PHYSICS-INFORMED CONDITIONAL DIFFUSION FOR MULTI-MODAL PDES

Anonymous authors
Paper under double-blind review

ABSTRACT

Many physical systems that are represented by partial differential equations (PDEs) admit multiple valid solutions, such as eigenstates of differential operators, or wave modes, yet most neural PDE surrogates are deterministic and collapse to averages. This multiplicity of solutions is especially predominant in various engineering and scientific domains ranging from acoustics and seismology to quantum systems. With the ability to generate or complete sparse measurements, diffusion-based approaches to solve PDEs by sampling physically valid solutions are gaining traction as an alternative to traditional numerical solvers. In this paper, we present a novel physics-informed conditional diffusion framework for multi-modal PDEs, called PDEDIFF, that learns distributions over solution fields from sparse, irregular samples while enforcing governing equations and boundary conditions through mesh-free residual penalties computed by automatic differentiation. PDEDIFF is capable of effectively solving PDEs with multiple valid solutions by learning $\mathbb{P}[Y|X]$, i.e., it learns a solution field Y for a corresponding input spatial information X. Unlike Physics-Informed Neural Networks (PINNs), which minimize residuals around expected values $\mathbb{E}[Y|X]$ and hence tend to regress toward a conditional mean, PDEDIFF samples diverse physically consistent solutions by integrating PDE residuals directly into the diffusion objective. Our results indicate that generative, physics-informed diffusion is a practical tool for uncertainty-aware and multi-modal PDE modeling in low-to-moderate dimensions.

1 INTRODUCTION

Accurate modeling and solving of partial differential equations (PDEs) is fundamental to advancing scientific disciplines, areas ranging from acoustics and seismology to quantum systems and fluid dynamics. Physics-Informed Neural Networks (PINNs) Raissi et al. (2019a) have emerged as a powerful approach for embedding known physical laws into machine learning models. In recent years, PINNs have excelled in incorporating domain-specific equations and information into the learning process by adding residual terms, which include differential equation and boundary conditions, to a regressor's loss, allowing data-efficient training from limited observations. Despite their ability to leverage domain knowledge and efficiently estimate conditional means ($\mathbb{E}[Y|X]$), PINNs collapse multi-modal target distributions to a single mean-field solution and hence blur distinct physically admissible solutions or eigenstates and fail to capture these multiple scenarios.

Diffusion models, driven by advances in deep generative modeling, have achieved remarkable performance in tasks requiring detailed sampling from complex probability distributions such as in Ho et al. (2020b); Song et al. (2020). In this paper, we propose using this capability of reversing a noise process to sample from solution fields.

Diffusion models have further advanced into multiple domains such as image synthesisRombach et al. (2022); Kawar et al. (2022); Kakinuma et al. (2025), healthcareCao et al. (2024); Chung & Ye (2022), and drug discoveryAlakhdar et al. (2024); Corso et al. (2023). Conditional variants can, in principle, capture the distribution over solution fields $\mathbb{P}[Y|X]$ rather than a single point estimate, offering critical advantages in problems where multiple solutions naturally arise such as parameterized PDEs or in eigen-PDEs. However, standard diffusion models typically do not incorporate physical constraints explicitly, resulting in samples that may violate conservation laws, boundary conditions, or any other operator constraints.

In consideration of the above problems with diffusion models, there are some recent works, like Bastek et al. (2025) and Shu et al. (2023), that include physics residual as penalties during training. Yet they targeted cases where the PDE's solution pairs can be sampled from a single mode distribution. Also, the above methods retained a grid-based input for training and generation that only support computation residuals using finite-difference stencils.

This ability to sample multiple modes is particularly advantageous in solving PDEs with multiple solutions, such as Schrodinger equation in quantum mechanics with multiple eigenfunctions Lundeen et al. (2011) or multiple wave solutions in Helmholtz equation. Another important setting where multiple solutions arise is in PDEs with unknown parameters or boundary/initial conditions. These unknown parameters can either be missing from data recorded during experiments or often be expensive to measure or reconstruct, thus requiring that solutions from multiple parameter settings be identified that fit the observed measurements. For example, an experimental dataset may contain records of acoustic waves using multiple sensors in a room, however the source locations (initial conditions) or room configuration (boundary conditions) for each experiment may not be fully recorded. In this case, solving the Helmholtz equation to sample multiple solutions that conform to the observed sensor readings, while accounting for the physics, would be of interest. In contrast, standard approaches such as finite difference solvers or even PINNs may either guess a few parameter values and present solutions for them, or learn a single mean estimated parameter value across all experiments.

Motivated by these challenges, we introduce PDEDIFF, a mesh-free, physics-informed conditional diffusion framework that learns distributions over solution fields from sparse, irregular samples while steering generation toward physically valid solutions. The model conditions on coordinates and uses automatic differentiation to evaluate PDE residuals and boundary or initial conditions directly at sampled points, therefore no fixed mesh or finite-difference stencils are required. PDEDIFF's interface is flexible and compatible with different PDEs and we only replace the residual and boundary conditions. Therefore, this makes PDEDIFF a complement to classical solvers as these methods are mesh-specific and equation-specific, whereas our method can be trained once and then work as a generative surrogate that can produce physics aware solutions at new coordinates without the need for re-meshing.

The key contributions of our work include the following:

- Generative solver for multi-modal PDEs: PDEDIFF learns distributions over solution fields and samples distinct, physically admissible modes instead of regressing to an average.
- Physics-conditioned sampler: A coordinate-conditioned encoder–decoder (CCED) denoiser that embeds both spatial coordinates and diffusion time, trained with a dual loss on data fidelity and residual physics.
- Mesh-free residual enforcement: Higher-order derivatives are computed via autograd, enabling irregular or sparse point clouds rather than fixed grids as used in previous work on physics-informed diffusion models.
- Comprehensive evaluations: We benchmark our method on Infinite Potential Wells, Gross-Pitaevskii Equation, and Helmholtz Equation, reporting the Wasserstein-1 distance and modality recall against analytical ground truth. We have also experimented on simpler toy problems with circles similar to the ones in Bastek et al. (2025). Our results indicate that PDEDIFF outperforms PINNs on PDEs with multiple solutions.

2 Related Work

2.1 Diffusion Models

With the development of diffusion models Song et al. (2020),Ho et al. (2020b), and Song et al. (2021), their applications for solving PDEs are emerging. Most of them involve a conditioned diffusion using score-based method to guide the solution based on sparse observations. Huang et al. (2024) proposed training procedure and sampling strategies that are conditioned on incoming observations to recover accurate PDEs trajectories based on the score-based approach Song et al. (2021). Qu et al. (2024) consider the inverse problem of data assimilation where they aim to recover the complete weather state from observations of various modalities. Rühling Cachay et al. (2023) proposed a fully data-driven framework using temporal interpolation as the forward process and forecasting as the reverse diffusion process for spatial-temporal forecasting problems. However,

relying solely on data-driven approaches may fail to capture the underlying physical laws, potentially leading to samples that lack physical consistency or generalizability beyond the training data.

2.2 PHYSICS-INFORMED MODELS

PINNs Raissi et al. (2017) have appeared as a robust framework for solving both forward and inverse problems governed by PDEs. The key idea behind PINNs is to incorporate the PDE directly into the loss function of a neural network. This is achieved by penalizing the network's predictions when they deviate from the physical laws represented by the PDE. As a result, PINNs can learn solutions even from sparse or noisy training data while ensuring that the learned function respects the underlying physics. The line of work by Jin et al. (2022) uses feed-forward neural network to learn the multiple eigenfunctions for the family of Schrödinger's equations. They achieved multiple solutions through penalizing with several physics-informed regularizers and iteratively training to learn new eigenfunctions. However, this approach is only applicable to eigenproblems and would not be generalized to other types of PDE.

Towards physics informed diffusion-based approaches, CocoGen Jacobsen et al. (2024) injected the governing equations into the sampling process of score-based generative models to enforce the consistency of the samples with the underlying PDE, but physics guidance was only applied to the last N steps. They also focus on the reconstruction problem that conditioned on grids of sparse measurement. Shysheya et al. (2024) focused on forecasting tasks with conditioning training and sampling via the score-based approach. Recent works by Bastek et al. (2025) and Shu et al. (2023) use the Denoising Diffusion Probabilistic Model (DDPM) architecture combined with physics-informed loss during training. The physics-informed guidance in Shu et al. (2023) is applied with some probability p_t (a hyperparameter) as an additional input to the UNET architecture in DDPM. Bastek et al. (2025) models the physics term as a virtual likelihood and combines this with the diffusion training loss. Above all, these works employed the finite difference method to compute the physics-informed guidance. This limited the generalization of the model as we need to construct the finite difference scheme for every new PDE, which can become complicated and unstable for high-order derivatives. Also, computing a finite difference required a fixed grid for consistency. Our work makes use of the automatic differentiation technique to circumvent these issues.

3 BACKGROUND

3.1 PARTIAL DIFFERENTIAL EQUATION CONSTRAINTS

We consider the following general form for a system of PDEs defined on some domain Ω

$$\begin{cases} \mathcal{F}(u(x)) = f(x), & x \in \Omega \\ \mathcal{B}(u(x)) = b(x), & x \in \partial\Omega \end{cases} \tag{1}$$

where $\mathcal{F}$ is the interior differential operator for instance, Laplacian or Hamiltonian, $\mathcal{B}$ encodes the boundary condition, $\partial\Omega$ is the boundary of domain Ω, and u is the solution field that satisfies the set of PDEs for all $x \in \Omega$ and boundary conditions $x \in \partial\Omega$. f and b are functions independent of u, usually representing external conditions applied to the system. In physics-informed learning setup, along with the data loss, which is the l_2 loss between the generated samples and ground truth samples, we also aim to minimize a physics residual R. This physics residual is calculated by taking l_2 difference measuring the extent to which boundary and interior physics constraints are satisfied at model prediction $\hat{u}$ as given in Raissi et al. (2019b).

$$R(\hat{u}(x), x) = \lambda_{int}\mathcal{L}_\Omega + \lambda_{bc}\mathcal{L}_{bc} \tag{2}$$

$$= \lambda_{int}||\mathcal{F}(\hat{u}(x)) - f(x)||_2 + \lambda_{bc}||\mathcal{B}(\hat{u}(x)) - b(x)||_2 \tag{3}$$

here $\hat{u}(x)$ is the solution predicted by a deep learning model, and $\mathcal{L}_{bc}$ is the physics residual of the boundary conditions and $\mathcal{L}_\Omega$ is the physics residual of PDE constraints defined in the interior of the domain Ω. λ_{int} and λ_{bc} are the weights that we want to consider for the respective residuals. For problems dealing with multiple solutions either through eigen-states or parameterized PDEs, we will be training the generative model to generate both the solution field and eigenvalues/parameters for the differential operators; we revisit this in detail in Section 4.1.

Usually when closed form of a differential operator does not exist, researchers generally opt for a finite difference approach to calculate the derivatives, but this method becomes very unstable for higher order derivatives and that is why we will be using automatic differentiation to eliminate this uncertainty in gradient calculation and making it more flexible than using a grid-based approach.

