High precision PINNs in unbounded domains: application to singularity formulation in PDEs

Anonymous authors
Paper under double-blind review

Abstract

We investigate the high-precision training of Physics-Informed Neural Networks (PINNs) in unbounded domains, with a special focus on applications to singularity formulation in PDEs. We propose a modularized approach and study the choices of neural network ansatz, sampling strategy, and optimization algorithm. When combined with rigorous computer-assisted proofs and PDE analysis, the numerical solutions identified by PINNs, provided they are of high precision, can serve as a powerful tool for studying singularities in PDEs. For 1D Burgers equation, our framework can lead to a solution with very high precision, and for the 2D Boussinesq equation, which is directly related to the singularity formulation in 3D Euler and Navier-Stokes equations, we obtain a solution whose loss is 4 digits smaller than that obtained in Wang et al. (2023) with fewer training steps. We also discuss potential directions for pushing towards machine precision for higher-dimensional problems.

1 Introduction

Singularity formulation is one of the key challenges in the study of partial differential equations (PDEs). Unlike well-posed equations, where one can apply classical existence and uniqueness theorems, singularities often occur in certain solutions of nonlinear PDEs, where we only have guarantees of existence for a short time, but the solution may blow up in finite time. The study of singularities often involves a case-by-case approach and is related to some of the most intriguing mathematics and physical properties, such as the onset of turbulence in the Navier-Stokes equations. The singularity of the Navier-Stokes equations is one of the seven Millennium Prize Problems Fefferman (2006): widely regarded as the most fundamental and challenging problem in analysis, and is still open. One of the key difficulties in the study of singularities is the lack of understanding of the singularity pattern and its mechanism. The computation of the singularity itself, or infinity, is intractable numerically. A general roadmap is thus to first propose a plausible singularity ansatz that renders the computation feasible, then find candidates of such blowup by numerical simulations, and finally verify the stability of such an ansatz by PDE analysis.

We are often interested in a special structure of singularity: self-similar singularity. Self-similarity relates to the invariance of the solution under scaling transformations and reduces singularity to the existence of a self-similar profile. To be precise, for the quantity of interest $u(x,t)$, we put the ansatz $u(x,t) = (T-t)^{-\alpha}U(x(T-t)^{-\beta})$, where U is the profile function independent of time, T is the blowup time, and $\alpha > 0, \beta$ are the scaling exponents to be determined. Now we can reduce the computation of an infinite u to the computation of a finite, smooth profile U, along with scaling exponents α, β to be inferred. Physics-Informed Neural Networks (PINNs) Raissi et al. (2019) serve as a powerful tool to find such profiles, with the scaling parameters jointly inferred as inverse problems. It was first introduced in the context of identifying singularity profiles in Wang et al. (2023) and has seen success even in problems that are unstable for traditional numerical methods. While PINNs offer a powerful tool to search for candidate profiles, solutions identified by PINNs are often of limited accuracy and far from applicable to rigorous PDE analysis, to the best of the authors' knowledge.

In this work, we aim to systematically study the high-precision training of PINNs, with a special focus on applications to solve profile equations governing singularity formulations in PDEs, in fluid dynamics and beyond. We will take a modularized perspective without diving into sophisticated tricks and investigate

the following aspects: a good neural network ansatz representing the profile function, a good sampling strategy to tackle the infinite domain with a special focus on imposing boundary conditions, and a good optimization algorithm to train the neural network. We apply our findings to the 1D Burgers equation and the 2D Boussinesq equation, obtaining a precision amenable to rigorous PDE analysis for the 1D Burgers equation and 4 digits better than Wang et al. (2023) with fewer training steps for the 2D Boussinesq equation. We remark that a later work Wang et al. (2025b) achieves even higher precision while discovering unstable blowups. The 2D Boussinesq equation shares many similarities with the 3D axisymmetric Euler equation for the ideal fluid without viscosity; see the pioneering works of Chen et al. (2021); Chen & Hou (2022; 2025) for the connection between the two equations, where the authors used the connection to establish singularity formulation for the 3D axisymmetric Euler with boundary.

2 Related Works

2.1 PINNs

Neural networks have witnessed success in solving PDEs and surrogate modeling in math and science. PINNs in particular have been widely used due to their flexibility and applicability to a wide range of problems Cai et al. (2021); Raissi et al. (2019); Karniadakis et al. (2021). The key idea of PINNs is to enforce the PDE constraints at a set of collocation points and to minimize the residual of the PDEs as a loss function. By posing the solution of PDEs as an optimization problem, PINNs are especially suited to solve inverse problems Raissi et al. (2020); Yuan et al. (2022); Lu et al. (2021b); Yu et al. (2022), where the solution of the PDE and the underlying parameters can be jointly inferred. In Wang et al. (2023), the authors used PINNs to study the blowup of the 1D Burgers equation, the 1D family of generalized Constantin-Lax Majda equations, and the 2D Boussinesq equation.

Another line of work, operator learning Kovachki et al. (2023), focuses on learning the solution operator instead of learning a single instance of solution, where Fourier Neural Operators (FNOs) Li et al. (2020; 2023a;b); Li et al. and DeepONets Lu et al. (2021a) are two families of representative works in this direction. Once a solution operator is learned, it can be evaluated in a resolution-free manner at any point in the domain. Data are often augmented to enhance the solution accuracy, while the loss function can also incorporate the PDE constraints, termed Physics Informed Neural Operator (PINO) Li et al. (2021). In Maust et al. (2022), the authors used PINO with Fourier continuation to study the blowup of the 1D Burgers equation.

2.2 Self-similar Singularity and Computer Assisted Proofs

Self-similar singularity of the ansatz $u(x,t) = (T-t)^{-\alpha}U(x(T-t)^{-\beta})$ is generic in the study of singularity formation in PDEs, where one uses the scaling invariance of the PDE and can reduce the computation of an infinite u to the computation of a finite, smooth (approximate) profile U. Such structures exist even in the simple Riccati ODE $u_t = u^2$ with an exact solution $u = (T-t)^{-1}$.

The approximate profile can be identified via explicit construction or numerical computation. Working in the rescaled, self-similar variables and performing stability analysis around the profile U provides a powerful tool to establish the singularity formation, for nonlinear Schrodinger equations McLaughlin et al. (1986); Merle & Raphael (2005), incompressible fluids Elgindi & Jeong (2019); Chen & Hou (2021); Chen et al. (2021); Chen & Hou (2022); Hou & Wang (2024), compressible fluids Merle et al. (2022a;b), and beyond. Until recently, most of the works relied on an explicit profile and spectral information of the associated linearized operator to establish linear and nonlinear stability. In Chen & Hou (2022; 2025), the authors used computer-assisted proofs with a sophisticated numerical profile obtained by solving the dynamic rescaling equations in time to obtain an approximate steady state. By analyzing the stability of the approximate profile, they established the singularity formation for the 2D Boussinesq equation and the 3D axisymmetric Euler equation with boundary. And in Hou et al. (2024); Chen et al. (2024a), the authors provided a framework using only local information for stability analysis, bypassing spectral information and allowing for numerical profiles with computer-assisted proofs, for problems beyond self-similarity.

