A NEW INITIALIZATION TO CONTROL GRADIENTS IN SINUSOIDAL NEURAL NETWORK

Anonymous authors
Paper under double-blind review

ABSTRACT

Proper initialization strategy is of primary importance to mitigate gradient explosion or vanishing when training neural networks. Yet, the impact of initialization parameters still lacks a precise theoretical understanding for several well-established architectures. Here, we propose a new initialization for networks with sinusoidal activation functions such as SIREN, focusing on gradients control, their scaling with network depth, their impact on training and on generalization. To achieve this, we identify a closed-form expression for the initialization of the parameters, differing from the original SIREN scheme. This expression is derived from fixed points obtained through the convergence of pre-activation distribution and the variance of Jacobian sequences. Controlling gradients prevents the emergence of inappropriate frequencies during estimation, thereby improving generalization. We further show that this initialization strongly influences training dynamics through the Neural Tangent Kernel framework (NTK). Finally, we benchmark SIREN with the proposed initialization against the original scheme and other baselines on function fitting and image reconstruction. The new initialization consistently outperforms state-of-the-art methods.

1 INTRODUCTION

1.1 CONTEXT AND MOTIVATION

Implicit neural representations (INRs) have become a prevalent tool for approximating functions in diverse applications, including signal encoding (Strümpler et al., 2022; Dupont et al., 2021), signal reconstruction (Park et al., 2019; Mildenhall et al., 2020), and solutions of partial differential equations (PDEs) (Raissi et al., 2019). A central challenge in these neural approximations is to recover the frequency spectrum of a target signal within reasonable training time and from limited data. In this context, standard multi-layer perceptrons (MLPs) used for INRs often suffer from *spectral bias*, whereby low-frequency components are preferentially learned compared to high-frequency details (Rahaman et al., 2019; Li et al., 2024). This bias can hinder performance, either by slowing training or by reducing precision, when the signal of interest contains significant high-frequency content. To mitigate this issue, several architectures have been proposed, such as positional encoding (Tancik et al., 2020) or networks with sinusoidal activation functions (SIREN, Sitzmann et al. (2020)), which enable faster learning of high-frequency components. However, increasing network depth in these methods has been empirically observed to introduce in the reconstructed function spurious high-frequency components absent from the target one (see, e.g., Ma et al. (2025)), leading to noisy representations and degraded generalization (i.e., the ability to interpolate the signal correctly).

In this work, we propose an initialization strategy for SIREN that bypasses two opposing pitfalls: (i) slow training and poor recovery of high-frequency details due to spectral bias in standard MLPs, and (ii) rapid training in deeper SIREN, which comes at the cost of spurious high-frequency artifacts and degraded generalization. Finding the right balance between these two extremes corresponds to locating the frontier between vanishing-gradient and exploding-gradient regimes. Operating in this regime, where gradients remain stable, is often referred to as computing at the edge of chaos (Yang & Schoenholz, 2017; Seleznova & Kutyniok, 2022), a concept from dynamical systems theory (Kelso et al., 1986; Langton, 1986). Building on this idea, we introduce an explicit initialization scheme for SIREN. Our method ensures that gradients neither

vanish nor explode with depth, allowing the network to capture high-frequency content without introducing spurious components, thereby enabling both stable and expressive learning dynamics. To better understand the critical role of the initialization in INRs, we complement our theoretical analysis with experiments based on the neural tangent kernel (NTK) framework (Jacot et al., 2018; Li et al., 2024). We find that controlling gradient propagation at initialization strongly influences the NTK eigenvalues, which determine the training speed of the frequencies associated with the corresponding eigenvectors. We further provide the scaling of these eigenvalues with network depth, showing that, under proper initialization, increasing depth accelerates learning while preserving a prescribed gradient distribution.

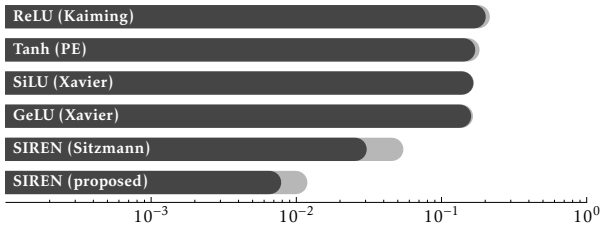

Figure 1: Generalization error over different problems averaged over different architecture depths for 1d, 2d and 3d multi-scaled function approximation. The results are displayed for different state-of-the-art architectures including the one proposed in this work (**SIREN Proposed**). See Appendix B.1 for details.

Beyond the NTK framework, our initialization prevents the degradation of deep neural network performance with increasing depth. We illustrate this property across several function fitting problem in Figure 1where a comparison of the performance of our initialization against the original `SIREN` scheme and other baselines is presented.

1.2 Related Work

Frequency representation. Our study will be based on the work of (Sitzmann et al., 2020), which introduced the `SIREN` architecture, a neural network with sinuosidal activation functions designed to effectively learn high-frequency functions by using a tunable parameter w_0 that controls the frequency range of the network. Architecturally, this approach is closely related to positional-encoding and Random Fourier feature, which also address the challenge of learning high-frequency signals (Tancik et al., 2020; Wang et al., 2021). However, `SIREN` requires careful tuning of w_0 depending on both the network architecture and the dataset (de Avila Belbute-Peres & Kolter, 2023). Moreover, the effect of network depth on performance remains poorly understood and has so far been studied mostly through empirical and observational analyses (Cai et al., 2024; Tancik et al., 2020). To the best of our knowledge, there is currently no work connecting theoretical gradient scaling with depth to the performance of such architectures.

Neural Tangent Kernel. The NTK framework provides a theoretical foundation for understanding the training dynamics of neural networks, and how the initialization properties affect the learning process (Jacot et al., 2018; Li et al., 2024). Some works have already focused on the frequency learning aspect of the NTK, either for the Fourier Features (Wang et al., 2021) or the `SIREN` architecture (de Avila Belbute-Peres & Kolter, 2023). These works have shown how the network architecture can be tailored to bypass the spectral bias. However they did not provide a full understanding of the impact of network depth on the networks properties and did not tackle the vanishing or exploding gradient impact on the learning dynamics.

Initialization. Our focus on initialization is closely related to the work of Glorot & Bengio (2010) and He et al. (2015), which introduced the now widely used Xavier and Kaiming initialization methods, respectively. Both approaches aim to maintain stable activation and gradient distributions across layers. Xavier initialization was developed for saturating nonlinearities such as hyperbolic tangent (`Tanh`), and is motivated by theoretical insights into variance preservation, though its derivation assumes an approximate linearization. Kaiming initialization was later introduced for rectified linear units (`ReLU`), which allows for exact variance calculations. Although commonly applied to smoother activation functions such as `GeLU` or `SiLU`, its theoretical justification in those cases is only approximate. In the context of SIREN, tailored initializations have been proposed (Sitzmann et al., 2020; de Avila Belbute-Peres & Kolter, 2023) to control the distribution of pre-activations layer by layer. However, these initializations are only approximate and fail to offer stability guar-

antees for deep SIREN architectures, where gradient growth remains uncontrolled, as we shall see later in this work.

Edge of Chaos. Previous works investigating the *Edge of Chaos*, which aims at controlling the correlation between neuron outputs, emphasize the importance of initialization for the stability of neural network statistics (including gradients). For traditional architectures and activation functions, Yang & Schoenholz (2017) showed that a careful control at initialization leads to improved training stability and performance. Some authors also investigated training at the edge of chaos in INR (Hayou et al., 2019; Seleznova & Kutyniok, 2022; Hayou et al., 2022), but did not address the case of sine activation functions.

1.3 Contributions

This works brings a deeper understanding over INR initialization for signal representation and training dynamic, with the following main contributions:

- An explicit derivation of the initialization for the SIREN architecture, which allows us to have an invariant distribution of the gradients across the layers. This is done by calculating the fixed point for the layer-wise gradient and the network output distribution, in the limit of infinite width and infinite depth.
- The understanding of the key concepts for controlled frequency learning, and how the initialization properties, through the NTK, shape the training dynamics of the network, leading to a controlled spectrum of the learned function.
- A series of experiments demonstrating the effectiveness of the proposed initialization method on multi-dimensional and multi-frequency function approximation.

Furthermore, we believe that the insights gained into the interplay between initialization, early training, and the vanishing/exploding gradient problem will be valuable beyond the specific case of INR with sinusoidal activations discussed in this paper.

2 Preliminaries

2.1 Generalities on Implicit representation of functions

Implicit neural representations have been introduced to find an approximation of a function $f : \Omega \mapsto \mathbb{R}^d$ from a dataset $\mathcal{D} = \{(\boldsymbol{x}_i, \boldsymbol{y}_i)_{i\in\mathbb{I}} \mid \boldsymbol{y}_i = f(\boldsymbol{x}_i)\}$. The goal is then to build a parametrized function $\Psi_\theta : \Omega \mapsto \mathbb{R}^d$. When this parametrized function is a neural network, it is commonly referred to as implicit neural representation (INR), Neural Fields (NerF), or Neural Implicit Functions.

In this work, we formally denote the involved neural network Ψ_θ, which can be written as the composition of L layers:

$$\Psi_\theta = h_{\theta_L} \circ \cdots \circ h_{\theta_1} \tag{1}$$

where each layer $\ell \in \{1, \dots, L\}$ is composed of n_ℓ neurons, parameterized by a set of parameters $\theta_\ell = (\mathbf{W}_\ell, \mathbf{b}_\ell)$ where $\mathbf{W}_\ell \in \mathbb{R}^{n_\ell \times n_{\ell-1}}$ are the weights and $\mathbf{b}_\ell \in \mathbb{R}^{n_\ell}$ the bias, and n_0 denotes the input dimension of the network. Each layer also relies on an activation function σ_ℓ applied element-wise. The ℓ-th layer is thus defined by

$$h_{\theta_\ell} = \sigma_\ell \odot \big(\mathbf{W}_\ell \cdot + \mathbf{b}_\ell\big). \tag{2}$$

For an input $\boldsymbol{x} \in \mathbb{R}^d$, the preactivation refers to

$$\mathbf{z}_\ell = \mathbf{W}_\ell \mathbf{h}_{\ell-1} + \mathbf{b}_\ell \qquad \text{where} \qquad \mathbf{h}_{\ell-1} = h_{\theta_{\ell-1}} \circ \ldots \circ h_{\theta_1}(\boldsymbol{x}). \tag{3}$$

The estimation of the parameters θ relies on the minimization of a loss $\mathcal{L}$ over a dataset $\mathcal{D} = \{(\boldsymbol{x}_i, \boldsymbol{y}_i)_{i\in\mathbb{I}}\}$:

$$\min_\theta \mathcal{L}(\theta) := \frac{1}{|\mathbb{I}|} \sum_{i\in\mathbb{I}} \|\Psi_\theta(\boldsymbol{x}_i) - \boldsymbol{y}_i\|^2. \tag{4}$$

The main challenges when considering INRs include selecting an appropriate architecture (i.e., parametrization and activation function), choosing a suitable initialization to insure output stability, and determining and efficient optimization strategy. In this work, we will focus on SIREN architectures (described in the next section). Regarding minimization strategy, we focus on gradient-based methods, leaving alternative minimization strategies outside the scope of our study.

