ReLUs Are Sufficient for Learning Implicit Neural Representations

Joseph Shenouda [1] **Yamin Zhou** [2] **Robert D. Nowak** [1]

Abstract

Motivated by the growing theoretical understanding of neural networks that employ the Rectified Linear Unit (ReLU) as their activation function, we revisit the use of ReLU activation functions for learning implicit neural representations (INRs). Inspired by second order B-spline wavelets, we incorporate a set of simple constraints to the ReLU neurons in each layer of a deep neural network (DNN) to remedy the spectral bias. This in turn enables its use for various INR tasks. Empirically, we demonstrate that, contrary to popular belief, one *can learn* state-of-the-art INRs based on a DNN composed of only ReLU neurons. Next, by leveraging recent theoretical works which characterize the kinds of functions ReLU neural networks learn, we provide a way to quantify the regularity of the learned function. This offers a principled approach to selecting the hyperparameters in INR architectures. We substantiate our claims through experiments in signal representation, super resolution, and computed tomography, demonstrating the versatility and effectiveness of our method. The code for all experiments can be found at `https://github.com/joeshenouda/relu-inrs`.

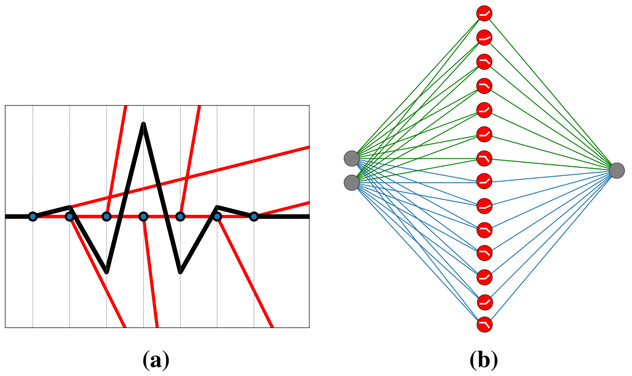


Figure 1. (Left) A plot of the second order B-spline wavelet activation function. The red lines indicate the seven non-local ReLU functions that make up a single second order B-spline wavelet, shown in black. (Right) A BW-ReLU neural network with two neurons represented as a constrained ReLU network with 14 neurons. Within each group the orientation of each ReLU relative to the others is fixed. The input and output weights are learned and shared across groups of neurons. Shared input/output weights are denoted by the same color.

1. Introduction

Recently, training deep neural networks (DNNs) to learn implicit neural representations (INRs) has led to advancements in various vision-related tasks. These include, but are not limited to, computer graphics (Mildenhall et al., 2021), image processing (Chen et al., 2021), and signal representation (Sitzmann et al., 2020). They have also shown great promise in biomedical imaging, where they can be used for sparse-view computed tomography (CT) (Sun et al., 2021; Wu et al., 2023). Many INR tasks involve learning continuous representations of images, unlike image classification tasks which train DNNs on high-dimensional data. INRs learn a continuous representation of an image by training a DNN on the low-dimensional coordinates of the image. For such imaging tasks, the success of the INR hinges on the ability of the DNN to efficiently approximate and learn high-frequency components of the image. This has, thus far, prohibited the use of DNNs with ReLU activations as they have been shown to exhibit a *spectral bias* (Rahaman et al., 2019)—an inherent bias of ReLU DNNs which causes them to struggle in approximating high-frequency functions when trained via gradient descent.

Therefore, in order to circumvent this problem while still utilizing the power of neural networks, practitioners in the INR community have adopted preprocessing techniques (Mildenhall et al., 2021; Tancik et al., 2020) and many non-standard activation functions. These include, but are not limited to, the sine function (Sitzmann et al., 2020), the Gaussian function (Ramasinghe & Lucey, 2022), or the complex Gabor wavelet (Saragadam et al., 2023). However, using these highly unorthodox activation functions has raised many questions about the theoretical properties of

[1]Department of Electrical and Computer Engineering, University of Wisconsin-Madison [2]Department of Computer Sciences, University of Wisconsin-Madison. Correspondence to: Joseph Shenouda <jshenouda@wisc.edu>.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024. Copyright 2024 by the author(s).

INRs (Yüce et al., 2022). Therefore, motivated by the fact that a majority of our theoretical understanding of neural networks is based on standard ReLU DNNs, we revisit the use of ReLU activations for learning INRs.

Consider a shallow ReLU neural network of the form

$$f_{\boldsymbol{\theta}}(\boldsymbol{x}) = \sum_{k=1}^{K} v_k \sigma(\boldsymbol{w}_k^T \boldsymbol{x} - b_k) \tag{1}$$

where $\sigma(z) = \max\{0, z\}$. The neural network function is a linear combination of a set of "atomic functions"-in this case the ReLU neurons $\{\sigma(\boldsymbol{w}_k^T \boldsymbol{x} - b_k)\}_{k=1}^{K}$. The monotonically increasing nature of the ReLU activation function implies that each ReLU neuron contributes globally to the function and is highly coherent with the others. When attempting to fit this network to data, if the weights of one neuron changes, all the other neurons in the layer must also adjust to correct for the influence of that one neuron. This makes the function highly sensitive to even small changes in the parameters, and results in a severely ill-conditioned optimization problem. This ill-conditioning mandates an inordinate number of iterations when training the network using gradient-type optimization methods, even when using modern optimizers such as Adam. This problem is exacerbated in INR tasks where the data is densely sampled, low dimensional and exhibits high frequencies.

Guided by this simple insight, our investigation takes a different approach than previous works and focuses solely on remedying the optimization problem. This is in contrast to wholly replacing the activation functions of the neurons which generate our features. Specifically, we examine whether a localized ReLU-based activation function can effectively address and overcome the ill-conditioning. To do this, we propose incorporating constraints to groups of ReLU neurons within each of the hidden layers in a ReLU DNN. For each group, all of the neurons share the same weights. Our constraints ensure that every seven ReLU neurons in the hidden layer is effectively applying the activation function $\psi(x) : \mathbb{R} \to \mathbb{R}$ where,

$$\begin{aligned}\psi(x) = &\frac{1}{6}(\sigma(x) + \sigma(x-3)) - \frac{16}{3}\sigma\left(x - 1.5\right) \\ &- \frac{8}{6}\left(\sigma\left(x - 0.5\right) + \sigma\left(x - 2.5\right)\right) \\ &+ \frac{23}{6}\left(\sigma(x-1) + \sigma(x-2)\right).\end{aligned} \tag{2}$$

This activation function corresponds to a second order B-spline wavelet introduced in Chui & Wang (1992); Unser et al. (1993). A plot of the second order B-spline wavelet in the univariate case is shown in Figure 1a. For ease of exposition we will refer to neural networks that use this activation function simply as BW-ReLU neural networks.

Thus, when training a BW-ReLU neural network of the form

$$g_{\boldsymbol{\theta}}(\boldsymbol{x}) = \sum_{k=1}^{K} v_k \psi(\boldsymbol{w}_k^T \boldsymbol{x} - b_k), \tag{3}$$

we are still learning a function which can be exactly represented by a ReLU neural network with $7K$ neurons as demonstrated in Figure 1b. At an intuitive level, the compact nature of these neurons allows for each neuron to fit different parts of the function without affecting the contributions from other neurons. In Section 3 we provide a more thorough discussion on the ill-conditioning and how the unique properties of the BW-ReLU can remedy it. Our experiments in Section 5 demonstrate the effectiveness of this approach on various INR tasks.

Next, having developed a method for learning ReLU-based INRs, we leverage the recently developed theory characterizing the kinds of functions ReLU neural networks learn when fit to data (Savarese et al., 2019; Ongie et al., 2020; Parhi & Nowak, 2021; Shenouda et al., 2023). We show how our BW-ReLU neural network functions fit into this mathematical framework and how we can measure the *variation norm* of these functions. This provides a measure of the regularity of the learned function. We also give new insights into how this regularity is effected when one reparameterizes the INR with a scaling parameter $c > 0$ such that the neurons are defined as $v \cdot \psi(c \cdot (\boldsymbol{w}^T \boldsymbol{x} - b))$. This heuristic is employed in all of the activation functions introduced for INRs despite being poorly understood (Sitzmann et al., 2020; Ramasinghe & Lucey, 2022; Saragadam et al., 2023). Moreover, we show how the variation norm of the BW-ReLU neural network provides a good indication for how well the network can generalize to unseen data. This suggests a principled way to tune INRs without the need for an additional validation dataset. In summary, our contributions are:

1. **A method for learning ReLU-based INRs**: By incorporating a simple set of constraints on the neurons of a ReLU neural network, we can overcome the ill-conditioning inherent to ReLU networks. We demonstrate that this approach can be used in multiple INR tasks and performs comparably to other INR architectures that use unconventional activation functions.