2.3 Towards High Precision Training

Various methods have been proposed in the literature to improve the accuracy of PINNs. One line of work focuses on a better representation of the solution. In Michaud et al. (2023); Wang & Lai (2024), the boosting technique was proposed, where a sum of a sequence of neural networks with decreasing magnitude was used to learn the solution; at each stage, a new neural network is trained to learn the residual. To overcome the spectral bias Rahaman et al. (2019) of multilayer perceptrons (MLPs), or the favor of learning low-frequency modes Xu et al. (2019b;a) in the solution, one can use Fourier feature encoding Sitzmann et al. (2020); Tancik et al. (2020); Ng et al. (2024), or different activation functions Jagtap et al. (2020b;a); Hong et al. (2022); Zhang et al. (2023); Wang et al. (2024a). In particular, Kolmogorov-Arnold Networks (KANs) Liu et al. (2024b;a) that leverage nonlinear learnable activation functions and the Kolmogorov-Arnold representation theorem were proposed and further investigated in the PINN setting Wang et al. (2024b); Shukla et al. (2024); Toscano et al. (2025). Another line of work improves the optimization landscape during the training of PINNs. Various optimizers, which we will detail in Subsection 3.3, have been proposed to improve the convergence rate. Adaptive design of points sampling Anagnostopoulos et al. (2024); Wu et al. (2023); Rigas et al. (2024) and adaptive weighting of different terms Wang et al. (2021); Xiang et al. (2022); McClenny & Braga-Neto (2023) in the loss function were also proposed to improve the accuracy of PINNs.

We will only focus on applying hard constraints and choosing a good optimizer in this work and leave the exploration of more sophisticated tricks for future work.

3 Methodology

We outline our methodology of high-precision training for PINNs on the whole space in this section. We work under the general formulation of the profile equation

$$L(U, \lambda) = 0,$$

where $U(y)$ is the profile function, λ is a set of scaling parameters to be determined, and L is the nonlinear differential operator. For our problems of interest, U=0 will be a trivial solution satisfying the equation.

3.1 Infinite Domain

The key challenges we are facing here are sampling and learning on an infinite domain. For a given budget of a finite number of sampling points, we need to sample the domain in a way that the resulting solution is accurate and generalizes well throughout the domain. In the meantime, We want the neural network to be able to represent the profile function and the initialization of parameters to favor learning of such representations. To this end, we adopt an exponential "mesh" in our sampling strategy: Consider an auxiliary variable z such that $y = \sinh(z) = \frac{e^z - e^{-z}}{2}$, and sample z uniformly in a finite region. Here we choose the sinh transformation as in Wang et al. (2023) to respect the parity of the functions, detailed in the subsequent subsection. Such a transformation maps roughly $z \in [-30, 30]$ to $y \in [-5 \times 10^{12}, 5 \times 10^{12}]$.

Boundary conditions are another important aspect when learning on the whole space. For our problems of interest, U by itself will not have sufficient decay at infinity, and one approach adopted in Wang et al. (2023) is to impose Neumann boundary conditions at infinity, or numerically on the boundary of the domain of the z variables. To rule out the trivial solution $U = 0$, we need to enforce a nondegeneracy condition, often posed at the origin. We will discuss the enforcement of these conditions in the following subsection. We refer to this formulation as boundary conditions using **weak asymptotics**.

On the other hand, we can enforce stronger information on the boundary. If we know the exact asymptotic behavior of the solution at infinity as g, for example a power law, we can introduce a smooth cutoff function χ with $\chi(0) = 0, \chi(\infty) = 1$ and the ansatz $U = \tilde{U} + \chi g$. We can then enforce Dirchlet boundary conditions at infinity for $\tilde{U}$ represented by the neural network. We refer to this formulation as boundary conditions using **exact asymptotics**. We will demonstrate for the 1D example that PINNs using exact asymptotics will outperform those using weak asymptotics by a large margin.

A priori, the exact asymptotics information is not available, and one can first train a neural network U_w with boundary conditions using weak information, and distill the information of asymptotics g from U_w. We refer to this formulation as boundary conditions using **hybrid asymptotics**. For example, for the 2D Boussinesq equation, borrowing ideas from Chen & Hou (2022), one can use function fitting and symbolic regression to extract asymptotics g from U_w, filtering out the noisy residues, such that g is a symbolic function approximating U_w at infinity. We will leave this approach to future work.

3.2 Hard Constraint

Hard constraints are important concepts in the parametrization of the solution space for PINNs. When enforced properly, they will guarantee physical properties of the solution Lu et al. (2021b); Richter-Powell et al. (2022); Mohan et al. (2023); Duruisseaux et al. (2024), and can impose the solution to be in the correct manifold. While most of the previous works focus on hard constraints of boundary conditions, we emphasize the enforcement of hard constraints in the following senses: the parity of the learned function and the nondegeneracy conditions. Empirically we observe a better convergence rate and a more stable solution when enforcing hard constraints.

Parity. For a function $f(y_i, \hat{y}_i)$ even/odd in the variable y_i, we train a neural network with the following ansatz $f = (f_{nn}(y_i, \hat{y}_i) \pm f_{nn}(-y_i, \hat{y}_i))/2$.

Nondegeneracy conditions. As discussed in the previous subsection, we need to enforce nondegeneracy conditions to rule out the trivial solution $U = 0$ when using weak asymptotics. For example, for the 1D Burgers equation, we know that U is odd and necessarily $U'(0) = -1$; we can enforce $U'''(0) = 6$. We will enforce a hard constraint via Taylor expansion at the origin as $U = -z + z^3 + z^4 U_1$, for an odd function U_1. Similarly for the 2D Boussinesq equation, we enforce $\partial_1 \Omega(0, 0) = -1$ and Ω is odd in z_1 via a Taylor expansion as $\Omega = -z_1 + z_1 z_2 \Omega_1 + z_1^2 \Omega_2$, where Ω_1, Ω_2 are even and odd functions in z_1 respectively.