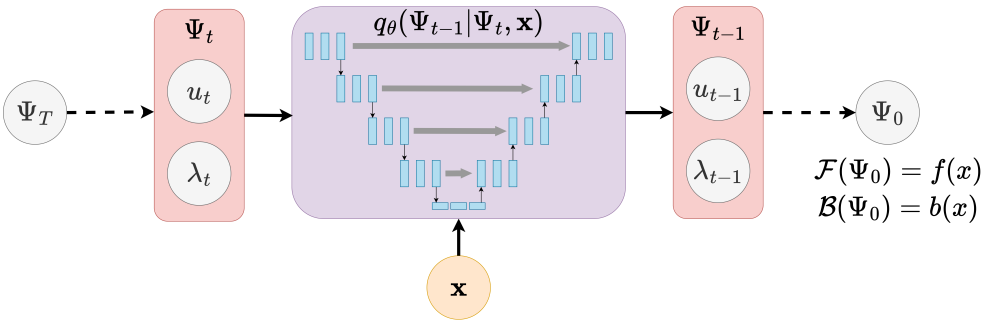


Figure 1: PDEDIFF denoising step denoises random noise into solution field and eigenvalues by conditioning on the spatial coordinates $\mathbf{x}$. The goal of this coordinate conditioned encoder decoder (CCED) architecture is to generate samples that follow the residual $\mathcal{F}$ and boundary conditions $\mathcal{B}$

3.2 Denoising Diffusion Probabilistic Model

Diffusion models are a class of deep generative models inspired by non-equilibrium thermodynamics. Ho et al. (2020b) models the forward process by iteratively adding noise to the data so that it resembles a random distribution. Formally, Eq. 4 defines the forward diffusion process, here u_0 is the true data and noise sampled from a gaussian distribution is added over time steps $t \in 1, \ldots, T$:

$$p(u_t|u_{t-1}) = \mathcal{N}(u_t; \sqrt{1-\beta_t}u_{t-1}, \beta_t I), \tag{4}$$

here $\{\beta_t\}_{t=1}^T$ controls the variance schedule in the above equation and after large enough T steps, u_T will resemble a sample from a gaussian distribution. The objective of a diffusion model is to learn a reverse (denoising) process that denoises a sample generated from $\mathcal{N}(0, I)$ into samples from the data distribution p_{data}. Mathematically, the model and the reverse process parameterized by model parameters θ is defined as:

$$q_\theta(u_{t-1}|u_t) = \mathcal{N}(u_{t-1}; \mu_\theta(u_t, t), \Sigma_\theta(u_t, t)), \tag{5}$$

This allows the generation of samples from complex data distributions. In the case of conditional diffusion models, additional information (such as class labels or prompts) is used to condition the denoising network. The model then approximates the full conditional density and generates outputs sampled from the distribution of a desired output class or prompt.

4 Method

In this section, we will dive deep into PDEDIFF, a novel conditioned diffusion model framework that learns a distribution corresponding to solution field and parameters in a PDE agnostic setting. We will begin with formalizing the learning task (Sec. 4.1), and derive the physics-informed conditional diffusion algorithm (Sec. 4.1). We will then describe the architecture and the training objective (Sec. 4.2 and Sec. 4.3) and then introduce the metrics used to benchmark the results (Sec. 4.4).

4.1 Problem Formulation

Similar to a standard data-driven approach, we will provide the independent variables as input to the diffusion model. The forward and reverse process of the diffusion model then outputs our dependent variables. For our problem setting, we will be working with different PDEs such as Time Independent Schrödinger's Equation (TISE), and Helmholtz equation. The input for each of the corresponding PDEs will be the spatial coordinates x and the output is the corresponding solution field and the parameter that satisfy the PDE. In this paper, we will be using $u(x)$ to denote the solution field or eigenfunction and λ will be used to denote the parameter of the differential operator or the eigenvalue. Both these terms will be used interchangeably as we discuss different PDEs.

Our method is designed to learn distributions over feasible solutions fields and eigenvalues of the PDE directly from data. For given spatial coordinates $x \in \Omega \subset \mathbb{R}^d$, the diffusion model is conditioned on x to generate the dependent variables $u(x)$ and λ. The goal of the model is to learn to generate samples that follow the below PDE:

$$\mathcal{H}_\lambda u(x) = g(x) \tag{6}$$

here $\mathcal{H}_\lambda$ is the differential operator (usually of second-order or higher order) dependent on parameter λ, and $g : \Omega \to \mathbb{R}$ is a known function, and $u : \Omega \to \mathbb{R}$ is the solution field. For simplicity, we only consider PDE with Dirichlet boundary condition, that is, $\mathcal{B}(u(x)) = 0$. But the method can be expand to more general setting of any multi-solution PDEs with non-zero boundary condition.

Given a distribution $p(x)$ where $x \in \Omega$, we are interested in learning the multi-modal conditional density function $p(u(x), \lambda | \, x)$ from the sparse dataset $\{(x^{(n)}, u^{(n)}, \lambda^{(n)})\}_{n=1}^{N}$, whose samples follow the PDEs. For simplicity, we will use $\Psi = (u, \lambda)$ to denote a data point, and $p(\Psi|x)$ to denote the probability distribution. The forward diffusion process is simulated by gradually adding controlled Gaussian noise to Ψ, which can be model as a conditional distribution $p(\Psi_{t+1}(x)|\Psi_t(x), x) \sim \mathcal{N}(\sqrt{1-\beta_t}\Psi_t(x), \beta_t I)$, where $\{\beta_t\}_{t=1}^{T}$ is a sequence of noise scheduler.

$$\underbrace{\Psi_T|x}_{\text{Gaussian noise}} \leftarrow \Psi_{T-1}|x \leftarrow \cdots \leftarrow \Psi_t|x \leftarrow \cdots \leftarrow \Psi_1|x \leftarrow \underbrace{\Psi_0|x}_{\text{real data}}$$

This process eventually yields a structured latent representation at timestep T. We adapt the method from Ho et al. (2020b); Song et al. (2020) to derive a simplified loss function for the denoising process. Note that the reverse distribution $q(\Psi_t(x)|\Psi_{t+1}(x), x)$ is intractable, we therefore use a neural network $\text{NN}(\Psi_t, x, t)$ to model the inverse diffusion steps $q_\theta(\Psi_{t-1}|\Psi_t, \Psi_0, x)$ that maximize the log-likelihood of $q(\Psi_0|x)$:

$$\log q(\Psi_0|x) = \log \int q(\Psi_{0:T}|x) d\Psi_{1:T} \tag{7}$$

This log-likelihood is not tractable due to unknown denoising process. Therefore, we will approximate it with evidence lower bound, which is easier to optimize:

$$\log q(\Psi_0|x) \geq \mathbb{E}_{p(\Psi_{1:T}|\Psi_0,x)} \left[\log \frac{q(\Psi_{0:T}|x)}{p(\Psi_{1:T}|\Psi_0, x)} \right] \tag{8}$$

Simplifying the evidence lower bound (further details in the Appendix A.1) gives us

$$\begin{aligned} \log q(\Psi_0|x) \geq \mathbb{E}_{p(\Psi_1|\Psi_0,x)} \left[\log \frac{q_0(\Psi_0|\Psi_1, x)}{p(\Psi_1|\Psi_0, x)} \right] \\ + \sum_{t=2}^{T} \underbrace{\mathbb{E}_{p(\Psi_t|\Psi_0,x)} \left[D_{KL}(p(\Psi_{t-1}|\Psi_t, \Psi_0, x) || q_\theta(\Psi_{t-1}|\Psi_t, x)) \right]}_{=\mathcal{L}_{\text{data},t}} \end{aligned} \tag{9}$$

In addition to maximizing the likelihood of $q(\Psi_0|x)$, we are also interested in incorporating the physics guidance in training the diffusion model. Our goal is also to minimize the likelihood of the residual $q(R(\Psi_0, x)|\Psi_0, x) \sim \mathcal{N}(R(\Psi_0, x), \sigma^2)$, where the variance σ^2 approaches 0 as the model learns Ψ_0. In this paper, we consider PDEs of the form $\mathcal{H}_\lambda u(x) = g(x)$. Below is how this *PDE-loss* or $\mathcal{L}_{\text{residual}}$ is defined:

$$\log q(R(\Psi_0, x)|\Psi_0, x) \propto \|\mathcal{H}_\lambda \hat{u}_0 - g\|_2^2 =: \mathcal{L}_{\text{residual}}(\hat{\Psi}_0) \tag{10}$$

4.2 Training and Sampling algorithms

We follow a similar training objective and approach as given in Ho et al. (2020a); Bastek et al. (2025), where we add a physics informed loss as a regularizer to the entire loss function. The training algorithm in Bastek et al. (2025) uses a finite difference stencil method to calculate the derivatives of the quantities required in the residual. We instead leverage the power of automatic differentiation (autodiff) to calculate the gradients for the residual equations. This strategy provides more flexibility to calculate higher order derivatives as compared to using finite difference stencils, which become unstable when calculating higher order derivatives. Moreover, the finite difference method can only feasible where the data points are arranged in a regular grid and all the points on the grid correspond to only a single solution, whereas PDEDIFF can train on data points that are randomly sampled from a multi-modal distribution and each generated sample can correspond to any of the feasible solutions.

We will now introduce the objective function of PDEDIFF that is a loss function composed of the standard reconstruction error and physics informed constraints. The loss function is defined as:

$$\mathcal{L}_{\text{total},t}(\hat{\Psi}_0) = \mathbb{E}_{t\sim[1:T], p(\Psi_{1:T}|x)} \left[c_{\text{data}} \mathcal{L}_{\text{data},t}(\Psi_0, \hat{\Psi}_0) + c_{\text{res}} \mathcal{L}_{\text{residual},t}(\hat{\Psi}_0) \right] \tag{11}$$

Algorithm 1 PDEDIFF Training

Input: Training dataset $\Psi_0 = (u_0(x), \lambda_0)$
Output: Trained denoising process CCED_θ
repeat
  $x \sim \text{Uniform}(\Omega)$
  $\Psi_0(x) \sim q(\Psi_0|x)$ (from training data)
  $t \sim \text{Uniform}\{1, \ldots, T\}$
  $\epsilon \sim \mathcal{N}(0, I)$
  $\bar{\alpha}_t = \prod_{s=1}^{t}(1 - \beta_t)$
  $\Psi_t(x) = \sqrt{\bar{\alpha}_t}\Psi_0(x) + \sqrt{1 - \bar{\alpha}_t}\,\epsilon$
  $\hat{\Psi}_0 = \text{CCED}_\theta(\Psi_t(x), x, t)$
  Compute $\nabla_\theta \mathcal{L}_{\text{total},t}(\hat{\Psi}_0)$ (as in 11)
  Update CCED_θ via GD
until converged

Algorithm 2 PDEDIFF Sampling

Input: Number of samples N, trained denoising process CCED_θ
Output: Set of samples $\{\Psi_0\}_{i=1}^{N}$
for $i = 1$ to N **do**
  Sample $x_i \sim \text{Uniform}(\Omega)$
  for $t = 1$ to T **do**
    $\hat{\Psi}_0(x_i) = \text{CCED}_\theta(\Psi_t(x_i), x_i, t)$
    Sample $\Psi_0(x_i)$ with DDIM Song et al. (2020)
  end for
end for

where we use it to train a denoising process as in Algorithm 1.

Here $\mathcal{L}_{\text{data}}$ is called the data loss which is the reconstruction error between the predicted output and the ground truth values, and for our framework we have used a standard Mean Square Error (MSE) loss. $\mathcal{L}_{\text{residual}}$ is the residual loss, which penalizes the model when the generated samples do not satisfy the underlying physics constraints, i.e., the solutions generated must follow the PDEs.

c_{data} and c_{res} are hyperparameters that we set during training to weight the importance of the data loss and physics loss, respectively. Since, we provide the model with the initial spatial and temporal information, we can leverage the use of autodiff to calculate high order derivatives for calculating the residual. This approach also allows us to use randomly sampled data points instead of going with a grid-based approach providing us with more flexibility with respect to the dataset used and also for the sampling part for inference. For PDEDIFF sampling Algorithm 2, we utilize the sampling method from Denoising Diffusion Implicit Model (DDIM) Song et al. (2020) to accelerate the generation process. Also, in step 4 of Algorithm 2, we could replace uniform samples of x_i with any other grid or mesh coordinates for applicable uses.