2.2 Choice of the architecture

This work focuses on the so called SIREN architecture, which stands for Sinusoidal Representation Network and introduced by Sitzmann et al. (2020). SIREN is a particular instance of equations 1-2.

that can be written in the following way with a linear final layer:

$$\Psi_\theta(\boldsymbol{x}) = \mathbf{W}_L \sin\Big(\mathbf{W}_{L-1}\, \sin(\dots \sin(\mathbf{W}_1 \boldsymbol{x} + \mathbf{b}_1)) + \mathbf{b}_{L-1}\Big) + \mathbf{b}_L. \tag{5}$$

This architecture enables the estimation of natural frequency decompositions in a broad range of problems while ensuring differentiability. The latter property is particularly important for PDE-related applications, such as physics-informed neural networks, where accurate derivatives are often essential.

3 Weight initialization

In the original SIREN implementation (Sitzmann et al., 2020), the weights and biases were initialized as

$$\mathbf{W}_\ell \sim \begin{cases} \mathcal{U}\Big(-\frac{\omega_0}{n_0}, \frac{\omega_0}{n_0}\Big), & \ell = 1, \\ \mathcal{U}\Big(-\frac{\sqrt{6}}{\sqrt{N}}, \frac{\sqrt{6}}{\sqrt{N}}\Big), & \ell \in \{2, \dots, L\}, \end{cases} \qquad \mathbf{b}_\ell \sim \mathcal{U}\Big(-\tfrac{1}{\sqrt{N}}, \tfrac{1}{\sqrt{N}}\Big),\ \ell \in \{1, \dots, L\}, \tag{6}$$

where $N \equiv n_\ell$ is the number of neurons per hidden layer, assumed to be the same across all layers, $L-1$ is the number of hidden layers, and $\mathcal{U}$ denotes the uniform distribution. ω_0 is an important tunable parameter, originally chosen to be 30, but in practice it must be adjusted according to the network architecture and the Nyquist frequency of the signal to be reconstructed (de Avila Belbute-Peres & Kolter, 2023).

Sitzmann et al. (2020) argued that the pre-activation of the ℓ-th layer, defined in equation 3, follows the distribution $\mathbf{z}_\ell \sim \mathcal{N}(0, 1)$, when the network is initialized following equation 6. In this regime, most of the signal is sufficiently small to propagates through the quasi-linear range of the sine activation function, while still preserving a meaningful nonlinear contribution. This has been emphasized as a key feature of the SIREN architecture. However, the initialization choice relied on approximate computations, did not provide constraints on gradients, and it has been observed that estimation quality decreases in the large-depth limit under such initialization (Cai et al., 2024). To address this, we propose the refined initialization:

$$\mathbf{W}_\ell \sim \begin{cases} \mathcal{U}\Big(-\frac{\omega_0}{n_0}, \frac{\omega_0}{n_0}\Big), & \ell = 1, \\ \mathcal{U}\Big(-\frac{c_w}{\sqrt{N}}, \frac{c_w}{\sqrt{N}}\Big), & \ell \in \{2, \dots, L\}, \end{cases} \qquad \mathbf{b}_\ell \sim \mathcal{N}(0, c_b^2),\ \ell \in \{1, \dots, L\}, \tag{7}$$

with $\mathcal{N}(0, c_b^2)$ the normal distribution of zero mean and variance c_b^2. This initialization introduces two parameters, c_w and c_b, to be determined. Using explicit computations, we will demonstrate in next section that the choice

$$c_w = \sqrt{\tfrac{6}{1+e^{-2}}} \qquad \text{and} \qquad c_b = \tfrac{1}{\sqrt{3}}\, c_w e^{-1} \tag{8}$$

guarantees important stability properties of the network. To illustrate the interest of this new initialization, we show in Figure 2 that it leads to a stable network whatever the depth L compared to the standard SIREN architecture initialized with equations 7-8. Indeed, the classical SIREN initialization suffers from gradient explosion for a deeper architecture, visible as spurious, noisy high-frequency features in the estimated high-resolution image. Moreover, it leads to significant improvement in the model estimation with respect to other methods. For instance, it preserves sharp features and enables fast training, compared to ReLu with Kaiming intialization of SiLu with Xavier initialization, which still do not have converged after 10 000 epochs.

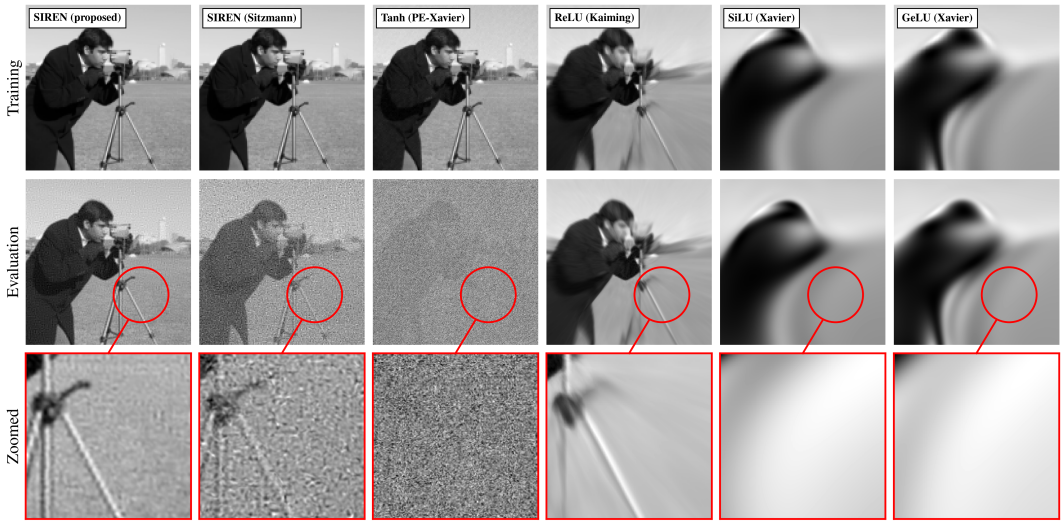

Figure 2: Comparison of several INR architectures and initializations on an image-fitting problem using an $L = 10$ hidden-layer neural network of width $N = 256$. We train the model on a set $(\boldsymbol{x}_i, y_i)_{i \in \mathbb{I}}$ where $\boldsymbol{x}_i$ is a location taken on a $|\mathbb{I}| = 128 \times 128$ uniformly spaced grid on $\Omega = [-1, 1]^2$ and y_i is the associated image value at this location. The top row shows the fitted 128×128 image. The middle row shows the estimation on an augmented resolution (512×512) to assess the model's generalization and the last row provides a zoom on part of the image. In all case, we use ADAM optimizer with learning rate 10^{-4} for 3000 epochs. The state-of-the-art architecture considered in this experiment are: `SIREN` (see Sitzmann et al. (2020)), `Tanh` with Positional Encoding (PE) (see Tancik et al. (2020)), `GeLU` (see Hendrycks & Gimpel (2016), `SiLU` (see Elfwing et al. (2018)) and the traditional `ReLU` (see Nair & Hinton (2010). To initialize these architectures we used for `SIREN` the previously discussed schemes and, for the others, two different schemes : **Xavier** (Glorot & Bengio, 2010) and **Kaiming** (He et al., 2015).

3.1 Pre-ctivation distribution

In the following, we derive the exact form of the pre-activation distribution in the limit of infinitely wide and deep neural networks, explicitly accounting for the influence of the bias term, which turns out to be crucial. More precisely, we show that, for any initialization in the parameter space (c_w, c_b), the pre-activation distribution converges to a fixed point.

Theorem 3.1 (Pre-activation distribution of `SIREN`)**.** *Considering `SIREN` network described in equation 5 where, for some $c_w, c_b \in \mathbb{R}^+$, and for every layer $\ell \in \{2, \dots, L\}$, the weight matrix $\mathbf{W}_\ell$ is initialized as a random matrix sampled from $\mathcal{U}(-c_w/\sqrt{N}, c_w/\sqrt{N})$, let $\mathbf{W}_1$ be sampled from $\mathcal{U}(-w_0/n_0, w_0/n_0)$, the bias $\mathbf{b}_\ell$ is initialized as a random vector sampled from $\mathcal{N}(0, c_b^2)$. Let $(\mathbf{z}_\ell)_{\ell \in \{1\dots,L\}}$ the pre-activation sequence defined in equation 3 and relying on an input $\boldsymbol{x} \in \mathbb{R}^{n_0}$. Then, in the limit of large N, the pre-activation sequence $(\mathbf{z}_\ell)_{\ell \in \mathbb{N}}$ converges in distribution to $\mathcal{N}(0, \sigma_a^2)$ with*

$$\sigma_a^2 = c_b^2 + \frac{c_w^2}{6} + \frac{1}{2}\mathcal{W}_0\left(-\frac{c_w^2}{3}e^{-\frac{c_w^2}{3} - 2c_b^2}\right). \tag{9}$$

Where $\mathcal{W}_0$ is the principal real branch of the Lambert function. Additionally, the sequence associated to the variance of the pre-activation $\left(\mathrm{Var}(\mathbf{z}_\ell)\right)_{\ell \in \mathbb{N}}$ converges to a fixed point σ_a, which is exponentially attractive for all values of $c_w \neq \sqrt{3}$.

The proof is provided in Appendix A.1.

