Scalable and Expressive Physics-Informed Neural Networks via Functional Tensor Decomposition

Sai Karthikeya Vemuri[1,2,*] Tim Büchner[1] Julia Niebling[2] Joachim Denzler[1]

[1]Computer Vision Group, Friedrich Schiller University, Ernst Abbe Platz 2, 07743 Jena, Germany
[2]Institute of Data Sciences, German Aerospace Center, Mälzerstraße 3-5, 07745 Jena, Germany
*Corresponding author: sai.karthikeya.vemuri@uni-jena.de

Preprint. This manuscript is currently under consideration at *Pattern Recognition Letters*. It extends our earlier conference paper (Vemuri et al., 2024b).

Abstract

Physics-Informed Neural Networks (PINNs) offer a promising framework for solving partial differential equations (PDEs) by embedding physical laws into neural network training. In our prior work, we introduced Functional Tensor Decomposition PINNs (FTD-PINNs), which leverage tensor decomposition to improve the scalability and efficiency of PINNs, especially for high-dimensional PDEs. This work extends our previous study on Functional Tensor Decomposition PINNs by systematically examining how decomposition structure and backend activations affect performance, efficiency, and stability across representative partial differential equations. Using Helmholtz and Klein-Gordon systems as benchmarks, we compare three decomposition modes (CP, Tucker, Tensor-Train) combined with four activation backends (Tanh, Tanh + PE, SIREN, and WIRE). Multi-seed and compute-aware evaluations reveal consistent accuracy-efficiency trade-offs: Tensor-Train provides the most balanced decomposition, while frequency-aware backends such as WIRE and SIREN improve convergence for oscillatory regimes. Pareto analyses across rank and collocation density highlight a clear inflection point of diminishing returns and show that backend choice shifts the optimal configuration across PDE types. Together, these results extend the original FTD-PINN framework with a broader empirical foundation and provide practical guidance for selecting decomposition ranks, collocation densities, and backend activations for efficient PINN design.

Keywords: Keyword1, Keyword2, Keyword3

1 Introduction

Deep learning has accelerated advancements in numerous fields, from computer vision and natural language processing to robotics and autonomous systems (Vaswani et al., 2017; Srivastava & Srivastava, 2013; Emerson & Nichols, 2020; Narayanan & Sundaresan, 2025; Hadri et al., 2021; Denzler & Niemann, 1996). Meanwhile, scientific domains, particularly physics and engineering, have improved by developing complex physical models and numerical methods for solving challenging problems. Despite these advancements in both areas, a crucial opportunity remains untapped, mainly: integrating physical knowledge directly into deep learning models. By embedding the

fundamental laws of physics into neural networks, we can create systems that learn from data and respect the underlying principles governing the real world (Karniadakis et al., 2021; Lin et al., 2021; Cuomo et al., 2022). This integration promises to be a paradigm-shifting advancement, offering the potential to solve problems that have long been computationally intractable using traditional methods.

Physics-Informed Neural Networks (PINNs) (Raissi et al., 2017, 2019; Lagaris et al., 1997) represent a promising direction in this integration. PINNs combine deep learning with constraints imposed by physical laws, enabling models to learn solutions to partial differential equations (PDEs) that adhere to known physical principles. PINNs can solve complex forward and inverse problems with relatively limited data by incorporating physical laws directly into the neural network architecture (Berkhahn & Ehrhardt, 2022; Vemuri & Denzler, 2023; Cuomo et al., 2022; Vemuri et al., 2024a). However, despite their successes, PINNs face significant challenges when applied to high-dimensional PDEs, primarily due to the curse of dimensionality. As dimensions increase, the collocation points needed to achieve an accurate solution grow exponentially. This leads to increased computational costs, making high-dimensional problems intractable (Krishnapriyan et al., 2021; Maddu et al., 2022; Wang et al., 2022, 2021; Vemuri et al., 2024b; Cho et al., 2023).

Our previous work (Vemuri et al., 2024b) addresses this by introducing functional tensor decomposition within the PINN framework. Functional Tensor Decomposition PINNs improve the efficiency and expressivity of physics-informed neural networks by separating variables through low-rank tensor representations. This formulation allows each dimension to be modeled by a compact sub-network while maintaining differentiability for PDE-based constraints, providing an interpretable balance between capacity and physical regularity.

Our earlier work established the feasibility of this approach and demonstrated that functional decomposition can reduce parameter count and training time without sacrificing accuracy. However, the initial study was limited to a small set of configurations and left several practical questions open: How do different decomposition modes (CP, Tucker, Tensor-Train) trade accuracy for efficiency? How do activation backends ranging from standard Tanh to frequency-aware functions such as SIREN Sitzmann et al. (2020) and WIRE Saragadam et al. (2023) affect convergence and stability?

This paper extends the original FTD-PINN study by systematically addressing these questions through controlled experiments on representative Helmholtz and Klein-Gordon equations. We evaluate multiple decomposition-backend combinations under varied ranks and collocation densities, analyzing performance in both accuracy and runtime. The results provide an empirical foundation for selecting architectures and hyperparameters when applying functional tensor decompositions in PINNs, clarifying their benefits and limits across PDE types.

2 Related Work

Enhancing PINNs by using multiple neural networks that work in some combination to produce more accurate and robust solutions. Moseley et al. (2023) proposes splitting the solution domain of a PDE into smaller subdomains, assigning a separate neural network to each one. Thus, each network focuses on learning the solution locally, mimicking a finite element approach. Similarly, Haghighat et al. (2021) introduces a framework in which a separate neural network is used for each variable in a five-variable model of solid mechanics. Cai et al. (2021), in their work on the two-phase Stefan problem, use two neural networks: one to capture the unknown interface between two material phases and the other to model the temperature distributions in each phase (Lu et al., 2021a; Lin et al., 2021).

Jin et al. (2022) proposes using multiple neural networks to improve the training of neural

operators, which learn the solution operator of a PDE instead of solving a specific one. Their architecture combines multiple branches and trunk networks to train deep neural operators more efficiently. Within the PINN space, Cho et al. (2023) introduced separable PINNs, assigning a neural network to each spatial dimension. This reduces the required collocation points and leverages forward-mode automatic differentiation to decrease memory usage and computational cost. Conversely, Krishnapriyan et al. (2021) introduced curriculum learning for PINNs by temporally decomposing the domain. Their curriculum learning approach trains the neural network sequentially across smaller temporal windows, helping mitigate convergence issues in long-time or stiff PDEs.

We propose functional tensor decompositions as a generalized framework for variable separation (Vemuri et al., 2024b). Using tensor decomposition formats, we separate input dimensions and assign dedicated neural networks to approximate each component within the PINN architecture. This results in a flexible and scalable strategy adaptable to the PDE problem.

3 Method

3.1 Physics-Informed Neural Networks

Physics-Informed Neural Networks (PINNs) are a class of models that integrate physical laws, typically in the form of partial differential equations (PDEs), into the training of neural networks (Raissi et al., 2017). Unlike data-driven models, PINNs directly embed governing equations into the optimization term, ensuring the network learns physics-informed solutions. Given a general PDE of the form:

$$\mathcal{F}(\mathbf{x}, u(\mathbf{x}), \nabla u(\mathbf{x}), \nabla^2 u(\mathbf{x}), \dots) = 0 \,, \tag{1}$$

where $\mathbf{x} \in \mathbb{R}^d$ denotes the spatial and/or temporal variables and $u(\mathbf{x})$ is the unknown solution function. The objective of a PINN is to approximate u using a neural network $\tilde{u}(\mathbf{x}; \theta)$ parameterized by θ. For brevity, we omit the θ in the equations.

The total loss $\mathcal{L}$ in a PINN comprises two terms: a data one, $\mathcal{L}_{\text{data}}$, which ensures alignment with available data, and a physics one $\mathcal{L}_{\text{physics}}$, which enforces satisfaction of the PDE at selected collocation points. These components are:

$$\mathcal{L}_{\text{data}} = \frac{1}{N} \sum_{i=1}^{N} \left(\tilde{u}(\mathbf{x}_i) - u_i^{\text{data}} \right)^2 \text{ and} \tag{2}$$

$$\mathcal{L}_{\text{physics}} = \frac{1}{M} \sum_{j=1}^{M} \left(\mathcal{F}(\mathbf{x}_j, \tilde{u}(\mathbf{x}_j), \nabla \tilde{u}(\mathbf{x}_j), \nabla^2 \tilde{u}(\mathbf{x}_j), \dots) \right)^2 , \tag{3}$$

where N and M denote the data and collocation points, respectively. The final optimization objective is

$$\mathcal{L} = \mathcal{L}_{\text{data}} + \lambda \cdot \mathcal{L}_{\text{physics}} \,, \tag{4}$$

with λ acting as a weighting factor to control the contribution of the physics loss (Vemuri & Denzler, 2023; McClenny & Braga-Neto, 2023; Wang et al., 2021).

Collocation points are sampled in the domain to ensure physical consistency across the solution space. As the dimensionality d increases, the required collocation points grow exponentially, restricting classical PINNs approaches. This problem, known as the curse of dimensionality, has been observed in recent studies (Vemuri & Denzler, 2023; Wang et al., 2021, 2022; Krishnapriyan et al., 2021; Maddu et al., 2022; Vemuri et al., 2025).

From a functional approximation perspective, PINNs are models learning from data and the PDE residual to approximate the unknown solution u. In this framework, we further extend this approach by representing the solution as the *outer product of univariate neural networks*, each assigned to a single input dimension. This aligns with the classical *separation of variables* approach but with greater generality and flexibility. Notably, our functional approach remains valid even when the PDE does not admit a separable form (Vemuri et al., 2024b).

The advantages of this decomposition-based formulation over standard PINNs include:

1. **Reduced collocation requirements**: Traditional PINNs require n^d input points for a d-dimensional problem. In contrast, our method requires only $n \cdot d$ points, thereby substantially reducing the computational burden.

2. **Implicit separation of variables**: The solution is expressed in a separable form without requiring the PDE to be classically separable.