4.3 Architecture

PDEDIFF employs a *coordinate conditioned encoder decoder (CCED)* denoiser that converts a noisy input into its corresponding denoised output by conditioning on the spatial coordinates (independent variable) and diffusion time-step. We pass the noisy input along with the spatial coordinates $x \in \Omega \subseteq \mathbb{R}^d$ and the diffusion time t into the denoiser where each layer is called a *Conditional Block*, which consists of a set of linear and embedding layers. Figure 1 demonstrates how the denoising model converts random noise into the corresponding solutions fields and eigenvalues. The details can be found in Appendix A.3.

Further, we use AdamW for training and we also experiment with different weights for the residual loss (c_{res}) (additional results on this is discussed in Appendix A.6). For baselines, we compare our method with a vanilla conditional diffusion model ($c_{\text{res}} = 0$) with no physics regularization, and we also compare PDEDIFF with a physics-informed neural network (PINN) that is fully connected and where we have used the same values of c_{res} for all the comparative studies and ablation results. Since for the training algorithm we do not require to implement a custom finite difference stencil and instead proceed with using autodiff, this makes our framework compatible with mesh-free datasets.

4.4 Quantification metrics

Given some spatial input $\mathbf{x} \in \Omega$, PDEDIFF would generate $\Psi_0 = (u_0, \lambda_0)$ corresponding to this $\mathbf{x}$. The PDEs that it is trying to solve has multiple correct solutions, for e.g., the first 3 wavefunctions and energy values for the Schrödinger's Equation could correspond to the following possible solutions $(u_0^{(1)}, \lambda_0^{(1)}), (u_0^{(2)}, \lambda_0^{(2)}), (u_0^{(3)}, \lambda_0^{(3)})$. Since each of these is a correct solution to the PDE, we need a metric that can effectively compare the generated samples with the true distribution or the ground truth solutions.

P-MSE: This metric calculates the mean squared error of the samples generated with the ground-truth eigenstates or solutions modes. Formally, the metric can be defined as below:

Setting		Vanilla		Physics-informed	
		PMSE	WD	PMSE	WD
1D (2 states)	P	0.822	0.539	1.052	0.344
	Q	0.017	0.086	**0.015**	**0.086**
1D (3 states)	P	0.371	0.737	0.460	0.442
	Q	0.022	0.055	**0.012**	**0.050**
1D GP (3 states)	P	1.550	2.036	1.291	2.418
	Q	0.140	0.195	**0.126**	**0.194**

Table 1: Performance comparison between PINN (P) and PDEDIFF (Q) across different problem settings for 1D examples. PMSE here is P-MSE and WD is the Wasserstein metric. The lower values represent better match between the ground truth and the sampled wavefunctions. Details of the energy states will be provided in Appendix A.5.1. The first 2 rows correspond to 1D Infinite Potential Well, and 1D GP corresponds to 1D Gross-Pitaevskii Equation.

Setting		Vanilla		Physics-informed	
		PMSE	WD	PMSE	WD
2D (3 states)	P	1.253	0.378	1.239	0.340
	Q	1.222	0.169	**1.194**	**0.156**
2D(4 states) degenerate	P	1.250	0.482	**1.243**	0.477
	Q	1.309	0.150	1.338	**0.124**
2D (4 states) non-degenerate	P	1.133	0.597	1.123	0.623
	Q	1.100	0.222	**1.091**	**0.205**
Helmholtz Equation	P	7.392	1.259	9.380	1.229
	Q	1.511	**0.958**	**1.142**	1.069

Table 2: Performance comparison between PINN (P) and PDEDIFF (Q) across different problem settings. PMSE here is P-MSE and WD is the Wasserstein metric. The lower values represent better match between the ground truth and the sampled wavefunctions. Details of the energy states will be provided in Appendix A.5.1.

$$P\text{-MSE} = \mathbb{E}_{\mathbf{x}\sim\Omega}\left[\left\|\left(\hat{u}_0(\mathbf{x}), \hat{\lambda}_0(\mathbf{x})\right) - \left(u^{(k(\mathbf{x}))}, \lambda^{(k(\mathbf{x}))}\right)\right\|_2^2\right] \tag{12}$$

$$\text{where} \quad k(\mathbf{x}) = \underset{i\in\{1,\ldots,M\}}{\arg\min} ||\hat{\lambda}_0(\mathbf{x}) - \lambda^{(i)}||_2 \tag{13}$$

Here, M is the number of feasible solution modes, $(\hat{u}_0(\mathbf{x}), \hat{\lambda}_0(\mathbf{x}))$ is the generated sample for a particular $\mathbf{x}$, and $k(\mathbf{x})$ is the function that assigns a particular sample to its closest solution field by comparing the parameter values and this state is then used to calculate the MSE.

Earth-Mover's Distance (Wasserstein-1): For two probability measures p, q on a domain $\Omega \subset \mathbb{R}^d$ we use the 1-Wasserstein distance (WD), here $\Gamma(p, q)$ denotes the set of couplings with marginals p and q

$$W_1(p, q) = \inf_{\gamma\in\Gamma(p,q)} \int_{\Omega\times\Omega} \| \mathbf{x} - \mathbf{y}\| \, \mathrm{d}\gamma(\mathbf{x}, \mathbf{y}), \tag{14}$$

This metric is used to compare the distances between two probability distributions and in some PDEs, wavefunctions have a special property, i.e., it also represents the probability densities of a particle, i.e., $|\psi|^2$ represents the probability density function of the particle. The ground truth distribution is represented by a linear combination of the probability distribution of the individual solution modes, and the generated distribution can be calculated by squaring the predicted $\psi(\mathbf{x})$.

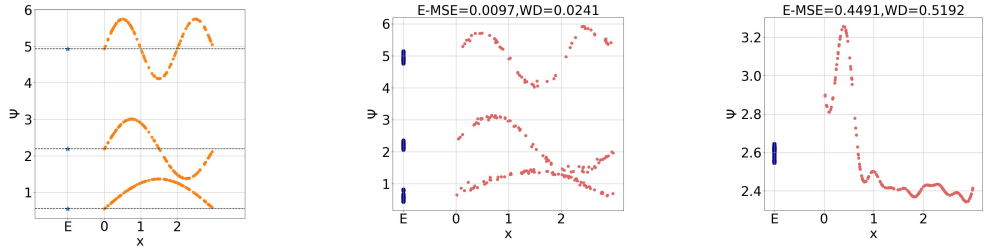
To compare the predicted probability distribution and the true distribution, we evaluate both the functions on the same set of point cloud $\mathbf{x}$. Then the distributions are normalized as below:

$$p_{\text{pred}}(\mathbf{x}) = \frac{|\psi_{\text{pred}}(\mathbf{x})|^2}{\sum_{\mathbf{x}'\in\mathcal{X}} |\psi_{\text{pred}}(\mathbf{x}')|^2} \qquad p_{\text{true}}(\mathbf{x}) = \sum_{i=1}^{N} \pi_i \frac{|\psi^{(i)}(\mathbf{x})|^2}{\sum_{\mathbf{x}'\in\mathcal{X}} |\psi^{(i)}(\mathbf{x}')|^2} \tag{15}$$

where $\sum_{i=1}^{n} \pi_i = 1$ (linear combination coefficients) and after calculating the probability distributions, we use the optimal transport library Flamary et al. (2021).

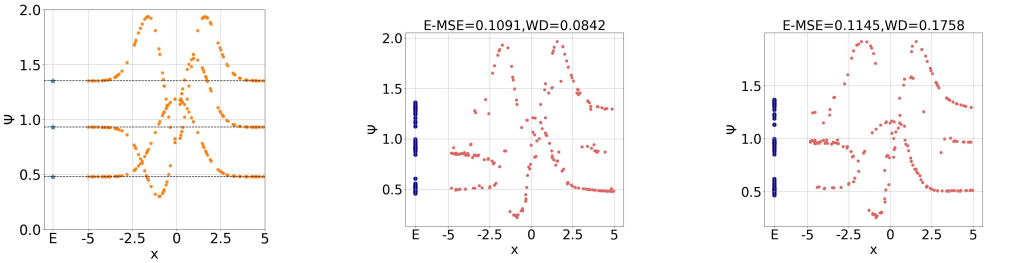
5 Experiment Results

In this section, we present our experiment setup and results for several PDE problems with multiple solutions such as Time Independent Schrödinger Equation in 1D and 2D infinite potential well, Helmholtz Equation, Gross–Pitaevskii Equation, and on toy examples involving circles



(a) Ground Truth (b) PDEDIFF (Physics-informed) (c) PINN

Figure 2: Comparing generated samples by PDEDIFF and PINN for 1D Schrödinger equation



(a) Ground Truth (b) PDEDIFF (Physics-informed) (c) PDEDIFF (no physics)

Figure 3: Comparing generated samples by PDEDIFF for 1D Gross-Pitaevskii equation

5.1 1D Schrödinger: Particle in a Box

We consider the time-independent Schrödinger equation for a quantum particle in a one-dimensional infinite potential well:

$$\underbrace{\left(-\frac{\hbar^2}{2m}\partial_x^2 + V(x) - E\right)}_{\mathcal{H}_E}\psi(x) = 0, \quad V(x) = \begin{cases} 0, & x_L < x < x_R \\ \infty, & \text{otherwise} \end{cases}$$

where $V(x)$ denotes the infinite potential well, or in other words, the space of a 1D box that contains the particle where it moves freely inside but trapped between the wall (x_L, x_R) defined as above. Note that ∂_x^2 are the second-order derivatives of ψ with respect to x and could be calculate using automatic differentiation to compute the physics informed loss during training. Given this PDE, we are trying to recover the solution field $\psi(x)$ and corresponding energy or parameter E. Results are presented in Table 1 and Figure 2.

5.2 1D Schrödinger: Gross-Pitaevskii equation

We consider also one-dimensional time-independent Gross-Pitaevskii equation, a form of non-linear Schrodinger equation. In specific, the Halmiltonian operator $\mathcal{H}$ is non-linear and written as follow:

$$\underbrace{\left(\frac{h^2\partial_x^2}{2m} + V(x) + U_0|\psi(x)|^2 - E\right)}_{\mathcal{H}_E}\psi(x) = 0$$

where U_0 is interaction force, $V(x) = \frac{1}{2}mx^2$ is the harmonic trapping potential. Not many non-linear Schodinger equations have closed-form solution, such as the one presented here. We obtain training samples and ground truth by solving the time-independent Gross-Pitaevskii equation numerically as in Chiofalo et al. (2000). Results are presented in Table 1 and Figure 3.

5.3 2D Schrödinger: Particle in a Box

We again consider the TISE for a quantum particle in a two-dimensional infinite potential well:

$$\underbrace{\left(-\frac{\hbar^2}{2m}(\partial_x^2 + \partial_y^2) + V(\mathbf{x}) - E\right)}_{\mathcal{H}_E}\psi(\mathbf{x}) = 0, \quad V(\mathbf{x}) = \begin{cases} 0, & x_L < x < x_R, \quad y_L < y < y_R \\ \infty, & \text{otherwise} \end{cases}$$

where $\mathbf{x} = (x, y)$ denotes the spatial coordinates and $V(\mathbf{x})$ denotes the 2D infinite potential well. ∂_x^2 and ∂_y^2 are the second-order partial derivatives of ψ with respect to x and y. The particle is confined inside the box $\mathbb{B} = [x_L, x_R] \times [y_L, y_R] \subset \mathbb{R}^2$ with Dirichlet boundary conditions, i.e., $\psi(\mathbf{x}) = 0$ for any $\mathbf{x} \in \partial\mathbb{B}$, where $\partial\mathbb{B}$ denotes the boundary of $\mathbb{B}$. The results are presented in Tab. 2.