Remark 3.1. *While the bias distribution is different in our initialization and in the original `SIREN` scheme, the choice $c_w = \sqrt{6}$ for the weight initialization can be recovered as a special case of equation 9 by imposing $\sigma_a = 1$, assuming $c_b = 0$, and by neglecting the correction term introduced by the Lambert function. Using the expansion $\mathcal{W}_0(x) = x + \mathcal{O}\left(x^2\right)$, this correction term can be*

estimated as $\sim e^{-2}$, which is small but not negligible[1]*. Accounting for this correction term enables more precise control over the pre-activation variance σ_a.*

Remark 3.2. *A stated in Theorem 3.1, the convergence of the pre-activation variance to σ_a is exponentially fast with respect to the depth L. Consequently, even shallow neural networks exhibit a pre-activation Gaussian distribution with a variance that is already very close to the fixed point σ_a.*

Deriving the fixed points of the pre-activation distribution is a necessary first step toward characterizing the layer-wise gradient distribution and for establishing the optimal initialization value for c_w and c_b, which we discuss in the next subsection.

3.2 Gradient distribution and stability

The distribution of Jacobian entries is another important property of neural networks that must be carefully controlled during initialization to avoid gradient vanishing (He et al., 2015; Yang & Schoenholz, 2017). In this work, we show that a tractable derivation is possible for the sine activation function. This result is described in Theorem 3.2. Combined with Theorem 3.1 it will enable us to propose a principled initialization strategy provided in Proposition 3.1.

Theorem 3.2 (Jacobian distribution of `SIREN`)**.** *Let $\mathbf{J}_\ell = \partial \mathbf{h}_\ell / \partial \mathbf{h}_{\ell-1}$ denote the Jacobian of the ℓ-th layer. Considering `SIREN` network described in equation 5, we have*

$$\mathbf{J}_\ell = \mathrm{diag}(\cos(\mathbf{z}_\ell))\ \mathbf{W}_\ell.$$

Under the same assumptions that for Theorem 3.1, assuming further that $\mathbf{W}_\ell$ and $\mathbf{z}_\ell$ are independent, and considering the limit of large N, each entry of $\mathbf{J}_\ell$ has zero mean and a variance $\widetilde{\sigma}_\ell^2$, such that the sequence $(\widetilde{\sigma}_\ell^2)_{\ell\in\mathbb{N}}$ converges to

$$\sigma_g^2 = \frac{c_w^2}{6N}\big(1 + e^{-2\sigma_a^2}\big).$$

For a given network, with input $\boldsymbol{x}$ and output $\Psi_\theta(\boldsymbol{x})$, Theorem 3.2 can be used to analyze the scaling behavior of gradients with respect to both the network parameters θ and the input coordinates $\boldsymbol{x}$. We denote by $\partial_{\theta_\ell}\Psi$ the gradient of the network output with respect to the parameters θ_ℓ of layer ℓ, and by $\partial_{\boldsymbol{x}}\Psi$ the gradient with respect to the input $\boldsymbol{x}$. By applying the chain rule, we have :

$$\frac{\partial \Psi_\theta(\boldsymbol{x})}{\partial \theta_\ell} = \frac{\partial \Psi_\theta}{\partial \mathbf{h}_{L-1}} \frac{\partial \mathbf{h}_{L-1}}{\partial \mathbf{h}_{L-2}} \cdots \frac{\partial \mathbf{h}_{\ell+1}}{\partial \mathbf{h}_\ell} \frac{\partial \mathbf{h}_\ell(\boldsymbol{x})}{\partial \theta_\ell}, \tag{10}$$

$$\frac{\partial \Psi_\theta(\boldsymbol{x})}{\partial \boldsymbol{x}} = \frac{\partial \Psi_\theta(\boldsymbol{x})}{\partial \mathbf{h}_{L-1}} \frac{\partial \mathbf{h}_{L-1}}{\partial \mathbf{h}_{L-2}} \cdots \frac{\partial \mathbf{h}_2}{\partial \mathbf{h}_1} \frac{\partial \mathbf{h}_1(\boldsymbol{x})}{\partial \boldsymbol{x}}. \tag{11}$$

These relations can be used to obtained scaling of the gradients variances with the network depth and width (see Appendix A.4 for a derivation):

$$\mathrm{Var}(\partial_{\theta_\ell}\Psi_\theta(\boldsymbol{x})) \propto\ N^{-1}\left(N\sigma_g^2\right)^{L-\ell-1} \quad \text{and} \quad \mathrm{Var}(\partial_{\boldsymbol{x}}\Psi_\theta(\boldsymbol{x}))\ \propto \omega_0^2\left(N\sigma_g^2\right)^{L-2}. \tag{12}$$

Remark 3.3. *From equation 12, we see that gradients in parameter space vanish or explode exponentially with network depth L, unless the scaling factor $N\sigma_g^2$ is close to 1. This is achieved by setting $\sigma_g = 1/\sqrt{N}$, ensuring that gradient variance remains stable across layers.*

In the regime $\sigma_g = 1/\sqrt{N}$, gradients with respect to input coordinates are also controlled and scale proportionally to ω_0 at initialization. In practice, we choose the Nyquist frequency of the estimation that scales as $|\mathbb{I}|^{1/n_0}$, assuming $\mathbb{I}$ evenly-spaced training points within a given spatio-temporal domain properly normalized in each direction. Higher-frequency components injected at initialization cannot be controlled and thus appear as noise. This motivates our proposed initialization at the *edge of chaos*, with $\sigma_g = 1/\sqrt{N}$ and $\omega_0 \sim |\mathbb{I}|^{1/n_0}$, which restricts the initial spectrum to the relevant range.

To conclude the analysis of the statistical properties of `SIREN` networks and derive the initialization scheme provided in equations 7-8, we identify the value of c_w and c_b allowing to control both the distribution of pre-activations, which requires setting $\sigma_a = 1$, and the scaling of gradients i.e. $\sigma_g = 1/\sqrt{N}$.

[1] A more precise estimate of this correction term can be obtained using equation 29, to be derived later.

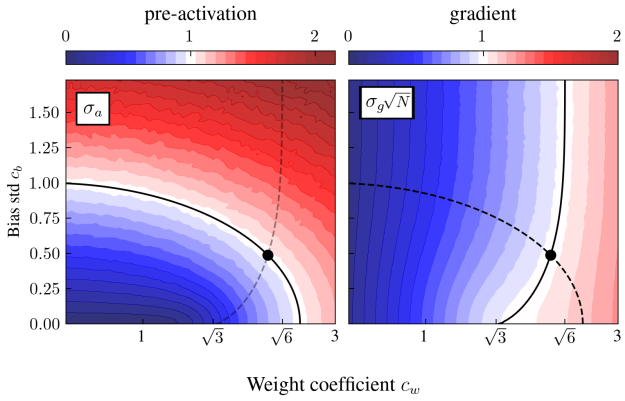

Figure 3: Experimental standard deviation of the pre-activation distribution (left) and of the layer-wise jacobian entries distribution (right), as a function of the parameters (c_w, c_b). The plain and dashed black lines indicate the theoretical predictions for $\sigma_a = 1$ and $\sigma_g = 1/\sqrt{N}$, following Theorems 3.1 and 3.2, respectively. The black dot indicates the initialization provided in Proposition 3.1.

Proposition 3.1. *Under the same assumptions that for Theorem 3.1, setting $\sigma_a = 1$ and $\sigma_g = 1/\sqrt{N}$, lead to the weight and bias variances, c_w and c_b defined in equation 8.*

We verified the validity of this theoretical analysis, involving careful calculations of the Jacobian and pre-activation distributions, through numerical experiments displayed in figure 3. These experiments were done 20 times using a `SIREN` neural network of width $N = 256$ of depth $L = 10$, with input dimension $n_0 = 1$, and output dimension $n_d = 1$, $w_0 = 1$, and following the initialization scheme in equations 7-8. The neural network is then evaluated using $|\mathbb{I}| = 500$ input points $\boldsymbol{x}_i$ uniformly spaced between $[-1, 1]$ to obtain the studied distributions

We now discuss how this initialization affects early stage of training.

4 Scaling of the Neural Tangent Kernel with Depth and Simplified Learning Dynamics

The Neural Tangent Kernel (NTK) framework is a linearized description of the training dynamics around initialization, allowing one to study how the network evolves in the early phase of training Jacot et al. (2018). When training neural networks, we typically use gradient descent to minimize the loss function, with updates $\theta_{t+1} = \theta_t - \mathrm{d}t\nabla_\theta \mathcal{L}(\theta_t)$, where $\mathrm{d}t$ is the learning rate and θ_t the parameter vector at iteration t.

To simplify we restrict ourselves to a scalar output neural network (i.e., $d = 1$). Then, we have for the mean-squared error loss $\mathcal{L}(\theta) = \sum_{i\in\mathbb{I}} \|\Psi_\theta(\boldsymbol{x}_i) - y_i\|^2/|\mathbb{I}|$, and in the continuous-time limit $\mathrm{d}t \to 0$, the residuals $u(\boldsymbol{x}_i, t) = \Psi_{\theta_t}(\boldsymbol{x}_i) - y_i$ satisfy

$$\frac{\mathrm{d}\boldsymbol{u}(t)}{\mathrm{d}t} = \mathbf{K}_{\theta_t}\boldsymbol{u}(t), \quad \mathrm{K}_{\theta_t,i,j} = \nabla_\theta \Psi_{\theta_t}(\boldsymbol{x}_i) \cdot \nabla_\theta \Psi_{\theta_t}(\boldsymbol{x}_j), \tag{13}$$

where $\boldsymbol{u}(t) = (u(\boldsymbol{x}_1, t), \ldots, u(\boldsymbol{x}_{|\mathbb{I}|}, t))$ and $\mathbf{K}_{\theta_t}$ is the NTK matrix.

Assuming the NTK remains constant during training ($\mathbf{K}_{\theta_t} \equiv \mathbf{K}_{\theta_0}$), the residuals evolve as

$$\boldsymbol{u}(t) = \exp(-t\mathbf{K}_{\theta_0})\boldsymbol{u}(0) = \sum_{i=1}^{|\mathbb{I}|} e^{-t\lambda_i}\langle \boldsymbol{u}(0), \boldsymbol{v}_i\rangle \boldsymbol{v}_i, \tag{14}$$

where $(\lambda_i, \boldsymbol{v}_i)$ are the eigenpairs of the initialized NTK $\mathbf{K}_{\theta_0}$, ordered so that $\lambda_1 \geq \cdots \geq \lambda_{|\mathbb{I}|} > 0$, and $\langle \cdot, \cdot \rangle$ the Euclidean scalar product. Thus, the early training dynamics is fully determined by the spectral properties of the NTK at initialization.

