Neural PDE Solvers with Physics Constraints

A Comparative Study of PINNs, DRM, and WANs

Jiakang Chen[1]

Supervisor: Simon Arridge

Abstract

Partial differential equations (PDEs) underpin models across science and engineering, yet analytical solutions are atypical and classical mesh-based solvers can be costly in high dimensions. This dissertation presents a unified comparison of three mesh-free neural PDE solvers, physics-informed neural networks (PINNs), the deep Ritz method (DRM), and weak adversarial networks (WANs), on Poisson problems (up to 5D) and the time-independent Schrödinger equation in 1D/2D (infinite well and harmonic oscillator), and extends the study to a laser-driven case of Schrödinger's equation via the Kramers-Henneberger (KH) transformation.

Under a common protocol, all methods achieve low L_2 errors (10^{-6}-10^{-9}) when paired with forced boundary conditions (FBCs), forced nodes (FNs), and orthogonality regularization (OG). Across tasks, PINNs are the most reliable for accuracy and recovery of excited spectra; DRM offers the best accuracy-runtime trade-off on stationary problems; WAN is more sensitive but competitive when weak-form constraints and FN/OG are used effectively. Sensitivity analyses show that FBC removes boundary-loss tuning, network width matters more than depth for single-network solvers, and most gains occur within 5000-10,000 epochs. The same toolkit solves the KH case, indicating transfer beyond canonical benchmarks.

We provide practical guidelines for method selection and outline the following extensions: time-dependent formulations for DRM and WAN, adaptive residual-driven sampling, parallel multi-state training, and neural domain decomposition. These results support physics-guided neural solvers as credible, scalable tools for solving complex PDEs.

Contents

Chapter 1

Introduction

Partial differential equations (PDEs) are the lingua franca of continuum models across mathematics, physics, and engineering. Classical examples include heat conduction governed by the heat equation [1], fluid motion described by the Navier-Stokes equations [17], electromagnetic waves captured by Maxwell's equations [3], and structural deformation modelled by the equations of elasticity. By encoding the dynamics of fields and forces, PDEs enable the prediction and optimization of system behaviour under varied initial and boundary conditions.

Exact analytical solutions are available only for special cases so numerical approximation is indispensable. The finite difference method (FDM) replaces derivatives with difference quotients on structured grids [30]; the finite volume method (FVM) enforces conservation over control volumes [12]; spectral and spectral element methods attain high-order accuracy for smooth solutions via global polynomial expansions [11]; and the finite element method (FEM), developed in the mid–twentieth century [21], uses unstructured meshes to handle complex geometries and heterogeneous media with great flexibility.

Despite their success, mesh-based techniques face persistent hurdles in modern applications. Generating high-quality meshes in high dimensions is costly: complex geometries often require substantial manual intervention, and degrees of freedom grow rapidly under refinement. These challenges have motivated mesh-free alternatives.

Recent advances in deep learning have produced neural PDE solvers that embed physical laws directly into training objectives. Physics-informed neural networks (PINNs) penalize the PDE residual and boundary conditions in the loss so that the learned surrogate satisfies the governing equations. The deep

Ritz method (DRM) minimizes the variational energy functional associated with elliptic problems, providing a principled weak formulation. Weak adversarial networks (WANs) cast solution and test functions as a generator-discriminator pair that optimizes a weak form adversarially. By incorporating physics constraints, these approaches can reduce data demands and avoid meshing, while remaining flexible across problem families. At the same time, they introduce distinct optimization pathologies (for example, spectral bias or adversarial instability) and design choices (loss weighting, sampling, and architecture) that materially affect accuracy and computational cost.

This dissertation undertakes a systematic study of three neural solvers, PINN, DRM, and WAN, on canonical elliptic and quantum benchmarks. First, we compare their performance on the Poisson equation and on the time-independent Schrödinger equation (infinite potential well and harmonic oscillator) in one and two spatial dimensions under a unified evaluation protocol. Second, we analyze and improve training through principled modifications, including forced boundary conditions (FBC) that enforce boundaries by construction, orthogonality-promoting penalties for excited states, and the use of fixed or forced nodes where nodal sets are known. Third, we extend the study beyond stationary settings by formulating and solving the laser-driven Schrödinger equation in the Kramers-Henneberger (KH) frame with neural methods, using it as a stress test of robustness and physical fidelity. Throughout, we report sensitivity analyses, ablations, and reproducibility checks, and we distil practical guidance for method selection and configuration.

The dissertation is organized as follows. Chapter 2 reviews classical numerical methods and emerging neural network-based approaches for PDEs, situating the present work in the literature. Chapter 3 lays out the methodological framework for PINNs, DRM, and WANs, specifies the benchmark problems and evaluation metrics, and details architectures, sampling, and optimization. Chapter 4 presents empirical results for the Poisson and stationary Schrödinger benchmarks, including both baseline implementations and targeted improvements. Chapter 5 formulates the laser-driven Schrödinger equation in the KH frame and develops a neural solver for the time-dependent setting. Chapter 6 synthesizes the findings, compares methods across tasks, discusses limitations and threats to validity, and outlines future directions. Chapter 7 concludes.

Chapter 2

Literature Review

Partial Differential Equations (PDEs) are widely used to formulate complex problems across the sciences. In this chapter, we will outline the basic PDE concepts before reviewing the classical numerical solution methods and their limitations. Then, we will move to the foundations of neural network and deep learning theory as, using the differentiation and optimization method, neural networks give a perfect platform for solving PDEs. Neural network-based PDE solvers will be discussed at the end of this chapter to ascertain the history and state of the art of this field. We also investigate their application to solving Schrödinger problems.

2.1 PDE Fundamentals

Partial differential equations (PDEs) are equations in which the unknown function u depends on multiple independent variables (typically spatial coordinates $\mathbf{x} \in \Omega \subset \mathbb{R}^d$ and possibly time t), and the equation involves partial derivatives of u. A general second-order PDE can be written in the form

$$F\big(\mathbf{x}, t,\, u,\, \nabla u,\, \nabla^2 u\big) \;=\; 0, \tag{2.1}$$

where ∇u and $\nabla^2 u$ denote the gradient and Hessian (matrix of second derivatives) of $u(\mathbf{x}, t)$. More generally, we can write PDEs of the kth order as

$$F\big(\mathbf{x}, t,\, u,\, Du,\, D^2 u, \cdots, D^k u\big) \;=\; 0, \tag{2.2}$$

where D^n is partial derivative operator.

2.1.1 Classification of PDEs

PDEs are called *linear* if they are linear in u and its derivatives. Second-order linear PDEs, with unknown function $u(x, y)$, can be written in the general form

$$a_1 u_{xx} + a_2 u_{xy} + a_3 u_{yx} + a_4 u_{yy} + a_5 u_x + a_6 u_y + a_7 u = f, \tag{2.3}$$

where a_i and f are functions of x and y only.

A linear PDE is deemed *homogeneous* when f is zero everywhere.

We define a *quasilinear* PDE as when a_i and f are functions of unknown and lower-order derivatives, and the highest order derivatives appear only as linear. For example, Einstein's equations of general relativity form a quasilinear PDE.

Finally, we have *fully nonlinear* PDEs which have nonlinearities on one or more of the highest-order derivatives. One well-known example is the Monge-Ampere equation [18] in differential geometry.

In our discussion, we will focus on second-order linear PDEs. These can be further classified to help us to understand their initial and boundary conditions. They can be classified by the sign of the eigenvalues of the coefficient matrix $A(\mathbf{x})$ in the principal part

$$\sum_{i,j=1}^{d} A_{ij}(\mathbf{x}) \frac{\partial^2 u}{\partial x_i \, \partial x_j}. \tag{2.4}$$

- **Elliptic:** λ_i all of same sign. E.g., Laplace's equation[1] $\Delta u = 0$, Poisson's equation $\Delta u = f$. Solutions are generally smooth and boundary-value problems are well-posed.

- **Parabolic:** one zero eigenvalue, rest of the same sign. E.g., the heat equation $u_t - \Delta u = 0$. These describe diffusion-like processes and require both initial and boundary data.

- **Hyperbolic:** mixed signs. E.g., the wave equation $u_{tt} - c^2 \Delta u = 0$. These model propagation phenomena and require Cauchy data (initial displacement and velocity) plus boundary conditions if the domain is bounded.

[1] Here, Δu denotes the Laplacian of u, which is the trace of the Hessian matrix.

2.1.2 Boundary and Initial Value Problems

For a problem to be well posed, good initial conditions (ICs) and boundary conditions (BCs) are important. If we have too many conditions, we will not find a solution. If the conditions are weak and too few, we will get non-unique solutions. Sometimes, small errors in ICs or BCs will cause large changes in the solution. Here are some common choices on ICs and BCs.

Boundary Conditions. On a spatial boundary $\partial\Omega$, one commonly prescribes the following boundary conditions:

- Dirichlet: $u(\mathbf{x}, t) = g_D(\mathbf{x}, t)$ for $\mathbf{x} \in \partial\Omega$.
- Neumann: $\nabla u(\mathbf{x}, t) \cdot \mathbf{n} = g_N(\mathbf{x}, t)$, where $\mathbf{n}$ is the outward normal.
- Robin (mixed): $\alpha u + \beta \nabla u \cdot \mathbf{n} = g_R$.

Initial Conditions. For time–dependent (parabolic or hyperbolic) PDEs, one must also specify

$$u(\mathbf{x}, 0) = u_0(\mathbf{x}).$$

In order to compare the neural network-based method, we will mainly focus on the time-independent case with Dirichlet boundary conditions.

2.2 Classical Numerical Methods

Not all the PDEs have exact solutions, in which case computational approximations are sought. Classical numerical methods for solving PDEs approximate the continuous problem by a discrete system that can be solved on a computer. The three most common approaches are finite difference methods (FDMs), finite element methods (FEMs), and spectral methods.

2.2.1 Finite Difference Methods

The finite difference methods (FDMs) are a powerful class of numerical techniques for solving PDEs by replacing continuous derivatives with discrete difference quotients. In essence, the spatial and temporal domains of a PDE are overlaid with a grid of points, and the derivatives at each point are approximated by algebraic combinations of neighbouring values. This process transforms the original problem into a system of linear (or non-linear) equations that can be

efficiently marched forward in time or solved iteratively until they converge on a steady state.

The difference quotients can be derived from Taylor's theorem. First, we expand the function $f(x)$ as:

$$f(x_0 + h) = f(x_0) + \frac{f'(x_0)}{1!}h + \frac{f^{(2)}(x_0)}{2!}h^2 + \cdots + \frac{f^{(n)}(x_0)}{n!}h^n + R_n(x), \quad (2.5)$$

where $R_n(x)$ is a remainder term. Now, we can approximate the first derivative as:

$$f'(x_0) = \frac{f(x_0 + h) - f(x_0)}{h} - \frac{R_1(x)}{h}. \quad (2.6)$$

The truncation error, $R_1(x)/h$, typically scales with h (first-order) but, by combining Taylor expansions at points $x_0 \pm h$, one can derive central-difference schemes with $\mathcal{O}(h^2)$ accuracy. Similar definitions extend to higher derivatives and multiple dimensions. In practice, we need to consider consistency, stability and convergence when using FDMs. Thus, we have different choice of stencil (e.g., forward, backward or central differences) shown in Fig 2.1 and time-stepping scheme (e.g., explicit, implicit or Crank-Nicolson) shown in Fig 2.2.

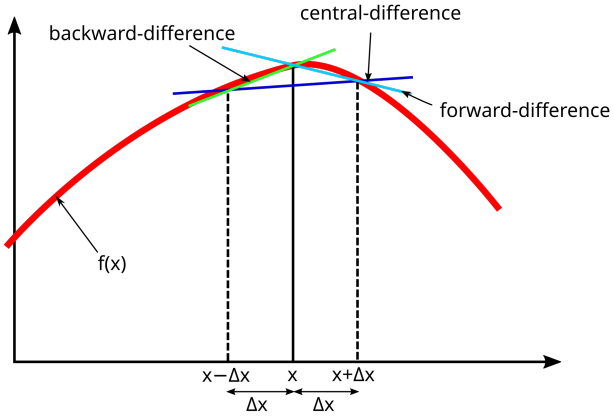
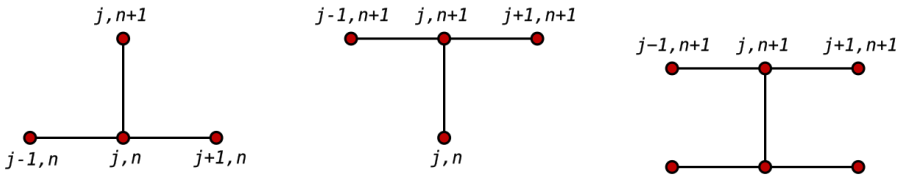


Figure 2.1: Finite-difference approximations at $x = 0$: forward, backward, and central differences [7].

Building on the Taylor-expansion derivation and our choice of difference approximations, finite-difference methods remain conceptually straightforward and trivial to implement on structured (Cartesian) grids. By widening the stencil, for example, moving from a three-point to a five-point, seven-point, or even compact scheme, the formal order of spatial accuracy can be raised (for instance,



(a) Explicit (forward Euler)

(b) Implicit (backward Euler)

(c) Crank-Nicolson

Figure 2.2: Comparison of time-stepping schemes: explicit, implicit, and Crank-Nicolson [7].

from second- to fourth- or sixth-order) without altering the mesh itself.

