mathematics

Article

Causality-Aware Training of Physics-Informed Neural Networks for Solving Inverse Problems

Jaeseung Kim [†] **and Hwijae Son** [*,†]

Department of Mathematics, Konkuk University, Seoul 05029, Republic of Korea; dgd05037@konkuk.ac.kr
* Correspondence: hwijaeson@konkuk.ac.kr
† These authors contributed equally to this work.

Abstract: Inverse Physics-Informed Neural Networks (inverse PINNs) offer a robust framework for solving inverse problems governed by partial differential equations (PDEs), particularly in scenarios with limited or noisy data. However, conventional inverse PINNs do not explicitly incorporate causality, which hinders their ability to capture the sequential dependencies inherent in physical systems. This study introduces Causal Inverse PINNs (CI-PINNs), a novel framework that integrates directional causality constraints across both temporal and spatial domains. Our approach leverages customized loss functions that adjust weights based on initial conditions, boundary conditions, and observed data, ensuring the model adheres to the system's intrinsic causal structure. To evaluate CI-PINNs, we apply them to three representative inverse PDE problems, including an inverse problem involving the wave equation and inverse source problems for the parabolic and elliptic equations, each requiring distinct causal considerations. Experimental results demonstrate that CI-PINNs significantly improve accuracy and stability compared to conventional inverse PINNs by progressively enforcing causality-driven conditions and data consistency. This work underscores the potential of CI-PINNs to enhance robustness and reliability in solving complex inverse problems across diverse physical domains.

Keywords: physics-informed neural networks (PINNs); inverse problems; respecting causality

MSC: 68T01; 68T07; 35R30

check for updates

Received: 21 February 2025
Revised: 21 March 2025
Accepted: 22 March 2025
Published: 24 March 2025

Citation: Kim, J.; Son, H. Causality-Aware Training of Physics-Informed Neural Networks for Solving Inverse Problems. *Mathematics* **2025**, *13*, 1057. https://doi.org/10.3390/math13071057

1. Introduction

Recent breakthroughs in deep learning have transformed numerous fields, including computer vision and natural language processing [1–7]. In parallel, partial differential equations (PDEs) and delay differential equations (DDEs) have long been essential tools for modeling nonlinear dynamics in natural and engineered systems. These equations capture complex phenomena such as turbulence, pattern formation, and chaotic behavior and have been the subject of extensive theoretical and numerical investigations. Recent studies have provided significant advancements in this field, as demonstrated in works such as [8–11]. Furthermore, recent investigations have extended our understanding of nonlinear and delay systems. Xiuli Xu [8] analyzed the asymptotic limit of the Navier–Stokes–Poisson–Korteweg system in a half-space, providing key insights into boundary layer phenomena and long-term dynamics in complex fluid systems. In the realm of geophysical flows, Boling Guo [10] studied the decay of solutions to a two-layer quasi-geostrophic model, elucidating the mechanisms of energy dissipation and stabilization in stratified environments. Moreover, Wentao Wang has made significant contributions to the stability analysis of delay systems; his work on the global exponential stability of periodic solutions for inertial delayed BAM neural networks [9] and his study employing the characteristics method to investigate delayed inertial neural networks [11] offer rigorous frameworks to ensure robustness in systems affected by delays. These studies deepen our theoretical understanding and underscore the importance of integrating accurate physical modeling into data-driven approaches.

This progress has given rise to Physics-Informed Neural Networks (PINNs), which incorporate governing equations, initial conditions, and boundary conditions directly into the neural network's loss function [12,13]. By leveraging automatic differentiation, PINNs accurately compute the necessary derivatives, ensuring that the learned solutions conform to the underlying physics even when data are sparse or noisy. In this framework, the unknown solution of a PDE is represented as a deep neural network, and all physical constraints are enforced simultaneously through a composite loss function [12–14]. PINNs have proven effective for solving forward problems such as fluid dynamics, heat conduction, and electromagnetics [13,15,16]. PINNs compute the solutions to the PDEs when all necessary conditions are provided.

Inverse problems governed by partial differential equations (PDEs), where one must infer unknown quantities in the system from limited measurements, are ubiquitous in science and engineering, with applications spanning material characterization to biomedical imaging. However, these problems are inherently challenging due to their ill-posed nature and the need to adhere to fundamental physical laws. Inverse PINNs address these challenges by incorporating data loss terms that align predictions with sparse and noisy observations while satisfying the physical principles. Recent progress in diverse scientific problems, including material science, biomedical imaging, and system identification, exhibits the inverse PINNs' effectiveness [13,17–20]. Additionally, Bayesian PINNs have been introduced to incorporate prior knowledge and quantify uncertainty in inverse problems, providing a probabilistic framework for PDE-constrained learning [21]. Meanwhile, sparsity-aware neural networks enforce parsimonious representations by penalizing unnecessary parameters, which can be particularly advantageous for high-dimensional or ill-posed systems [22]. Integrating these approaches with causal training strategies may further enhance the robustness and interpretability of inverse PDE solvers.

Despite their effectiveness, both conventional and inverse PINNs share a critical limitation: they often fail to enforce the inherent causal structure of physical systems, especially in time-dependent problems. Temporal causality mandates that solutions evolve naturally from initial conditions and progress continuously. Without enforcing this principle, training may converge to non-physical or unstable solutions. Recent studies have introduced causal training frameworks [23] and augmented Lagrangian methods [24] to tackle this issue by prioritizing learning during early time steps or near boundaries. More recently, the training paradigm that respects temporal causality has been developed by incorporating forward numerical solvers and multiple neural networks [25,26]. However, researchers have not yet adequately adapted these approaches for inverse problems, which require balancing the influence of additional observational data with physical constraints because current methods focus primarily on temporal causality. This limitation arises from the restrictive design of the causal weights, which motivates the current work.

In this paper, we propose a novel framework, Causal Inverse PINNs (CI-PINNs), a causality-respecting training framework of PINNs for solving inverse problems. Unlike conventional approaches, CI-PINNs integrate multidirectional causal weights—covering temporal, spatial, and data dimensions—directly into the loss function. This innovative strategy enables the model to prioritize supervised regions, such as initial and boundary conditions and observational data, thereby enforcing the natural causal structure of the underlying physical system. Consequently, our framework produces more stable and physically consistent reconstructions.

However, the development and application of CI-PINNs come with several technical challenges. First, the design and tuning of the causal weights require careful consideration; the weights must accurately reflect the significance of various regions in the domain while ensuring a smooth transition from supervised to unsupervised areas. In particular, selecting an optimal causality parameter (e.g., ϵ) is critical, as inappropriate values can lead to insufficient propagation of supervision or cause an overly rapid convergence that masks the true dynamics of the system. The computational overhead introduced by computing multi-dimensional causal weights can be nontrivial, especially when scaling to high-dimensional or more complex physical problems. Addressing these challenges is essential for ensuring that the benefits of the proposed method are realized in practical applications.

