Implicit Neural Representations with Periodic Activation Functions

Vincent Sitzmann[*]
sitzmann@cs.stanford.edu

Julien N. P. Martel[*]
jnmartel@stanford.edu

Alexander W. Bergman
awb@stanford.edu

David B. Lindell
lindell@stanford.edu

Gordon Wetzstein
gordon.wetzstein@stanford.edu

Stanford University
vsitzmann.github.io/siren/

Abstract

Implicitly defined, continuous, differentiable signal representations parameterized by neural networks have emerged as a powerful paradigm, offering many possible benefits over conventional representations. However, current network architectures for such implicit neural representations are incapable of modeling signals with fine detail. They also fail to accurately model spatial and temporal derivatives, which is necessary to represent signals defined implicitly by differential equations. We propose to leverage periodic activation functions for implicit neural representations and demonstrate that these networks, dubbed sinusoidal representation networks or SIRENs, are ideally suited for representing complex natural signals and their derivatives. We analyze SIREN activation statistics to propose a principled initialization scheme and demonstrate the representation of images, wavefields, video, sound, three-dimensional shapes, and their derivatives. Further, we show how SIRENs can be leveraged to solve challenging boundary value problems, such as particular Eikonal equations (yielding signed distance functions), the Poisson equation, and the Helmholtz and wave equations. Lastly, we combine SIRENs with hypernetworks to learn priors over the space of SIREN functions. Please see the project website for a video overview of the proposed method and all applications.

1 Introduction

We are interested in a class of functions Φ that satisfy equations of the form:

$$\mathcal{C}\left(\mathbf{x}, \Phi, \nabla_{\mathbf{x}}\Phi, \nabla^2_{\mathbf{x}}\Phi, \ldots\right) = 0, \quad \Phi : \mathbf{x} \mapsto \Phi(\mathbf{x}). \tag{1}$$

In this implicit problem formulation, a functional $\mathcal{C}$ takes as input the spatial or spatio-temporal coordinates $\mathbf{x} \in \mathbb{R}^m$ and, optionally, derivatives of Φ with respect to these coordinates. Our goal is then to learn a neural network that parameterizes Φ to map $\mathbf{x}$ to some quantity of interest while satisfying the constraint presented in Equation (1). Thus, Φ is implicitly defined by the relation modeled by $\mathcal{C}$ and we refer to neural networks that parameterize such implicitly defined functions as *implicit neural representations*. As we show in this paper, a surprisingly wide variety of problems across scientific fields fall into this form, such as modeling many different types of discrete signals in image, video, and audio processing using a continuous and differentiable representation, learning 3D shape representations via signed distance functions [1–4], and, more generally, solving boundary value problems, such as the Poisson, Helmholtz, or wave equations.

[*] These authors contributed equally to this work.

34th Conference on Neural Information Processing Systems (NeurIPS 2020), Vancouver, Canada.

A continuous parameterization offers several benefits over alternatives, such as discrete grid-based representations. For example, due to the fact that Φ is defined on the continuous domain of $\mathbf{x}$, it can be significantly more memory efficient than a discrete representation, allowing it to model fine detail that is not limited by the grid resolution but by the capacity of the underlying network architecture. As an example, we show how our SIREN architecture can represent complex 3D shapes with networks using only a few hundred kilobytes whereas naive mesh representations of the same datasets require hundreds of megabytes. Being differentiable implies that gradients and higher-order derivatives can be computed analytically, for example using automatic differentiation, which again makes these models independent of conventional grid resolutions. Finally, with well-behaved derivatives, implicit neural representations may offer a new toolbox for solving inverse problems, such as differential equations.

For these reasons, implicit neural representations have seen significant research interest over the last year (Sec. 2). Most of these recent representations build on ReLU-based multilayer perceptrons (MLPs). While promising, these architectures lack the capacity to represent fine details in the underlying signals, and they typically do not represent the derivatives of a target signal well. This is partly due to the fact that ReLU networks are piecewise linear, their second derivative is zero everywhere, and they are thus incapable of modeling information contained in higher-order derivatives of natural signals. While alternative activations, such as tanh or softplus, are capable of representing higher-order derivatives, we demonstrate that their derivatives are often not well behaved and also fail to represent fine details.

To address these limitations, we leverage MLPs with periodic activation functions for implicit neural representations. We demonstrate that this approach is not only capable of representing details in the signals better than ReLU-MLPs, or positional encoding strategies proposed in concurrent work [5], but that these properties also uniquely apply to the derivatives, which is critical for many applications we explore in this paper.

To summarize, the contributions of our work include:

- A continuous implicit neural representation using periodic activation functions that fits complicated signals, such as natural images and 3D shapes, and their derivatives robustly.
- An initialization scheme for training these representations and validation that distributions of these representations can be learned using hypernetworks.
- Demonstration of applications in image, video, and audio representation; 3D shape reconstruction; solving first-order differential equations to estimate a signal from its gradients; and solving second-order differential equations.

2 Related Work

Implicit neural representations. Recent work has demonstrated the potential of fully connected networks as continuous, memory-efficient implicit representations for shape parts [6, 7], objects [1, 4, 8, 9], or scenes [10–12]. These representations are typically trained from some form of 3D data as either signed distance functions [1, 4, 8–12] or occupancy networks [2, 13]. In addition to representing shape, some of these models have been extended to also encode object appearance [3, 5, 10, 14, 15], which can be trained using (multiview) 2D image data using neural rendering [16]. Temporally aware extensions [17] and variants that add part-level semantic segmentation [18] have also been proposed.