However, that very simplicity becomes a liability on complex or unstructured domains. To fit an irregular boundary into a Cartesian layout, one must introduce coordinate transformations or ghost-point interpolations, which, while practical, often erode the scheme's formal convergence rate and can undermine stability when enforcing sophisticated boundary conditions.

2.2.2 Finite Element Method

The finite element method (FEM) is a versatile numerical technique for solving boundary-value problems governed by partial differential equations (PDEs). In FEM, the continuous domain Ω is discretized into a finite number of simple geometric subdomains called elements, each defined by a set of nodes $\{x_i\}_{i=1}^{n_e}$ and associated shape functions $\{N_i(x)\}_{i=1}^{n_e}$. Figure 2.3 combines a 2D mesh illustration and a 1D hat-function plot to clarify both the discretization process and the interpolation mechanism.

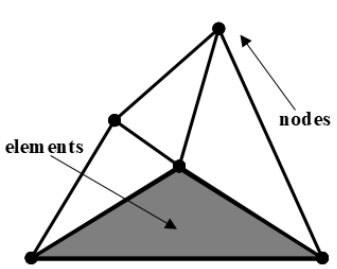
Within each element e, the unknown field $u(x)$ is interpolated by the nodal values u_i using the shape functions

$$u_h(x)\big|_e = \sum_{i=1}^{n_e} N_i(x)\, u_i, \tag{2.7}$$

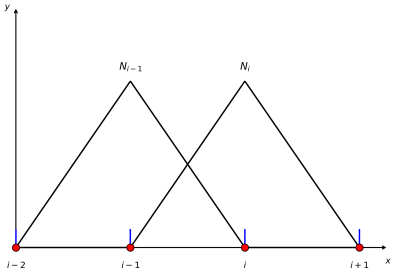
where $N_i(x)$ are typically low-order polynomials (e.g., linear or quadratic) chosen to balance accuracy against computational cost.

Starting from the strong form of a governing PDE, for example, the linear elasticity equilibrium equation

$$-\nabla\cdot\big(\mathbf{C} : \nabla u\big) = f \quad \text{in } \Omega, \tag{2.8}$$



(a) 2D triangular mesh showing elements and nodes.



(b) 1D linear shape functions $N_{i-1}(x)$ and $N_i(x)$ over a uniform mesh.

Figure 2.3: (a) Discretization of domain Ω into finite elements; (b) Linear hat functions illustrating local interpolation.

one derives the weak (variational) form by multiplying with a test function v, integrating over Ω, and integrating by parts. Enforcing natural boundary conditions on Γ_N yields

$$\int_\Omega (\nabla v)^\mathsf{T}\, \mathbf{C}\, \nabla u_h \, \mathrm{d}\Omega = \int_\Omega v\, f \, \mathrm{d}\Omega + \int_{\Gamma_N} v\, \bar{t} \, \mathrm{d}\Gamma \quad \forall\, v \in V, \tag{2.9}$$

where V is the space of admissible test functions satisfying essential boundary conditions on Γ_D.

Each element contributes a local stiffness matrix $\mathbf{K}^e$ and load vector $\mathbf{F}^e$, which assemble into the global system

$$\mathbf{K}\,\mathbf{U} = \mathbf{F}. \tag{2.10}$$

Here, $\mathbf{K} \in \mathbb{R}^{N\times N}$ is a sparse, symmetric, positive-definite matrix, $\mathbf{U}$ collects all nodal unknowns, and $\mathbf{F}$ is the global load vector. Efficient solution of this system typically employs direct solvers (e.g., LU or Cholesky factorization) or iterative solvers (e.g., conjugate gradient, generalized minimal residual method (GMRES) with preconditioning).

FEM excels at handling complex geometries, heterogeneous materials, and varied boundary conditions. Adaptive mesh refinement enhances accuracy by refining elements where solution gradients are high, while parallel solvers, multigrid techniques, and model-order reduction mitigate computational cost.

2.2.3 Spectral Methods

Spectral methods solve boundary-value problems by approximating the unknown field $u(x)$ with a truncated global expansion

$$u_N(x) = \sum_{k=0}^{N} \hat{u}_k \, \phi_k(x), \tag{2.11}$$

where the basis functions $\{\phi_k\}$ are chosen to be orthogonal over the domain, such as Fourier modes on periodic intervals or Chebyshev polynomials on $[-1, 1]$. Two main variants exist: the Galerkin spectral approach, which enforces orthogonality of the residual to the basis functions, and the pseudospectral (collocation) approach, which requires the PDE residual to vanish at a set of collocation points. By exploiting the global support of the basis, spectral methods achieve exponential (spectral) convergence for sufficiently smooth solutions: the error decays faster than any power of N. Computational efficiency is obtained via the fast Fourier transform for Fourier bases or via fast transforms for orthogonal polynomials.

Despite their high accuracy per degree of freedom, classical spectral methods are naturally restricted to simple geometries and uniform parameterizations. To extend spectral accuracy to complex domains, spectral element methods partition the domain into elements and apply high-order polynomial expansions within each element, combining the strengths of FEM and spectral techniques.

2.2.4 Strengths and Limitations of Classical Methods

Classical numerical schemes for PDEs, including finite-difference, finite-element, finite-volume, and spectral methods, are underpinned by well-established theoretical foundations. Their convergence and stability properties are rigorously characterized, and reliable error estimates guide mesh refinement or basis-order selection. A mature software ecosystem (for example, FEniCS, deal.II, and PETSc) offers production-ready implementations, powerful meshing tools, and optimized solvers tailored for large-scale sparse systems, making these methods the default choice in many engineering and physical applications.

However, several intrinsic challenges persist. The curse of dimensionality drives an exponential increase in degrees of freedom in high-dimensional problems, rapidly inflating computational cost and memory demands. Generating and adapting meshes becomes increasingly complex for evolving or highly irregu-

lar geometries, often requiring manual intervention or sophisticated algorithms. Furthermore, integrating observational data and solving inverse problems typically necessitates adjunct frameworks, such as adjoint-state methods, which add algorithmic and implementation complexity.

2.3 Motivation for Neural-Based Solvers

The limitations of classical numerical methods for solving partial differential equations (discussed above) have driven the exploration of other approaches. Neural network-based solvers offer a paradigm shift by providing mesh-free approximations that can efficiently handle these complexities, motivated by the need for faster, more adaptable, and data-efficient solutions in fields like fluid dynamics, quantum mechanics, and biomedical engineering.

The first motivation came from the universal approximation theorem [5], which posits that the neural networks can represent any continuous function to arbitrary accuracy, enabling them to parametrize PDE solutions directly without explicit discretization. This is good for high dimensional PDEs, where classical methods suffer exponential growth in computational cost. As deep learning technique and computing hardware advance, neural network solutions also improve.

Another key driver is the ability to integrate physics and data addressing scenarios where traditional methods falter, such as inverse problems (e.g., parameter estimation from sparse measurements) or multi-physics simulations involving coupled equations. Neural solvers can learn from both simulated and real-world data, improving robustness and predictive accuracy for nonlinear or stiff PDEs.

In summary, the motivation for neural network-based PDE solvers lies in overcoming the bottlenecks of classical methods through flexibility, efficiency, and physics-data fusion, paving the way for broader applications in scientific computing and engineering.

2.4 Primer on Neural Networks

Feedforward neural networks (FNNs) provide the fundamental function approximation framework for PDE solvers, while automatic differentiation enables efficient and exact computation of the derivatives required by PDE residuals and variational formulations.

A feedforward neural network (FNN) defines a parametric mapping

$$\mathbf{y} = f_\theta(\mathbf{x}), \quad \theta = \{W^{(l)}, \mathbf{b}^{(l)}\}_{l=1}^{L}, \tag{2.12}$$

where $\mathbf{x} \in \mathbb{R}^{d_{\text{in}}}$ is the input vector, $\mathbf{y} \in \mathbb{R}^{d_{\text{out}}}$ is the output vector, and θ denotes all trainable weights and biases. The mapping is constructed as a sequence of affine transformations and nonlinear activations:

$$\begin{aligned} \mathbf{h}^{(l)} &= \sigma\Big(W^{(l)}\mathbf{h}^{(l-1)} + \mathbf{b}^{(l)}\Big), \quad l = 1, \dots, L-1, \\ \mathbf{y} &= W^{(L)}\mathbf{h}^{(L-1)} + \mathbf{b}^{(L)}, \end{aligned} \tag{2.13}$$

with $\mathbf{h}^{(0)} = \mathbf{x}$ and $\sigma(\cdot)$ a nonlinear activation function (e.g., tanh, sin, or the rectified linear unit function, abbreviated ReLU). The universal approximation theorem [5] states that, given a sufficiently large number of neurons and a non-polynomial activation function, f_θ can approximate any continuous function on a compact domain to arbitrary accuracy. This makes FNNs natural candidates for representing the solutions $u(\mathbf{x})$ of PDEs, where $\mathbf{x}$ may include spatial and temporal coordinates.

A central requirement for neural network-based PDE solvers is the ability to evaluate derivatives of the network output $u_\theta(\mathbf{x})$ with respect to its input variables. For example, in a PDE of the form

$$\mathcal{N}\big[u(\mathbf{x})\big] = 0, \quad \mathbf{x} \in \Omega, \tag{2.14}$$

the operator $\mathcal{N}$ may involve first- or higher-order derivatives such as $\frac{\partial u}{\partial x_i}$ or $\frac{\partial^2 u}{\partial x_i \partial x_j}$. Computing these derivatives accurately is essential for forming the PDE residual.

Modern deep learning frameworks implement automatic differentiation (AD), which applies the chain rule of calculus to computational graphs. If the network output is given by a composition of elementary operations

$$u_\theta(\mathbf{x}) = (f_L \circ f_{L-1} \circ \cdots \circ f_1)(\mathbf{x}), \tag{2.15}$$

then the gradient, with respect to either the parameters θ or the inputs $\mathbf{x}$, can be computed exactly (up to machine precision) by propagating derivatives backward through the graph:

$$\frac{\partial u_\theta}{\partial \mathbf{x}} = \frac{\partial f_L}{\partial f_{L-1}} \cdot \frac{\partial f_{L-1}}{\partial f_{L-2}} \cdots \frac{\partial f_1}{\partial \mathbf{x}}. \tag{2.16}$$

This process, known as reverse-mode AD, has computational complexity comparable to a forward evaluation of the network, making it highly efficient.

For PDE learning, AD enables direct incorporation of the governing equations into the training loss. For instance, for the one-dimensional, time-independent Schrödinger equation

$$-\frac{\hbar^2}{2m}\frac{\mathrm{d}^2u}{\mathrm{d}x^2} + V(x)u(x) - Eu(x) = 0, \tag{2.17}$$

the second derivative $\frac{\mathrm{d}^2u_\theta}{\mathrm{d}x^2}$ can be computed exactly via AD, allowing the residual

$$\mathcal{R}(x) = -\frac{\hbar^2}{2m}u''_\theta(x) + V(x)u_\theta(x) - Eu_\theta(x) \tag{2.18}$$

to be embedded directly in the loss function. This capability eliminates the need for symbolic differentiation or numerical finite differences, both of which can be cumbersome or inaccurate for high-dimensional problems. However, it requires the activation function to be sufficiently differentiable. For example, ReLU would not be suitable for second-order PDEs due to its non-differentiability at zero.

In summary, feedforward neural networks offer a flexible function representation, and automatic differentiation provides an exact and efficient means of evaluating the derivatives required by PDE residuals or variational forms. Together, these properties form the computational backbone of modern neural-network-based PDE solvers such as physics-informed neural networks (PINNs), the deep Ritz method (DRM), and weak adversarial networks (WANs).

2.5 A Broad Survey of Neural PDE Methods

The idea of representing the solution to partial differential equations (PDEs) using neural network trial functions can be traced back to the 1990s, when Dissanayake and Phan-Thien first proposed a neural trial-function approach for PDEs. However, due to limitations in computational hardware and neural network theory at the time, these methods were not widely adopted. The advent of modern deep learning, particularly the development of automatic differentiation frameworks, revolutionized this landscape by enabling the efficient computation of PDE residuals directly within neural network training pipelines. This breakthrough made it possible to embed the governing equations of physical systems directly into the loss function, leading to the rapid development of neural

network-based PDE solvers.

In the following, we review recent neural network-based methods for solving PDEs, focusing on three major frameworks, physics-informed neural networks (PINNs), the deep Ritz method (DRM), and weak adversarial networks (WANs), before highlighting several other significant developments in the past five years.

2.5.1 Physics-Informed Neural Networks

PINNs have emerged as a transformative approach in scientific computing, offering a mesh-free and flexible framework for solving both forward and inverse PDE problems. Introduced in their modern form by Raissi, Perdikaris, and Karniadakis [22], PINNs employ a deep feedforward neural network trained to minimize a composite loss function that includes the PDE residual and any relevant boundary or initial condition errors. This ensures that the learned surrogate model not only fits observational data (if available) but also strictly adheres to the governing physical laws.

The general PINN workflow involves sampling collocation points in the domain, evaluating the PDE residual at these points via automatic differentiation, and adjusting the network weights to minimize residuals and boundary mismatches. The same framework extends naturally to inverse problems by treating unknown parameters as additional trainable variables. PINNs have been successfully applied to diverse fields including fluid dynamics, solid mechanics, and heat transfer, often achieving high accuracy with sparse data.

