SCIENCE CHINA
Physics, Mechanics & Astronomy

• Article •
Special Topic: Quantum Computing for Fluids

October 2025 Vol. 68 No. 10: 104703
https://doi.org/10.1007/s11433-025-2649-9

A quantum-inspired deep neural network framework for physically constrained PDEs

Jinsong Tang[1,2], Jia Xiong[1], Ali Minaeian[1], Yekang Jie[3], and Shiying Xiong[1,4*]

[1] *School of Aeronautics and Astronautics, Zhejiang University, Hangzhou 310007, China*
[2] *Huanjiang Laboratory, Zhejiang University, Zhuji 311899, China*
[3] *College of Artificial Intelligence, Tianjin University of Science and Technology, Tianjin 300457, China*
[4] *National Key Laboratory of Aerospace Physics in Fluids, Mianyang 621000, China*

Received January 21, 2025; accepted March 17, 2025; published online June 23, 2025

We propose a novel quantum-inspired deep neural network framework (QIDNNF) for solving partial differential equations (PDEs), specifically Schrödinger equation and those derived through Schrödingerization. QIDNNF integrates fundamental quantum mechanics principles, including global phase invariance and normalization, to ensure unitary quantum dynamics and the preservation of conservation laws. Through numerical experiments, QIDNNF exhibits superior stability over finite difference schemes for large time steps, improved long-term accuracy over neural ordinary differential equations (Neural ODEs) and physics-informed neural networks (PINNs), and predictive precision unaffected by variations in initial phase angles. Furthermore, QIDNNF effectively models real-world physical systems, including 1D nonlinear wave propagation and 2D and 3D flow evolution, demonstrating their accuracy and consistency in simulating complex physical phenomena.

partial differential equations, global phase invariance, normalization, Schrödingerization, neural ODEs

PACS number(s): 02.60.Lj, 02.90.+p, 47.11.-j

Citation: J. Tang, J. Xiong, A. Minaeian, Y. Jie, and S. Xiong, A quantum-inspired deep neural network framework for physically constrained PDEs, Sci. China-Phys. Mech. Astron. **68**, 104703 (2025), https://doi.org/10.1007/s11433-025-2649-9

1 Introduction

Solving partial differential equations (PDEs) and ordinary differential equations (ODEs) is essential for modeling a broad range of physical phenomena such as fluid dynamics, heat transfer, and electromagnetic processes [1]. Two main computational approaches are typically employed: physics-based numerical methods [2] and data-driven machine learning (ML) techniques [3]. While classical numerical methods, such as finite difference and finite element methods, have proven effective in many cases, they may face challenges when dealing with nonlinearities, multiscale effects, and the growing computational demands of complex systems [4-8]. ML techniques, particularly neural networks, present a promising alternative by learning from datasets to uncover complex patterns without depending on explicit mathematical formulations. This capacity for generalization across diverse systems, coupled with their efficiency in handling large-scale problems, has led to the growing adoption of ML for solving PDEs. Applications of this approach extend across a wide range of fields, including fluid dynamics, materials science, and other complex domains [9-13]. Nonetheless, ML methods are constrained by factors such as reliance on large, high-quality datasets, interpretability issues, and high training costs.

*Corresponding author (email: shiying.xiong@zju.edu.cn)

© Science China Press 2025

While numerical methods emphasize precision and controllability, and ML focuses on flexibility and computational efficiency, the synergistic potential of their complementary strengths has driven increasing interest in hybrid methodologies that combine physical principles with neural networks for solving PDEs [14]. Physics-informed neural networks (PINNs), introduced by Raissi et al. [15], incorporate physical equations into the network's loss function, enabling solutions to problems governed by differential equations. PINNs have shown success in areas such as fluid dynamics [16], heat conduction [17], and solid mechanics [18, 19]. However, challenges continue to exist when scaling these approaches to tackle high-dimensional or complex systems [20, 21]. Beyond generic neural networks, specialized frameworks have been developed to enhance the modeling of constrained and domain-specific physical systems. Neural ordinary differential equations (Neural ODEs) [22-24] capture continuous-time dynamics by learning time derivatives, providing a natural and flexible framework for systems governed by ODEs. Symplectic networks, specifically designed for Hamiltonian systems [25], ensure the preservation of symplectic structures, enabling accurate and energy-conserving simulations over extended time scales. Roe network [26], tailored for hyperbolic conservation laws, delivers enhanced stability and precision, particularly in resolving complex phenomena such as shock waves and discontinuities, as well as other networks designed for structural dynamics [27, 28].

Building on the Schrödingerization framework proposed by Jin et al. [29-31], which unifies classical ODEs and PDEs within the structure of Schrödinger equation for modeling complex nonlinear systems, we introduce the quantum-inspired deep neural network framework (QIDNNF). This framework predicts the evolution of Schrödinger equation, including those derived from general ODEs and PDEs via Schrödingerization, while rigorously preserving fundamental quantum principles such as phase invariance and normalization.

The QIDNNF utilizes a two-step data flow methodology to enforce distinct, non-interacting constraints. In the first step, we apply an integral averaging technique to smooth high-frequency phase oscillations arising during the quantum state's evolution, approximating them through a summation process that aggregates rapid phase changes over time or space. This averaging reduces the impact of oscillations and mitigates fluctuations that could destabilize the numerical solution, thereby enhancing the system's stability and ensuring the physical consistency of the quantum state throughout its evolution. In the second step, we employ a semi-implicit scheme that implicitly handles certain evolution terms on the right-hand side of the Schrödinger equation, thereby avoiding the numerical instability and error accumulation typical of explicit methods through implicit wave function updates at each time step. This approach effectively preserves the wave functions normalization over large time steps, ensuring the physical consistency of the quantum system, while also facilitating nonlinear scaling of input data without compromising accuracy. By maintaining the independence of these two steps, we ensure that the constraints do not interfere with each other, allowing each aspect to be independently optimized, resulting in a more efficient and accurate solution to the Schrödinger equation.

To evaluate the effectiveness of this method, we conduct a series of numerical experiments across various problem sets, comparing its performance with established techniques including fourth-order Runge-Kutta (RK4), Neural ODEs, and PINNs, highlighting its robustness and accuracy. We further extend the application of QIDNNF to a range of PDE problems, encompassing linear and nonlinear systems, as well as high-dimensional and time-dependent scenarios. Their effectiveness is demonstrated through test cases such as vortex dynamics, nonlinear wave propagation, and incompressible Schrödinger flow, demonstrating QIDNNFs ability to accurately simulate complex physical phenomena while maintaining physical consistency.

This work introduces QIDNNF as a computational methodology for solving PDEs, grounded in physical principles that ensure consistent enforcement of constraints throughout the solution process. We provide numerical examples comparing QIDNNF with traditional methods, such as finite difference methods, using benchmark problems that include linear and nonlinear fluid flows with time-dependent dynamics. We apply QIDNNF to a range of PDEs, including linear, nonlinear, high-dimensional, and time-dependent scenarios, demonstrating their effectiveness through test cases such as vortex dynamics, nonlinear wave propagation, and incompressible Schrödinger flow.

The paper is structured as follows: sect. 2 introduces the mathematical framework for the Schrödinger equation and the QIDNNF architecture. Sect. 3 presents the training and evaluation of QIDNNF on benchmark problems, highlighting its advantages over traditional methods. Sect. 4 explores the application of QIDNNF to different physical problems. Sect. 5 summarizes the key findings, addresses the limitations of the proposed approach, and outlines potential directions for future research.

2 Mathematical framework

This section outlines the mathematical foundations of QIDNNF, starting with the Schrödingerization process that transforms PDEs into a Schrödinger-like form. Two funda-

mental quantum principlesłphase invariance and normalizationłare integrated into the neural network architecture to ensure the physical accuracy of the solutions. Building on these elements, QIDNNF is presented as a robust method for solving PDEs while preserving key quantum properties, such as probability conservation and coherent wave function evolution, throughout the solution process.

2.1 The Schrödinger equation and PDEs

We present a neural network-based methodology for addressing a diverse range of PDEs. The general form of a PDE is given by

$$\frac{\partial \boldsymbol{u}}{\partial t} = \boldsymbol{f}\left(\mathcal{D}\boldsymbol{u}, \boldsymbol{u}, \boldsymbol{x}, t\right), \quad (\boldsymbol{x}, t) \in \mathbb{R}^{n_x} \times [0, T), \tag{1}$$

with the initial condition $\boldsymbol{u}(\boldsymbol{x}, 0) = \boldsymbol{u}_0(\boldsymbol{x})$ and appropriate boundary conditions [1]. Here, $\boldsymbol{u}(\boldsymbol{x}, t)$ denotes an n_u-dimensional vector field, $\mathcal{D}$ represents the spatial operator, and $\boldsymbol{f}$ defines the function governing the time evolution of $\boldsymbol{u}$.

Directly solving PDEs typically involves discretizing both the spatial and temporal domains, a process that can be computationally demanding, especially for high-dimensional or time-dependent systems. Moreover, the presence of nonlinear terms in many PDEs further complicates the solution process. To mitigate these challenges, we employ Schrödingerization, a technique that transforms the original PDE variables $\boldsymbol{u}$ into a wave function $\boldsymbol{\psi}$. This reformulation recasts the problem into a form governed by the Schrödinger equation, thereby allowing us to exploit the well-established mathematical framework of quantum mechanics. The linear structure of the Schrödinger equation simplifies the treatment of nonlinearities, while its stability properties render the approach more numerically tractable.