3. **Modular learning**: By assigning a distinct neural network to each dimension, the architecture allows more expressive representations and may help avoid local minima in complex solution spaces.

3.2 Functional Tensor Decomposition for PINNs

The classical separation of variables expresses a multivariate function as a sum or product of univariate functions. While powerful, this approach is limited to functions with a separable structure (Raisinghania, 1991). We extend this concept by representing a multivariate function as the outer product of univariate neural networks, one per given input dimension of the PDE. This updated formulation now includes functions that are not classically separable.

Let $f : \mathbb{R}^d \mapsto \mathbb{R}$ be PDE solution. We approximate f using a group of univariate neural networks $\{\hat{g}_i(x_i; \theta_i)\}_{i=1}^d$ as:

$$\hat{f}(x_1, \ldots, x_d) = \bigotimes_{i=1}^{d} \hat{g}_i(x_i; \theta_i)\,. \tag{5}$$

Each $\hat{g}_i$ is a neural network that learns along the i-th input axis. In our prior work (Vemuri et al., 2024b), we show that this outer product structure satisfies a universal approximation property, i.e., the outer product of sufficiently expressive univariate networks can approximate any continuous multivariate function on a compact domain. We also establish error bounds that relate the total approximation error to the quality of each component. For theoretical background, we refer to (Vemuri et al., 2024b; Messiter & Shamash, 1985; Herath, 2022; Hornik et al., 1989; Hornik, 1991; Cybenko, 1989).

Tensor Decomposition Formats To construct such separable approximations, we draw from classical tensor decomposition techniques (Kolda & Hong, 2020; Kolda & Bader, 2009). We define the functional form of each decomposition using neural networks to learn the decomposition components, as illustrated in the graphical abstract. We focus on three established decompositions:

- **Canonic-Polyadic (CP)** (Hitchcock, 1927): Approximates a d-dimensional tensor using d factor matrices. For a function f, the CP decomposition is expressed as:

$$f(x_1, x_2, \ldots, x_d) \approx [[A_1(x_1), A_2(x_2), \ldots, A_d(x_d)]]\,, \tag{6}$$

  where $A_i(x_i) \in \mathbb{R}^{n_i \times R}$ are rank-R factor matrices.

- **Tensor Train (TT)** (Oseledets, 2011): Represents a function using a sequence of low-rank tensor cores connected in a chain. Each core A_i has the shape $A_i \in \mathbb{R}^{R_{i-1} \times n_i \times R_i}$, with $R_0 = R_d = 1$. TT improves numerical stability by limiting connections to adjacent dimensions.
- **Tucker (TU)** (Tucker, 1966): Generalizes CP by introducing a learnable core tensor $\mathcal{C}$ and factor matrices along each dimension:

$$f(x_1, \ldots, x_d) \approx [[\mathcal{C}; A_1(x_1), A_2(x_2), \ldots, A_d(x_d)]], \tag{7}$$

  where $\mathcal{C} \in \mathbb{R}^{R_1 \times \cdots \times R_d}$ is the core tensor and $A_i \in \mathbb{R}^{n_i \times R_i}$.

The decomposition structure directly influences how spatial and temporal derivatives propagate through the network during the computation of PDE residuals. In a standard PINN, gradients concerning all input variables are entangled within a single dense network. In contrast, the functional tensor decomposition splits the mapping into univariate subnetworks, allowing derivatives to be computed along independent axes and combined through tensor operations. This changes the conditioning and coupling of the residual terms: CP decompositions promote complete separability, TT restricts coupling to adjacent dimensions (improving numerical stability), and Tucker introduces a shared core capturing global dependencies.

4 Experiments

This section extends the experimental analysis in our earlier work (Vemuri et al., 2024b). The goal is to systematically assess how decomposition mode, decomposition rank, and backend architecture jointly influence accuracy, efficiency, and stability across representative PDEs.

The FTD-PINN framework, as defined in (Vemuri et al., 2024b), is characterized by three controllable components, each of which contributes to determining the model's performance in terms of computational efficiency and solution accuracy. These components are: (i) the tensor decomposition mode, (ii) the decomposition rank, and (iii) the neural network architecture.

Our earlier work (Vemuri et al., 2024b) focused on evaluating the efficacy and speed of different tensor decomposition modes (CP, TT, and Tucker) in approximating PDE solutions compared to other state-of-the-art PINNs. That initial analysis was foundational in introducing these decomposition formats within the PINN framework. In this extended study, we expand our investigation to include analyses of the components of FTD-PINN and their individual and combined effects on performance. We retain the standard PINN loss and optimization procedure, with no modifications to the weighting or optimizer; the focus here is on how different functional decompositions and backends influence performance.

4.1 Experimental Setup

All experiments are performed on two canonical PDEs that capture complementary behaviors, steady-state oscillations, and transient wave propagation, serving as benchmarks for decomposition-based PINNs.

- **Klein-Gordon Equation** in (2+1)D: A time-dependent PDE used to model wave propagation in relativistic quantum mechanics (Lu et al., 2021b; Cho et al., 2023), given in Equation 8:

$$\begin{aligned} f &= \partial_{tt} u - \Delta u + u^2 \,, \\ u(x, y, 0) &= x + y + x \cdot y \cdot \sin(2t) \,, \\ [x, y] &\in [-1, 1]^2, t \in [0, 10] \,. \end{aligned} \tag{8}$$

where u is the solution to be solved in domain (x, y, t). The operators ∂_{tt} and Δ are the second-order time derivative and Laplacian, respectively. The ground truth is given as $u = (x + y) \cdot \cos(2t)$ and visualized in Figure 1.

- **Helmholtz Equation** in 3D: A steady-state wave equation that frequently arises in acoustics, electromagnetics, and fluid dynamics Cuomo et al. (2022), given in Equation 9 and visualized in Figure 2:

$$\begin{aligned} \Delta u + k^2 u &= q, x \in [-1, 1]^3 \,, \\ u(x) &= 0, x \in \partial\Omega \,, \\ q &= -(a_1\pi)^2 u - (a_2\pi)^2 u \\ &\quad - (a_3\pi)^2 u + k^2 u \,, \\ u &= \prod_{i=1}^{3} \sin(a_i \pi x_i) \,. \end{aligned} \tag{9}$$

  where u is the underlying solution of the PDE in the domain $\Omega \in [-1, 1]^3$ and $\partial\Omega$ is the boundary of Ω.

These equations are selected for their real-world relevance, complex features such as high-frequency oscillations, and the known difficulty that classical PINNs face in accurately solving them (Wang et al., 2022, 2021; Krishnapriyan et al., 2021). Our earlier work also used them, enabling consistent comparison while extending the analysis to additional architectural variables. We report the mean and standard deviation of the relative L_2 error for each configuration across multiple runs.

We examine four backend architectures: Tanh, Tanh with positional encoding (PE), SIREN, and WIRE. Each backend models the univariate components within the tensor decomposition. All networks are trained using identical hyperparameters to ensure consistency across experiments.

- **Hyperbolic Tangent** (Dubey et al., 2022): A smooth, bounded activation commonly used in PINN architectures.
- **SIREN** (Sitzmann et al., 2020): Sinusoidal activation functions designed for implicit neural representations are known for superior performance in learning high-frequency patterns in PDEs.
- **Positional Encoding (PE)** (Mildenhall et al., 2021): Added to the input layer to inject frequency bias, improving convergence and helping escape local minima (Wang et al., 2021).
- **WIRE** (Saragadam et al., 2023): A Gabor Wavelet activation combines sinusoidal and Gaussian components to capture global frequency and localized features. This blends the benefits of SIRENs and localized learning through wavelets, given by $\psi(x) = \cos(\omega x) \cdot \exp((-sx)^2)$, where ω controls frequency and s controls scale.

All models are trained using Adam (Kingma & Ba, 2014) with a learning rate of 10^{-3} for 50,000 epochs. This setup ensures fair comparisons across different architectures. Following Saragadam et al. (2023), we initially set $\omega = s = 10$ for baseline comparisons and perform a focused sweep over (ω, s) to study their effect on convergence and residual smoothness.

In addition to backend architectures, we study the effects of decomposition **rank** and the **collocation points**, two parameters critical to the scalability and expressivity of FTD-PINNs. We vary the rank in the range $R \in \{16, 32, 64, 128, 256\}$ and test the same range for the collocation points (NC) per dimension. Our analysis is organized around the following questions:

