10 Appendix

10.1 Proof of Theorem 5.1

Proof. The evolution in parameter space is described by the differential equation

$$\frac{d}{dt}\theta(t) = -\frac{1}{n}\sum_{i=1}^{n} D_\theta g_{\theta(t)}(x_i)^T \nabla c(g_{\theta(t)}(x_i), y_i).$$

The evolution of the corresponding function $g_{\theta(t)}$ is given by pushing this differential equation forward to function space by acting on both sides with the derivative $D_\theta g_{\theta(t)}(x)$:

$$\begin{aligned}\frac{d}{dt}g_{\theta(t)}(x) = D_\theta g_{\theta(t)}(x)\frac{d}{dt}\theta(t) &= -\frac{1}{n}\sum_{i=1}^{n} D_\theta g_{\theta(t)}(x)\, D_\theta g_{\theta(t)}(x_i)^T \nabla c(g_{\theta(t)}(x_i), y_i)\\ &= -\frac{1}{n}\sum_{i=1}^{n} \mathcal{K}(\theta, x, x_i)\nabla c(g_{\theta(t)}(x_i), y_i),\end{aligned}$$

where $\mathcal{K}$ the extension of the tangent kernel associated to f_θ by zero outside of the compact neighbourhood K of the data manifold, i.e.

$$\mathcal{K}(\theta, x, x') = \begin{cases} D_\theta f_\theta(x)\, D_\theta f_\theta(x')^T, & \text{if } x, x' \in K \\ 0, & \text{otherwise.}\end{cases}$$

The evolution equation for $\mathcal{F}[g_{\theta(t)}]$ follows easily from the Liebniz integral rule:

$$\frac{d}{dt}\mathcal{F}[g_{\theta(t)}] = \mathcal{F}\left[\frac{d}{dt}g_{\theta(t)}\right].$$

Now, by our hypothesis on f_θ that $x \mapsto D_\theta f_\theta(x)$ is bounded over compact sets, one has that $x \mapsto \mathcal{K}(\theta, x, x_i)$ is L^1 for each i, hence that $\frac{d}{dt}g_{\theta(t)}$ is an L^1 function. By the Riemann-Lebesgue lemma its Fourier transform vanishes at infinity as stated. □

The same result can be argued for discrete-time gradient descent as follows. At a given time step T, the gradient update is given by the equation

$$\theta_{T+1} - \theta_T = -\frac{\eta}{n}\sum_{i=1}^{n} D_\theta f_{\theta_T}(x_i)^T \nabla c(f_{\theta_T}(x_i)),$$

where η is the step size. One wishes to show that the difference $x \mapsto f_{\theta_{T+1}}(x) - f_{\theta_T}(x)$, extended by zero for x outside of the compact data manifold K, has Fourier transform vanishing at infinity. To first order in η, one can approximate this difference by

$$-\frac{\eta}{n}\sum_{i=1}^{n} D_\theta f_{\theta_T}(x) D_\theta f_{\theta_T}(x_i)^T \nabla c(f_{\theta_T}(x_i)),$$

again extended by zero for x outside of K. Spectral bias for gradient descent then follows (at least approximately, for $\eta << 1$) from the same Riemann-Lebesgue argument that we used for gradient flow.

10.2 Smoothness, generalization, and the the empirical risk minimization (ERM)

The ERM framework provides a well-established framework for studying the generalization in learnable models. The smoothness is a property which stems from the empirical risk minimization framework, and has been used since the earliest days of ML to quantify generalization (in regression). In summary, given a set of hypotheses (models) that minimizes the empirical risk (with training data), the ERM framework prefers a solution that minimizes the true risk (with respect to the actual data distribution) with a higher probability. When extra prior knowledge is unavailable on the true data distribution, ERM suggests that the best solution would be the one that minimizes the least complex solution that minimizes the empirical risk (under the realizability assumption). This can be primarily achieved using two regularization techniques: 1) regularizing the parameters of the model or 2) regularizing the function output itself. Popular regularizations on NNs, Lasso regression, Ridge regression etc. fall into the first category, and spline, polynomial regression with regularized derivatives fall into the second category Reinsch (1967); Kimeldorf and Wahba (1970); Craven and Wahba (1978); Kohler et al. (2002). A more recent example is Heiss et al. (2019). It should be noted that both these techniques lead to smooth solutions with bounded (higher-order) derivatives.