3.3 Optimizer: Self-Scaled BFGS Methods

A common practice of training PINNs is to use the Adam optimizer. As a stochastic first-order method, Adam is known to be robust and efficient in training deep neural networks and can empirically escape local minima. To further improve convergence to the minimizer, one can apply second-order methods with a higher convergence rate, like L-BFGS, after training with Adam for a few epochs. While this seems to be a gold standard in the training of PINNs Rathore et al. (2024), various optimizers have been investigated, including variants of second-order quasi-Newton methods Rathore et al. (2024); Wang et al. (2025a), and optimizers using natural gradients Müller & Zeinhofer (2023); Jnini et al. (2024); Chen et al. (2024b). We highlight and use the self-scaled BFGS methods proposed in Al-Baali (1998); Al-Baali et al. (2014) and introduced to the PINNs context in Urbán et al. (2025); Kiyani et al. (2025). BFGS methods use an approximation of the inverse of the Hessian matrix to precondition the gradient for the update direction. To be precise, consider the parameters Θ_k and learning rate α_k at step k, with loss function $\mathcal{J}(\Theta)$, then the update rule for Θ is

$$\Theta_{k+1} = \Theta_k - \alpha_k H_k \nabla \mathcal{J}(\Theta_k).$$

Different choices of updating the approximate inverse Hessian H_k lead to different optimizers, and L-BFGS in particular is a memory-efficient way for the updates by storing only vectors instead of the whole matrix. The self-scaled BFGS methods use a scaling compared to the standard BFGS update of the inverse Hessian. More precisely, for the auxiliary variables

$$s_k = \Theta_{k+1} - \Theta_k, \quad y_k = \mathcal{J}(\Theta_{k+1}) - \mathcal{J}(\Theta_k),$$
$$v_k = \sqrt{y_k \cdot H_k y_k} \left[\frac{s_k}{y_k \cdot s_k} - \frac{H_k y_k}{y_k \cdot H_k y_k} \right],$$

we have for the scalers τ_k and ϕ_k:

$$H_{k+1} = \frac{1}{\tau_k} \left[H_k - \frac{H_k y_k \otimes H_k y_k}{y_k \cdot H_k y_k} + \phi_k v_k \otimes v_k \right] + \frac{s_k \otimes s_k}{y_k \cdot s_k},$$

where the original BFGS corresponds to the choices $\tau_k = \phi_k = 1$. While this is only a simple modification of the original BFGS, the authors in Urbán et al. (2025) demonstrated a much improved convergence rate across a variety of benchmarks, including the Helmholtz equation, the nonlinear Poisson equation, the nonlinear Schrödinger equation, the Korteweg-De Vries equation, the viscous Burgers equation, the Allen-Cahn equation, 3D Navier-Stokes: Beltrami flow, and the lid-driven cavity. We use the self-scaled Broyden methods proposed in Urbán et al. (2025); see equations (13)-(23) therein for details on the choices of τ_k and ϕ_k.

On the role of minibatch training or random resampling. One of the common practices when training PINNs is to use random resampling of the collocation points. This can enhance the performance of SGD-based methods like Adam empirically. However, full-batch second-order methods like BFGS with supposedly higher-order accuracy do not adapt well to random resampling since they rely on past trajectories for Hessian updates. One empirical observation, as proposed in Wang & Lai (2024), is that when one uses an optimizer with fixed resolution like BFGS, it will be able to generalize in the regions where sampling points are sufficient. However, in the undersampled regions, the learned solution generalizes poorly. In an abstract form, there exists a critical batchsize N_c, such that when $N > N_c$, fixed sampling will be preferred, while for $N < N_c$, fixed sampling will have very bad generalization. N_c would depend on both the equation and the scale of the neural network. Empirically, we observe that roughly 10k points are sufficient for generalization with fixed training points. We resample every 1000 epochs to further reduce overfitting. See details on the choices of the batch size in the experiments section.

4 Experiments

In this section, we describe our numerical experiments on the blowup profiles for 1D Burgers equation and 2D Boussinesq Equation. When training both equations, we denote the PDE by $L(U(y)) = 0$ and the boundary condition by $B(U) = 0$. We use auxiliary variables $z = \sinh^{-1} y$ as in Subsection 3.1 and consider the following combination of interior, boundary, and smoothness losses as in Wang et al. (2023)

$$\begin{aligned} loss &= 0.1(L_i + L_s) + L_b \\ &= 0.1\Big(\frac{1}{N_i}\sum_{j=1}^{N_i}[\hat{L}(U_{nn}(z_j))]^2 + \frac{1}{N_s}\sum_{j=1}^{N_s}|\nabla_{z_j}\hat{L}(U_{nn}(z_j))|^2\Big) + \frac{1}{N_b}\sum_{j=1}^{N_b}[\hat{B}(U_{nn})]^2, \end{aligned} \tag{1}$$

where $U_{nn}(z)$ is supposed to approximate $\hat{U}(z) = U(y)$ in the z-variables, and $\hat{L}$, $\hat{B}$ denotes the PDE and the boundary condition transformed in the z-variables; see Wang et al. (2023) for a concrete formula for the 2D Boussinesq equation.

4.1 Burgers Equation

For the 1D Burgers equation

$$u_t + uu_x = 0, \tag{2}$$

consider the self-similar ansatz that respects the scaling symmetry

$$u(x,t) = (1-t)^{\lambda}U(y), \quad y = x(1-t)^{-1-\lambda}. \tag{3}$$

The profile equation for U used for the PDE loss in (1) is

$$-\lambda U + ((1+\lambda)y + U)U_y = 0. \tag{4}$$

We impose an odd symmetry on U, and the profile equation has implicit solutions

$$y + U + CU^{1+1/\lambda} = 0. \tag{5}$$

for any constant C, as in the setting of Wang et al. (2023), where we know that the most stable solutions correspond to $\lambda = 0.5$ and there are nonsmooth solutions at e.g. $\lambda = 0.4$.

In this example, we assume that we first train the neural network on a bounded domain and infer the correct λ already, for example via the method in Wang et al. (2023). Now we focus on fixing λ and learn U on the unbounded domain. Using an MLP with activation function tanh, 4 layers and 20 neurons per layer and a hard constraint on parity, we use the optimizer SSBroyden1 as in Urbán et al. (2025) with 20000 epochs and resampling every 1000 epochs. z is sampled uniformly on $[0, 30]$ with a batchsize 10000 for both the interior and smoothness losses, corresponding to a domain $[0, 5 \times 10^{12}]$ in the y variables.

For the formulation using weak asymptotics as in Subsection 3.1, we use the Neumann boundary condition $U_y = 0$ and enforce hard constraint of nondegeneracy conditions as in Subsection 3.2. For the formulation using exact asymptotics as in Subsection 3.1, we use Dirichlet boundary condition $\tilde{U} = 0$ and the cutoff function $\chi = (\frac{y}{1+y})^{15}$, since the far field is captured by the exact asymptotics $g = -y^{\frac{\lambda}{1+\lambda}}$.

We present the following results of $\lambda = 0.4, 0.5$ using weak and exact asymptotics: see Figure 1 for the equation residue of the solution at the final stage and Figure 2 for the evolution of the losses. We are able to achieve high accuracy over a large domain, but using exact asymptotics is preferred for both the smooth and nonsmooth case of λ.