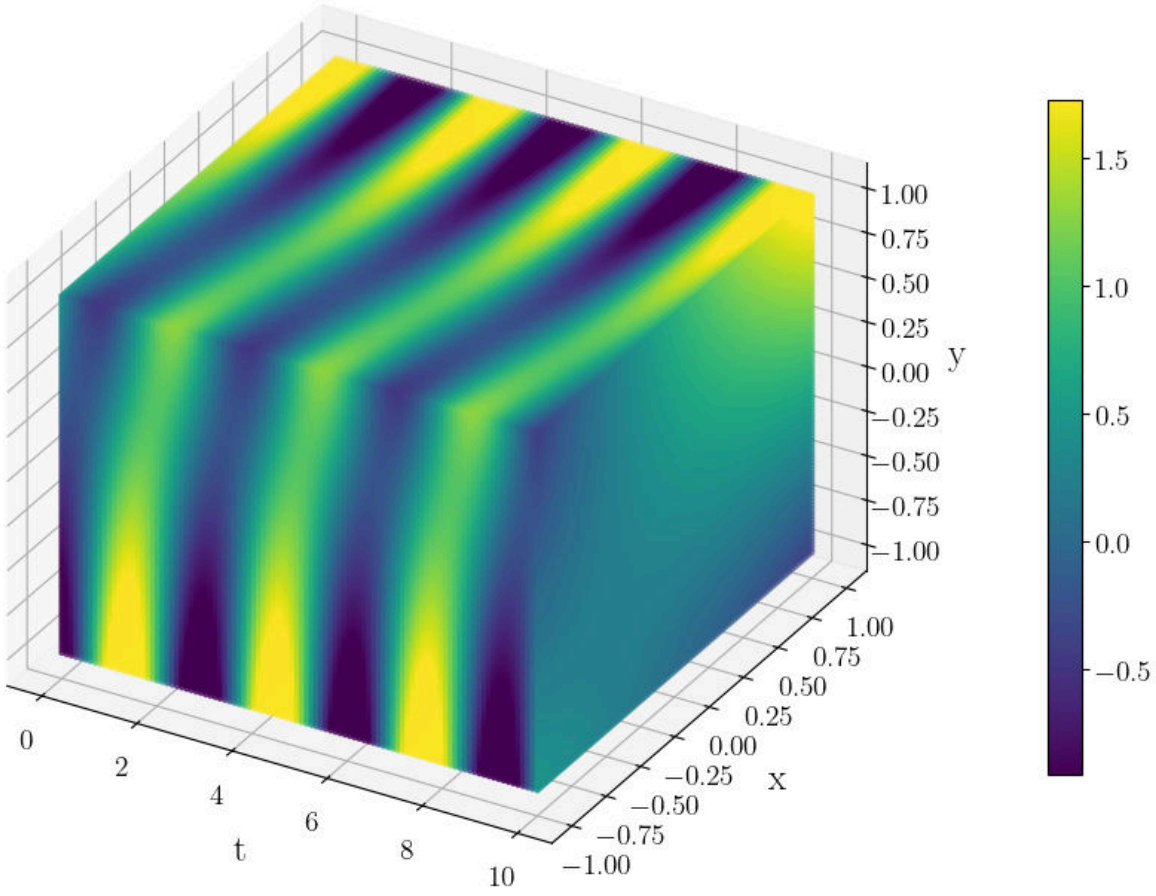


Figure 1: Visualization of the Klein-Gordon PDE described in Equation 8.

1. How do different decomposition modes (CP, TT, Tucker) trade off accuracy, stability, and computational efficiency across PDEs?
2. How do backend activations (Tanh, PE, SIREN, WIRE) influence convergence behavior and residual smoothness?
3. How do decomposition rank and collocation density affect the accuracy–efficiency balance of FTD-PINNs?

For this, we present heatmaps of L_2 errors between the ground truth and PINN solutions for various combinations of collocation points (NC) and decomposition rank (R) across three tensor decomposition modes (CP, TT, Tucker) and four backend architectures. This results in 12 heatmaps, each illustrating how error varies with NC and R for a given mode-backend pair, offering insights into the interplay among the different constituents that form a PINN.

First, the heatmaps for Helmholtz PDE are given in Figure 3. Increasing the collocation points consistently improves solution accuracy, regardless of the backend. The WIRE architecture yields the lowest errors among backends, followed by SIREN. This is seen in the Table 1, where the top three lowest errors for each tensor decomposition mode are presented. This performance is at-

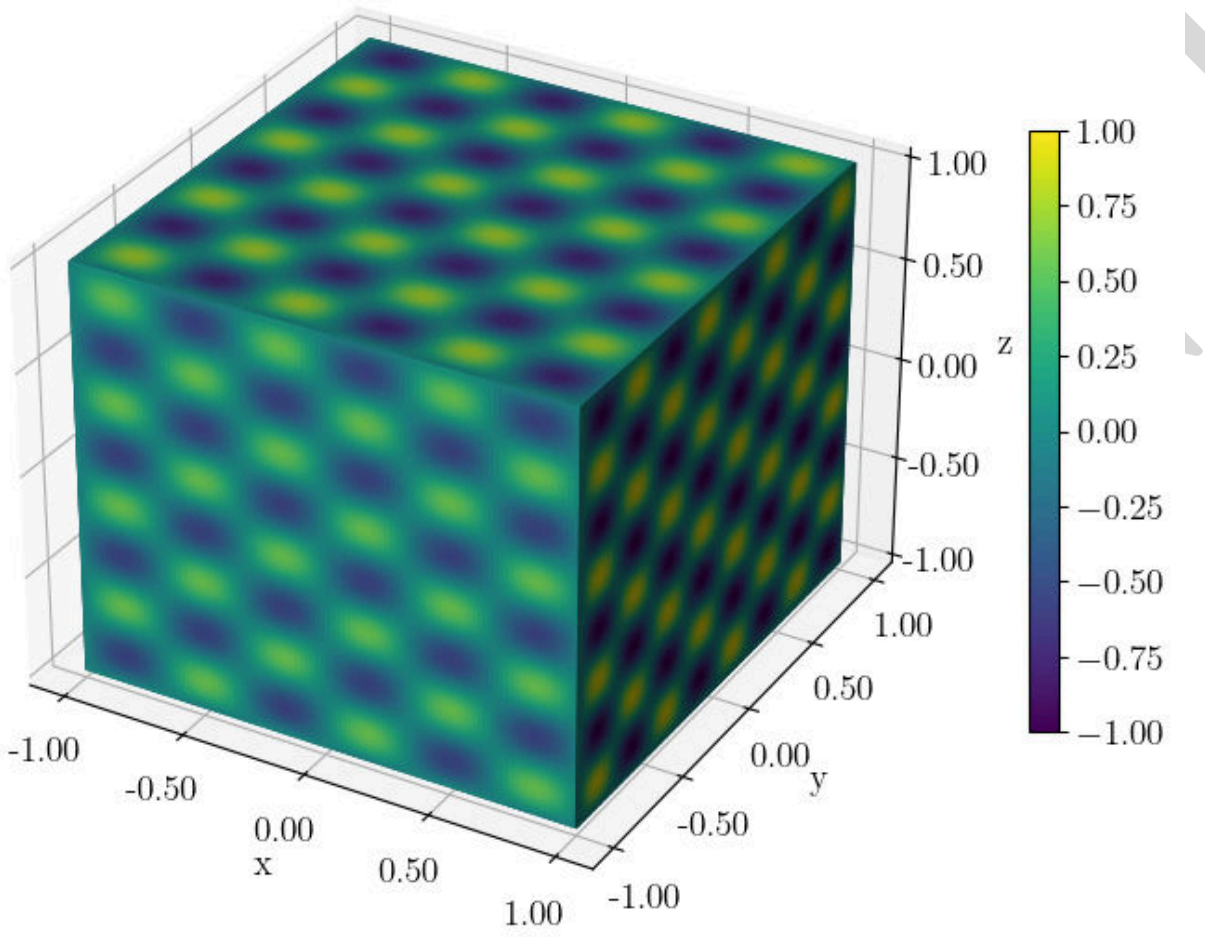


Figure 2: Visualization of the Helmholtz PDE described in Equation 9.

tributed to the expressive Gabor wavelet activations in WIRE, which effectively capture both local and high-frequency features, and the sinusoidal nature of SIREN, which matches the behavior of Helmholtz PDE. Across decomposition modes, TT consistently outperforms the others, even with lower ranks and fewer collocation points, as seen in most combinations. Its sequential structure enables more stable training and smoother information propagation (Kolda & Hong, 2020; Kolda & Bader, 2009; Oseledets, 2011). Additionally, incorporating positional encoding significantly improves performance over standard *Tanh* alone, highlighting its role in accelerating convergence and improving solution quality. The heatmaps for the Klein-Gordon equation (with color bar capped at 0.1) show overall lower L_2 errors, indicating the problem is generally easier to solve, in Figure 4. This aligns with its traveling-wave nature, which lacks the high-frequency components that make the Helmholtz PDE more challenging. The general trend remains consistent: increasing the number of collocation points typically improves performance. However, with the TT decomposition and Tanh backend, higher ranks and dense collocation occasionally worsen results, likely due to poor local minima. This effect is absent in more expressive backends, such as WIRE or PE, highlighting the limitations of simpler activations. The top three performing combinations are in the Table 2.

We construct Pareto frontiers of the mean L_2 error versus time for training the Klein-Gordon