5.4 HELMHOLTZ EQUATION

Another case study is the Helmholtz equation that arises from applications in heat conduction or acoustics . The Helmholtz equation is an inhomogeneous elliptic PDE express as follow:

$$\underbrace{\nabla^2\psi(x) + k^2\psi(x)}_{\mathcal{H}_k\psi} = f(x)$$

where integer or real valued parameter k is known as frequency or wave number, f is a source function that describe the emission source of the wave $\psi(x)$, with $x \in \Omega$. Our goal is to recover the high resolution propagation of the wave through the environment $\psi(x)$ and its frequency k. Training data is simulated by solving the second-order finite-difference linear system $\mathcal{H}\psi = f$ with low fidelity. The results are presented in Table 2.

6 DISCUSSION

We observed that PDEDIFF consistently achieves low MSE and WD metric, with physics-informed PDEDIFF achieving the best result in almost all cases (Tab. 1, 2) This demonstrates that PDEDIFF is capable of modeling, generating multimodal distributions and achieves higher numerical accuracy compared to the ground truth. As shown in Fig. 2, PDEDIFF effectively captures the multiple modes present in the training data, while standard feedforward neural networks tend to regress to the mean, failing to represent the distinct modes and instead producing oversmoothed approximations.

In the case of 1D Schrödinger equation, PDEDIFF has lower errors in both MSE and Wasserstein distance. Incorporating physics-informed guidance further improves performance. Nevertheless, Schrödinger equations have trivial solution where $\psi(x) = 0, E = 0$. This causes problems as with our physics guidance, since the physics loss is also minimized (equal to 0) if the model infers the trivial solution, as observed in our ablation study in A.6, 10 and 11. We also demonstrate the capabilities of PDEDIFF on 2D Schrödinger equation. In the problem setting of 3 waves, the model is able to learn the different eigenstates and produce samples that are much closer to the true distribution and have a lower MSE loss. In the case where there are 4 waves, the model was trained on both setting wherestates where there are degenerate eigenstates, i.e., 2 eigenstates had the same energy but different wavefunctions, and non-degenerate eigenstates. PDEDIFF is yet able to distinguish the degenerate energy states and generate samples that have a lower Wasserstein metric.

In inhomogeneous Helmholtz Equation, we tested our framework with 2 parameters. In this setting, we observed that the model pick up the low frequency wave better and it sometimes missed the peak in high frequency wave. Also, WD of the vanilla PDEDIFF is slightly better compare to that of the physics informed PDEDIFF. This possibly due to the fact that Helmholtz solutions are sound wave and not well representable as a probability distribution.

7 CONCLUSION

We introduced **PDEDIFF**, a physics-informed conditional diffusion framework that samples *entire ensembles* of solution fields and eigenvalues for multiple solution PDEs. In contrast, PINNs regress to a conditional mean or collapse to one of the modes. We have also demonstrated that our approach is more flexible than other approaches to incorporating physics in diffusion models in terms of the data samples being mesh-free and also for calculating the physics residuals using automatic differentiation. Also, unlike previous work, we provide the first evidence that physics informed diffusion models can provide better results than physics informed neural networks for PDEs with multiple solutions, and thus are more faithful to the physical law that they must follow. Our work shows a potential towards learning and generating new data for multi-modal PDEs.

Looking ahead, our framework can potentially be used to explore methods to discover unseen solution fields and eigenvalues that are not observed in data. Similar to recent works, such as Jin et al. (2022), extending PDEDIFF to learning solution fields in a data-free environment is an interesting direction. Scaling this framework to noisy and high-dimensional experimental datasets would enable developing real-time digital simulations of problems that could be studied in much detail and also accelerate the growth of drug discoveries and sustainable quantum technologies.

REFERENCES

Amira Alakhdar, Barnabas Poczos, and Newell Washburn. Diffusion models in de novo drug design. *Journal of Chemical Information and Modeling*, 64(19):7238–7256, 2024. doi: 10.1021/acs.jcim.4c01107. URL https://doi.org/10.1021/acs.jcim.4c01107. PMID: 39322943.

Jan-Hendrik Bastek, WaiChing Sun, and Dennis Kochmann. Physics-informed diffusion models. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=tpYeermigp.

Sattik Basu and Sarma L. Rani. Generalized acoustic helmholtz equation and its boundary conditions in a quasi 1-d duct with arbitrary mean properties and mean flow. *Journal of Sound and Vibration*, 512:116377, 2021. ISSN 0022-460X. doi: https://doi.org/10.1016/j.jsv.2021.116377. URL https://www.sciencedirect.com/science/article/pii/S0022460X21004296.

Chentao Cao, Zhuo-Xu Cui, Yue Wang, Shaonan Liu, Taijin Chen, Hairong Zheng, Dong Liang, and Yanjie Zhu. High-frequency space diffusion model for accelerated mri. *IEEE Transactions on Medical Imaging*, 43(5):1853–1865, 2024. doi: 10.1109/TMI.2024.3351702.

M. L. Chiofalo, S. Succi, and M. P. Tosi. Ground state of trapped interacting bose-einstein condensates by an explicit imaginary-time algorithm. *Phys. Rev. E*, 62:7438–7444, Nov 2000. doi: 10.1103/PhysRevE.62.7438. URL https://link.aps.org/doi/10.1103/PhysRevE.62.7438.

Hyungjin Chung and Jong Chul Ye. Score-based diffusion models for accelerated mri. *Medical Image Analysis*, 80:102479, 2022. ISSN 1361-8415. doi: https://doi.org/10.1016/j.media.2022.102479. URL https://www.sciencedirect.com/science/article/pii/S1361841522001268.

Gabriele Corso, Hannes Stärk, Bowen Jing, Regina Barzilay, and Tommi S. Jaakkola. Diffdock: Diffusion steps, twists, and turns for molecular docking. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=kKF8_K-mBbS.

Rémi Flamary, Nicolas Courty, Alexandre Gramfort, Mokhtar Z. Alaya, Aurélie Boisbunon, Stanislas Chambon, Laetitia Chapel, Adrien Corenflos, Kilian Fatras, Nemo Fournier, Léo Gautheron, Nathalie T.H. Gayraud, Hicham Janati, Alain Rakotomamonjy, Ievgen Redko, Antoine Rolet, Antony Schutz, Vivien Seguy, Danica J. Sutherland, Romain Tavenard, Alexander Tong, and Titouan Vayer. Pot: Python optimal transport. *J. Mach. Learn. Res.*, 22(1), January 2021. ISSN 1532-4435.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *CoRR*, abs/2006.11239, 2020a. URL https://arxiv.org/abs/2006.11239.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 6840–6851. Curran Associates, Inc., 2020b. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/4c5bcfec8584af0d967f1ab10179ca4b-Paper.pdf.

Jiahe Huang, Guandao Yang, Zichen Wang, and Jeong Joon Park. Diffusionpde: Generative pde-solving under partial observation, 2024. URL https://arxiv.org/abs/2406.17763.

Christian Jacobsen, Yilin Zhuang, and Karthik Duraisamy. Cocogen: Physically-consistent and conditioned score-based generative models for forward and inverse problems, 2024. URL https://arxiv.org/abs/2312.10527.

Henry Jin, Marios Mattheakis, and Pavlos Protopapas. Physics-informed neural networks for quantum eigenvalue problems. In *2022 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–8, 2022. doi: 10.1109/IJCNN55064.2022.9891944.

Yusuke Kakinuma, Takamichi Miyata, Kaito Hosono, and Hirotsugu Kinoshita. Zero-shot image inpainting using pretrained latent diffusion models. In *2025 International Conference on Artificial Intelligence in Information and Communication (ICAIIC)*, pp. 0803–0807, 2025. doi: 10.1109/ICAIIC64266.2025.10920775.

Bahjat Kawar, Michael Elad, Stefano Ermon, and Jiaming Song. Denoising diffusion restoration models. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, NIPS '22. Curran Associates Inc., 2022. ISBN 9781713871088.

Jeff S. Lundeen, Brandon Sutherland, Aabid Patel, Corey Stewart, and Charles Bamber. Direct measurement of the quantum wavefunction. *Nature*, 474(7350):188–191, 2011. doi: 10.1038/nature10120.

Yongquan Qu, Juan Nathaniel, Shuolin Li, and Pierre Gentine. Deep generative data assimilation in multimodal setting. 06 2024. doi: 10.1109/CVPRW63382.2024.00050.

M. Raissi, P. Perdikaris, and G.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, 2019a. ISSN 0021-9991. doi: https://doi.org/10.1016/j.jcp.2018.10.045. URL https://www.sciencedirect.com/science/article/pii/S0021999118307125.

Maziar Raissi, Paris Perdikaris, and George Em Karniadakis. Physics informed deep learning (part i): Data-driven solutions of nonlinear partial differential equations. *arXiv preprint arXiv:1711.10561*, 2017.

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, 2019b.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Bjorn Ommer. High-Resolution Image Synthesis with Latent Diffusion Models . In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10674–10685, Los Alamitos, CA, USA, June 2022. IEEE Computer Society. doi: 10.1109/CVPR52688.2022.01042. URL https://doi.ieeecomputersociety.org/10.1109/CVPR52688.2022.01042.

Salva Rühling Cachay, Bo Zhao, Hailey Joren, and Rose Yu. Dyffusion: A dynamics-informed diffusion model for spatiotemporal forecasting. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 45259–45287. Curran Associates, Inc., 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/8df90a1440ce782d1f5607b7a38f2531-Paper-Conference.pdf.

Dule Shu, Zijie Li, and Amir Barati Farimani. A physics-informed diffusion model for high-fidelity flow field reconstruction. *Journal of Computational Physics*, 478:111972, April 2023. ISSN 0021-9991. doi: 10.1016/j.jcp.2023.111972. URL http://dx.doi.org/10.1016/j.jcp.2023.111972.

Aliaksandra Shysheya, Cristiana Diaconu, Federico Bergamin, Paris Perdikaris, José Miguel Hernández-Lobato, Richard E. Turner, and Emile Mathieu. On conditional diffusion models for PDE simulations. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=nQl8EjyMzh.

Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *CoRR*, abs/2010.02502, 2020. URL https://arxiv.org/abs/2010.02502.

Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=PxTIG12RRHS.