Under the Schrödingerization process, the transformation is defined as

$$\boldsymbol{\psi} = \mathscr{S}\left(\boldsymbol{u}\right), \tag{2}$$

where $\boldsymbol{\psi}$ represents the wave function, and $\mathscr{S}$ is a mapping function that transforms the original PDE variables $\boldsymbol{u}$ into a suitable form that allows the problem to be expressed in terms of the Schrödinger equation. This transformation enables numerical methods and neural networks to inherit the favorable properties of quantum mechanics.

The Schrödinger equation governs the evolution of the wave function $\boldsymbol{\psi}$ [32],

$$\mathrm{i}\hbar\frac{\partial}{\partial t}\boldsymbol{\psi} = \boldsymbol{H}\boldsymbol{\psi}, \tag{3}$$

with the initial condition $\boldsymbol{\psi}(0) = \boldsymbol{\psi}_0$. Here, $\boldsymbol{\psi} = (\psi_1, \ldots, \psi_{n_\psi})^{\mathrm{T}}$ is an n_ψ-dimensional complex vector, and $\boldsymbol{H}$ is the Hermitian operator corresponding to the system's Hamiltonian, governing its total energy dynamics. i is the imaginary unit, and $\hbar$ is Plancks constant, which we set to unity for simplicity.

2.2 Quantum principles

To model wave function evolution governed by eq. (3) using neural networks, we adopt the Neural ODEs formulation:

$$\frac{\mathrm{d}\boldsymbol{\psi}}{\mathrm{d}t} = \mathcal{F}_{\boldsymbol{\theta}}\left(\boldsymbol{\psi}, t\right), \tag{4}$$

where $\boldsymbol{\psi}$ represents the discretized wave function, and $\boldsymbol{\theta}$ are the neural network parameters. In this formulation, spatial coordinates $\boldsymbol{x}$ and differential operators are approximated by discrete grids and neural network representations, respectively.

Although this approach enables the numerical solution of differential equations using methods such as Euler or Runge-Kutta, it does not inherently ensure that the physical constraints required by quantum systems are satisfied. In the absence of these constraints, such as wave function normalization, the numerical predictions of the wave functions evolution may diverge from the true quantum dynamics, unless these constraints are explicitly incorporated into the model. To address this, we summarize the two key principles that govern quantum state evolution.

Property 1 (Global phase invariance) In quantum mechanics, the wave function remains invariant under global phase transformations, meaning that multiplying the wave function by a global phase factor $\mathrm{e}^{\mathrm{i}\phi}$ does not alter its physical validity. Specifically, if $\boldsymbol{\psi}$ is a solution to eq. (3), then $\mathrm{e}^{\mathrm{i}\phi}\boldsymbol{\psi}$ is also a valid solution for any global phase ϕ. This is because the global phase does not affect observable quantities such as probability densities, which depend on the absolute square of the wave function $|\boldsymbol{\psi}|^2$. The phase factor, being a complex number of unit magnitude, leaves these physical properties unchanged. This invariance reflects a fundamental symmetry in quantum systems, where different representations of the same quantum state, differing only by a global phase, are considered physically equivalent. Hence, the global phase of a wave function is unobservable and does not influence the dynamics described by the Schrödinger equation.

Property 2 (Normalization preservation) In quantum mechanics, the Schrödinger equation preserves the normalization of the wave function over time due to its unitary nature. The wave function $\boldsymbol{\psi}$ encodes the probabilities of a quantum system, and its normalization ensures that the total probability of finding the particle in all space sums to 1. If the wave function is initially normalized, meaning $\|\boldsymbol{\psi}_0\| = 1$ at time $t = 0$, the Schrödinger equation guarantees that this nor-

malization is maintained as the system evolves. This occurs because the evolution governed by the Schrödinger equation is unitary, meaning it conserves the norm of the wave function at all time. Thereby, the wave function at any later time will also satisfy $\|\boldsymbol{\psi}(t)\| = 1$, ensuring the conservation of total probability density throughout the system's dynamics.

2.3 Quantum-inspired deep neural network framework

We introduce the QIDNNF that merges the principles of quantum mechanics with traditional ODEs through neural networks, providing an approach for modeling dynamic systems that uphold quantum mechanical consistency while harnessing the expressive capabilities of neural networks. Figure 1 [33] compares three methodologies for solving Schrödinger's equation: quantum circuits, classical neural networks, and the proposed QIDNNF. Quantum circuits theoretically provide a robust framework for accuracy and physical consistency by inherently adhering to quantum principles. Classical neural networks, while powerful, fail to respect these principles, often producing results that deviate from physical laws, such as non-conservation of probability or energy. In contrast, QIDNNF integrates quantum mechanical constraints, including phase invariance (Property 1) and normalization (Property 2), into their architecture, enabling them to achieve solutions as accurate as those from quantum circuits. This makes QIDNNF both computationally efficient and consistent with quantum mechanical principles, bridging the gap between classical computation and quantum accuracy while mitigating the deficiencies of standard neural networks in solving physics-informed problems.

The QIDNNF employs a two-step methodology. First, an integral averaging technique is applied to smooth out rapid phase oscillations in the wave function over a finite interval, thereby mitigating numerical errors caused by high-frequency fluctuations. Second, a semi-implicit scheme is utilized to balance explicit and implicit updates, preserving the normalization of the wave function at each time step while allowing larger time steps without compromising accuracy or normalization.

Phase integration operation We define the network $\mathcal{F}_{\boldsymbol{\theta}}$ to incorporate a phase parameter α into the existing network $\mathcal{G}_{\boldsymbol{\theta}}$ and perform a periodic integral over the full phase cycle $[0, 2\pi]$. Specifically, the operation is given by

$$\mathcal{F}_{\boldsymbol{\theta}}(\boldsymbol{y}, t) = \int_0^{2\pi} \mathcal{G}_{\boldsymbol{\theta}}\left(\boldsymbol{y}\mathrm{e}^{\mathrm{i}\alpha}, t\right)\mathrm{e}^{-\mathrm{i}\alpha}\,\mathrm{d}\alpha. \tag{5}$$

Theorem 1 The network designed within the QIDNNF satisfies the phase invariance property (Property 1).

Proof To establish global phase invariance, we must demonstrate that for any global phase p, the network output remains invariant under the transformation $\boldsymbol{y}\mathrm{e}^{\mathrm{i}p}$. Specifically, we aim to prove

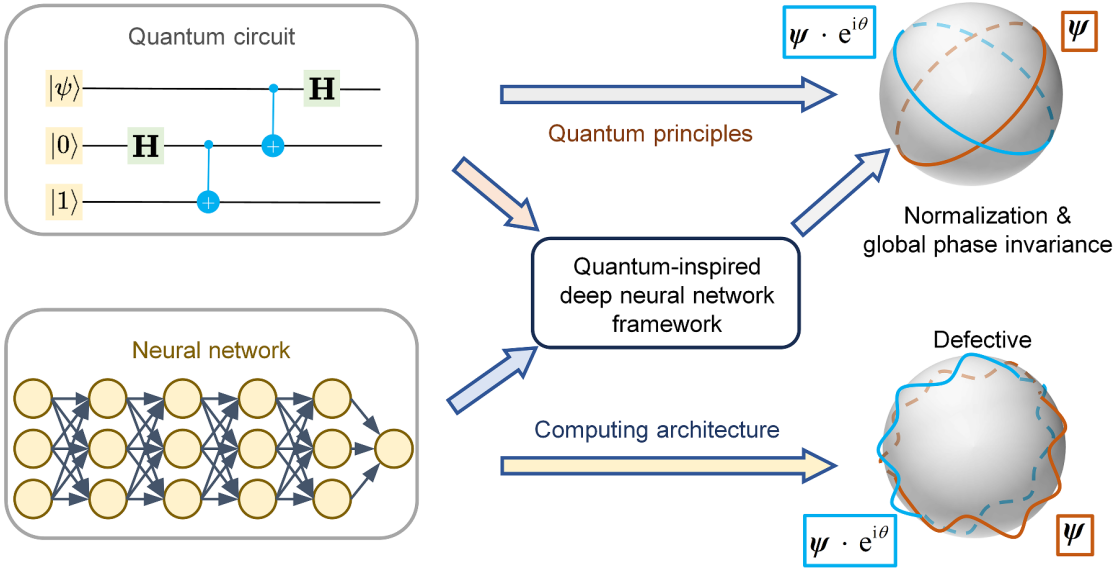

Figure 1 (Color online) Comparison of methodologies for solving Schrödinger equation. Quantum circuits [33] theoretically provide a robust framework for accuracy and physical consistency by inherently adhering to quantum principles. Classical neural networks, while effective, often violate these principles, leading to non-physical results. In contrast, QIDNNF integrates quantum mechanical constraints, such as normalization and phase invariance, achieving accuracy comparable to quantum circuits.

$$\mathcal{F}_{\theta}\left(\boldsymbol{y}\mathrm{e}^{\mathrm{i}p}, t\right) = \mathrm{e}^{\mathrm{i}p}\mathcal{F}_{\theta}\left(\boldsymbol{y}, t\right). \tag{6}$$

We compute the integral as follows:

$$\mathcal{F}_{\theta}\left(\boldsymbol{y}\mathrm{e}^{\mathrm{i}p}, t\right) = \int_0^{2\pi} \mathcal{G}_{\theta}\left(\boldsymbol{y}\mathrm{e}^{\mathrm{i}(\alpha+p)}, t\right)\mathrm{e}^{-\mathrm{i}\alpha}\,\mathrm{d}\alpha. \tag{7}$$

By making the change of variables $\beta = \alpha + p$, we obtain

$$\mathcal{F}_{\theta}\left(\boldsymbol{y}\mathrm{e}^{\mathrm{i}p}, t\right) = \mathrm{e}^{\mathrm{i}p}\int_0^{2\pi} \mathcal{G}_{\theta}\left(\boldsymbol{y}\mathrm{e}^{\mathrm{i}\beta}, t\right)\mathrm{e}^{-\mathrm{i}\beta}\,\mathrm{d}\beta. \tag{8}$$

This demonstrates that the network is invariant under global phase transformations, thereby confirming the phase invariance property.