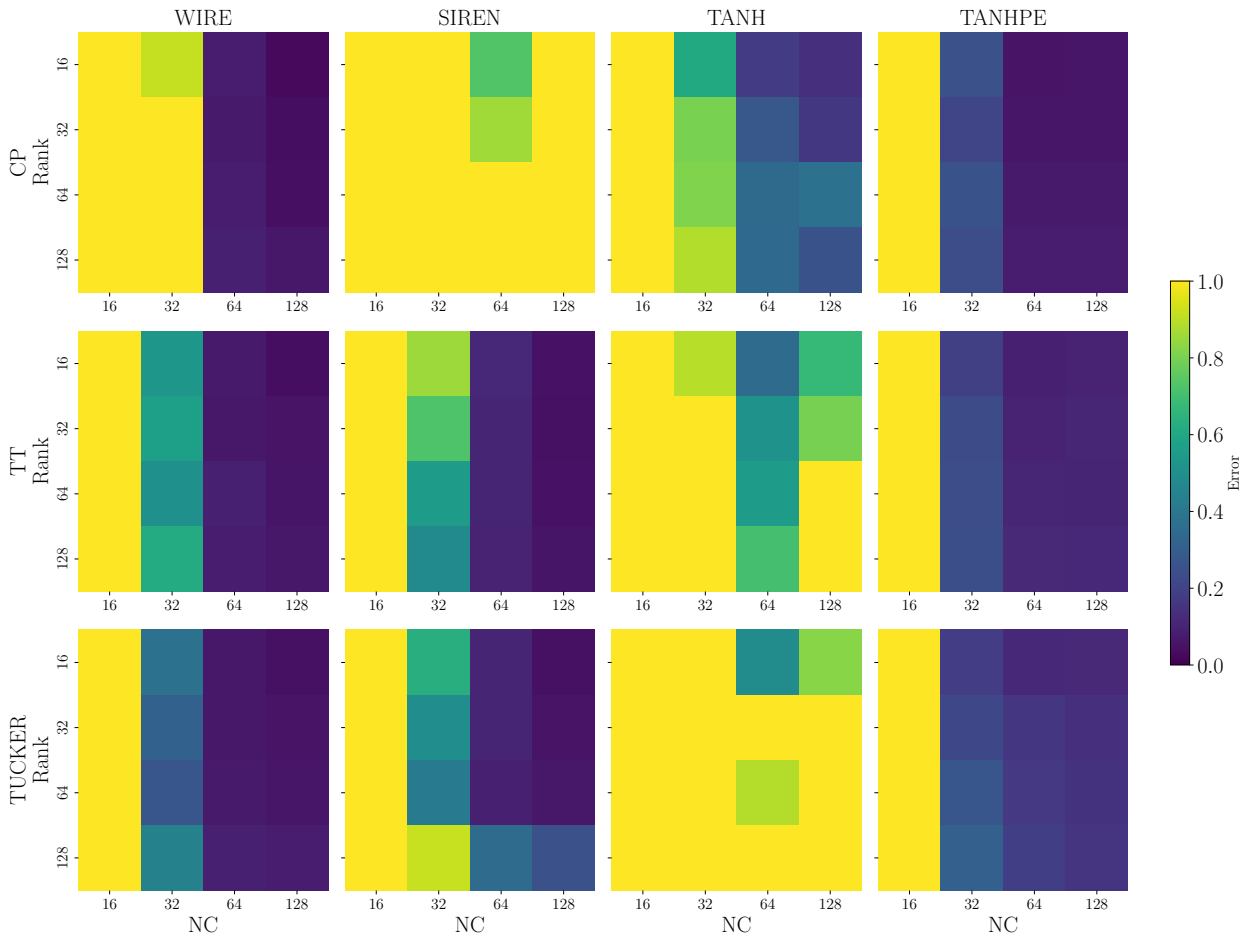


Figure 3: Heatmaps of L_2 error between ground truth and PINN solution for 3D Helmholtz Equation for combinations of collocation points (NC), decomposition rank (R), given for three decomposition modes (CP, TT, and Tucker), and four different backends. The general trend we observe among backends is that WIRE and Positional Encoding perform best.

PDE to analyze the efficiency-accuracy relationship, as shown in Figure 5. Each marker represents a trained configuration of a given decomposition mode (shape), rank (color), and number of collocation points (size). The Pareto curve traces configurations that achieve the best attainable accuracy for a given computational cost, highlighting the trade-off boundary beyond which additional computation yields diminishing returns. This analysis allows a fair comparison between decomposition modes and backends without relying on arbitrary hyperparameter choices. The fact that most configurations cluster near the frontier indicates that the explored rank-collocation settings already operate close to the optimal efficiency regime for this problem. Furthermore, we observe that the WIRE backend has more points near the front than the tanh function, underscoring WIRE's effectiveness as a backend. The Pareto frontiers exhibit a definite inflection point, marking the optimal trade-off between accuracy and runtime. Configurations near this inflection point represent the most efficient balance: improving accuracy further would require a disproportionately higher computational cost. Across both backends, similar trends are observed: CP and TT decompositions dominate near the Pareto inflection point, typically at moderate ranks (16-32) and collocation densities around 64, indicating that neither very low nor very high parameterizations are optimal.

Table 1: L_2 errors for the top three best-performing combinations for each tensor decomposition mode on the Helmholtz PDE benchmark. We observe that WIRE Saragadam et al. (2023) performs best, followed by SIREN Sitzmann et al. (2020), with all the models having large collocation points (NC).

Mode	Backend	Rank	NC	L_2 Error
CP	WIRE	16	128	**0.0094 ± 0.0019**
CP	WIRE	32	128	0.0100 ± 0.0030
CP	WIRE	64	128	0.0117 ± 0.0007
TT	WIRE	16	128	0.0122 ± 0.0035
TT	SIREN	16	128	0.0132 ± 0.0059
TT	SIREN	32	128	0.0147 ± 0.0050
TU	WIRE	16	128	0.0153 ± 0.0012
TU	SIREN	16	128	0.0141 ± 0.0055
TU	SIREN	32	128	0.0158 ± 0.0037

For visualization clarity, only configurations with an error below 0.6 and training time under 250 seconds are plotted; thus, some high-cost or low-performing runs are omitted. These trends suggest that efficient configurations occur in a narrow band of model capacity, offering a practical guideline for selecting rank and collocation settings.

Since the initial results established WIRE as the best-performing architecture, we examine the Gabor Wavelet activation used in WIRE (Saragadam et al., 2023) in more detail. We evaluate the effect of wavelet activation parameters ω (frequency) and s (scale) with a fixed rank and collocation points ($NC = R = 64$) on the CP decomposition mode. We range ω and s from 0 to 50 in five equidistant intervals over ten independent runs. The mean L_2 errors are shown as heat maps. Figure 6a shows results for the Helmholtz equation. Pure Gaussian activations ($\omega = 0$) and pure sinusoidal ones ($s = 0$, like SIRENs (Sitzmann et al., 2020)) are shown in the first row and column, respectively. Sinusoidal-dominant configurations perform best and are consistent with the equation's high-frequency nature. The lowest error (0.0000955) is achieved at $\omega = 40$, $s = 0$. This activation outperforms all the other combinations of ranks and architecture (see Table 1). Figure 6b presents results for the Klein-Gordon equation, where balanced values of ω and s yield better accuracy. Pure activations underperform, with the best result (0.00052) occurring at $\omega = 10$, $s = 10$, matching the equation's traveling wave behavior. Similarly, as seen in the previous case, this combination outperforms all the other combinations of ranks and architectures (see Table 2). Figure 7 further examines how varying ω at selected s values influences accuracy. For the Helmholtz PDE, increasing ω consistently reduces the error's mean and standard deviation across scales, mirroring the benefit of higher frequency resolution for oscillatory solutions. In contrast, for the Klein-Gordon PDE, larger ω values lead to unstable or less accurate behavior across most scales, likely because the activation's high-frequency bias conflicts with the equation's smoother traveling-wave dynamics.

Overall, these results indicate that optimal WIRE parameters are PDE-specific: high-frequency activations benefit Helmholtz-type problems, while moderate (ω, s) values are preferable for mixed or transient dynamics. As a general guideline, initializing WIRE with $\omega = s = 10$ provides a robust balance before PDE-specific tuning.

To verify that the observations from the benchmark PDEs generalize beyond these benchmark cases, we apply the same framework to more PDEs (A.1,A.2). In these experiments, we follow the

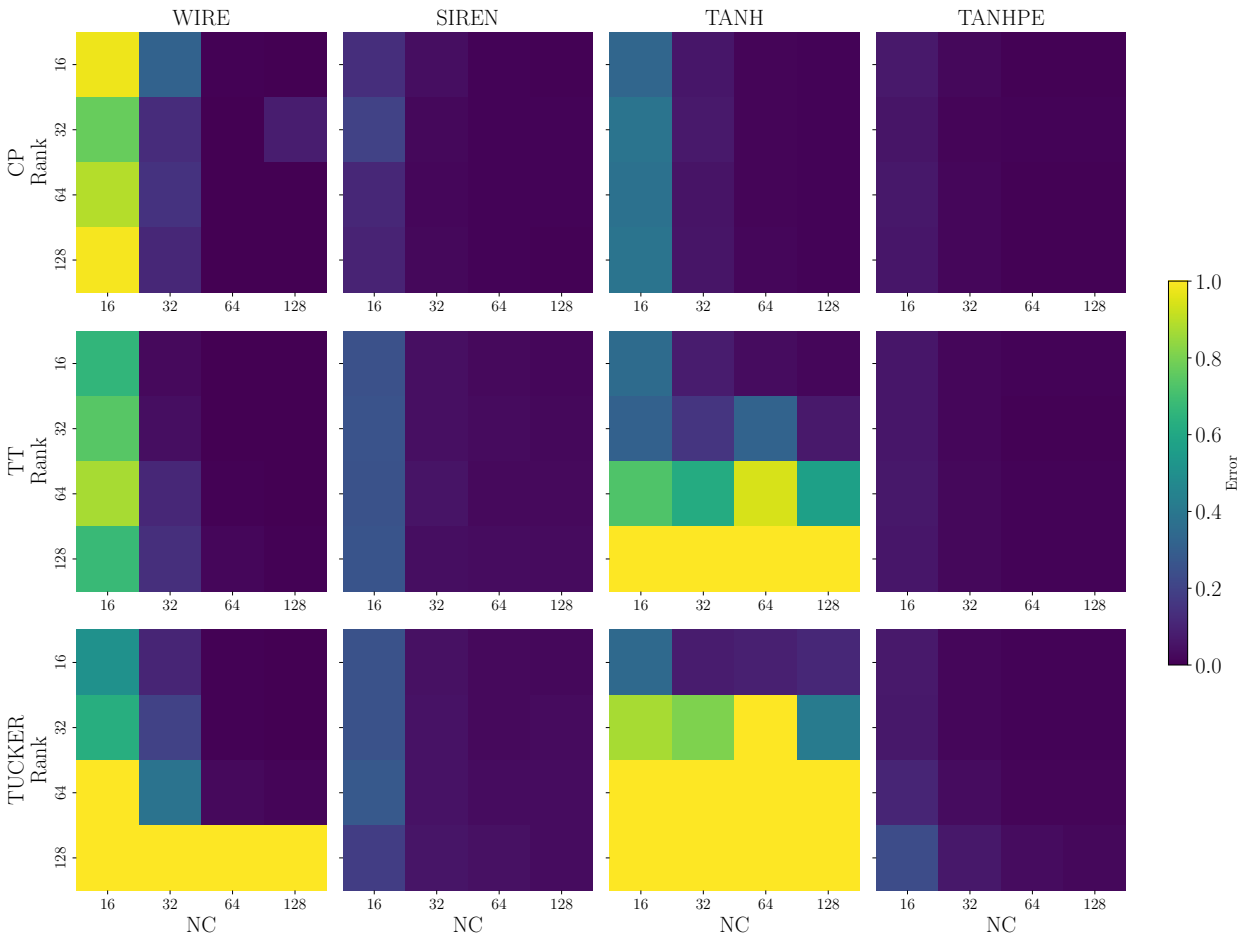


Figure 4: Heatmaps of L_2 error between ground truth and PINN solution for Klein-Gordon PDE for various collocation points (NC), decomposition rank (R), given for three decomposition modes (CP, TT, and TU), and four different backends. The general trend is that, irrespective of the backend, a rank with a larger collocation points leads to a better solution.

recommendations derived from the Pareto analysis and use moderate rank and collocation settings, along with the best-performing backends identified in the main study. Moreover, we extend the application of FTD-PINNs to parametric PDE settings, demonstrating their versatility (see B). In addition, we summarize the key empirical findings in a practitioner guide that provides practical recommendations for selecting the decomposition mode, rank, collocation density, and backend architecture (C).