Periodic nonlinearities. Periodic nonlinearities have been investigated repeatedly over the past decades, but have so far failed to robustly outperform alternative activation functions. Early work includes Fourier neural networks, engineered to mimic the Fourier transform via single-hidden-layer networks [19, 20]. Other work explores neural networks with periodic activations for simple classification tasks [21–23], equation learning [24], and recurrent neural networks [25–29]. For such models, the training dynamics have been investigated [30], and it has been shown that they have universal function approximation properties [31–33]. Compositional pattern producing networks [34, 35] also leverage periodic nonlinearities, but rely on a combination of different nonlinearities via evolution in a genetic algorithm framework. Motivated by the discrete cosine transform, Klocek et al. [36] leverage cosine activation functions for image representation but they do not study the derivatives of these representations or other applications explored in our work. Inspired by these and

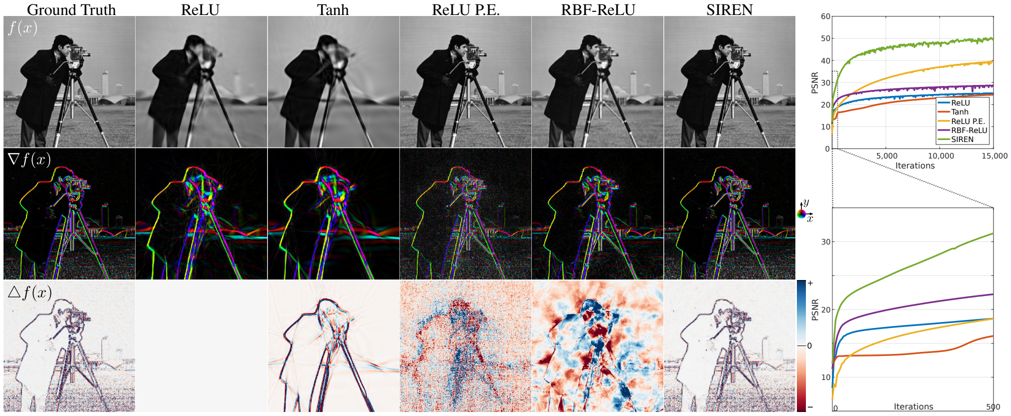

Figure 1: Comparison of different neural network architectures fitting the implicit representation of an image (ground truth: top left). The representation is only supervised on the target image but we also show first- and second-order derivatives of the function fit in rows 2 and 3, respectively.

other seminal works, we explore MLPs with periodic activation functions for applications involving implicit neural representations and their derivatives, and we propose principled initialization and generalization schemes.

Neural DE Solvers. Neural networks have long been investigated in the context of solving differential equations (DEs) [37], and have previously been introduced as implicit representations for this task [38]. Early work on this topic involved simple neural network models, consisting of MLPs or radial basis function networks with few hidden layers and hyperbolic tangent or sigmoid nonlinearities [38–41]. The limited capacity of these shallow networks typically constrained results to 1D solutions or simple 2D surfaces. Modern approaches to these techniques leverage recent optimization frameworks and auto-differentiation, but use similar architectures based on MLPs. Still, solving more sophisticated equations with higher dimensionality, more constraints, or more complex geometries is feasible [42–45]. However, we show that the commonly used MLPs with smooth, non-periodic activation functions fail to accurately model high-frequency information and higher-order derivatives even with dense supervision.

Neural ODEs [46] are related to this topic, but are very different in nature. Whereas implicit neural representations can be used to directly solve ODEs or PDEs from supervision on the system dynamics, neural ODEs allow for continuous function modeling by pairing a conventional ODE solver (e.g., implicit Adams or Runge-Kutta) with a network that parameterizes the dynamics of a function. The proposed architecture may be complementary to this line of work.

3 Formulation

Our goal is to solve problems of the form presented in Equation (1). We cast this as a feasibility problem, where a function Φ is sought that satisfies a set of M constraints $\{\mathcal{C}_m(\mathbf{a}(\mathbf{x}), \Phi(\mathbf{x}), \nabla\Phi(\mathbf{x}), ...)\}_{m=1}^{M}$, each of which relate the function Φ and/or its derivatives to quantities $\mathbf{a}(\mathbf{x})$:

$$\text{find } \Phi \text{ subject to } \mathcal{C}_m\big(\mathbf{a}(\mathbf{x}), \Phi(\mathbf{x}), \nabla\Phi(\mathbf{x}), ...\big) = 0,\ \forall \mathbf{x} \in \Omega_m,\ m = 1, \ldots, M \tag{2}$$

This problem can be cast in a loss function that penalizes deviations from each of the constraints on their domain Ω_m:

$$\mathcal{L} = \int_{\Omega} \sum_{m=1}^{M} \mathbf{1}_{\Omega_m}(\mathbf{x})\, \|\mathcal{C}_m(\mathbf{a}(\mathbf{x}), \Phi(\mathbf{x}), \nabla\Phi(\mathbf{x}), ...)\|\, \mathrm{d}\mathbf{x}, \tag{3}$$

with the indicator function $\mathbf{1}_{\Omega_m}(\mathbf{x}) = 1$ when $\mathbf{x} \in \Omega_m$ and 0 when $\mathbf{x} \notin \Omega_m$. In practice, the loss function is enforced by sampling Ω. A dataset $\mathcal{D} = \{(\mathbf{x}_i, \mathbf{a}_i(\mathbf{x}))\}_i$ is a set of tuples of coordinates $\mathbf{x}_i \in \Omega$ along with samples from the quantities $\mathbf{a}(\mathbf{x}_i)$ that appear in the constraints. Thus, the loss in Equation (3) is enforced on coordinates $\mathbf{x}_i$ sampled from the dataset, yielding the loss

$\tilde{\mathcal{L}} = \sum_{i\in\mathcal{D}} \sum_{m=1}^{M} \mathbf{1}_{\Omega_m}(\mathbf{x}) \, \|\mathcal{C}_m(\mathbf{a}(\mathbf{x}_i), \Phi(\mathbf{x}_i), \nabla\Phi(\mathbf{x}_i), ...)\|$. In practice, the dataset $\mathcal{D}$ is sampled dynamically at training time, approximating $\mathcal{L}$ better as the number of samples grows, as in Monte Carlo integration.

We parameterize functions Φ_θ as fully connected neural networks with parameters θ, and solve the resulting optimization problem using gradient descent. The derivatives in Eq. (3) such as $\nabla_\mathbf{x}\Phi_\theta$ correspond to the gradient of the network's outputs with respect to its inputs. Those can be computed with auto-differentiation [47]. They are automatically added to the computation graph when defining the loss function, thus enabling the optimization of the weights θ during training.

3.1 Periodic Activations for Implicit Neural Representations

We propose SIREN, a simple neural network architecture for implicit neural representations that uses the sine as a periodic activation function:

$$\Phi\left(\mathbf{x}\right) = \mathbf{W}_n\left(\phi_{n-1} \circ \phi_{n-2} \circ \ldots \circ \phi_0\right)\left(\mathbf{x}\right) + \mathbf{b}_n, \quad \phi_i\left(\mathbf{x}_i\right) = \sin\left(\mathbf{W}_i\mathbf{x}_i + \mathbf{b}_i\right). \tag{4}$$

Here, $\phi_i : \mathbb{R}^{M_i} \mapsto \mathbb{R}^{N_i}$ is the i^{th} layer of the network. It consists of the affine transform defined by the weight matrix $\mathbf{W}_i \in \mathbb{R}^{N_i \times M_i}$ and the biases $\mathbf{b}_i \in \mathbb{R}^{N_i}$ applied on the input $\mathbf{x}_i \in \mathbb{R}^{M_i}$, followed by the sine nonlinearity applied to each component of the resulting vector.