Frequency bias in the NTK framework. Equation 14 shows that modes associated with large eigenvalues decay quickly, while those with small eigenvalues converge slowly, with characteristic

timescale $1/\lambda_i$. As illustrated in Fig. 4 for the 1D case, and as observed in related settings (see e.g. Wang et al. (2021)), the leading eigenmodes (small i) of the NTK can be identified with low-frequency Fourier modes, whereas higher-frequency components (large i) correspond to smaller eigenvalues λ_i. This explains the spectral bias of neural networks in the lazy training regime, i.e. the regime of nearly constant NTK. This also highlights the importance of controlling the spectrum $\{\lambda_i\}_{i\in\{1,\dots,|\mathbb{I}|\}}$ in order to faithfully capture all relevant frequencies of the target signal.

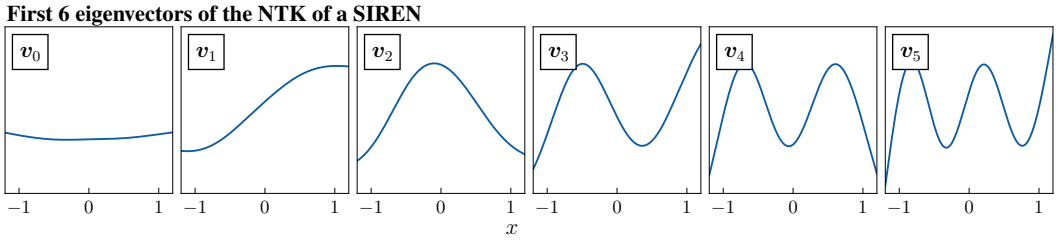

Figure 4: The first six eigenvectors $\boldsymbol{v}_0, \dots, \boldsymbol{v}_5$ of the NTK matrix $\mathbf{K}_{\theta_0}$, ordered by decreasing eigenvalue $\lambda_0 > \lambda_1 > \cdots > \lambda_5$. The NTK matrix was computed numerically on a uniform grid of $|\mathbb{I}| = 500$ points over the interval $\Omega = [-1, 1]$ using a SIREN network of width $N = 512$ and of depth $L = 8$ and using $\omega_0 = 1$. The eigenvectors exhibit increasingly oscillatory behavior as the mode index grows, consistent with their interpretation as Fourier-like modes. This observation confirms the spectral structure predicted by our analysis and highlights the tendency of the NTK to prioritize low-frequency components associated with larger eigenvalues.

Empirical scaling of NTK eigenvalues and network gradients. To highlight the importance of initialization in the limit of large network depth, we conducted an experiment comparing the original SIREN initialization (cf. equation 6) the proposed one (cf. equations 7-8), and another one with the default PyTorch initialization, corresponding to $c_b = 1/\sqrt{N}$ and $c_w = 1$. We varied the depth L while fixing $N = 256$, $|\mathbb{I}| = 200$, and $\omega_0 = 1$. In figure 5, we plot the normalized NTK trace (mean eigenvalue) expressed as $\mathrm{Tr}(\mathbf{K}_{\theta_0})/|\mathbb{I}|N$, together with the gradient norm $\|\partial_x \Psi_{\theta_0}\|$ as functions of network depth. We use the NTK trace as a computationally convenient proxy for the typical eigenvalue behavior as depth increases. With the original SIREN initialization, we observe exponential growth of both the NTK eigenvalues and the input gradients. In this case, increasing depth accelerates training but also causes gradient explosion in input space. This corresponds to spurious high-frequency components absent from the target signal, which degrade generalization, here understood as smooth interpolation between data points. With PyTorch initialization, the NTK eigenvalues decrease until reaching a plateau, while the gradient in input coordinate space vanishes. By contrast, with our proposed initialization, the NTK eigenvalues increases linearly with depth while the gradients remain constant. Consequently, the effective learning rate increases with depth L, while the input-space gradients stay normalized, preventing the emergence of noise at spatial scales smaller than ω_0 in the initial state. These behavior is practically confirmed on real training cases such as image fitting example (i.e, image super-resolution) depicted in figure 2.

Interpretation of the scalings. The scaling of gradients with $(\sigma_g\sqrt{N})^L$ is expected from section 3.2, with $\sigma_g\sqrt{N} \approx \sqrt{1.2}$ for SIREN, $= 1$ for our proposed initialization, and $= \sqrt{1/3}$ for PyTorch initialization. Similarly, it is possible to explain the NTK eigenvalue scaling. We note first that diagonal element of the NTK matrix are $\mathrm{K}_{\theta_0,i,i} = |\nabla_\theta \Psi_{\theta_0}(\boldsymbol{x}_i)|^2$. From this and the zero mean property of every gradient distribution, we relate the average eigenvalue of the NTK denoted $\bar{\lambda}$ to the variance of gradients in parameter space:

$$\bar{\lambda} = \tfrac{1}{|\mathbb{I}|}\,\mathrm{Tr}(\mathbf{K}_{\theta_0}) = N^2 \sum_{\ell=1}^{L} \mathrm{Var}\left[\nabla_{\mathrm{W}_\ell} \Psi_{\theta_0}(\boldsymbol{x}_i)\right] + N \sum_{\ell=1}^{L} \mathrm{Var}\left[\nabla_{\mathrm{b}_\ell} \Psi_{\theta_0}(\boldsymbol{x}_i)\right], \tag{15}$$

where $\mathrm{W}_\ell, \mathrm{b}_\ell$ are respectively a weight and a bias of the ℓ-th layer. The sum involving weights parameters being dominant, we neglect the sum on bias terms in the following. When $\sigma_g^2 \neq 1/N$, using equation 10, we obtain a geometric sum, leading to

$$\tfrac{1}{|\mathbb{I}|N}\,\mathrm{Tr}(\mathbf{K}_{\theta_0}) \propto \frac{(N\sigma_g^2)^{L+1} - 1}{N\sigma_g^2 - 1}. \tag{16}$$

- If $\sigma_g/\sqrt{N} > 1$ (SIREN original), then $\bar{\lambda} \propto (N\sigma_g^2)^L$ and the NTK explodes exponentially with depth L. This exponential scaling for the NTK eigenvalues without proper initialization was observed experimentally in de Avila Belbute-Peres & Kolter (2023), yet without precise discussion on the causes and the effect of such behavior, since their focus was on the choice of ω_0 rather than on weight and bias initialization.
- If $\sigma_g\sqrt{N} < 1$ (SIREN PyTorch), NTK eigenvalues become independent from the depth L in the large depth limit, yielding slow convergence, together with vanishing gradients.
- If $\sigma_g\sqrt{N} = 1$ (SIREN proposed), equation 16 does not apply. Each term of the sum on weight parameters in equation 15 gives the same contribution, leading to $\bar{\lambda} \propto L$, which is consistent with the results plotted figure 5 for our proposed initialization.

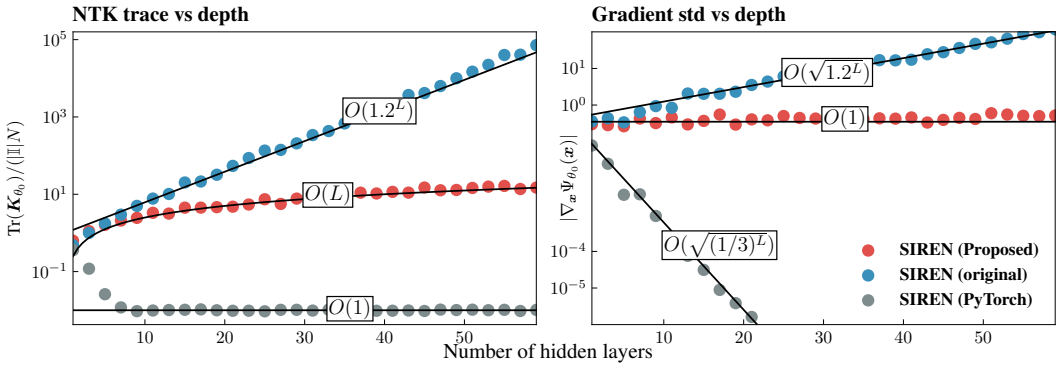

Figure 5: The left plot stands for the scaling of the mean eigenvalue of the NTK matrix over the number of layer. The right plot stands for the scaling of the gradient of the network (in input coordinate space) with the number of layers. The experimental setup and hyper-parameters are the same as in figure 4, except for the network depth which varies here.

5 Discussion, Conclusion, Perspectives

We proposed a new initialization scheme for sinusoidal neural networks that prevents gradient explosion and vanishing. The parametrization is derived analytically by examining the variances of pre-activations and layer-to-layer Jacobians in the limit of infinitely wide and deep networks. This approach removes the need for architectural tricks such as skip connections or empirical hyperparameter tuning to stabilize deep models. By analyzing both the neural tangent kernel and input-space gradients, we showed that this initialization enables deep networks to train with learning rates that scale linearly with depth, while suppressing spurious noise above the Nyquist frequency in implicit neural representations. Whereas prior work motivated the use of sine activations by noting that derivatives of SIRENs remain well-behaved, our study goes further by providing a deeper theoretical analysis. We demonstrate that sinusoidal architectures not only preserve these desirable properties but also admit stronger theoretical justification. Indeed, it enables us to explicitly compute key variances of network distributions.

Among the initialization constraints, controlling the Jacobian variance ($\sigma_g^2 = 1/N$) is essential, while the pre-activation variance ($\sigma_a^2 = 1$) may be adapted and optimized in future work. Although this study focuses on signal encoding with a quadratic loss, future work could extend the approach to more complex losses, including physics-informed settings, with potential applications in atmospheric and oceanic field reconstruction. More broadly, our results may encourage wider adoption of sine activations in machine learning.

REPRODUCIBILITY

Code Implementation. All source code used in our experiments is provided in the supplementary material, including implementations of the architectures used for comparison.

Models and Architectures. Details on the choice of activation functions are given in the main text. Initialization methods and architectural specifications for each model are described within the corresponding experimental sections.