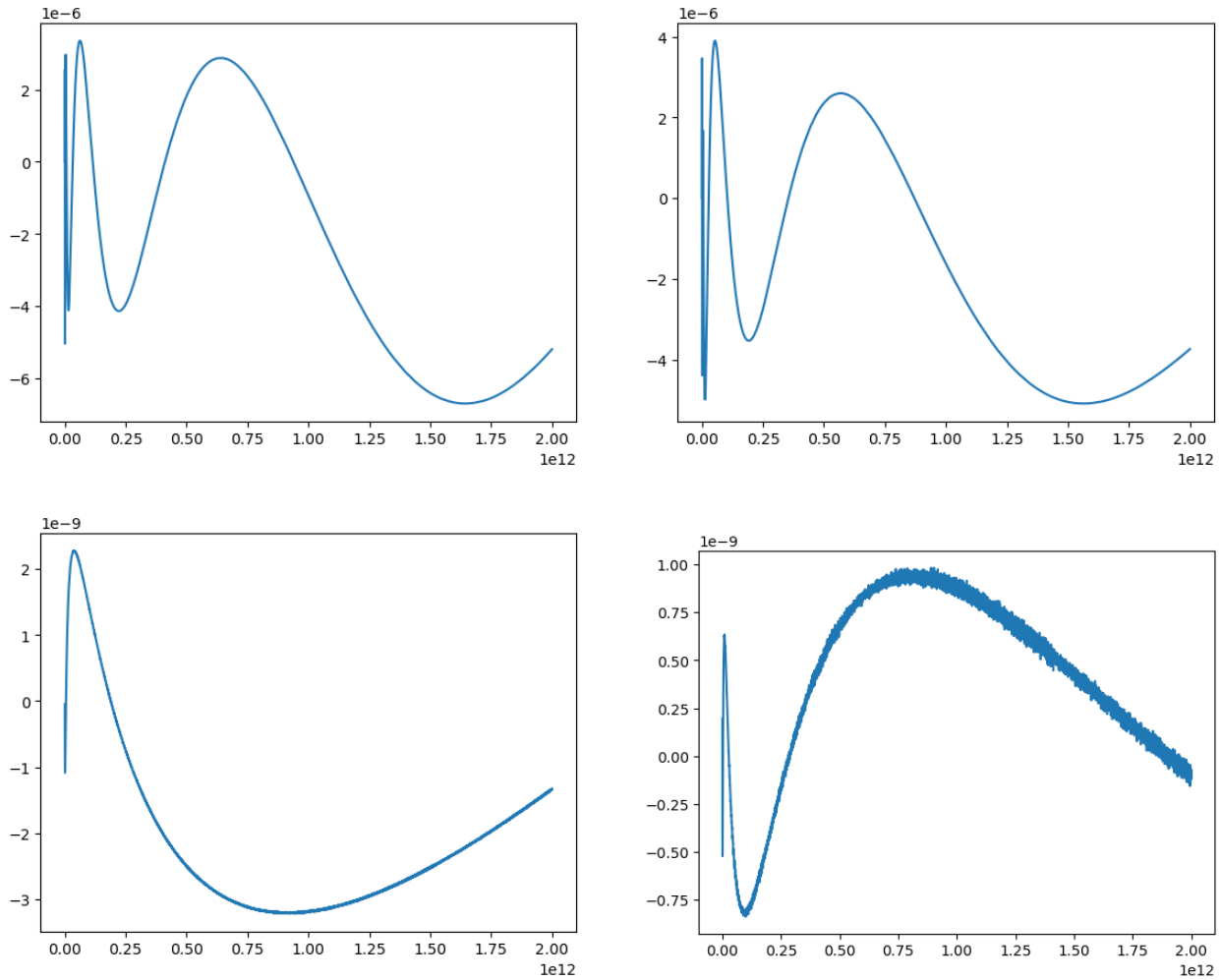

Figure 1: Final residue in a large domain for 1D Burgers. Upper: weak asymptotics; down: exact asymptotics; left: $\lambda = 0.4$; right: $\lambda = 0.5$.