Despite their versatility, vanilla PINNs can suffer from slow or unstable convergence, particularly for stiff PDEs or when the loss terms (PDE residuals vs. boundary conditions) have competing scales. In response, several extensions have been proposed. One line of work focuses on domain decomposition and parallelization, exemplified by extended PINNs (XPINNs) [6], which generalize PINNs to a space–time domain decomposition using multiple subnetworks trained in parallel on smaller subdomains to reduce learning complexity and improve scalability. Another set of improvements targets adaptive sampling and loss balancing, including methods that dynamically adjust collocation points and loss weights to stabilize training. Probabilistic formulations, such as Bayesian PINNs (BPINNs) [27], incorporate Bayesian inference to provide uncertainty estimates over the predicted solution. Variational PINNs (VPINNs) [9], reformulate the PDE in variational form, minimizing an energy functional rather than the raw residual. Architectural variants have also been developed, including

physics-informed convolutional and recurrent networks for spatio-temporal PDEs. Hybrid pre-processing approaches, such as the physics-informed Gaussian (PIG) model [8], combine feature embeddings using Gaussian functions with PINN, improving accuracy in data-sparse regions.

2.5.2 The Deep Ritz Method

When a PDE admits a variational (energy) formulation, the deep Ritz method introduced by E and Yu [2] offers a compelling alternative. Instead of enforcing the PDE's differential form, the deep Ritz method minimizes an energy functional whose minimizer is the PDE solution. This approach naturally accommodates nonlinear PDEs, adapts to regions of high solution complexity, and scales well to high-dimensional problems using Monte Carlo integration for functional evaluation. The variational framework has inspired further developments, such as the deep double Ritz method [25], which introduces a nested optimization in which one network approximates the trial solution, while the other optimizes the test functions in an inner loop, enhancing accuracy for complex variational problems.

2.5.3 Weak Adversarial Networks

Weak adversarial networks, proposed by Zang et al. [29], reformulate PDEs in their weak form and solve them via adversarial optimization. Two networks are trained simultaneously: a primal network $u_\theta(x)$ approximates the PDE solution, while an adversarial test network $v_\phi(x)$ seeks functions that maximize the residual of the weak form. This setup creates a min-max game analogous to generative adversarial networks (GANs), driving the solution network to satisfy the PDE for all admissible test functions. WANs are particularly well-suited for PDEs where strong-form residuals are difficult to compute or unstable.

2.5.4 Other Neural Network PDE Solvers

Several other architectures expand beyond these three frameworks. The deep Galerkin method (DGM), introduced by Sirignano and Spiliopoulos [23], targets high-dimensional PDEs (up to 100–200 dimensions) by directly sampling and minimizing residuals without a mesh. Another rapidly growing area is operator learning, where the goal is to learn the entire PDE solution operator from data. Notable examples include DeepONet [16] and the Fourier neural operator

(FNO) [13], the latter operating in the frequency domain to efficiently learn solution mappings for families of PDEs.

2.5.5 Applications to Schrödinger Equations

Neural network-based PDE solvers have found significant application in wave dynamics and quantum systems, particularly involving Schrödinger-type equations. Pu *et al.* [20] developed an enhanced PINN with neuron-wise locally adaptive activation functions for the derivative nonlinear Schrödinger equation (DNLS), achieving accelerated convergence and improved fidelity in simulating soliton and rogue wave solutions. Wang *et al.* [26] extended multi-layer PINNs to recover data-driven rogue wave solutions of the defocusing NLS equation, demonstrating the method's ability to capture parameter-sensitive nonlinear phenomena. Yuan *et al.* [28] applied PINNs to modified NLS models, highlighting their adaptability to varying governing equations. Harcombe *et al.* [4] further showcased PINNs in identifying localized eigenstates in disordered media, extending their utility beyond conventional wave systems.

Collectively, these studies underscore the rapid evolution of neural network-based PDE solvers, from foundational PINNs to variational and adversarial formulations, and from general-purpose frameworks to specialized adaptations for complex wave phenomena, marking a vibrant and expanding frontier in computational physics and applied mathematics.

2.6 Summary

In this chapter, we have reviewed the fundamental concepts of partial differential equations (PDEs) and neural networks, followed by an overview of classical numerical methods for solving PDEs and the emerging class of neural-network-based approaches. Particular emphasis was placed on three prominent frameworks: physics-informed neural networks (PINNs), the deep Ritz method (DRM), and weak adversarial networks (WANs), along with their recent developments and applications.

In the subsequent chapters, we will systematically compare the performance of PINNs, DRM, and WANs to both the Schrödinger equation and selected benchmark problems, with a focus on predicting higher-order wavefunctions and the treatment of higher-dimensional cases. Detailed mathematical descriptions of these methods are provided in Chapter 3. Finally, we will apply PINNs,

DRM, and WANs to a novel potential in strong-field laser physics, the Kramers-Henneberger (KH) potential, and assess their performance in this context.

Chapter 3

Methodology

This chapter presents the methodological framework used to implement and compare three neural network based solvers for partial differential equations (PDEs): physics informed neural networks (PINNs), weak adversarial networks (WANs), and the deep Ritz method (DRM). The study focuses on two canonical PDE families, the Poisson equation and the time independent Schrödinger equation with representative potentials (infinite well and harmonic oscillator). The goal is to establish a common basis for fair comparison of accuracy, stability, and efficiency.

We first formalize each problem class by specifying the governing equation, the spatial domain Ω, and the boundary conditions. Where analytic references exist, we provide exact solutions to anchor quantitative evaluation. We then introduce the three solvers in a unified notation, derive their training objectives, and outline the general workflow for each method, including sampling and constraint handling. Finally, we describe the implementation and evaluation protocol, covering the computing platform and software stack, the network architectures and training schedules, the hyperparameters used across tasks, and the metrics reported, such as the relative L_2 error on holdout grids, boundary residuals, and wall clock time.

3.1 Problem Setups

To fully examine the capabilities of different neural network–based methods, we select a set of benchmark PDEs that are both well understood and representa-

tive of common challenges. We focus on the Poisson equation as a canonical elliptic problem, and on the time-independent Schrödinger equation with several potentials to test eigenvalue and normalization constraints. Wherever possible, exact solutions are provided for quantitative comparison.

3.1.1 Poisson's Equation

Poisson's equation is a canonical linear elliptic PDE that arises in electrostatics, Newtonian gravitation, and many other physical settings. Its general form, in d spatial dimensions, is

$$-\Delta u(\mathbf{x}) = f(\mathbf{x}), \qquad \mathbf{x} \in \Omega \subset \mathbb{R}^d, \tag{3.1}$$

where $u : \Omega \to \mathbb{R}$ is the unknown field, $f : \Omega \to \mathbb{R}$ is a prescribed source term, and Ω is a bounded domain with boundary $\partial\Omega$. In this work, we focus on homogeneous Dirichlet boundary conditions:

$$u(\mathbf{x}) = 0, \quad \mathbf{x} \in \partial\Omega. \tag{3.2}$$

As an analytic benchmark, let $\Omega = (0,1)^d$ and choose

$$f(\mathbf{x}) = \prod_{i=1}^{d} \sin(\pi x_i), \qquad \mathbf{x} = (x_1, \ldots, x_d). \tag{3.3}$$

Then the boundary-value problem

$$\begin{cases} -\Delta u(\mathbf{x}) = \displaystyle\prod_{i=1}^{d} \sin(\pi x_i), & \mathbf{x} \in (0,1)^d, \\ u(\mathbf{x}) = 0, & \mathbf{x} \in \partial(0,1)^d, \end{cases} \tag{3.4}$$

has the exact solution

$$u(\mathbf{x}) = \frac{1}{d\,\pi^2} \prod_{i=1}^{d} \sin(\pi x_i). \tag{3.5}$$

Indeed, since $\Delta\Big(\prod_{i=1}^{d} \sin(\pi x_i)\Big) = -d\,\pi^2 \prod_{i=1}^{d} \sin(\pi x_i)$ and the product vanishes on $\partial(0,1)^d$, the above u satisfies both the PDE and the boundary conditions. The familiar one- and two-dimensional cases are recovered by setting $d = 1$ and

$d = 2$:

$$\begin{aligned} u(x) &= \frac{1}{\pi^2}\sin(\pi x), \\ u(x,y) &= \frac{1}{2\pi^2}\sin(\pi x)\sin(\pi y). \end{aligned} \tag{3.6}$$

If $\Omega = (0, L)^d$, an equally convenient choice is

$$f(\mathbf{x}) = \prod_{i=1}^{d} \sin\Big(\frac{\pi x_i}{L}\Big), \tag{3.7}$$

for which the unique solution with $u|_{\partial\Omega} = 0$ is

$$u(\mathbf{x}) = \frac{L^2}{d\,\pi^2} \prod_{i=1}^{d} \sin\Big(\frac{\pi x_i}{L}\Big). \tag{3.8}$$

This construction extends naturally to any spatial dimensionality $d \in \mathbb{N}$. In our experiments, we will investigate from $d = 1$ to $d = 5$.

3.1.2 Time-Independent Schrödinger equation

The time-independent Schrödinger equation governs stationary states of nonrelativistic quantum systems and is widely used across physics and chemistry. In d spatial dimensions on a domain $\Omega \subset \mathbb{R}^d$, it reads

$$-\frac{\hbar^2}{2m}\,\Delta\psi(\mathbf{x}) + V(\mathbf{x})\,\psi(\mathbf{x}) = E\,\psi(\mathbf{x}), \qquad \mathbf{x} \in \Omega, \tag{3.9}$$

where $\psi(\mathbf{x})$ is the wavefunction, $V(\mathbf{x})$ the external potential, and E the associated energy. For bound states, the Born interpretation requires normalization:

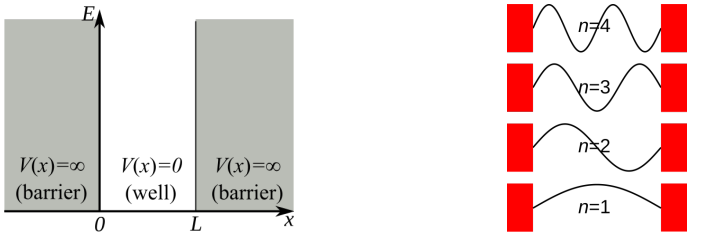
$$\int_\Omega |\psi(\mathbf{x})|^2\, d\mathbf{x} = 1, \tag{3.10}$$

while boundary conditions are dictated by the physics of the problem. For example, $\psi(\mathbf{x}) \to 0$ as $\|\mathbf{x}\| \to \infty$ on unbounded domains, or homogeneous Dirichlet conditions on impenetrable walls.

In what follows, we compute ground and excited stationary states for two canonical benchmark potentials: the infinite square well and the harmonic oscillator.

Infinite Potential Well

The infinite potential well is an archetypal and exactly solvable model in quantum mechanics: a particle is confined to a finite region with zero potential inside and an infinite barrier outside (see Figure 3.1). The wavefunction, therefore, satisfies homogeneous Dirichlet boundary conditions (i.e. it vanishes on the walls). Because closed-form solutions are available, the model is ideal for validating numerical methods while isolating fundamental effects such as energy quantization.



(a) Infinite square-well potential in 1D. (b) wavefunctions $\psi_n(x)$.

Figure 3.1: Infinite square well and representative stationary states [19].

In one dimension, on the interval $\Omega = (0, L)$, the time-independent Schrödinger equation with boundary conditions is

$$\begin{cases} -\dfrac{\hbar^2}{2m}\dfrac{d^2\psi}{dx^2} = E\,\psi, & x \in (0, L), \\ \psi(0) = \psi(L) = 0. \end{cases} \tag{3.11}$$

The normalized stationary states and their energies are

$$\psi_n(x) = \sqrt{\frac{2}{L}}\sin\!\Big(\frac{n\pi x}{L}\Big), \quad E_n = \frac{\hbar^2\pi^2}{2mL^2}\,n^2, \quad n = 1, 2, \ldots. \tag{3.12}$$

In two dimensions, for a rectangular box $\Omega = (0, L_x) \times (0, L_y)$, the problem reads

$$\begin{cases} -\dfrac{\hbar^2}{2m}\left(\partial_{xx} + \partial_{yy}\right)\psi(x, y) = E\,\psi(x, y), & (x, y) \in \Omega, \\ \psi = 0, & \text{on } \partial\Omega. \end{cases} \tag{3.13}$$

Following separation of variables, the normalized solutions and corresponding

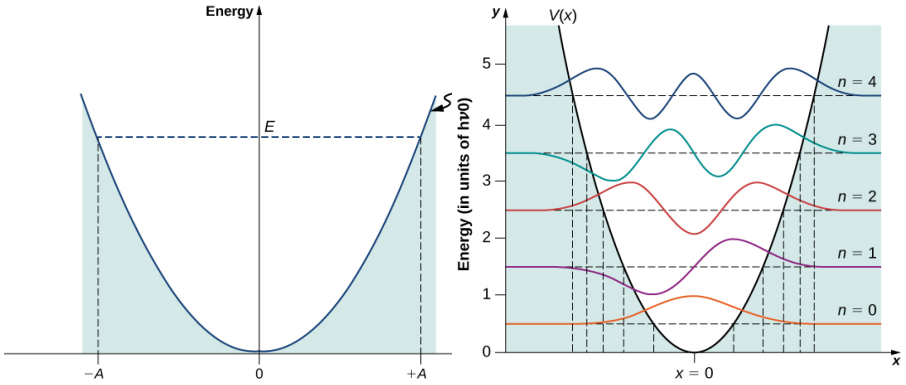
energies are

$$\psi_{n_x,n_y}(x,y) = \sqrt{\frac{2}{L_x}}\sqrt{\frac{2}{L_y}}\,\sin\!\left(\frac{n_x\pi x}{L_x}\right)\sin\!\left(\frac{n_y\pi y}{L_y}\right),$$
$$E_{n_x,n_y} = \frac{\hbar^2\pi^2}{2m}\left(\frac{n_x^2}{L_x^2}+\frac{n_y^2}{L_y^2}\right), \quad n_x, n_y \in \mathbb{N}. \tag{3.14}$$

Degeneracy arises naturally in two dimensions. For a square box ($L_x = L_y$), the spectrum depends only on $n_x^2 + n_y^2$; for example, $(n_x, n_y) = (1, 2)$ and $(2, 1)$ share the same energy.

