On the Spectral Bias of Neural Networks

Nasim Rahaman [*] [1] [2] **Aristide Baratin** [*] [1] **Devansh Arpit** [1] **Felix Draxler** [2] **Min Lin** [1] **Fred A. Hamprecht** [2]
Yoshua Bengio [1] **Aaron Courville** [1]

Abstract

Neural networks are known to be a class of highly expressive functions able to fit even random input-output mappings with 100% accuracy. In this work we present properties of neural networks that complement this aspect of expressivity. By using tools from Fourier analysis, we highlight a learning bias of deep networks towards low frequency functions – i.e. functions that vary globally without local fluctuations – which manifests itself as a frequency-dependent learning speed. Intuitively, this property is in line with the observation that over-parameterized networks prioritize learning simple patterns that generalize across data samples. We also investigate the role of the shape of the data manifold by presenting empirical and theoretical evidence that, somewhat counter-intuitively, learning higher frequencies gets *easier* with increasing manifold complexity.

1. Introduction

The remarkable success of deep neural networks at generalizing to natural data is at odds with the traditional notions of model complexity and their empirically demonstrated ability to fit arbitrary random data to perfect accuracy (Zhang et al., 2017a; Arpit et al., 2017). This has prompted recent investigations of possible implicit regularization mechanisms inherent in the learning process which induce a bias towards low complexity solutions (Neyshabur et al., 2014; Soudry et al., 2017; Poggio et al., 2018; Neyshabur et al., 2017).

In this work, we take a slightly shifted view on implicit regularization by suggesting that low-complexity functions are *learned faster* during training by gradient descent. We expose this bias by taking a closer look at neural networks through the lens of Fourier analysis. While they can approximate arbitrary functions, we find that these networks prioritize learning the low frequency modes, a phenomenon we call the *spectral bias*. This bias manifests itself not just in the process of learning, but also in the parameterization of the model itself: in fact, we show that the lower frequency components of trained networks are more robust to random parameter perturbations. Finally, we also expose and analyze the rather intricate interplay between the spectral bias and the geometry of the data manifold by showing that high frequencies get easier to learn when the data lies on a lower-dimensional manifold of complex shape embedded in the input space of the model. We focus the discussion on networks with rectified linear unit (ReLU) activations, whose continuous piece-wise linear structure enables an analytic treatment.

Contributions[1]

1. We exploit the continuous piecewise-linear structure of ReLU networks to evaluate its Fourier spectrum (Section 2).

2. We find empirical evidence of a *spectral bias*: i.e. lower frequencies are learned first. We also show that lower frequencies are more robust to random perturbations of the network parameters (Section 3).

3. We study the role of the shape of the data manifold: we show how complex manifold shapes can facilitate the learning of higher frequencies and develop a theoretical understanding of this behavior (Section 4).

2. Fourier analysis of ReLU networks

2.1. Preliminaries

Throughout the paper we call 'ReLU network' a scalar function $f : \mathbb{R}^d \mapsto \mathbb{R}$ defined by a neural network with L hidden layers of widths $d_1, \cdots d_L$ and a single output neuron:

$$f(\mathbf{x}) = \left(T^{(L+1)} \circ \sigma \circ T^{(L)} \circ \cdots \circ \sigma \circ T^{(1)}\right)(\mathbf{x}) \qquad (1)$$

[*] Equal contribution [1] Mila, Quebec, Canada [2] Image Analysis and Learning Lab, Ruprecht-Karls-Universität Heidelberg, Germany. Correspondence to: Nasim Rahaman <nasim.rahaman@live.com>, Aristide Baratin <aristide.baratin@umontreal.ca>, Devansh Arpit <devansharpit@gmail.com>.

Proceedings of the 36th International Conference on Machine Learning, Long Beach, California, PMLR 97, 2019. Copyright 2019 by the author(s).

[1] Code: https://github.com/nasimrahaman/SpectralBias

where each $T^{(k)} : \mathbb{R}^{d_{k-1}} \to \mathbb{R}^{d_k}$ is an affine function ($d_0 = d$ and $d_{L+1} = 1$) and $\sigma(\mathbf{u})_i = \max(0, u_i)$ denotes the ReLU activation function acting elementwise on a vector $\mathbf{u} = (u_1, \cdots u_n)$. In the standard basis, $T^{(k)}(\mathbf{x}) = W^{(k)}\mathbf{x} + \mathbf{b}^{(k)}$ for some weight matrix $W^{(k)}$ and bias vector $\mathbf{b}^{(k)}$.

ReLU networks are known to be continuous piece-wise linear (CPWL) functions, where the linear regions are convex polytopes (Raghu et al., 2016; Montufar et al., 2014; Zhang et al., 2018; Arora et al., 2018). Remarkably, the converse also holds: every CPWL function can be represented by a ReLU network (Arora et al., 2018, Theorem 2.1), which in turn endows ReLU networks with universal approximation properties. Given the ReLU network f from Eqn. 1, we can make the piecewise linearity explicit by writing,

$$f(\mathbf{x}) = \sum_{\epsilon} 1_{P_\epsilon}(\mathbf{x}) \left(W_\epsilon \mathbf{x} + \mathbf{b}_\epsilon\right) \tag{2}$$

where ϵ is an index for the linear regions P_ϵ and 1_{P_ϵ} is the indicator function on P_ϵ. As shown in Appendix B in more detail, each region corresponds to an *activation pattern*[2] of all hidden neurons of the network, which is a binary vector with components conditioned on the sign of the input of the respective neuron. The $1 \times d$ matrix W_ϵ is given by

$$W_\epsilon = W^{(L+1)} W_\epsilon^{(L)} \cdots W_\epsilon^{(1)} \tag{3}$$

where $W_\epsilon^{(k)}$ is obtained from the original weight $W^{(k)}$ by setting its j^{th} column to zero whenever the neuron j of the k^{th} layer is inactive.

2.2. Fourier Spectrum

In the following, we study the structure of ReLU networks in terms of their Fourier representation, $f(\mathbf{x}) := (2\pi)^{d/2} \int \tilde{f}(\mathbf{k})\, e^{i\mathbf{k}\cdot\mathbf{x}} \mathbf{dk}$, where $\tilde{f}(\mathbf{k}) := \int f(\mathbf{x})\, e^{-i\mathbf{k}\cdot\mathbf{x}} \mathbf{dx}$ is the Fourier transform[3]. Lemmas 1 and 2 yield the explicit form of the Fourier components (we refer to Appendix C for the proofs and technical details).