Interestingly, any derivative of a SIREN *is itself a composition of* SIRENs, as the derivative of the sine is a cosine, i.e., a phase-shifted sine (see supplemental). Therefore, the derivatives of a SIREN inherit the properties of SIRENs, enabling us to supervise any derivative of SIREN with "complicated" signals. In our experiments, we demonstrate that when a SIREN is supervised using a constraint $\mathcal{C}_m$ involving the derivatives of Φ, the function realized by the neural network Φ_θ remains well behaved, which is crucial in solving many problems, including boundary value problems (BVPs). In contrast to conventional nonlinearities such as the hyperbolic tangent or the ReLU, the sine is periodic and therefore, non-local. Intuitively, this provides SIREN with a degree of shift invariance, as it may learn to apply the same function to different input coordinates.

We will show that SIRENs can be initialized with some control over the distribution of activations, allowing us to create deep architectures. Furthermore, SIRENs converge significantly faster than baseline architectures, fitting, for instance, a single image in a few hundred iterations, taking a few seconds on a modern GPU, while featuring higher image fidelity (Fig. 1).

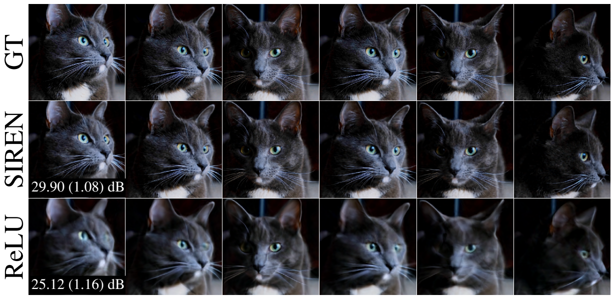

Figure 2: Example frames from fitting a video with SIREN and ReLU-MLPs. Our approach faithfully reconstructs fine details like the whiskers. Mean (and standard deviation) of the PSNR over all frames is reported.

A simple example: fitting an image. Consider the case of finding the function $\Phi : \mathbb{R}^2 \mapsto \mathbb{R}^3$ that parameterizes a given discrete image f in a continuous fashion. The image defines a dataset $\mathcal{D} = \{(\mathbf{x}_i, f(\mathbf{x}_i))\}_i$ of pixel coordinates $\mathbf{x}_i = (x_i, y_i)$ associated with their RGB colors $f(\mathbf{x}_i)$. The only constraint $\mathcal{C}$ is that Φ should output image colors at pixel coordinates. $\mathcal{C}$ solely depends on both Φ (none of its derivatives) and $f(\mathbf{x}_i)$, with the form: $\mathcal{C}(f(\mathbf{x}_i), \Phi(\mathbf{x})) = \Phi(\mathbf{x}_i) - f(\mathbf{x}_i)$ which can be translated into the loss $\tilde{\mathcal{L}} = \sum_i \|\Phi(\mathbf{x}_i) - f(\mathbf{x}_i)\|$. In Fig. 1, we fit Φ_θ using comparable network architectures with different activation functions to a natural image. We supervise this experiment only on the image values, but also visualize the gradients $\nabla\Phi$ and Laplacians $\Delta\Phi$. While only two approaches, a ReLU network with positional encoding (P.E.) [5, 48] and our SIREN, accurately represent the ground truth image $f\left(\mathbf{x}\right)$, SIREN is the only network capable of also representing the derivatives of the signal. Additionally, we run a simple experiment where we fit a short video with 300 frames and with a resolution of 512×512 pixels using both ReLU and SIREN MLPs. As seen in Figure 2, our approach is successful in representing this video with an average peak signal-to-noise ratio close to 30 dB, outperforming the ReLU baseline by about 5 dB. We also show the flexibility of SIRENs by representing audio signals in the supplement.

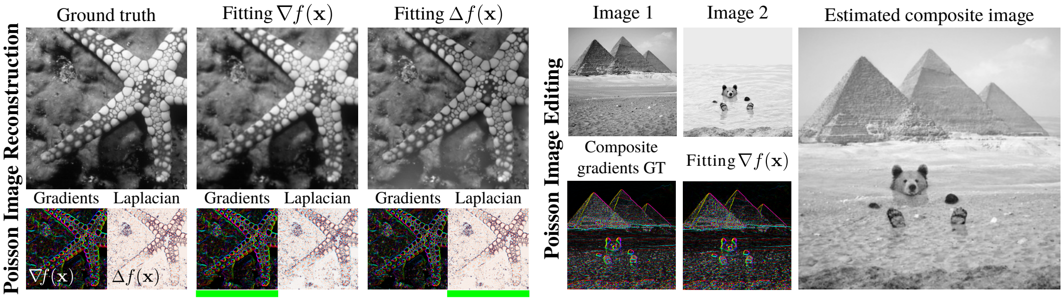

Figure 3: **Poisson image reconstruction:** An image (left) is reconstructed by fitting a SIREN, supervised either by its gradients or Laplacians (underlined in green). The results, shown in the center and right, respectively, match both the image and its derivatives well. **Poisson image editing:** The gradients of two images (top) are fused (bottom left). SIREN allows for the composite (right) to be reconstructed using supervision on the gradients (bottom right).

3.2 Distribution of activations, frequencies, and a principled initialization scheme

We present a principled initialization scheme necessary for the effective training of SIRENs. While presented informally here, we discuss further details, proofs and empirical validation in the supplemental material. The key idea in our initialization scheme is to preserve the distribution of activations through the network so that the final output at initialization does not depend on the number of layers. Note that building SIRENs without carefully chosen weights yielded poor performance both in accuracy and in convergence speed.