We organize this paper as follows. First, we offer a comprehensive review of PINNs by examining the core mechanisms used to solve forward problems and exploring the extensions developed for inverse problems. This review synthesizes insights from seminal works (e.g., [12,27–29]) as well as modern reviews on physics-informed machine learning. Second, we offer an in-depth examination of state-of-the-art developments in inverse PINNs and related parameter identification methods, incorporating recent advances in Bayesian approaches and sparse identification techniques as detailed in [21,22]. Third, we introduce the concept of CI-PINNs—a novel methodology that leverages temporal, spatial, and data causal weights to balance residual minimization. Building on recent causal training strategies [23], we propose an approach that addresses the imbalance between observational data and physical constraints. Finally, we validate the proposed framework through benchmark inverse problems, including an inverse problem for the wave equation, a parabolic inverse source problem, and an elliptic inverse source problem. The experimental results demonstrate substantial improvements in convergence behavior and reconstruction fidelity compared to conventional inverse PINNs.

This work deepens our understanding of how physics-informed machine learning can address challenging inverse problems. It advances the state of the art through a causality-aware training paradigm. By explicitly enforcing the sequential structure inherent in time-dependent and spatially distributed processes, our framework provides a robust, generalizable solution for a broad spectrum of scientific and engineering applications.

2. Preliminaries

2.1. Physics-Informed Neural Networks (PINNs)

Jo et al. [27] proposed PINNs to infer the solutions of partial differential equations (PDEs). Consider a generic time-dependent PDE defined as:

$$u_t + \mathcal{N}[u] = 0, \qquad x \in D, \quad t \in [0,T], \tag{1}$$

$$I[u] = g(x), \qquad x \in D, \tag{2}$$

$$B[u] = h(x,t), \qquad x \in \partial D, \quad t \in [0,T], \tag{3}$$

where $\mathcal{N}[\cdot]$ denotes a differential operator, and $I[\cdot]$ and $B[\cdot]$ represent the initial and boundary operator, respectively. In the PINN literature, the solution to the PDEs (1)–(3) is approximated using a neural network $u_\theta(x,t)$, where θ denotes the vector consisting of all trainable parameters, including weights and biases. To ensure that $u_\theta(x,t)$ satisfies the PDE, the model is trained by minimizing a composite loss function evaluated over the (x,t) grid. This loss function is designed to enforce the conditions in Equations (1)–(3) [12,13].

Specifically, the training process minimizes the following Mean Squared Error (MSE)-based loss:

$$\mathcal{L}(\theta) = \lambda_r \mathcal{L}_r(\theta) + \lambda_{tb}\mathcal{L}_{tb}(\theta) + \lambda_{sb}\mathcal{L}_{sb}(\theta).$$

with

$$\begin{aligned}
\mathcal{L}_r(\theta) &= \frac{1}{N_r}\sum_{i=1}^{N_r}\left|\frac{\partial u_\theta}{\partial t}(x_r^i,\ t_r^i) + \mathcal{N}[u_\theta](x_r^i,\ t_r^i)\right|^2, \\
\mathcal{L}_{tb}(\theta) &= \frac{1}{N_{tb}}\sum_{i=1}^{N_{tb}}\left|I[u_\theta](x_{tb}^i, 0) - g(x_{tb}^i)\right|^2, \\
\mathcal{L}_{sb}(\theta) &= \frac{1}{N_{sb}}\sum_{i=1}^{N_{sb}}\left|B[u_\theta](x_{sb}^i, t_{sb}^i) - h(x_{sb}^i)\right|^2.
\end{aligned} \tag{4}$$

Here, $\mathcal{L}_r(\theta)$ is the residual loss that ensures the PDE holds at interior points, $\mathcal{L}_{tb}(\theta)$ enforces the initial conditions, and $\mathcal{L}_{sb}(\theta)$ enforces the boundary conditions.

The grid points $\{x_r^i,\ t_r^i\}_{i=1}^{N_r}$, $\{x_{tb}^i,\ t_{tb}^i\}_{i=1}^{N_{tb}}$, and $\{x_{sb}^i,\ t_{sb}^i\}_{i=1}^{N_{sb}}$ are sampled from the domain. In our case, these points are uniformly sampled such that $\{x_r^i,\ t_r^i\}_{i=1}^{N_r} \in D \times [0,\ T]$, $\{x_{tb}^i,\ t_{tb}^i\}_{i=1}^{N_{tb}} \in D \times \{0\}$, and $\{x_{sb}^i,\ t_{sb}^i\}_{i=1}^{N_{sb}} \in \partial D \times [0, T]$.

To efficiently compute gradients concerning input variables or the network parameters θ, PINNs leverage automatic differentiation. This capability is crucial for accurately calculating the PDE residuals and ensuring the network adheres to the underlying physics. Additionally, the hyper-parameters $\{\lambda_r, \lambda_{tb}, \lambda_{sb}\}$ can be manually set or adapted during training to balance the contributions of the different loss terms, thus optimizing the model's performance.

2.2. Inverse PINNs

PINNs can naturally be extended to address inverse problems, which involve inferring unknown quantities of a physical system from observed data. The unknown quantities are generally represented as parameters, functions, or other hidden physical properties included in the PDE. Inverse problems aim to determine the unknown quantities that are aligned with both the observed data and the governing equations defined in (1), (2), and (3). Such problems arise in various fields, including material science, biomedical imaging, and fluid dynamics.

Consider a PDE with unknown quantity f defined by:

$$\begin{aligned}
u_t + \mathcal{N}[u, f] &= 0, && x \in D, \quad t \in [0, T], \\
I[u] &= g(x), && x \in D, \\
B[u] &= h(x, t), && x \in \partial D, \quad t \in [0, T]. \\
u &= u_d, && (x, t) \in D_d.
\end{aligned}$$

In this formulation, the additional constraint

$$u = u_d, \quad (x, t) \in D_d,$$

specifies that, in the data subdomain D_d, the solution u must match the observed data u_d. This condition provides an extra supervision signal to guide the inversion process and ensure that the learned solution is consistent with the available measurements in D_d. The inverse PINNs approximate both f and u using neural networks f_η and u_θ, respectively, thereby ensuring that the governing PDEs are satisfied while aligning the network's output with the observed data in D_d.