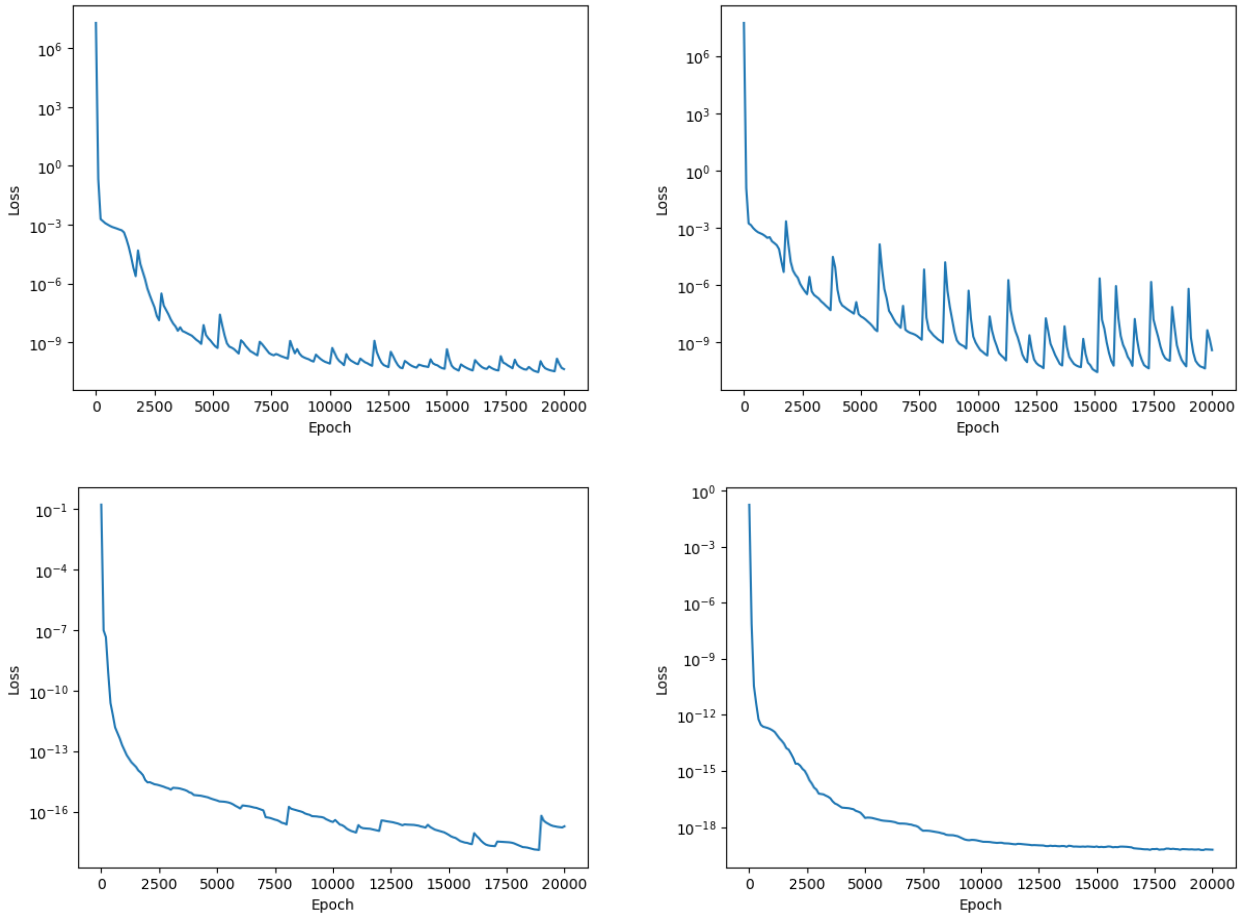

Figure 2: Trajectory of losses for 1D Burgers. Upper: weak asymptotics; down: exact asymptotics; left: $\lambda = 0.4$; right: $\lambda = 0.5$.

4.2 Boussinesq Equation

For the 2D Boussinesq equation on the half plane, in vorticity form with the self-similar ansatz, we get the following profile equations for $(\Omega, U_1, U_2, \Phi, \Psi)$ as in Wang et al. (2023):

$$\begin{gathered}
\Omega + ((1+\lambda)(y_1, y_2)^T + (U_1, U_2)^T) \cdot \nabla\Omega = \Phi, \\
(2 + \partial_{y_1} U_1)\Phi + ((1+\lambda)(y_1, y_2)^T + (U_1, U_2)^T) \cdot \nabla\Phi = -\partial_{y_1} U_2 \Psi, \\
(2 + \partial_{y_2} U_2)\Psi + ((1+\lambda)(y_1, y_2)^T + (U_1, U_2)^T) \cdot \nabla\Psi = -\partial_{y_2} U_1 \Phi, \\
\partial_{y_1} U_1 + \partial_{y_2} U_2 = 0, \quad \Omega = \partial_{y_1} U_2 - \partial_{y_2} U_1, \quad \partial_{y_1}\Psi = \partial_{y_2}\Phi,
\end{gathered}$$

where (Ω, U_1, Φ) are odd and (U_2, Ψ) are even in y_1 and we are in the half plane $y_2 \geq 0$.

For the boundary conditions, we impose a non-penetration boundary condition $U_2(y_1, 0) = 0$ along with decaying weak asymptotics at the far field, with Dirichlet boundary conditions $\Phi = \Psi = 0$ and Neumann boundary conditions for the velocity field $\nabla(U_1, U_2)^T = 0$. For the nondegeneracy condition, we impose $\partial_{y_1}\Omega(0,0) = -1$ and use Taylor expansion to enforce a hard constraint as in Subsection 3.2. We find that enforcing a hard constraint is much more effective to avoid converging to a trivial solution than enforcing soft constraints.

For each function, we use a 7-layer MLP with width 30, hard constraints on parity, and activation function $\text{SiLU} = \frac{x}{1+e^{-x}}$ to better model the growth at the far field. For sampling, we sample 1000 points on each boundary of the square $(z_1, z_2) \in [0, 30]^2$, and 5000 points each for the interior and smoothness losses, where we sample (z_1, z_2) with equal probability uniformly on $[0, 30]^2$ and $[0, 5]^2$ for the interior loss and with equal probability uniformly on $[0, 3]^2$ and $[0, 0.5]^2$ for the smoothness loss, ensuring smoothness near the origin. Again, we are computing effectively in a large domain $[0, 5 \times 10^{12}]^2$ in the y variables.

For optimization, we use Adam for 10000 epochs with resampling, followed by the optimizer SSBroyden 1 as in Urbán et al. (2025) with 40000 epochs and resampling every 1000 epochs. The learning rate of Adam is set to be 0.001 for the functions and 0.1 with $\beta = (0.9, 0.9)$ for λ, with a decay of 0.9 after 5000 epochs.

We present the final profiles and the residue of the PDEs near the origin respectively in Figure 3 and 4, and the evolution of losses in Figure 5. Compared to Wang et al. (2023), we achieve a training loss of 4 digits smaller and equation residues of 2 digits smaller. We remark that due to computational constraints and the cost of a full batch optimizer involving the approximation of the Hessian matrix, this is the largest neural network affordable. We use 10 days of CPU time on a MacPro 2019 with 2.5GHz 28-core Intel Xeon W processor.