To this end, let us first consider the output distribution of a single sine neuron with the uniformly distributed input $x \sim \mathcal{U}(-1, 1)$. The neuron's output is $y = \sin(ax + b)$ with $a, b \in \mathbb{R}$. It can be shown that for any $a > \frac{\pi}{2}$, i.e. spanning at least half a period, the output of the sine is $y \sim \text{arcsin}(-1, 1)$, a special case of a U-shaped Beta distribution and independent of the choice of b. We can now reason about the output distribution of a neuron. Taking the linear combination of n inputs $\mathbf{x} \in \mathbb{R}^n$ weighted by $\mathbf{w} \in \mathbb{R}^n$, its output is $y = \sin\big(\mathbf{w}^T\mathbf{x} + b\big)$. Assuming this neuron is in the second layer, each of its inputs is arcsine distributed. When each component of $\mathbf{w}$ is uniformly distributed such as $w_i \sim \mathcal{U}(-c/\sqrt{n}, c/\sqrt{n}), c \in \mathbb{R}$, we show (see supplemental) that the dot product converges to the normal distribution $\mathbf{w}^T\mathbf{x} \sim \mathcal{N}(0, c^2/6)$ as n grows. Finally, feeding this normally distributed dot product through another sine is also arcsine distributed for any $c > \sqrt{6}$. Note that the weights of a SIREN can be interpreted as angular frequencies while the biases are phase offsets. Thus, larger frequencies appear in the networks for weights with larger magnitudes. For $|\mathbf{w}^T\mathbf{x}| < \pi/4$, the sine layer will leave the frequencies unchanged, as the sine is approximately linear. In fact, we empirically find that a sine layer keeps spatial frequencies approximately constant for amplitudes such as $|\mathbf{w}^T\mathbf{x}| < \pi$, and increases spatial frequencies for amplitudes above this value[2].

Hence, we propose to draw weights with $c = \sqrt{6}$ so that $w_i \sim \mathcal{U}(-\sqrt{6/n}, \sqrt{6/n})$. This ensures that the input to each sine activation is normal distributed with a standard deviation of 1. Since only a few weights have a magnitude larger than π, the frequency throughout the sine network grows only slowly. Finally, we propose to initialize the first layer of the sine network with weights so that the sine function $\sin(\omega_0 \cdot \mathbf{W}\mathbf{x} + \mathbf{b})$ spans multiple periods over $[-1, 1]$. We found $\omega_0 = 30$ to work well for all the applications in this work. The proposed initialization scheme yielded fast and robust convergence using the ADAM optimizer for all experiments in this work.

4 Experiments

In this section, we leverage SIRENs to solve challenging boundary value problems using different types of supervision of the derivatives of Φ. We first solve the Poisson equation via direct supervision of its derivatives. We then solve a particular form of the Eikonal equation, placing a unit-norm constraint on gradients, parameterizing the class of signed distance functions (SDFs). SIREN significantly

[2] Formalizing the distribution of output frequencies throughout SIRENs proves to be a hard task and is out of the scope of this work.

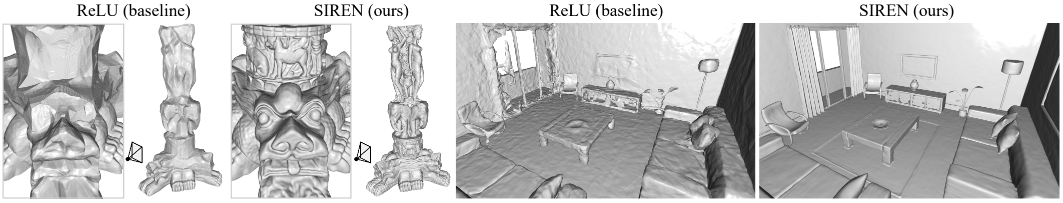

Figure 4: Shape representation. We fit signed distance functions parameterized by implicit neural representations directly on point clouds. Compared to ReLU implicit representations, our periodic activations significantly improve detail of objects (left) and complexity of entire scenes (right).

outperforms ReLU-based representations of SDFs, capturing large scenes at a high level of detail. We then solve the second-order Helmholtz partial differential equation, and the challenging inverse problem of full-waveform inversion. Finally, we combine SIRENs with hypernetworks, learning a prior over the space of parameterized functions. Those experiments are summarized in Section 4 of the supplemental, and additional experiments and details can be found in Section 5–11 in the supplemental. All code and data is publicly available on the project webpage[3].

4.1 Solving the Poisson Equation

We demonstrate that the proposed representation is not only able to accurately represent a function and its derivatives, but that it can also be supervised solely by its derivatives, i.e., the model is never presented with the actual function values, but only values of its first or higher-order derivatives.

An intuitive example representing this class of problems is the Poisson equation. The Poisson equation is perhaps the simplest elliptic partial differential equation (PDE) which is crucial in physics and engineering, for example to model potentials arising from distributions of charges or masses. In this problem, an unknown ground truth signal f is estimated from discrete samples of either its gradients ∇f or Laplacian $\Delta f = \nabla \cdot \nabla f$ as

$$\mathcal{L}_{\text{grad.}} = \int_{\Omega} \|\nabla_{\mathbf{x}}\Phi(\mathbf{x}) - \nabla_{\mathbf{x}} f(\mathbf{x})\| \, d\mathbf{x}, \quad \text{or} \quad \mathcal{L}_{\text{lapl.}} = \int_{\Omega} \|\Delta\Phi(\mathbf{x}) - \Delta f(\mathbf{x})\| \, d\mathbf{x}. \tag{5}$$

Poisson image reconstruction. Solving the Poisson equation enables the reconstruction of images from their derivatives. We show results of this approach using SIREN in Fig. 3. Supervising the implicit representation with either ground truth gradients via $\mathcal{L}_{\text{grad.}}$ or Laplacians via $\mathcal{L}_{\text{lapl.}}$ successfully reconstructs the image. Remaining intensity variations are due to the ill-posedness of the problem.

Poisson image editing. Images can be seamlessly fused in the gradient domain [49]. For this purpose, Φ is supervised using $\mathcal{L}_{\text{grad.}}$ of Eq. (5), where $\nabla_{\mathbf{x}} f(\mathbf{x})$ is a composite function of the gradients of two images $f_{1,2}$: $\nabla_{\mathbf{x}} f(\mathbf{x}) = \alpha \cdot \nabla f_1(x) + (1-\alpha) \cdot \nabla f_2(x),\ \alpha \in [0,1]$. Fig. 3 shows two images seamlessly fused with this approach.

