee, and R. D. Nowak, "Vector-valued variation spaces and width bounds for DNNs: Insights on weight decay regularization," 2023, *arXiv:2305.16534*.

[40] J. W. Siegel and J. Xu, "Sharp bounds on the approximation rates, metric entropy, and *n*-widths of shallow neural networks," *Found. Comput. Math.*, early access, 2022, doi: 10.1007/s10208-022-09595-3.

[41] R. Tibshirani, "Regression shrinkage and selection via the lasso," *J. Roy. Statist. Soc., Ser. B (Methodological)*, vol. 58, no. 1, pp. 267–288, Jan. 1996, doi: 10.1111/j.2517-6161.1996.tb02080.x.

[42] M. Unser, "From kernel methods to neural networks: A unifying variational formulation," 2022, *arXiv:2206.14625*.

[43] M. Unser, "Ridges, neural networks, and the Radon transform," *J. Mach. Learn. Res.*, vol. 24, no. 37, pp. 1–33, 2023.

[44] M. Unser, J. Fageot, and J. P. Ward, "Splines are universal solutions of linear inverse problems with generalized TV regularization," *SIAM Rev.*, vol. 59, no. 4, pp. 769–793, 2017, doi: 10.1137/16M1061199.

[45] M. Vetterli, P. Marziliano, and T. Blu, "Sampling signals with finite rate of innovation," *IEEE Trans. Signal Process.*, vol. 50, no. 6, pp. 1417–1428, Jun. 2002, doi: 10.1109/TSP.2002.1003065.

[46] H. Wang, S. Agarwal, and D. Papailiopoulos, "Pufferfish: Communication-efficient models at no extra cost," in *Proc. 3rd Mach. Learn. Syst.*, 2021, pp. 365–386.

[47] L. Yang, J. Zhang, J. Shenouda, D. Papailiopoulos, K. Lee, and R. D. Nowak, "A better way to decay: Proximal gradient training algorithms for neural nets," in *Proc. 14th Annu. Workshop Optim. Mach. Learn. (OPT)*, 2022.

[48] M. Yuan and Y. Lin, "Model selection and estimation in regression with grouped variables," *J. Roy. Statist. Soc., Ser. B (Statist. Methodology)*, vol. 68, no. 1, pp. 49–67, Feb. 2006, doi: 10.1111/j.1467-9868.2005.00532.x.

SP