Lemma 1. *The Fourier transform of ReLU networks decomposes as,*

$$\tilde{f}(\mathbf{k}) = i \sum_{\epsilon} \frac{W_\epsilon \mathbf{k}}{k^2} \tilde{1}_{P_\epsilon}(\mathbf{k}) \tag{4}$$

where $k = \|\mathbf{k}\|$ and $\tilde{1}_P(\mathbf{k}) = \int_P e^{-i\mathbf{k}\cdot\mathbf{x}} \mathbf{dx}$ is the Fourier transform of the indicator function of P.

The Fourier transform of the indicator over linear regions appearing in Eqn. 4 are fairly intricate mathematical objects. Diaz et al. (2016) develop an elegant procedure for evaluating it in arbitrary dimensions via a recursive application of Stokes theorem. We describe this procedure in detail[4] in Appendix C.2, and present here its main corollary.

Lemma 2. *Let P be a full dimensional polytope in $\mathbb{R}^d$. Its Fourier spectrum takes the form:*

$$\tilde{1}_P(\mathbf{k}) = \sum_{n=0}^{d} \frac{D_n(\mathbf{k}) 1_{G_n}(\mathbf{k})}{k^n} \tag{5}$$

where G_n is the union of n-dimensional subspaces that are orthogonal to some n-codimensional face of P, $D_n : \mathbb{R}^d \to \mathbb{C}$ is in $\Theta(1)$ $(k \to \infty)$ and 1_{G_n} the indicator over G_n.

Lemmas 1, 2 together yield the main result of this section.

Theorem 1. *The Fourier components of the ReLU network f_θ with parameters θ is given by the rational function:*

$$\tilde{f}_\theta(\mathbf{k}) = \sum_{n=0}^{d} \frac{C_n(\theta, \mathbf{k}) 1_{H_n^\theta}(\mathbf{k})}{k^{n+1}} \tag{6}$$

where H_n^θ is the union of n-dimensional subspaces that are orthogonal to some n-codimensional faces of some polytope P_ϵ and $C_n(\cdot, \theta) : \mathbb{R}^d \to \mathbb{C}$ is $\Theta(1)$ $(k \to \infty)$.

Note that Eqn 6 applies to general ReLU networks with arbitrary width and depth[5].

Discussion. We make the following two observations. First, the spectral decay of ReLU networks is highly anisotropic in large dimensions. In almost all directions of $\mathbb{R}^d$, we have a k^{-d-1} decay. However, the decay can be as slow as k^{-2} in specific directions orthogonal to the $d-1$ dimensional faces bounding the linear regions[6].

Second, the numerator in Eqn 6 is bounded by $N_f L_f$ (cf. Appendix C.3), where N_f is the number of linear regions and $L_f = \max_\epsilon \|W_\epsilon\|$ is the Lipschitz constant of the network. Further, the Lipschitz constant L_f can be bounded as (cf. Appendix C.6):

$$L_f \le \prod_{k=1}^{L+1} \|W^{(k)}\| \le \|\theta\|_\infty^{L+1} \sqrt{d} \prod_{k=1}^{L} d_k \tag{7}$$

where $\|\cdot\|$ is the spectral norm and $\|\cdot\|_\infty$ the max norm, and d_k is the number of units in the k-th layer. This makes the

[2] We adopt the terminology of Raghu et al. (2016); Montufar et al. (2014).

[3] Note that general ReLU networks need not be squared integrable: for instance, the class of two-layer ReLU networks represent an arrangement of hyperplanes (Montufar et al., 2014) and hence grow linearly as $x \to \infty$. In such cases, the Fourier transform is to be understood in the sense of tempered distributions acting on rapidly decaying smooth functions ϕ as $\langle \tilde{f}, \phi \rangle = \langle f, \tilde{\phi} \rangle$. See Appendix C for a formal treatment.

[4] We also generalize the construction to tempered distributions.

[5] Symmetries that might arise due to additional assumptions can be used to further develop Eqn 6, see e.g. Eldan & Shamir (2016) for 2-layer networks.

[6] Note that such a rate is *not* guaranteed by piecewise smoothness alone. For instance, the function $\sqrt{|x|}$ is continuous and smooth everywhere except at $x = 0$, yet it decays as $k^{-1.5}$ in the Fourier domain.

bound on L_f scale exponentially in depth and polynomial in width. As for the number N_f of linear regions, Montufar et al. (2014) and Raghu et al. (2016) obtain tight bounds that exhibit the same scaling behaviour (Raghu et al., 2016, Theorem 1). In Appendix A.5, we qualitatively ablate over the depth and width of the network to expose how this reflects on the Fourier spectrum of the network.

3. Lower Frequencies are Learned First

We now present experiments showing that networks tend to fit *lower frequencies first* during training. We refer to this phenomenon as the *spectral bias*, and discuss it in light of the results of Section 2.

3.1. Synthetic Experiments

Experiment 1. The setup is as follows[7]: Given frequencies $\kappa = (k_1, k_2, ...)$ with corresponding amplitudes $\alpha = (A_1, A_2, ...)$, and phases $\phi = (\varphi_1, \varphi_2, ...)$, we consider the mapping $\lambda : [0, 1] \to \mathbb{R}$ given by

$$\lambda(z) = \sum_i A_i \sin(2\pi k_i z + \varphi_i). \tag{8}$$

A 6-layer deep 256-unit wide ReLU network f_θ is trained to regress λ with $\kappa = (5, 10, ..., 45, 50)$ and $N = 200$ input samples spaced equally over $[0, 1]$; its spectrum $\tilde{f}_\theta(k)$ in expectation over $\varphi_i \sim U(0, 2\pi)$ is monitored as training progresses. In the first setting, we set equal amplitude $A_i = 1$ for all frequencies and in the second setting, the amplitude increases from $A_1 = 0.1$ to $A_{10} = 1$. Figure 1 shows the normalized magnitudes $|\tilde{f}_\theta(k_i)|/A_i$ at various frequencies, as training progresses with full-batch gradient descent. Further, Figure 2 shows the learned function at intermediate training iterations. The result is that lower frequencies (i.e. smaller k_i's) are regressed first, regardless of their amplitudes.