To achieve this, the loss function for inverse PINNs builds upon the standard PINNs' loss function by incorporating an additional data loss term [30]. The total loss function is expressed as:

$$\mathcal{L}(\theta, \eta) = \lambda_r \mathcal{L}_r(\theta, \eta) + \lambda_{tb} \mathcal{L}_{tb}(\theta) + \lambda_{sb} \mathcal{L}_{sb}(\theta) + \lambda_d \mathcal{L}_d(\theta)$$

where initial condition loss $\mathcal{L}_{tb}(\theta, \eta)$ and boundary condition loss $\mathcal{L}_{tb}(\theta, \eta)$ are defined as in the forward PINNs' formulation. The additional residual loss term and data loss term are given by:

$$\mathcal{L}_r(\theta, \eta) = \frac{1}{N_r} \sum_{i=1}^{N_r} \left| \frac{\partial u_\theta}{\partial t}(x_r^i,\ t_r^i) + \mathcal{N}[u_\theta, f_\eta](x_r^i,\ t_r^i) \right|^2,$$
$$\mathcal{L}_d(\theta) = \frac{1}{N_d} \sum_{i=1}^{N_d} \left| u_\theta(x_d^i, t_d^i) - u_{obs}^i \right|^2,$$

where u_{obs}^i denotes the observed data at the sampled points $\{x_d^i, t_d^i\}_{i=1}^{N_d}$. The hyper-parameter λ_d controls the influence of this data loss relative to the other loss components. By optimizing both θ and η, inverse PINNs seek a solution that respects the governing PDEs while matching the observed data.

However, conventional inverse PINNs face a significant limitation: they do not explicitly account for the causal nature of dynamic systems. Temporal causality ensures that the predicted solution evolves naturally from the initial conditions and progresses consistently. Without such considerations, the model may converge to unstable or non-physical solutions, especially in time-dependent problems. Recent studies have proposed frameworks that integrate causal training mechanisms [23], which sequentially prioritize residuals to enforce causality. These advancements lay the foundation for extending inverse PINNs into causality-aware domains—a concept we explore further in this work.

2.3. Causal Training for Physics-Informed Neural Networks

Conventional PINN training can yield solutions that neglect temporal causality, which is crucial for dynamic systems. A causal training framework was introduced in [23] to address this issue, aligning the training process with the natural sequential structure of physical systems.

First, the training objective is reformulated to prioritize learning time sequentially. Specifically, the residual loss function is modified by incorporating a weighting mechanism that emphasizes earlier time points. The basic loss function (4) is rewritten by separating the contributions of time and space as follows:

$$\begin{aligned} \mathcal{L}_r(\theta) &= \frac{1}{N_t} \sum_{i=1}^{N_t} \mathcal{L}_r(t_i, \theta) \\ &= \frac{1}{N_t N_x} \sum_{i=1}^{N_t} \sum_{j=1}^{N_x} \left| \frac{\partial u_\theta}{\partial t}(x_r^j, t_r^i) + \mathcal{N}[u_\theta](x_r^j, t_r^i) \right|^2. \end{aligned} \quad (5)$$

Here, N_t and N_x denote the number of temporal and spatial discretization points, respectively—that is, N_t is the total number of time points sampled with $[0, T]$ and N_x is the number of spatial grid points sampled within the domain D. The loss function for the spatial component at a given time t_r^i becomes:

$$\mathcal{L}_r(t_r^i, \theta) = \frac{1}{N_x} \sum_{j=1}^{N_x} \left| \frac{\partial u_\theta}{\partial t}(x_r^j, t_r^i) + \mathcal{N}[u_\theta](x_r^j, t_r^i) \right|^2.$$

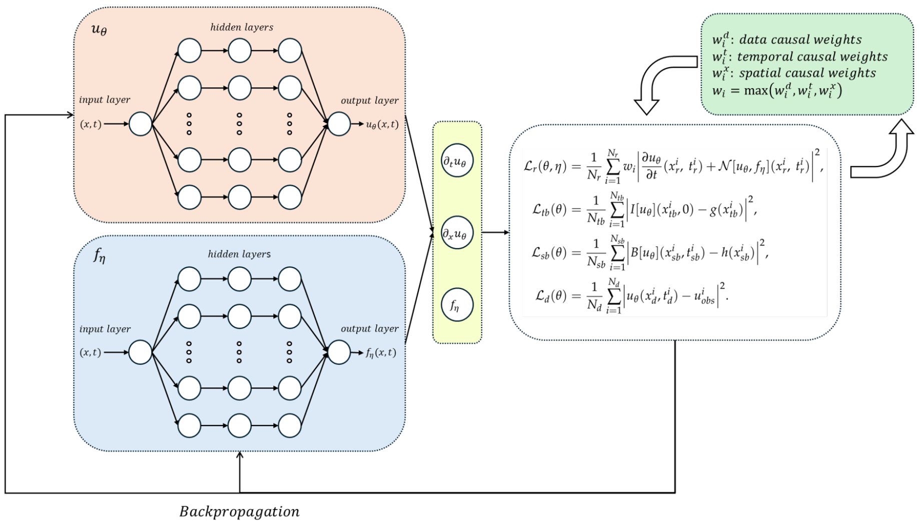
Next, to enforce the inherent causality of the system, a time-dependent weighting function w_i is introduced. This function assigns greater importance to earlier time steps so that the model focuses on minimizing residuals at these points before addressing later times:

$$w_i = \exp\left(-\epsilon \sum_{k=1}^{i-1} \mathcal{L}_r(t_k, \theta) \right), \quad \text{for } i = 2, 3, \dots, N. \tag{6}$$

Here, ϵ is the causality parameter controlling the weight decay. In [23], ϵ was experimentally chosen from the range [100, 10, 1, 0.1, 0.01, 0.001]. When applying w_i to the loss function (5), the overall training objective becomes:

$$\mathcal{L}_r(\theta) = \frac{1}{N_t} \sum_{i=1}^{N_t} \exp\left(-\epsilon \sum_{k=1}^{i-1} \mathcal{L}_r(t_k, \theta) \right) \mathcal{L}_r(t_i, \theta).$$

Although this weighting mechanism is designed primarily for temporal causality, it can be extended to spatial dimensions. During training, introducing an analogous spatial weighting function can prioritize points closer to given conditions—such as boundary values or observed data. This spatial extension complements the temporal weighting mechanism and lays the groundwork for addressing both temporal and spatial causality, as further explored in the context of inverse problems. Figure 1 illustrates the proposed CI-PINN algorithm.



u_θ
hidden layers
input layer
(x,t)
output layer
$u_\theta(x,t)$
f_η
hidden layers
input layer
(x,t)
output layer
$f_\eta(x,t)$
$\partial_t u_\theta$
$\partial_x u_\theta$
f_η
w_i^d: data causal weights
w_i^t: temporal causal weights
w_i^x: spatial causal weights
$w_i = \max(w_i^d, w_i^t, w_i^x)$
$\mathcal{L}_r(\theta, \eta) = \frac{1}{N_r} \sum_{i=1}^{N_r} w_i \left| \frac{\partial u_\theta}{\partial t}(x_r^i, t_r^i) + \mathcal{N}[u_\theta, f_\eta](x_r^i, t_r^i) \right|^2,$
$\mathcal{L}_{tb}(\theta) = \frac{1}{N_{tb}} \sum_{i=1}^{N_{tb}} \left| I[u_\theta](x_{tb}^i, 0) - g(x_{tb}^i) \right|^2,$
$\mathcal{L}_{sb}(\theta) = \frac{1}{N_{sb}} \sum_{i=1}^{N_{sb}} \left| B[u_\theta](x_{sb}^i, t_{sb}^i) - h(x_{sb}^i) \right|^2,$
$\mathcal{L}_d(\theta) = \frac{1}{N_d} \sum_{i=1}^{N_d} \left| u_\theta(x_d^i, t_d^i) - u_{obs}^i \right|^2.$
Backpropagation

Figure 1. Schematic illustration of CI-PINNs. The definitions of causal weights vary depending on the problem and are detailed in the following sections.