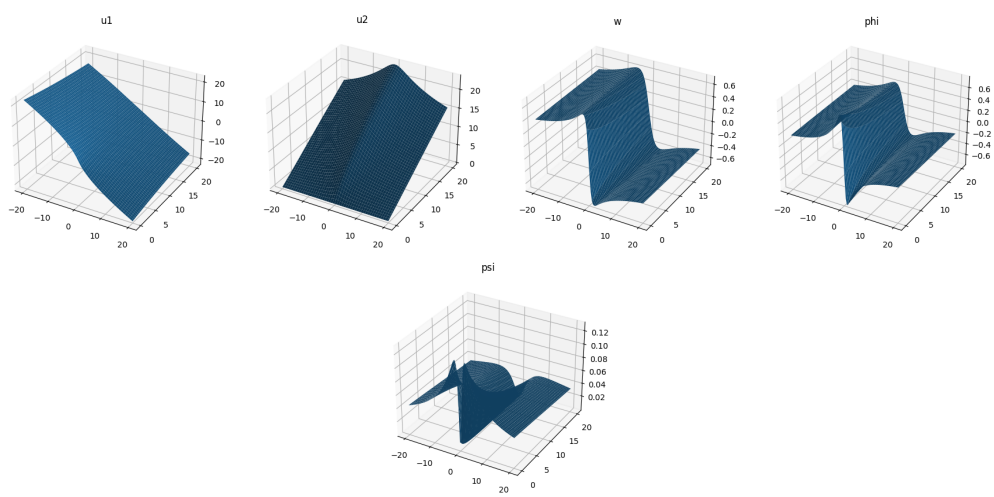

Figure 3: Final profiles for 2D Boussinesq

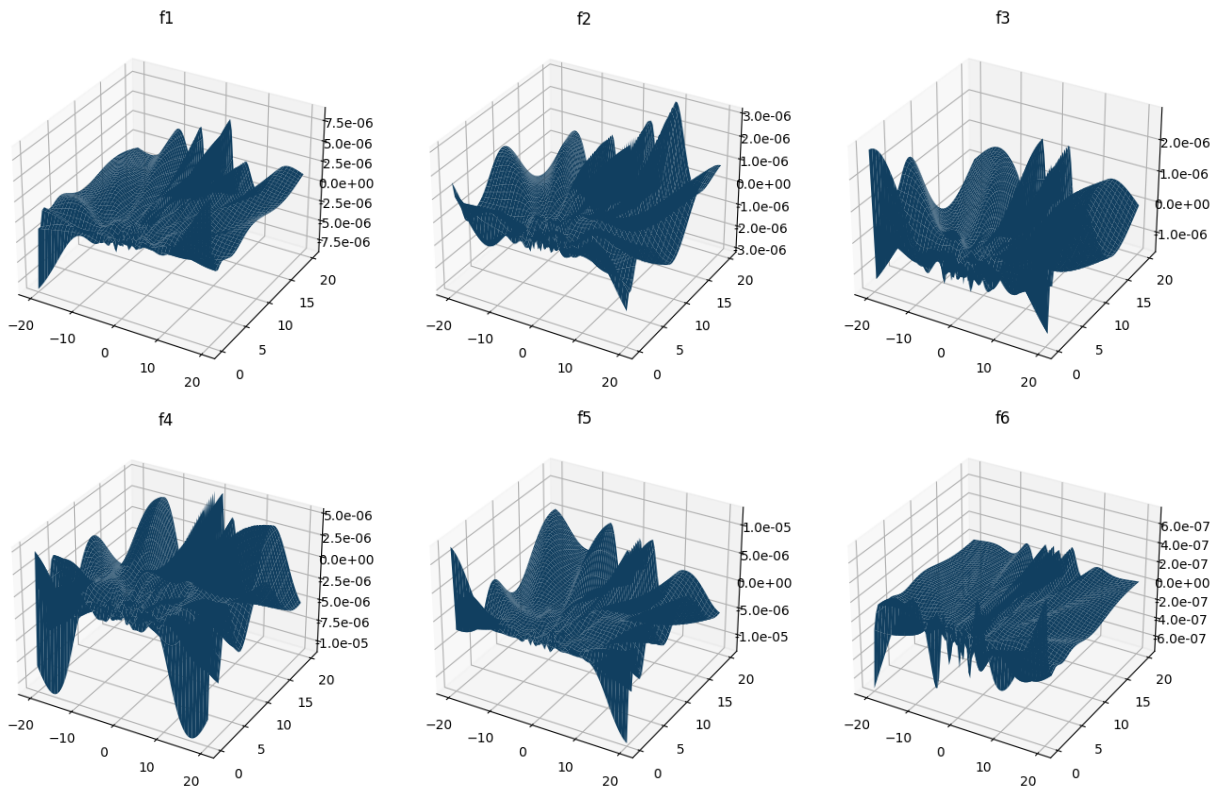

Figure 4: Final equation residues for 2D Boussinesq

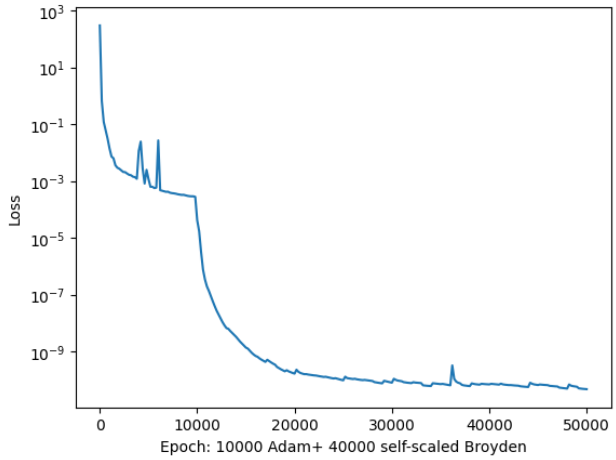

Figure 5: Trajectory of losses for 2D Boussinesq: 10000 Adam iterations followed by 40000 self-scaled Broyden iterations.

5 Conclusions and Future Work

We demonstrate the importance of enforcing appropriate asymptotics, enforcing hard constraints, and adopting a better optimizer for solving PDEs using neural networks on an infinite domain. We achieve better accuracy for problems crucial to the study of singularity formulations. As a future direction, we believe a better enforcement of far-field asymptotics, formulated as hybrid asymptotics, might have the potential of driving PDE residues to machine precision, potentially amenable to rigorous computer-assisted proofs using the profiles identified by the neural networks. Another direction is to use PINO, the idea of operator learning on a range of scaling parameters, to learn a collection of profiles with different scalings.

References

M Al-Baali. Numerical experience with a class of self-scaling quasi-newton algorithms. *Journal of optimization theory and applications*, 96:533–553, 1998.

Mehiddin Al-Baali, Emilio Spedicato, and Francesca Maggioni. Broyden's quasi-newton methods for a nonlinear system of equations and unconstrained optimization: a review and open problems. *Optimization Methods and Software*, 29(5):937–954, 2014.

Sokratis J Anagnostopoulos, Juan Diego Toscano, Nikolaos Stergiopulos, and George Em Karniadakis. Residual-based attention in physics-informed neural networks. *Computer Methods in Applied Mechanics and Engineering*, 421:116805, 2024.

Shengze Cai, Zhiping Mao, Zhicheng Wang, Minglang Yin, and George Em Karniadakis. Physics-informed neural networks (pinns) for fluid mechanics: A review. *Acta Mechanica Sinica*, 37(12):1727–1738, 2021.

Jiajie Chen and Thomas Y Hou. Finite time blowup of 2D Boussinesq and 3D Euler equations with $C^{1,\alpha}$ velocity and boundary. *Communications in Mathematical Physics*, 383(3):1559–1667, 2021.

Jiajie Chen and Thomas Y Hou. Stable nearly self-similar blowup of the 2D Boussinesq and 3D Euler equations with smooth data. *arXiv preprint arXiv:2210.07191*, 2022.

Jiajie Chen and Thomas Y Hou. Stable nearly self-similar blowup of the 2d boussinesq and 3d euler equations with smooth data ii: rigorous numerics. *Multiscale Modeling & Simulation*, 23(1):25–130, 2025.

Jiajie Chen, Thomas Y Hou, and De Huang. On the finite time blowup of the De Gregorio model for the 3D Euler equations. *Communications on pure and applied mathematics*, 74(6):1282–1350, 2021.

Jiajie Chen, Thomas Y Hou, Van Tien Nguyen, and Yixuan Wang. On the stability of blowup solutions to the complex ginzburg-landau equation in $\mathbb{R}^d$. *arXiv preprint arXiv:2407.15812*, 2024a.