Experiment 2. Our goal here is to illustrate a phenomenon that complements the one highlighted above: lower frequencies are more *robust* to parameter perturbations. The set up is the same as in Experiment 1. The network is trained to regress a target function with frequencies $\kappa = (10, 15, 20, ..., 45, 50)$ and amplitudes $A_i = 1 \,\forall\, i$. After convergence to θ^*, we consider random (isotropic) perturbations $\theta = \theta^* + \delta\hat{\theta}$ of given magnitude δ, where $\hat{\theta}$ is a random unit vector in parameter space. We evaluate the network function f_θ at the perturbed parameters, and compute the magnitude of its discrete Fourier transform at frequencies k_i to obtain $|\tilde{f}_\theta(k_i)|$. We also average over 100 samples of $\hat{\theta}$ to obtain $|\tilde{f}_{\mathbb{E}\theta}(k_i)|$, which we normalize by $|\tilde{f}_{\theta*}(k_i)|$. Finally, we average over the phases ϕ (see Eqn 8). The result, shown in Figure 3, demonstrates that higher frequencies are significantly less robust than the lower ones, guiding the intuition that expressing higher frequencies requires the parameters to be finely-tuned to work together. In other words, parameters that contribute towards expressing high-frequency components occupy a small volume in the parameter space. We formalize this in Appendix D.

Discussion . Multiple theoretical aspects may underlie these observations. First, for a fixed architecture, recall that the numerator in Theorem 1 is[8] $\mathcal{O}(L_f)$ (where L_f is the Lipschitz constant of the function). However, L_f is bounded by the parameter norm, which can only increase gradually during training by gradient descent. This leads to the higher frequencies being learned[9] late in the optimization process. To confirm that the bound indeed increases as the model fits higher frequencies, we plot in Fig 1 the spectral norm of weights of each layer during training for both cases of constant and increasing amplitudes.

Second (cf. Appendix C.4), the exact form of the Fourier spectrum yields that for a fixed direction $\hat{\mathbf{k}}$, the spectral decay rate of the parameter gradient $\partial\tilde{f}/\partial\theta$ is at most one exponent of k lower than that of $\tilde{f}$. If for a fixed $\hat{\mathbf{k}}$ we have $\tilde{f} = \mathcal{O}(k^{-\Delta-1})$ where $1 \leq \Delta \leq d$, we obtain for the residual $h = f - \lambda$ and (continuous) training step t:

$$\left|\frac{d\tilde{h}(\mathbf{k})}{dt}\right| = \left|\frac{d\tilde{f}(\mathbf{k})}{dt}\right| = \underbrace{\left|\frac{d\tilde{f}(\mathbf{k})}{d\theta}\right|}_{\mathcal{O}(k^{-\Delta})} \overbrace{\left|\frac{d\theta}{dt}\right|}^{|\eta \cdot d\mathcal{L}/d\theta|} = \mathcal{O}(k^{-\Delta}) \tag{9}$$

where we use the fact that $d\theta/dt$ is just the learning rate times the parameter gradient of the loss which is independent[10] of k, and assume that the target function λ is fixed. Eqn 9 shows that the rate of change of the residual decays with increasing frequency, which is what we find in Experiment 1.

3.2. Real-Data Experiments

While Experiments 1 and 2 establish the spectral bias by explicitly evaluating the Fourier coefficients, doing so becomes prohibitively expensive for larger d (e.g. on MNIST). To tackle this, we propose the following set of experiments to measure the effect of spectral bias indirectly on MNIST.

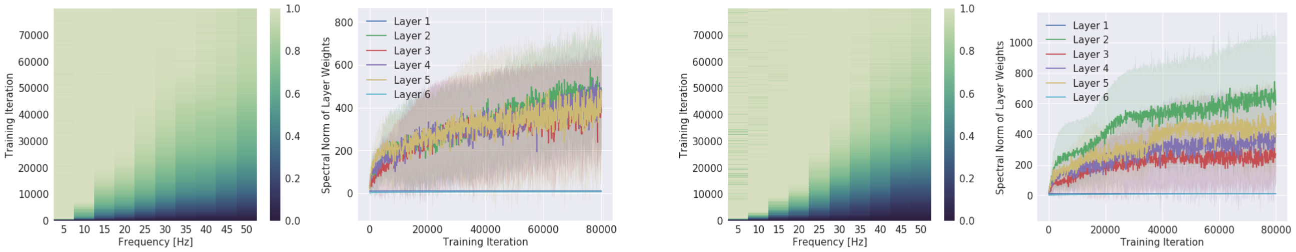
Experiment 3. In this experiment, we investigate how the validation performance dependent on the frequency of noise

[7] More experimental details and additional plots are provided in Appendix A.1.

[8] The tightness of this bound is verified empirically in appendix A.5.

[9] This assumes that the Lipschitz constant of the (noisy) target function is larger than that of the network at initialization.

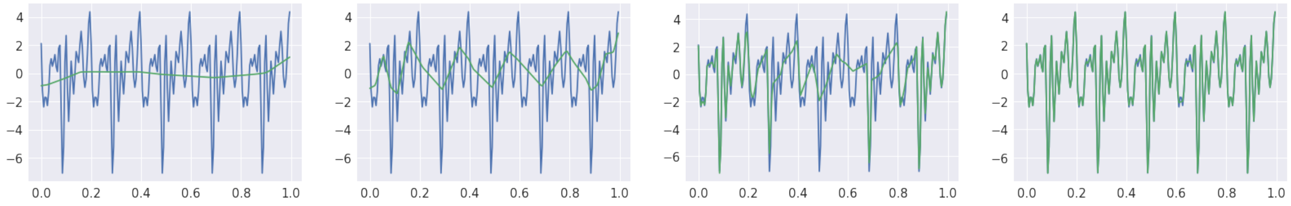
[10] Note however that the loss term might involve a sum or an integral over all frequencies, but the summation is over a different variable.



(a) Equal Amplitudes

(b) Increasing Amplitudes

Figure 1. Left (a, b): Evolution of the spectrum (x-axis for frequency) during training (y-axis). The colors show the measured amplitude of the network spectrum at the corresponding frequency, normalized by the target amplitude at the same frequency (i.e. $|\tilde{f}_{k_i}|/A_i$) and the colorbar is clipped between 0 and 1. Right (a, b): Evolution of the spectral norm (y-axis) of each layer during training (x-axis). Figure-set (a) shows the setting where all frequency components in the target function have the same amplitude, and (b) where higher frequencies have larger amplitudes. **Gist**: We find that even when higher frequencies have larger amplitudes, the model prioritizes learning lower frequencies first. We also find that the spectral norm of weights increases as the model fits higher frequency, which is what we expect from Theorem 1.