5 Conclusion

In this work, we extended our earlier Functional Tensor Decomposition PINN (FTD-PINN) framework through a focused architectural study. We systematically evaluated how tensor decomposition modes, ranks, and activation backends influence accuracy and efficiency in solving representative PDEs. The analysis, supported by Pareto trade-offs and a detailed WIRE parameter study, demonstrates that expressive frequency-aware backends (WIRE, SIREN) consistently improve convergence. At the same time, moderate ranks and collocation densities achieve the best accuracy-cost

Table 2: L_2 errors for the top three best-performing combinations for each tensor decomposition mode on the Klein-Gordon PDE benchmark. We observe that the WIRE architecture outperforms all other combinations Saragadam et al. (2023), and all the models have large collocation points (NC).

Mode	Backend	Rank	NC	L_2 Error
CP	WIRE	16	128	0.0002 ± 0.0001
CP	WIRE	64	128	0.0002 ± 0.0001
CP	WIRE	128	128	0.0002 ± 0.0001
TT	WIRE	16	64	0.0002 ± 0.0000
TT	WIRE	16	128	0.0001 ± 0.0000
TT	WIRE	32	128	0.0001 ± 0.0000
TU	WIRE	16	64	0.0002 ± 0.0001
TU	WIRE	16	128	0.0001 ± 0.0001
TU	WIRE	32	64	0.0001 ± 0.0000

balance. These results clarify how decomposition structure and activation choice jointly determine training stability and computational efficiency. The work provides practical guidance for configuring PINN architectures under fixed compute budgets and establishes a strong foundation for future studies on automated rank-backend tuning and adaptive collocation strategies (Song et al., 2021; Takamoto et al., 2022; Escapil-Inchauspé & Ruz, 2023).

References

Berkhahn, S., & Ehrhardt, M. (2022). A physics-informed neural network to model covid-19 infection and hospitalization scenarios. *Advances in Continuous and Discrete Models*, *2022*, 61. doi:10.1186/s13662-022-03733-5.

Cai, S., Wang, Z., Wang, S., Perdikaris, P., & Karniadakis, G. E. (2021). Physics-Informed Neural Networks for Heat Transfer Problems. *Journal of Heat Transfer*, *143*, 060801. doi:10.1115/1.4050542.

Cho, J., Nam, S., Yang, H., Yun, S.-B., Hong, Y., & Park, E. (2023). Separable physics-informed neural networks. In *Advances in Neural Information Processing Systems 36 (NeurIPS 2023)*.

Cuomo, S., Di Cola, V. S., Giampaolo, F., Rozza, G., Raissi, M., & Piccialli, F. (2022). Scientific machine learning through physics–informed neural networks: Where we are and what's next. *Journal of Scientific Computing*, *92*, 88. doi:10.1007/s10915-022-01939-z.

Cybenko, G. (1989). Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals and Systems*, *2*, 303–314. doi:10.1007/BF02551274.

Denzler, J., & Niemann, H. (1996). 3d data driven prediction for active contour models based on geometric bounding volumes. *Pattern recognition letters*, *17*, 1171–1178.

Dubey, S. R., Singh, S. K., & Chaudhuri, B. B. (2022). Activation functions in deep learning: A comprehensive survey and benchmark. *Neurocomputing*, *503*, 92–108. doi:10.1016/j.neucom.2022.06.111.

Emerson, T. H., & Nichols, J. M. (2020). Fitting local, low-dimensional parameterizations of optical turbulence modeled from optimal transport velocity vectors. *Pattern Recognition Letters*, *133*, 123–128. doi:https://doi.org/10.1016/j.patrec.2019.10.023.

Escapil-Inchauspé, P., & Ruz, G. A. (2023). Hyper-parameter tuning of physics-informed neural networks: Application to helmholtz problems. *Neurocomputing*, *561*, 126826. doi:https://doi.org/10.1016/j.neucom.2023.126826.

Hadri, A., Laghrib, A., & Oummi, H. (2021). An optimal variable exponent model for magnetic resonance images denoising. *Pattern Recognition Letters*, *151*, 302–309. doi:https://doi.org/10.1016/j.patrec.2021.08.031.

Haghighat, E., Raissi, M., Moure, A., Gomez, H., & Juanes, R. (2021). A physics-informed deep learning framework for inversion and surrogate modeling in solid mechanics. *Computer Methods in Applied Mechanics and Engineering*, *379*, 113741. doi:10.1016/j.cma.2021.113741.

Herath, M. I. U. (2022). *Multivariate Regression using Neural Networks and Sums of Separable Functions*. Ph.D. thesis Ohio University USA.

Hitchcock, F. L. (1927). The expression of a tensor or a polyadic as a sum of products. *Journal of Mathematics and Physics*, *6*, 164–189. doi:10.1002/sapm192761164.

Hornik, K. (1991). Approximation capabilities of multilayer feedforward networks. *Neural Networks*, *4*, 251–257. doi:10.1016/0893-6080(91)90009-T.

Hornik, K., Stinchcombe, M., & White, H. (1989). Multilayer feedforward networks are universal approximators. *Neural Networks*, *2*, 359–366. doi:10.1016/0893-6080(89)90020-8.

Jin, P., Meng, S., & Lu, L. (2022). Mionet: Learning multiple-input operators via tensor product. *SIAM Journal on Scientific Computing*, *44*, A3490–A3514. doi:10.1137/22M1477751.

Karniadakis, G. E., Kevrekidis, I. G., Lu, L., Perdikaris, P., Wang, S., & Yang, L. (2021). Physics-informed machine learning. *Nature Reviews Physics*, *3*, 422–440. doi:10.1038/s42254-021-00314-5.

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

Kolda, T. G., & Bader, B. W. (2009). Tensor decompositions and applications. *SIAM Review*, *51*, 455–500. doi:10.1137/07070111X.

Kolda, T. G., & Hong, D. (2020). Stochastic gradients for large-scale tensor decomposition. *SIAM Journal on Mathematics of Data Science*, *2*, 1066–1095. doi:10.1137/19M1266265.

Krishnapriyan, A., Gholami, A., Zhe, S., Kirby, R., & Mahoney, M. W. (2021). Characterizing possible failure modes in physics-informed neural networks. *Advances in neural information processing systems*, *34*, 26548–26560. doi:10.5555/3540261.3542294.

Lagaris, I., Likas, A., & Fotiadis, D. (1997). Artificial neural network methods in quantum mechanics. *Computer Physics Communications*, *104*, 1–14. doi:10.1016/S0010-4655(97)00054-4.

Lin, C., Maxey, M., Li, Z., & Karniadakis, G. E. (2021). A seamless multiscale operator neural network for inferring bubble dynamics. *Journal of Fluid Mechanics*, *929*, A18. doi:10.1017/jfm.2021.866.

Lu, L., Jin, P., Pang, G., Zhang, Z., & Karniadakis, G. E. (2021a). Learning nonlinear operators via deeponet based on the universal approximation theorem of operators. *Nature Machine Intelligence*, *3*, 218–229. doi:10.1038/s42256-021-00302-5.

Lu, L., Meng, X., Mao, Z., & Karniadakis, G. E. (2021b). DeepXDE: A deep learning library for solving differential equations. *SIAM Review*, *63*, 208–228. doi:10.1137/19M1274067.

Maddu, S., Sturm, D., Müller, C. L., & Sbalzarini, I. F. (2022). Inverse dirichlet weighting enables reliable training of physics informed neural networks. *Machine Learning: Science and Technology*, *3*, 015026. doi:10.1088/2632-2153/ac3712.

McClenny, L. D., & Braga-Neto, U. M. (2023). Self-adaptive physics-informed neural networks. *Journal of Computational Physics*, *474*, 111722. doi:10.1016/j.jcp.2022.111722.

Messiter, M., & Shamash, Y. (1985). Product and sum separable functions. *IEEE Transactions on Automatic Control*, *30*, 694–697. doi:10.1109/TAC.1985.1104018.

Mildenhall, B., Srinivasan, P. P., Tancik, M., Barron, J. T., Ramamoorthi, R., & Ng, R. (2021). Nerf: representing scenes as neural radiance fields for view synthesis. *Commun. ACM*, *65*, 99–106. doi:10.1145/3503250.

Moseley, B., Markham, A., & Nissen-Meyer, T. (2023). Finite basis physics-informed neural networks (fbpinns): a scalable domain decomposition approach for solving differential equations. *Advances in Computational Mathematics*, *49*, 62. doi:10.1007/s10444-023-10065-9.

Narayanan, R., & Sundaresan, V. (2025). Medlessynth-ld: Lesion synthesis using physics-based noise models for robust lesion segmentation in low-data medical imaging regimes. *Pattern Recognition Letters*, *188*, 155–163. doi:https://doi.org/10.1016/j.patrec.2024.12.011.

Oseledets, I. V. (2011). Tensor-train decomposition. *SIAM Journal on Scientific Computing*, *33*, 2295–2317. doi:10.1137/090752286.

Raisinghania, M. (1991). *Ordinary and Partial Differential Equations*. S. Chand Publishing.

Raissi, M., Perdikaris, P., & Karniadakis, G. E. (2017). Physics informed deep learning (part ii): Data-driven discovery of nonlinear partial differential equations.

Raissi, M., Perdikaris, P., & Karniadakis, G. E. (2019). 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. doi:10.1016/j.jcp.2018.10.045.

Saragadam, V., LeJeune, D., Tan, J., Balakrishnan, G., Veeraraghavan, A., & Baraniuk, R. G. (2023). Wire: Wavelet implicit neural representations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 18507–18516).