The intuition for this partially stems from the fact that reducing the bandwidth of a signal can be considered as minimizing noise, whereas noise corresponds to higher frequencies in natural signals. Almost every spectral-bias-related recent work also uses low-frequency solutions, hence solutions with bounded second-order derivatives, as a proxy for measuring generalization Xu et al. (2019a;b). A few application-specific examples would be recent Neural Radiance Field works Fridovich-Keil et al. (2022); Chen et al. (2022), where smooth (tri-linear) interpolation is used to generalize to unseen coordinates.

10.3 Initializing deep networks with higher bandwidths

Initializing deep classification networks – that consume high dimensional inputs such as images – such that they have higher bandwidths is not straightforward. Therefore, we explore alternative ways to initialize networks with higher bandwidths in low-dimensional settings, and extrapolate the learned insights to higher dimensions.

For all the experiments, we consider a fully connected 4-layer ReLU network with 1-dimensional inputs. First, we sample a set of values from white Gaussian noise, and train the network with these target values using MSE loss. In the second experiment, we threshold the sampled values to obtain a set of binary labels, and then train the network with binary cross-entropy loss. For the third experiment, we use a network with four outputs. Then, we separate the sampled values into four bins, and obtain four labels. Then, we train the network with cross-entropy loss. We compute the Fourier spectra of each of the trained networks after convergence. The results are shown in Fig.5.

As depicted, we can use mean squared error (MSE) or cross-entropy (CE) loss along with random labels to initialize the networks with higher bandwidth. However, we observed that, in practice, deep networks take an infeasible amount of time to converge with the MSE loss. Therefore, we use cross-entropy loss with random labels to initialize the networks in image classification settings.

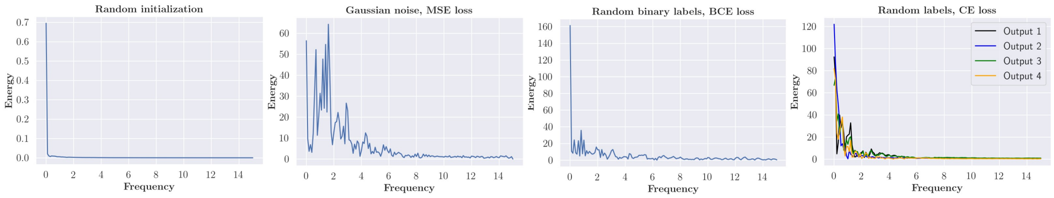

Figure 5: We visualize the spectra of networks after training them with different loss functions and label sampling schemes (the rightmost three plots). All shown methods are able to obtain higher bandwidths than random initialization (leftmost plot). Note that the scale in the $y-$axis is different for each plot. However, in practice, deep classification networks take an infeasible amount of time to converge with MSE loss. Hence, we chose random labels with cross-entropy loss to initialize the deep classification networks with higher bandwidths.

In order to verify that training with random labels indeed induces higher bandwidths on deep classification networks, we visualize the histograms of their first order gradients of the averaged outputs w.r.t. the inputs. It is straightforward to show that (similar to second-order gradients) higher first-order gradients lead to higher bandwidth. For simplicity, consider a function $f : \mathbb{R} \rightarrow \mathbb{R}$. Then,

$$f(x) = \int_{\infty}^{\infty} \hat{f}(k) e^{2\pi i k x} dk$$

It follows that,

$$|\frac{df(x)}{dx}| = |2\pi i \int_{\infty}^{\infty} k\hat{f}(k) e^{2\pi i k x} dk| \tag{5}$$

$$\leq |2\pi| \int_{\infty}^{\infty} |k\hat{f}(k)| dk. \tag{6}$$

Therefore,

$$\max_{x \in \epsilon} |\frac{df(x)}{dx}| \leq |2\pi| \int_{\infty}^{\infty} |k\hat{f}(k)| dk. \tag{7}$$

This conclusion can be directly extrapolated to higher-dimensional inputs, where the Fourier transform is also high dimensional. Hence, we feed a batch of images to the networks, and calculate the gradients of the averaged output layer with respect to the input image pixels. Then, we plot the histograms of the gradients (Fig. 6). As illustrated, training with random labels induces higher gradients, and thus, higher bandwidth. Table. 2 compares generalization of deep networks on ImageNet.