(a) Iteration 100

(b) Iteration 1000

(c) Iteration 10000

(d) Iteration 80000

Figure 2. The learnt function (green) overlayed on the target function (blue) as the training progresses. The target function is a superposition of sinusoids of frequencies $\kappa = (5, 10, ..., 45, 50)$, equal amplitudes and randomly sampled phases.

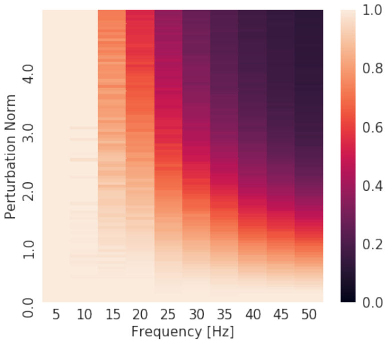


Figure 3. Normalized spectrum of the model (x-axis for frequency, colorbar for magnitude) with perturbed parameters as a function of parameter perturbation (y-axis). The colormap is clipped between 0 and 1. We observe that the lower frequencies are more robust to parameter perturbations than the higher frequencies.

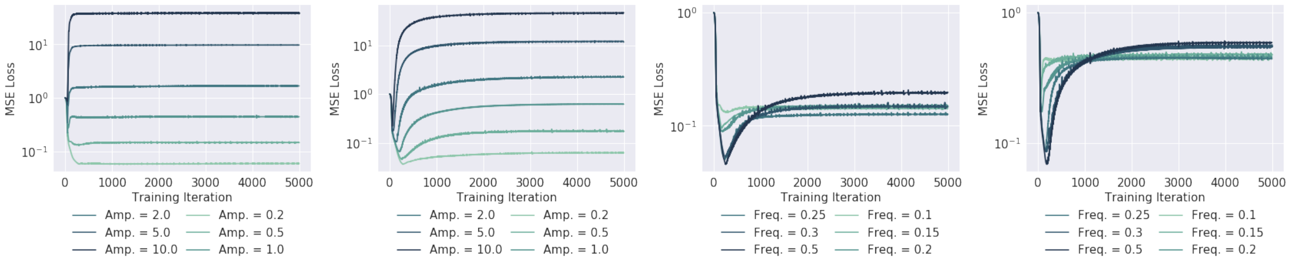
added to the training target. We find that the best validation performance on MNIST is particularly insensitive to the magnitude of high-frequency noise, yet it is adversely affected by low-frequency noise. We consider a target (binary) function $\tau_0 : X \to \{0, 1\}$ defined on the space $X = [0, 1]^{784}$ of MNIST inputs. Samples $\{\mathbf{x}_i, \tau_0(\mathbf{x}_i)\}_i$ form a subset of the MNIST dataset comprising samples $\mathbf{x}_i$ belonging to two classes. Let $\psi_k(\mathbf{x})$ be a *noise function*:

$$\psi_k(\mathbf{x}) = \sin(k\|\mathbf{x}\|) \tag{10}$$

corresponding to a *radial wave* defined on the 784-dimensional input space[11]. The final target function τ_k is then given by $\tau_k = \tau_0 + \beta\psi_k$, where β is the effective amplitude of the noise. We fit the same network as in Experiment 1 to the target τ_k with the MSE loss. In the first set of experiments, we ablate over k for a pair of fixed βs, while in the second set we ablate over β for a pair of fixed ks. In Figure 4, we show the respective validation loss curves, where the validation set is obtained by evaluating τ_0 on a separate subset of the data, i.e. $\{\mathbf{x}_j, \tau_0(\mathbf{x}_j)\}_j$. Figure 11 (in appendix A.3) shows the respective training curves.

Discussion. The profile of the loss curves varies significantly with the frequency of noise added to the target. In Figure 4a, we see that the validation performance is adversely affected by the amplitude of the low-frequency noise, whereas Figure 4b shows that the amplitude of high-

[11] The rationale behind using a radial wave is that it induces oscillations (simultaneously) along all spatial directions. Another viable option is to induce oscillations along the principle axes of the data: we have verified that the key trends of interest are preserved.



(a) $k = 0.1$ (b) $k = 1$ (c) $\beta = 0.5$ (d) $\beta = 1.$

Figure 4. (a,b,c,d): Validation curves for various settings of noise amplitude β and frequency k. Corresponding training curves can be found in Figure 11 in appendix A.3. **Gist**: Low frequency noise affects the network more than their high-frequency counterparts. Further, for high-frequency noise, one finds that the validation loss dips early in the training. Both these observations are explained by the fact that network readily fit lower frequencies, but learn higher frequencies later in the training.

frequency noise does not significantly affect the best validation score. This is explained by the fact that the network readily fits the noise signal if it is low frequency, whereas the higher frequency noise is only fit later in the training. In the latter case, the dip in validation score early in the training is when the network has learned the low frequency true target function τ_0; the remainder of the training is spent learning the higher-frequencies in the training target τ, as we shall see in the next experiment. Figures 4c and 4d confirm that the dip in validation score exacerbates for increasing frequency of the noise. Further, we observe that for higher frequencies (e.g. $k = 0.5$), increasing the amplitude β does not significantly degrade the best performance at the dip, confirming that the network is fairly robust to the amplitude of high-frequency noise.

Finally, we note that the dip in validation score was also observed by Arpit et al. (2017) with i.i.d. noise[12] in a classification setting.