Quantum Harmonic Oscillator

The quantum harmonic oscillator is the canonical quantum analogue of the classical spring-mass system. It models, among many applications, the vibrational motion of atoms in crystals and molecular normal modes. The confining parabolic potential drives the wavefunction to decay at infinity, and closed-form solutions make the problem a reliable benchmark for validating numerical methods. A representative one-dimensional case, together with its potential and stationary states, is shown in Figure 3.2.



(a) Harmonic potential in 1D. (b) Wavefunctions $\psi_n(x)$.

Figure 3.2: Harmonic oscillator potential and representative stationary states [14].

In one dimension ($x \in \mathbb{R}$), the time-independent Schrödinger equation is

$$-\frac{\hbar^2}{2m}\frac{d^2\psi}{dx^2} + \frac{1}{2}m\omega^2 x^2\,\psi(x) = E\,\psi(x), \tag{3.15}$$

with normalizable solutions

$$\psi_n(x) = \frac{1}{\sqrt{2^n n!}} \left(\frac{m\omega}{\pi\hbar}\right)^{1/4} e^{-\frac{m\omega x^2}{2\hbar}} H_n\!\left(\sqrt{\frac{m\omega}{\hbar}}\, x\right), \quad n = 0, 1, 2, \ldots \tag{3.16}$$

and equally spaced energies

$$E_n = \hbar\omega\big(n + \tfrac{1}{2}\big)\,. \tag{3.17}$$

Here, H_n are the physicists' Hermite polynomials,

$$H_n(z) = (-1)^n e^{z^2} \frac{d^n}{dz^n}\left(e^{-z^2}\right). \tag{3.18}$$

Note that n starts at 0, and the ground-state energy $E_0 = \frac{1}{2}\hbar\omega$ reflects the zero-point motion mandated by the uncertainty principle.

In two dimensions ($x, y \in \mathbb{R}$), the Hamiltonian separates. For a (possibly anisotropic) oscillator with frequencies $\omega_x, \omega_y > 0$,

$$-\frac{\hbar^2}{2m}\left(\partial_{xx} + \partial_{yy}\right)\psi(x,y) + \frac{1}{2}m\big(\omega_x^2 x^2 + \omega_y^2 y^2\big)\,\psi(x,y) = E\,\psi(x,y). \tag{3.19}$$

Separable, normalized stationary states and their energies are

$$\begin{aligned}\psi_{n_x,n_y}(x,y) &= \frac{1}{\sqrt{2^{\,n_x+n_y}\, n_x!\, n_y!}} \left(\frac{m\sqrt{\omega_x\omega_y}}{\pi\hbar}\right)^{1/2} \exp\Big[-\frac{m}{2\hbar}\big(\omega_x x^2 + \omega_y y^2\big)\Big] \\ &\quad \times H_{n_x}\!\left(\sqrt{\frac{m\omega_x}{\hbar}}\, x\right) H_{n_y}\!\left(\sqrt{\frac{m\omega_y}{\hbar}}\, y\right), \\ E_{n_x,n_y} &= \hbar\omega_x\big(n_x + \tfrac{1}{2}\big) + \hbar\omega_y\big(n_y + \tfrac{1}{2}\big), \quad n_x, n_y \in \mathbb{N}_0.\end{aligned} \tag{3.20}$$

In the isotropic case ($\omega_x = \omega_y = \omega$), the spectrum depends only on $N = n_x + n_y$:

$$E_N = \hbar\omega\,(N+1), \quad N = 0, 1, 2, \ldots, \tag{3.21}$$

with degeneracy $N + 1$ arising from all pairs (n_x, n_y) such that $n_x + n_y = N$.

3.2 Neural Network-Based Methods

In following parts, we introduce three neural network based approaches for solving PDEs: physics–informed neural networks (PINNs), the deep Ritz method

(DRM), and weak adversarial networks (WANs). Each method employs a neural network Ansatz to approximate the solution while incorporating the underlying physical or variational structure of the PDE.

3.2.1 Physics–Informed Neural Networks

PINNs [22] embed the governing equations directly into the loss function. Let

$$\begin{cases} \mathcal{N}[u](\mathbf{x}) = 0, & \mathbf{x} \in \Omega, \\ u(\mathbf{x}) = g(\mathbf{x}), & \mathbf{x} \in \partial\Omega, \end{cases} \tag{3.22}$$

denote a general PDE with boundary data g. We approximate u by a neural network $u(\mathbf{x};\theta)$. Define three loss components:

$$\mathcal{L}_{\text{int}}(\theta) = \frac{1}{N_{\text{int}}} \sum_{i=1}^{N_{\text{int}}} \left| \mathcal{N}[u(\mathbf{x}_i^{\text{int}};\theta)] \right|^2, \tag{3.23}$$

$$\mathcal{L}_{\text{bc}}(\theta) = \frac{1}{N_{\text{bc}}} \sum_{i=1}^{N_{\text{bc}}} \left| u(\mathbf{x}_i^{\text{bc}};\theta) - g(\mathbf{x}_i^{\text{bc}}) \right|^2, \tag{3.24}$$

$$\mathcal{L}_{\text{data}}(\theta) = \frac{1}{N_d} \sum_{i=1}^{N_d} \left| u(\mathbf{x}_i^{d};\theta) - g(\mathbf{x}_i^{d}) \right|^2, \tag{3.25}$$

where $N_{\text{int}}, N_{\text{bc}}, N_d$ are the counts of the interior, boundary, and (optional) measurement points, respectively. The total loss is a weighted sum,

$$\mathcal{L}(\theta) = \lambda_{\text{int}}\, \mathcal{L}_{\text{int}} + \lambda_{\text{bc}}\, \mathcal{L}_{\text{bc}} + \lambda_d\, \mathcal{L}_{\text{data}}. \tag{3.26}$$

Tuning $\{\lambda\}$ involves balancing the enforcement of the PDE residual, boundary conditions, and any available data. Training proceeds by sampling collocation points in Ω and $\partial\Omega$, computing gradients via automatic differentiation, and optimizing θ with Adam [10] or L–BFGS [15].

3.2.2 Deep Ritz Method

The deep Ritz method [2] reformulates a self–adjoint PDE as the minimization of an energy functional. For example, the Poisson problem

$$-\Delta u = f \text{ in } \Omega, \quad u = 0 \text{ on } \partial\Omega \tag{3.27}$$

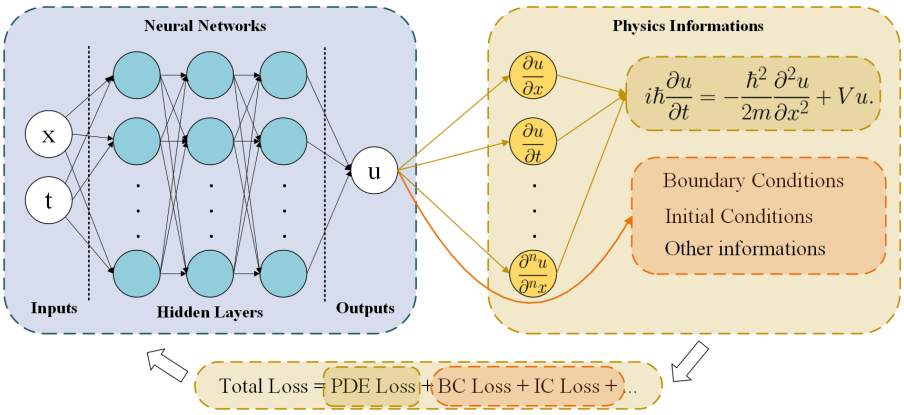


Figure 3.3: Workflow of a PINN solving the time–dependent Schrödinger equation.

has the variational form

$$\min_{u \in H_0^1(\Omega)} I[u] \text{ with } I[u] = \int_\Omega \Big(\tfrac{1}{2} |\nabla u|^2 - f\, u \Big)\, d\mathbf{x}. \tag{3.28}$$

We approximate $u(\mathbf{x};\theta)$ by a neural network and replace the integral by Monte Carlo sampling:

$$I_N(\theta) = \frac{|\Omega|}{N} \sum_{i=1}^{N} \Big(\tfrac{1}{2} |\nabla u(\mathbf{x}_i;\theta)|^2 - f(\mathbf{x}_i)\, u(\mathbf{x}_i;\theta) \Big). \tag{3.29}$$

Dirichlet conditions are enforced by adding a penalty term:

$$\widetilde{I}_N(\theta) = I_N(\theta) + \beta\, \frac{|\partial\Omega|}{N_\partial} \sum_{j=1}^{N_\partial} \big| u(\mathbf{x}_j^\partial;\theta) \big|^2, \tag{3.30}$$

with large β. Minimizing $\widetilde{I}_N$ recovers the weak solution.

For eigenvalue problems such as the time–independent Schrödinger equation, one minimizes the Rayleigh quotient,

$$E[\psi] = \frac{\displaystyle\int \Big(\frac{\hbar^2}{2m} |\nabla\psi|^2 + V|\psi|^2 \Big)\, d\mathbf{x}}{\displaystyle\int |\psi|^2\, d\mathbf{x}}, \tag{3.31}$$

subject to $\|\psi\|_{L^2} = 1$. In practice, one optimizes

$$\mathcal{L}(\theta) = J_N(\theta) + \lambda_{\text{norm}}\Big(\|\psi(\cdot;\theta)\|_{L^2}^2 - 1\Big)^2 + \beta \times \text{BC penalty}, \tag{3.32}$$

where J_N is the Monte Carlo-approximated numerator. Higher eigenstates impose additional orthogonality penalties $\int \psi\, \psi_k \, d\mathbf{x} = 0$.

3.2.3 Weak Adversarial Networks

WANs [29], illustrated in Figure 3.4, blend the weak formulation with an adversarial training paradigm. Starting from the bilinear form for a general PDE,

$$\langle \mathcal{A}[u], v\rangle = \int_\Omega \big(\nabla u \cdot \nabla v - f\, v\big)\, d\mathbf{x}, \tag{3.33}$$

A WAN combines two networks:

- a primal network $u(\mathbf{x};\theta)$ approximating the solution, and
- an adversarial network $v(\mathbf{x};\eta)$ serving as a test function.

The weak residual is

$$\mathcal{R}(\theta,\eta) = \big|\langle \mathcal{A}[u(\cdot;\theta)], v(\cdot;\eta)\rangle\big|^2. \tag{3.34}$$

Training solves the saddle–point problem

$$\min_\theta \max_\eta \; \lambda_{\text{weak}}\, \mathcal{R}(\theta,\eta) + \lambda_{\text{bc}}\, \mathcal{L}_{\text{bc}}(\theta), \tag{3.35}$$

where $\mathcal{L}_{\text{bc}}$ enforces boundary conditions as before. In each iteration, η is updated to maximize the weak residual (identifying the worst-case test function), and θ is updated to minimize it. This adversarial coupling enhances stability in high dimensions and complex geometries.

3.3 Implementation Details and Evaluation Protocol

3.3.1 Experimental Platform and Software

All experiments were conducted on a laptop equipped with an NVIDIA GeForce RTX 3080 GPU (running CUDA version 11.7) and an Intel Core i9 CPU, sup-

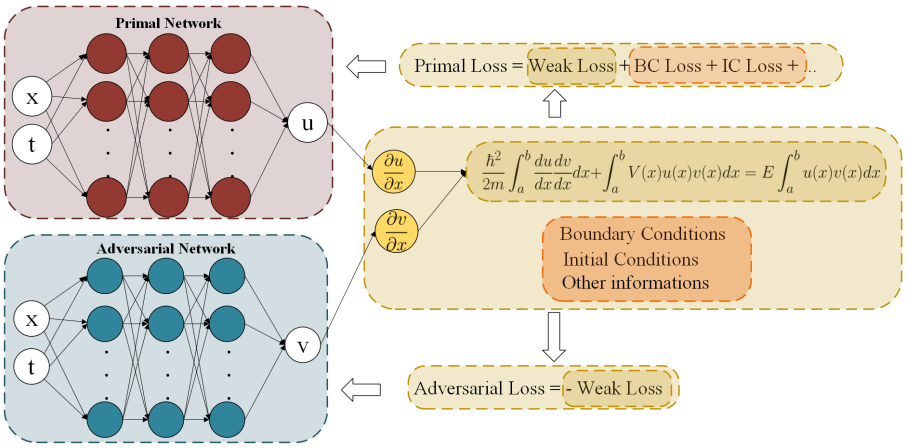


Figure 3.4: WAN architecture applied to the Schrödinger equation in its weak form.

ported by 32 GB of RAM. The software environment consisted of Python 3.9, PyTorch 1.12, and NumPy 1.22. To ensure reproducibility, random seeds for PyTorch, NumPy, and CUDA were fixed at runtime. The complete codebase, including the `requirement.txt` specification and scripts for data generation, training, and evaluation, is publicly available at `https://github.com/JiakangC/Neural-Network-Based-PDE-Solver.git`.