2. **Insights on INR generalization**: We present a way to measure the regularity of BW-ReLU neural networks by leveraging recent theoretical results on the kinds of functions ReLU neural networks learn. This regularity is measured in terms of the variation norm. We discuss how this perspective provides insights into some of the heuristics employed in learning INRs and how BW-ReLU neural networks with a lower variation norm tend to generalize better.

2. Related Works

The ReLU is perhaps the simplest activation function utilized in DNNs. Thus much of the recent DNN theory has focused on this setting (Bach, 2017; Savarese et al., 2019; Arora et al., 2018; Ongie et al., 2020; Parhi & Nowak, 2021). They are also useful in practice as they induce sparse activation which can be exploited for compression or speeding up inference (Li et al., 2023; Kurtz et al., 2020; Mirzadeh et al., 2024).

However, neural networks with ReLU activations are typically not utilized for INR tasks due to their *spectral bias* (Rahaman et al., 2019). To remedy this, pre-processing techniques (Tancik et al., 2020; Mildenhall et al., 2021) and unconventional activation functions (Sitzmann et al., 2020; Ramasinghe & Lucey, 2022; Saragadam et al., 2023) have been used alongside or instead of traditional ReLU DNNs. Our results brings into question the necessity of these unorthodox approaches by showing that a ReLU-based DNN can be trained for various INR tasks.

Our approach is inspired by B-spline wavelets which were first developed and studied in (Chui & Wang, 1992; Unser et al., 1993; 1992). Moreover, our BW-ReLU neural networks are very related to the concept of ridgelets which were originally developed and studied in (Candes, 1998; Candès & Donoho, 1999).

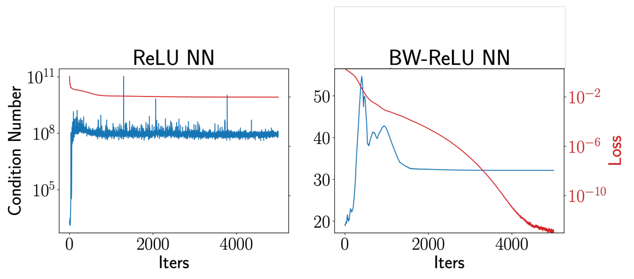


Figure 2. Condition number of feature embedding matrix generated by ReLU vs. BW-ReLU neural networks during training. We see that the ReLU produces a severely ill-conditioned feature matrix at initialization and throughout training. In contrast, the BW-ReLU neural network enjoys a very well conditioned feature matrix all throughout training. The rate of convergence is also correlated with how well conditioned the feature matrix in both cases.

3. Remedying ReLU Neural Networks

In this section, we provide a more thorough discussion on why ReLU neural networks tend to exhibit a spectral bias when trained to fit low-dimensional datasets. We formalize the intuitions presented in the introduction and explain how the strong coherence between ReLU neurons results in an ill-conditioned optimization problem, which significantly slows down convergence. We also discuss key properties of the second-order B-spline wavelet, which remedy this ill-conditioning (and consequently the spectral bias) making them suitable for INR tasks.

Consider approximating a univariate function $u : D \to \mathbb{R}$ by a univariate ReLU neural network $f : D \to \mathbb{R}$ of the form

$$f(x) = \sum_{k=1}^{K} v_k \sigma(w_k x - b_k). \tag{4}$$

Where $x \in D, \quad w_k \in \{-1, 1\}$ and $v_k, b_k, c \in \mathbb{R}$. Here $D = [r_1, r_2]$ is a bounded domain, K denotes the width of the network and $\sigma : \mathbb{R} \to \mathbb{R}$ denotes the ReLU activation function defined as $\sigma(\cdot) = \max\{0, \cdot\}$. The restriction of the input weights w_k to $\{-1, 1\}$ follows from the homogeneity of the ReLU (i.e., for any $\alpha > 0$ we have that $\sigma(\alpha z) = \alpha\sigma(z)$). Now due to the fact that $\sigma(z) = z - \sigma(-z)$ we can further restrict the input weights to be $+1$ by introducing a skip connection

$$f(x) = c + ax + \sum_{k=1}^{K} v_k \sigma(x - b_k) \tag{5}$$

where $\quad x \in D$ and $v_k, b_k, c, a \in \mathbb{R}$. By only considering the approximation on the domain D, setting $c = u(r_1)$ and $b_k \in [-1, 1]$ the network can be further reduced to

$$f_{\boldsymbol{\theta}}(x) = c + \sum_{k=1}^{K} v_k \sigma(x - b_k) \tag{6}$$

where $\boldsymbol{\theta} = (v_k, b_k)_{k=1}^K$ denotes the parameters of the network.

Now suppose we train the neural network to approximate u by sampling the function and training the network on a univariate dataset $(x_i, y_i)_{i=1}^N$ to minimizing the ℓ_2 loss. This corresponds to solving the following optimization problem

$$\min_{\boldsymbol{\theta}} \|\boldsymbol{\Phi}^T \boldsymbol{v} - \boldsymbol{y}\|_2^2 = \min_{\boldsymbol{\theta}} \boldsymbol{v}^T \boldsymbol{\Phi}\boldsymbol{\Phi}^T \boldsymbol{v} - 2(\boldsymbol{\Phi}\boldsymbol{y})^T \boldsymbol{v}, \tag{7}$$

where $\boldsymbol{y} \in \mathbb{R}^N$ is a vector containing the labels for all N samples and $\boldsymbol{v} \in \mathbb{R}^K$ is a vector containing the output weight of each neuron. Moreover, $\boldsymbol{\Phi} \in \mathbb{R}^{K \times N}$ is the feature embedding matrix, which depends on the input biases and is learned throughout training of the network,

$$\boldsymbol{\Phi}_i = \left[\sigma(x_1 - b_i), \cdots, \sigma(x_N - b_i)\right]. \tag{8}$$

Neural networks typically solve (7) via gradient descent. If we consider the gradient step update on just the output weights $\boldsymbol{v}$ then it is clear from (7) that this is equivalent to taking a gradient descent step on the least squares problem over a fixed set of features $\boldsymbol{\Phi}$. The effectiveness of each gradient step to minimize the objective is dependent on the condition number of the Hessian, $\mathbf{\Phi}\mathbf{\Phi}^T$(Boyd & Vandenberghe, 2004). For a real symmetric matrix $A \in \mathbb{R}^{m\times m}$ the condition number is defined as $\kappa(A) = \frac{\lambda_1}{\lambda_m}$ where λ_1 and λ_m are the largest and smallest eigenvalues respectively.

In the context of least squares, if the matrix $\mathbf{\Phi}\mathbf{\Phi}^T$ has a very high condition number then the curvature of the loss landscape (with respect to the output weights $\boldsymbol{v}$) will vary dramatically in different directions. This hinders the effectiveness of each gradient step and results in requiring many more gradient steps to converge.

When using ReLU activations, the feature embedding matrix $\mathbf{\Phi}\mathbf{\Phi}^T$ is typically severely ill-conditioned which results in a slow converge. This ill-conditioning of the features is present at initialization and persists throughout training, see for example Figure 2. To understand why, consider two ReLUs with biases that are close to each other. In this case they are nearly colinear (causing ill-conditioning). On the other hand, if all the neurons are orthogonal to each other (impossible with ReLUs), then $\mathbf{\Phi}\mathbf{\Phi}^T$ would be perfectly well conditioned.

To understand this quantitatively, consider approximating the univariate function $u : [-1, 1] \to \mathbb{R}$ with a ReLU neural network using a fixed set of neurons. Instead of minimizing the ℓ_2 loss over a set of finite samples we will instead consider minimizing the L_2 loss between the neural network $f_{\boldsymbol{\theta}}$ and the function u

$$\min_{\boldsymbol{v}} \frac{1}{2}\int_D \left(u(x) - u(-1) - \sum_{k=1}^{K} v_k \sigma(x - b_k)\right)^2 dx. \tag{9}$$

A simple expansion shows that solving this optimization problem is equivalent to solving

$$\min_{\boldsymbol{v}} \boldsymbol{v}^T \mathbf{G}_\sigma \boldsymbol{v} - \boldsymbol{r}_{u,\sigma}^T \boldsymbol{v}. \tag{10}$$

Where $\boldsymbol{r}_{u,\sigma} \in \mathbb{R}^K$ is defined as

$$(\boldsymbol{r}_{u,\sigma})_i = \int_D (u(x) - u(-1))\sigma(x - b_i). \tag{11}$$

Moreover, $\mathbf{G}_\sigma \in \mathbb{R}^{K\times K}$ is the Gram matrix for the feature embedding matrix and is defined as,

$$\mathbf{G}_{i,j} := \int_D \sigma(x - b_i)\sigma(x - b_j)dx. \tag{12}$$

Note that this is analogous to $\mathbf{\Phi}\mathbf{\Phi}^T$ discussed above. Therefore, the condition number of $\mathbf{G}_\sigma$ determines how ill-conditioned our problem is and indicates how effective each gradient step will be. Theorem 4 in Zhang et al. (2023) quantifies the condition number of $\mathbf{G}_\sigma$.