Experiment 4. To investigate the dip observed in Experiment 3, we now take a more direct approach by considering a generalized notion of frequency. To that end, we project the network function to the space spanned by the orthonormal eigenfunctions φ_n of the Gaussian RBF kernel (Braun et al., 2006). These eigenfunctions φ_n (sorted by decreasing eigenvalues) resemble sinusoids (Fasshauer, 2011), and the index n can be thought of as being a proxy for the frequency, as can be seen from Figure 6 (see Appendix A.4 for additional details and supporting plots). While we will call $\tilde{f}[n]$ as the spectrum of the function f, it should be understood as $\tilde{f}[n] = \langle f_{\mathcal{H}}, \varphi_n \rangle_{\mathcal{H}}$, where $f_{\mathcal{H}} \in \text{span}\{\varphi_n\}_n$ and $f_{\mathcal{H}}(\mathbf{x}_i) = f(\mathbf{x}_i)$ on the MNIST samples $\mathbf{x}_i \in X$. This allows us to define a noise function as:

$$\psi_\gamma(\mathbf{x}) = \sum_{n}^{N} \left(\frac{n}{N}\right)^{\gamma} \varphi_n(\mathbf{x}) \tag{11}$$

where N is the number of available samples and $\gamma = 2$. Like in Experiment 3, the target function is given by $\tau = \tau_0 + \beta\psi$, and the same network is trained to regress τ. Figure 5 shows the (generalized) spectrum τ and τ_0, and that of f as training progresses. Figure 13 (in appendix) shows the corresponding dip in validation loss, where the validation set is same as the training set but with true target function τ_0 instead of the noised target τ.

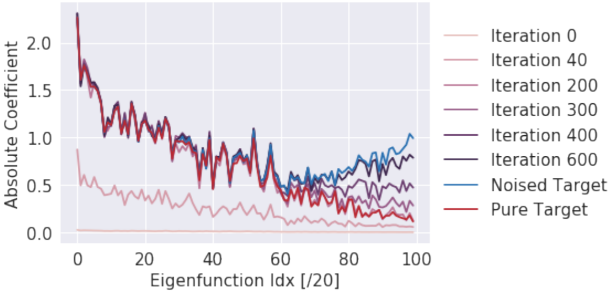


Figure 5. Spectrum of the network as it is trained on MNIST target with high-frequency noise (*Noised Target*). We see that the network fits the true target at around the 200th iteration, which is when the validation score dips (Figure 13 in appendix).

Discussion. From Figure 5, we learn that the drop in validation score observed in Figure 4 is exactly when the higher-frequencies of the noise signal are yet to be learned. As the network gradually learns the higher frequency eigenfunctions, the validation loss increases while the training loss continues to decrease. Thus these experiments show that the phenomenon of spectral bias persists on non-synthetic data and in high dimensional input spaces.

4. Not all Manifolds are Learned Equal

In this section, we investigate subtleties that arise when the data lies on a lower dimensional manifold embedded in the higher dimensional input space of the model. We find

[12] Recall that i.i.d. noise is white-noise, which has a constant Fourier spectrum magnitude in expectation, i.e. it also contains high-frequency components.

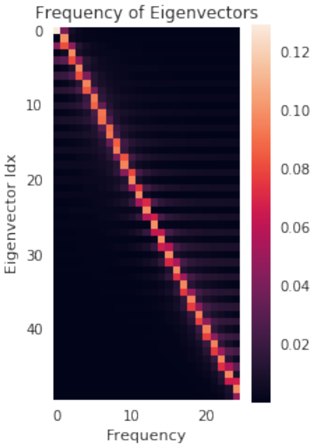


Figure 6. Spectrum (x-axis for frequency, colorbar for magnitude) of the n-th (y-axis) eigenvector of the Gaussian RBF kernel matrix $K_ij = k(\mathbf{x}_i, \mathbf{x}_j)$, where the sample set is $\{x_i \in [0,1]\}_{i=1}^{50}$ is $N = 50$ uniformly spaced points between 0 and 1 and k is the Gaussian RBF kernel function. **Gist:** The eigenfunctions with increasing n roughly correspond to sinusoids of increasing frequency. Refer to Appendix A.4 for more details.

that the *shape* of the data-manifold impacts the learnability of high frequencies in a non-trivial way. As we shall see, this is because low frequency functions in the input space may have high frequency components when restricted to lower dimensional manifolds of complex shapes. We demonstrate results in an illustrative minimal setting[13], free from unwanted confounding factors, and present a theoretical analysis of the phenomenon.

Manifold hypothesis. We consider the case where the data lies on a lower dimensional *data manifold* $\mathcal{M} \subset \mathbb{R}^d$ embedded in input space (Goodfellow et al., 2016), which we assume to be the image $\gamma([0,1]^m)$ of some injective mapping $\gamma : [0,1]^m \to \mathbb{R}^d$ defined on a lower dimensional latent space $[0,1]^m$. Under this hypothesis and in the context of the standard regression problem, a target function $\tau : \mathcal{M} \to \mathbb{R}$ defined on the data manifold can be identified with a function $\lambda = \tau \circ \gamma$ defined on the latent space. Regressing τ is therefore equivalent to finding $f : \mathbb{R}^d \to \mathbb{R}$ such that $f \circ \gamma$ matches λ. Further, assuming that the data probability distribution μ supported on $\mathcal{M}$ is induced by γ from the uniform distribution U in the latent space $[0,1]^m$, the mean square error can be expressed as:

$$\begin{aligned}&\mathrm{MSE}_{\mu}^{(\mathbf{x})}[f,\tau] = \mathbb{E}_{\mathbf{x}\sim\mu}|f(\mathbf{x}) - \tau(\mathbf{x})|^2 = \\ &\mathbb{E}_{\mathbf{z}\sim U}|(f(\gamma(\mathbf{z})) - \lambda(\mathbf{z})|^2 = \mathrm{MSE}_{U}^{(\mathbf{z})}[f\circ\gamma,\lambda] \qquad (12)\end{aligned}$$

Observe that there is a vast space of degenerate solutions f that minimize the mean squared error – namely all functions on $\mathbb{R}^d$ that yield the same function when restricted to the data manifold $\mathcal{M}$.

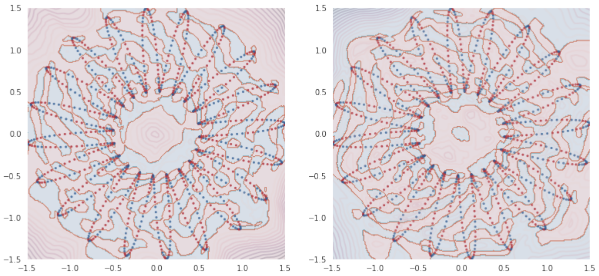


Figure 7. Functions learned by two identical networks (up to initialization) to classify the binarized value of a sine wave of frequency $k = 200$ defined on a $\gamma_{L=20}$ manifold. Both yield close to perfect accuracy for the samples defined on the manifold (scatter plot), yet they differ significantly elsewhere. The shaded regions show the predicted class (Red or Blue) whereas contours show the confidence (absolute value of logits).