4.2 Representing Shapes with Signed Distance Functions

Inspired by recent work on shape representation with differentiable signed distance functions (SDFs) [1, 4, 9], we fit SDFs directly on oriented point clouds using both ReLU-based implicit neural representations and SIRENs. This amounts to solving a particular Eikonal boundary value problem that constrains the norm of spatial gradients $|\nabla_{\mathbf{x}}\Phi|$ to be 1 almost everywhere. Note that ReLU networks are seemingly ideal for representing SDFs, as their gradients are locally constant and their second derivatives are 0. Solving the Eikonal equation with an implicit neural representation with ReLU activations was previously proposed in [9]. We fit a SIREN to an oriented point cloud using a loss of the form

$$\mathcal{L}_{\text{sdf}} = \int_{\Omega} \big\| \, |\nabla_{\mathbf{x}}\Phi(\mathbf{x})| - 1 \big\| \, d\mathbf{x} + \int_{\Omega_0} \|\Phi(\mathbf{x})\| + \big(1 - \langle \nabla_{\mathbf{x}}\Phi(\mathbf{x}), \mathbf{n}(\mathbf{x}) \rangle\big) \, d\mathbf{x} + \int_{\Omega \setminus \Omega_0} \psi\big(\Phi(\mathbf{x})\big) \, d\mathbf{x}, \tag{6}$$

[3] https://vsitzmann.github.io/siren/

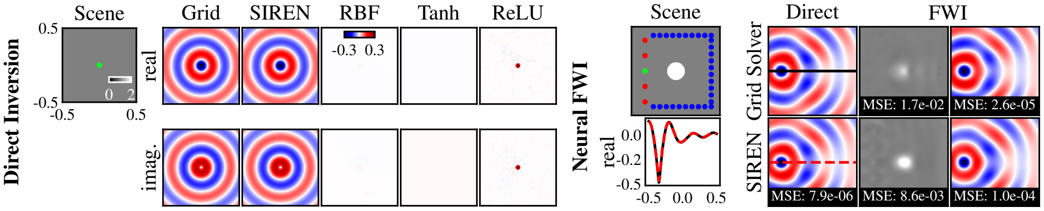

Figure 5: **Direct Inversion:** We solve the Helmholtz equation for a single point source placed at the center of a medium (green dot) with uniform wave propagation velocity (top left). The SIREN solution closely matches a principled grid solver [52] while other network architectures fail to find the correct solution. **Neural Full-Waveform Inversion (FWI):** A scene contains a source (green) and a circular wave velocity perturbation centered at the origin (top left). With the scene velocity known *a priori*, SIREN directly reconstructs a wavefield that closely matches a principled grid solver [52] (bottom left, middle left). For FWI, the velocity and wavefields are reconstructed with receiver measurements (blue dots) from sources triggered in sequence (green, red dots). The SIREN velocity model outperforms a principled FWI solver [53], accurately predicting wavefields. FWI MSE values are calculated across all wavefields and the visualized real wavefield corresponds to the green source.

Here, $\psi(\Phi(\mathbf{x})) = \exp(-\alpha \cdot |\Phi(\mathbf{x})|), \alpha \gg 1$ penalizes off-surface points for creating SDF values close to 0. Ω is the whole domain and we denote the zero-level set of the SDF as Ω_0. The model $\Phi(\mathbf{x})$ is supervised using oriented points sampled on a mesh, where we require the SIREN to respect $\Phi(\mathbf{x}) = 0$ and its normals $\mathbf{n}(\mathbf{x}) = \nabla f(\mathbf{x})$. During training, each minibatch contains an equal number of points on and off the mesh, each one randomly sampled over Ω. As seen in Fig. 4, the proposed periodic activations significantly increase the details of objects and the complexity of scenes that can be represented by these neural SDFs, parameterizing a full room from the ICL-NUIM dataset [50] with only a single five-layer fully connected neural network. This is in contrast to concurrent work that addresses the same failure of conventional MLP architectures to represent complex or large scenes by locally decoding a discrete representation, such as a voxelgrid, into an implicit neural representation [11, 12, 51]. We note that the resulting representations can be quite compact. For instance, the Thai statue shown in Figure 4 is reconstructed at a high fidelity while requiring only 260 kB while the naive mesh representation of this dataset requires 293 MB. Similarly, the SIREN representation of the room requires only about 1 MB whereas the naive mesh representation requires 579 MB. Please refer to the supplemental material for additional discussions on compression capabilities of SIREN.

4.3 Solving the Helmholtz and Wave Equations

The Helmholtz and wave equations are second-order partial differential equations related to the physical modeling of diffusion and waves. They are closely related through a Fourier-transform relationship, with the Helmholtz equation given as

$$H(m)\,\Phi(\mathbf{x}) = -f(\mathbf{x}), \text{ with } H(m) = \left(\Delta + m(\mathbf{x})\,w^2\right). \tag{7}$$

Here, $f(\mathbf{x})$ represents a known source function, $\Phi(\mathbf{x})$ is the unknown wavefield, and the squared slowness $m(\mathbf{x}) = 1/c(\mathbf{x})^2$ is a function of the wave velocity $c(\mathbf{x})$. In general, the solutions to the Helmholtz equation are complex-valued and require numerical solvers to compute. As the Helmholtz and wave equations follow a similar form, we discuss the Helmholtz equation here, with additional results and discussion for the wave equation in the supplement.

Solving for the wavefield. We solve for the wavefield by parameterizing $\Phi(\mathbf{x})$ with a SIREN. To accommodate a complex-valued solution, we configure the network to output two values, interpreted as the real and imaginary parts. Training is performed on randomly sampled points $\mathbf{x}$ within the domain $\Omega = \{\mathbf{x} \in \mathbb{R}^2 \mid \|\mathbf{x}\|_\infty < 1\}$. The network is supervised using a loss function based on the Helmholtz equation:

$$\mathcal{L}_{\text{Helmholtz}} = \int_\Omega \lambda(\mathbf{x})\,\|H(m)\Phi(\mathbf{x}) + f(\mathbf{x})\|\,\mathrm{d}\mathbf{x},$$

with $\lambda(\mathbf{x}) = k$, a hyperparameter, when $f(\mathbf{x}) \neq 0$ (corresponding to the inhomogeneous contribution to the Helmholtz equation) and $\lambda(\mathbf{x}) = 1$ otherwise (for the homogenous part). Each minibatch

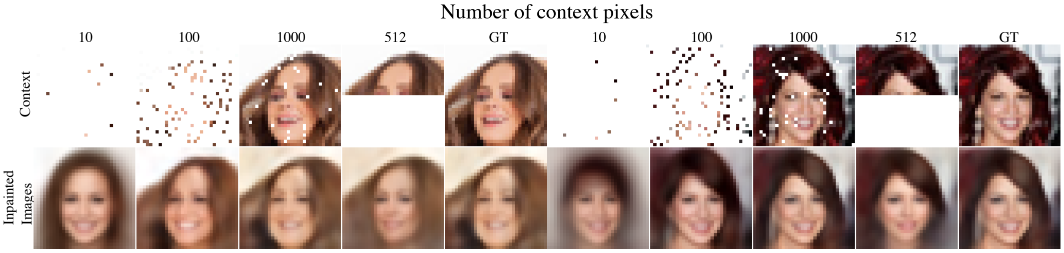

Figure 6: Generalizing across implicit functions parameterized by SIRENs on the CelebA dataset [56]. Image inpainting results are shown for various numbers of context pixels in O_j.

contains samples from both contributions and k is set so the losses are approximately equal at the beginning of training. In practice, we use a slightly modified form of Equation (7) to include the perfectly matched boundary conditions that are necessary to ensure a unique solution [52] (see supplement for details).