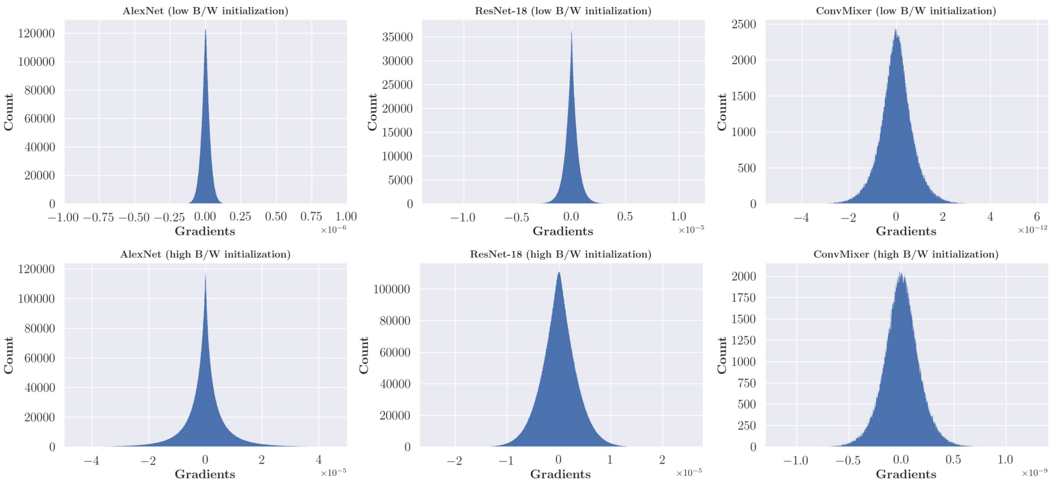

Figure 6: The histograms of the first-order gradients of the outputs with respect to the inputs (a batch of training images) are shown. Low and high bandwidth initializations correspond to Xavier initialization and pre-training with random labels, respectively. Not that the $x-$axis scales are different in each plot. As depicted, training with random labels leads to higher gradients, validating that it indeed leads to higher bandwidths.

ImageNet				
Model	Random initialization		High B/W initialization	
	Train accuracy	Test accuracy	Train accuracy	Test accuracy
VGG16	100%	68.19%	100%	55.48%
ResNet-18	100%	66.93%	100%	49.17%
ConvMixer	100%	74.19%	100%	45.68%

Table 2: **Generalization of deep networks in classification over ImageNet.** When the models are initialized with higher bandwidths (pre-trained on random labels), the test accuracy drops. We do not use data augmentation in these experiments. We only use three models for this experiment due to the extensive resource usage when training on random labels over ImageNet.

10.4 Convergence-decay rates of frequencies matter for generalization

Earlier, we showed that although all neural networks admit spectral bias, the convergence-decay rates of frequencies change across network types and initialization schemes. Below, we show that these decay rates play an essential role in generalization.

We use a Gaussian network for this experiment. We initialize two instances of the network by 1) using a weight distribution $\mathcal{N}(0, 0.03)$, and 2) pre-training the network on a DC signal. In both instances, the network has low bandwidth. Then, we train the network on sparse training data sampled from $3\sin(0.4\pi x) + 5\sin(0.2\pi x)$. The results are shown in Fig. 7. Observe that although both networks start from low bandwidth, they exhibit different generalization properties. This is because, having a lower convergence-decay hinders smooth interpolations even in cases where the networks have low initial bandwidth. This is expected, since then, the optimization will begin to affect the higher frequencies before the lower frequencies are converged.