Our findings from the previous section suggest that neural networks are biased towards expressing a particular subset of such solutions, namely those that are low frequency. It is also worth noting that there exist methods that restrict the space of solutions: notably adversarial training (Goodfellow et al., 2014) and Mixup (Zhang et al., 2017b).

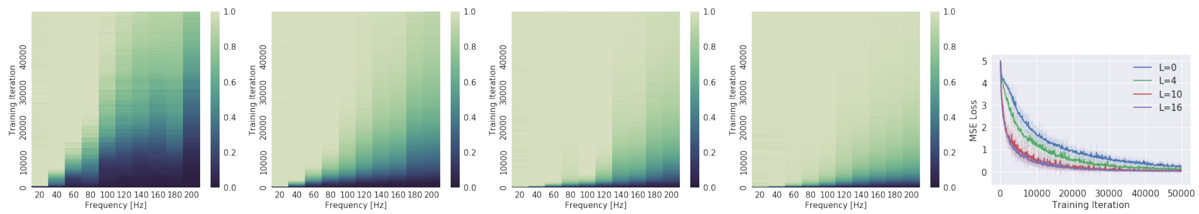
Experimental set up. The experimental setting is designed to afford control over both the shape of the data manifold and the target function defined on it. We will consider the family of curves in $\mathbb{R}^2$ generated by mappings $\gamma_L : [0,1] \to \mathbb{R}^2$ given by

$$\begin{aligned}\gamma_L(z) =& R_L(z)(\cos(2\pi z), \sin(2\pi z)) \\ \text{where } R_L(z) =& 1 + \frac{1}{2}\sin(2\pi L z) \qquad (13)\end{aligned}$$

Here, $\gamma_L([0,1])$ defines the data-manifold and corresponds to a flower-shaped curve with L petals, or a unit circle when $L = 0$ (see e.g. Fig 7). Given a signal $\lambda : [0,1] \to \mathbb{R}$ defined on the latent space $[0,1]$, the task entails learning a network $f : \mathbb{R}^2 \to \mathbb{R}$ such that $f \circ \gamma_L$ matches the signal λ.

Experiment 5. The set-up is similar to that of Experiment 1, and λ is as defined in Eqn. 8 with frequencies $\kappa = (20, 40, ..., 180, 200)$, and amplitudes $A_i = 1\,\forall\, i$. The model f is trained on the dataset $\{\gamma_L(z_i), \lambda(z_i)\}_{i=1}^{N}$ with $N = 1000$ uniformly spaced samples z_i between 0 and 1. The spectrum of $f \circ \gamma_L$ in expectation over $\varphi_i \sim U(0, 2\pi)$ is monitored as training progresses, and the result is shown in Fig 8 for various L. Fig 8e shows the corresponding mean squared error curves. More experimental details in appendix A.2.

[13] We include additional experiments on MNIST and CIFAR-10 in appendices A.6 and A.7.



(a) $L = 0$ (b) $L = 4$ (c) $L = 10$ (d) $L = 16$ (e) Loss curves

Figure 8. (a,b,c,d): Evolution of the network spectrum (x-axis for frequency, colorbar for magnitude) during training (y-axis) for the same target functions defined on manifolds γ_L for various L. Since the target function has amplitudes $A_i = 1$ for all frequencies k_i plotted, the colorbar is clipped between 0 and 1. (e): Corresponding learning curves. **Gist**: Some manifolds (here with larger L) make it easier for the network to learn higher frequencies than others.

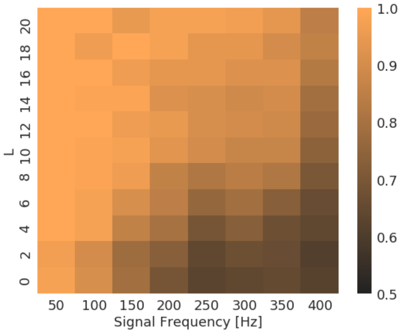


Figure 9. Heatmap of training accuracies of a network trained to predict the binarized value of a sine wave of given frequency (x-axis) defined on γ_L for various L (y-axis).

The results demonstrate a clear attenuation of the spectral bias as L grows. Moreover, Fig 8e suggests that the larger the L, the easier the learning task.

Experiment 6. Here, we adapt the setting of Experiment 5 to binary classification by simply thresholding the function λ at 0.5 to obtain a binary target signal. To simplify visualization, we only use signals with a single frequency mode k, such that $\lambda(z) = \sin(2\pi kz + \varphi)$. We train the same network on the resulting classification task with cross-entropy loss[14] for $k \in \{50, 100, ..., 350, 400\}$ and $L \in \{0, 2, ..., 18, 20\}$. The heatmap in Fig 9 shows the classification accuracy for each (k, L) pair. Fig 7 shows visualizations of the functions learned by the same network, trained on $(k, L) = (200, 20)$ under identical conditions up to random initialization.

Observe that increasing L (i.e. going up a column in Fig 9) results in better (classification) performance for the same target signal. This is the same behaviour as we observed in Experiment 5 (Fig 8a-d), but now with binary cross-entropy loss instead of the MSE.

Discussion. These experiments hint towards a rich interaction between the shape of the manifold and the effective difficulty of the learning task. The key mechanism underlying this phenomenon (as we formalize below) is that the relationship between frequency spectrum of the network f and that of the fit $f \circ \gamma_L$ is mediated by the embedding map γ_L. In particular, we argue that a given signal defined on the manifold is easier to fit when the coordinate functions of the manifold embedding itself has high frequency components. Thus, in our experimental setting, the same signal embedded in a flower with more petals can be captured with lower frequencies of the network.