Sitzmann, V., Martel, J. N., Bergman, A. W., Lindell, D. B., & Wetzstein, G. (2020). Implicit neural representations with periodic activation functions. In *Proc. NeurIPS*.

Song, C., Alkhalifah, T., & Waheed, U. (2021). Solving the frequency-domain acoustic vti wave equation using physics-informed neural networks. *Geophysical Journal International*, *225*, 846–859. doi:10.1093/gji/ggab010.

Srivastava, R., & Srivastava, S. (2013). Restoration of poisson noise corrupted digital images with nonlinear pde based filters along with the choice of regularization parameter estimation. *Pattern Recognition Letters*, *34*, 1175–1185. doi:https://doi.org/10.1016/j.patrec.2013.03.026.

Takamoto, M., Praditia, T., Leiteritz, R., MacKinlay, D., Alesiani, F., Pflüger, D., & Niepert, M. (2022). PDEBench Datasets. doi:10.18419/DARUS-2986.

Tucker, L. R. (1966). Some mathematical notes on three-mode factor analysis. *Psychometrika*, *31*, 279–311. doi:10.1007/BF02289464.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). Attention is all you need. In *Advances in Neural Information Processing Systems* (pp. 5998–6008).

Vemuri, S. K., Büchner, T., & Denzler, J. (2024a). Estimating soil hydraulic parameters for unsaturated flow using physics-informed neural networks. In *Computational Science – ICCS 2024* (pp. 338–351). Cham: Springer Nature Switzerland. doi:10.1007/978-3-031-63759-9_37.

Vemuri, S. K., Büchner, T., & Denzler, J. (2025). F-inr: Functional tensor decomposition for implicit neural representations. *arXiv preprint arXiv:2503.21507*, .

Vemuri, S. K., Büchner, T., Niebling, J., & Denzler, J. (2024b). Functional tensor decompositions for physics-informed neural networks. In *Pattern Recognition: 27th International Conference, ICPR 2024, Kolkata, India, December 1–5, 2024, Proceedings, Part XXV* (p. 32–46). Berlin, Heidelberg: Springer-Verlag. doi:10.1007/978-3-031-78389-0_3.

Vemuri, S. K., & Denzler, J. (2023). Gradient statistics-based multi-objective optimization in physics-informed neural networks. *Sensors*, *23*. doi:10.3390/s23218665.

Wang, S., Teng, Y., & Perdikaris, P. (2021). Understanding and mitigating gradient flow pathologies in physics-informed neural networks. *SIAM Journal on Scientific Computing*, *43*, A3055–A3081. doi:10.1137/20M1318043.

Wang, S., Yu, X., & Perdikaris, P. (2022). When and why pinns fail to train: A neural tangent kernel perspective. *Journal of Computational Physics*, *449*, 110768. doi:10.1016/j.jcp.2021.110768.

A Solving Additional PDEs

The main paper benchmarks how the core design choices of FTD PINNs, namely the decomposition mode and rank, the collocation density, and the backend MLP, affect solution accuracy and runtime. In this appendix, we apply the same framework to additional PDEs to show that the conclusions hold for more than just the two main benchmarks.

We consider two three-dimensional PDEs with distinct dominant behavior. One is transport-dominated (Advection–Diffusion), and one is nonlinear (Allen–Cahn). Throughout these experiments, we keep the training pipeline and the decomposition modes unchanged. We use settings for rank and collocation density, with NC = rank $\in [32, 64]$, motivated by the region of the Pareto trade-off observed in the main benchmarks. As backends, we use WIRE, SIREN, and Tanh with positional encoding, since they performed best in the main study.

All PDEs are solved on a cubic domain. Dirichlet boundary values are prescribed from a manufactured ground truth solution. We report the relative L_2 errors for the best-performing configurations.

A.1 Three-dimensional Advection–Diffusion

We consider a steady advection–diffusion equation

$$-\varepsilon \Delta u(\mathbf{x}) + \mathbf{b} \cdot \nabla u(\mathbf{x}) = f(\mathbf{x}) \,, \tag{10}$$

$$\mathbf{x} = (x, y, z) \in [-1, 1]^3 \,, \tag{11}$$

with Dirichlet boundary conditions $u(\mathbf{x}) = u_{\text{gt}}(\mathbf{x})$ for $\mathbf{x} \in \partial\Omega$. Here, $\varepsilon > 0$ is the diffusion coefficient and $\mathbf{b} \in \mathbb{R}^3$ is a constant advection velocity.

We use a manufactured solution

$$u_{\text{gt}}(x, y, z) = \exp(x) \cdot \sin(\pi y) \cdot \sin(2\pi z) \,, \tag{12}$$

and define the forcing term by substituting u_{gt} into Equation 11, that is,

$$f(\mathbf{x}) = -\varepsilon \Delta u_{\text{gt}}(\mathbf{x}) + \mathbf{b} \cdot \nabla u_{\text{gt}}(\mathbf{x}) \,. \tag{13}$$

This benchmark is transport-dominated at small ε and highlights the model's ability to represent directional gradients reliably. In this case, we chose $\varepsilon = 0.001$. The solution visualization is shown in Figure 8, and the corresponding L_2 errors for solving this PDE are shown in Table 3.

A.2 Three-dimensional Allen–Cahn Type Nonlinear PDE

We next consider a steady nonlinear reaction diffusion equation of Allen–Cahn type

$$-\Delta u(\mathbf{x}) + \alpha \left(u(\mathbf{x})^3 - u(\mathbf{x}) \right) = f(\mathbf{x}) \,, \tag{14}$$

$$\mathbf{x} = (x, y, z) \in [-1, 1]^3 \,, \tag{15}$$

with Dirichlet boundary conditions $u(\mathbf{x}) = u_{\text{gt}}(\mathbf{x})$ on $\partial\Omega$, and a reaction strength $\alpha > 0$.

We use the manufactured solution

$$u_{\text{gt}}(x, y, z) = 0.5 + 0.25 \, \sin(\pi x) \, \sin(\pi y) \, \sin(\pi z) \,, \tag{16}$$

and define the forcing term

Table 3: L_2 errors for the top performing combinations for each tensor decomposition mode on the Advection-Diffusion PDE benchmark. We observe that the optimal parameter combinations with rank = NC = 64 achieve the best performance, and the WIRE backing is found to be more accurate and robust across all modes.

Mode	Backend	Rank	NC	L_2 Error
CP	Tanh + PE	32	64	0.0042 ± 0.0022
CP	WIRE	32	64	0.0024 ± 0.0017
CP	WIRE	64	64	0.0022 ± 0.0016
TT	Tanh + PE	32	64	0.0053 ± 0.0026
TT	WIRE	32	64	0.0019 ± 0.0008
TT	WIRE	64	64	0.0018 ± 0.0007
TU	Tanh + PE	32	64	0.0086 ± 0.0044
TU	WIRE	32	64	0.0020 ± 0.0005
TU	WIRE	64	64	0.0028 ± 0.0018

$$f(\mathbf{x}) = -\Delta u_{\text{gt}}(\mathbf{x}) + \alpha \left(u_{\text{gt}}(\mathbf{x})^3 - u_{\text{gt}}(\mathbf{x}) \right) . \tag{17}$$

This benchmark includes a nonlinear residual and is useful for testing the stability of physics-informed optimization when the residual includes higher-order terms in u. The visualization is shown in Figure 9, and the corresponding L_2 errors for solving this PDE are given in Table 4.

Table 4: L_2 errors for the top performing combinations for each tensor decomposition mode on the Allen-Cahn PDE benchmark. We observe that the optimal parameter combinations with rank = NC = 64 achieve the best performance, and the WIRE backing is found to be more accurate and robust across all modes.

Mode	Backend	Rank	NC	L_2 Error
CP	Tanh + PE	32	64	0.0052 ± 0.0006
CP	Tanh + PE	64	64	0.0046 ± 0.0015
CP	WIRE	64	64	0.0046 ± 0.0002
TT	Tanh + PE	32	64	0.0083 ± 0.0033
TT	Tanh + PE	64	64	0.0070 ± 0.0026
TT	WIRE	64	64	0.0056 ± 0.0005
TU	Tanh + PE	32	64	0.0067 ± 0.0010
TU	Tanh + PE	64	64	0.0058 ± 0.0011
TU	WIRE	64	64	0.0038 ± 0.0013

B Extension to Parametric PDEs

We now evaluate FTD PINNs on a parametric PDE family. The goal is to show that the method can solve not only a single PDE instance, but also a family of PDEs indexed by a continuous parameter. In this setting, we treat the parameter as an additional input dimension and learn a single model that represents $u(\mathbf{x}, \mu)$ over both the spatial and parameter domains.

Parametric Helmholtz Equation We consider a parametric family of Helmholtz equations

$$\Delta u(x, y; \mu) + \lambda u(x, y; \mu) = q(x, y; \mu)\,, \tag{18}$$

$$(x, y) \in [-1, 1]^2,\ \ \mu \in [\mu_{\min}, \mu_{\max}]\,, \tag{19}$$

with homogeneous Dirichlet boundary conditions $u = 0$ on $\partial\Omega$. We define the manufactured solution as

$$u(x, y; \mu) = \sin((1 + \mu)\pi x)\sin(\pi y)\,, \tag{20}$$

which yields the source term

$$q(x, y; \mu) = -\left(((1 + \mu)\pi)^2 + \pi^2 - \lambda\right) u(x, y; \mu)\,. \tag{21}$$

This family produces increasingly oscillatory solutions as μ increases. The visualization for various μ is shown in Figure 10, and the corresponding results of solving this parametric family of PDEs are given in Table 5.

C Practitioner's Guide to FTD-PINNs

This study introduces several interacting hyperparameters. The goal of this section is to summarize practical guidelines that follow consistently from the experiments in the main paper and the supplementary benchmarks.

Table 5: L_2 errors for the top-performing combinations for each tensor decomposition mode on the parametric family of Helmholtz PDE benchmark. The error is calculated on ten test μ held out from the training set to show that the model effectively learns the family of solutions.