Experiments. Each experiment is reported with its hyperparameters (e.g., learning rate, optimizer, number of epochs) in the relevant sections or figures. All experiments were run with fixed random seeds to ensure exact reproducibility of the reported results.

ACKNOWLEDGEMENTS

We gratefully acknowledge the support of the Centre Blaise Pascal's IT test platform at ENS de Lyon (Lyon, France) for the Machine Learning facilities. The platform operates the SIDUS solution (Quemener & Corvellec, 2013) developed by Emmanuel Quemener.

REFERENCES

Zhicheng Cai, Hao Zhu, Qiu Shen, Xinran Wang, and Xun Cao. Batch normalization alleviates the spectral bias in coordinate networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 25160–25171, June 2024.

Filipe de Avila Belbute-Peres and J Zico Kolter. Simple initialization and parametrization of sinusoidal networks via their kernel bandwidth. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=yVqC6gCNf4d.

Emilien Dupont, Adam Golinski, Milad Alizadeh, Yee Whye Teh, and Arnaud Doucet. Coin: Compression with implicit neural representations. In *International Conference on Learning Representations*, 2021. URL http://arxiv.org/abs/2103.03123v2.

Stefan Elfwing, Eiji Uchibe, and Kenji Doya. Sigmoid-weighted linear units for neural network function approximation in reinforcement learning. *Neural Networks*, 107:3–11, 2018. ISSN 0893-6080. doi: https://doi.org/10.1016/j.neunet.2017.12.012. URL https://www.sciencedirect.com/science/article/pii/S0893608017302976. Special issue on deep reinforcement learning.

Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In Yee Whye Teh and Mike Titterington (eds.), *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, volume 9 of *Proceedings of Machine Learning Research*, pp. 249–256, Chia Laguna Resort, Sardinia, Italy, 13–15 May 2010. PMLR. URL https://proceedings.mlr.press/v9/glorot10a.html.

Soufiane Hayou, Arnaud Doucet, and Judith Rousseau. On the impact of the activation function on deep neural networks training. In *International conference on machine learning*, pp. 2672–2680. PMLR, 2019.

Soufiane Hayou, Arnaud Doucet, and Judith Rousseau. The curse of depth in kernel regime, 13 Dec 2022. URL https://proceedings.mlr.press/v163/hayou22a.html.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pp. 1026–1034, 2015. doi: 10.1109/ICCV.2015.123.

Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv: Learning*, 2016. URL https://api.semanticscholar.org/CorpusID:125617073.

Arthur Jacot, Franck Gabriel, and Clément Hongler. Neural tangent kernel: Convergence and generalization in neural networks. *Advances in neural information processing systems*, 31, 2018.

J A S Kelso, A J Mandell, M F Shlesinger, and N H Packard. *Dynamic Patterns in Complex Systems*, chapter 3. Addison-Wesley, 1986.

Christopher G Langton. Studying artificial life with cellular automata. *Physica D: Nonlinear Phenomena*, 22(1):120–149, 1986. ISSN 0167-2789. doi: https://doi.org/10.1016/0167-2789(86)90237-X. URL https://www.sciencedirect.com/science/article/pii/016727898690237X. Proceedings of the Fifth Annual International Conference.

Yicheng Li, Zixiong Yu, Guhan Chen, and Qian Lin. On the eigenvalue decay rates of a class of neural-network related kernel functions defined on general domains, 2024. URL https://arxiv.org/abs/2305.02657.

Mingze Ma, Qingtian Zhu, Yifan Zhan, Zhengwei Yin, Hongjun Wang, and Yinqiang Zheng. Robustifying fourier features embeddings for implicit neural representations, 2025. URL https://arxiv.org/abs/2502.05482.

Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *ECCV*, 2020.

Vinod Nair and Geoffrey E. Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th International Conference on International Conference on Machine Learning*, ICML'10, pp. 807–814, Madison, WI, USA, 2010. Omnipress. ISBN 9781605589077.

Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

Emmanuel Quemener and Marianne Corvellec. Sidus—the solution for extreme deduplication of an operating system. *Linux J.*, 2013(235), November 2013. ISSN 1075-3583.

Nasim Rahaman, Aristide Baratin, Devansh Arpit, Felix Draxler, Min Lin, Fred Hamprecht, Yoshua Bengio, and Aaron Courville. On the spectral bias of neural networks. In Kamalika Chaudhuri and Ruslan Salakhutdinov (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 5301–5310. PMLR, 09–15 Jun 2019. URL https://proceedings.mlr.press/v97/rahaman19a.html.

Maziar Raissi, Paris Perdikaris, and George E Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, 378:686–707, 2019.

Mariia Seleznova and Gitta Kutyniok. Neural tangent kernel beyond the infinite-width limit: Effects of depth and initialization. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato (eds.), *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 19522–19560. PMLR, 17–23 Jul 2022. URL https://proceedings.mlr.press/v162/seleznova22a.html.

Vincent Sitzmann, Julien Martel, Alexander Bergman, David Lindell, and Gordon Wetzstein. Implicit neural representations with periodic activation functions. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 7462–7473. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/53c04118df112c13a8c34b38343b9c10-Paper.pdf.

Yannick Strümpler, Janis Postels, Ren Yang, Luc Van Gool, and Federico Tombari. Implicit neural representations for image compression. In Shai Avidan, Gabriel Brostow, Moustapha Cissé, Giovanni Maria Farinella, and Tal Hassner (eds.), *Computer Vision – ECCV 2022*, pp. 74–91, Cham, 2022. Springer Nature Switzerland. ISBN 978-3-031-19809-0.

Matthew Tancik, Pratul P. Srinivasan, Ben Mildenhall, Sara Fridovich-Keil, Nithin Raghavan, Utkarsh Singhal, Ravi Ramamoorthi, Jonathan T. Barron, and Ren Ng. Fourier features let networks learn high frequency functions in low dimensional domains. *NeurIPS*, 2020.

Sifan Wang, Hanwen Wang, and Paris Perdikaris. On the eigenvector bias of fourier feature networks: From regression to solving multi-scale pdes with physics-informed neural networks. *Computer Methods in Applied Mechanics and Engineering*, 384:113938, 2021. ISSN 0045-7825. doi: https://doi.org/10.1016/j.cma.2021.113938. URL https://www.sciencedirect.com/science/article/pii/S0045782521002759.

Ge Yang and Samuel Schoenholz. Mean field residual networks: On the edge of chaos. *Advances in neural information processing systems*, 30, 2017.

A Mathematical Appendix

A.1 Input distribution

Theorem (Restatement of Theorem 3.1)**.** *Considering SIREN network described in equation 5 where, for some $c_w, c_b \in \mathbb{R}^+$, and for every layer $\ell \in \{2, \dots, L\}$, the weight matrix $\mathbf{W}_\ell$ is initialized as a random matrix sampled from $\mathcal{U}(-c_w/\sqrt{N}, c_w/\sqrt{N})$, and the bias $\mathbf{b}_\ell$ is initialized as a random vector sampled from $\mathcal{N}(0, c_b^2)$. Let $(\mathbf{z}_\ell)_{\ell \in \{1\dots,L\}}$ the preactivation sequence defined in equation 3 and relying on an input $\boldsymbol{x} \in \mathbb{R}^{n_0}$. Then, in the limit of large N, the preactivation sequence $(\mathbf{z}_\ell)_{\ell \in \mathbb{N}}$ converges in distribution to $\mathcal{N}(0, \sigma_a^2)$ where*

$$\sigma_a^2 = c_b^2 + \frac{c_w^2}{6} + \frac{1}{2}\mathcal{W}_0\left(-\frac{c_w^2}{3}e^{-\frac{c_w^2}{3}-2c_b^2}\right) \tag{17}$$

with $\mathcal{W}_0$ is the principal real branch of the Lambert function. Additionally, the sequence associated to the variance of the preactivation $\big(\mathrm{Var}(\mathbf{z}_\ell)\big)_{\ell \in \mathbb{N}}$ converges to a fixed point σ_a, which is exponentially attractive for all values of $c_w \neq \sqrt{3}$.

Proof. The proof can be split in three steps: (i) prove that the sequence of preactivations follows a Gaussian distribution (cf. Lemma A.1), (ii) give an expression of the variance of the output of a sin activation when the input follows a zero-mean Gaussian distribution of s.t.d. σ_a (cf. Lemma A.2), (iii) provides the expression of the variance of each element of the preactivation sequence using the result in (ii) and proves its convergence to a fixed point σ_a (cf. Lemma A.3).

Lemma A.1. *Considering SIREN network described in equation 5 where, for some $c_w, c_b \in \mathbb{R}^+$, and for every layer $\ell \in \{2, \dots, L\}$, the weight matrix $\mathbf{W}_\ell$ is initialized as a random matrix sampled from $\mathcal{U}(-c_w/\sqrt{N}, c_w/\sqrt{N})$, and the bias $\mathbf{b}_\ell$ is initialized as a random vector sampled from $\mathcal{N}(0, c_b^2)$. Let $(\mathbf{z}_\ell)_{\ell \in \{1\dots,L\}}$ the preactivation sequence defined in equation 3 and relying on an input $\boldsymbol{x} \in \mathbb{R}^{n_0}$. Then, in the limit of large N, each element of the preactivation sequence $(\mathbf{z}_\ell)_{\ell \in \mathbb{N}}$ is distributed according to a zero-mean Gaussian distribution.*

Proof. We recall that for the first layer, $\mathbf{h}_0 = \boldsymbol{x}$ and, for every $\ell \in \{1, \dots, L\}$,

$$\mathbf{h}_\ell = \sin\big(\mathbf{W}_\ell \mathbf{h}_{\ell-1} + \mathbf{b}_\ell\big).$$

Since the sine activation is an odd function, it preserves the zero-mean property of any distribution: if $\mathbf{z}_\ell = \mathbf{W}_\ell \mathbf{h}_{\ell-1} + \mathbf{b}_\ell$ has zero mean, then $\mathbf{h}_\ell$ will also have zero mean. This property propagates layer by layer.

As $\mathbf{W}_1$ and $\mathbf{b}_1$ are assumed to have zero mean (by definition, cf. equation 7) and $\boldsymbol{x}$ is a deterministic vector, it ensures that the first-layer pre-activation has zero-mean. Moreover, as $\mathbf{W}_\ell$ and $\mathbf{b}_\ell$ are assumed to have zero mean the zero-mean property holds for all subsequent pre-activations $\mathbf{z}_\ell$ and $\mathbf{h}_\ell$.