Theorem 3.1 ((Zhang et al., 2023))**.** *Suppose $D = [-1, 1]$ and $\{b_j\}_{j=1}^K$ are quasi-evenly spaced on $[-1, 1]$, $b_j = -1 + \frac{2(j-1)}{K} + o(\frac{1}{K})$. Let $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_K \geq 0$ be the eigenvalues of the Gram matrix $\mathbf{G}_\sigma$, then the condition number of $\mathbf{G}_\sigma$ satisfies*

$$\kappa(\mathbf{G}_\sigma) = \lambda_1/\lambda_K = \Omega(K^3).$$

The theorem shows us that even when the ReLU neurons are maximally separated, solving (7) for the output weights is a severely ill-conditioned problem. Moreover, the condition number of the feature matrix grows at a cubic rate. Thus, as we increase the number of neurons in the network, which increases the approximation power of the model, we are simultaneously hindering the ability of gradient descent to optimize the model.

We can potentially remedy this ill-conditioning by instead considering second order B-spline wavelets as our activation function (2) and approximating u by the following BW-ReLU neural network,

$$g_{\boldsymbol{\theta}}(x) = \sum_{k=1}^{K} v_k \psi(w_k x - b_k). \tag{13}$$

As discussed earlier this is equivalent to incorporating constraints on a regular ReLU neural network such that each group of ReLU neurons share weight and have a fixed orientation relative to the other neurons in the group. We first present a simple proposition establishing that any ReLU neural network can be represented by a BW-ReLU neural network over a bounded domain.

Proposition 3.2. *Let $f : [-1, 1] \to \mathbb{R}$ denote a ReLU neural network with K neurons. The network is of the form*

$$f(x) = c + \sum_{k=1}^{K} v_k \sigma(w_k x - b_k) \tag{14}$$

where $v_k \in \mathbb{R}$, $b_k \in [-1, 1]$, and $w_k \in \{-1, +1\}$ are the parameters of the model. Then there exists a BW-ReLU neural network $g : \mathbb{R} \to \mathbb{R}$ with the same number of neurons of the form

$$g(x) = c + \sum_{k=1}^{K} 24 v_k \psi\left(\frac{1}{4}(w_k x - b_k)\right) \tag{15}$$

such that g represents f on the bounded domain $[-1, 1]$.

The proof is in Appendix A. The proposition establishes the fact that on a bounded domain (the setting relevant to INRs) any function which we can represent using a ReLU neural network can also be represented by a BW-ReLU neural network with the same number of neurons. Therefore, no representation power is lost by incorporating these constraints into the ReLU network.

Now consider approximating the function $u : [-1, 1] \to \mathbb{R}$ by a univariate BW-ReLU neural network optimizing over the output weights with a fixed set of neurons. The L_2 loss in this case is,

$$\min_{\boldsymbol{v}} \frac{1}{2} \int_0^1 \left(u(x) - \sum_{k=1}^{K} v_k \psi(w_k x - b_k) \right)^2 dx. \tag{16}$$

This is equivalent to solving

$$\min_{\boldsymbol{v}} \boldsymbol{v}^T \mathbf{G}_\psi \boldsymbol{v} - \boldsymbol{r}_{u,\psi}^T \boldsymbol{v} \tag{17}$$

where

$$(\mathbf{G}_\psi)_{i,j} = \int_0^1 \psi(w_i x - b_i)\psi(w_j x - b_j) dx. \tag{18}$$

Our next theorem shows that the Gram matrix in this setting is far better conditioned than in the case of ReLUs.

Theorem 3.3. *Suppose $D = [-1, 1]$ and consider a BW-ReLU neural network with $K = 2^J - 1$ neurons of the form,*

$$\psi_{j,k}(x) = 2^{j/2}\psi\left(2^j \frac{3}{2}(x+1) - k\right) \quad \begin{aligned} &j = 0, \dots, J-1 \\ &k = 0, \dots, 2^j - 1, \end{aligned}$$

for any $J \in \mathbb{N}^+$. For each scale $j = 0, \cdots, J-1$ we have $k = 0, \dots, 2^j - 1$ shifted versions of the B-spline wavelets. Let $\lambda_1 \geq \lambda_2 \cdots \geq \lambda_K \geq 0$ be the eigenvalues of $\mathbf{G}_\psi$. Using these neurons the condition number of $\mathbf{G}_\psi$ satisfies

$$\kappa(\mathbf{G}_\psi) = \lambda_1/\lambda_K = \mathcal{O}(1). \tag{19}$$

The proof is in Appendix B, we note that the normalization constant is not required and merely simplifies the analysis. The proof relies on key properties of the B-spline wavelets. In particular, the fact that they are *semiorthogonal*, this ensures that wavelets of different scale are orthogonal to each other. For instance in the dyadic wavelet system developed in the theorem $\langle \psi_{j,k}, \psi_{i,\ell} \rangle = 0$ for any $i \neq j$ and any $k, \ell \in \mathbb{Z}$.

In Figure 2 we present a numerical example of how the condition number of the feature embedding matrix changes during training when using a ReLU or BW-ReLU neural network to fit a univariate function. This illustrates how the Gram matrix of the feature embeddings remain well conditioned both at initialization and throughout training for the BW-ReLU neural network while the ReLU neural network suffers from a poorly conditioned feature embedding matrix all throughout training. We also see that the well conditioned feature embedding matrix correlates with a faster rate of convergence.

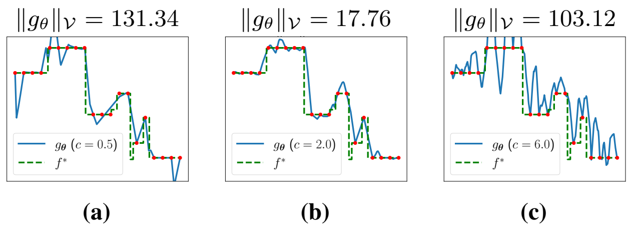


Figure 3. The variation norm of BW-ReLU neural networks, $g_{\boldsymbol{\theta}}$ with various scales trained on univariate data. The red dots indicate our samples from the ground truth function f^*. We see that making c too low leads to a poor fit to the data and a very high variation norm. On the other hand making c to large results in a very oscillatory fit to the data. The interpolator which generalizes best corresponds to the one with the lowest variation norm.

Moreover, our experiments in Section 5 also demonstrate that deep BW-ReLU neural networks are not susceptible to this ill-conditioning and can be utilized for real INR tasks.

The Gaussian (Ramasinghe & Lucey, 2022) and Gabor wavelets (Saragadam et al., 2023) have also been utilized as activation functions for INR tasks due to their localized nature. However, there is little theory characterizing the kinds of functions such networks learn and the effects their hyperparameters have on the learned function. In the next section we leverage the fact that we are ultimately learning a ReLU neural network. This allows us to utilize much of the recent theory characterizing of the function space associated with ReLU neural networks giving insights into some of the heuristics used when training INRs.

4. Variation Norm and Scale

Having demonstrated how we *can* learn ReLU-based INRs by imposing constraints on groups of neurons, we now explain the benefits of obtaining such a representation. Due to the prevalence and simplicity of the ReLU activation function, much of our theoretical understanding of DNNs has been focused on those with ReLU activations. In particular, a line of work (Bach, 2017; Savarese et al., 2019; Ongie et al., 2020; Parhi & Nowak, 2021; Wojtowytsch, 2020; Parhi & Nowak, 2023a; Siegel & Xu, 2023; Bartolucci et al., 2023; Chen, 2023; Shenouda et al., 2023; Zeno et al., 2023) has investigated the *kinds* of functions which are learned when fitting ReLU neural networks to data. This mathematical framework has provided many insights into the inner workings and success of neural networks. For instance, it sheds light on why neural networks seemingly break the curse of dimensionality (Klusowski & Barron, 2018) and how they differ from kernel methods (Parhi & Nowak, 2023b). Moreover, this perspective offers an explanation for the role of various heuristics that are commonly employed when training neural networks such as weight decay, bottleneck linear layers and skip connections (Parhi & Nowak, 2022; Shenouda et al., 2023). We will soon see how this perspective can also provide insights into one of the hyperparameters uniquely used when learning INRs.