Normalization operation To ensure the preservation of wave function normalization during time evolution, we introduce a normalization factor λ. The wave function update rule is given by

$$\boldsymbol{y}^{n+1} = \lambda\boldsymbol{y}^{n} + \mathcal{Q}(\mathcal{F}_{\theta}, \boldsymbol{y}^{n}, t^{n}, \Delta t), \tag{9}$$

where $\mathcal{Q}$ represents the time integration operator. For instance, in the case of the Euler method, we have

$$\mathcal{Q}(\mathcal{F}_{\theta}, \boldsymbol{y}^{n}, t^{n}, \Delta t) = \Delta t\mathcal{F}_{\theta}(\boldsymbol{y}^{n}, t^{n}).$$

Assuming that the wave function at the previous time step is normalized ($\|\boldsymbol{y}^{n}\| = 1$), the normalization factor λ is computed as

$$\lambda = \sqrt{1 - [\mathrm{Im}(\boldsymbol{y}^{n}\overline{\mathcal{Q}})]^2} - \mathrm{Re}(\boldsymbol{y}^{n}\overline{\mathcal{Q}}), \tag{10}$$

where $\overline{\mathcal{Q}}$ represents the complex conjugate of $\mathcal{Q}$; Re and Im denote the real and imaginary parts, respectively.

Theorem 2 The calculation of λ in eq. (10) ensures that the wave function obtained from eq. (9) satisfies the normalization property (Property 2).

Proof To verify the normalization condition, we compute the squared norm of $\boldsymbol{y}^{n+1}$:

$$\|\boldsymbol{y}^{n+1}\|^2 = \left(\lambda\overline{\boldsymbol{y}^{n}} + \overline{\mathcal{Q}}\right)(\lambda\boldsymbol{y}^{n} + \mathcal{Q}). \tag{11}$$

By substituting the assumption that $\|\boldsymbol{y}^{n}\| = 1$, we find that the normalization factor λ is determined to satisfy $\|\boldsymbol{y}^{n+1}\| = 1$, thereby preserving the normalization at each time step.

We remark that by combining phase invariance and normalization operations, QIDNNF ensures its predictions adhere to the foundational principles of quantum mechanics. Embedding these principles extends the capabilities of traditional ML models, enabling QIDNNF to deliver accurate and physically consistent predictions for both quantum-inspired and classical systems.

As shown in Figure 2, the QIDNNF follows a structured operational flow to ensure accurate and physically consistent predictions. It begins with generating a high-precision dataset of wave function evolution using numerical methods, ensuring accurate ground truth data for training. This dataset is fed into the neural network, where predictions of wave function evolution are compared to the ground truth through a loss function that incorporates physical constraints, such as phase invariance and normalization. Training iteratively minimizes the loss, and once the loss function converges, the network is considered well-trained and ready for prediction. The initialized wave function is then input into the trained network, which predicts the wave function's evolution efficiently and accurately. Finally, reverse Schrödingerization is applied to map the predicted wave function back to physical fields of interest, such as energy or density, ensuring the outputs remain interpretable and physically meaningful.

3 Performance verification

We conduct a series of numerical tests to validate the predictive capabilities of QIDNNF in modeling the evolution of complex dynamical systems while ensuring physical consistency and computational efficiency. In this benchmark, we evaluate QIDNNF on a linear system of two coupled wave functions governed by a constant Hamiltonian, $\boldsymbol{\mathcal{H}} = \sigma_1 + \sigma_3$, where σ_1 and σ_3 are Pauli matrices representing coupling and energy level differences, respectively. The initial conditions for the system are specified as $\boldsymbol{\psi}^0 = \mathrm{e}^{\mathrm{i}\theta}\left[\psi_1^0\ \psi_2^0\right]$, with θ representing the initial global phase. The initial wave functions are given by $\left[\psi_1^0\ \psi_2^0\right] = \frac{1}{\sqrt{2}}\left[\mathrm{i}\ 1\right]$. This setup tests QIDNNF's ability to respect phase invariance, handle complex-valued dynamics, and maintain normalization during wave function evolution. By comparing the predicted wave function against high-precision numerical solutions, the benchmark assesses QIDNNF's accuracy, consistency with quantum mechanical principles, and ability to model coupled dynamics efficiently. We note that all examples in this paper were performed on the same computer hardware, consisting of an Intel Core i9-14900K processor running at 3.20 GHz, 64 GB of RAM, and an NVIDIA GeForce RTX 4080 SUPER GPU with 32 GB of memory.

3.1 Training settings

Algorithm 1 outlines the procedure for advancing the wave function in QIDNNF while maintaining phase invariance and normalization. The process begins with initializing an accumulator $\boldsymbol{s}$ to collect contributions from phase integration, calculating the integration step size $h = 2\pi/N^{\mathrm{int}}$, and setting

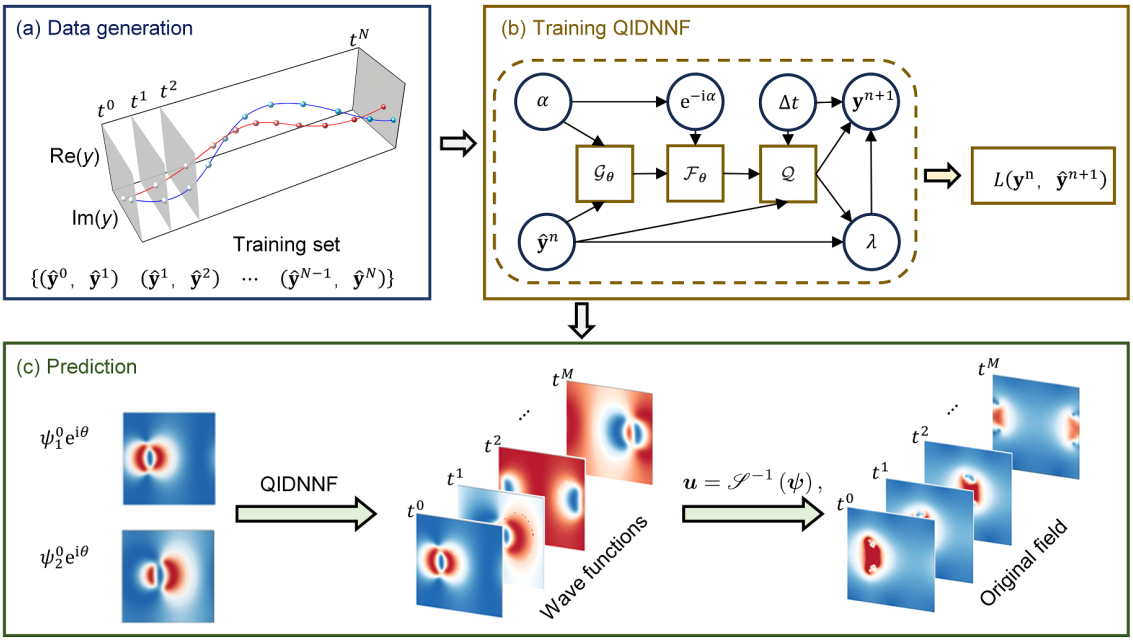

Figure 2 (Color online) Operational flow of QIDNNF: (a) data generation, (b) training QIDNNF, and (c) prediction. High-precision numerical methods generate the training dataset for wave function evolution as accurate ground truth data. A loss function, incorporating physical constraints like phase invariance and normalization, guides training by comparing predictions to the ground truth. Once trained, the model takes an initial wave function as input to efficiently predict its evolution under any global phase transformation. Reverse Schrödingerization then maps the predicted wave function back to interpretable and physically meaningful fields of interest.