3.3.2 Model Architectures and Training

We evaluated a fully connected network (FCN) as our baseline model, comprising three hidden layers with 50 neurons each and employing hyperbolic tangent activation functions. All weights were initialized using Xavier initialization to promote stable training dynamics. Optimization was performed using the Adam optimizer with an initial learning rate of 10^{-3}, over a maximum of 10 000 to 50 000 epochs.

The loss function consisted of at least three primary terms: the PDE or weak-form residual, the boundary condition enforcement, and an optional data supervision term. The coefficients for these terms were carefully tuned within the range $[1, 10\,000]$ to balance their contributions effectively. Although the supervised data term is not essential for PINNs, our experiments demonstrated that incorporating even a modest amount of labelled data significantly enhanced solution accuracy. To systematically explore this effect, we trained the network

on the first 25% of the available data.

For enhanced performance, we further explored architectural modifications, including increasing the network depth (e.g., adding more hidden layers) and the number of neurons per layer. Additionally, we introduced custom sine activation functions, which have been shown to better capture oscillatory behaviours inherent in certain PDE solutions, such as those from quantum mechanics. To further strengthen the physics-informed aspects of the model, we incorporated additional loss terms designed to enforce specific physical constraints; these will be detailed below.

3.3.3 Evaluation Protocol

Model performance was primarily quantified using the relative L_2 error, defined as

$$\varepsilon_{L_2} = \frac{\|u_{\text{NN}} - u_{\text{exact}}\|_{L_2(\Omega)}}{\|u_{\text{exact}}\|_{L_2(\Omega)}}, \tag{3.36}$$

along with the training time required to achieve convergence. For problems involving the Schrödinger equation, we also report the eigenvalue error $|E_{\text{NN}} - E_{\text{exact}}|$, where applicable, to assess the accuracy of energy level predictions. Throughout the training process, the PDE residual norm was monitored to track convergence and ensure adherence to the governing equations.

Computational efficiency was characterized by recording the runtime to reach the best L_2 error and average duration per epoch, offering insights into the practical costs of the methods.

3.3.4 Numerical Experiments

We applied the proposed framework to the benchmark problems outlined in Section 3.1, beginning with Poisson equation tests, from one to five dimensions, to validate the basic setup, before advancing to quantum mechanical systems such as the infinite square well and the quantum harmonic oscillator. Our primary focus, however, was on the Schrödinger equations, where the physics-informed approaches demonstrate particular advantages in handling eigenvalue problems and boundary conditions.

As an initial testbed, we considered one-dimensional problems. For the one-dimensional infinite potential well, we adopted the foundational approach from Raissi et al. [22], balancing the PDE residual, boundary conditions, and optional data terms via tunable weight parameters, as described in Section 3.2.

This configuration served as our baseline and was denoted as "BC" (Boundary Condition weighting). To enhance boundary enforcement, we introduced a forced boundary condition directly into the neural network architecture; we label this variant "FB" (Forced Boundary). Recognizing the need for consistency across multiple energy states in eigenvalue problems, we proposed two extensions to handle higher modes effectively.

The first extension incorporated an orthogonality penalty, a technique commonly used to ensure that predicted eigenfunctions remain orthogonal, as required by quantum mechanics; we refer to this extension as "OG" (Orthogonality). The second novel approach, termed "FN" (Forced Nodes), involved strategically controlling the positions of nodal points (where the wave function vanishes) to partition the domain into smaller subregions with enforced zero boundaries. This decomposition simplifies the learning task by breaking it into more manageable segments. Detailed descriptions of these techniques, including their implementation and rationale, are presented alongside the results in the next chapter.

The same suite of techniques, BC, FB, OG, and FN, was subsequently applied to the one-dimensional quantum harmonic oscillator. In this problem, we introduced an additional trainable parameter for the energy eigenvalue, initialized with the exact value as a guess to guide the optimization process and improve convergence.

Finally, we extended these methodologies to higher-dimensional settings. Using FB as the baseline for boundary enforcement, we compared it against OG and FN for both the infinite potential well and the quantum harmonic oscillator in two dimensions. This progression allowed us to evaluate the scalability of the approaches, assessing how well they generalize to increased complexity while maintaining accuracy and efficiency.

Chapter 4

Results and Analysis for Standard Problems

This chapter presents the empirical findings from applying physics-informed neural networks (PINNs), deep Ritz method (DRM) and weak adversarial networks (WANs) to the baseline Poisson equation and standard Schrödinger problems (infinite potential well in 1D and 2D; quantum harmonic oscillator in 1D and 2D). To provide a comprehensive evaluation, results are divided into baseline implementations (standard setups without modifications) and optimized versions (incorporating improvements like additional loss terms, hard boundary conditions, and advanced architectures). This structure highlights initial limitations (e.g., highL_2errors in baselines due to trivial solutions or spectral bias) and demonstrates how targeted enhancements improve performance, enabling fair cross-method comparisons.

4.1 Baseline Results

4.1.1 Poisson Equation Baselines

We evaluate the efficacy of our method in solving the Poisson equation across multiple dimensions without relying on any training data. In the following table, we present the performance from one to five dimensions, considering two distinct boundary conditions: 'BC' for weight-controlled boundaries and 'FB' for forced boundaries. Table 4.1 summarizes these outcomes.

Table 4.1: Results for Multi-dimensional Poisson Equation

	Dim	1	2	3	4	5
PINN	BC	2.58E−03	1.87E−02	2.94E−02	3.75E−02	1.29E−01
	FB	4.11E−05	3.20E−05	6.27E−05	1.04E−04	8.87E−05
DRM	BC	4.66E−01	4.93E−01	3.47E−01	2.30E−01	1.70E−01
	FB	1.01E−04	2.11E−04	2.77E−04	3.78E−04	3.10E−04
WAN	BC	1.07E−01	1.93E−02	3.42E−02	3.86E−02	1.38E−01
	FB	1.12E−03	4.88E−04	5.62E−04	6.42E−04	8.01E−04

4.1.2 Schrödinger Equation Baselines

One-Dimensional Infinite Potential Well

To benchmark our approach against established baselines, we compare it on the one-dimensional infinite potential well, examining states from the ground state ($n = 1$) to the fourth excited state ($n = 5$). We use '-' to denote methods that fail to converge, defined as cases where the L_2 error exceeds 1×10^{-3}, while run times are recorded for achieving the best L_2 error. These comparisons are detailed in Table 4.2. In Figure 4.1, we illustrate the predicted wavefunction (Figure 4.1a) and loss evolution (Figure 4.1b) for each method.

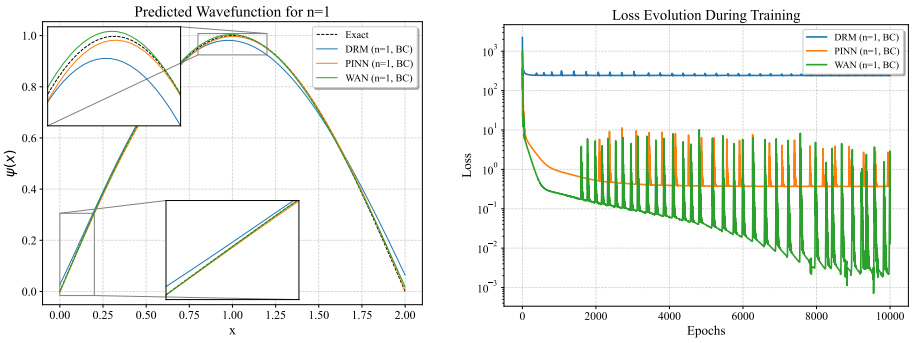
Table 4.2: Baseline Comparison for Infinite Potential Well

	PINN		DRM		WAN	
n	L_2 Error	Run Time	L_2 Error	Run Time	L_2 Error	Run Time
1	6.22E−05	21.6	4.62E−04	19.2	2.16E−05	112.8
2	7.79E−06	105.3	-	-	1.03E−05	343.5
3	5.90E−07	19.0	-	-	4.72E−06	350.3
4	5.18E−06	88.0	-	-	1.33E−04	204.3
5	1.92E−07	55.4	-	-	-	-

In Figure 4.1, we show the predicted wavefunction (Figure 4.1a) and loss evolution (Figure 4.1b) for different methods.

One-Dimensional Quantum Harmonic Oscillator

Similarly, we assess the baseline methods on the one-dimensional quantum harmonic oscillator, spanning states from the ground state ($n = 0$) to the fourth excited state ($n = 4$). As before, '-' indicates failure when the L_2 error is larger



(a) Predicted wavefunction for $n = 1$.

(b) Loss evolution during training.

Figure 4.1: Comparison of wavefunction predictions and loss for $n = 1$.

than 1×10^{-3}, with run times noted for the optimal L_2 error. The results are shown in Table 4.3. Figure 4.2 shows the predicted wavefunction (Figure 4.2a) and loss evolution (Figure 4.2b) for the various methods.

Table 4.3: Baseline Comparison for Quantum Harmonic Oscillation

	PINN		DRM		WAN	
n	L_2 Error	Run Time	L_2 Error	Run Time	L_2 Error	Run Time
0	1.30E−08	162.6	3.60E−08	67.1	3.84E−05	418.0
1	2.15E−07	175.4	-	-	2.33E−04	138.2
2	1.12E−07	178.2	-	-	2.34E−04	224.2
3	1.02E−07	198.0	-	-	-	-
4	1.96E−07	179.3	-	-	-	-

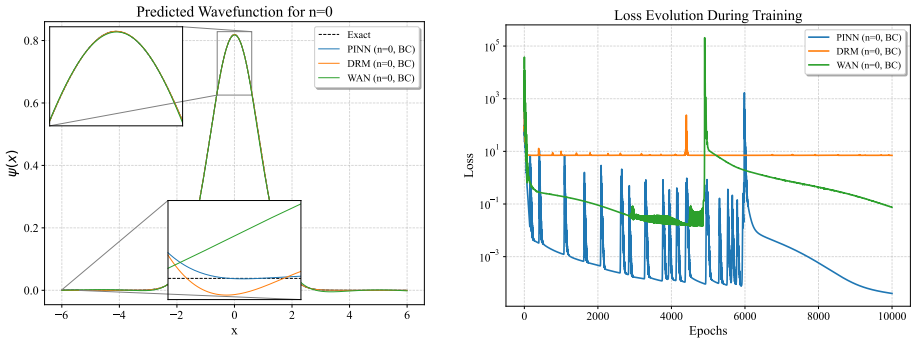
In Figure 4.2, we show the predicted wavefunction[1] (Figure 4.2a) and loss evolution (Figure 4.2b) for different methods.

4.2 Optimized Results and Improvements

4.2.1 Improvements on One-Dimensional Problems

To enhance consistency in fitting higher states, we incorporate forced boundary and forced node conditions. This enables the neural network to better identify the specific state being fitted. Table 4.4 presents the results for the one-dimensional

[1]Further plots can be found in the Appendix.



(a) Predicted wavefunction for $n = 1$.

(b) Loss evolution during training.

Figure 4.2: Comparison of wavefunction predictions and loss for $n = 0$.

infinite potential well under forced boundary (FB), orthogonality (OG), and forced node (FN) conditions. The model was trained sequentially from the ground state ($n = 1$) to the 5th excited state ($n = 5$), with the best L_2 error recorded over 50 000 epochs. Any L_2 errors exceeding 1×10^{-3} are denoted by '-', indicating failure to converge on the solution. Shaded gray cells highlight the best performance in each case. Note that ground state results are identical across methods, as their initial conditions remain unaffected; only higher-order states are influenced by the OG and FN conditions.

Table 4.4: Results for One-Dimensional Infinite Potential Well

		$n = 1$	$n = 2$	$n = 3$	$n = 4$	$n = 5$
PINN	FB	2.65E−06	9.34E−06	1.05E−06	1.47E−07	4.36E−08
	FN	2.65E−06	4.71E−07	5.24E−07	7.87E−08	1.78E−08
DRM	FB	1.84E−07	-	-	-	-
	OG	1.84E−07	3.92E−06	4.51E−06	5.72E−06	6.88E−06
	FN	1.84E−07	4.63E−07	2.66E−06	8.68E−05	9.94E−05
WAN	FB	8.79E−09	1.67E−06	2.11E−05	7.36E−05	-
	OG	8.79E−09	1.30E−05	2.38E−05	1.29E−05	3.75E−06
	FN	4.80E−09	6.88E−09	8.50E−09	2.56E−07	3.14E−05

Using the same setup, Table 4.5 presents the optimized results for the one-dimensional quantum harmonic oscillator over 50 000 epochs of training, starting from the ground state ($n = 0$) to the fourth excited state.