These efforts have revealed that for neural networks with ReLU activations the common regularization technique of weight decay corresponds to regularizing a certain *variation norm* (Kurková & Sanguineti, 2001). This norm is related to total variation and is the appropriate norm for functions represented by ReLU neural networks. This norm provides a measure of smoothness for the learned function and, remarkably, can be easily computed in terms of the weights of the model (Neyshabur et al., 2015; Savarese et al., 2019; Parhi & Nowak, 2021). In the univariate case, the variation norm is the total variation of the derivative of the function $f_{\boldsymbol{\theta}}$ defined by the neural network. If $f_{\boldsymbol{\theta}}$ has a continuous derivative, then this is equivalent to the L_1 norm of the second derivative. The variation norm is also well-defined for continuous functions having discontinuous derivatives (in which case the second derivative will be a generalized function having Dirac impulses). In the multivariate case, the variation norm is essentially the L_1 norm of the Radon transform of the Laplacian (second derivative operator) of the function. This is equivalent to considering the L_1 norm of the second derivative of the function along each direction of the multidimensional domain. For a ReLU neuron, the directional second derivative is 0 in all but one direction determined by the orientation of the neuron. And in that direction the second derivative is an impulse with magnitude equal to the slope of the ReLU. Thus, the variation norm has a clear connection to the smoothness of the function, as measured by the size of the second derivatives. We refer the reader to the recent survey Parhi & Nowak (2023a) for more details. We now present this variation norm and show how it can be computed for our BW-ReLU neural networks.

First, consider a function $f_{\boldsymbol{\theta}}$ represented by a ReLU neural network of the form,

$$f_{\boldsymbol{\theta}}(\boldsymbol{x}) = \sum_{k=1}^{K} \boldsymbol{v}_k \sigma(\boldsymbol{w}_k^T \boldsymbol{x} - b_k), \tag{20}$$

where $\boldsymbol{\theta} = (\boldsymbol{v}_k, \boldsymbol{w}_k, b_k)_{k=1}^K$ and $\sigma(\cdot) : \mathbb{R} \to \mathbb{R}$ is the ReLU activation function applied elementwise. Each neuron, $\eta_{\boldsymbol{v},\boldsymbol{w}}(\boldsymbol{x}) = \boldsymbol{v}\sigma(\boldsymbol{w}^T \boldsymbol{x} - b)$ contributes to the variation norm of the function. Explicitly, the variation norm for each ReLU neuron is,

$$\|\eta_{\boldsymbol{v},\boldsymbol{w}}\|_{\mathcal{V}} = \|\boldsymbol{v}\|_2 \|\boldsymbol{w}\|_2, \tag{21}$$

where $\|\cdot\|_{\mathcal{V}}$ denotes the variation norm of a function. Note the biases are not regularized. The norm of the entire function is computed by summing up the contribution from each neuron such that,

$$\|f_{\boldsymbol{\theta}}\|_{\mathcal{V}} = \sum_{k=1}^{K} \|\boldsymbol{v}_k\|_2 \|\boldsymbol{w}_k\|_2. \tag{22}$$

Now consider a function represented by a BW-ReLU neural network of the form

$$g_{\boldsymbol{\theta}}(\boldsymbol{x}) = \sum_{k=1}^{K} \boldsymbol{v}_k \psi(\boldsymbol{w}_k^T \boldsymbol{x} - b_k). \tag{23}$$

This network can be exactly represented by a ReLU neural network with $7K$ neurons by the definition of the B-spline wavelet activation function (2). For each BW-ReLU neuron $\gamma_{\boldsymbol{w},\boldsymbol{v}}(\boldsymbol{x}) = \boldsymbol{v}\psi(\boldsymbol{w}^T \boldsymbol{x} - b)$ its variation norm is found by summing up the variation norm of each of the scaled ReLUs in the definition of the B-spline wavelet (2), from this we get that

$$\|\gamma_{\boldsymbol{w},\boldsymbol{v}}\|_{\mathcal{V}} = 16\|\boldsymbol{v}\|_2 \|\boldsymbol{w}\|_2. \tag{24}$$

We can see that it is simply a multiple of the total variation of each ReLU neuron. Again, since the variation norm of a neural network function consists of summing up the variation norm of each neuron we have that for the BW-ReLU neural network its variation norm can be explicitly computed as,

$$\|g_{\boldsymbol{\theta}}\|_{\mathcal{V}} = 16 \sum_{k=1}^{K} \|\boldsymbol{v}\|_2 \|\boldsymbol{w}\|_2. \tag{25}$$

4.1. Understanding Scaling via Regularization

It is common to introduce an additional hyperparameter to the neurons used in INR applications, which scales each activation function as follows. Consider an activation function ζ. Each neuron applies this activation function to $\boldsymbol{w}^T \boldsymbol{x} - b$, producing the output $\zeta(\boldsymbol{w}^T \boldsymbol{x} - b)$. Most INR networks employ an additional activation *scaling* parameter $c > 0$, so that each neuron output is instead computed by $\zeta(c(\boldsymbol{w}^T \boldsymbol{x} - b))$. The scaling parameter can have a significant impact for certain activation functions, while being ineffectual others.

To illustrate, first consider the ReLU activation. In this case, $\sigma(c(\boldsymbol{w}^T \boldsymbol{x} - b)) = c\sigma(\boldsymbol{w}^T \boldsymbol{x} - b)$, so the the scaling affects the magnitude of output but has no other effect. This is due to the fact that the ReLU activation is homogeneous (linear activations and leaky ReLUs are also homogeneous in this way). However, the activations commonly used in INR applications are inhomogeneous. Take for example, the commonly used sine activation function $\sin(\boldsymbol{w}^T \boldsymbol{x} - b)$ (Sitzmann et al., 2020). In this case if we scale by c, then $\sin(c(\boldsymbol{w}^T \boldsymbol{x} - b))$ has the same output range/magnitude but will have faster or slower oscillations relative to $\sin(\boldsymbol{w}^T \boldsymbol{x} - b)$, if $c > 1$ or $c < 1$.

Including the scaling parameter with inhomogeneous activations functions, like the sine function, can change the shape and variations of the activation function. Activation functions like the Gaussian (Ramasinghe & Lucey, 2022) the complex Gabor wavelets (Saragadam et al., 2023) and the B-spline wavelet used here, are compressed or dilated depending on whether $c > 1$ or $c < 1$, which also makes the support of the activation function smaller or larger, accordingly. These observations illustrate why scaling the activation function may have significant effects on INR performance. Generally speaking, large values of c tend to increase oscillations or variations in the activation function (and possibly decrease the support of the activation function). This fact has been used to explain why large values of c may help capture more detailed structure in INR applications (Yüce et al., 2022).

The story, however, is a bit subtle. Consider a standard training problem of the form

$$\min_{\boldsymbol{\theta}} \sum_{i=1}^{N} \ell\Big(\boldsymbol{y}_i, \sum_{k=1}^{K} \boldsymbol{v}_k \zeta\big(c(\boldsymbol{w}_k^T \boldsymbol{x}_i - b_k)\big)\Big), \tag{26}$$

where the minimization is with respect to all the weights and biases. If we place no restrictions on the allowable ranges of the input weights and biases, then the scaling factor c can simply be absorbed into the weights and biases. So why does c play a crucial role? One answer is that the neural networks are trained via gradient descent methods, typically initialized with small random weights. This, coupled with the fact that the training objective is nonconvex, tends to favor solutions that fit the data with weights of small magnitudes (even though the data might also be fit using much larger weights) (Vardi & Shamir, 2021). This is sometimes referred to as the implicit regularization of gradient descent.

To understand why the scaling factor c can play a significant role, it is enlightening to consider explicit regularization in the form of the commonly used weight decay regularization term, which is proportional to the sum of squared weights in the network. This is supported by established theoretical connections between weight decay and implicit regularization (Chizat & Bach, 2020). The overall training objective is to minimize the sum of losses plus the weight decay regularization term

$$\sum_{i=1}^{N} \ell\Big(\boldsymbol{y}_i, \sum_{k=1}^{K} \boldsymbol{v}_k \zeta\big(c(\boldsymbol{w}_k^T \boldsymbol{x}_i - b_k)\big)\Big) + \lambda \sum_{k=1}^{K} \|\boldsymbol{v}_k\|_2^2 + \|\boldsymbol{w}_k\|_2^2,$$

where ℓ is a loss function and $\lambda > 0$ is the weight decay parameter. We can simply reparameterize the problem by absorbing c into the weights to obtain an equivalent optimization

$$\sum_{i=1}^{N} \ell\Big(\boldsymbol{y}_i, \sum_{k=1}^{K} \boldsymbol{v}_k \zeta(\boldsymbol{w}_k^T \boldsymbol{x}_i - b_k)\Big) + \lambda \sum_{k=1}^{K} \|\boldsymbol{v}_k\|_2^2 + \frac{\|\boldsymbol{w}_k\|_2^2}{c^2}.$$