Mode	Backend	Rank	NC	L_2 Error
CP	Tanh + PE	32	64	0.0120 ± 0.0075
CP	Tanh + PE	64	64	0.0097 ± 0.0014
CP	WIRE	64	64	0.0088 ± 0.0002
TT	Tanh + PE	32	64	0.0104 ± 0.0066
TT	Tanh + PE	64	64	0.0093 ± 0.0018
TT	WIRE	64	64	0.0086 ± 0.0009
TU	Tanh + PE	32	64	0.0131 ± 0.0091
TU	Tanh + PE	64	64	0.0095 ± 0.0044
TU	WIRE	64	64	0.0086 ± 0.0031

C.1 Backends

WIRE and TanhPE are consistently strong choices. WIRE and SIREN are particularly effective when the solution contains a high-frequency structure. TanhPE is a strong baseline when a simpler activation is preferred, but frequency bias is still required. For new PDEs, we recommend starting with WIRE or TanhPE and using SIREN as a robust alternative.

C.2 Rank and Collocation Points

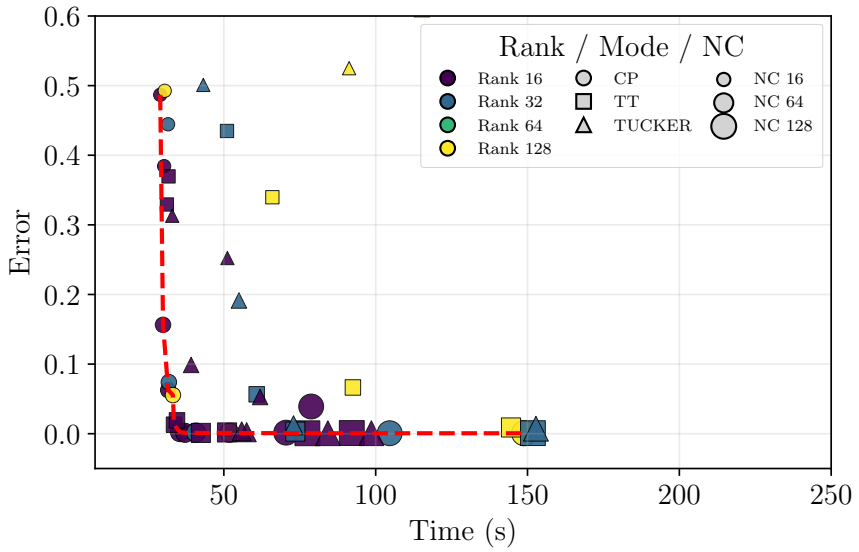
Increasing rank and collocation points improves accuracy, but the improvement saturates. We observe that the best trade-off between runtime and error often occurs at moderate settings. In our experiments, the knee of the Pareto curve typically occurs at ranks in $\{16, 32\}$ with NC around 64. These settings are often sufficient to obtain low errors while avoiding unnecessary computation.

C.3 Decomposition Modes

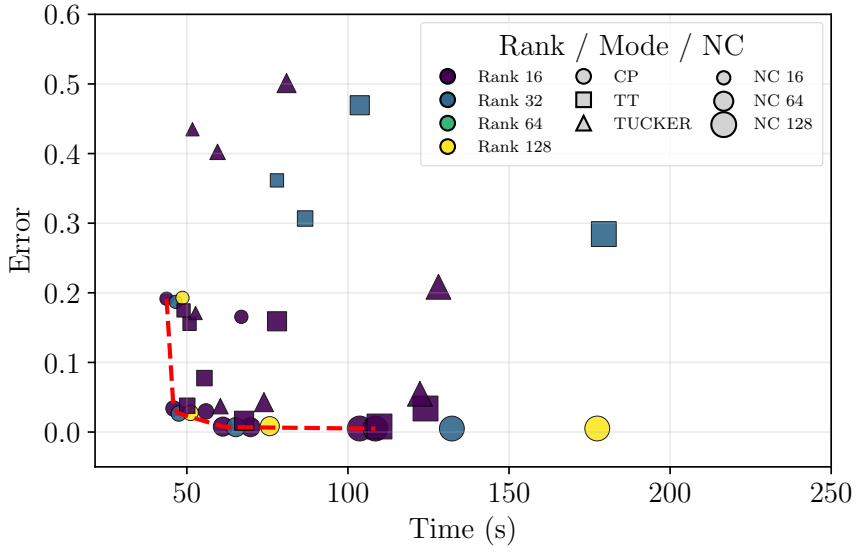
CP and TT frequently dominate the Pareto knee region. Tucker can achieve strong accuracy at higher ranks, but it tends to be more expensive due to its core tensor. For practitioners, CP and TT are good default choices when computing is limited. Tucker is a reasonable option when additional compute is acceptable.

C.4 Parametric PDEs

For parametric PDEs, we recommend treating the parameter as an additional input axis and using the same decomposition structure across space and parameter. This enables a single model to represent a family of PDE solutions. In practice, it is helpful to evaluate generalization by training on a subset of parameter values and testing on unseen values, while keeping the spatial domain unchanged.

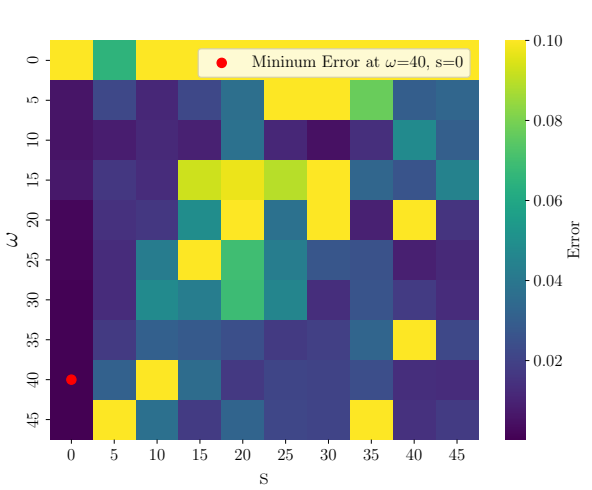


(a) The Pareto curve for *WIRE* backend.

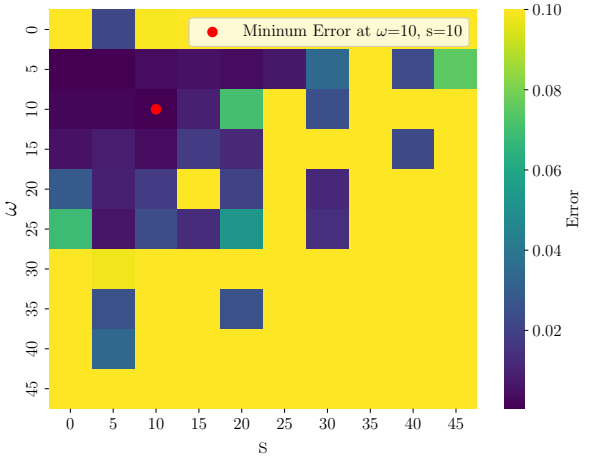


(b) The Pareto curve for *Tanh* backend.

Figure 5: The Pareto curve (dashed line) connects non-dominated configurations in the error-runtime space, showing the optimal trade-off boundary; most points lie close to the curve, indicating efficient model behavior. The WIRE backend has more points on/near the Pareto curve than the Tanh backend, indicating that the WIRE backend outperforms the Tanh.

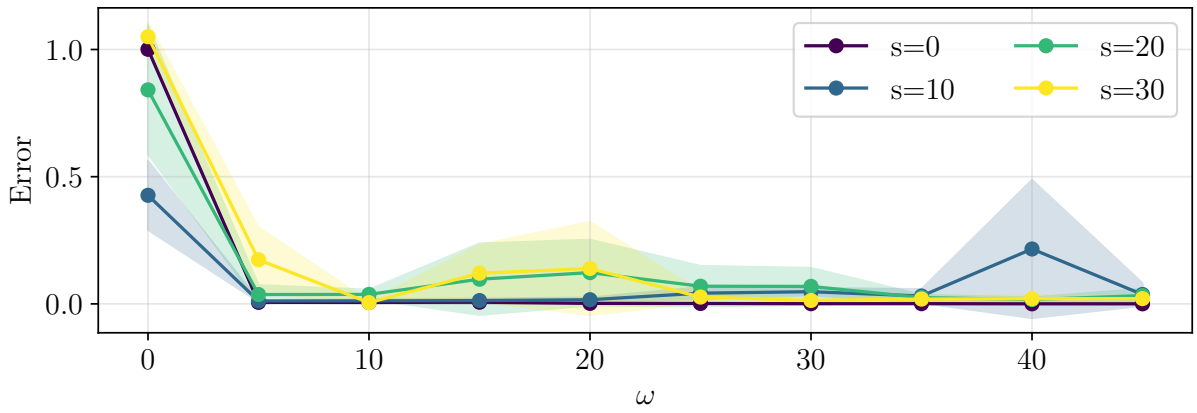


(a) Heatmap of L_2 errors of solving the Helmholtz equation using CP-PINN with wavelet activation for varied scale (s) and frequency (ω) parameters.

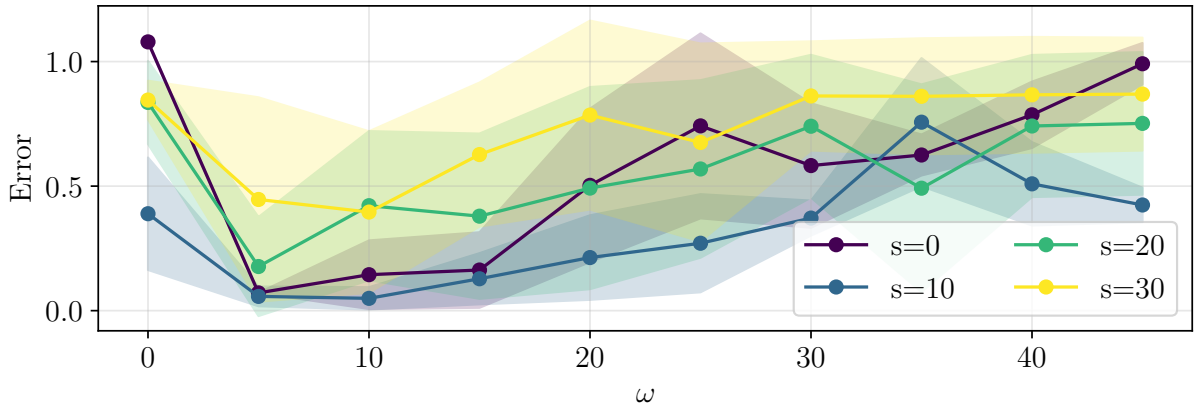


(b) Heatmap of L_2 errors of solving the Klein-Gordon equation using CP-PINN with wavelet activation for varied scale (s) and frequency (ω) parameters.