Results are shown in Fig. 5 for solving the Helmholtz equation in two dimensions with spatially uniform wave velocity and a single point source (modeled as a Gaussian with $\sigma^2 = 10^{-4}$). The SIREN solution is compared with a principled solver [52] as well as other neural network solvers. All evaluated network architectures use the same number of hidden layers as SIREN but with different activation functions. In the case of the RBF network, we prepend a Gaussian RBF layer with 1024 hidden units and use a tanh activation for all the other layers. SIREN is the only representation capable of producing a high-fidelity reconstruction of the wavefield. We also note that the tanh network has a similar architecture to recent work on neural PDE solvers [44], except we increase the network size to match SIREN.

Neural full-waveform inversion (FWI). In many wave-based sensing modalities (radar, sonar, seismic imaging, etc.), one attempts to probe and sense across an entire domain using sparsely placed sources (i.e., transmitters) and receivers. FWI uses the known locations of sources and receivers to jointly recover the entire wavefield and other physical properties, such as permittivity, density, or wave velocity. Specifically, the FWI problem can be described as [54]

$$\underset{m,\Phi}{\arg\min} \sum_{i=1}^{N} \int_{\Omega} \|\text{Ш}_r(\Phi_i(\mathbf{x}) - r_i(\mathbf{x}))\| \, d\mathbf{x} \text{ s.t. } H(m)\,\Phi_i(\mathbf{x}) = -f_i(\mathbf{x}),\ 1 \le i \le N,\ \forall \mathbf{x} \in \Omega, \quad (8)$$

where there are N sources, Ш_r samples the wavefield at the receiver locations, and $r_i(x)$ models receiver data for the ith source.

We first use a SIREN to directly solve Eq. 7 for a known wave velocity perturbation, obtaining an accurate wavefield that closely matches that of a principled solver [52] (see Fig. 5, right). Without *a priori* knowledge of the velocity field, FWI is used to jointly recover the wavefields and velocity. Here, we use 5 sources and place 30 receivers around the domain, as shown in Fig. 5. Using the principled solver, we simulate the receiver measurements for the 5 wavefields (one for each source) at a single frequency of 3.2 Hz, which is chosen to be relatively low for improved convergence. We pre-train SIREN to output 5 complex wavefields and a squared slowness value for a uniform velocity. Then, we optimize for the wavefields and squared slowness using a penalty method variation [54] of Eq. 8 (see the supplement for additional details). In Fig. 5, we compare to an FWI solver based on the alternating direction method of multipliers [53, 55]. With only a single frequency for the inversion, the principled solver is prone to converge to a poor solution for the velocity. As shown in Fig. 5, SIREN converges to a better velocity solution and accurate solutions for the wavefields. All reconstructions are performed or shown at 256×256 resolution to avoid noticeable stair-stepping artifacts in the circular velocity perturbation.

4.4 Learning a Space of Implicit Functions

A powerful concept that has emerged for implicit representations is to learn priors over the space of functions that define them [1, 2, 10]. Here we demonstrate that the function space parameterized by SIRENs also admits the learning of powerful priors. Each of these SIRENS Φ_j are fully defined by their parameters $\boldsymbol{\theta}_j \in \mathbb{R}^l$. Assuming that all parameters $\boldsymbol{\theta}_j$ of a class exist in a k-dimensional subspace of $\mathbb{R}^l$, $k < l$, then these parameters can be well modeled by latent code vectors in $\mathbf{z} \in \mathbb{R}^k$. Like in neural processes [57–59], we condition these latent code vectors on partial observations of the signal $O \in \mathbb{R}^m$ through an encoder

$$C : \mathbb{R}^m \to \mathbb{R}^k, \quad O_j \mapsto C(O_j) = \mathbf{z}_j, \tag{9}$$

and use a ReLU hypernetwork [60], to map the latent code to the weights of a SIREN, as in [10]:

$$\Psi : \mathbb{R}^k \to \mathbb{R}^l, \quad \mathbf{z}_j \mapsto \Psi(\mathbf{z}_j) = \boldsymbol{\theta}_j. \tag{10}$$

We replicated the experiment from [57] on the CelebA dataset [56] using a set encoder. Additionally, we show results using a convolutional neural network encoder which operates on sparse images. Interestingly, this improves the quantitative and qualitative performance on the inpainting task. At test time, this enables reconstruction from sparse pixel observations, and, thereby, inpainting. Fig. 6 shows test-time reconstructions from a varying number of pixel observations. Note that these inpainting results were all generated using the same model, with the same parameter values. Tab. 1 reports a quantitative comparison to [57], demonstrating that generalization over SIREN representations is at least equally as powerful as generalization over images.

Table 1: Quantitative comparison to Conditional Neural Processes [57] (CNPs) on the 32×32 CelebA test set. The pixel-wise mean squared errors are reported.

Number of Context Pixels	10	100	1000
CNP [57]	0.039	0.016	0.009
Set Encoder + Hypernet.	0.035	0.013	0.009
CNN Encoder + Hypernet.	**0.033**	**0.009**	**0.008**

5 Discussion and Conclusion

The question of how to represent a signal is at the core of many problems across science and engineering. Implicit neural representations may provide a new tool for many of these by offering a number of potential benefits over conventional continuous and discrete representations. We demonstrate that periodic activation functions are ideally suited for representing complex natural signals and their derivatives using implicit neural representations. We also prototype several boundary value problems that our framework is capable of solving robustly. There are several exciting avenues for future work, including the exploration of other types of inverse problems and applications in areas beyond implicit neural representations, for example neural ODEs [46]. While we demonstrate the feasibility of generalizing across signals represented by SIREN networks, the fidelity of the resulting representations is limited—investigating effective alternatives is an important direction for future work. An immediate application of SIREN may be the compression of large-scale 3D models, as SIREN may represent them at a high visual fidelity with a relativily small number of parameters and resulting small file sizes.