Second, we prove that the preactivation sequence is distributed according to a Gaussian. We first rewrite each element of the preactivation sequence as

$$\mathrm{z}_{\ell,i} = \sum_{j=1}^{N} \mathrm{W}_{\ell,i,j}\mathrm{h}_{\ell-1,j} + \mathrm{b}_{\ell,i}\,. \tag{18}$$

As a sum of two Gaussian stays Gaussian and because $\mathbf{b}_\ell$ is assumed to be Gaussian with a standard deviation σ_b, the main purpose here is then to prove that $\sum_{j=1}^{N} \mathrm{W}_{\ell,i,j}\mathrm{h}_{\ell-1,j}$ follow a Gaussian distribution.

Thanks to the Central Limit Theorem, whatever is the distribution of $\mathrm{h}_{\ell-1,j}$, the term $\sum_{j=1}^{N} \mathrm{W}_{\ell,i,j}\mathrm{h}_{\ell-1,j}$ converges in distribution to a Gaussian distribution in the limit of large N. Since the bias is also normally sampled, each component $\mathrm{z}_{\ell,i}$ follows a gaussian distribution in the same large N limit, with zero mean and a variance denoted σ_a^2.

To compute this variance, let us first compute the variance of each summand denoted $\sigma^2_{\ell,i,j}$, given by the product of two independent random variables with zero mean, namely $\mathrm{W}_{\ell,i,j}$ and $\mathrm{h}_{\ell-1,j}$,

$$\sigma^2_{\ell,i,j} = \mathrm{Var}\left[\mathrm{W}_{\ell,i,j}\right]\mathrm{Var}\left[\mathrm{h}_{\ell-1,j}\right],. \tag{19}$$

Since $\mathrm{W}_{\ell,i,j}$ is uniformly distributed on $[-c_w/\sqrt{N}, c_w/\sqrt{N}]$, we have:

$$\mathrm{Var}\left[\mathrm{W}_{\ell,i,j}\right] = \frac{c_w^2}{3N}. \tag{20}$$

While the variance of $\mathrm{h}_{\ell-1,j}$ is still unknown, we can express it from the knowledge of $\mathbf{z}_{\ell-1}$, leading to

$$\sigma^2_{\ell,i,j} = \frac{c_w^2}{3N}\mathrm{Var}\left[\sin(\mathrm{z}_{\ell-1,j})\right]. \tag{21}$$

whose expression of $\mathrm{Var}\left[\sin(\mathrm{z}_{\ell-1,j})\right]$ will be provided later.

As the bias variance follows a Gaussian distribution as described in equation 7, the variance of all the elements of the preactivation z_ℓ is

$$\sigma^2_\ell = \frac{c_w^2}{3}\mathrm{Var}\left[\sin(\mathrm{z}_{\ell-1})\right] + c_b^2. \tag{22}$$

□

Lemma A.2. *Let z be a normally distributed random variable and zero mean $z \sim \mathcal{N}(0, \sigma^2)$. Then we have :*

$$\mathrm{Var}\left[\sin\left(z\right)\right] = \frac{1}{2}\left(1 - e^{2\sigma^2}\right). \tag{23}$$

Proof of Lemma A.2. The proof combined the properties of the Gaussian distribution with the fact that the sine function is an odd function. We have:

$$\mathrm{Var}\left[\sin\left(z\right)\right] = \mathbb{E}\left[\sin^2(z)\right] - \mathbb{E}\left[\sin(z)\right]^2$$

Since $\sin$ is odd and since the expectation of z is zero, we have $\mathbb{E}\left[\sin(z)\right] = 0$. In addition, using $\sin^2(z) = (1 - \cos(2z))/2$, we obtain

$$\mathbb{E}\left[\sin^2(z)\right] = \frac{1}{2} - \frac{1}{2}\mathbb{E}\left[\cos(2z)\right].$$

The characteristic function of the Gaussian distribution with zero mean and variance σ_a is given by:

$$g_z(t) = \mathbb{E}(e^{itz}) = e^{-\frac{1}{2}t^2\sigma^2}.$$

Now we notice that

$$\mathbb{E}\left[\cos(2z)\right] = \mathbb{E}\left[\Re\left[e^{i2z}\right]\right] = \Re\left[g_z(2)\right] = e^{-2\sigma_a^2}.$$

The first equality uses the linearity of the mean. This leads to the final result:

$$\mathrm{Var}\left[\sin\left(z\right)\right] = \frac{1}{2}\left(1 - e^{-2\sigma^2}\right).$$

□

Lemma A.3. *Considering SIREN network described in equation 5 where, for some $c_w, c_b \in \mathbb{R}^+$, and for every layer $\ell \in \{1, \ldots, L\}$, the weight matrix $\mathbf{W}_\ell$ is initialized as a random matrix sampled from $\mathcal{U}(-c_w/\sqrt{N}, c_w/\sqrt{N})$, and the bias $\mathbf{b}_\ell$ is initialized as a random vector sampled from $\mathcal{N}(0, c_b^2)$. Let $\boldsymbol{x} \in \mathbb{R}^{n_0}$. Then, in the limit of large N, the preactivation sequence $(\mathbf{z}_\ell)_{\ell\in\{1\ldots,L\}}$ defined in equation 3 is distributed according to a Gaussian distribution with zero-mean and, for every ℓ, a variance*

$$\sigma^2_\ell = \frac{c_w^2}{6}\left(1 - e^{-2\sigma^2_{\ell-1}}\right) + c_b^2$$

Moreover, the sequence $(\sigma^2_\ell)_{\ell\in\mathbb{N}}$ converges to

$$\sigma_a^2 = c_b^2 + \frac{c_w^2}{6} + \frac{1}{2}\mathcal{W}_{0,-1}\left(-\frac{c_w^2}{3}e^{-\frac{c_w^2}{3} - 2c_b^2}\right),$$

with $\mathcal{W}_{0,-1}$ the two real branches of the Lambert W function. And for $c_w \neq \sqrt{3}$, this convergence is exponentially fast.

Proof of Lemma A.3.

Fixed Point Value : Combining equation 22 and equation A.3, the variance of the pre-activation at layer ℓ is

$$\sigma_\ell^2 = \frac{c_w^2}{6}\left(1 - e^{-2\sigma_{\ell-1}^2}\right) + c_b^2$$

To characterize the fixed point of the sequence $(\sigma_\ell^2)_{\ell\in\mathbb{N}}$, we define a function f as

$$f(x) = \frac{c_w^2}{6}\left(1 - e^{-2x}\right) + c_b^2. \tag{24}$$

The fixed point of this function is given by the solution of the equation $f(x) = x$. Rearranging the different term gives:

$$\frac{c_w^2}{6} + c_b^2 - x = \frac{c_w^2}{6}e^{-2x}. \tag{25}$$

Using $y = \frac{c_w^2}{6} + c_b^2 - x$ yields

$$ye^{-2y} = \frac{c_w^2}{6}e^{-2(\frac{c_w^2}{6}+c_b^2)}.$$

Then, using the definition of the real valued Lambert W function, we get

$$y = -\frac{1}{2}\mathcal{W}_k\left(-\frac{c_w^2}{3}e^{-2(\frac{c_w^2}{6}+c_b^2)}\right), \quad \text{where} \quad k \in \{-1, 0\}.$$

The $\mathcal{W}_0$ branch is called the principal branch and is defined on $(-e^{-1}, +\infty)$. The $\mathcal{W}_{-1}$ branch is defined for $(-e^{-1}, 0)$. To obtain a positive variance, the branch to consider is $\mathcal{W}_0$, as illustrated numerically in figure 6.

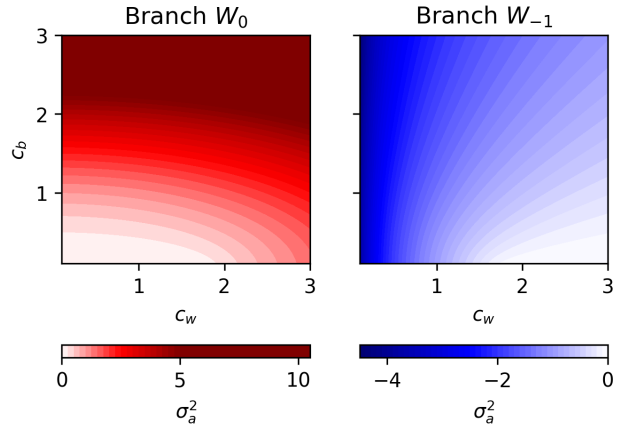

Figure 6: The σ_a solution emerging from the W_0 branch on the left and W_{-1} branch on the right

Convergence Speed : To quantify the convergence towards the fixed point σ_a^2, consider the derivative of f at the fixed point:

$$f'(\sigma_a^2) = \frac{c_w^2}{3}e^{-2\sigma_a^2}.$$

The fixed point is exponentially attractive whenever $f'(\sigma_a^2) < 1$, which is immediately satisfied for $c_w < \sqrt{3}$. For $c_w > \sqrt{3}$, Lemma A.3 gives

$$f'(\sigma_a^2) = 2(-f(\sigma_a) + \frac{c_w^2}{6} + c_b^2) = -\mathcal{W}_0\left(-\tfrac{c_w^2}{3}\, e^{-c_w^2/3 - 2c_b^2}\right).$$

Since

$$-\tfrac{1}{e} < -\tfrac{c_w^2}{3}\, e^{-c_w^2/3-2c_b^2} < 0,$$

the properties of the principal branch $\mathcal{W}_0$ imply $|f'(\sigma_a^2)| < 1$. Hence, the fixed point is exponentially attractive for all values of $c_w \neq \sqrt{3}$, and convergence occurs rapidly.