Both objectives have the same global minima. The second objective clearly reveals the effect of the scaling factor. If $c > 1$, then there is less regularization applied to the input weights compared to the output weights, and vice-versa if $c < 1$. So if $c > 1$, then the regularizer encourages solutions with *larger* input weights and hence increased oscillations or variations in inhomogeneous activation functions, like the sine, Gabor, or B-spline wavelet activations. In the case of localized activations like the wavelets, larger values of c also reduce the support (spatial scale) of the activation functions. However, unlike other activation functions, for B-spline wavelets the scaling parameter's effect on the regularity of the function can be readily understood in terms of the variation norm. Let us reparameterize our BW-ReLU neural network with a fixed scale $c > 0$ such that,

$$g_{\boldsymbol{\theta}}(\boldsymbol{x}) = \sum_{k=1}^{K} \boldsymbol{v}_k \psi(c \cdot (\boldsymbol{w}_k^T \boldsymbol{x} - b_k)). \tag{27}$$

It follows from the previous discussion that the variation norm of the function represented by this neural network can be computed as,

$$\|g_{\boldsymbol{\theta}}\|_{\mathcal{V}} = 16c \sum_{k=1}^{K} \|\boldsymbol{w}_k\|_2 \|\boldsymbol{v}_k\|_2. \tag{28}$$

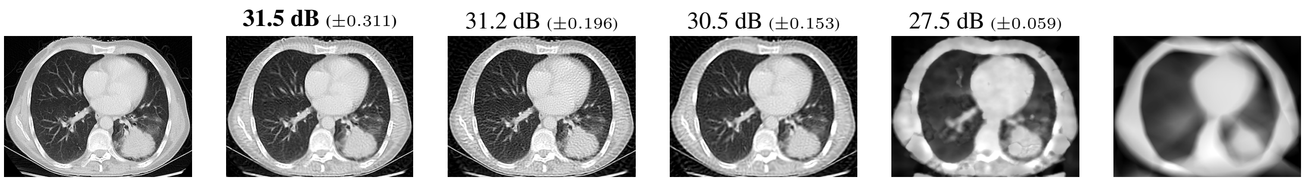
Thus a very high c result in functions that is more irregular while using lower values of c can lead to smoother functions.

In Figure 3 we present a simple univariate data fitting problem to illustrate the role of the scaling parameter c and how it can effect regularity of the learned function. We see that the interpolator which generalizes best is the one that minimizes the variation norm. Using a c value which is too large results in a very oscillatory function with a high variation norm. However, if we instead make c too small then the output weight must increase considerably to compensate for the wider B-spline wavelets. Moreover, in Section 5.4 we illustrate how the variation norm can be a good indicator for how well our learned function will perform when using INRs to solve inverse problems.

5. Experiments

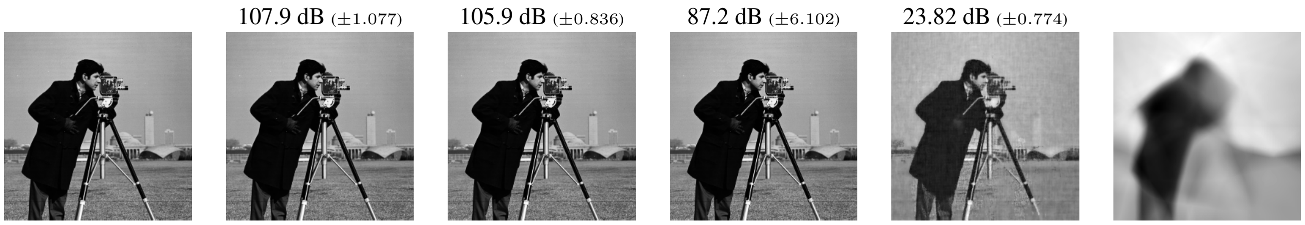
Here we demonstrate how our BW-ReLU neural networks can be as effective as other INR architectures for three INR tasks. We compare our method against SIREN (Sitzmann et al., 2020), WIRE (Saragadam et al., 2023) and ReLUs + Positional Encoding (P.E.) introduced in (Mildenhall et al., 2021)[1]. The hyperparameters for each of the INR architectures were tuned to to give the best results. For all of our experiments we utilized a three hidden layer DNN. The

[1]All the code for reproducing the experiments can be found at `https://github.com/joeshenouda/relu-inrs`.



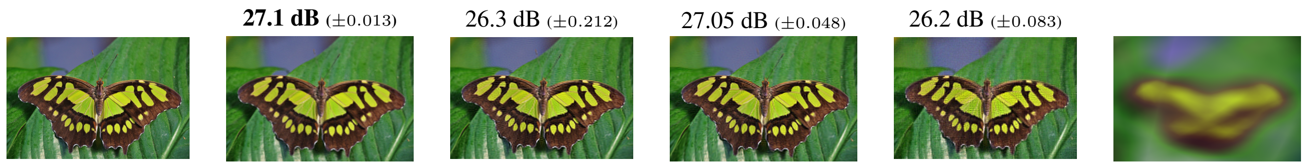
(a) Original **(b)** BW-ReLU **(c)** SIREN **(d)** WIRE **(e)** ReLU+P.E. **(f)** ReLU

Figure 4. Experiments on computed tomography reconstruction for various INR architectures. We report average PSNR and standard error across five random trials.



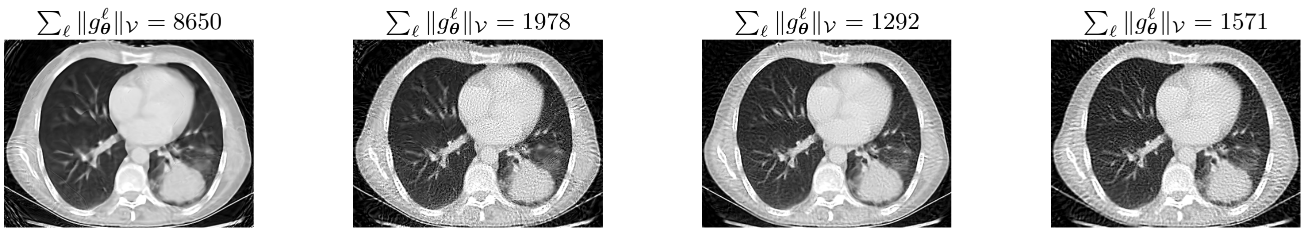
(a) Original **(b)** BW-ReLU **(c)** SIREN **(d)** WIRE **(e)** ReLU + P.E. **(f)** ReLU

Figure 5. Experiments on signal representation for various INR architectures. We report average PSNR and standard error across five random trials.



(a) Original **(b)** BW-ReLU **(c)** SIREN **(d)** WIRE **(e)** ReLU+P.E. **(f)** ReLU

Figure 6. Experiments on the super resolution task for various INR architectures. We report average PSNR and standard error across five random trials.


$\sum_\ell \|g^\ell_{\theta}\|_{\mathcal{V}} = 8650$
$\sum_\ell \|g^\ell_{\theta}\|_{\mathcal{V}} = 1978$
$\sum_\ell \|g^\ell_{\theta}\|_{\mathcal{V}} = 1292$
$\sum_\ell \|g^\ell_{\theta}\|_{\mathcal{V}} = 1571$

(a) PSNR: 29.9 dB $c = 1$ **(b)** PSNR: 28.7 dB $c = 2$ **(c)** PSNR: **32.1 dB** $c = 3$ **(d)** PSNR: 29.1 dB $c = 5$

Figure 7. Four BW-ReLU DNNs trained on the CT reconstruction task with different values of c. All networks are trained to the same training loss. We see that the c which produces the highest PSNR corresponds to the one with the lowest variation norm across all layers.

full training details for all experiments can be found in Appendix D.

5.1. Computed Tomography(CT) Reconstruction

In this experiment we simulated CT reconstruction by taking 100 equally spaced CT measurements of a 326×435 chest X-ray image (Clark et al., 2013). Figure 4 shows the results compared to other INR architectures showing that our BW-ReLU neural networks perform just as well and perhaps slightly better than conventional INR architectures.

5.2. Signal Representation

In this experiment we fit the four INR architectures to the standard 256×256 cameraman image. The results are shown in Figure 5 where we have also included the case of using a traditional ReLU DNN for comparison. The results show that our method can perform comparably to the newly introduced INR architectures achieving a high PSNR as fast as the other methods.

5.3. Super resolution

We implemented $4\times$ super resolution on a single image from the DIV2K image dataset (Agustsson & Timofte, 2017). In this case our BW-ReLU neural networks performs slightly better than the rest. The results on all four INR architectures are shown in Figure 6.