Table 4.5: Results for One-Dimensional Quantum Harmonic Oscillator

		$n=0$	$n=1$	$n=2$	$n=3$	$n=4$
PINN	FB	3.72E−08	1.19E−07	1.81E−08	1.42E−08	1.73E−08
	FN	1.19E−07	8.83E−07	6.06E−05	6.88E−04	-
DRM	FB	9.99E−09	-	-	-	-
	OG	9.99E−09	7.38E−08	8.97E−08	8.15E−08	8.11E−08
	FN	9.99E−09	2.18E−07	1.12E−05	1.94E−05	-
WAN	FB	1.93E−05	8.59E−05	6.20E−04	7.61E−04	-
	OG	1.93E−05	5.38E−05	9.37E−05	5.89E−05	2.39E−04
	FN	1.93E−05	8.36E−04	-	-	-

4.2.2 Extensions to High-Dimensionality Problems

Building on the optimization strategies applied in the one-dimensional case, we extend our approach to two-dimensional problems. Starting with the forced boundary condition, Table 4.6 shows the results for the two-dimensional infinite potential well. We investigate states ranging from the ground state (1,1) to (3,3), including mixed modes such as (2,1) and (3,1). Similarly, Table 4.7 displays the corresponding results for the two-dimensional quantum harmonic oscillator, starting from the ground state (0,0).

Table 4.6: Results for Two-Dimensional Infinite Potential Well

	PINN		DRM		WAN	
n	FB	FN	OG	FN	OG	FN
$(1,1)$	6.56E−10	6.56E−10	7.99E−06	7.99E−06	2.16E−05	6.61E−06
$(2,1)$	1.22E−08	2.83E−09	1.02E−05	4.85E−06	1.03E−05	4.85E−06
$(2,2)$	1.10E−07	7.23E−11	-	3.99E−06	4.72E−06	2.57E−05
$(3,1)$	2.04E−07	2.31E−09	-	6.88E−06	1.33E−04	1.24E−05
$(3,2)$	1.83E−06	4.92E−10	-	2.82E−05	-	9.35E−06
$(3,3)$	6.61E−05	1.39E−09	-	1.93E−05	-	1.31E−04

4.3 Sensitivity and Ablations

To demonstrate the data-free nature of our approach, we evaluate its performance in solving the time-independent Schrödinger equation without any training data points for both one-dimensional and two-dimensional systems, including the

Table 4.7: Results for Two-Dimensional Quantum Harmonic Oscillator

	PINN		DRM		WAN	
n	FB	FN	OG	FN	OG	FN
$(0,0)$	6.71E−07	6.71E−07	7.99E−06	7.99E−06	2.16E−05	6.61E−06
$(1,0)$	2.38E−06	3.12E−04	1.02E−05	4.85E−06	1.03E−05	4.85E−06
$(1,1)$	6.49E−05	2.69E−03	-	3.99E−06	4.72E−06	2.57E−05
$(2,0)$	2.09E−05	1.22E−03	-	6.88E−06	1.33E−04	1.24E−05
$(2,1)$	1.34E−04	-	-	2.82E−05	-	9.35E−06
$(2,2)$	9.51E−05	-	-	1.93E−05	-	1.31E−04

infinite potential well and quantum harmonic oscillator. The results encompass various eigenstates, highlighting the accuracy and convergence achieved solely through physics-informed constraints. These findings are summarized in Table 4.8.

Table 4.8: No-Data Training Results for 1D and 2D Schrödinger Systems

	IPW		QHO	
method	1D	2D	1D	2D
PINN	4.03E−04	2.45E−01	3.53E−08	2.20E−07
DRM	1.46E−07	2.55E−06	1.14E−08	1.04E−07
WAN	1.68E−06	1.47E−06	1.93E−05	1.04E−04

Furthermore, we conduct an ablation study to assess the sensitivity of our neural network architecture by varying the number of hidden layers from 1 to 4 and the number of neurons per layer among 10, 50, and 100. This analysis is performed on the one-dimensional infinite potential well at the fourth excited state ($n = 5$), selected due to its increased complexity compared to the ground state, which allows for a more rigorous evaluation of architectural impacts on optimization and error metrics. The outcomes, including L_2 errors, are presented in Table 4.9.

4.4 Summary

This chapter presents the empirical results from the experiments described in Chapter 3. Using a comprehensive evaluation protocol, we compare baseline and optimized implementations of PINN, DRM, and WAN on the Poisson equation

Table 4.9: Ablation Study on Network Architecture for 1D Infinite Potential Well ($n = 5$)

		1	2	3	4
PINN	10	3.98E−02	2.22E−02	2.90E−02	3.83E−02
	50	2.02E−04	3.28E−08	1.78E−08	9.93E−09
	100	2.02E−05	1.37E−08	9.42E−09	8.64E−09
DRM	10	4.77E−02	1.68E−01	3.59E−02	4.52E−02
	50	1.33E−01	3.29E−06	4.96E−07	1.33E−07
	100	6.34E−04	7.56E−07	1.34E−07	1.13E−07
WAN	10	1.47E−01	1.33E−02	5.74E−03	6.29E−04
	50	9.71E−06	1.43E−01	7.23E−07	4.06E−04
	100	3.39E−05	4.06E−04	2.27E−04	2.47E−03

and canonical Schrödinger problems (1D/2D infinite well, 1D/2D harmonic oscillator). Baselines expose common failure modes, such as trivial solutions, spectral bias, and elevated L_2 errors, while optimized variants incorporate targeted improvements (e.g., additional loss terms, hard boundary enforcement, and enhanced architectures) that markedly reduce these issues. Together, these results enable a fair, like-for-like comparison and clarify the strengths, limitations, and trade-offs of each method. A detailed analysis and discussion of implications is provided in Chapter 6.

Chapter 5

Schrödinger Equation with Kratzer-Hellmann Potential

The interaction of intense laser fields with quantum systems gives rise to rich dynamical behaviour, often requiring nonperturbative treatment. A particularly useful framework for addressing atoms or electrons in strong oscillatory fields is the Kramers-Henneberger (KH) transformation, which moves into a non-inertial frame co-moving with the classical quiver motion induced by the laser. In this chapter, we formulate the time-dependent Schrödinger equation with the KH potential and develop a neural network-based solver to approximate its solution. This chapter comprises: (i) theoretical background and derivation of the KH potential, (ii) a presentation of numerical experiments and results, and (iii) a summary. Comparison and broader discussion with other PDEs and methods will be deferred to Chapter 6.

5.1 Theoretical Background

When a quantum particle (e.g., an electron bound in an atom or molecule) is subjected to a strong, oscillatory electromagnetic field, such as intense laser light, the interaction can drive a large-amplitude quivering motion. In the laboratory frame, this manifests as a time-dependent forcing term that makes the Schrödinger equation rapidly oscillatory and, in strong-field regimes, nonperturbative. A powerful way to tame and reinterpret these dynamics is to move into a non-inertial frame that co-moves with the classical oscillatory displacement induced by

the laser. This is the essence of the Kramers-Henneberger (KH) transformation: instead of treating the laser as an external time-dependent field acting on a stationary binding potential, one views the binding potential as oscillating in time while the electron is 'stationary' in the moving frame. In many circumstances, this recasting leads to clearer physical intuition and simplifications such as effective potentials and stabilization phenomena.

5.1.1 Classical Quivering and Frame Change

Consider a free electron driven by a linearly polarized laser field in the dipole approximation. The classical equation of motion (ignoring magnetic and relativistic effects) is

$$m\ddot{\boldsymbol{\alpha}}(t) = -\mathbf{F}(t) \tag{5.1}$$

where $\mathbf{F}(t)$ is the electric field of the laser. Integrating twice gives the quivering displacement

$$\boldsymbol{\alpha}(t) = \frac{1}{m\omega^2} F_0 \sin(\omega t)\hat{\mathbf{e}}. \tag{5.2}$$

This is the classical oscillatory shift that a free electron undergoes under laser excitation. The KH transformation is a time-dependent unitary change of variables that shifts coordinates by this classical displacement. We define a new wavefunction in the KH frame:

$$\Psi(\mathbf{r}, t) = \exp\left[-\frac{i}{\hbar} m\dot{\boldsymbol{\alpha}}(t) \cdot \mathbf{r}\right] \Phi(\mathbf{r} + \boldsymbol{\alpha}(t), t). \tag{5.3}$$

In physical terms, this amounts to going into the accelerating frame following the quivering and applying the appropriate phase to maintain unitarity. Substituting this into the original time-dependent Schrödinger equation with a binding potential $V(\mathbf{r})$ and the laser field in the length gauge transforms the direct laser coupling into a time-dependent shifted potential:

$$i\hbar \frac{\partial}{\partial t} \Phi(\mathbf{r}, t) = \left[-\frac{\hbar^2}{2m}\nabla^2 + V(\mathbf{r} + \boldsymbol{\alpha}(t))\right] \Phi(\mathbf{r}, t) \tag{5.4}$$

That is, in the KH frame, the electron evolves under a potential that oscillates in time according to the classical quiver trajectory, known as the Kramers-Henneberger potential:

$$V_{\mathrm{KH}}(\mathbf{r}, t) := V(\mathbf{r} + \boldsymbol{\alpha}(t)). \tag{5.5}$$

The laser field no longer appears explicitly as a driving term; its effect is encoded in the oscillatory shift of the binding potential.

5.1.2 Cycle Averaging and Effective Potential

If the driving laser is periodic with angular frequency ω, i.e., $\boldsymbol{\alpha}(t+T) = \boldsymbol{\alpha}(t)$ with $T = 2\pi/\omega$, and the frequency is large compared to the intrinsic energy scales of the bound system, then the rapid oscillations of the KH potential can be smoothed out by averaging over one period. We define the cycle-averaged potential as

$$\bar{V}_{\mathrm{KH}}(\mathbf{r}) := \frac{1}{T}\int_0^T V(\mathbf{r}+\boldsymbol{\alpha}(t))\; dt. \tag{5.6}$$

Under this approximation, the time-dependent KH Schrödinger equation

$$i\hbar\frac{\partial}{\partial t}\Phi(\mathbf{r},t) = \left[-\frac{\hbar^2}{2m}\nabla^2 + V(\mathbf{r}+\boldsymbol{\alpha}(t))\right]\Phi(\mathbf{r},t) \tag{5.7}$$

is approximated by the time-independent eigenvalue equation

$$\left[-\frac{\hbar^2}{2m}\nabla^2 + \bar{V}_{\mathrm{KH}}(\mathbf{r})\right]\psi(\mathbf{r}) = E\psi(\mathbf{r}) \tag{5.8}$$

where E is interpreted as the leading-order quasi-energy of the dressed state. This approximation corresponds to keeping the zeroth-order term in a high-frequency (Floquet-Magnus or van Vleck) expansion of the effective Hamiltonian. Corrections to this are systematically computable as higher-order (in $1/\omega$) terms that reintroduce residual time dependence.

5.1.3 Problem Setup

We consider the short-range potential [24]

$$V(x) = V_0\,\frac{\exp\left[-\sqrt{x^2+16}\right]}{\sqrt{x^2+6.27^2}}, \tag{5.9}$$

with $V_0 = -24.856$ a.u.

Averaging over one driving period yields the cycle-averaged KH potential shown in Fig. 5.1. In our study, we solve the time-independent Schrödinger equation with $\alpha = 10$ for the ground state through to the third excited states. We then perform a parameter sweep over α.

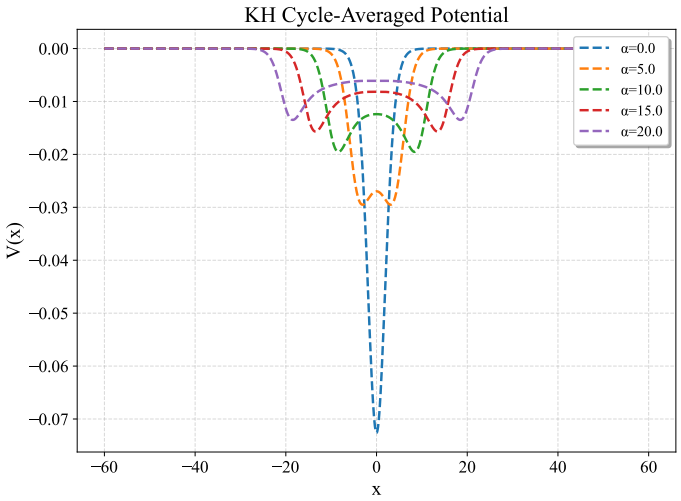


Figure 5.1: KH cycle-averaged potential.

For evaluation, we obtain reference ("ground-truth") solutions using a finite difference method.

5.2 Results

We begin with the best-performing configuration from prior tests on other Schrödinger problems and train for 10 000 epochs. Using 25% and 50% of the training data, we report results at $\alpha = 10$ for the ground to third excited states.

Table 5.1: Results for the one-dimensional KH Schrödinger equation.

	Data	$n = 0$	$n = 1$	$n = 2$	$n = 3$	Run time
PINN	25%	-	-	-	-	-
	50%	8.10E−09	2.67E−09	3.96E−09	8.27E−09	107
DRM	25%	1.47E−07	1.86E−08	1.37E−06	3.56E−08	101
	50%	1.74E−09	2.29E−09	4.51E−09	9.04E−09	102
WAN	25%	-	-	-	-	-
	50%	2.08E−08	9.16E−09	6.30E−09	1.34E−08	270

Using 50% of the training data, we next sweep α from 0 to 20 for the ground state only.

Figure 5.2 shows the predicted wavefunctions for $n = 0$ to $n = 3$. Because

Table 5.2: Ground-state results across different values of α.

	$\alpha = 0$	$\alpha = 5$	$\alpha = 10$	$\alpha = 15$	$\alpha = 20$	Run time
PINN	1.73E−08	5.86E−09	4.92E−09	4.72E−09	1.55E−08	114
DRM	4.02E−09	4.08E−09	8.00E−09	4.46E−09	3.69E−09	91
WAN	7.95E−08	2.04E−08	6.01E−09	5.66E−09	2.92E−09	460

all three methods achieve relatively small L_2 errors, small vertical offsets are applied to the curves for clearer visualisation.