To understand this mathematically, we address the following questions: given a target function λ, how small can the frequencies of a solution f be such that $f \circ \gamma = \lambda$? And further, how does this relate to the geometry of the data-manifold $\mathcal{M}$ induced by γ? To find out, we write the Fourier transform of the composite function,

$$\widetilde{(f \circ \gamma)}(\mathbf{l}) = \int \mathbf{dk} \tilde{f}(\mathbf{k}) P_\gamma(\mathbf{l}, \mathbf{k}) \tag{14}$$

$$\text{where } P_\gamma(\mathbf{l}, \mathbf{k}) = \int_{[0,1]^m} \mathbf{dz}\, e^{i(\mathbf{k} \cdot \gamma(\mathbf{z}) - \mathbf{l} \cdot \mathbf{z})}$$

The kernel P_γ depends on only γ and elegantly encodes the correspondence between frequencies $\mathbf{k} \in \mathbb{R}^d$ in input space and frequencies $\mathbf{l} \in \mathbb{R}^m$ in the latent space $[0, 1]^m$. Following a procedure from Bergner et al., we can further investigate the behaviour of the kernel in the regime where the stationary phase approximation is applicable, i.e. when $l^2 + k^2 \to \infty$ (cf. section 3.2. of Bergner et al.). In this regime, the integral P_γ is dominated by critical points $\bar{\mathbf{z}}$ of its phase, which satisfy

$$\mathbf{l} = J_\gamma(\bar{\mathbf{z}})\,\mathbf{k} \tag{15}$$

where $J_\gamma(\mathbf{z})_{ij} = \nabla_i \gamma_j(\mathbf{z})$ is the $m \times d$ Jacobian matrix of γ. Non-zero values of the kernel correspond to pairs $(\mathbf{l}, \mathbf{k})$ such that Eqn 15 has a solution. Further, given that the components of γ (i.e. its coordinate functions) are defined

[14] We use Pytorch's `BCEWithLogitsLoss`. Internally, it takes a sigmoid of the network's output (the logits) before evaluating the cross-entropy.

on an interval $[0,1]^m$, one can use their Fourier series representation together with Eqn 15 to obtain a condition on their frequencies (shown in appendix C.7). More precisely, we find that the i-th component of the RHS in Eqn 15 is proportional to $\mathbf{p}\tilde{\gamma}_i[\mathbf{p}]k_i$ where $\mathbf{p} \in \mathbb{Z}^m$ is the frequency of the coordinate function γ_i. This yields that we can get arbitrarily large frequencies l_i if $\tilde{\gamma}_i[\mathbf{p}]$ is large[15] enough for large $\mathbf{p}$, even when k_i is fixed.

This is precisely what Experiments 5 and 6 demonstrate in a minimal setting. From Eqn 13, observe that the coordinate functions have a frequency mode at L. For increasing L, it is apparent that the frequency magnitudes l (in the latent space) that can be expressed with the same frequency k (in the input space) increases with increasing L. This allows the remarkable interpretation that the neural network function can express large frequencies on a manifold (l) with smaller frequencies w.r.t its input domain (k), provided that the coordinate functions of the data manifold embedding itself has high-frequency components.

5. Related Work

A number of works have focused on showing that neural networks are capable of approximating arbitrarily complex functions. Hornik et al. (1989); Cybenko (1989); Leshno et al. (1993) have shown that neural networks can be universal approximators when given sufficient width; more recently, Lu et al. (2017) proved that this property holds also for width-bounded networks. Montufar et al. (2014) showed that the number of linear regions of deep ReLU networks grows polynomially with width and exponentially with depth; Raghu et al. (2016) generalized this result and provided asymptotically tight bounds. There have been various results of the benefits of depth for efficient approximation (Poole et al., 2016; Telgarsky, 2016; Eldan & Shamir, 2016). These analysis on the expressive power of deep neural networks can in part explain why over-parameterized networks can perfectly learn random input-output mappings (Zhang et al., 2017a).

Our work more directly follows the line of research on implicit regularization in neural networks trained by gradient descent (Neyshabur et al., 2014; Soudry et al., 2017; Poggio et al., 2018; Neyshabur et al., 2017). In fact, while our Fourier analysis of deep ReLU networks also reflects the width and depth dependence of their expressivity, we focused on showing a learning bias of these networks towards simple functions with dominant lower frequency components. We view our results as a first step towards formalizing the findings of Arpit et al. (2017), where it is empirically shown that deep networks prioritize learning simple patterns of the data during training.

A few other works studied neural networks through the lens of harmonic analysis. For example, Candès (1999) used the ridgelet transform to build constructive procedures for approximating a given function by neural networks, in the case of oscillatory activation functions. This approach has been recently generalized to unbounded activation functions by Sonoda & Murata (2017). Eldan & Shamir (2016) use insights on the support of the Fourier spectrum of two-layer networks to derive a worse-case depth-separation result. Barron (1993) makes use of Fourier space properties of the target function to derive an architecture-dependent approximation bound. In a concurrent and independent work, Xu et al. (2018) make the same observation that lower frequencies are learned first. The subsequent work by Xu (2018) proposes a theoretical analysis of the phenomenon in the case of 2-layer networks with sigmoid activation, based on the spectrum of the sigmoid function.

In light of our findings, it is worth comparing the case of neural networks and other popular algorithms such that kernel machines (KM) and K-nearest neighbor classifiers. We refer to the Appendix E for a detailed discussion and references. In summary, our discussion there suggests that 1. DNNs strike a good balance between function smoothness and expressivity/parameter-efficiency compared with KM; 2. DNNs learn a smoother function compared with KNNs since the spectrum of the DNN decays faster compared with KNNs in the experiments shown there.

6. Conclusion

We studied deep ReLU networks through the lens of Fourier analysis. Several conclusions can be drawn from our analysis. While neural networks can approximate arbitrary functions, we find that they favour *low frequency* ones – hence they exhibit a bias towards smooth functions – a phenomenon that we called *spectral bias*. We also illustrated how the geometry of the data manifold impacts expressivity in a non-trivial way, as high frequency functions defined on complex manifolds can be expressed by lower frequency network functions defined in input space.

We view future work that explore the properties of neural networks in Fourier domain as promising. For example, the Fourier transform affords a natural way of measuring how fast a function can change within a small neighborhood in its input domain; as such, it is a strong candidate for quantifying and analyzing the *sensitivity* of a model – which in turn provides a natural measure of complexity (Novak et al., 2018). We hope to encourage more research in this direction.

[15] Consider that the data-domain is bounded, implying that $\tilde{\gamma}$ cannot be arbitrarily scaled.

Acknowledgements

The authors would like to thank Joan Bruna, Rémi Le Priol, Vikram Voleti, Ullrich Köthe, Steffen Wolf, Lorenzo Cerrone, Sebastian Damrich and the anonymous reviewers for their valuable feedback.