Zhuo Chen, Jacob McCarran, Esteban Vizcaino, Marin Soljačić, and Di Luo. Teng: Time-evolving natural gradient for solving pdes with deep neural nets toward machine precision. *arXiv preprint arXiv:2404.10771*, 2024b.

Valentin Duruisseaux, Miguel Liu-Schiaffini, Julius Berner, and Anima Anandkumar. Towards enforcing hard physics constraints in operator learning frameworks. In *ICML 2024 AI for Science Workshop*, 2024.

Tarek M Elgindi and In-Jee Jeong. Finite-time singularity formation for strong solutions to the axi-symmetric 3D Euler equations. *Annals of PDE*, 5(2):16, 2019.

Charles L Fefferman. Existence and smoothness of the navier-stokes equation. *The millennium prize problems*, 57(67):22, 2006.

Qingguo Hong, Jonathan W Siegel, Qinyang Tan, and Jinchao Xu. On the activation function dependence of the spectral bias of neural networks. *arXiv preprint arXiv:2208.04924*, 2022.

Thomas Y Hou and Yixuan Wang. Blowup analysis for a quasi-exact 1D model of 3D Euler and Navier-Stokes. *Nonlinearity*, 37(3):035001, 2024.

Thomas Y Hou, Van Tien Nguyen, and Yixuan Wang. L^2-based stability of blowup with log correction for semilinear heat equation. *arXiv preprint arXiv:2404.09410*, 2024.

Ameya D Jagtap, Kenji Kawaguchi, and George Em Karniadakis. Locally adaptive activation functions with slope recovery for deep and physics-informed neural networks. *Proceedings of the Royal Society A*, 476 (2239):20200334, 2020a.

Ameya D Jagtap, Kenji Kawaguchi, and George Em Karniadakis. Adaptive activation functions accelerate convergence in deep and physics-informed neural networks. *Journal of Computational Physics*, 404:109136, 2020b.

Anas Jnini, Flavio Vella, and Marius Zeinhofer. Gauss-newton natural gradient descent for physics-informed computational fluid dynamics. *arXiv preprint arXiv:2402.10680*, 2024.

George Em Karniadakis, Ioannis G Kevrekidis, Lu Lu, Paris Perdikaris, Sifan Wang, and Liu Yang. Physics-informed machine learning. *Nature Reviews Physics*, 3(6):422–440, 2021.

Elham Kiyani, Khemraj Shukla, Jorge F Urbán, Jérôme Darbon, and George Em Karniadakis. Which optimizer works best for physics-informed neural networks and kolmogorov-arnold networks? *arXiv preprint arXiv:2501.16371*, 2025.

Nikola Kovachki, Zongyi Li, Burigede Liu, Kamyar Azizzadenesheli, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Neural operator: Learning maps between function spaces with applications to pdes. *Journal of Machine Learning Research*, 24(89):1–97, 2023.

Zongyi Li, Samuel Lanthaler, Catherine Deng, Yixuan Wang, Kamyar Azizzadenesheli, and Anima Anandkumar. Scale-consistent learning with neural operators. In *Neurips 2024 Workshop Foundation Models for Science: Progress, Opportunities, and Challenges*.

Zongyi Li, Nikola Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Fourier neural operator for parametric partial differential equations. *arXiv preprint arXiv:2010.08895*, 2020.

Zongyi Li, Hongkai Zheng, Nikola Kovachki, David Jin, Haoxuan Chen, Burigede Liu, Kamyar Azizzadenesheli, and Anima Anandkumar. Physics-informed neural operator for learning partial differential equations. *ACM/JMS Journal of Data Science*, 2021.

Zongyi Li, Daniel Zhengyu Huang, Burigede Liu, and Anima Anandkumar. Fourier neural operator with learned deformations for pdes on general geometries. *Journal of Machine Learning Research*, 24(388):1–26, 2023a.

Zongyi Li, Nikola Kovachki, Chris Choy, Boyi Li, Jean Kossaifi, Shourya Otta, Mohammad Amin Nabian, Maximilian Stadler, Christian Hundt, Kamyar Azizzadenesheli, et al. Geometry-informed neural operator for large-scale 3d pdes. *Advances in Neural Information Processing Systems*, 36:35836–35854, 2023b.

Ziming Liu, Pingchuan Ma, Yixuan Wang, Wojciech Matusik, and Max Tegmark. Kan 2.0: Kolmogorov-arnold networks meet science. *arXiv preprint arXiv:2408.10205*, 2024a.

Ziming Liu, Yixuan Wang, Sachin Vaidya, Fabian Ruehle, James Halverson, Marin Soljačić, Thomas Y Hou, and Max Tegmark. Kan: Kolmogorov-arnold networks. *arXiv preprint arXiv:2404.19756*, 2024b.

Lu Lu, Pengzhan Jin, Guofei Pang, Zhongqiang Zhang, and George Em Karniadakis. Learning nonlinear operators via deeponet based on the universal approximation theorem of operators. *Nature machine intelligence*, 3(3):218–229, 2021a.

Lu Lu, Raphael Pestourie, Wenjie Yao, Zhicheng Wang, Francesc Verdugo, and Steven G Johnson. Physics-informed neural networks with hard constraints for inverse design. *SIAM Journal on Scientific Computing*, 43(6):B1105–B1132, 2021b.

Haydn Maust, Zongyi Li, Yixuan Wang, Daniel Leibovici, Oscar Bruno, Thomas Hou, and Anima Anandkumar. Fourier continuation for exact derivative computation in physics-informed neural operators. *arXiv preprint arXiv:2211.15960*, 2022.

Levi D McClenny and Ulisses M Braga-Neto. Self-adaptive physics-informed neural networks. *Journal of Computational Physics*, 474:111722, 2023.

DW McLaughlin, GC Papanicolaou, C Sulem, and PL Sulem. Focusing singularity of the cubic Schrödinger equation. *Physical Review A*, 34(2):1200, 1986.

F. Merle, P. Raphaël, I. Rodnianski, and J. Szeftel. On the implosion of a compressible fluid II: Singularity formation. *Annals of Mathematics*, 196(2):779 – 889, 2022a. doi: 10.4007/annals.2022.196.2.4. URL https://doi.org/10.4007/annals.2022.196.2.4.

F. Merle, P. Raphaël, I. Rodnianski, and J. Szeftel. On the implosion of a compressible fluid I: Smooth self-similar inviscid profiles. *Annals of Mathematics*, 196(2):567 – 778, 2022b. doi: 10.4007/annals.2022.196.2.3. URL https://doi.org/10.4007/annals.2022.196.2.3.

Frank Merle and Pierre Raphael. The blow-up dynamic and upper bound on the blow-up rate for critical nonlinear Schrödinger equation. *Annals of mathematics*, pp. 157–222, 2005.

Eric J Michaud, Ziming Liu, Uzay Girit, and Max Tegmark. The quantization model of neural scaling. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=3tbTw2ga8K.