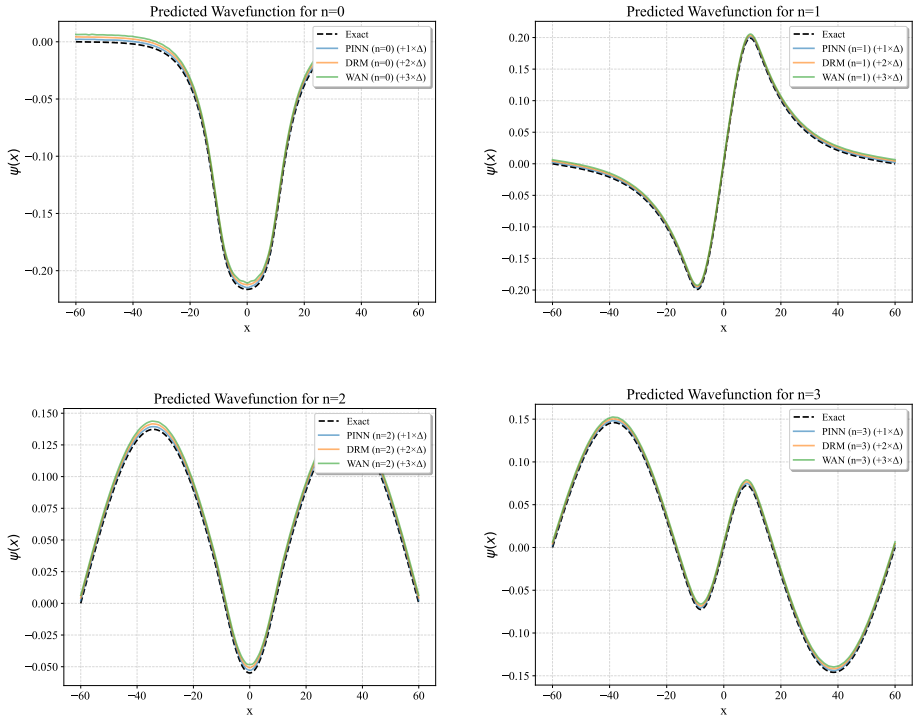


Figure 5.2: Predicted wavefunctions for $n = 0$ to $n = 3$ under the KH potential.

5.3 Summary

In this chapter, we introduced an end-to-end neural network solver for the time-dependent Schrödinger equation formulated in the Kramers-Henneberger

(KH) frame. Leveraging the KH transformation to move into a non-inertial frame co-moving with the laser-driven quivering, we derived the one-dimensional, cycle-averaged KH potential and cast the stationary problem into an eigenvalue-learning task suitable for neural approximation.

We instantiated and compared several neural PDE solvers, most notably physics-informed neural networks (PINNs), deep Ritz methods (DRM), and weak-adversarial approaches (WAN), under controlled training budgets.Solutions were validated against finite-difference reference data. At a representative coupling $\alpha = 10$, the solver accurately recovered the ground through third excited states; we further examined robustness via a sweep over $\alpha \in \{0, 5, 10, 15, 20\}$ for the ground state. Across these studies, the predicted wavefunctions matched the reference solutions with consistently small L_2 errors while maintaining practical training times. A broader comparison with other PDE families and solver classes, including discussions of scalability and generalisation, will be provided in Chapter 6.

Chapter 6

Discussion

This chapter synthesizes the results from Chapters 4 and 5 to assess three neural PDE solvers, physics-informed neural networks (PINNs), the deep Ritz method (DRM), and weak adversarial networks (WANs), on the benchmark problems defined in Chapters 3 and 5. Our objectives are to identify the strengths and limitations of each method across tasks, to explain failure modes and how targeted adjustments improve outcomes, and to provide practical guidance for choosing and configuring these methods for new problems. We begin with the Poisson equation and compare performance across spatial dimensions while examining training factors such as boundary-condition enforcement (soft penalties versus forced boundary conditions, FBC), sampling strategies, network architectures, and loss weighting. We then analyze the time-independent Schrödinger equation in one and two dimensions (infinite potential well and harmonic oscillator), interpreting the mechanisms behind the observed behaviours. Throughout, we cross-reference training strategies and distil recommendations applicable to similar PDE problems. Finally, we incorporate the time-dependent Kramers-Henneberger (KH) formulation as a demanding case study to probe robustness and physical fidelity.

6.1 Key Findings

Across the Poisson benchmarks, FBC consistently outperforms soft, weighted boundary penalties. Under FBC, PINN achieves the lowest L_2 error at a fixed budget of 10 000 epochs, whereas DRM has the shortest runtime. Because

all three methods solve the multi-dimensional Poisson setups without marked degradation, there is no evidence of a dimensionality bottleneck under our settings: performance remains stable as the dimensionality increases up to five (see Table 4.1). Given its accuracy-speed balance, DRM is an attractive default when moderate accuracy and fast fits are required, while PINN is preferable when the priority is the lowest error within a fixed training budget.

For the Schrödinger benchmarks, with soft boundary conditions and only 25% of domain data, all methods recover the ground state accurately in the infinite potential well (IPW) (Table 4.2). As we move to higher excited states, DRM collapses to the ground state, PINN continues to recover successive states reliably, and WAN succeeds up to the fourth excited state ($n = 5$) before failing. Similar patterns appear in the quantum harmonic oscillator (QHO) baseline (Table 4.3). Overall, PINN consistently attains the best accuracy across states, while DRM typically converges fastest but often only for the ground state under baseline settings. Representative runtimes illustrate this trade-off: for IPW, DRM requires 19.2 s versus 21.6 s for PINN; for the QHO, DRM requires 67.1 s versus 162.6 s for PINN.

Targeted improvements materially change this picture. Replacing soft boundary penalties with FBC improves all three methods solving the 1D IPW problem: the PINN L_2 error drops from 6.22×10^{-5} to 2.65×10^{-6}; DRM improves from 4.62×10^{-4} to 1.84×10^{-7}; and WAN improves from 2.16×10^{-5} to 8.79×10^{-9} (Table 4.4). Adding orthogonality regularization (OG) stabilizes excited-state recovery: using the DRM, the per-state differences are small but overall accuracy is high, while in WAN the OG term yields strong L_2 performance on higher states (Table 4.5). Enforcing fixed/forced nodes (FN), i.e., known nodal locations, is particularly effective in the IPW problems (both 1D and 2D) where node spacing is regular; under FN, WAN achieves the best L_2 error for $n = 1$–3, and PINN is best for $n = 4, 5$ (Tables 4.6 and 4.7). FN is less effective in QHO due to non-uniform node spacing and it requires prior nodal information, which limits generality. In two-dimensional Schrödinger problems, degeneracies (e.g., $(1, 2)$ versus $(2, 1)$) complicate OG; penalties can inadvertently mix degenerate states unless guided by initial data or symmetry cues. Our 2D experiments succeed at low states using data-driven disambiguation, but OG degrades at higher degeneracy levels.

6.2 Cross-Method Comparison

The outcomes from the various methods are summarised below for orientation; the measures themselves were given in previous chapters (Poisson in Table 4.1; IPW/QHO baselines in Tables 4.2–4.3; improved settings in Tables 4.4–4.7).

Table 6.1: Cross-method summary by task and dimension. "Accuracy" reflects the lowest L_2 among methods; "Runtime" the shortest Runtime.

	Poisson	IPW		QHO	
Metric	1D-5D	1D	2D	1D	2D
Accuracy	PINN	PINN	PINN	PINN	DRM
Runtime	DRM	DRM	DRM	DRM	DRM

In the Poisson family, adopting FBC removes sensitivity to the boundary-loss weight and improves stability across methods (Table 4.1). In Schrödinger problems, FBC yields modest gains for PINN (whose baselines already satisfy BCs) but large gains for DRM and WAN. However, FBC alone does not resolve high-state failures. FN is effective for the IPW, where nodes are regularly spaced, but unreliable for the QHO, where nodes are irregular. OG injects physical structure but is sensitive to degeneracy and to the accuracy of lower-state approximations. Treating the energy as a trainable parameter modestly improves PINN and DRM in QHO problems, while WAN appears hindered, likely due to generator-discriminator dynamics (compare the no-data results in Table 4.8 with the supervised baselines in Tables 4.2–4.3).

6.3 Sensitivity, Ablations, and Reproducibility

The experiments reveal clear sensitivities to loss weighting and supervision. Increasing the data-fit weight (partial supervision) improves baselines, whereas over-weighting the PDE residual degrades performance. In no-data settings focused on ground states, DRM outperforms others with L_2 errors in the 10^{-6}-10^{-8} range; PINN fails on the 2D IPW under these constraints while succeeding in 1D; and WAN matches DRM on the IPW but is weaker on QHO problems (Table 4.8). Most of the achievable accuracy is reached within roughly 10 000 epochs; extending to 50 000 yields diminishing returns (for example, improvements from 10^{-8} to 10^{-9}). For practical runs, 5000 epochs often deliver 10^{-4}–10^{-5} L_2 error. Increasing the number of neurons per layer consistently improves PINN and DRM

(single-network methods). For WAN, this trend holds only in shallow, one-layer settings; deeper settings require careful generator–discriminator width balance (our default used half the generator width for the discriminator). Increasing depth helps PINN and DRM modestly; for WAN, three layers perform best, and both too shallow and too deep configurations harm stability. These sensitivities are summarized in Table 4.9 and visualized in Fig. 6.1.

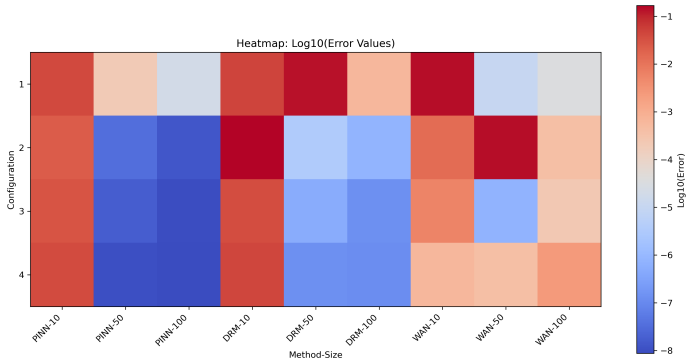


Figure 6.1: Error heatmap across architectures for $n = 1$, illustrating sensitivity to width and depth (lower is better).

The ablation studies further clarify causal contributions. Removing FBC and reverting to soft BCs increases boundary residuals and destabilizes training; dropping OG leads to mode mixing in excited states; and disabling FN especially harms IPW excited-state accuracy. Each configuration was repeated five times with fixed seeds, and variability was small to negligible. Seeds are fixed in the repository for deterministic sampling of collocation and boundary points. In our test problems, uniform sampling outperformed purely random sampling and produced more stable training, an effect visible in the consistency across repetitions in Table 4.8 and in the architecture-level robustness in Table 4.9.

6.4 KH Potential - Discussion

Guided by the insights from solving the IPW and QHO problems, we next consider the KH frame for strong-field physics. The potential and wavefunction in this setting exhibit complex structure, and training is correspondingly less smooth. With OG and only 25% of domain data, DRM alone converges reliably; increasing to 50% domain coverage enables all three methods to perform well.

PINN and DRM have comparable runtimes, whereas WAN is approximately twice as slow. When sweeping the KH parameter α, ground-state experiments show DRM achieving the best accuracy-speed trade-off, and the qualitative trends mirror the improvements observed under FBC and OG in the simpler benchmarks.

6.5 Practitioner-Oriented Guidelines

Translating the comparison into practice, we recommend selecting the method by matching problem traits to solver strengths. Problems with simple spectra and regular nodal structure (e.g., IPW) benefit from DRM or WAN augmented by FN; problems with irregular nodal spacing or pronounced degeneracy (e.g., QHO, higher-dimensional Schrödinger) are better served by PINN with FBC and OG. When supervision is scarce or unavailable, DRM is the most robust for ground states, while WAN requires careful discriminator-generator balancing and may underperform if energy is learned as a free parameter. In all cases, FBC should be preferred whenever feasible to avoid brittle boundary-loss tuning; OG should be added judiciously and evaluated for degeneracy-induced mixing. These choices can be summarized succinctly by pairing each task with the smallest configuration that achieves the desired L_2 tolerance within the time budget, using the results reported in Tables 4.4–4.7 as reference points.

6.6 Threats to Validity and Limitations

The conclusions are conditioned on the chosen architectures, optimizers, and sampling strategies. Although sensitivity analyses in this chapter indicate that width contributes more than depth for single-network solvers and that most gains appear within the first 10 000 epochs, other hyperparameter combinations may alter the speed-accuracy frontier. FN requires prior knowledge of nodal locations and therefore lacks generality for arbitrary potentials. OG can entangle degenerate manifolds in higher dimensions unless guided by additional symmetry-aware constraints or targeted initialization. Finally, wall-clock comparisons depend on hardware and implementation details and, while the relative rankings have been consistent across repetitions, absolute times should be interpreted as indicative rather than definitive.

6.7 Future Directions

Several focused extensions can broaden the scope and strengthen the robustness of the present study. On the physics side, a natural progression is from low-dimensional benchmarks to three-dimensional, single-particle Schrödinger problems where analytic or high-precision numerical references enable tight validation of accuracy-cost scaling. Interacting systems are a principled step beyond: two particles in three dimensions yield a six-dimensional configuration space that becomes tractable by separating centre-of-mass and relative coordinates, exploiting symmetries (spin, parity, angular momentum), and embedding those invariances in the architecture or loss. Ultimately, many-body settings remain challenging for classical discretizations; here, neural solvers could act as fast surrogates for reduced models or pair with low-rank and tensor-network parameterizations to control complexity while retaining physical fidelity.