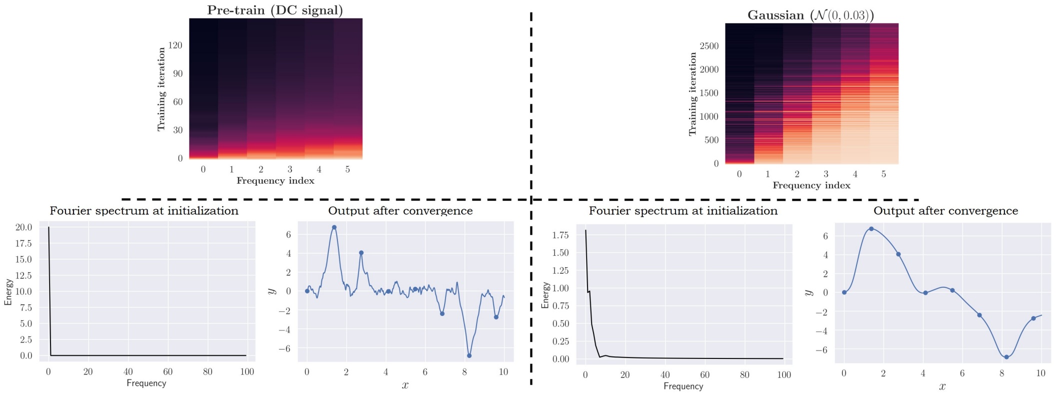

Figure 7: **The effect of convergence-decay rate of frequencies on generalization.** *Left block*: We pre-train a Gaussian network on a DC signal to obtain low initial bandwidth. Nevertheless, the network still converges to a non-smooth solution. *Right block*: The Gaussian network is initialized using a random Gaussian distribution ($\mathcal{N}(0, 0.03)$). This method also leads to lower bandwidth. However, in this scenario, the network is able to converge to a smooth solution. At the top, the convergence of frequency components – starting from the corresponding initialization – is shown when training on a signal $g(x) = \sum_{n=1}^{6} \sin(10\pi n x)$. Note that a lower convergence decay rate leads to bad generalization.

To further verify this, we conduct another experiment; see Fig. 8.

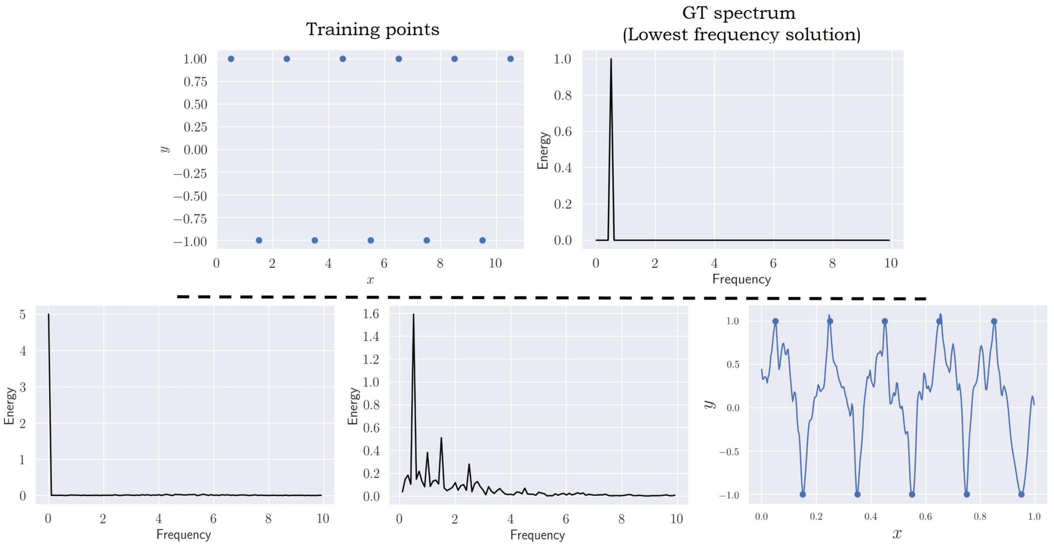

Figure 8: The top block shows sparsely sampled training points from $\sin(\pi x)$ and the corresponding lowest frequency solution that fits the training data. The bottom block shows the spectra of a Gaussian network initialized by pre-training on a DC signal. Even though the network adds a spike at the lowest frequency solution, higher frequencies are also added to the spectrum due to the low convergence-decay rate. This results in a non-smooth interpolation.