Concurrent work investigates directions related to our approach. Locally decoding a discrete voxelgrid into an implicit neural representation [11, 12, 51] similarly enables the representation of fine detail. Tancik et al. [61] extend the previously proposed first-layer positional encoding [5, 48] and investigate its properties through the perspective of the neural tangent kernel [62].

Broader Impact

The proposed SIREN representation enables accurate representations of natural signals, such as images, audio, and video in a deep learning framework. This may be an enabler for downstream tasks involving such signals, such as classification for images or speech-to-text systems for audio. Such applications may be leveraged for both positive and negative ends. SIREN may in the future further enable novel approaches to the generation of such signals. This has potential for misuse in impersonating actors without their consent. For an in-depth discussion of such so-called DeepFakes, we refer the reader to a recent review article on neural rendering [16].

Acknowledgments and Disclosure of Funding

V.S., A.W.B., and D.B.L. were supported by a Stanford Graduate Fellowship. J.N.P.M was supported by a Swiss National Science Foundation Fellowship (P2EZP2-181817). G.W. was supported by a Sloan Fellowship, by the NSF (award numbers 1553333 and 1839974), and a PECASE by the ARO.

References

[1] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation. In *Proc. CVPR*, 2019.

[2] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3d reconstruction in function space. In *Proc. CVPR*, 2019.

[3] Shunsuke Saito, Zeng Huang, Ryota Natsume, Shigeo Morishima, Angjoo Kanazawa, and Hao Li. Pifu: Pixel-aligned implicit function for high-resolution clothed human digitization. In *Proc. ICCV*, pages 2304–2314, 2019.

[4] Matan Atzmon and Yaron Lipman. Sal: Sign agnostic learning of shapes from raw data. In *Proc. CVPR*, 2020.

[5] 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 *Proc. ECCV*, 2020.

[6] Kyle Genova, Forrester Cole, Daniel Vlasic, Aaron Sarna, William T Freeman, and Thomas Funkhouser. Learning shape templates with structured implicit functions. In *Proc. ICCV*, pages 7154–7164, 2019.

[7] Kyle Genova, Forrester Cole, Avneesh Sud, Aaron Sarna, and Thomas Funkhouser. Deep structured implicit functions. *arXiv preprint arXiv:1912.06126*, 2019.

[8] Mateusz Michalkiewicz, Jhony K Pontes, Dominic Jack, Mahsa Baktashmotlagh, and Anders Eriksson. Implicit surface representations as layers in neural networks. In *Proc. ICCV*, pages 4743–4752, 2019.

[9] Amos Gropp, Lior Yariv, Niv Haim, Matan Atzmon, and Yaron Lipman. Implicit geometric regularization for learning shapes. In *Proc. ICML*, 2020.

[10] Vincent Sitzmann, Michael Zollhöfer, and Gordon Wetzstein. Scene representation networks: Continuous 3d-structure-aware neural scene representations. In *Proc. NeurIPS 2019*, 2019.

[11] Chiyu Jiang, Avneesh Sud, Ameesh Makadia, Jingwei Huang, Matthias Nießner, and Thomas Funkhouser. Local implicit grid representations for 3d scenes. In *Proc. CVPR*, pages 6001–6010, 2020.

[12] Songyou Peng, Michael Niemeyer, Lars Mescheder, Marc Pollefeys, and Andreas Geiger. Convolutional occupancy networks. In *Proc. ECCV*, 2020.

[13] Zhiqin Chen and Hao Zhang. Learning implicit fields for generative shape modeling. In *Proc. CVPR*, pages 5939–5948, 2019.

[14] Michael Oechsle, Lars Mescheder, Michael Niemeyer, Thilo Strauss, and Andreas Geiger. Texture fields: Learning texture representations in function space. In *Proc. ICCV*, 2019.

[15] Michael Niemeyer, Lars Mescheder, Michael Oechsle, and Andreas Geiger. Differentiable volumetric rendering: Learning implicit 3d representations without 3d supervision. In *Proc. CVPR*, 2020.

[16] Ayush Tewari, Ohad Fried, Justus Thies, Vincent Sitzmann, Stephen Lombardi, Kalyan Sunkavalli, Ricardo Martin-Brualla, Tomas Simon, Jason Saragih, Matthias Nießner, et al. State of the art on neural rendering. *Proc. Eurographics*, 2020.

[17] Michael Niemeyer, Lars Mescheder, Michael Oechsle, and Andreas Geiger. Occupancy flow: 4d reconstruction by learning particle dynamics. In *Proc. ICCV*, 2019.

[18] Amit Kohli, Vincent Sitzmann, and Gordon Wetzstein. Semantic implicit neural scene representations with semi-supervised training. *Proc. 3DV 2020*, 2020.

[19] Ronald Gallant and Halbert White. There exists a neural network that does not make avoidable mistakes. In *Proc. IEEE Int. Conference on Neural Networks*, pages 657–664, 1988.

[20] Abylay Zhumekenov, Malika Uteuliyeva, Olzhas Kabdolov, Rustem Takhanov, Zhenisbek Assylbekov, and Alejandro J Castro. Fourier neural networks: A comparative study. *arXiv preprint arXiv:1902.03011*, 2019.

[21] Josep M Sopena, Enrique Romero, and Rene Alquezar. Neural networks with periodic and monotonic activation functions: a comparative study in classification problems. In *Proc. ICANN*, 1999.

[22] Kwok-wo Wong, Chi-sing Leung, and Sheng-jiang Chang. Handwritten digit recognition using multilayer feedforward neural networks with periodic and monotonic activation functions. In *Object recognition supported by user interaction for service robots*, volume 3, pages 106–109. IEEE, 2002.

[23] Giambattista Parascandolo, Heikki Huttunen, and Tuomas Virtanen. *Taming the waves: sine as activation function in deep neural networks*, 2016. URL `https://openreview.net/forum?id=Sks3zF9eg`.

[24] Subham S Sahoo, Christoph H Lampert, and Georg Martius. Learning equations for extrapolation and control. *arXiv preprint arXiv:1806.07259*, 2018.

[25] Peng Liu, Zhigang Zeng, and Jun Wang. Multistability of recurrent neural networks with nonmonotonic activation functions and mixed time delays. *IEEE Trans. on Systems, Man, and Cybernetics: Systems*, 46 (4):512–523, 2015.