Figure 6: Results for varying the wavelet parameters frequency (ω) and scale (s) for benchmark PDEs for a fixed architecture.



(a) Ablation on ω and s for Helmholtz PDE.



(b) Ablation on ω and s for Klein-Gordon PDE.

Figure 7: Ablation study on WIRE hyperparameters: Frequency (ω) and scale s for Helmholtz (a) and Klein-Gordon (b) PDEs. The trend of solutions is different for both PDEs, suggesting that the shape of the solution has an effect on the choice of ω and s.

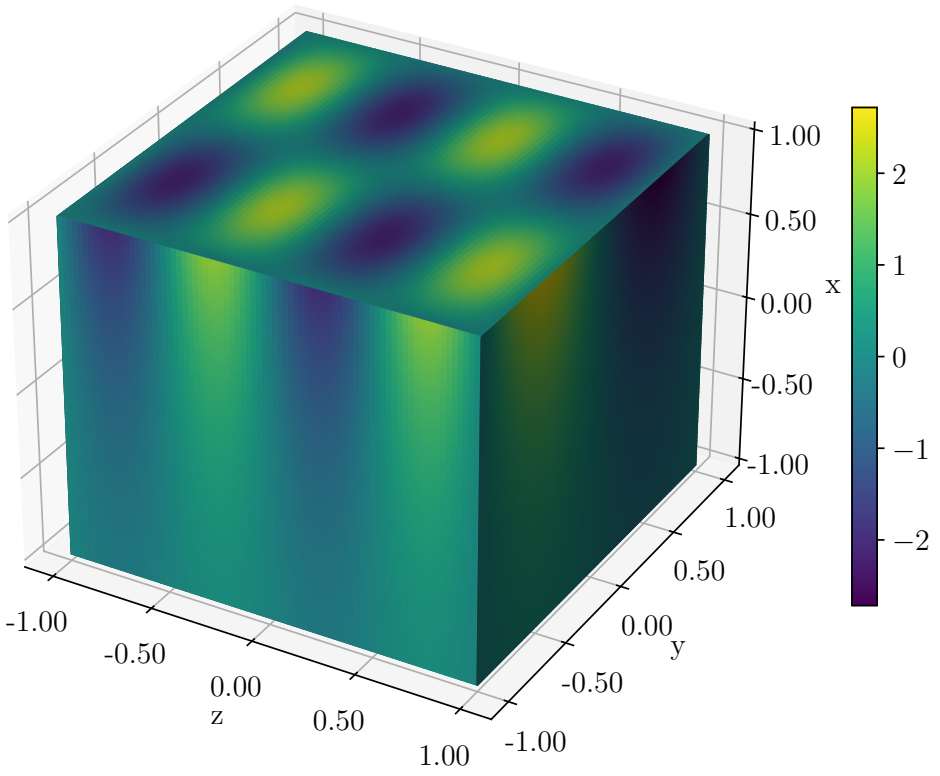


Figure 8: Visualization of three-dimensional Advection-Diffusion Equation, as given in Equation 11. The direction of diffusion is x

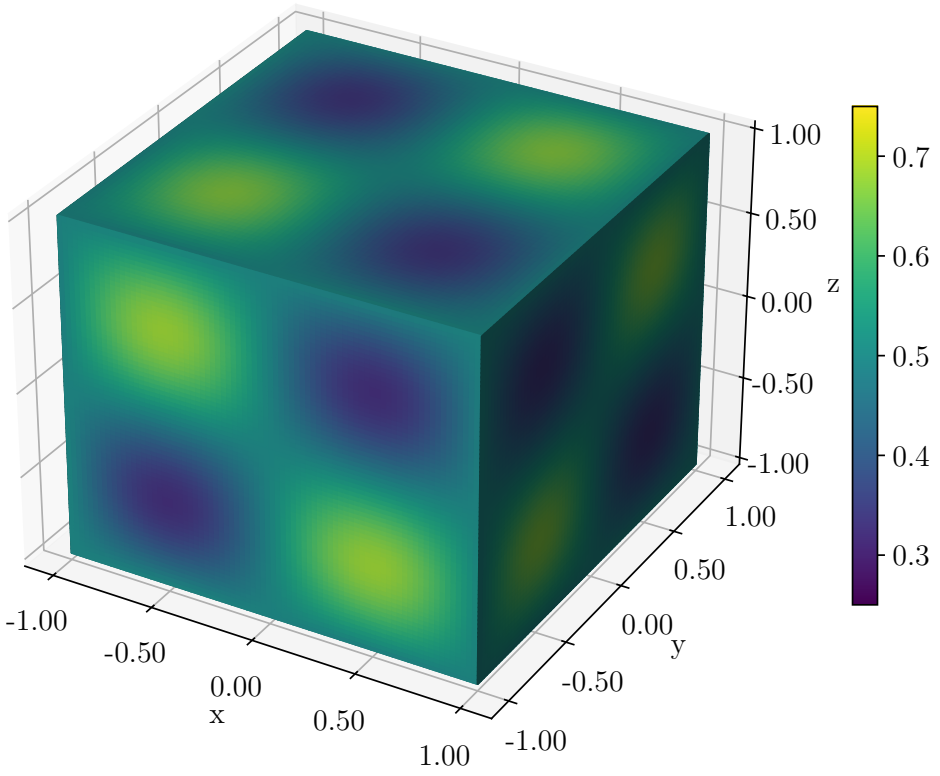


Figure 9: Visualization of 3D Allen-Cahn type equation, as in Equation 14.

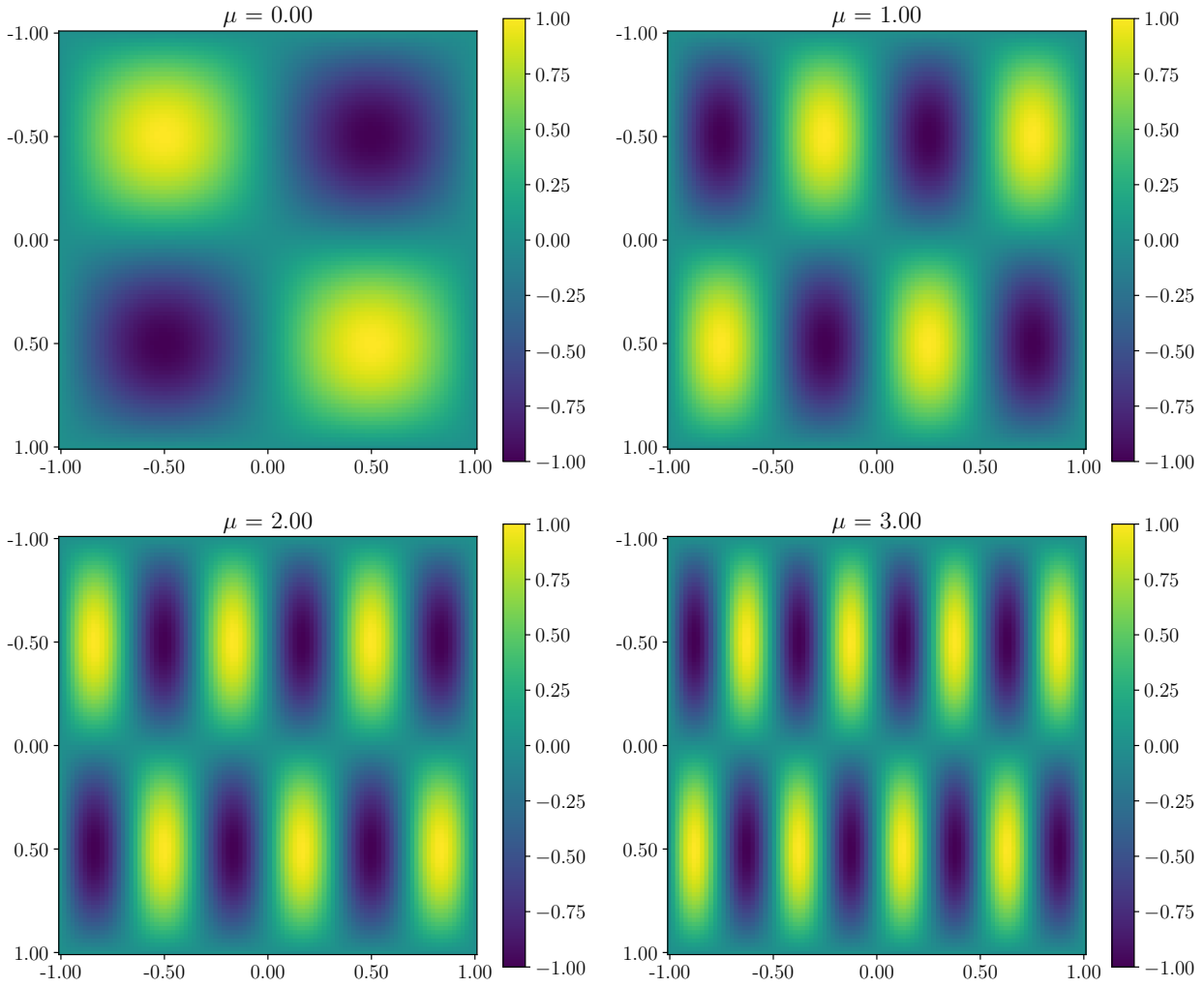


Figure 10: Visualization of parametric Helmholtz family of PDEs at various μ, as given in Equation 19. As μ increases, the frequency of oscillations increases.