Methodologically, forced boundary conditions (FBC) should remain the default boundary-enforcement mechanism, as they remove brittle loss balancing and concentrate optimization on interior residuals. Time dependence, handled natively by PINNs, can be extended to DRM and WAN. For DRM, the stationary energy-minimization principle generalizes to the time-dependent Schrödinger equation by minimizing an action functional over spacetime (e.g., a least-action or McLachlan variational principle) with temporal boundary or initial conditions imposed by construction; this yields a spacetime deep Ritz formulation that preserves variational structure. For WAN, the weak form can be lifted to spacetime by allowing the discriminator (or test-function network) to integrate over both spatial and temporal domains; periodic or Floquet conditions can be enforced in the KH frame by design. In both cases, careful scaling of temporal and spatial residuals is essential to avoid stiffness, and shared spacetime representations can reduce the need for explicit time marching while remaining mesh-free.

Adaptive, error-driven sampling is a complementary lever. Residual-based refinement, multi-resolution schedules, and simple curricula (e.g., homotopy in potential strength or in the KH parameter α) can concentrate collocation points where they matter most: near sharp potential features, along nodal sets of excited states, and in regions with slow PINN/DRM convergence. Parallel multi-state training with a shared trunk and state-specific heads, coupled to explicit block-orthogonality constraints or a Gram-Schmidt layer, amortizes features across eigenstates and reduces Runtime compared to sequential deflation, while

mitigating error propagation from low to high quantum numbers.

The role of fixed/forced nodes (FN) merits deeper study. FN is effective when nodal locations are known and regular (as in the infinite well), but it is brittle for irregular spectra (e.g., the harmonic oscillator) and requires prior solution knowledge. A more general alternative is neural domain decomposition: multiple subnetworks solve on overlapping subdomains with continuity and flux-matching penalties at interfaces (a neural analogue of Schwarz or mortar methods). This preserves the benefits of FBC locally, adapts naturally to complex geometries or unknown nodal patterns, and allows residual-driven refinement of subdomain boundaries. Where node information is partially available, one can jointly learn approximate nodal sets with a small auxiliary predictor and refine them a posteriori using residual indicators, rather than hard-coding fixed nodes.

Two additional strands can enhance practicality. First, operator learning provides amortized solvers over families of potentials, delivering rapid predictions for parameter sweeps and high-quality warm starts for PINN/DRM/WAN fine-tuning on specific instances. Second, stability, efficiency, and credibility benefit from staged optimizers (Adam followed by L–BFGS), discriminator regularization and two-time-scale updates for WAN, symmetry-aware parameterizations (e.g., rotational or gauge equivariance in KH), mixed precision, multi-GPU data parallelism, and systematic uncertainty quantification via ensembles or Bayesian variants alongside residual-based error indicators. Together, these directions aim for robust, scalable neural PDE solvers applicable to higher-dimensional, interacting, and time-dependent quantum systems where classical techniques become costly.

Chapter 7

Conclusion

This dissertation investigated three neural PDE solvers, physics-informed neural networks (PINNs), the deep Ritz method (DRM), and weak adversarial networks (WANs), across a broad suite of benchmarks: solving multi-dimensional Poisson problems and multi-state one- and two-dimensional Schrödinger equations for the infinite potential well and the quantum harmonic oscillator. Under a unified evaluation protocol, all three approaches achieved low solution errors, typically in the range 10^{-6} to 10^{-9}, when complemented by principled design choices such as forced boundary conditions (FBC), fixed or forced nodes (FN) where nodal sets are known, and orthogonality regularization (OG) for reliable recovery of excited states. Building on these foundations, we formulated and solved the laser-driven Schrödinger equation in the Kramers-Henneberger (KH) frame, demonstrating that the same methodological toolkit extends to a demanding, time-independent setting and enabling an end-to-end Schrödinger solver for this class of problems.

The comparative study yields several clear takeaways. PINN emerges as a dependable, general-purpose choice: with FBC and modest architectural tuning, it consistently delivers strong accuracy across Poisson and Schrödinger tasks and scales smoothly to higher excited states. DRM is particularly attractive for stationary problems, combining fast convergence with competitive accuracy; where time dependence is absent, or can be handled through a variational extension, DRM often attains the best accuracy–runtime trade-off. WAN, while more sensitive to capacity balance and training dynamics due to its two-network structure, performs well in regimes where weak-form constraints and FN or OG are leveraged effectively; with careful regularization and scheduling, it achieves accurate solutions on several benchmarks.

Beyond numerical performance, the thesis clarifies why these methods succeed under specific configurations. FBC removes brittle boundary-loss tuning and shifts optimization effort to the interior residual. OG prevents mode collapse in spectral problems by separating eigenstates during training, provided degeneracies are handled with care. FN is powerful when nodal information is reliable and regularly spaced, as in the infinite well, but benefits from more general strategies, such as neural domain decomposition, when nodes are unknown or irregular, as in the harmonic oscillator. Sensitivity analyses and ablations show that most gains accrue within a moderate training budget, that width is usually more impactful than depth for single-network solvers, and that uniform or residual-informed sampling stabilizes training.

There are, of course, limitations. Performance depends on reasonable hyperparameter choices and architectures; degeneracies in higher dimensions can stress OG without symmetry-aware guidance; WAN requires disciplined discriminator regularization. Nonetheless, the evidence across Chapters 4-6 supports a pragmatic recipe: prefer FBC whenever feasible; use PINN as the default for accuracy and for excited spectra; choose DRM where rapid stationary solves are paramount; and deploy WAN when weak-form advantages or FN and OG structure can be exploited. The extension to the KH frame indicates that these principles transfer beyond canonical benchmarks and into driven quantum dynamics.

Looking ahead, several directions appear especially promising: variational and weak-form extensions that bring fully time-dependent DRM and WAN to parity with PINNs, adaptive and residual-driven sampling to focus computation where it matters most, parallel multi-state training with explicit orthogonality constraints to reduce wall-clock runtime, and neural domain decomposition to replace hand-crafted FN in problems with unknown nodal patterns. Operator-learning surrogates, symmetry-aware parameterizations, and systematic uncertainty quantification can further improve scalability, stability, and credibility. Taken together, the results suggest that neural PDE solvers, when anchored by sound physical priors and disciplined training practice, offer a viable and increasingly mature alternative to classical discretizations for complex, high-dimensional scientific problems.

Acknowledgements

I would like to express my sincere gratitude to my supervisor, Prof Simon, for his guidance and invaluable advice throughout my research. I am also thankful to Prof Carla who gives the physics guidance of this project.

A special thanks to my parents for their unwavering support and encouragement during my studies.

Bibliography

[1] John Rozier Cannon. *The one-dimensional heat equation*. Encyclopedia of Mathematics and its Applications. Cambridge University Press, Cambridge, 1984.

[2] Weinan E and Bing Yu. The deep ritz method: a deep learning-based numerical algorithm for solving variational problems, September 2017. arXiv:1710.00211 [cs].

[3] Damian P. Hampshire. A derivation of maxwell's equations using the heaviside notation. *Philosophical Transactions. Series A, Mathematical, Physical, and Engineering Sciences*, 376(2134):20170447, December 2018. TLDR: This work demonstrates that it is the conservation of charge that couples time-varying E-fields and B-field and that Faraday's Law can be derived without any relativistic assumptions about Lorentz invariance and widens the choice of axioms, or starting points, for understanding electromagnetism.

[4] Liam Harcombe and Quanling Deng. Physics-informed neural networks for discovering localised eigenstates in disordered media. *Journal of Computational Science*, 73:102136, November 2023.

[5] Kurt Hornik, Maxwell Stinchcombe, and Halbert White. Multilayer feedforward networks are universal approximators. *Neural Networks*, 2(5):359–366, 1989.

[6] Ameya D. Jagtap and George Em Karniadakis. Extended physics-informed neural networks (XPINNs): a generalized space-time domain decomposition based deep learning framework for nonlinear partial differential equations. *Communications in Computational Physics*, 28(5):2002–2041, January 2020. Publisher: Global Science Press.

[7] Kakitc. Finite difference method, May 2025. Page Version ID: 1291248732.

[8] Namgyu Kang, Jaemin Oh, Youngjoon Hong, and Eunbyung Park. PIG: physics-informed gaussians as adaptive parametric mesh representations, March 2025. arXiv:2412.05994 [cs].

[9] E. Kharazmi, Z. Zhang, and G. E. Karniadakis. Variational physics-informed neural networks for solving partial differential equations, November 2019. arXiv:1912.00873 [cs].

[10] Diederik P. Kingma and Jimmy Ba. Adam: a method for stochastic optimization, January 2017. arXiv:1412.6980 [cs].

[11] Dimitri Komatitsch and Jeroen Tromp. Introduction to the spectral element method for three-dimensional seismic wave propagation. *Geophysical Journal International*, 139(3):806–822, December 1999.

[12] Randall J. LeVeque. *Finite volume methods for hyperbolic problems*. Cambridge University Press, August 2002. Google-Books-ID: QazcnD7GUoUC.

[13] Zongyi Li, Nikola Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Fourier neural operator for parametric partial differential equations, May 2021. arXiv:2010.08895 [cs].

[14] Samuel J Ling, Jeff Sanny, and William Moebs. University physics volume 3.

[15] Dong C. Liu and Jorge Nocedal. On the limited memory BFGS method for large scale optimization. *Mathematical Programming*, 45(1):503–528, August 1989. TLDR: The numerical tests indicate that the L-BFGS method is faster than the method of Buckley and LeNir, and is better able to use additional storage to accelerate convergence, and the convergence properties are studied to prove global convergence on uniformly convex problems.

[16] Lu Lu, Pengzhan Jin, and George Em Karniadakis. DeepONet: learning nonlinear operators for identifying differential equations based on the universal approximation theorem of operators. *Nature Machine Intelligence*, 3(3):218–229, March 2021. arXiv:1910.03193 [cs].

[17] Doug McLean. *Understanding aerodynamics: arguing from the real physics*. John Wiley & Sons, December 2012. Google-Books-ID: UE3sxu28R0wC.

[18] Connor Mooney. The Monge-Ampère Equation, June 2018. arXiv:1806.09945 [math].

[19] Papa November. Particle in a box, July 2025. Page Version ID: 1303215041.

[20] Juncai Pu, Jun Li, and Yong Chen. Solving localized wave solutions of the derivative nonlinear schrödinger equation using an improved PINN method. *Nonlinear Dynamics*, 105(2):1723–1739, July 2021.

[21] Shivendra Nath Rai, D. V. Ramana, and A. Manglik. *Dynamics of Earth's fluid system*. CRC Press, January 2002. Google-Books-ID: 05EI8dlRwNkC.

[22] 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, February 2019.

[23] Justin Sirignano and Konstantinos Spiliopoulos. DGM: a deep learning algorithm for solving partial differential equations. *Journal of Computational Physics*, 375:1339–1364, December 2018. arXiv:1708.07469 [q-fin].

[24] A. Tasnim Aynul, L. Cruz Rodriguez, and C. Figueira De Morisson Faria. Quantum beating and cyclic structures in the phase-space dynamics of the kramers-henneberger atom. *Physical Review A*, 111(4):43102, April 2025.

[25] Carlos Uriarte, David Pardo, Ignacio Muga, and Judit Muñoz-Matute. A deep double ritz method for solving partial differential equations using neural networks. *Computer Methods in Applied Mechanics and Engineering*, 405:115892, February 2023. TLDR: The Deep Double Ritz Method is called, which combines two neural networks for approximating trial functions and optimal test functions along a nested double Ritz minimization strategy, and numerical results on different diffusion and convection problems support the robustness of the method.

[26] Li Wang and Zhenya Yan. Data-driven rogue waves and parameter discovery in the defocusing nonlinear schrödinger equation with a potential using the PINN deep learning. *Physics Letters A*, 404:127408, July 2021. TLDR: The multi-layer PINN deep learning method is used to study the data-driven rogue wave solutions of the defocusing nonlinear Schr\"odinger (NLS) equation with the time-dependent potential by considering several initial conditions such as the rogue wave.

[27] Liu Yang, Xuhui Meng, and George Em Karniadakis. B-PINNs: bayesian physics-informed neural networks for forward and inverse PDE problems with noisy data, March 2020. arXiv:2003.06097 [stat].

[28] Wen-Xuan Yuan, Rui Guo, and Yi-Ning Gao. Physics-informed neural network method for the modified nonlinear schrödinger equation. *Optik*, 279:170739, May 2023.

[29] Yaohua Zang, Gang Bao, Xiaojing Ye, and Haomin Zhou. Weak adversarial networks for high-dimensional partial differential equations. *Journal of Computational Physics*, 411:109409, June 2020. arXiv:1907.08272 [math] TLDR: This paper converts the problem of finding the weak solution of PDEs into an operator norm minimization problem induced from the weak formulation, and parameterized as the primal and adversarial networks respectively, which are alternately updated to approximate the optimal network parameter setting.

[30] M. Necati Özişik, Helcio R. B. Orlande, Marcelo J. Colaço, and Renato M. Cotta. *Finite difference methods in heat transfer*. CRC Press, Boca Raton, 2 edition, July 2017.