5.4. Low Norm Solutions and Inverse Problems

For our last experiment we trained the BW-ReLU neural network for the CT reconstruction task with three different scaling parameters. In Figure 7 all three BW-ReLU neural networks that were used to reconstruct the image achieved the same training loss on the CT measurement data. However we see that the model with the smallest variation norm across all 3 layers corresponds to the best reconstruction. This suggests a principled methodology for choosing the scaling parameter c in the case of inverse problems.

6. Conclusion

In this work we presented a simple way to utilize ReLU DNNs for INR tasks. Unlike previous works we focusing solely on remedying the ill-conditioning of the optimization problem without sacrificing the ReLU by drawing a connection to B-spline wavelets. We then related our methodology to the function space associated with ReLU neural networks and showed how this framework can be useful in understanding and quantifying the regularity of our INRs. This connection suggests a more principled approach to tuning INRs. For future work it would interesting to apply this technique to more INR tasks. In particular neural radiance fields and physics informed neural networks.

Acknowledgement

The authors would like to thank Liu Yang and Rahul Parhi for helpful conversations. The authors would also like to thank Ahmet Alacaoglu for his insights in the later stages of this project. Joseph Shenouda was supported by the ONR MURI grant N00014-20-1-2787 and NSF grant DMS-2023239. Robert Nowak was supported in part by the NSF grants DMS-2134140 and DMS-2023239, the ONR MURI grant N00014-20-1-2787, and the AFOSR/AFRL grant FA9550-18-1-0166.

Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none which we feel must be specifically highlighted here.

References

Agustsson, E. and Timofte, R. Ntire 2017 challenge on single image super-resolution: Dataset and study. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pp. 126–135, 2017.

Arora, R., Basu, A., Mianjy, P., and Mukherjee, A. Understanding deep neural networks with rectified linear units. In *International Conference on Learning Representations*, 2018. URL `https://openreview.net/forum?id=B1J_rgWRW`.

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

Bartolucci, F., De Vito, E., Rosasco, L., and Vigogna, S. Understanding neural networks with reproducing kernel banach spaces. *Applied and Computational Harmonic Analysis*, 62:194–236, 2023.

Boyd, S. P. and Vandenberghe, L. *Convex optimization*. Cambridge university press, 2004.

Candes, E. J. *Ridgelets: theory and applications*. Stanford University, 1998.

Candès, E. J. and Donoho, D. L. Ridgelets: A key to higher-dimensional intermittency? *Philosophical Transactions of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences*, 357(1760):2495–2509, 1999.

Chen, Y., Liu, S., and Wang, X. Learning continuous image representation with local implicit image function. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 8628–8638, 2021.

Chen, Z. Multi-layer neural networks as trainable ladders of hilbert spaces. In *International Conference on Machine Learning*. PMLR, 2023.

Chizat, L. and Bach, F. Implicit bias of gradient descent for wide two-layer neural networks trained with the logistic loss. In *Conference on Learning Theory*, pp. 1305–1338. PMLR, 2020.

Chui, C. K. and Wang, J.-z. On compactly supported spline wavelets and a duality principle. *Transactions of the American Mathematical Society*, 330(2):903–915, 1992.

Clark, K., Vendt, B., Smith, K., Freymann, J., Kirby, J., Koppel, P., Moore, S., Phillips, S., Maffitt, D., Pringle, M., et al. The cancer imaging archive (tcia): maintaining and operating a public information repository. *Journal of digital imaging*, 26:1045–1057, 2013.

Horn, R. A. and Johnson, C. R. *Matrix analysis*. Cambridge university press, 2012.

Klusowski, J. M. and Barron, A. R. Approximation by combinations of relu and squared relu ridge functions with ℓ^1 and ℓ^0 controls. *IEEE Transactions on Information Theory*, 64(12):7649–7656, 2018.

Kurková, V. and Sanguineti, M. Bounds on rates of variable-basis and neural-network approximation. *IEEE Transactions on Information Theory*, 47(6):2659–2665, 2001.

Kurtz, M., Kopinsky, J., Gelashvili, R., Matveev, A., Carr, J., Goin, M., Leiserson, W., Moore, S., Shavit, N., and Alistarh, D. Inducing and exploiting activation sparsity for fast inference on deep neural networks. In *International Conference on Machine Learning*, pp. 5533–5543. PMLR, 2020.

Li, Z., You, C., Bhojanapalli, S., Li, D., Rawat, A. S., Reddi, S., Ye, K., Chern, F., Yu, F., Guo, R., et al. The lazy neuron phenomenon: On emergence of activation sparsity in transformers. In *Conference on Parsimony and Learning (Recent Spotlight Track)*, 2023.

Mildenhall, B., Srinivasan, P. P., Tancik, M., Barron, J. T., Ramamoorthi, R., and Ng, R. Nerf: Representing scenes as neural radiance fields for view synthesis. *Communications of the ACM*, 65(1):99–106, 2021.

Mirzadeh, I., Alizadeh, K., Mehta, S., Del Mundo, C. C., Tuzel, O., Samei, G., Rastegari, M., and Farajtabar, M. ReLU strikes back: Exploiting activation sparsity in large language models. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=osoWxY8q2E.

Neyshabur, B., Salakhutdinov, R. R., and Srebro, N. Path-sgd: Path-normalized optimization in deep neural networks. *Advances in neural information processing systems*, 28, 2015.

Ongie, G., Willett, R., Soudry, D., and Srebro, N. A function space view of bounded norm infinite width relu nets: The multivariate case. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=H1lNPxHKDH.

Parhi, R. and Nowak, R. D. Banach space representer theorems for neural networks and ridge splines. *The Journal of Machine Learning Research*, 22(1):1960–1999, 2021.

Parhi, R. and Nowak, R. D. What kinds of functions do deep neural networks learn? insights from variational spline theory. *SIAM Journal on Mathematics of Data Science*, 4 (2):464–489, 2022.

Parhi, R. and Nowak, R. D. Deep learning meets sparse regularization: A signal processing perspective. *IEEE Signal Processing Magazine*, 40(6):63–74, 2023a. doi: 10.1109/MSP.2023.3286988.

Parhi, R. and Nowak, R. D. Near-minimax optimal estimation with shallow relu neural networks. *IEEE Transactions on Information Theory*, 69(2):1125–1140, 2023b.

Rahaman, N., Baratin, A., Arpit, D., Draxler, F., Lin, M., Hamprecht, F., Bengio, Y., and Courville, A. On the spectral bias of neural networks. In *International Conference on Machine Learning*, pp. 5301–5310. PMLR, 2019.

Ramasinghe, S. and Lucey, S. Beyond periodicity: Towards a unifying framework for activations in coordinate-mlps. In *European Conference on Computer Vision*, pp. 142–158. Springer, 2022.

Saragadam, V., LeJeune, D., Tan, J., Balakrishnan, G., Veeraraghavan, A., and Baraniuk, R. G. Wire: Wavelet implicit neural representations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 18507–18516, 2023.

Savarese, P., Evron, I., Soudry, D., and Srebro, N. How do infinite width bounded norm networks look in function space? In *Conference on Learning Theory*, pp. 2667–2690. PMLR, 2019.

Shenouda, J., Parhi, R., Lee, K., and Nowak, R. D. Variation spaces for multi-output neural networks: Insights on multi-task learning and network compression. *arXiv preprint arXiv:2305.16534*, 2023.

Siegel, J. W. and Xu, J. Characterization of the variation spaces corresponding to shallow neural networks. *Constructive Approximation*, pp. 1–24, 2023.

Sitzmann, V., Martel, J., Bergman, A., Lindell, D., and Wetzstein, G. Implicit neural representations with periodic activation functions. *Advances in neural information processing systems*, 33:7462–7473, 2020.

Sun, Y., Liu, J., Xie, M., Wohlberg, B., and Kamilov, U. S. Coil: Coordinate-based internal learning for tomographic imaging. *IEEE Transactions on Computational Imaging*, 7:1400–1412, 2021. doi: 10.1109/TCI.2021.3125564.

Tancik, M., Srinivasan, P., Mildenhall, B., Fridovich-Keil, S., Raghavan, N., Singhal, U., Ramamoorthi, R., Barron, J., and Ng, R. Fourier features let networks learn high frequency functions in low dimensional domains. *Advances in Neural Information Processing Systems*, 33: 7537–7547, 2020.

Unser, M., Aldroubi, A., and Eden, M. On the asymptotic convergence of b-spline wavelets to gabor functions. *IEEE Transactions on Information Theory*, 38(2):864–872, 1992.

Unser, M., Aldroubi, A., and Eden, M. A family of polynomial spline wavelet transforms. *Signal processing*, 30(2): 141–162, 1993.

Vardi, G. and Shamir, O. Implicit regularization in relu networks with the square loss. In *Conference on Learning Theory*, pp. 4224–4258. PMLR, 2021.