Hyperparameter Selection: In all subsequent CI-PINN experiments, we utilize the causality parameter ϵ as introduced above to control the decay behavior of causal weights. To determine an optimal value for ϵ, we experimentally evaluated candidate values from the set $\{10, 1, 0.1, 0.01, 0.001\}$. Candidate values that resulted in insufficient propagation (i.e., the weights did not spread effectively from the observation and boundary regions) or led to an overly rapid convergence (i.e., the weights quickly converged to 1 across the domain) were excluded. Among the remaining candidates, we selected the value that produced the smallest L^2 relative error on the validation set as the optimal ϵ. This systematic selection process ensures that the causal weights are balanced and effectively guide the training process for stable and accurate reconstructions. The following sections illustrate how these ideas are applied to specific inverse problems.

Causal weights: In Section 2.3, we introduced the concept of causal weights using notation similar to w_i in Equation (6), focusing primarily on the temporal dimension. In this work, we adopt a more general representation for the causal weights, $w_d(x_i), w_t(t_i), w_{xy}(x_i, y_i)$. Henceforth, we use w to denote these causal weights in all scenarios. Each experiment section will specify how w is constructed (e.g., temporal, spatial, or data causal weights) to emphasize critical regions such as boundaries, initial conditions, or observation points. Although the formulation of these weights is based on a relatively simple functional form, their effective integration into our inverse PINN framework poses several challenges. Unlike conventional applications, where a simple decay function is applied to a single scalar term, our method requires a coordinated balance among multiple causal weights across all dimensions. Determining an optimal value for ϵ is particularly demanding because the chosen parameter must allow supervision to propagate smoothly from regions with strong data support (e.g., boundaries and initial conditions) to areas with sparse information, all without compromising convergence or stability. In essence, while the mathematical basis for the weights is well established, combining them in a multidimensional setting to enforce the complex causal structure of physical systems necessitates careful analysis and calibration, highlighting the sophisticated nature of our approach.

3. Inverse Problem for the Wave Equation

3.1. The Underlying Inverse Problem

The wave equation is a fundamental model for physical phenomena such as acoustics, electromagnetics, and elasticity. As a representative linear hyperbolic PDE [31], it provides a valuable testbed for assessing the proposed CI-PINN framework. Let $D \subset \mathbb{R}^d$ be an open, bounded, and simply connected set with smooth boundary $\partial\Omega$. We consider the wave equation with zero Dirichlet boundary conditions:

$$\begin{aligned} u_{tt} - \Delta u &= f, \quad \forall (x,t) \in D \times (0,T), \\ u &\equiv 0, \quad \forall (x,t) \in \partial D \times (0,T), \end{aligned}$$

where Δ denotes the spatial Laplace operator and $f \in L^2(D_T)$ denotes a source function with $D_T = D \times (0,T)$. In the forward problem, if the initial conditions:

$$\begin{aligned} u(x,0) &= u_0(x), \quad \forall x \in D, \\ u_t(x,0) &= u_1(x), \quad \forall x \in D, \end{aligned}$$

are prescribed with $u_0 \in L^2(D)$ and $u_1 \in H^{-1}(D)$, classical theory guarantees that a unique solution $u(x,t)$ exists. In our study, the specific solution we aim to approximate is given by

$$u(x,t) = \sin(\pi x)\cos(2\pi t).$$

In many practical applications, however, the initial data u_0 and u_1 are unavailable. Instead, we obtain measurements of the solution over an observation subdomain $D' \subset \bar{D}$. That is, we are provided with

$$u(x,t) = g, \quad (x,t) \in D' \times (0,T).$$

The inverse problem is a data assimilation problem of recovering the unknown initial conditions and, consequently, the entire solution $u(x,t)$ of the wave equation from the available partial data. In other words, the solution we seek is a function $u(x,t)$ that satisfies the wave equation and the zero boundary conditions throughout $D \times (0,\ T)$ while simultaneously matching the measurement data $g(x,t)$ on $D' \times (0,\ T)$ and implicitly determining the initial state. Although the lack of initial data renders the inverse problem ill posed in the classical sense, stability can be recovered under additional conditions. In particular, if the observation subdomain D' satisfies the Geometric Control Conditions (GCCs) [32,33], which, in essence, ensures that every generalized bicharacteristic (or "light ray") of the wave operator, including reflected paths, intersects D', then conditional stability estimates can be established. As shown in Theorem 5.2 (cf. [31]), under the GCCs, the energy of the solution (measured in appropriate norms) can be bounded in terms of the measurement data g, the source term f, and the residual of the wave operator. These conditional stability results provide a theoretical foundation for the existence and uniqueness of the solution to the inverse problem in a conditional sense. By incorporating the PDE residual, boundary conditions, and measurement data into the loss function, the CI-PINN is trained to approximate a solution $u^*(x,t)$ that converges to the true solution $u(x,t)$ when training error is sufficiently small.

3.2. Residuals and Network Smoothness

For the neural network $u_\theta(x,t)$ with parameters $\theta \in \Theta$, we assume sufficient smoothness (C^k with $k \geq 2$) over D_T. The PDE residual is defined as

$$\mathcal{R}_\theta(x,t) = \partial_{tt} u_\theta - \Delta u_\theta - f(x,t), \quad \forall (x,t) \in D_T.$$

To enforce the boundary conditions in $\partial D \times (0,T)$, we define the boundary residual as

$$\mathcal{R}_{sb,\theta} = u_\theta, \quad \forall (x,t) \in \partial D \times (0,T).$$

Furthermore, the data residual on the observation domain D'_T is

$$\mathcal{R}_{d,\theta} = u_\theta - g, \quad \forall (x,t) \in D'_T.$$

3.3. Causal Weights

Since this inverse problem does not involve initial conditions, we apply spatial and data causal weights to prioritize learning in critical spatial regions. In our framework, we define two types of causal weights.

Data weights: These weights focus on regions where observation data g are provided. For each spatial points x_i, we set

$$w_d(x_i) = \begin{cases} 1, & x_i \in D'_T \\ \exp\left(-\epsilon \sum_{k=1}^{i-1} \mathcal{L}_r(x_k, \theta, f) \right), & x_i \in D_T \setminus D'_T. \end{cases}$$

Spatial weights: These weights emphasize the contribution of the boundary conditions. For each spatial point x_i, we define

$$w_x(x_i) = \begin{cases} 1, & x_i \in \partial D \\ \exp\left(-\epsilon \sum_{k=1}^{N_x - i} \mathcal{L}_r(x_{N_x - k + 1}, \theta, f) \right), & x_i \in D \setminus \partial D. \end{cases}$$

The final causal weight at each point is then given by

$$w_{xd}(x_i) = \max(w_d(x_i), w_x(x_i)).$$

As depicted in the right panel of Figure 2, the data weights propagate upward from the observation region D'_T into the rest of the domain, while the spatial weights move inward from the boundary ∂D. This combined weighting mechanism first emphasizes critical regions (observation data and boundary conditions). Then, their influence gradually extends to the entire domain, promoting a more stable and accurate solution.