A Appendix

A.1 Derivations of Simplified Training Loss

Continuing from Sec 4.1 of the main paper, we had briefly discussed the problem formulation of PDEDIFF. The inverse diffusion step for $q_\theta(\Psi_{t-1}|\Psi_t, \Psi_0, x)$ that maximizes the log-likelihood of $q(\Psi_0|x)$ is defined in Eq. 7 of the paper as:

$$\log q(\Psi_0|x) = \log \int q(\Psi_{0:T}|x) d\Psi_{1:T} \tag{16}$$

Below are the steps for deriving and simplifying the lower evidence bound for our conditioned DDPM. Note that log of an expectation over a distribution might be intractable, so we consider optimizing the evidence lower bound like previous work Ho et al. (2020b); Song et al. (2020).

$$\begin{aligned}
\log q(\Psi_0|x) &= \log \int q(\Psi_{0:T}|x) d\Psi_{1:T} \\
&= \log \int \frac{q(\Psi_{0:T}|x)}{p(\Psi_{1:T}|\Psi_0, x)} p(\Psi_{1:T}|\Psi_0, x) d\Psi_{1:T} \\
&= \log \mathbb{E}_{p(\Psi_{1:T}|\Psi_0,x)} \left[\frac{q(\Psi_{0:T}|x)}{p(\Psi_{1:T}|\Psi_0, x)} \right] \\
&\geq \mathbb{E}_{p(\Psi_{1:T}|\Psi_0,x)} \left[\log \frac{q(\Psi_{0:T}|x)}{p(\Psi_{1:T}|\Psi_0, x)} \right] \\
&= \mathbb{E}_{p(\Psi_{1:T}|\Psi_0,x)} \left[\log \frac{q(\Psi_T|\Psi_0, x) \prod_{t=1}^{T} q(\Psi_{t-1}|\Psi_t, x)}{\prod_{t=1}^{T} p(\Psi_t|\Psi_{t-1}, \Psi_0, x)} \right] \\
&= \mathbb{E}_{p(\Psi_{1:T}|\Psi_0,x)} \left[\log \frac{q(\Psi_T|x) q_0(\Psi_0|\Psi_1, x) \prod_{t=1}^{T-1} q_\theta(\Psi_t|\Psi_{t+1}, x)}{\prod_{t=}^{T} p(\Psi_t|\Psi_{t-1}, \Psi_0, x)} \right] \\
&= \mathbb{E}_{p(\Psi_{1:T}|\Psi_0,x)} \left[\log \frac{q(\Psi_T|x) q_0(\Psi_0|\Psi_1, x) \prod_{t=2}^{T} q_\theta(\Psi_{t-1}|\Psi_t, x)}{\prod_{t=1}^{T} p(\Psi_t|\Psi_{t-1}, \Psi_0, x)} \right]
\end{aligned}$$

Observed that products inside the log term could be rewritten as

$$\begin{aligned}
\log q(\Psi_0|x) &\geq \mathbb{E}_{p(\Psi_{1:T}|\Psi_0,x)} \left[\log \frac{q(\Psi_T|x) q_0(\Psi_0|\Psi_1, x)}{p(\Psi_1|\Psi_0, x)} + \sum_{t=2}^{T} \log \frac{q_\theta(\Psi_{t-1}|\Psi_t, x)}{p(\Psi_t|\Psi_{t-1}, \Psi_0, x)} \right] \\
&= \mathbb{E}_{p(\Psi_{1:T}|\Psi_0,x)} \left[\log \frac{q(\Psi_T|x) p_0(\Psi_0|\Psi_1, x)}{p(\Psi_1|\Psi_0, x)} \right] \\
&\quad + \sum_{t=2}^{T} \mathbb{E}_{p(\Psi_{1:T}|\Psi_0,x)} \left[\log \frac{q_\theta(\Psi_{t-1}|\Psi_t, x)}{p(\Psi_t|\Psi_{t-1}, \Psi_0, x)} \right] \\
&= \mathbb{E}_{p(\Psi_1|\Psi_0,x)} \left[\log \frac{q_0(\Psi_0|\Psi_1, x)}{p(\Psi_1|\Psi_0, x)} \right] + \sum_{t=2}^{T} \mathbb{E}_{p(\Psi_{t-1}, \Psi_t|\Psi_0,x)} \left[\log \frac{q_\theta(\Psi_{t-1}|\Psi_t)}{p(\Psi_t|\Psi_{t-1}, \Psi_0, x)} \right] \\
&= \mathbb{E}_{p(\Psi_1|\Psi_0,x)} \left[\log \frac{q_0(\Psi_0|\Psi_1, x)}{p(\Psi_1|\Psi_0, x)} \right] \\
&\quad + \sum_{t=2}^{T} \underbrace{\mathbb{E}_{p(\Psi_t|\Psi_0,x)} \left[D_{KL}(p(\Psi_{t-1}|\Psi_t, \Psi_0, x) || q_\theta(\Psi_{t-1}|\Psi_t, x)) \right]}_{=\mathcal{L}_{\text{data},t}}
\end{aligned}$$

A.2 Details of the steps for training and sampling algorithm

We are also providing details of the training and sampling algorithm here for completeness. The forward diffusion process is simulated by gradually adding controlled Gaussian noise to Ψ, which

can be model as a conditional distribution $p(\Psi_{t+1}(x)|\Psi_t(x), x) \sim \mathcal{N}(\sqrt{1-\beta_t}\Psi_t(x), \beta_t I)$, where $\{\beta_t\}_{t=1}^T$ is a sequence of noise scheduler.

A.2.1 Latent steps for training

We can use reparametrization trick to write $p(\Psi_{t+1}(x)|\Psi_t(x), x)$ as

$$\Psi_{t+1}(x) = \sqrt{1-\beta_t}\Psi_t(x) + \sqrt{\beta_t}\epsilon_t, \quad \epsilon_t \sim \mathcal{N}(0, I)$$

We can unroll Ψ_{t+1} based on Ψ_{t-1}

$$\begin{aligned}\Psi_{t+1}(x) &= \sqrt{1-\beta_t}(\sqrt{1-\beta_{t-1}}\Psi_{t-1}(x) + \sqrt{\beta_{t-1}}\epsilon_{t-1}) + \sqrt{\beta_t}\epsilon_t, \quad \epsilon_{t-1} \sim \mathcal{N}(0, I)\\ &= \sqrt{1-\beta_t}\sqrt{1-\beta_{t-1}}\Psi_{t-1}(x) + \sqrt{1-\beta_t}\sqrt{\beta_{t-1}}\epsilon_{t-1} + \sqrt{\beta_t}\epsilon_t\\ &= \sqrt{1-\beta_t}\sqrt{1-\beta_{t-1}}\Psi_{t-1}(x) + \sqrt{1-(1-\beta_t)(1-\beta_{t-1})}\epsilon\end{aligned}$$

where $\epsilon \sim \mathcal{N}(0, I)$; and the third equal sign holds since sum of two Gaussian distribution is also a Gaussian distribution.

Unroll with respect to t, we have Ψ_{t+1} in term of Ψ_0

$$\begin{aligned}\Psi_{t+1}(x) &= \sqrt{1-\beta_t}\sqrt{1-\beta_{t-1}}\Psi_{t-1}(x) + \sqrt{1-(1-\beta_t)(1-\beta_{t-1})}\epsilon\\ &= \prod_{s=1}^{t+1}\sqrt{1-\beta_s}\Psi_0(x) + \sqrt{1-\prod_{s=1}^{t+1}(1-\beta_s)}\epsilon\end{aligned}$$

So for brevity of notation, we denote

$$\bar{\alpha}_{t+1} = \prod_{s=1}^{t+1}(1-\beta_s) \tag{17}$$

as used in Algorithm 1 of the main paper.

A.2.2 DDIM sampling

We adapt the sampling technique from Song et al. (2020) for more efficient sampling. Details are provided here for completeness.

In step 6 of Algorithm 2, we obtain

$$\hat{\Psi}_0(x_i) = \text{CCED}_\theta(\Psi_t(x_i), x_i, t) \tag{18}$$

We can update denoise step $\Psi_{t-1}(x_i)$ using the predicted $\hat{\Psi}_0(x_i)$ and $\Psi_t(x_i)$

$$\Psi_{t-1}(x_i) = \sqrt{\bar{\alpha}_{t-1}}\hat{\Psi}_0(x_i) + \sqrt{\frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}}\left(\Psi_t(x_i) - \sqrt{\bar{\alpha}_t}\hat{\Psi}_0(x_i)\right) \tag{19}$$

with $\bar{\alpha}_t$ defined as in A.2.1.

A.3 Architecture Implementation

In Sec. 4.3 of the main paper, we had briefly discussed that PDEDIFF employs a coordinate-conditioned encoder decoder (CCED) denoiser that converts a noisy input into its corresponding denoised output by conditioning on the spatial coordinates.

The first step in the CCED forward pass is the encoder, where the noisy input, $\mathbf{x}$ and t are passed through this encoder layer that has multiple sets of the *Conditional Block* and between each of these blocks, the activation function used is *softplus* or *tanh* based on the problem setting and the residual equations involved. This encoder converts all this data into a low-dimensional latent representation.

The next step is the bottleneck layer that also consists of the *Conditional Block* and captures the encoding within this low dimensional latent space to capture critical information required for denoising the input.

After the bottleneck layer, the latent information passes through the decoder, similar to the encoder, transforms the latent information into the original domain to give us the predicted noise. Every layer of this decoder also has a skip connection from the encoder layer to effectively relay the encodings to improve efficiency and accuracy.

We implement PDEDIFF in `PyTorch 2.5.1` with `CUDA 12.0` on an NVIDIA A100 GPU. All random seeds were fixed using `torch.manual_seed`, `np.random.seed`, and `torch.cuda.manual_seed`.

We use a lightweight CCED to predict $(\hat{\psi}, \hat{E})$ given noisy input $\mathbf{z}$, the diffusion timestep $t \in \{0, \ldots, T-1\}$, and the spatial coordinate $\mathbf{x}$:

Diffusion Hyperparameters:

- *Steps*: $T = 100$.
- *Noise schedule*: cosine, $\beta_t \in [10^{-5}, 10^{-2}]$
- *Objective*: DDPM L_{simple} on x_0 plus physics residual weight c_{res}
- *Optimizer*: AdamW, lr $= 5 \times 10^{-4}$
- *Training*: 1000 updates, batch 32
- *Sampling*: deterministic DDIM "single-step"

The implementation of our code can be found at: code

A.4 Metrics for toy examples

We discussed the P-MSE and Wasserstein metric for Schrodinger equations. For toy examples where we consider settings involve circles, the metrics are defined slightly different from that of the ones described in Section 4.4 of the main paper since we do not have explicit eigenvalues for circle setups, and distributions on circles are not similar to those belongs to wavefunctions. We used mean squared error (MSE) to and Wasserstein distance on uniform distribution of circles (setting dependent) to evaluate the generated samples against ground truth values.

A.4.1 Mean squared error

This metric calculates the mean squared error of the samples generated with the ground truth circles coordinates. We compute the squared distance to all possible y-coordinates for an spatial x and simply choose the minimum. Formally, the metric can be defined as below:

$$\text{MSE} = \mathbb{E}_{\mathbf{x} \sim \Omega} \left[\min_{y_i \in \mathcal{Y}} \|\hat{y} - y_i\|_2^2 \right]$$
$$\text{where} \quad \mathcal{Y} \text{ is set of possible solution of } y\text{-coordinates}$$

A.4.2 Wasserstein distance on uniform distributions

For two probability measures p, q on a domain $\Omega \subset \mathbb{R}^d$ Denotes $\Gamma(p, q)$ the set of couplings with marginals p and q

$$W_1(p, q) = \inf_{\gamma \in \Gamma(p,q)} \int_{\Omega \times \Omega} \| \mathbf{x} - \mathbf{y} \| \, \mathrm{d}\gamma(\mathbf{x}, \mathbf{y}), \tag{20}$$

where $\Gamma(p, q)$ denotes the set of couplings with marginals p and q.