Wojtowytsch, S. On the banach spaces associated with multi-layer relu networks: Function representation, approximation theory and gradient descent dynamics. *CSIAM Transactions on Applied Mathematics*, 1:387–440, 06 2020. doi: 10.4208/csiam-am.20-211.

Wu, Q., Feng, R., Wei, H., Yu, J., and Zhang, Y. Self-supervised coordinate projection network for sparse-view computed tomography. *IEEE Transactions on Computational Imaging*, 2023.

Yüce, G., Ortiz-Jiménez, G., Besbinar, B., and Frossard, P. A structured dictionary perspective on implicit neural representations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 19228–19238, 2022.

Zeno, C., Ongie, G., Blumenfeld, Y., Weinberger, N., and Soudry, D. How do minimum-norm shallow denoisers look in function space? In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL `https://openreview.net/forum?id=gdzxWGGxWE`.

Zhang, S., Zhao, H., Zhong, Y., and Zhou, H. Why shallow networks struggle with approximating and learning high frequency: A numerical study. *arXiv preprint arXiv:2306.17301*, 2023.

A. Proof of Lemma 3.2

Proof. Let $\psi(x)$ denote the second order B-spline wavelet. We can express this as a linear combination of seven ReLU functions,

$$\psi(x) = \frac{1}{6}\sigma(x) - \frac{8}{6}\sigma\left(x - \frac{1}{2}\right) + \frac{23}{6}\sigma(x-1) - \frac{16}{3}\sigma\left(x - \frac{3}{2}\right) + \frac{23}{6}\sigma(x-2) - \frac{8}{6}\sigma\left(x - \frac{5}{2}\right) + \frac{1}{6}\sigma(x-3). \quad (29)$$

With this representation of the B-spline wavelet, it is clear that one ReLU on the interval $[-1, 1]$ can be represented in terms of a B-spline wavelet as

$$\sigma(x)\mathbf{1}_{[-1,1]} = 24\psi\left(\frac{1}{4}x\right)\mathbf{1}_{[-1,1]}, \quad (30)$$

where $\mathbf{1}_{[-1,1]}$ denotes an indicator function on $[-1, 1]$. Therefore if we define $g : [-1, 1] \to \mathbb{R}$ as

$$g(x) = c + \sum_{k=1}^{K} 24 v_k \psi\left(\frac{1}{4}(w_k x - b_k)\right) \quad (31)$$

then clearly

$$f(x) = g(x) \quad \forall x \in [-1, 1]. \quad (32)$$

$\square$

B. Proof of Theorem 3.3

Our proof relies on Gershgorin's circle theorem which we recall here.

Theorem B.1 (Gershgorin circle theorem (Horn & Johnson, 2012)). *Let $A \in \mathbb{R}^{n\times n}$ with entries a_{ij}. For $i = 1, \cdots, n$ let R_i be the sum of the absolute values of off diagonals in each row of A*

$$R_i(A) = \sum_{i\neq j} |a_{ij}|.$$

Consider the n Gershgorin discs defined as

$$\{z \in \mathbb{C} : |z - a_{ii}| \leq R_i(A)\}.$$

Then every eigenvalue of A lies within at least one of the Gershgorin discs.

We now proceed to the proof of Theorem 3.3.

Proof of Theorem 3.3. Our proof follows by bounding the sum of the absolute values of the off-diagonals of each row of $\mathbf{G}_\psi$ and then employing Gershgorin's circle theorem. Recall that, for each scale $j = 0, \ldots, J-1$ we have $k = 0, 1, \ldots, 2^j - 1$ shifted versions of the neurons at this scale. For ease of notation we define the neurons as

$$\psi_{j,k}(x) := 2^{j/2} \cdot \psi(2^j(3/2)(x+1) - k).$$

Note that the scaling factor $2^{j/2}$ ensures that the L_2 norm of all the neurons are equal. The elements in $\mathbf{G}_\psi \in \mathbb{R}^{K\times K}$ consist of inner products between every pair of neurons.

The diagonal entries of $\mathbf{G}_\psi$ are simply the squared L_2 norm of each neuron,

$$\begin{aligned}\|\psi_{j,k}\|_2^2 &= \int_D (2^{j/2}\psi(2^j(3/2)(x+1) - k))^2 dx \\ &= \int_{-1}^{1} (2^{j/2}\psi(2^j(3/2)(x+1) - k))^2 dx\end{aligned}$$

Let $t = \frac{3}{2}(x+1)$, then by a change of variables we have

$$\|\psi_{j,k}\|_2^2 = \frac{2}{3}\int_0^3 (2^{j/2}\psi(2^j t - k))^2 dt = \frac{2}{3}\left(\frac{1}{4}\right) = \frac{1}{6}.$$

The $\frac{1}{4}$ follows from the fact that $\|\psi\|_2^2 = \frac{1}{4}$ and our normalization ensures that translations and dilation of the wavelet preserve its norm.

For the first row of $\mathbf{G}_\psi$ we have,

$$R_1(\mathbf{G}_\psi) = 0.$$

This follows from the fact that each column in the first row consists of an inner product between $\psi_{0,0}$ and $\psi_{j,k}$ for all resolution of $j > 0$ and all possible shifts at each resolution. Since the B-spline wavelets are semiorthogonal (Chui & Wang, 1992) we must have that whenever $i \neq j$ and for any $k, \ell \in \mathbb{Z}$,

$$\langle \psi_{j,k}, \psi_{i,\ell} \rangle_{L_2(D)} = 0.$$

Where $\langle \cdot, \cdot \rangle_{L_2}$ denotes the L_2 inner product of two functions. From this we can conclude that $(\mathbf{G}_\psi)_{0,\ell} = 0$ for all $\ell = 0, \cdots, K-1$.

Now for the rest of the rows each row is identified with a neuron $\psi_{j,k}$ at a certain scale 2^j and shift k. The columns in this row are inner products of $\psi_{j,k}$ with all the other neurons in the network. These are the off-diagonals that we will bound. For each neuron $\psi_{j,k}$ its inner product with all other neurons can be broken up into two cases.

Case 1: The other neurons are at a different scale. The first case consists of inner products between other neurons which are at a different scale than $\psi_{j,k}$ i.e. $\langle \psi_{j,k}, \psi_{m,p} \rangle_{L_2}$ for $j \neq m$ but any k, p. Again, thanks to the semiorthogonality of the B-spline wavelets (Chui & Wang, 1992) we have

$$\langle \psi_{j,k}, \psi_{m,p} \rangle_{L_2(D)} = 0. \quad (33)$$

Case 2: The other neurons are at the same scale. The second type of inner products are of the form,

$$\langle \psi_{j,k}, \psi_{j,p} \rangle_{L_2(D)}, \quad (34)$$

where $k, p = 0, \ldots, 2^j - 1$ and $k \neq p$. By the compactness of the B-spline wavelets, if $|k - p| \geq 3$ then the inner product is zero. Therefore it remains to bound

$$\langle \psi_{j,k}, \psi_{j,p} \rangle_{L_2(D)}$$

for the case when $|k-p| = 1$ or $|k-p| = 2$. A direct computation, again applying a change of variables, reveals that for the case of $|k-p| = 1$ we have

$$\frac{2}{3}\int_0^3 2^{j/2}\psi(2^j t) \cdot 2^{j/2}\psi(2^j t - 1) dt = 0.030864 = C_1.$$

Furthermore, when $|k - \ell| = 2$ another computation reveals that

$$\frac{2}{3}\int_0^3 2^{j/2}\psi(2^j t) \cdot 2^{j/2}\psi(2^j t - 2) dt = -0.0030864 = C_2.$$

At each scale j we could have at most two neuron functions which are at the same scale j but shifted by 1 and at most two neuron functions which are at the same scale but shifted by 2. Therefore, by Gershgorin's circle theorem we have that for all the eigenvalues of $\mathbf{G}_\psi$ they must satisfy,

$$\lambda_i \in \left\{ z \in \mathbb{R} : \left| z - \frac{1}{6} \right| \leq 2\left(|C_1| + |C_2|\right) \right\}.$$

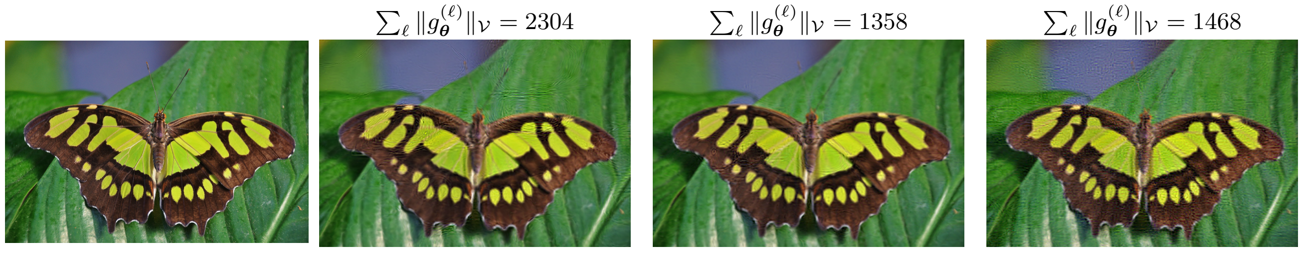
where $2(|C_1| + |C_2|) < \frac{1}{6}$. Therefore,

$$\kappa(\mathbf{G}_\psi) = \mathcal{O}(1). \quad (35)$$

□

C. Further Experiments on the Variation Norm

In this section we provide two more experiments demonstrating the correspondence between the optimal c parameter and the network with the smallest variation norm across all layers. Our first experiment demonstrates that the same observation holds for the superresolution experiment in Section 5. The results are shown in Figure 8. We also repeated the experiment in Figure 7 on an alternative CT image. The results are shown in Figure 9.