Algorithm 1 Phase-averaged and normalization forward evolution for QIDNNF

Input: $\boldsymbol{\psi}^n$, N^{int}, Δt;
Output: $\boldsymbol{\psi}^{n+1}$;
$\boldsymbol{s} \leftarrow 0$; $h \leftarrow 2\pi/N^{\text{int}}$; $\alpha \leftarrow h$;
for $i \leftarrow 0$ to $N^{\text{int}} - 1$ **do**
  $\boldsymbol{s} \leftarrow \boldsymbol{s} + \Delta t \cdot \boldsymbol{\mathcal{G}}_{\boldsymbol{\theta}}\left(\boldsymbol{\psi}^n \mathrm{e}^{\mathrm{i}\alpha}\right)\mathrm{e}^{-\mathrm{i}\alpha}$; ▷ $\boldsymbol{\mathcal{G}}_{\boldsymbol{\theta}}$: Neural network
  $\alpha \leftarrow \alpha + h$;
end for ▷ Numerical phase integration
$\lambda \leftarrow \sqrt{1 - [\mathrm{Im}(\boldsymbol{\psi}^n \bar{\boldsymbol{s}})]^2} - \mathrm{Re}(\boldsymbol{\psi}^n \bar{\boldsymbol{s}})$; ▷ Normalization
$\boldsymbol{\psi}^{n+1} \leftarrow \lambda \boldsymbol{\psi}^n + \boldsymbol{s}$ ▷ Euler method

the initial phase $\alpha = h$. The algorithm then iterates over N^{int} steps, where, in each step, the wave function is modified by a phase factor $\mathrm{e}^{\mathrm{i}\alpha}$ and passed through the neural network $\boldsymbol{\mathcal{G}}_{\boldsymbol{\theta}}$ to compute its contribution to the evolution. This contribution is then adjusted by the conjugate phase factor $\mathrm{e}^{-\mathrm{i}\alpha}$ to ensure proper integration, and the result is accumulated into $\boldsymbol{s}$. After this, a normalization factor λ is computed based on the current wave function and the integrated contributions, ensuring that the wave function retains its unit norm. Finally, the next state of the wave function $\boldsymbol{\psi}^{n+1}$ is calculated using the Euler method by combining the scaled current wave function $\lambda\boldsymbol{\psi}^n$ with the accumulated contributions $\boldsymbol{s}$.

All training data are generated using the RK4 method. For the QIDNNF, the wave function is computed over a specified time interval, with an initial phase angle $\theta = 0$ and a time step size of $\Delta t = 0.01$. Data pairs are then constructed by taking the wave function at the previous time step as the input and the wave function at the subsequent time step as the output. A dataset is built using 400 such data pairs, which are then randomly shuffled. The dataset is split into a training set and a test set in a 9:1 ratio. The training process employs a batch size of 128 and spans 2000 epochs. The network comprises four fully connected layers, each with 16 neurons, and utilizes a Sigmoid activation function. The Adam optimizer is applied with an initial learning rate of 0.001, which decays by a factor of 0.9 every 50 epochs during training.

We conduct an ablation study to investigate the impact of different loss functions on both the training and validation performance of the proposed network [34, 35]. During both phases, we alternate between L1 loss and mean squared error (MSE) loss [36], as depicted in Figure 3. The results show that both loss functions lead to satisfactory convergence, but the network trained with L1 loss exhibits faster convergence on the validation data compared to the network trained with MSE loss. Specifically, L1 loss accelerates the learning process, reaching optimal performance more quickly, whereas MSE loss requires more epochs to achieve similar accuracy. These findings demonstrate that L1 loss not only speeds up the training process but also enhances validation efficiency, making it particularly beneficial for applications.

We investigate the effect of various network configurations on training performance, using L1 loss for both training and validation. As shown in Figure 4(a), the number of neurons,

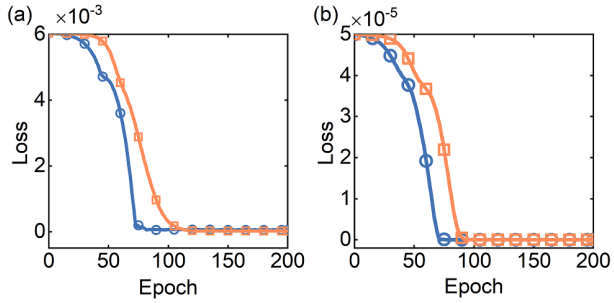

Figure 3 (Color online) Convergence of the validation loss for different loss functions: (a) L1 loss, and (b) MSE loss. The dark blue and orange curves represent training with L1 and MSE loss functions, respectively.

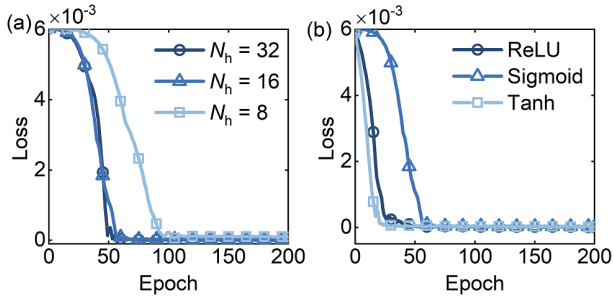

Figure 4 (Color online) Loss convergence for various network configurations: (a) impact of varying the number of neurons in the hidden layers, and (b) influence of different activation functions.

N_h, in the hidden layers can affect the network's ability to learn. Specifically, a limited number of neurons restrict the network's learning capacity. However, as N_h increases, the network rapidly converges to a stable solution, mitigating issues like vanishing gradients, similar to the behavior seen in ResNet architectures. In Figure 4(b), we explore the influence of different activation functions, using $N_\mathrm{h} = 16$. The results demonstrate that Tanh, ReLU, and Sigmoid activation functions all lead to stable convergence.

To further evaluate the performance of QIDNNF, we increase the time step and compare its results with those of the RK4 method. Figure 5 presents the wave function evolution over 200 time steps with $\Delta t = 0.5$. At this larger time step, the RK4 method suffers from numerical dissipation, indicating its limitations in long-term dynamics. In contrast, QIDNNF accurately reproduces the wave function evolution, demonstrating its robustness and stability for long-term predictions. This is attributed to QIDNNF's ability to combine data-driven optimization with embedded physical principles, making it well-suited for applications requiring high fidelity over extended time periods. These results highlight the superiority of QIDNNF in preserving wave function normalization and handling variations in initial phase conditions.

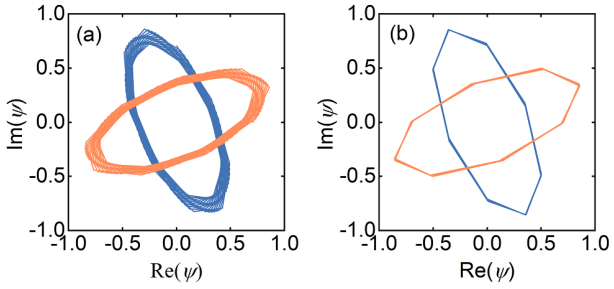

Figure 5 (Color online) Wave function predictions with a large time step of $\Delta t = 0.5$: (a) RK4 and (b) QIDNNF. The dark blue and orange curves represent the trajectories of particles 1 and 2, respectively.

3.2 Comparison of QIDNNF with Neural ODEs and PINNs

We compare the predictive performance of several methods, using the RK4 method with a small time step of $\Delta t = 0.01$ as the reference. The methods considered include QIDNNF, Neural ODEs, and PINNs, with all models trained using L1 loss, the Sigmoid activation function, and a hidden layer size of $N_\mathrm{h} = 16$. Both QIDNNF and Neural ODEs predict the wave function at subsequent time steps based on the wave function at previous time steps, whereas PINNs directly predict the corresponding wave function from the input time.

Figure 6 compares the predictions of wave function evolution from various methods for the same initial global phase angle ($\theta = 0$) as the training data. For PINNs, the loss function consists of both data and physical components, expressed as $\mathcal{L} = \mathcal{L}_\mathrm{data} + \alpha\mathcal{L}_\mathrm{physical}$, where α is a weight coefficient for the physical loss term. When $\alpha = 0$, the physical loss is omitted, resulting in a purely data-driven model. The results reveal differences in performance among the methods. Neural ODEs predict wave function increments using an Euler-based time-stepping scheme. However, relying on repeated incremental updates causes small errors to accumulate over time, leading to noticeable deviations from the true trajectory unless extremely small time steps are used, significantly increasing computational cost. Although PINNs incorporate a physical loss term, the lack of a time integration process hinders their ability to optimize effectively, especially with limited data. By directly predicting specific values instead of incrementally modeling wave function evolution, PINNs fail to capture dynamic behavior accurately, causing lagging predictions. As the loss weight α increases, prioritizing physical consistency over data accuracy, overfitting to physical loss exacerbates trajectory distortion. The combined effects of lagging predictions and trajectory distortion underline the limitations of PINNs in accurately modeling dynamic temporal behaviors, particularly when physical constraints dominate the optimization process. In contrast,

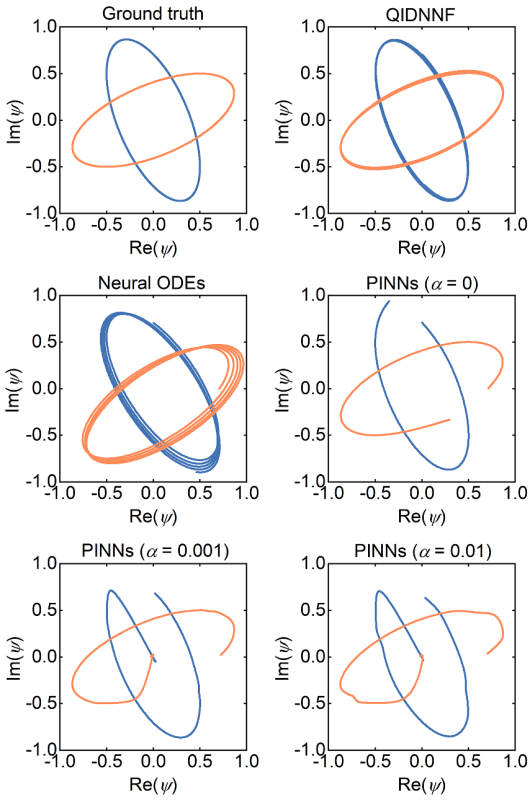

Figure 6 (Color online) Comparison of wave function trajectories predicted by different methods for an initial phase angle of $\theta = 0$ over the time interval $t = 0$ to $t = 20$. The dark blue and orange curves represent the trajectories of particles 1 and 2, respectively.