Let $\{x_i\}_{i=1}^n \subset \mathbb{R}$ denote sampled x-coordinates and let $\{\hat{y}_i\}_{i=1}^n$ be the predicted y-values from a model. We define a uniform distribution on $(x_i, \hat{y}_i)$ pairs. We define the predicted distribution to be

$$p_{\text{pred}}((x_i, \hat{y}_i)) = \frac{1}{n} \delta_{(x_i, \hat{y}_i)}$$

Unit circle. We define the distribution for ground truth pairs (x_i, y_i) for each x_i uniformly sampled from $[-1, 1]$ as follow

$$p_{\text{true}}((x_i, y_i)) = \frac{1}{2n}\delta_{(x_i,y_i)}, \quad \text{where } \delta_{(x_i,y_i)} = \begin{cases} 1, & \text{if } x_i^2 + y_i^2 = 1 \\ 0, & \text{otherwise} \end{cases}$$

Disjoint circles We define the distribution for ground truth pairs (x_i, y_i) for each x_i uniformly sampled from $[-1, 3]$ as follow

$$p_{\text{true}}((x_i, y_i)) = \sum_{k=1}^{2} \pi_k \frac{1}{2n}\delta_{(x_i,y_i)}$$

where

$$\delta_{(x_i,y_i)} = \begin{cases} 1, & \text{if } x_i^2 + y_i^2 = 1, x_i < 1 \text{ or } (x_i - 2)^2 + (y_i - 2)^2 = 1, x_i \geq 1 \\ 0, & \text{otherwise} \end{cases}$$

and π_k is the weight if (x_i, y_i) belongs to circle k ($k = 1$ for circle centers at $(0, 0)$ and $k = 2$ for circle centers at $(2, 2)$)

Concentric circles We define the distribution for ground truth pairs (x_i, y_i) for each x_i uniformly sampled from $[-1, 1]$ as follow

$$p_{\text{true}}((x_i, y_i)) = \sum_{k=1}^{2} \pi_k \frac{1}{2kn}\delta_{(x_i,y_i)}$$

where

$$\delta_{(x_i,y_i)} = \begin{cases} 1, & \text{if } (x_i, y_i) \in \mathcal{C} \\ 0, & \text{otherwise} \end{cases}$$

with

$$\begin{aligned} \mathcal{C} &= \mathcal{C}_1 \cup \mathcal{C}_2 \\ \text{with } \mathcal{C}_1 &= \{(x_i, y_i) : x_i^2 + y_i^2 = 9, \text{ and } x_i \notin [-1, 1]\} \\ \mathcal{C}_2 &= \{(x_i, y_i) : x_i^2 + y_i^2 = 1 \text{ or } x_i^2 + y_i^2 = 9, \text{ and } x_i \in [-1, 1]\} \end{aligned}$$

and π_k is the weight if (x_i, y_i) belongs to $\mathcal{C}_k$.

A.5 Experiment Setup

In this section, we continue from Section 5 of the main paper and discuss the results of PDEDIFF in the problem settings defined earlier. We first discuss more details of the problem settings, i.e. how did we generate data for training and hyperparameters used in each physics setting.

A.5.1 1DSchrodinger: Infinite potential well/ Particle in a box

We consider the time-independent Schrödinger equation for a particle in a one-dimensional infinite potential well:

$$E\psi(x) = \underbrace{\left(-\frac{\hbar^2}{2m}\partial_x^2 + V(x)\right)}_{H}\psi(x)$$

where $V(x)$ denotes the infinite potential well, or in other words, the space of a 1D box that contains the particle where it moves freely inside but trapped between the wall (x_L, x_R). The potential well can be defined as

$$V(x) = \begin{cases} 0, & x_L < x < x_R \\ \infty, & \text{otherwise} \end{cases}$$

In this setting, the particle is confined within (x_L, x_R), with Dirichlet boundary conditions $\psi(x_L) = \psi(x_R) = 0$.

Solving for $\psi(x)$, we can do it analytically as outside of x_L, x_R, $\psi(x) = 0$, so the wavefunction takes the form:

$$\psi(x) = \sqrt{\frac{2}{L}} \sin\left(\frac{n\pi x}{L}\right)$$

where n are positive integer values.

Dataset Generation: The dataset for 1D infinite potential well was generated by uniformly sampling spatial coordinates between $[0, L_x]$. After sampling the spatial coordinates, we analytically calculate the wavefunction and the energy values. For our experiments, we simplify the constant values as $\hbar = 1, m = 1$, and the size of the well is taken as $L_x = 3$. We generate 100 data points for each of the first few eigenstates for training, i.e. we choose $n = 1, 2, 3$ for the 1D TISE 3 waves setting, and $n = 1, 2$ for the 1D TISE 2 waves setting.

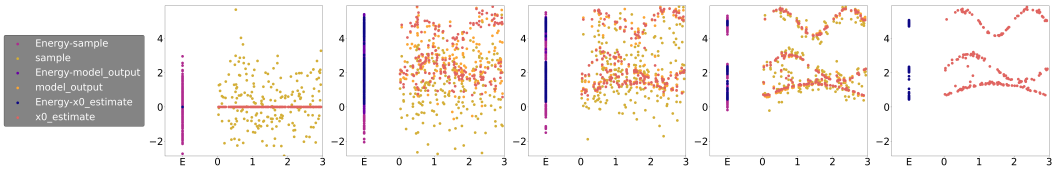


Figure 4: PDEDIFF denoising steps with $c_{res} = 0.001$ (left to right: denoise step $t = 100, 70, 40, 10, 0$). The left-most subplot shows noisy samples from Gaussian and each of the steps to the right shows the denoising process. The right-most subplot show the generated samples Ψ_0.

A.5.2 1D Gross-Pitaevskii equation

We consider also one-dimensional time-independent Gross-Pitaevskii equation, a form of non-linear Schrodinger equation. In specific, the Halmiltonian operator $\mathcal{H}$ is non-linear and written as follow:

$$E\psi(x) = \underbrace{\left(\frac{h^2\partial_x^2}{2m} + V(x) + U_0|\psi(x)|^2\right)}_{\mathcal{H}}\psi(x)$$

where U_0 is interaction force, $V(x) = \frac{1}{2}mx^2$ is the harmonic trapping potential.

Not many non-linear Schodinger equations have closed-form solution, such as the one presented here. We obtain training samples and ground truth by solving the time-independent Gross-Pitaevskii equation via an explicit imaginary-time algorithm as in Chiofalo et al. (2000).

Setup: In this setting, the particle is free to move within $(x_L, x_R) \subset \mathbb{R}$, with Dirichlet boundary conditions $\psi(x_L) = \psi(x_R) = 0$. We generate 100 data points for each wavefunction and their corresponding energy using the algorithm in Chiofalo et al. (2000) for training and generate 200 points to evaluate our framework against the ground truth.

A.5.3 2DSchrodinger: Square infinite potential well

We next consider the time-independent Schrödinger equation for a particle in a two-dimensional square infinite potential well:

$$E\psi(\mathbf{x}) = \underbrace{\left(-\frac{\hbar^2}{2m}(\partial_x^2 + \partial_y^2) + V(\mathbf{x})\right)}_{H}\psi(\mathbf{x})$$

where $V(\mathbf{x})$ denotes the infinite potential well, or in other words, the space of a 2D box that contains the particle where it moves freely inside but trapped between the walls $\mathbb{B} = [x_L, x_R] \times [y_L, y_R] \subset \mathbb{R}^2$. The potential well can be defined as:

$$V(\mathbf{x}) = \begin{cases} 0, & x_L < x < x_R, \quad y_L < y < y_R \\ \infty, & \text{otherwise} \end{cases}$$

In this setting, the particle is confined within $\mathbb{B}$,with Dirichlet boundary conditions, i.e., $\psi(\mathbf{x}) = 0$ for any $\mathbf{x} \in \partial\mathbb{B}$, where $\partial\mathbb{B}$ denotes the boundary of the box $\mathbb{B}$.

Solving for $\psi(\mathbf{x})$, we can do it analytically as outside of $\mathbb{B}$, $\psi(\mathbf{x}) = 0$, so the wavefunction takes the form:

$$\psi(\mathbf{x}) = \frac{2}{L} \sin\left(\frac{n_x \pi x}{L}\right) \sin\left(\frac{n_y \pi y}{L}\right)$$

where n_x, n_y are positive integer values.

Set up: The dataset for 2D infinite potential well was generated by uniformly sampling spatial coordinates between $x = [0, L_x], y = [0, L_y]$. After sampling the spatial coordinates, we analytically calculate the wavefunction and the energy values. For our experiments, we simplify the constant values as $\hbar = 1, m = 1$, and the size of the well is taken as $L_x = L_y = 3$. For each eigenstate, we sample 200 data points for all the states in total. For 3 wave setting, we choose the states (n_x, n_y) as $(1, 1), (2, 1), (2, 2)$, and for the 4 wave settings, we experiment on degenerate states $(1, 1), (1, 2), (2, 1), (2, 2)$, i.e. some wavefunctions has the same energy, and nondegenerate states $(1, 1), (2, 1), (3, 1), (4, 1)$, i.e. every wavefunctions has distinct energy.

A.5.4 Helmholtz Equation

The Helmholtz equation is an inhomogeneous elliptic PDE that usually arises from applications in heat conduction, ultrasound or acoustics Basu & Rani (2021). It can be express as follow:

$$\underbrace{\nabla^2 \psi(x) + k^2 \psi(x)}_{\mathcal{H}_k \psi} = f(x)$$

where integer or real valued parameter k is known as frequency or wave number, f is a source function that describe the emission source of the wave $\psi(x)$, with $x \in \Omega$. Our goal is to recover the high resolution propagation of the wave through the environment $\psi(x)$ and its frequency k. Training data is simulated by solving the second-order finite-difference linear system $\mathcal{H}\psi = f$ with low fidelity. The results are presented in Table 2.

Setup: We generate 128 data points for each wave mode and their corresponding frequency k using the linear algebra solver on the discrete differential operator by second-order finite differences for training and generate 256 points to evaluate our framework against the ground truth.

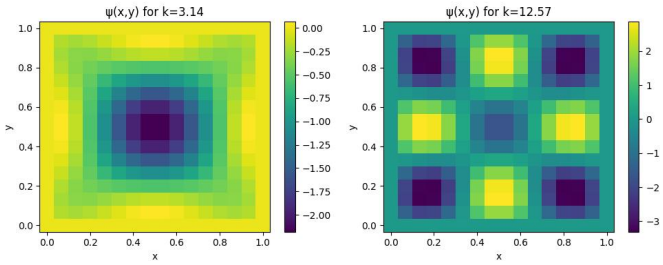
A.5.5 Toy examples

In this section, we discuss in more details for the experiment setup for the toy examples with circles. Given some x coordinates that are uniformly sampled from a given range (such as $[-1, 1]$ for unit circle). We want to generate the corresponding y coordinates that satisfy physics guidance.

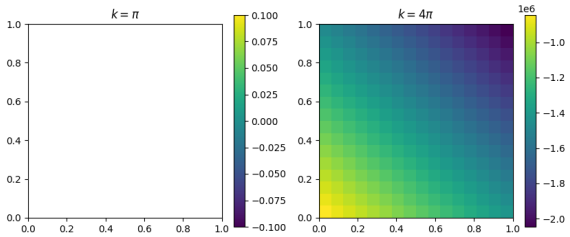
Unit circle. This problem could be formulated as a simple problem $\sqrt{1 - x^2} = \pm y$, in which the "eigen-values" ± 1 is not explicitly exposed to the model. We instead required the model to learn this and output $y|x$ from provided training pairs (x, y) and the physical constraints $x^2 + y^2 = 1$. Note that for a given x, there are 2 possible values for $y-$coordinates. We generate 100 points on the unit circle with coordinates (x, y) satisfying $x^2 + y^2 = 1$ for training.