$\sum_\ell \|g_{\boldsymbol{\theta}}^{(\ell)}\|_{\mathcal{V}} = 2304$
$\sum_\ell \|g_{\boldsymbol{\theta}}^{(\ell)}\|_{\mathcal{V}} = 1358$
$\sum_\ell \|g_{\boldsymbol{\theta}}^{(\ell)}\|_{\mathcal{V}} = 1468$

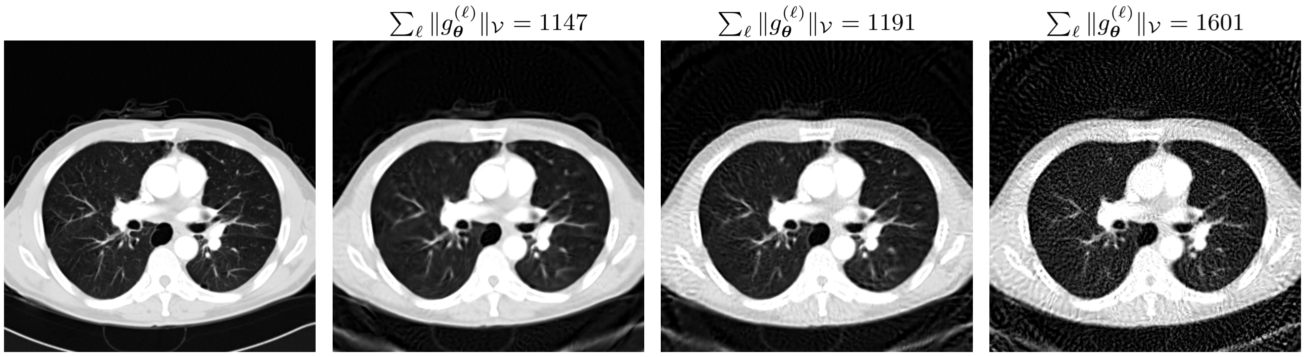
Original
Full Resolution Image

(a) PSNR: 26.25 dB
$c = 2$

(b) PSNR: **27.01 dB**
$c = 3$

(c) PSNR: 25.29 dB
$c = 5$

Figure 8. Three BW-ReLU DNNs trained on the superresolution task with different values of c. The c which produces the highest PSNR corresponds to the one with the lowest variation norm across all layers.


$\sum_\ell \|g_{\boldsymbol{\theta}}^{(\ell)}\|_{\mathcal{V}} = 1147$
$\sum_\ell \|g_{\boldsymbol{\theta}}^{(\ell)}\|_{\mathcal{V}} = 1191$
$\sum_\ell \|g_{\boldsymbol{\theta}}^{(\ell)}\|_{\mathcal{V}} = 1601$

Original
CT

(a) PSNR: **33.9 dB**
$c = 2$

(b) PSNR: 32.5 dB
$c = 3$

(c) PSNR: 26.1 dB
$c = 5$

Figure 9. Three BW-ReLU DNNs trained on a different CT reconstruction task with different values of c. All networks are trained to the same training loss. Again the c which produces the highest PSNR corresponds to the one with the lowest variation norm across all layers.

D. Training Details for Experiments

For all experiments we applied an exponential learning rate scheduler of the form

$$\eta(t) = \eta_0(r)^{t/T},$$

where η_0 is the initial learning rate, T is the total epochs we trained for and r is the decay rate.

D.1. Signal Representation

For the BW-ReLU DNN we used an initial learning rate of $\eta_0 = 4e - 3$ a scaling parameter applied to each neuron of $c = 9$. For SIREN we trained with an initial learning rate of $\eta_0 = 2e - 3$ and set the scale $\omega_0 = 50$. For WIRE we trained with a learning rate of $\eta_0 = 1e - 3$ setting $\sigma_0 = 10$ and $\omega_0 = 20$. Finally for ReLU + positional encoding we used an initial learning rate of $\eta_0 = 4e - 3$. All models were trained for 1000 epochs and the decay rate for the learning rate was $r = 0.1$. The architecture consisted of 3 hidden layers and 300 neurons per layer.

D.2. Super Resolution

We performed super resolution by training the DNN to minimize the mean squared error between a low resolution image of the Butterfly and a downsampled version of the full sample image produced by the DNN. For the BW-ReLU DNN we trained with an initial learning rate of $\eta_0 = 3e-3$ and scaling parameter $c = 3$. For SIREN we used an initial learning rate of $\eta_0 = 2e-3$ and set $\omega_0 = 12$. For WIRE we used an initial learning rate of $\eta_0 = 3e-3$ with $\omega_0 = 8$ and $\sigma_0 = 6$. The ReLU + positional encoding architecture used an initial learning rate of $\eta_0 = 4e-3$. All models were trained for 2000 epochs and the decay rate for the learning rate scheduler was $r = 0.2$. In all cases the architecture consisted of 3 hidden layers and 256 neurons per layer.

D.3. CT Reconstruction

We computed 100 equally spaced CT measurements of the X-ray image in the Radon domain. The DNN was trained by computing the mean squared error between the Radon transform of the image generated by the DNN and the ground truth CT measurements. For the BW-ReLU neural network we used a learning rate of $\eta_0 = 2e-3$ with $c = 3$. For SIREN we trained with a learning rate of $\eta_0 = 1e-3$ and set $\omega_0 = 25$. For WIRE a learning rate of $\eta_0 = 5e-3$ was used setting $\omega_0 = 7$ and $\sigma_0 = 10$. Finally for the ReLU + positional encoding we trained with a learning rate of $\eta_0 = 3e-3$. All methods were trained for 10000 epochs and the decay rate for the learning rate scheduler was $r = 0.1$. In all cases the architecture consisted of 3 hidden layers and 300 neurons per layer.

D.4. Low Norm Solutions and Inverse Problems

The learning rate we used in this experiment was varied for each value of c to ensure that all the networks achieved equal training loss. We considered the DNN as a composition of three shallow BW-ReLU neural networks and summed up the variation norm of each layer according to (28). In particular for the architecture defined as,

$$g(\boldsymbol{x}) = \mathbf{W}_3\psi\left(c \cdot \mathbf{W}_2\psi\left(c \cdot \mathbf{W}_1\psi\left(c \cdot \mathbf{W}_0\boldsymbol{x}\right)\right)\right).$$

We can treat it as a composition of 3 shallow nets of the form

$$\begin{aligned} g^{(1)}(\boldsymbol{x}) &= \mathbf{W}_1\psi\left(c \cdot \mathbf{W}_0\boldsymbol{x}\right) \quad \boldsymbol{x} \in \mathbb{R}^2 \\ g^{(2)}(\boldsymbol{x}) &= \mathbf{W}_2\psi\left(c \cdot \mathbf{I}\boldsymbol{x}\right) \quad \boldsymbol{x} \in \mathbb{R}^K \\ g^{(3)}(\boldsymbol{x}) &= \mathbf{W}_3\psi(c \cdot \mathbf{I}\boldsymbol{x}) \quad \boldsymbol{x} \in \mathbb{R}^K \end{aligned}$$

and then the whole DNN is

$$g(\boldsymbol{x}) = g_3 \circ g_2 \circ g_1.$$

The sum of the variation norm across all layers would be

$$\sum_{\ell} \|g^{(\ell)}\|_{\mathcal{V}} = 16c\left(\sum_{k}^{K} \|\boldsymbol{w}_k^{(1)}\|_2 \|\boldsymbol{w}_k^{(0)}\|_2 + \sum_{k=1}^{K} \|\boldsymbol{w}_k^{(2)}\|_2 + \sum_{k=1}^{K} \|\boldsymbol{w}_k^{(3)}\|_2\right). \tag{36}$$