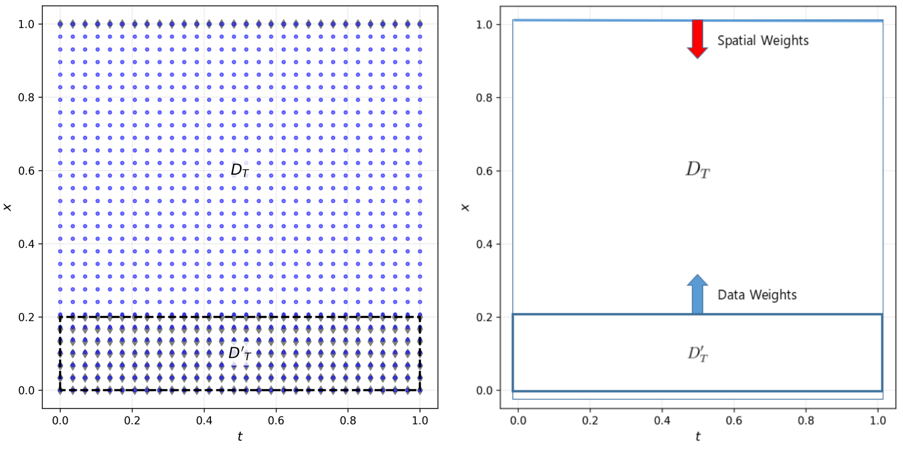


Figure 2. (**Left**) Visualization of the domain D_T and observation domain D'_T for the wave equation. Blue dots represent interior collocation points, while gray dots indicate boundary and observation data points where supervision is provided. (**Right**) An illustration depicting the propagation of data weights upward from the observation subdomain D'_T into the domain D_T, as well as the inward propagation of spatial weights from the domain boundaries. Arrows indicate the direction of influence for each weight type.

3.4. Loss Function

The overall loss function for the inverse problem is constructed by combining the weighted residual loss, the boundary loss, and the data loss:

$$\mathcal{L}(\theta) = \sum_{\substack{i=1 \\ (x_i,t_i)\in D_T}}^{N_{int}} w_{xd}(x_i)|\mathcal{R}_\theta(x_i,t_i)|^2 + \sum_{\substack{i=1 \\ (x_i,t_i)\in \partial D_T}}^{N_{sb}} |\mathcal{R}_{sb,\theta}(x_i,t_i)|^2 + \sum_{\substack{i=1 \\ (x_i,t_i)\in D'_T}}^{N_d} |\mathcal{R}_{d,\theta}(x_i,t_i)|^2,$$

where the causal weights w are applied as described above.

3.5. Numerical Experiments

In our experiments, the source term f and the observation data g are generated directly from the exact solution. Specifically, with the exact solution defined as

$$u(x,t) = \sin(\pi x)\cos(2\pi t)$$

we substitute into the wave equation

$$\partial_{tt}u - \Delta u = f,$$

which yields $f = 0$. Similarly, the observation data g are obtained by evaluating $u(x,t)$ on the observation domain $D' = (0, 0.2)$.

For the numerical implementation, we use a fully connected neural network with four hidden layers, each containing 24 neurons and employing the *tanh* activation function. The network takes a 2D input (x,t) and outputs the scalar prediction $u(x,t)$. We use the Adam optimizer with a learning rate of 1×10^{-4} for 50,000 epochs.

Figure 2 illustrates the domain D_T, the observation domain D'_T, and the interior collocation points, boundary points, and observation points.

The first row of Figure 3 presents the exact solution alongside the CI-PINNs' predicted solution. The second row illustrates the pointwise absolute error and the L^2 highlighting that CI-PINNs achieve significantly lower errors compared to conventional PINNs and Causal PINNs (Section 2.3 [23]). Specifically, the L^2 relative error for conventional PINNs is 0.004321; for Causal PINNs, it is 0.003865. In contrast, CI-PINNs reduce this error to 0.00206, a reduction of approximately 52% compared to conventional PINNs and 47% compared to Causal PINNs. This substantial improvement underscores the effectiveness of incorporating spatial causal weights in enhancing solution accuracy. Additionally, as summarized in Table 1, the training speed for CI-PINNs remains comparable to that of conventional PINNs, indicating that the accuracy gains do not come at a significant computational cost.

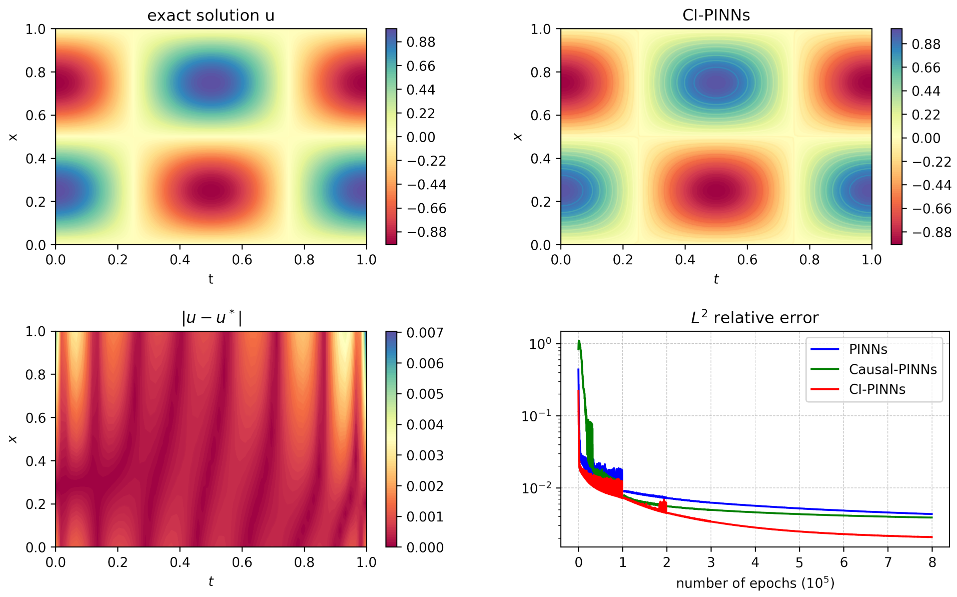


Figure 3. (**Upper left**) Exact solution $u(x,t)$; (**upper right**) CI-PINNs' predicted solution; (**lower left**) pointwise absolute error; (**lower right**) L^2 relative error during training. Three methods are compared: PINNs (blue), Causal PINNs (green), and CI-PINNs (red).