QIDNNF demonstrates fine performance by combining data-driven learning with embedded physical principles. The predicted wave function trajectories align closely with the ground truth, maintaining both accurate directionality and magnitude. These results underscore QIDNNF's advantages over Neural ODEs and PINNs in accurately modeling wave function evolution.

The preservation of wave function normalization is further analyzed for both Neural ODEs and QIDNNF, as shown in Figure 7. QIDNNF consistently preserves normalization throughout the evolution process, even under varying initial conditions, ensuring that the predictions remain physically consistent. In contrast, Neural ODEs struggle to preserve normalization, as evidenced by their amplitude evolution curves deviating significantly from the normalization ground truth across all initial conditions. This issue stems from the absence of explicit mechanisms to enforce conservation properties during the time-stepping process, resulting in prediction instability and underscoring a critical area for potential improvement.

Figures 6 and 7 illustrate the extrapolation capability of QIDNNF. Although trained on data from the time interval $[0, 4]$, QIDNNF accurately predicts the wave function over

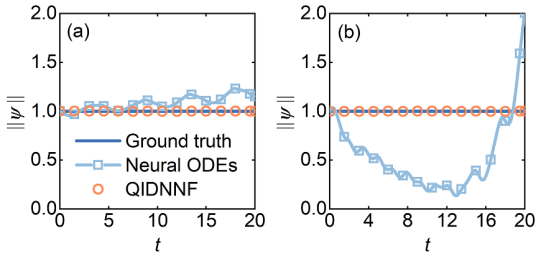

Figure 7 (Color online) Amplitude evolution of the wave function in a linear system: (a) and (b) show results for initial phase angles of 0 and $\pi/3$, respectively. QIDNNF consistently preserves wave function normalization, while Neural ODEs exhibit significant deviations due to their failure to maintain normalization.

the extended time interval $[0, 20]$. Furthermore, with training based on only two consecutive time steps, QIDNNF can predict up to 2000 time steps from an initial condition.

Training time is measured for different sample sizes, and the inference time for 2000 consecutive steps, along with the prediction accuracy under fixed initial conditions, is recorded. The results are summarized in Table 1. The prediction error is defined as $e = \sum_{i=1}^{2} \left|\psi_i^{\mathrm{P}} - \psi_i^{\mathrm{G}}\right| / \sum_{i=1}^{2} \left|\psi_i^{\mathrm{G}}\right|$, where the superscripts "P" and "G" denote the predicted and ground truth values, respectively. Identical training protocols are employed for all methods: each model is trained for 2000 epochs with a batch size of 128, and both training and inference are performed with a fixed time step size of $\Delta t = 0.01$.

A comparative analysis of the statistical results shows that QIDNNF incurs higher training costs than the other two methods for equivalent training sample sizes, due to the inclusion of additional physics-informed computations. However, it achieves significantly lower prediction errors. It can be reasonably concluded that Neural ODE and PINNs would require higher learning costs than QIDNNF to achieve comparable error levels. This is supported by the results in the table, which indicate that models trained with 2000 samples in the baseline methods still fall short of the accuracy that QIDNNF achieves with only 400 samples.

We note that QIDNNF does require a longer inference time compared to Neural ODEs and PINNs, as shown in Table 1. However, we have observed that QIDNNF tends to be more robust with respect to the time step size. By increasing the time step size, it is possible to improve computational efficiency significantly, without sacrificing the accuracy of the predictions.

4 Numerical results

We evaluate the versatility and efficiency of QIDNNF across a wide range of physical problems, including linear and nonlinear systems, low- and high-dimensional frameworks, and

Table 1 A comparison of the training and prediction performance for different methods with varying training sample sizes

	Sample sizes	Training time (s)	Inference time (s)	Error e
Neural ODE	400	44.96	2.67	0.6343
	1000	93.68	2.79	0.2064
	2000	165.66	3.03	0.0561
PINNs ($\alpha = 0.001$)	400	46.98	0.01	0.8893
	1000	107.12	0.01	0.7265
	2000	202.45	0.01	0.3275
QIDNNF	400	78.74	9.11	0.0031
	1000	219.11	8.49	3.9186×10^{-4}
	2000	421.92	9.55	1.1820×10^{-4}

both ODEs and PDEs. The results show QIDNNF's flexibility in handling complex scenarios, such as fluid mechanics, while ensuring physical consistency, computational efficiency, and high accuracy, positioning them as a potential tool for modeling dynamic systems.

4.1 Lagrange vortices

We examine a nonlinear Schrödinger system derived from the Lagrange vortex method (LVM). A detailed explanation of the governing equations and the mathematical formulation is provided in Appendix A1. The system consists of eight vortex particles with initial positions:

$$\begin{cases} \boldsymbol{x} = [0, 0, 0.2, -0.2, 0, 0, 0.2, -0.2], \\ \boldsymbol{y} = [1.7, 1.3, 1.5, 1.5, -1.3, -1.7 - 1.5 - 1.5], \end{cases}$$

where each particle is assigned a vortex strength $\Gamma_i = 1/4$ $(i = 1, 2, \ldots, 8)$.

In this numerical experiment, the RK4 method with a small time step is employed as the reference solution. Using an initial phase of 0, we generate 2000 training samples with a time step of $\Delta t = 0.02$. The network architecture is similar to that used in the previous example, consisting of 5000 training epochs. Each epoch is trained with a batch size of 128, starting with an initial learning rate of 0.1, which is reduced by a factor of 0.9 every 100 epochs.

Figure 8 illustrates the flow field at different time, visualized using 1000000 passive particles induced by vortex particles, with orange circles marking their positions. The trained network predicts 4000 steps with a time step of $\Delta t = 0.02$, and a backward-tracking algorithm [37] is used to visualize the flow field. The first, second, and third rows represent the results for ground truth, QIDNNF, and Neural ODEs, respectively, while the first, second, and third columns correspond to time $t = 10$, 40, and 80. The results show that QIDNNF provides highly accurate long-term predictions, closely matching the ground truth. Training on data spanning the time interval $[0, 40]$, QIDNNF accurately predicts

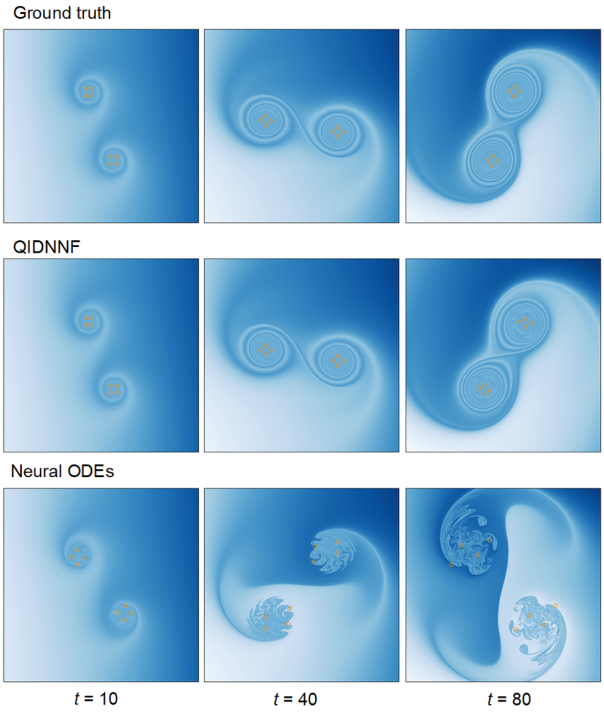

Figure 8 (Color online) Flow field visualized at different time using 1000000 passive particles induced by vortex particles, with orange circles marking the positions of the vortex particles. The first, second, and third rows show the results for ground truth, QIDNNF, and Neural ODEs, respectively. The first, second, and third columns correspond to time $t = 10$, 40, and 80.

results over the extended time interval $[0, 80]$, illustrating its robust extrapolation capability, even for highly nonlinear problems. In contrast, Neural ODEs exhibit significant errors due to the absence of physical conservation principles during training, leading to inaccuracies in particle evolution and errors in the reconstructed flow field.

4.2 Nonlinear wave

Nonlinear wave equations, such as the 1D nonlinear Schrödinger equation (NLS), govern many significant physical phenomena, including Bose-Einstein condensates and

plasma waves [15]. As detailed in Appendix A2, we consider an initial condition characterized by a hyperbolic secant distribution over the domain $x \in [-5, 5]$, specifically $\psi_0 = 2\,\mathrm{sech}(x)$, with periodic boundary conditions $\psi(-5, t) = \psi(5, t)$ and $\psi_x(-5, t) = \psi_x(5, t)$.

Using the methodology outlined in ref. [15], the spatial domain is discretized into 256 mesh points, and 200 training datasets of normalized wave functions are generated over the time interval $t \in [0, \pi/2]$ with a time step of $\Delta t = \pi/400$. A Unet model [38] with 1D convolutional neural networks is employed for training, as described in Table 2. Training is performed using the Adam optimizer with an initial learning rate of 0.001, which is reduced by 10% every 10 epochs, and a batch size of 8.