References

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.

Arpit, D., Jastrzebski, S., Ballas, N., Krueger, D., Bengio, E., Kanwal, M. S., Maharaj, T., Fischer, A., Courville, A., Bengio, Y., et al. A closer look at memorization in deep networks. *arXiv preprint arXiv:1706.05394*, 2017.

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

Bengio, Y. et al. Learning deep architectures for ai. *Foundations and trends® in Machine Learning*, 2(1):1–127, 2009.

Bergner, S., Möller, T., Weiskopf, D., and Muraki, D. J. A spectral analysis of function concatenations and its implications for sampling in direct volume visualization.

Braun, M. L., Lange, T., and Buhmann, J. M. Model selection in kernel methods based on a spectral analysis of label information. In *Joint Pattern Recognition Symposium*, pp. 344–353. Springer, 2006.

Candès, E. J. Harmonic analysis of neural networks. *Applied and Computational Harmonic Analysis*, 6(2):197–218, 1999.

Cybenko, G. Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals, and Systems (MCSS)*, 2(4):303–314, 1989.

Devroye, L., Györfi, L., and Lugosi, G. Consistency of the k-nearest neighbor rule. In *A Probabilistic Theory of Pattern Recognition*, pp. 169–185. Springer, 1996.

Diaz, R., Le, Q.-N., and Robins, S. Fourier transforms of polytopes, solid angle sums, and discrete volume. *arXiv preprint arXiv:1602.08593*, 2016.

Draxler, F., Veschgini, K., Salmhofer, M., and Hamprecht, F. A. Essentially no barriers in neural network energy landscape. *arXiv preprint arXiv:1803.00885*, 2018.

Eldan, R. and Shamir, O. The power of depth for feedforward neural networks. In *Conference on Learning Theory*, pp. 907–940, 2016.

Fasshauer, G. E. Positive definite kernels: past, present and future. *Dolomite Research Notes on Approximation*, 4: 21–63, 2011.

Goodfellow, I., Bengio, Y., and Courville, A. *Deep Learning*. MIT Press, 2016. http://www.deeplearningbook.org.

Goodfellow, I. J., Shlens, J., and Szegedy, C. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

Hammer, B. and Gersmann, K. A note on the universal approximation capability of support vector machines. *Neural Processing Letters*, 17(1):43–53, 2003.

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

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Kolsbjerg, E. L., Groves, M. N., and Hammer, B. An automated nudged elastic band method. *The Journal of chemical physics*, 145(9):094107, 2016.

Leshno, M., Lin, V. Y., Pinkus, A., and Schocken, S. Multilayer feedforward networks with a nonpolynomial activation function can approximate any function. *Neural networks*, 6(6):861–867, 1993.

Lu, Z., Pu, H., Wang, F., Hu, Z., and Wang, L. The expressive power of neural networks: A view from the width. In *Advances in Neural Information Processing Systems*, pp. 6231–6239, 2017.

Ma, S. and Belkin, M. Diving into the shallows: a computational perspective on large-scale shallow learning. In *Advances in Neural Information Processing Systems*, pp. 3781–3790, 2017.

Miyato, T., Kataoka, T., Koyama, M., and Yoshida, Y. Spectral normalization for generative adversarial networks. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=B1QRgziT-.

Montufar, G. F., Pascanu, R., Cho, K., and Bengio, Y. On the number of linear regions of deep neural networks. In *Advances in neural information processing systems*, pp. 2924–2932, 2014.

Neyshabur, B., Tomioka, R., and Srebro, N. In search of the real inductive bias: On the role of implicit regularization in deep learning. *arXiv preprint arXiv:1412.6614*, 2014.

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

Novak, R., Bahri, Y., Abolafia, D. A., Pennington, J., and Sohl-Dickstein, J. Sensitivity and generalization in neural networks: an empirical study. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=HJC2SzZCW.

Poggio, T., Kawaguchi, K., Liao, Q., Miranda, B., Rosasco, L., Boix, X., Hidary, J., and Mhaskar, H. Theory of deep learning iii: the non-overfitting puzzle. Technical report, Technical report, CBMM memo 073, 2018.

Poole, B., Lahiri, S., Raghu, M., Sohl-Dickstein, J., and Ganguli, S. Exponential expressivity in deep neural networks through transient chaos. In Lee, D. D., Sugiyama, M., Luxburg, U. V., Guyon, I., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 29*, pp. 3360–3368. Curran Associates, Inc., 2016.

Raghu, M., Poole, B., Kleinberg, J., Ganguli, S., and Sohl-Dickstein, J. On the expressive power of deep neural networks. *arXiv preprint arXiv:1606.05336*, 2016.

Rasmussen, C. E. Gaussian processes in machine learning. In *Advanced lectures on machine learning*, pp. 63–71. Springer, 2004.

Serov, V. *Fourier series, Fourier transform and their applications to mathematical physics*. Springer, 2017.

Sonoda, S. and Murata, N. Neural network with unbounded activation functions is universal approximator. *Applied and Computational Harmonic Analysis*, 43(2):233–268, 2017.

Soudry, D., Hoffer, E., Nacson, M. S., Gunasekar, S., and Srebro, N. The implicit bias of gradient descent on separable data. *arXiv preprint arXiv:1710.10345*, 2017.

Spivak, M. *Calculus On Manifolds: A Modern Approach To Classical Theorems Of Advanced Calculus*. CRC press, 2018.

Telgarsky, M. Benefits of depth in neural networks. *Conference on Learning Theory (COLT), 2016*, 2016.

Xu, Z. J. Understanding training and generalization in deep learning by fourier analysis. *arXiv preprint arXiv:1808.04295*, 2018.

Xu, Z.-Q. J., Zhang, Y., and Xiao, Y. Training behavior of deep neural network in frequency domain. *arXiv preprint arXiv:1807.01251*, 2018.

Zhang, C., Bengio, S., Hardt, M., Recht, B., and Vinyals, O. Understanding deep learning requires rethinking generalization. *International Conference on Learning Representations (ICLR)*, 2017a.

Zhang, H., Cisse, M., Dauphin, Y. N., and Lopez-Paz, D. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*, 2017b.

Zhang, L., Naitzat, G., and Lim, L.-H. Tropical geometry of deep neural networks. *arXiv preprint arXiv:1805.07091*, 2018.