Table 1. Summary of the maximum number of training iterations and the corresponding iteration time (in seconds) for the wave, parabolic, and elliptic inverse problems. The "Max iteration" row indicates the total training iterations for each problem, while the remaining rows show the average iteration time for the vanilla PINNs and CI-PINNs, respectively.

		Wave	**Parabolic**	**Elliptic**
Max iteration		8×10^5	5×10^5	8×10^5
Iteration time (s)	PINNs	0.006	0.026	0.009
	CI-PINNs	0.006	0.026	0.019

The experimental results demonstrate that CI-PINNs significantly outperform conventional PINNs in reconstructing the wave equation's solution $u(x,t)$. The effective incorporation of spatial causal weights derived from observation and boundary considerations guides the training process to prioritize regions of high importance, thereby enhancing stability and accuracy.

Figure 4 displays the evolution of the spatial causal weights during training. As illustrated in the right panel of Figure 2, the data weights propagate upward from the observation region, while the boundary weights move inward from the domain edges. At early epochs, the weights are concentrated in the supervised regions (boundaries and observation areas), ensuring that these regions receive priority. As training progresses, the influence of the weights gradually spreads across the domain, contributing to enhanced stability and overall accuracy.

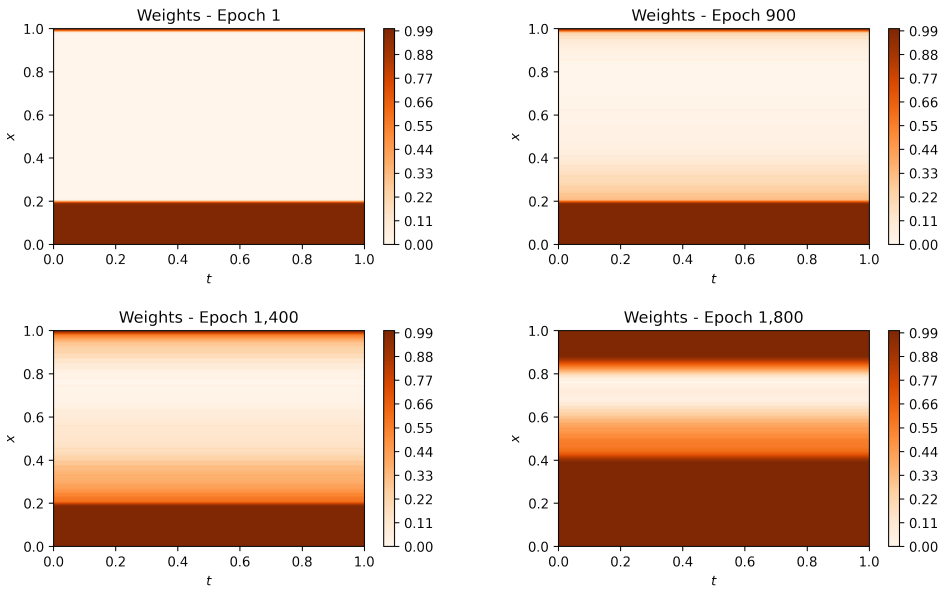


Figure 4. Evolution of causal weights over different epochs. In training, weights concentrate on critical regions; the influence spreads across the domain as training proceeds.

4. Inverse Source Problem for a Parabolic Equation

4.1. The Underlying Inverse Problem

To demonstrate the versatility of our CI-PINN framework, we consider an inverse source problem governed by a parabolic equation as described in [34]. Parabolic equations are widely used to model diffusion processes such as heat conduction and pollutant dispersion. The governing equation for the parabolic system is given by:

$$\begin{aligned} \partial_t u + \Delta u &= F(x,y,t), && (x,y,t) \in D_T := D \times (0,T], \\ u(x,y,t) &= b(x,y,t), && (x,y,t) \in \partial D_T := \partial D \times (0,T], \\ u(x,y,t) &= u_0(x,y), && (x,y) \in D, \end{aligned}$$

where $D \subset \mathbb{R}^d$ is a bounded domain with a smooth boundary, $b(x,y,t)$ denotes the boundary conditions, and $u_0(x,y)$ represents the initial condition. For the forward problem, if the source function $F(x,y,t)$, and the initial and boundary data are all known, it is well posed in appropriate function spaces. In many practical applications, however, the internal source function $F(x,y,t) = f(x,y)g(t)$ is difficult to measure directly. Instead, one assumes $g(t)$ is given in advance and recovers $f(x,y)$ by using the final time measurement

$$u(x,y,T) = \phi(x,y), \quad (x,y) \in D.$$

Thus, the inverse problem is to recover the unknown spatial source $f(x,y)$ (and, consequently, the entire solution $u(x,y,t)$) from the final time data $\phi(x,y)$. It is well known that the inverse source problem of recovering $f(x,y)$ from the final time data is inherently ill posed, meaning that small perturbations in $\phi(x,y)$ could lead to significant errors in $f(x,y)$. Nevertheless, when $\phi(x,y)$ is noise free and sufficiently regular (for instance, $\phi(x,y) \in H^2(D)$), uniqueness and conditional stability results (see, e.g., [35–37]) ensure that a unique solution exists in a regularized sense. We consider a two-dimensional model with $D = [0,1] \times [0,1]$ and $T = \frac{4}{3}$. The exact solution of the parabolic problem is chosen as

$$u(x,y,t) = \sin(\pi t)\sin(\pi x)\sin(\pi y),$$

for $(x,y,t) \in D_T := [0,1] \times [0,1] \times [0, \frac{4}{3}]$. This function satisfies the zero initial condition $u(x,y,0) = 0$ and homogeneous boundary conditions. From direct computation, the exact source term is given by

$$f(x,y) = \sin(\pi x)\sin(\pi y),$$

and the known temporal source term is

$$g(t) = \pi\cos(\pi t) + 2\pi^2\sin(\pi t).$$

Thus, the exact final time measurement is

$$u(x,y,\frac{4}{3}) = \phi(x,y) = \sin(\frac{4\pi}{3})\sin(\pi x)\sin(\pi y).$$

In our reconstruction scheme, both $u(x,y,t)$ and $f(x,y)$ are approximated by deep neural networks $u_\theta(x,y,t)$ and $f_\eta(x,y)$, respectively. We incorporate derivative terms of the residuals into the loss function to mitigate the inherent ill-posed nature of the inverse source problem, thereby enforcing higher regularity in the solution. This approach stabilizes the inversion and yields improved reconstruction accuracy compared to standard least-squares loss functions. Our numerical experiments demonstrate that this framework yields enhanced accuracy in recovering the unknown source $f(x,y)$ and the solution $u(x,y,t)$.

4.2. Residuals and Network Smoothness

By using the *tanh* activation function, we can assume that the neural network approximations u_θ and f_η belong to $C^\infty(\bar{D} \times [0,T])$. Therefore, the network $u_\theta(x,y,t)$ and its derivatives up to second order are uniformly bounded on $\bar{D} \times [0,T]$.