QIDNNF is applied to predict the wave function evolution over the extended time interval $t \in [0, \pi]$, thereby validating their extrapolation capability in solving PDEs. The results of continuous predictions over 400 time steps are presented in Figure 9. The top part of the figure compares the spatial distribution of $|\psi|$ predicted by QIDNNF with the ground truth at three specific time points, arranged from left to right. This comparison highlights the models accuracy in capturing the dynamics of the wave function. The bottom part illustrates the long-term prediction of the spatial distribution of $|\psi|$, demonstrating the stability and robustness of QIDNNF in modeling extended temporal behavior.

Table 2 Unet structure information

	1D Unet	2D Unet	3D Unet
Input	2×256	4×128×128	4×64×32×32
Encoder layer 1	64	64	64
Encoder layer 2	128	128	128
Encoder layer 3	256	256	256
Encoder layer 4	512	512	512
Decoder layer 1	256	256	256
Decoder layer 2	128	128	128
Decoder layer 3	64	64	64
Output later	2×256	4×128×128	4×64×32×32
Periodic extension	2×512	4×380×380	4×74×62×62
Kernal size	5×1	5×5	3×5×5

4.3 Incompressible Schrödinger flow

The incompressible Schrödinger flow (ISF) problem as detailed in Appendix A3 is reformulated to integrate seamlessly with the QIDNNF, enabling efficient and physically consistent solutions for vortical flows. The framework is further extended to 2D and 3D scenarios using tailored Unet architectures, with their layer configurations and parameters detailed in Table 2. These models provide accurate, scalable solutions for complex vortical flow dynamics.

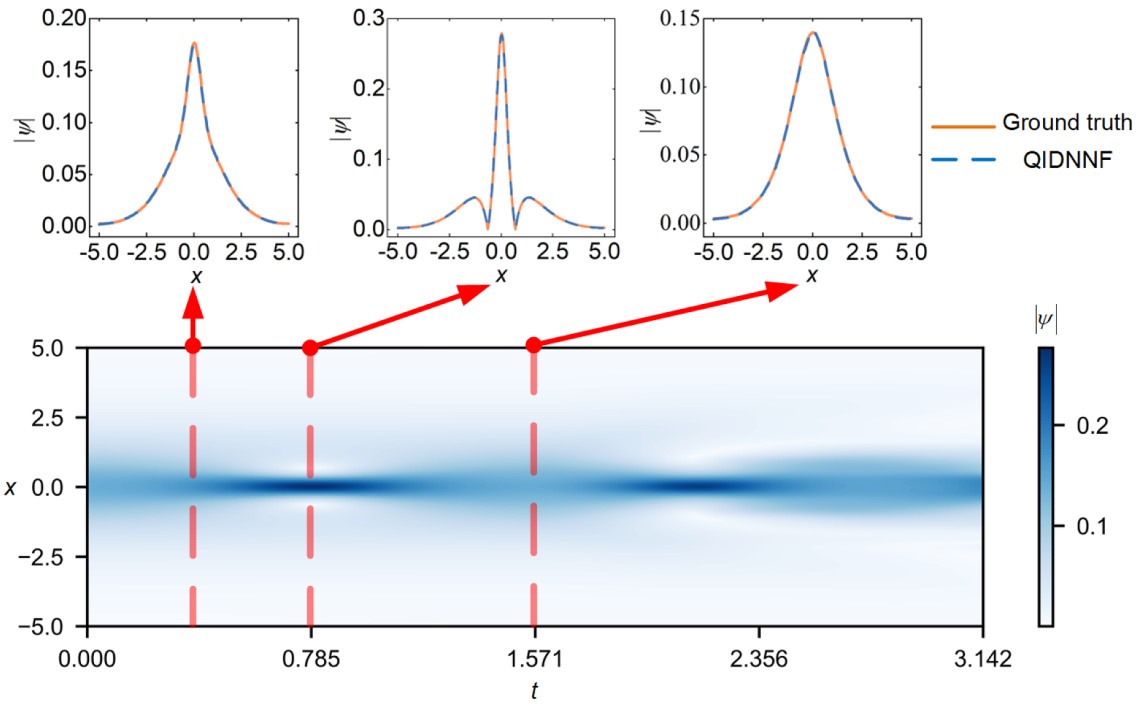

Figure 9 (Color online) Visualization of wave function evolution: the long-term spatial distribution of $|\psi|$ predicted by QIDNNF is shown below, with comparisons to the ground truth at three specific time points displayed above, arranged sequentially from left to right.

2D flow The evolution of a 2D flow generated by four concentrated vortex centers is studied as a benchmark case. The initial wave functions for the ISF are initialized using algorithm 2, which computes the wave functions ψ_1^0 and ψ_2^0 over the spatial domain $(x, y) \in [0, 2\pi]^2$. This process involves calculating radial distances, applying exponential decay functions, and performing normalization and projection operations to ensure compatibility with the ISF governing equations.

Algorithm 2 Initialization of 2D flow

Input: $(x, y) \in [0, 2\pi]^2$;

Output: ψ_1^0, ψ_2^0;

$r_{x,1} = \frac{4}{\pi}(x-\pi)$; $r_{y,1} = \frac{4}{\pi}(y-1.4\pi)$; $r_1^2 = r_{x,1}^2 + r_{y,1}^2$;

$r_{x,2} = -\frac{4}{\pi}(x-\pi)$; $r_{y,2} = -\frac{4}{\pi}(y-0.6\pi)$; $r_2^2 = r_{x,2}^2 + r_{y,2}^2$;

$d_1 = \exp\left\{-\left(r_1^2/9\right)^4\right\}$; $d_2 = \exp\left\{-\left(r_2^2/9\right)^4\right\}$;

$\psi_1^* = \frac{2r_{x,1}d_1 + \mathrm{i}\left(1+r_1^2-2d_1\right)}{1+r_1^2} \cdot \frac{2r_{x,2}d_2 + \mathrm{i}\left(1+r_2^2-2d_2\right)}{1+r_2^2}$; $\psi_2^* = 0.01$;

$\left(\psi_1^0, \psi_2^0\right) =$ Normalization $\left(\psi_1^*, \psi_2^*\right)$ ▷ Algorithm a1

$\left(\psi_1^0, \psi_2^0\right) =$ Projection $\left(\psi_1^0, \psi_2^0\right)$ ▷ Algorithm a1

Training data are generated using the ISF solver as the ground truth, with a time step of $\Delta t = 0.002$ and a mesh resolution of 128×128. A total of 1000 datasets are employed for training, with the Adam optimizer used to minimize the loss function. The training process begins with an initial learning rate of 0.001, which decays by a factor of 0.9 every 20 epochs, over a total of 1000 epochs to ensure convergence.

The QIDNNF is utilized to predict the evolution of the flow field over 200 time steps. Figure 10 compares the predicted flow fields with the ground truth at selected time points. The background illustrates the vorticity in the z-direction through white-to-blue contours, while the velocity field is represented by arrows, with color indicating the magnitude of velocity, ranging from white to orange. The results validate the precise and stable simulation capabilities of QIDNNF for 2D fluid problems, particularly highlighting its ability for long-term continuous extrapolation.

3D smoke The application of deep learning to 3D PDEs remains an evolving research area. In this work, we extend the QIDNNF to tackle 3D problems, specifically focusing on a smoke simulation scenario. The setup and data used in this study are adapted from the work of Chern et al. [39].

For this simulation, we use the solutions from the ISF solver as the ground truth, with a time step of $\Delta t = 1/24$ and a mesh resolution of $64 \times 32 \times 32$. The dataset consists of 500 pairs of sequential time steps, which are employed for training. The model is trained using the Adam optimizer with an initial learning rate of 0.001, decayed by 10% every 10 epochs, and the training process spans 1000 epochs.

Once trained, the model is used to predict the evolution of the flow field over 200 time steps, with the results visualized using 20000000 passive particles. Figure 11 illustrates the dynamic interaction of two vortex rings, where each alternately passes through the other in a periodic motion. This

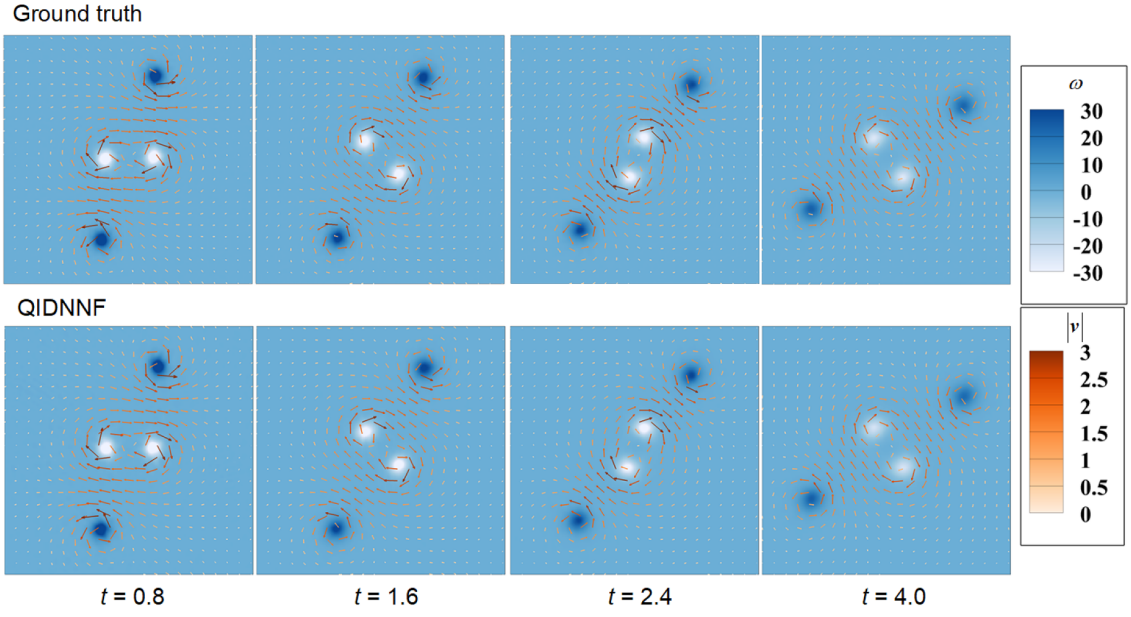

Figure 10 (Color online) Visualization of the 2D flow field. The background shows the vorticity in the z-direction, represented by white-to-blue contours, while the velocity field is depicted by arrows colored according to the velocity magnitude, with white-to-orange coloring. The first and second rows show the results for ground truth and QIDNNF, respectively. The four columns correspond to time $t = 0.8$, 1.6, 2.4, and 4.0.