Disjoint circle. To study if the diffusion model is able to learn distinct distributions, we generate 200 points, with 100 points on a unit circle with coordinates (x, y) satisfying $x^2 + y^2 = 1$ with centroid at $(0, 0)$, and another 100 points on a circle with coordinates (x, y) satisfying $(x-2)^2+(y-2)^2 = 1$ with center at $(2, 2)$. The two circles do not overlap each other.

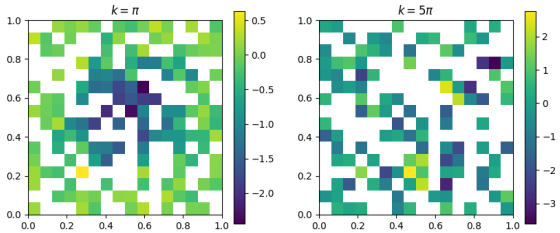
Concentric circle. We generate 200 points, with 100 points on a circle with coordinates (x, y) satisfying $x^2 + y^2 = 1$, and another 100 on a circle with coordinates (x, y) satisfying $x^2 + y^2 = 9$. The two circles share the same centroid at $(0, 0)$. This creates a more complicated setting where for some x coordinates, there exist 4 possible solutions. Figure 8 illustrates that PDEDIFF manages to distinguish different distributions, while PINN collapsed to only learning the mean of the data.



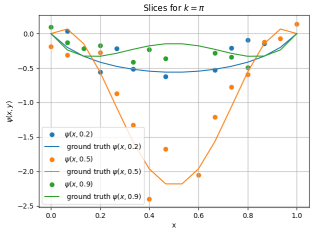
(a) Ground truth for Helmholtz equation



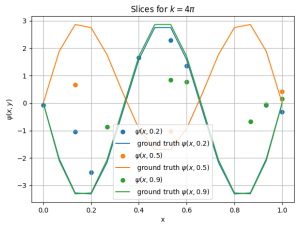
(b) PINN prediction for Helmholtz equation



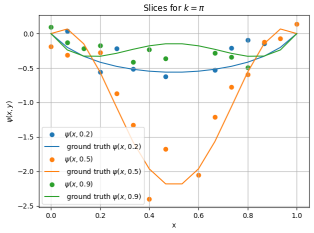
(c) PDEDIFF prediction for Helmholtz equation



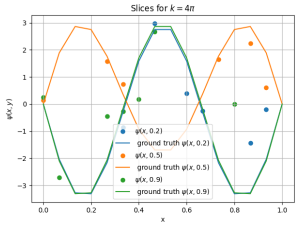
(d) PDEDIFF generation with $c_{res} = 0$, $k = \pi$



(e) PDEDIFF generation with $c_{res} = 0$, $k = 4\pi$



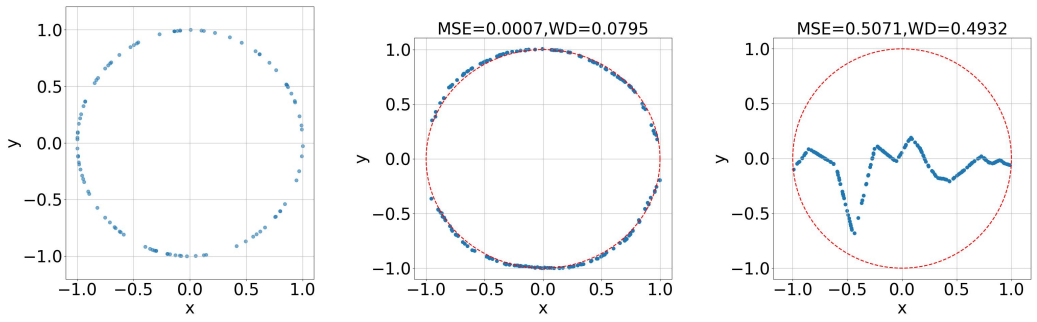
(f) PDEDIFF generation with $c_{res} = 0.001$, $k = \pi$



(g) PDEDIFF generation with $c_{res} = 0.001$, $k = 4\pi$

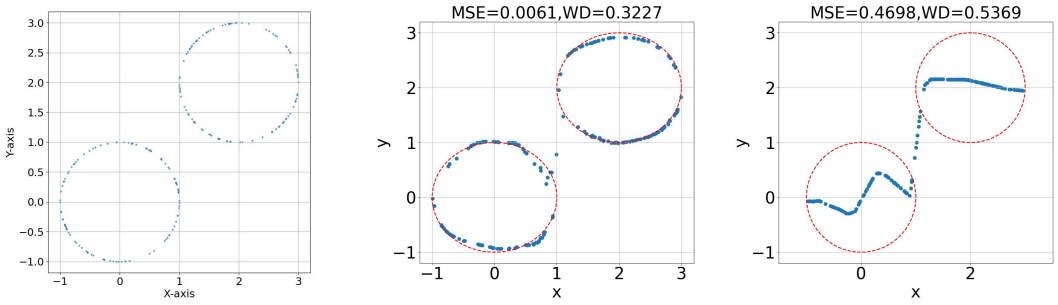
Figure 5: Generated samples by physics-informed PDEDIFF with different c_{res} values

For circle settings, we consider the weighting for physics informed loss $c_{res} = 0.01$. To obtain new samples for visualization and error quantification, we generate 200 points with Algorithm 2 of the main paper. The generation by physics-informed PDEDIFF with $c_{res} = 0.01$ can be visualize at figures 6b, 7b, 8b for unit circle, disjoint circles and concentric circles, respectively. In comparison, generation by PINN (also with $c_{res} = 0.01$) are shown in Figures 6c, 7c, 8c, for unit circle, disjoint circles and concentric circles, respectively. For each toy setting, we run 5 experiments for each models on different seeds and present the average and one standard deviation of the metrics obtained with each model in table 3. In general, the performance of PDEDIFF with physics-informed loss outperforms that of PINN with similar residual coefficient c_{res}, assessed using the P-MSE and the Wasserstein metric. The experiments with toy examples of circles verify the performance of our method and show potential of physics-informed diffusion in distinguishing multimode distribution.



(a) Ground Truth (b) PDEDIFF (Physics-informed) (c) PINN

Figure 6: Comparison of generated samples by physics-informed PDEDIFF and PINN for unit circle. The dotted red line shows the ground truth and the blue dots are the inference point clouds.

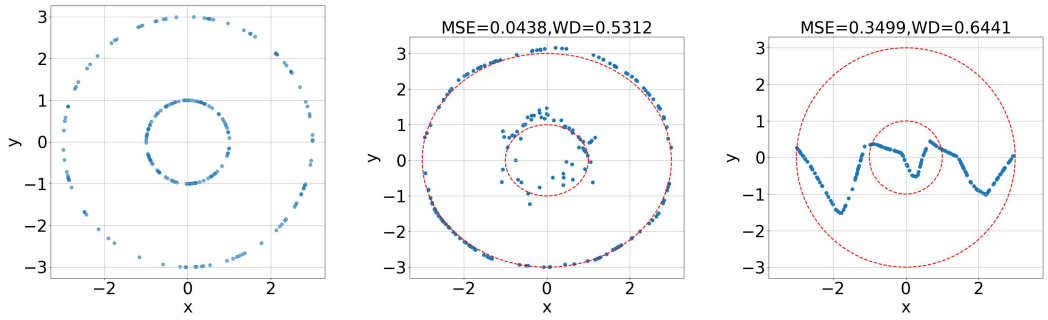


(a) Ground Truth (b) PDEDIFF (Physics-informed) (c) PINN

Figure 7: Comparison of generated samples by physics-informed PDEDIFF and PINN for disjoint circles. The dotted red line shows the ground truth and the blue dots are the inference point clouds.

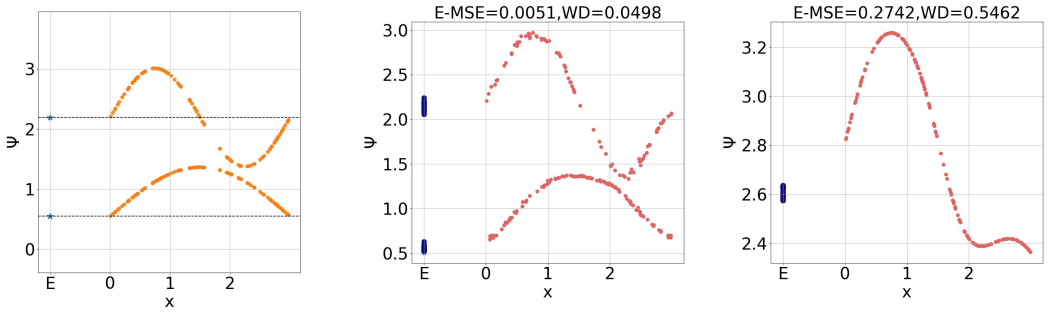
A.6 Ablation Study

To study which weight c_{res} would be best guided our physics-informed PDEDIFF, for each setting, we ran 5 experiments for each value $c_{res} \in \{0, 0.001, 0.005, 0.01, 0.05, 0.1, 1, 10\}$ and average the P-MSE and WD metrics. The best average metrics for non-zero c_{res} for each setting of Schrodinger equation are presented in Table 1 and Table 2 of the main paper.



(a) Ground Truth (b) PDEDIFF (Physics-informed) (c) PINN

Figure 8: Comparison of generated samples by physics-informed PDEDIFF and PINN for concentric circles. The dotted red line shows the ground truth and the blue dots are the inference point clouds.



(a) Ground Truth (b) PDEDIFF (Physics-informed) (c) PINN

Figure 9: Comparison of generated samples by physics-informed PDEDIFF and PINN for 1D Schrödinger equation with 2 wavefunctions in infinite potential well.

A.6.1 On weights for residual loss and metrics

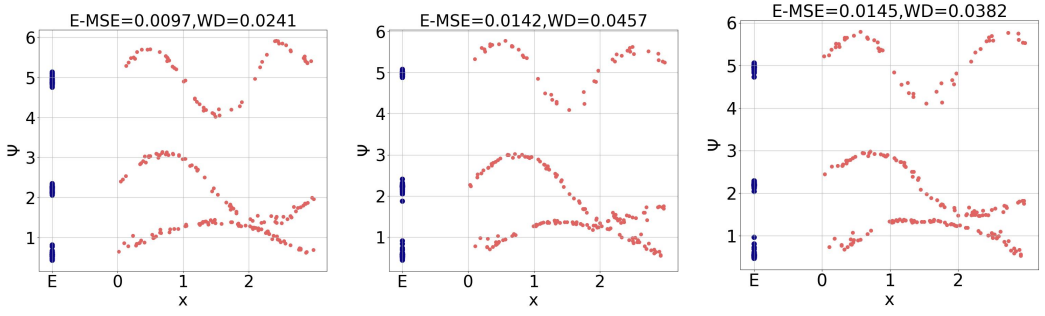
In the circle experiments, where the ground truth distribution $\mathbb{P}(Y|X)$ has finitely many modes, PDEDIFF successfully captures all branches of the distribution. PINN, however, treats the problem as a regression task, learning the conditional mean $\mathbb{E}(Y|X)$ and thus fails to accurately predict the y-values for given x-coordinates. More visualizations of our toy examples that demonstrate this behavior can be found in 8, 7, 6.

As we increase the c_{res} value for the physics constraint, this reduces our training loss as well as the MSE metric (results shown in table 4, 5). Small c_{res} values (0.001, 0.005, 0.01) give low P-MSE and WD metric, with $c_{res} = 0.001$ achieving the best P-MSE and WD for learning 3 wavefunctions for 1D TISE, and $c_{res} = 0.005$ achieving the best metrics for learning 2 wavefunctions for 1D TISE, as shown in table 4. An example of the learned denoising process with $c_{res} = 0.001$ is shown in figure 4.