The PDE residual is defined as

$$\mathcal{R}_{\theta,\eta}(x,y,t) = \partial_t u_\theta(x,y,t) + \Delta u_\theta(x,y,t) - f_\eta(x,y)g(t), \quad (x,y,t) \in D_T.$$

To enforce the boundary and initial conditions, we define:

$$\begin{aligned} \mathcal{R}_{sb,\theta}(x,y,t) &= u_\theta(x,y,t), \quad (x,y,t) \in \partial D_T, \\ \mathcal{R}_{tb,\theta}(x,y) &= u_\theta(x,y), \quad x \in D. \end{aligned}$$

and the data residual as

$$\mathcal{R}_{d,\theta}(x,y) = u_\theta(x,y,T) - \phi(x,y), \quad x \in D$$

4.3. Causal Weights

To enhance stability and accuracy, we incorporate causal weights, prioritizing learning in critical regions across the spatial dimensions x and y and the temporal dimension t. Given that boundary conditions, initial conditions, and observation data are specified, we define separate causal weights for each dimension.

Spatial Weights for x and y: For each spatial coordinate, we define weights that emphasize the domain boundaries. For the x-dimension:

$$w_x(x_i) = \begin{cases} 1, & x_i \in \partial D, \\ \max\left[\exp\left(-\epsilon \sum_{k=1}^{i-1} \mathcal{L}_r(x_k, \theta, \eta)\right), \right. & \\ \qquad \left. \exp\left(-\epsilon \sum_{k=1}^{N_x - i} \mathcal{L}_r(x_{N_x-k+1}, \theta, \eta)\right)\right], & x_i \in D \setminus \partial D. \end{cases}$$

Similarly, for the y-dimension:

$$w_y(y_i) = \begin{cases} 1, & y_i \in \partial D, \\ \max\left[\exp\left(-\epsilon \sum_{k=1}^{i-1} \mathcal{L}_r(y_k, \theta, \eta)\right), \right. & \\ \qquad \left. \exp\left(-\epsilon \sum_{k=1}^{N_y - i} \mathcal{L}_r(y_{N_y-k+1}, \theta, \eta)\right)\right]. & y_i \in D \setminus \partial D. \end{cases}$$

Temporal weights: For the time dimension, we define the causal weights to emphasize the initial time:

$$w_t(t_i) = \begin{cases} 1, & t_i = 0, \\ \exp\left(-\epsilon \sum_{k=1}^{i-1} \mathcal{L}_r(t_k, \theta, \eta)\right), & t_i \in (0, T]. \end{cases}$$

Data weights: Since data are provided at $t = T$ in the time dimension, we can define the weights with a focus on this point as follows:

$$w_d(t_i) = \begin{cases} 1, & t_i = T, \\ \exp\left(-\epsilon \sum_{k=1}^{N_d - i} \mathcal{L}_r(t_{N_t-k+1}, \theta, \eta)\right), & t_i \in [0, T). \end{cases}$$

The overall causal weight for grid points (x_i, y_i, t_i) is then given by

$$w_{xytd}(x_i, y_i, t_i) = \max\{w_x(x_i), w_y(y_i), w_t(t_i), w_d(t_i)\}$$

These weights are incorporated into the loss function to prioritize learning near boundaries, initial conditions, and observation data.

4.4. Loss Function

The standard loss function combines the residuals for the PDE, boundary, initial, and data constraints:

$$\mathcal{L}^s(\theta, \eta) = \sum_{\substack{i=1 \\ (x_i, y_i, t_i) \in D_T}}^{N_{int}} w_{xytd}(x_i, y_i, t_i) |\mathcal{R}_{\theta,\eta}(x_i, y_i, t_i)|^2 + \sum_{\substack{i=1 \\ (x_i, y_i) \in D_{t=T}}}^{N_d} |\mathcal{R}_{d,\theta}(x_i, y_i)|^2 + \\ \sum_{\substack{i=1 \\ (x_i, y_i) \in D}}^{N_{tb}} |\mathcal{R}_{tb,\theta}(x_i, y_i)|^2 + \sum_{\substack{i=1 \\ (x_i, y_i, t_i) \in \partial D_T}}^{N_{sb}} |\mathcal{R}_{sb,\theta}(x_i, y_i, t_i)|^2.$$

Following the modified loss proposed in [34], we further incorporate terms that involve time derivatives of the residual and additional quadrature weights, leading to

$$
\begin{aligned}
\mathcal{L}(\theta,\eta) = & \sum_{\substack{i=1 \\ (x_i,y_i)\in D_{t=T}}}^{N_d} |\mathcal{R}_{d,\theta}(x_i,y_i)|^2 + \\
& \sum_{\substack{i=1 \\ (x_i,y_i,t_i)\in D_T}}^{N_{int}} w^0_{xytd}(x_i,y_i,t_i)|\mathcal{R}_{\theta,\eta}(x_i,y_i,t_i)|^2 + \\
& \sum_{\substack{i=1 \\ (x_i,y_i,t_i)\in D_T}}^{N_{int}} w^1_{xytd}(x_i,y_i,t_i)|\partial_t\mathcal{R}_{\theta,\eta}(x_i,y_i,t_i)|^2 + \\
& \sum_{\substack{i=1 \\ (x_i,y_i)\in D}}^{N_{tb}} |\mathcal{R}_{tb,\theta}(x_i,y_i)|^2 + \sum_{\substack{i=1 \\ (x_i,y_i)\in D}}^{N_{tb}} |\Delta\mathcal{R}_{tb,\theta}(x_i,y_i)|^2 + \\
& \sum_{\substack{i=1 \\ (x_i,y_i,t_i)\in \partial D_T}}^{N_{sb}} |\mathcal{R}_{sb,\theta}(x_i,y_i,t_i)|^2 + \sum_{\substack{i=1 \\ (x_i,y_i,t_i)\in \partial D_T}}^{N_{sb}} |\partial_t\mathcal{R}_{sb,\theta}(x_i,y_i,t_i)|^2 + \\
& \sum_{\substack{i=1 \\ (x_i,y_i,t_i)\in \partial D_T}}^{N_{sb}} |\partial_t^2\mathcal{R}_{sb,\theta}(x_i,y_i,t_i)|^2,
\end{aligned}
$$

where w_i^0, w_i^1 are causal weights for PDE and PDE differentiated concerning t, respectively. These additional terms help enforce the PDE and its temporal derivatives, improving overall reconstruction accuracy.

4.5. Numerical Experiments

We conduct experiments by adopting the setup in [34]. Two separate neural networks approximate $u_\theta(x,y,t)$ and $f_\eta(x,y)$. Both networks are fully connected with two hidden layers containing 20 neurons each and use the *tanh* activation function.

The total number of collocation points is $N_{int} = 50 \times 50 \times 50$, and the points are sampled uniformly from D_T. Boundary points are chosen along $\partial D \times [0,T]$, initial points are taken at $t = 0$, and data points correspond to the observation domain at $t = \frac{4}{3}$. This diversified sampling ensures that the model learns from all critical regions.