Arvind T Mohan, Nicholas Lubbers, Misha Chertkov, and Daniel Livescu. Embedding hard physical constraints in neural network coarse-graining of three-dimensional turbulence. *Physical Review Fluids*, 8(1):014604, 2023.

Johannes Müller and Marius Zeinhofer. Achieving high accuracy with pinns via energy natural gradient descent. In *International Conference on Machine Learning*, pp. 25471–25485. PMLR, 2023.

Jakin Ng, Yongji Wang, and Ching-Yao Lai. Spectrum-informed multistage neural networks: Multiscale function approximators of machine precision. *arXiv preprint arXiv:2407.17213*, 2024.

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 *International conference on machine learning*, pp. 5301–5310. PMLR, 2019.

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.

Maziar Raissi, Alireza Yazdani, and George Em Karniadakis. Hidden fluid mechanics: Learning velocity and pressure fields from flow visualizations. *Science*, 367(6481):1026–1030, 2020.

Pratik Rathore, Weimu Lei, Zachary Frangella, Lu Lu, and Madeleine Udell. Challenges in training pinns: A loss landscape perspective. *arXiv preprint arXiv:2402.01868*, 2024.

Jack Richter-Powell, Yaron Lipman, and Ricky TQ Chen. Neural conservation laws: A divergence-free perspective. *Advances in Neural Information Processing Systems*, 35:38075–38088, 2022.

Spyros Rigas, Michalis Papachristou, Theofilos Papadopoulos, Fotios Anagnostopoulos, and Georgios Alexandridis. Adaptive training of grid-dependent physics-informed kolmogorov-arnold networks. *arXiv preprint arXiv:2407.17611*, 2024.

Khemraj Shukla, Juan Diego Toscano, Zhicheng Wang, Zongren Zou, and George Em Karniadakis. A comprehensive and fair comparison between mlp and kan representations for differential equations and operator networks. *arXiv preprint arXiv:2406.02917*, 2024.

Vincent Sitzmann, Julien Martel, Alexander Bergman, David Lindell, and Gordon Wetzstein. Implicit neural representations with periodic activation functions. *Advances in neural information processing systems*, 33: 7462–7473, 2020.

Matthew Tancik, Pratul Srinivasan, Ben Mildenhall, Sara Fridovich-Keil, Nithin Raghavan, Utkarsh Singhal, Ravi Ramamoorthi, Jonathan Barron, and Ren Ng. Fourier features let networks learn high frequency functions in low dimensional domains. *Advances in neural information processing systems*, 33:7537–7547, 2020.

Juan Diego Toscano, Vivek Oommen, Alan John Varghese, Zongren Zou, Nazanin Ahmadi Daryakenari, Chenxi Wu, and George Em Karniadakis. From pinns to pikans: Recent advances in physics-informed machine learning. *Machine Learning for Computational Science and Engineering*, 1(1):1–43, 2025.

Jorge F Urbán, Petros Stefanou, and José A Pons. Unveiling the optimization process of physics informed neural networks: How accurate and competitive can pinns be? *Journal of Computational Physics*, 523: 113656, 2025.

Sifan Wang, Yujun Teng, and Paris Perdikaris. Understanding and mitigating gradient flow pathologies in physics-informed neural networks. *SIAM Journal on Scientific Computing*, 43(5):A3055–A3081, 2021.

Sifan Wang, Ananyae Kumar Bhartari, Bowen Li, and Paris Perdikaris. Gradient alignment in physics-informed neural networks: A second-order optimization perspective. *arXiv preprint arXiv:2502.00604*, 2025a.

Yixuan Wang, Jonathan W Siegel, Ziming Liu, and Thomas Y Hou. On the expressiveness and spectral bias of kans. *arXiv preprint arXiv:2410.01803*, 2024a.

Yizheng Wang, Jia Sun, Jinshuai Bai, Cosmin Anitescu, Mohammad Sadegh Eshaghi, Xiaoying Zhuang, Timon Rabczuk, and Yinghua Liu. Kolmogorov arnold informed neural network: A physics-informed deep learning framework for solving pdes based on kolmogorov arnold networks. *arXiv preprint arXiv:2406.11045*, 2024b.

Yongji Wang and Ching-Yao Lai. Multi-stage neural networks: Function approximator of machine precision. *Journal of Computational Physics*, 504:112865, 2024.

Yongji Wang, C-Y Lai, Javier Gómez-Serrano, and Tristan Buckmaster. Asymptotic self-similar blow-up profile for three-dimensional axisymmetric euler equations using neural networks. *Physical Review Letters*, 130(24):244002, 2023.

Yongji Wang, Mehdi Bennani, James Martens, Sébastien Racanière, Sam Blackwell, Alex Matthews, Stanislav Nikolov, Gonzalo Cao-Labora, Daniel S Park, Martin Arjovsky, et al. Discovery of unstable singularities. *arXiv preprint arXiv:2509.14185*, 2025b.

Chenxi Wu, Min Zhu, Qinyang Tan, Yadhu Kartha, and Lu Lu. A comprehensive study of non-adaptive and residual-based adaptive sampling for physics-informed neural networks. *Computer Methods in Applied Mechanics and Engineering*, 403:115671, 2023.

Zixue Xiang, Wei Peng, Xu Liu, and Wen Yao. Self-adaptive loss balanced physics-informed neural networks. *Neurocomputing*, 496:11–34, 2022.

Zhi-Qin John Xu, Yaoyu Zhang, Tao Luo, Yanyang Xiao, and Zheng Ma. Frequency principle: Fourier analysis sheds light on deep neural networks. *arXiv preprint arXiv:1901.06523*, 2019a.

Zhi-Qin John Xu, Yaoyu Zhang, and Yanyang Xiao. Training behavior of deep neural network in frequency domain. In *Neural Information Processing: 26th International Conference, ICONIP 2019, Sydney, NSW, Australia, December 12–15, 2019, Proceedings, Part I 26*, pp. 264–274. Springer, 2019b.

Jeremy Yu, Lu Lu, Xuhui Meng, and George Em Karniadakis. Gradient-enhanced physics-informed neural networks for forward and inverse pde problems. *Computer Methods in Applied Mechanics and Engineering*, 393:114823, 2022.

Lei Yuan, Yi-Qing Ni, Xiang-Yun Deng, and Shuo Hao. A-pinn: Auxiliary physics informed neural networks for forward and inverse problems of nonlinear integro-differential equations. *Journal of Computational Physics*, 462:111260, 2022.

Shijun Zhang, Hongkai Zhao, Yimin Zhong, and Haomin Zhou. Why shallow networks struggle with approximating and learning high frequency: A numerical study. *arXiv preprint arXiv:2306.17301*, 2023.