For 1D Schrodinger cases, a higher c_{res}, i.e. $c_{res} = 0.1, 1, 10$ resulted in a flatter curve distribution (figure 11) compared to the distribution learned with a smaller $c_{res} = 0.001, 0.005, 0.01$ (figure 10). One reason might be that strong physics residuals make the distribution converge to the trivial solution where the wavefunction $\psi(x) = 0$ and the energy $E = 0$. This trivial solution also satisfies the Schrodinger equation of the form $\mathcal{H}\psi = E\psi$. In addition, more points tend to concentrate around the first predicted eigenvalue, or also known as mode collapsed. This occurs when we enforce a strong penalty on the physics-informed loss.

Across all settings, the residual weight (c_{res}) is balancing the two scenarios: a small value allows for more flexible generation, while for larger values it forces the model to learn a trivial solution. As observed in Table 4, increasing c_{res}, causes the model to collapse to a single mode, hence the MSE loss decreases significantly, but the Wasserstein distance increases.

Similarly, for the 2D Schrodinger cases, we can see a similar trend with respect to c_{res} in Table 5. For the harder 4 wave case, PDEDIFF achieves a better Wasserstein metric as compared to the other methods. Since the "average" eigenstate for the 4 waves is the degenerate state, i.e., the eigenstates with the same energy $((1, 2), (2, 1))$, PINN is able to give a much lower P-MSE value, but we can see from the Wasserstein metric, the samples generated have collapsed onto this average state and is not able to generate a mixture of all the learned states. PDEDIFF in this case is still able to prevent this catastrophic mode collapse and generate a good mixture of eigenstates.

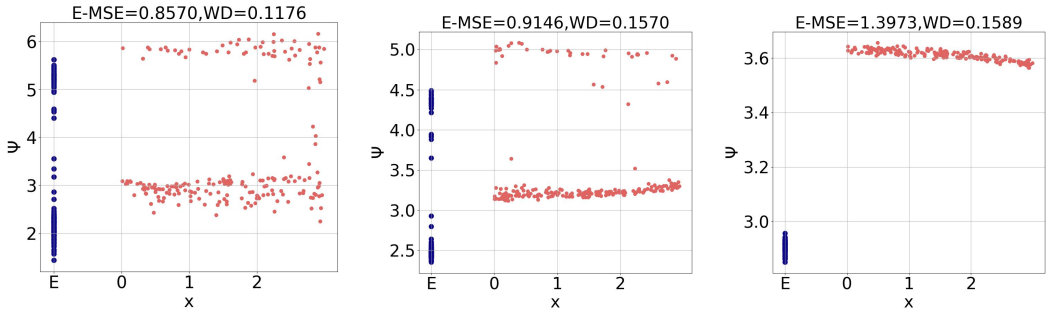


(a) PDEDIFF with $c_{res} = 0.001$ (b) PDEDIFF with $c_{res} = 0.005$ (c) PDEDIFF with $c_{res} = 0.01$

Figure 10: Generated samples by physics-informed PDEDIFF with different c_{res} values

A.7 Limitations and Future Work

As briefly discussed in the main paper, we have demonstrated that PDEDIFF provides a more flexible approach towards incorporating physics in diffusion models as the data is no longer bound to a mesh and the calculation of physics residual can be performed using autograd instead of writing custom finite difference kernels. We have also presented evidence that adding a physics informed loss to a conditional diffusion model can generate solution of PDEs that are more faithful to the underlying physics as compared to using a PINN or a vanilla diffusion model.



(a) PDEDIFF with $c_{res} = 0.1$ (b) PDEDIFF with $c_{res} = 1$ (c) PDEDIFF with $c_{res} = 10$

Figure 11: Generated samples by physics-informed PDEDIFF with different c_{res} values

Setting		Unit circle		Disjoint circles		Concentric circles	
c_{res}		P	Q	P	Q	P	Q
0	MSE	0.389±0.058	0.0028±0.0014	0.489±0.053	0.013±0.0023	0.277±0.085	0.052±0.014
	WD	0.475±0.010	0.0116±0.0192	0.533±0.015	0.421±0.055	0.907±0.032	0.560±0.038
0.01	MSE	0.379±0.085	0.0014±0.0006	0.486±0.052	0.012±0.0038	0.274±0.092	0.052±0.012
	WD	0.471±0.013	0.0852±0.0191	0.532±0.015	0.417±0.089	0.906±0.037	0.549±0.030

Table 3: Performance comparison between PINN (P) and PDEDIFF (Q) across different problem settings. Here, MSE is as defined in section A.4.1 and WD is the Wasserstein metric for uniform distribution on circles defined in section A.4.2. The error is range of one standard deviation.

Setting		1D (2 waves: $n = 1, 2$)		1D (3 waves: $n = 1, 2, 3$)	
c_{res}		P	Q	P	Q
0	PMSE	0.822 ± 0.016	0.0168 ± 0.010	0.371 ± 0.031	0.0218 ± 0.013
	WD	0.539 ± 0.022	0.0863 ± 0.059	0.737 ± 0.023	0.0549 ± 0.009
0.001	PMSE	0.974 ± 0.247	0.0137 ± 0.013	0.460 ± 0.435	0.0123 ± 0.012
	WD	0.423± 0.052	0.0865 ± 0.063	0.422 ± 0.151	0.0499 ± 0.019
0.005	PMSE	1.052 ± 0.329	0.0146 ± 0.011	0.521 ± 0.079	0.0299 ± 0.011
	WD	0.244 ± 0.031	0.0862 ± 0.061	0.315 ± 0.088	0.0716 ± 0.014
0.01	PMSE	1.106 ± 0.371	0.0392 ± 0.032	0.534 ± 0.167	0.0347 ± 0.006
	WD	0.313 ± 0.020	0.113 ± 0.037	0.266 ± 0.064	0.0838 ± 0.022
0.05	PMSE	1.923 ± 0.456	0.239 ± 0.157	1.330 ± 0.528	0.532 ± 0.222
	WD	0.327 ± 0.069	0.152 ± 0.057	0.399 ± 0.154	0.176 ± 0.130
0.1	PMSE	3.414 ± 1.49	0.748 ± 0.411	7.63 ± 6.148	0.807 ± 0.108
	WD	0.587 ± 0.130	0.153 ± 0.057	0.503 ± 0.132	0.111 ± 0.013
1	PMSE	3325 ± 2996	1.05 ± 0.317	6989 ± 6330	0.971 ± 0.171
	WD	0.306 ± 0.206	0.188 ± 0.476	0.224 ± 0.149	0.145 ± 0.059
10	PMSE	3540 ± 2741	2.05 ± 1.753	7552 ± 5669	1.56 ± 1.610
	WD	0.236 ± 0.110	0.170 ± 0.009	0.154 ± 0.056	0.143 ± 0.0241

Table 4: Performance comparison between PINN (P) and PDEDIFF (Q) across different problem settings. PMSE here is P-MSE and WD is the Wasserstein metric. The lower values represent better match between the ground truth and the sampled wavefunctions. The error is range of one standard deviation.

Setting		2D (3 waves: $(1,1),(2,1),(2,2)$)		2D (4 waves: $(1,1),(1,2),(2,1),(2,2)$)	
		P	Q	P	Q
0	PMSE	1.253 ± 0.147	1.222 ± 0.071	1.250 ± 0.059	1.309 ± 0.073
	WD	0.378 ± 0.072	0.169 ± 0.037	0.482 ± 0.073	0.150 ± 0.042
0.001	PMSE	1.239 ± 0.098	1.194 ± 0.097	1.243 ± 0.055	1.338 ± 0.074
	WD	0.340 ± 0.075	0.155 ± 0.037	0.477 ± 0.078	0.124 ± 0.051
0.005	PMSE	1.229 ± 0.061	1.201 ± 0.098	1.223 ± 0.038	1.390 ± 0.147
	WD	0.363 ± 0.100	0.187 ± 0.042	0.481 ± 0.077	0.202 ± 0.118
0.01	PMSE	1.184 ± 0.062	1.174 ± 0.098	1.181 ± 0.041	1.414 ± 0.192
	WD	0.368 ± 0.076	0.173 ± 0.047	0.527 ± 0.105	0.194 ± 0.103
0.05	PMSE	1.004 ± 0.087	0.923 ± 0.095	1.006 ± 0.042	1.088 ± 0.211
	WD	0.444 ± 0.058	0.238 ± 0.047	0.592 ± 0.094	0.386 ± 0.114
0.1	PMSE	0.793 ± 0.049	0.769 ± 0.145	0.790 ± 0.041	0.928 ± 0.109
	WD	0.502 ± 0.083	0.264 ± 0.0599	0.597 ± 0.069	0.326 ± 0.098
1	PMSE	49.531 ± 3.639	1.103 ± 1.247	21.005 ± 1.826	0.815 ± 0.165
	WD	0.239 ± 0.047	0.338 ± 0.083	0.211 ± 0.035	0.301 ± 0.032
10	PMSE	1474.578 ± 293.978	3.059 ± 2.304	460.817 ± 56.397	0.808 ± 0.201
	WD	0.169 ± 0.044	0.248 ± 0.048	0.160 ± 0.046	0.392 ± 0.162

Table 5: Performance comparison between PINN (P) and PDEDIFF (Q) across different problem settings. PMSE is P-MSE and WD is the Wasserstein metric. Lower values indicate better agreement with the ground truth wavefunctions.

Setting		1D GP (3 waves)	
		P	Q
0	PMSE	1.550 ± 0.804	0.140 ± 0.007
	WD	2.036 ± 1.222	0.195 ± 0.043
0.001	PMSE	1.291 ± 0.479	0.126 ± 0.019
	WD	2.418 ± 1.084	0.194 ± 0.033
0.005	PMSE	1.550 ± 0.804	0.118 ± 0.028
	WD	2.040 ± 1.222	0.338 ± 0.186
0.01	PMSE	1.550 ± 0.804	0.139 ± 0.032
	WD	2.036 ± 1.222	0.333 ± 0.187
0.05	PMSE	1.550 ± 0.804	0.134 ± 0.018
	WD	2.040 ± 1.222	0.937 ± 0.734
0.1	PMSE	1.550 ± 0.804	0.189 ± 0.089
	WD	2.040 ± 1.222	0.644 ± 0.453
1	PMSE	1.550 ± 0.804	0.248 ± 0.137
	WD	2.040 ± 1.222	1.100 ± 0.565
10	PMSE	1.550 ± 0.804	0.286 ± 0.204
	WD	2.040 ± 1.222	0.453 ± 0.072

Table 6: Performance comparison between PINN (P) and PDEDIFF (Q) for the 1D GP (3 waves) setting. PMSE is P-MSE and WD is the Wasserstein metric. Lower values indicate better agreement with the ground truth wavefunctions.

Our method lays the foundation for many possible extensions, such as developing a training algorithm to train the diffusion model to learn unseen eigenstates when a particular subset of eigenstates have been provided for training. Another possible future extension could be to develop a data-free approach where no dataset will be required and the denoising step can be guided to generate solutions using the physics informed regularizer.

Previously, we had also mentioned that scaling this framework to noisy and high-dimensional experimental datasets could potentially accelerate the development of real-time digital simulations of various problems. We believe that our method could be extended to other fields such as to drug discoveries, healthcare and finance, where systems can be modeled as PDEs and these PDEs will have multiple solutions.