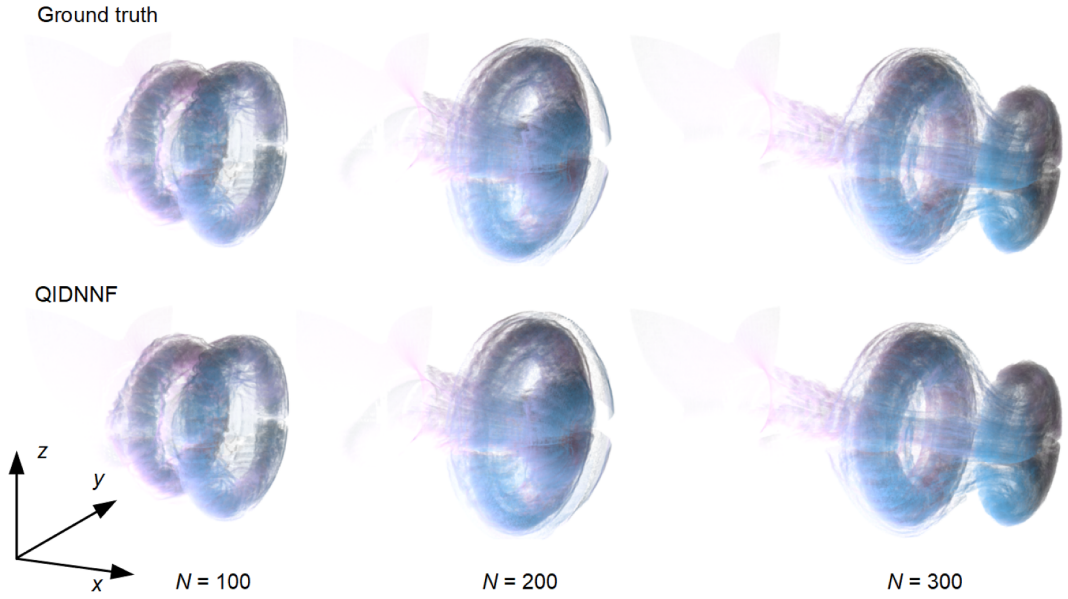

Figure 11 (Color online) Visualization of 3D leapfrogging smoke at different time steps, rendered with 20000000 passive particles. The results illustrate that QIDNNF accurately captures the leapfrogging phenomenon induced by the alternating motion of two vortex rings, with predictions closely aligned with the ground truth. The first and second rows show the results for ground truth and QIDNNF, respectively. The first, second, and third columns correspond to time steps N = 100, 200, and 300.

leapfrogging behavior arises from the velocity fields generated by the rings: as one advances, it induces a flow that propels the trailing ring forward, allowing it to pass through the leader. The roles then reverse, creating a visually recurring phenomenon. The predictions, generated by QIDNNF, closely align with the ground truth, accurately capturing the dynamic behavior of the flow.

5 Conclusion

This study presents QIDNNF, a novel deep learning framework that integrates quantum principles, including phase invariance and normalization, to solve Schrödinger's equation and its Schrödingerization-derived forms. Building on Neural ODEs, QIDNNF enhances its functionality with phase integral and normalization operations, enabling efficient solutions for a wide range of PDEs.

Through numerical experiments, QIDNNF demonstrates improvements over existing deep learning methods, offering higher accuracy, greater stability, and faster convergence. These advantages are further validated across several test cases, including vortex dynamics, nonlinear wave propagation, and incompressible Schrödinger flow, showing QIDNNF's effectiveness in simulating complex physical phenomena while maintaining physical consistency. In addition, QIDNNF shows promising potential in extrapolation capability. It is able to predict results over longer time intervals based on training data from shorter time spans and can provide accurate predictions over multiple consecutive time steps, even when trained on just two sequential time steps.

While the results are promising, further research is needed to explore QIDNNF's applicability to a wider range of systems. One key avenue for future work is the integration of QIDNNF with quantum computing architectures, which could further improve its efficiency, accuracy, and robustness, allowing it to tackle increasingly complex physical problems.

This work was supported by the National Natural Science Foundation of China (Grant Nos. 12302294, and 12432010), the National Key R&D Program of China (Grant No. 2023YFB4502600), and the Research Fund of the National Key Laboratory of Aerospace Physics in Fluids (Grant No. 2024-APF-KFZD-07). Jia Xiong also acknowledges the support from Zhejiang University's "Qi Zhen Wen Xue" Innovation Platform.

Conflict of interest The authors declare that they have no conflict of interest.

1 L. C. Evans, *Partial Differential Equations* (American Mathematical Society, Providence, 2022).
2 W. Gautschi, *Numerical Analysis* (Springer Science and Business Media, Boston, 2011).
3 X. Wu, L. Ma, and Z. Zuo, Aerosp. Sci. Tech. **149**, 109152 (2024).
4 M. Li, J. Zhao, N. Wang, and S. Chen, Comput. Methods Appl. Mech. Eng. **380**, 113793 (2021).
5 X. Li, and S. Li, Math. Comput. Simul. **203**, 538 (2023).
6 J. Tang, G. Chen, and Y. Ge, Comput. Struct. **275**, 106920 (2023).
7 Y. H. Huang, Z. H. Xia, M. P. Wan, Y. P. Shi, and S. Y. Chen, Sci. China-Phys. Mech. Astron. **62**, 44711 (2019).
8 C. Wang, Z. P. Qiu, and D. Wu, Sci. China-Phys. Mech. Astron. **57**, 698 (2014).
9 B. Li, J. Garicano-Mena, and E. Valero, J. Comput. Phys. **468**, 111495 (2022).