This concludes the proof of the Lemma A.3, and of the Theorem 3.1. □

□

A.2 GRADIENT DISTRIBUTION

Theorem (Restatement of Theorem 3.2)**.** *Let $\mathbf{J}_\ell = \partial \mathbf{h}_\ell / \partial \mathbf{h}_{\ell-1}$ denote the Jacobian of the ℓ-th layer. Under the same assumptions that for Theorem 3.1 and assuming that $\mathbf{W}_\ell$ and $\mathbf{z}_\ell$ are independent, we have*

$$\mathbf{J}_\ell = \mathrm{diag}(\cos(\mathbf{z}_\ell))\ \mathbf{W}_\ell.$$

In the limit of large N, each entry of $\mathbf{J}_\ell$ has zero mean and a sequence of variance $\widetilde{\sigma}_\ell^2$ such that the sequence $(\widetilde{\sigma}_\ell^2)_{\ell\in\mathbb{N}}$ that converges to

$$\sigma_g^2 = \frac{c_w^2}{6N}\big(1 + e^{-2\sigma_a^2}\big).$$

Proof. An element of the Jacobian of the ℓ-th layer are writen as:

$$\frac{\partial \mathrm{h}_{\ell,i}}{\partial \mathrm{h}_{\ell-1,k}} = \mathrm{W}_{\ell,i,k} \cos\left(\sum_{j=1}^{N} \mathrm{W}_{\ell,i,j}\mathrm{h}_{\ell-1,j} + \mathrm{b}_{\ell,i}\right) = \mathrm{W}_{\ell,i,k} \cos(\mathrm{z}_{\ell,i})$$

with $\mathrm{z}_{\ell,i}$ the i^{th} component of preactivation vector defined in equation 3. Assuming independance of variable $\mathrm{W}_{\ell,i,k}$ and $\cos(\mathrm{z}_{\ell,i})$, the variance of their product denoted $\widetilde{\sigma}_\ell^2$ can be expressed as the product of their variance:

$$\widetilde{\sigma}_\ell^2 = \mathrm{Var}\left[\mathrm{W}_{\ell,i,k}\right] \mathrm{Var}\left[\cos(\mathrm{z}_{\ell,i})\right].$$

Considering the same arguments as for Theorem 3.1 and replacing $\sin$ by $\cos$, the sequence $(\widetilde{\sigma}_\ell)_{\ell\in\mathbb{N}}$ converges to

$$\sigma_g^2 = \frac{c_w^2}{6N}(1 + e^{-2\sigma_a^2}),$$

with σ_a^2 the limit variance of the pre-activation, given by Theorem 3.1.

□

A.3 PROOF OF PROPOSITION EQUATION 3.1

We propose to initialize the weights and biases of SIREN networks as follows:

$$\mathbf{W}_\ell \sim \begin{cases} \mathcal{U}\left(-\frac{\omega_0}{n_0}, \frac{\omega_0}{n_0}\right), & \ell = 1, \\ \mathcal{U}\left(-\frac{c_w}{\sqrt{N}}, \frac{c_w}{\sqrt{N}}\right), & \ell \in \{2, \dots, L\}, \end{cases}$$

and

$$\mathbf{b}_\ell \sim \mathcal{N}(0, c_b^2),\ \ell \in \{1, \dots, L\}.$$

To control the distribution of pre-activations and the scaling of gradients we impose $\sigma_a^2 = 1$ and $\sigma_g^2 = \frac{1}{N}$, i.e.,

$$c_b^2 + \frac{c_w^2}{6} + \frac{1}{2}\mathcal{W}_0\left(-\frac{c_w^2}{3}e^{-\frac{c_w^2}{3} - 2c_b^2}\right) = 1, \tag{26}$$

and

$$\frac{c_w^2}{6N}\big(1 + e^{-2}\big) = \frac{1}{N}. \tag{27}$$

From equation 27, we get

$$c_w = \sqrt{\frac{6}{1 + e^{-2}}}. \tag{28}$$

Combining this result with equation 26 leads to an implicit equation for c_b^2. To obtain an explicit expression, it is convenient to use the fixed-point equation 25 with $x = \sigma_a = 1$ leading to:

$$\frac{c_w^2}{6}\left(1 - e^{-2}\right) + c_b^2 = 1, \tag{29}$$

which, using equation (28), simplifies to

$$c_b^2 = \frac{c_w^2 e^{-2}}{3}. \tag{30}$$

This initialization ensures that, in the limit of infinitely deep and wide networks, the distributions of inputs and gradients remain normalized. In practice, it enables stable information propagation through the network while keeping the sinusoidal activation function in a marginally nonlinear regime.

A.4 Derivation of the Proposed Scaling

Let $\Psi_\theta(\boldsymbol{x})$ defined as in equation 5 a scalar output function, initialized as in the previous theorems, and considering a given value of σ_g resulting from the initialization.

Derivation of the parameter-wise Gradient scaling: Considering a weight-parameter W_ℓ, i, j with $\ell > 1$ of the ℓ-th layer, we study the scalar $\frac{\partial \Psi_\theta(\boldsymbol{x})}{\partial \mathrm{W}_{\ell,i,j}}$, which can be rewritten as :

$$\frac{\partial \Psi_\theta(\boldsymbol{x})}{\partial \mathrm{W}_{\ell,i,j}} = \frac{\partial \Psi_\theta}{\partial \mathbf{h}_{L-1}} \frac{\partial \mathbf{h}_{L-1}}{\partial \mathbf{h}_{L-2}} \cdots \frac{\partial \mathbf{h}_{\ell+1}}{\partial \mathbf{h}_\ell} \frac{\partial \mathbf{h}_\ell(\boldsymbol{x})}{\partial \mathrm{W}_{\ell,i,j}}$$

Then from theorem 3.2 under the choice of our initialization we know that the Jacobian matrices $\mathbf{J}_\ell = \partial \mathbf{h}_\ell / \partial \mathbf{h}_{\ell-1}$ have variance σ_g^2/N in the limit of large l and large N. Moreover, we have from the definition of Ψ_θ the expression of the vector $\frac{\partial \Psi_\theta}{\partial \mathbf{h}_{L-1}} = \mathbf{W}_L$ with $\mathrm{Var}(\mathbf{W}_L) \sim 1/N$. Let us consider first the sensitivity vector $\mathbf{g}_\ell$:

$$\mathbf{g}_\ell = \frac{\partial \Psi_\theta}{\partial \mathbf{h}_{L-1}} \frac{\partial \mathbf{h}_{L-1}}{\partial \mathbf{h}_{L-2}} \cdots \frac{\partial \mathbf{h}_{\ell+1}}{\partial \mathbf{h}_\ell}. \tag{31}$$

Owing to the impact of matrix multiplication on every components, we have $\mathrm{Var}(\mathbf{g}_\ell) \sim (N\sigma_g^2)^{L-\ell-1}/N$. Let us now consider now the term $\frac{\partial \mathbf{h}_\ell(\boldsymbol{x})}{\partial \mathrm{W}_{\ell,i,j}}$. This is a zero vector except for the i-th component, verifying $\frac{\partial \mathbf{h}_{\ell,i}(\boldsymbol{x})}{\partial \mathrm{W}_{\ell,i,j}} = \mathbf{h}_{\ell-1,j} \cos(\mathbf{W}_{\ell-1,i,:}\mathbf{h}_{\ell-1} + \mathbf{b}_i)$, with variance $\mathrm{Var}(\frac{\partial \mathbf{h}_{\ell,i}(\boldsymbol{x})}{\partial \mathrm{W}_{\ell,i,j}}) \sim 1$. Hence, the parameter-wise gradient can be rewritten as:

$$\frac{\partial \Psi_\theta(\boldsymbol{x})}{\partial \mathrm{W}_{\ell,i,j}} = \mathbf{g}_{\ell,i} \mathbf{h}_{\ell-1,j} \cos(\mathbf{W}_{\ell-1,i,:}\mathbf{h}_{\ell-1} + \mathbf{b}_i).$$

Assuming independence between $\mathbf{g}_{\ell,i}$ and $\frac{\partial \Psi_\theta(\boldsymbol{x})}{\partial \mathrm{W}_{\ell,i,j}}$, we finally obtain the desired variance scaling, namely $\mathrm{Var}(\frac{\partial \Psi_\theta(\boldsymbol{x})}{\partial \mathrm{W}_{\ell,i,j}}) \sim (N\sigma_g^2)^{L-\ell-1}/N$.

Derivation of the input-wise Gradient scaling: Following the same notations as above, we have:

$$\frac{\partial \Psi_\theta(\boldsymbol{x})}{\partial \boldsymbol{x}} = \frac{\partial \Psi_\theta(\boldsymbol{x})}{\partial \mathbf{h}_{L-1}} \frac{\partial \mathbf{h}_{L-1}}{\partial \mathbf{h}_{L-2}} \cdots \frac{\partial \mathbf{h}_2}{\partial \mathbf{h}_1} \frac{\partial \mathbf{h}_1(\boldsymbol{x})}{\partial \boldsymbol{x}}.$$

Recalling that $\mathbf{g}_1$, has variance $\mathrm{Var}(\mathbf{g}_1) \sim (N\sigma_g^2)^{L-2}/N$. In that case the $1/N$ factor will cancel out due to the term $\frac{\partial \mathbf{h}_1(\boldsymbol{x})}{\partial \boldsymbol{x}}$. Indeed, we have:

$$\frac{\partial \mathbf{h}_1(\boldsymbol{x})}{\partial \boldsymbol{x}} = \mathrm{diag}(\cos(\mathbf{W}_1 \boldsymbol{x} + \mathbf{b}))\ \mathbf{W}_1,$$

which is a non-trivial matrix of variance $\mathrm{Var}(\frac{\partial \mathbf{h}_1(\boldsymbol{x})}{\partial \boldsymbol{x}}) \sim w_0^2$, for both the original and proposed SIREN initialization. Focusing on one input coordinate x_i, we get:

$$\frac{\partial \Psi_\theta(\boldsymbol{x})}{\partial x_i} = \mathbf{g}_1\, \mathrm{diag}(\cos(\mathbf{W}_1 \boldsymbol{x} + \mathbf{b}))\ \mathbf{W}_{1,:,i} = \sum_j \mathbf{g}_{1,j}\, (\mathrm{diag}(\cos(\mathbf{W}\boldsymbol{x} + \mathbf{b}))\ \mathbf{W}_{1,:,i})_j.$$

The variance of each term scales as $\sim (N\sigma_g^2)^{L-2}/N$. Supposing independence between each summand leads to $\mathrm{Var}(\frac{\partial \Psi_\theta(\boldsymbol{x})}{\partial \boldsymbol{x}}) \sim (N\sigma_g^2)^{L-2} w_0^2$.

B EXPERIMENTAL APPENDIX

To explore the impact of the initialization on the performances of several Neural Network architectures we studied multiple problems such as function and image fitting.