10.5 Analyzing the loss landscapes

The flat minima conjecture has been studied since the early work of Hochreiter and Schmidhuber (1994) and Hochreiter and Schmidhuber (1997). More recently, empirical works showed that the generalization of deep networks is related to the flatness of the minima it is converged to during training (Chaudhari et al., 2019; Keskar et al., 2016). In order to measure the flatness of loss landscapes, different metrics have been proposed (Tsuzuku et al., 2020; Rangamani et al., 2019; Hochreiter and Schmidhuber, 1994; 1997). In particular, Chaudhari et al. (2019) and Keskar et al. (2016) showed that minima with low Hessian spectral norm generalize better. In this paper also, we use Hessian-related metrics to measure the flatness. Since the spectral norm alone is not ideal for analyzing the loss landscape of models with a large number of parameters, we also compute the trace and the expected eigenvalue of the Hessian. For computing the Hessian, we use the library provided by Yao et al. (2020). Fig 9 and Table 3 depict a comparison of loss landscapes in several deep models. Note that our proposed high B/W initialization scheme provides an ideal platform to compare the loss landscapes with different generalization properties.

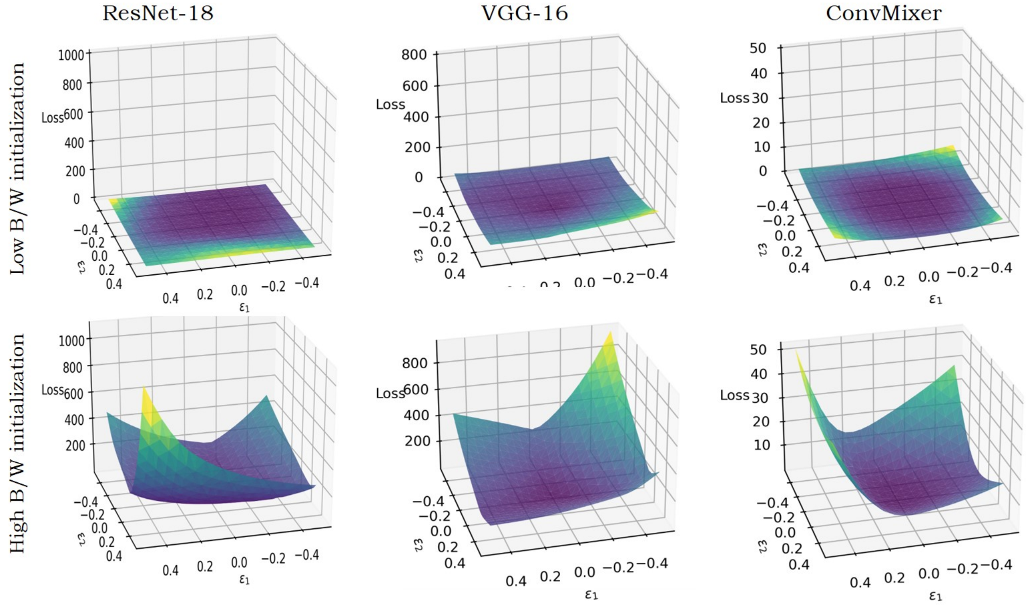

Figure 9: **Loss landscapes of deep networks trained on CIFAR10.** The proposed high B/W initialization scheme provides an ideal platform to compare the flatness of minima with different generalization properties. Note that ReLU networks exhibit behaviour consistent with the flat minima conjecture.

Model	Hessian-trace	Spectral norm
ResNet-18 (low B/W)	13560.76	2805.47
ResNet-18 (high B/W)	28614.19	4121.36
VGG-16 (low B/W)	10102.51	1112.07
VGG-16 (high B/W)	14483.90	3214.57
ConvMixer (low B/W)	0.3242	0.028
ConvMixer (high B/W)	3.49	0.445

Table 3: Quantitative comparison of the flatness of minima in deep networks. Note that Note that ReLU networks exhibit behaviour consistent with the flat minima conjecture.

Model	Initialization	Hessian trace	$\mathbb{E}[\epsilon]$	Spectral norm
ReLU	High B/W	134213.36	0.95	257875.23
ReLU	Low B/W	31110.73	0.04	49781.58
Gaussian	High B/W	40478.82	0.21	12596.89
Gaussian	Low B/W	59447.46	0.32	26519.66

Table 4: The trace, expected eigenvalue ($\mathbb{E}[\epsilon)$], and the spectral norm of the loss-Hessian are shown (averaged over 20 signals). Higher values indicate a sharper minimum. As illustrated, while the ReLU network obeys the flat minima conjecture, the Gaussian network behaves oppositely.