10 L. Chen, X. Zhong, H. Li, J. Wu, B. Lu, D. Chen, S. P. Xie, L. Wu, Q. Chao, C. Lin, et al., Nat. Commun. **15**, 6425 (2024).
11 W. Chen, J. Kou, W. Yang, and S. Pan, Aerosp. Sci. Tech. **150**, 109175 (2024).
12 L. Herrmann, and S. Kollmannsberger, Comput Mech **74**, 281 (2024).
13 Z. Zhang, Y. Li, W. Zhou, and W. Yao, Sci. China-Phys. Mech. Astron. **68**, 244611 (2025).
14 G. E. Karniadakis, I. G. Kevrekidis, L. Lu, P. Perdikaris, S. Wang, and L. Yang, Nat. Rev. Phys. **3**, 422 (2021).
15 M. Raissi, P. Perdikaris, and G. E. Karniadakis, J. Comput. Phys. **378**, 686 (2019).
16 S. Cai, C. Gray, and G. E. Karniadakis, IEEE Trans. Instrum. Meas. **73**, 1 (2024).
17 S. P. S. Gogate, M. C. Bharathi, and R. B. Kudenatti, J. Heat Transfer **143**, 060801 (2021).
18 J. Bai, T. Rabczuk, A. Gupta, L. Alzubaidi, and Y. Gu, Comput. Mech. **71**, 543 (2023).
19 Y. Zhou, H. Wang, B. Wu, L. G. Wang, and X. Chen, Particuology **96**, 126 (2025).
20 S. Cai, Z. Mao, Z. Wang, M. Yin, and G. E. Karniadakis, Acta Mech. Sin. **37**, 1727 (2021).
21 X. Meng, Z. Li, D. Zhang, and G. E. Karniadakis, Comput. Methods Appl. Mech. Eng. **370**, 113250 (2020).
22 R. T. Q. Chen, Y. Rubanova, J. Bettencourt, and D. Duvenaud, *Neural ordinary differential equations*, in: *Advances in Neural Information Processing Systems 31 (NeurIPS 2018)* (Curran Associates, Inc., New York, 2018), pp. 6571-6583.
23 K. He, X. Zhang, S. Ren, and J. Sun, in *Deep residual learning for image recognition*: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, 2016, pp. 770-778.
24 Q. Zhu, X. Li, and W. Lin, Chaos-An Interdiscip. J. Nonlinear Sci. **33**, 031101 (2023).
25 S. Greydanus, M. Dzamba, and J. Yosinski, *Hamiltonian neural networks* in: *Advances in Neural Information Processing Systems 32 (NeurIPS 2019)* (Curran Associates, Inc., New York, 2019), pp. 15379-15389.
26 Y. Tong, S. Xiong, X. He, S. Yang, Z. Wang, R. Tao, R. Liu, and B. Zhu, Numer. Meth Eng. **125**, e7406 (2024).
27 J. Tang, L. Qian, J. Ma, L. Chen, G. Chen, Z. Chen, and W. Huang, Knowledge-Based Syst. **296**, 111853 (2024).
28 M. Huang, Z. Du, C. Liu, Y. Zheng, T. Cui, Y. Mei, X. Li, X. Zhang, and X. Guo, Extreme Mech. Lett. **56**, 101887 (2022).
29 S. Jin, N. Liu, and Y. Yu, Phys. Rev. Lett. **133**, 230602 (2024); Phys. Rev. A. **108**, 032603 (2023).
30 S. Jin, and N. Liu, Quantum Sci. Technol. **9**, 035047 (2024).
31 S. Jin, X. Li, N. Liu, and Y. Yu, J. Comput. Phys. **498**, 112707 (2024).
32 A. Messiah, *Quantum Mechanics* (Courier Corporation, New York, 2014).
33 Z. Y. Chen, Q. Zhou, C. Xue, X. Yang, G. C. Guo, and G. P. Guo, Sci. Bull. **63**, 964 (2018).
34 Y. Tong, S. Xiong, X. He, G. Pan, and B. Zhu, J. Comput. Phys. **437**, 110325 (2021).
35 S. Xiong, Y. Tong, X. He, S. Yang, C. Yang, and B. Zhu, in *Nonseparable symplectic neural networks*: *Proceedings of International Conference on Learning Representations (ICLR 2021)*, Virtual, 2021, pp. 1-14.
36 Y. Bengio, I. Goodfellow, and A. Courville, *Deep Learning* (MIT Press, Cambridge, 2017).
37 Y. Yang, D. I. Pullin, and I. Bermejo-moreno, J. Fluid Mech. **654**, 233 (2010).
38 O. Ronneberger, P. Fischer, and T. Brox, in *U-Net: Convolutional networks for biomedical image segmentation*: *Proceedings of International Conference on Medical Image Computing and Computer-Assisted Intervention (MICCAI 2015)* (Springer, Cham, 2015), pp. 234-241.
39 A. Chern, F. Knöppel, U. Pinkall, P. Schröder, and S. Weißmann, ACM Trans. Graph. **35**, 1 (2016).
40 S. Xiong, R. Tao, Y. Zhang, F. Feng, and B. Zhu, ACM Trans. Graph. **40**, 1 (2021).

Appendix

A1 Lagrangian vortex method

The NS equation represented by vorticity $\boldsymbol{\omega} = \nabla \times \boldsymbol{u}$ is

$$\frac{\mathrm{D}\boldsymbol{\omega}}{\mathrm{D}t} = (\boldsymbol{\omega} \cdot \nabla)\,\boldsymbol{u} + \nu \nabla^2 \boldsymbol{\omega} + \nabla \times \boldsymbol{f}, \tag{a1}$$

where $\boldsymbol{u}$ is the veolocity, ν indicates the kinematic viscosity, and $\boldsymbol{f}$ represents the body force. Eq. (a1) discretized by the LVM into the ODEs representing the n_d-dimensional position evolution of N_p vorticity particles [40]:

$$\begin{cases} \dfrac{\mathrm{d}\boldsymbol{\Gamma}_i}{\mathrm{d}t} = \boldsymbol{\gamma}_i, \\ \dfrac{\mathrm{d}\boldsymbol{x}_i}{\mathrm{d}t} = \dfrac{1}{2\pi(n_d-1)} \displaystyle\sum_{j=1, j\neq i}^{N_p} \frac{\boldsymbol{\Gamma}_i \times \left(\boldsymbol{x}_i - \boldsymbol{x}_j\right)}{\left|\boldsymbol{x}_i - \boldsymbol{x}_j\right|^{n_d}} + \boldsymbol{v}_i, \end{cases} \tag{a2}$$

where $\boldsymbol{x}_i \in \mathbb{R}^{n_d}$ denotes the spatial position of the vortex particle i ($i = 1, 2, \cdots, N_p$). $\boldsymbol{\Gamma}_i$ is the circulation integral of $\boldsymbol{\omega}$ for the i-th computing element, indicating the strength of the vortex particle. $\boldsymbol{\gamma}_i$ and $\boldsymbol{v}_i$ are strength change rate and drift velocity, respectively, neither of which are considered in an ideal 2D flow. We use a complex variable $\phi_i = x_i + \mathrm{i}y_i$ to describe the 2D vortex particle position, so that eq. (a2) can be rewritten as

$$\frac{\mathrm{d}\phi_i}{\mathrm{d}t} = \frac{1}{2\pi} \sum_{j=1, j\neq i}^{N_p} \frac{\Gamma_i \times \left(\phi_i - \phi_j\right)}{\left|\phi_i - \phi_j\right|^2}. \tag{a3}$$

Eq. (a3) has been preliminarily displayed in the form of Schrödinger, and it is not difficult to find that it has the global phase invariance property described in Property 1. Specifically, if all phases ϕ_i are shifted by the same constant phase α, i.e., $\phi_i \to \phi_i + \alpha$, the equation remains valid. To obtain the standard Schrödinger equation form, we simply need to normalize ϕ_i.

A2 Nonlinear Schrödinger equation

The wave function of the 1D NLS satisfies the governing equation:

$$\begin{cases} \mathrm{i}\dfrac{\partial \psi}{\partial t} + \dfrac{1}{2}\dfrac{\partial^2 \psi}{\partial x^2} + |\psi|^2 \psi = 0, \\ x \in [x_0, x_1], t \in [t_0, t_1], \end{cases} \tag{a4}$$

along with the initial condition $\psi(x, t = t_0) = \psi_0(x)$, and boundary conditions $\mathcal{B}(x_0, x_1, t) = 0$.

A3 Incompressible Schrödinger flow

Chern et al. [39] utilize a dual-wave function approach $\boldsymbol{\psi} = [\psi_1, \psi_2]$ to reformulate an incompressible fluid system, known as the ISF, where the flow field's velocity $\boldsymbol{v} = (v_1, v_2, v_3)^{\mathrm{T}}$ and dimensionless density ρ are reexpressed as

$$v_i = \hbar \left\langle \frac{\partial \boldsymbol{\psi}}{\partial x_i}, \mathrm{i}\boldsymbol{\psi} \right\rangle_{\mathbb{R}}, \quad i = 1, 2, 3, \ \rho = \|\boldsymbol{\psi}\|, \tag{a5}$$

where $\langle \boldsymbol{\alpha}, \boldsymbol{\beta} \rangle_{\mathbb{R}} = \mathrm{Re}\left(\langle \boldsymbol{\alpha}, \boldsymbol{\beta} \rangle_{\mathbb{C}}\right) = \mathrm{Re}(\overline{\alpha}_1\beta_1 + \overline{\alpha}_2\beta_2)$. Employing mapping eq. (a5), the incompressible fluid equations can be transformed into a Schrödinger equation:

$$\mathrm{i}\hbar \frac{\partial \boldsymbol{\psi}}{\partial t} = -\frac{\hbar^2}{2}\Delta\boldsymbol{\psi} + p\boldsymbol{\psi}, \tag{a6}$$

where p indicates pressure potential. The incompressibility condition $\nabla \cdot \boldsymbol{v} = 0$ is enforced through the constraint imposed by the wave function, $\langle \Delta\boldsymbol{\psi}, \mathrm{i}\boldsymbol{\psi} \rangle_{\mathbb{R}} = 0$. The ISF equations are typically solved using a projection method, as concisely outlined in algorithm a1.

Algorithm a1 ISF solver

Input: $\boldsymbol{\psi}^{n-1}, \hbar, \Delta t$;
Output: $\boldsymbol{\psi}^{n}, \boldsymbol{v}^{n}$;
1: $\boldsymbol{\psi}^{*} \leftarrow \boldsymbol{\psi}^{n-1} + \Delta t \cdot \mathrm{i}\frac{\hbar}{2}\Delta\boldsymbol{\psi}^{n-1}$; ▹ Wave function predictor (FFT)
2: $v_i^{*} \leftarrow \left\langle \frac{\partial \boldsymbol{\psi}^{*}}{\partial x_i}, \mathrm{i}\boldsymbol{\psi}^{*} \right\rangle_{\mathbb{R}}$; ▹ Velocity predictor
3: $\boldsymbol{\psi}^{*} \leftarrow \boldsymbol{\psi}^{*} / \|\boldsymbol{\psi}^{*}\|$; ▹ Normalization
4: $\Delta\phi \leftarrow \nabla \cdot \boldsymbol{v}^{*}$; $\boldsymbol{\psi}^{n} \leftarrow \mathrm{e}^{\mathrm{i}\frac{\phi}{\hbar}}\boldsymbol{\psi}^{*}$; ▹ Pressure projection
5: $\boldsymbol{v}^{n} \leftarrow \left\langle \frac{\partial \boldsymbol{\psi}^{n}}{\partial x_i}, \mathrm{i}\boldsymbol{\psi}^{n} \right\rangle_{\mathbb{R}}$