[26] Renée Koplon and Eduardo D Sontag. Using fourier-neural recurrent networks to fit sequential input/output data. *Neurocomputing*, 15(3-4):225–248, 1997.

[27] M Hisham Choueiki, Clark A Mount-Campbell, and Stanley C Ahalt. Implementing a weighted least squares procedure in training a neural network to solve the short-term load forecasting problem. *IEEE Trans. on Power systems*, 12(4):1689–1694, 1997.

[28] René Alquézar Mancho. *Symbolic and connectionist learning techniques for grammatical inference*. Universitat Politècnica de Catalunya, 1997.

[29] JM Sopena and R Alquezar. Improvement of learning in recurrent networks by substituting the sigmoid activation function. In *Proc. ICANN*, pages 417–420. Springer, 1994.

[30] Michal Rosen-Zvi, Michael Biehl, and Ido Kanter. Learnability of periodic activation functions: General results. *Physical Review E*, 58(3):3606, 1998.

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

[32] Shaobo Lin, Xiaofei Guo, Feilong Cao, and Zongben Xu. Approximation by neural networks with scattered data. *Applied Mathematics and Computation*, 224:29–35, 2013.

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

[34] Kenneth O Stanley. Compositional pattern producing networks: A novel abstraction of development. *Genetic programming and evolvable machines*, 8(2):131–162, 2007.

[35] Alexander Mordvintsev, Nicola Pezzotti, Ludwig Schubert, and Chris Olah. Differentiable image parameterizations. *Distill*, 3(7):e12, 2018.

[36] Sylwester Klocek, Łukasz Maziarka, Maciej Wołczyk, Jacek Tabor, Jakub Nowak, and Marek Śmieja. Hypernetwork functional image representation. In *Proc. ICANN*, pages 496–510. Springer, 2019.

[37] Hyuk Lee and In Seok Kang. Neural algorithm for solving differential equations. *Journal of Computational Physics*, 91(1):110–131, 1990.

[38] Isaac E Lagaris, Aristidis Likas, and Dimitrios I Fotiadis. Artificial neural networks for solving ordinary and partial differential equations. *IEEE Trans. on neural networks*, 9(5):987–1000, 1998.

[39] Shouling He, Konrad Reif, and Rolf Unbehauen. Multilayer neural networks for solving a class of partial differential equations. *Neural networks*, 13(3):385–396, 2000.

[40] Nam Mai-Duy and Thanh Tran-Cong. Approximation of function and its derivatives using radial basis function networks. *Applied Mathematical Modelling*, 27(3):197–220, 2003.

[41] Leah Bar and Nir Sochen. Unsupervised deep learning algorithm for pde-based forward and inverse problems. *arXiv preprint arXiv:1904.05417*, 2019.

[42] Justin Sirignano and Konstantinos Spiliopoulos. Dgm: A deep learning algorithm for solving partial differential equations. *Journal of Computational Physics*, 375:1339–1364, 2018.

[43] Maziar Raissi. Deep hidden physics models: Deep learning of nonlinear partial differential equations. *The Journal of Machine Learning Research*, 19(1):932–955, 2018.

[44] 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.

[45] Jens Berg and Kaj Nyström. A unified deep artificial neural network approach to partial differential equations in complex geometries. *Neurocomputing*, 317:28–41, 2018.

[46] Tian Qi Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural ordinary differential equations. In *Proc. NIPS*, pages 6571–6583, 2018.

[47] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. In *Proc. NIPS Workshops*, 2017.

[48] Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. In *Proc. NeurIPS*, pages 1177–1184, 2008.

[49] Patrick Pérez, Michel Gangnet, and Andrew Blake. Poisson image editing. *ACM Trans. on Graphics*, 22 (3):313–318, 2003.

[50] Ankur Handa, Thomas Whelan, John McDonald, and Andrew J Davison. A benchmark for rgb-d visual odometry, 3d reconstruction and slam. In *Proc. ICRA*, pages 1524–1531. IEEE, 2014.

[51] Rohan Chabra, Jan Eric Lenssen, Eddy Ilg, Tanner Schmidt, Julian Straub, Steven Lovegrove, and Richard Newcombe. Deep local shapes: Learning local sdf priors for detailed 3d reconstruction. *arXiv preprint arXiv:2003.10983*, 2020.

[52] Zhongying Chen, Dongsheng Cheng, Wei Feng, and Tingting Wu. An optimal 9-point finite difference scheme for the helmholtz equation with pml. *International Journal of Numerical Analysis & Modeling*, 10 (2), 2013.

[53] Hossein S Aghamiry, Ali Gholami, and Stéphane Operto. Improving full-waveform inversion by wavefield reconstruction with the alternating direction method of multipliers. *Geophysics*, 84(1):R139–R162, 2019.

[54] Tristan Van Leeuwen and Felix J Herrmann. Mitigating local minima in full-waveform inversion by expanding the search space. *Geophysical Journal International*, 195(1):661–667, 2013.

[55] Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, Jonathan Eckstein, et al. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine learning*, 3(1):1–122, 2011.

[56] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *Proc. ICCV*, December 2015.

[57] Marta Garnelo, Dan Rosenbaum, Chris J Maddison, Tiago Ramalho, David Saxton, Murray Shanahan, Yee Whye Teh, Danilo J Rezende, and SM Eslami. Conditional neural processes. *arXiv preprint arXiv:1807.01613*, 2018.

[58] SM Ali Eslami, Danilo Jimenez Rezende, Frederic Besse, Fabio Viola, Ari S Morcos, Marta Garnelo, Avraham Ruderman, Andrei A Rusu, Ivo Danihelka, Karol Gregor, et al. Neural scene representation and rendering. *Science*, 360(6394):1204–1210, 2018.

[59] Hyunjik Kim, Andriy Mnih, Jonathan Schwarz, Marta Garnelo, Ali Eslami, Dan Rosenbaum, Oriol Vinyals, and Yee Whye Teh. Attentive neural processes. *Proc. ICLR*, 2019.

[60] David Ha, Andrew Dai, and Quoc V Le. Hypernetworks. In *Proc. ICLR*, 2017.

[61] 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. In *Proc. NeurIPS*, 2020.

[62] Arthur Jacot, Franck Gabriel, and Clément Hongler. Neural tangent kernel: Convergence and generalization in neural networks. In *Proc. NeurIPS*, pages 8571–8580, 2018.