B.1 IMAGE FITTING EXPERIMENTS

After comparing multiple architecture in the main content, we will focus here on the degradation of the reconstruction with the increasing of the network depth as highlighted in the paper, we conducted this experiment

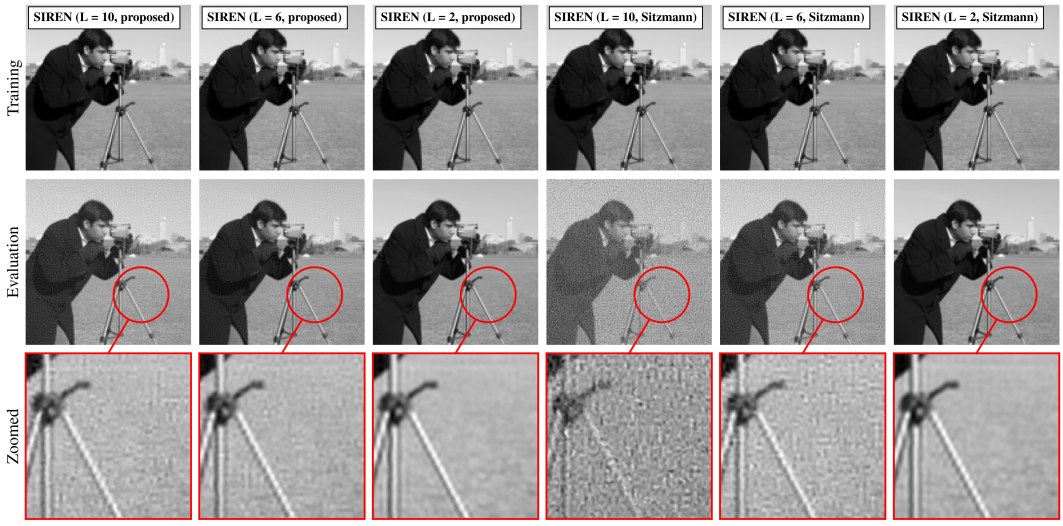

Figure 7: Comparison of the original and proposed SIREN initialization (presented in equations 7-8) on an image-fitting problem using a neural network of varying depth and of width $N = 256$. We train the model on a set $(\boldsymbol{x}_i, \boldsymbol{y}_i)_{i \in \mathbb{I}}$ where $\boldsymbol{x}_i$ is a location taken on a $|\mathbb{I}| = 128 \times 128$ uniformly spaced grid on $[-1, 1]^2$ and y_i is the associated image value at this location. The top row shows the fitted 128×128 image. The middle row shows the estimation on an augmented resolution (512×512) to assess the model's generalization and the last row provides a zoom on part of the image. In all case, we use ADAM optimizer with learning rate 10^{-4} for 10 000 epochs.

B.2 1D FITTING EXPERIMENTS

For the 1D fitting experiments, we generated synthetic data by sampling from a multi-scale function:

$$f_{1d}(x) = \sin(3x) + 0.7\cos(8x) + 0.3\sin(40x + 1) + \exp(-x^2)$$

To explore the impact of initialization on the performance of various neural network architectures, we studied two tasks: function fitting and PDE solving. Since image and video fitting reduce to function fitting, we focus on it. This choice lets us control the target function's frequency content. As a result, we can probe the different scales present in the data.

The results plotted figure 8 show that our proposed initialization matches or exceeds the accuracy of the traditional SIREN architecture for fitting a function. Moreover, it delivers significantly lower generalization error compared to the original SIREN. Notably, the Tanh-based positional-encoding network also shows strong generalization performance, despite its slightly higher training error.

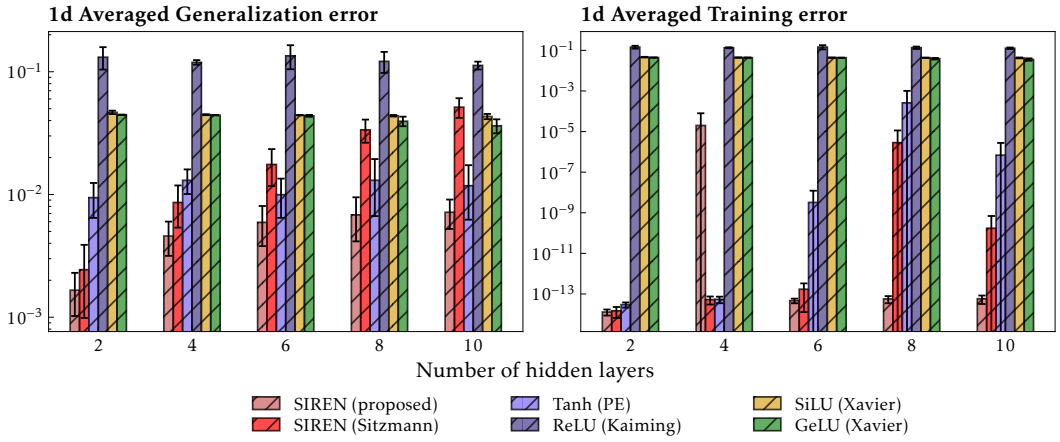

Figure 8: 1d Averaged generalization and training error for the 1D fitting problem. The results are averaged over 10 runs for each architecture of width $N = 128$. The error bars represent the standard deviation of the results.

B.3 2D Fitting Experiments

We applied the same methodology to a two-dimensional, multi-scale test function:

$$f_{2d}(x, y) = \sin(3x)\cos(3y) + \sin(15x - 2)\cos(15y) + \exp\big(-(x^2 + y^2)\big),$$

for $(x, y) \in [-1, 1]^2$. The exponential term ensures no architecture can represent the function trivially. We sampled 3600 random training points, giving a Nyquist frequency above 15. Each network was trained for 5000 epochs using Adam (learning rate 10^{-4}) under various initialization schemes. We then evaluated generalization error on 10 000 test points. The comparative results appear in Fig. 9.

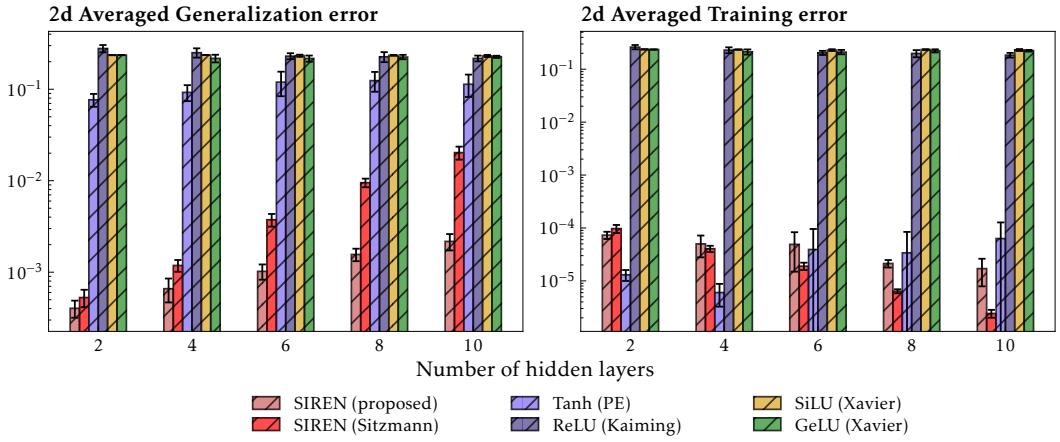

Figure 9: 2d Averaged generalization and training error for the 2D fitting problem. The results are averaged over 10 runs for each architecture of width $N = 1238$. The error bars represent the standard deviation of the results.

The results mirror the 1D fitting experiments. Our proposed initialization clearly outperforms all other architectures on the generalization task. At the same time, it maintains a very low training error, comparable to the SIREN architecture.

B.4 3D Fitting Experiments

For the 3D fitting experiments, we use the same framework as in 1D and 2D. We test a three-dimensional function with multi-scale features:

$$f_{3d}(x, y, z) = \sin(5x)\cos(12y)\sin(3z) + \exp\big(-(x^2 + y^2 + z^2)\big),$$

for $(x, y, z) \in [-1, 1]^3$. The exponential term prevents trivial representation by any architecture. We sample 8000 random training points, ensuring a Nyquist frequency above 12. Each network trains for 5000 epochs using Adam with learning rate 10^{-4} under various initialization schemes. We then evaluate generalization error on 70 000 test points. The results appear in Fig. 10.

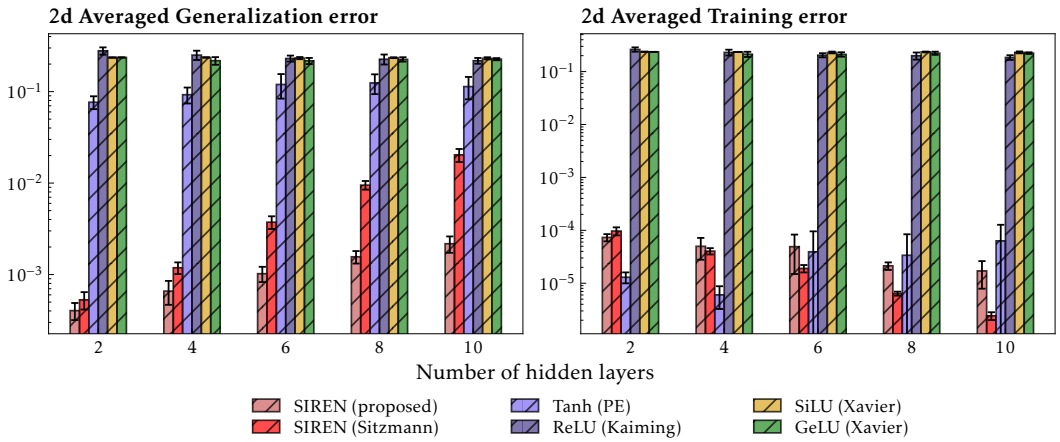

Figure 10: 3d Averaged generalization and training error for the 2D fitting problem. The results are averaged over 10 runs for each architecture of width $N = 128$. The error bars represent the standard deviation of the results.

Once again, our proposed initialization delivers strong results. It clearly outperforms all other architectures on generalization. Its fitting error remains very low, only slightly above the classic SIREN. Interestingly, as the number of layers increases, SIREN's training error decreases alongside rising high-frequency content. This suggests that fitting high frequencies may harm generalization—a drawback our method avoids.