The initial learning rate is 1×10^{-3}, decaying by a factor of 10 every 10^5 iterations to facilitate convergence. In our experiments, the optimal value of ϵ was determined to be 0.1. The loss function incorporates the causal weights, emphasizing critical regions near the observation data at $t = \frac{4}{3}$ and along the boundaries.

Figure 5 shows the reconstruction accuracy. The upper panels compare the exact source term $f(x,y)$ with its reconstruction obtained using CI-PINNs, showing excellent agreement. The lower left panel illustrates the absolute error $|f - f^*|$ between the exact and predicted solutions, with only minor discrepancies in regions where observation data are sparse. Notably, the lower right panel shows the L^2 relative error convergence: conventional PINNs converge to a final error of 0.0855, Causal PINNs reduce this to 0.0673, and CI-PINNs further lower it to 0.0397, an improvement of approximately 53.6% over conventional PINNs. This substantial improvement confirms that integrating 3D causal weights enhances reconstruction accuracy significantly. Moreover, despite the cubic nature of the 3D causal weights, the computational overhead remains minimal. The iteration speed for CI-PINNs is comparable to that of conventional PINNs, indicating that the improved stability and accuracy do not come at a steep computational cost (Table 1).

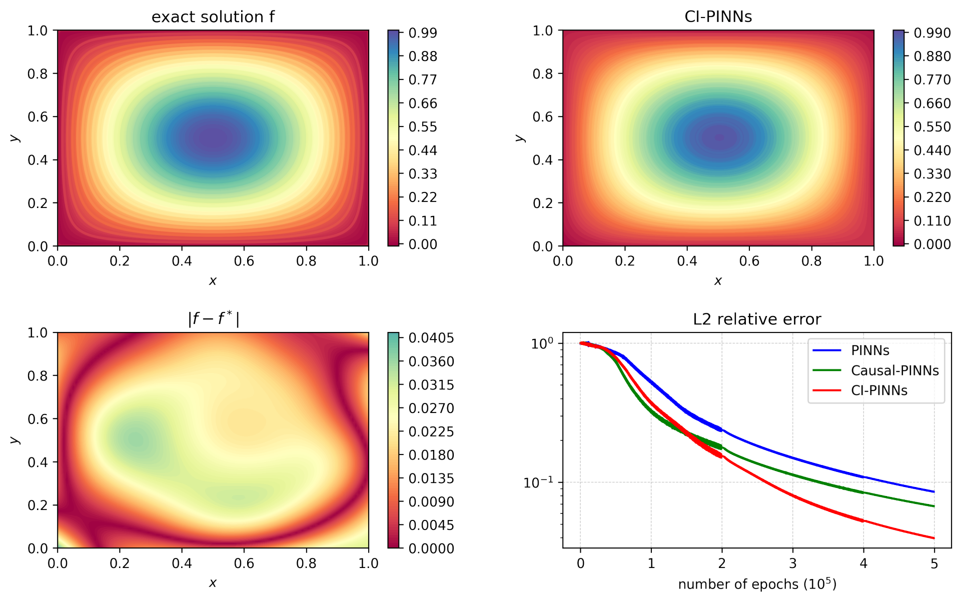


Figure 5. (**Upper left**) Exact source term $f(x, y)$; (**upper right**) reconstructed source term using CI-PINNs; (**lower left**) pointwise absolute error $|f - f^*|$ between the exact solution f^* and the CI-PINNs' prediction; (**lower right**) L^2 relative error convergence during training for three methods: PINNs (blue), Causal PINNs (green), and CI-PINNs (red).

Figure 6 visualizes the evolution of the 3D causal weights. Each column represents a different time step (e.g., $t = \frac{3}{15}, \frac{7}{15}, \frac{11}{15}, 1$), and each row corresponds to a training epoch (e.g., 400, 800, 1,200, 1,600). Initially, the weights are concentrated on the boundaries, initial conditions, and observation regions; as training progresses, they gradually propagate into the interior, enabling the network to capture the solution's local and global features.

Overall, the experimental results validate that CI-PINNs significantly outperform conventional PINNs for the inverse source problem of a parabolic equation. The effective integration of 3D causal weights, which balance the contributions from the spatial and temporal domains, results in faster convergence and improved accuracy in determining both the solution $u(x, y, t)$ and the unknown spatial source term $f(x, y)$

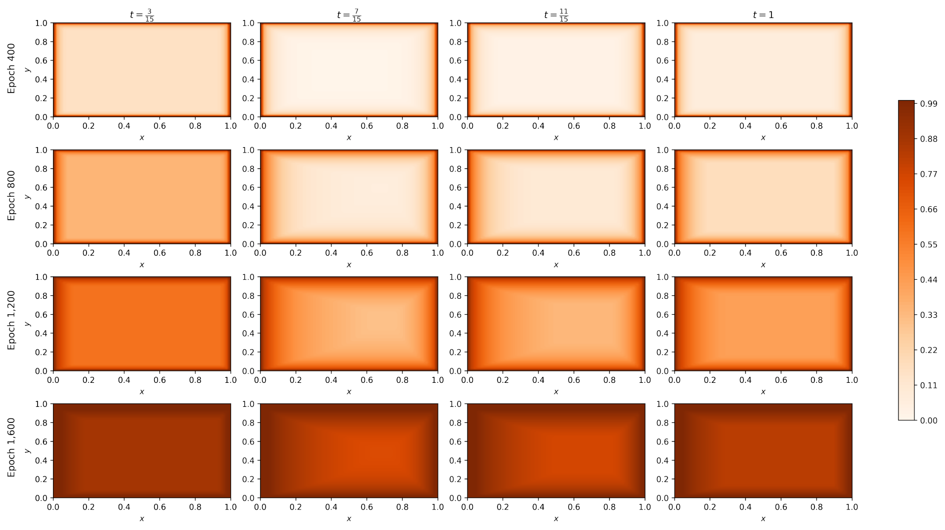


Figure 6. Evolution of the 3D causal weights during training for the parabolic inverse problem. Each column represents a different time step (e.g., $t = \frac{3}{15}, \frac{7}{15}, \frac{11}{15}, 1$), and each row corresponds to a training epoch (e.g., 400, 800, 1200, 1600). Initially, the weights concentrate on critical regions (boundaries, initial conditions, and observation areas), and as training progresses, they gradually propagate into the interior, facilitating improved accuracy and stability.

5. Inverse Source Problem for Elliptic Equation

5.1. The Underlying Inverse Problem

To illustrate the adaptability of our CI-PINN framework, we consider an inverse source problem governed by an elliptic equation as detailed in [38]. Elliptic equations are fundamental for describing steady-state phenomena such as electrostatics, thermal equilibrium, and fluid flow. Let $D \subset \mathbb{R}^2$ be a bounded, connected domain with a smooth boundary ∂D. We consider the elliptic problem

$$\mathcal{P}u := -\nabla \cdot (\sigma(x,y)\nabla u(x,y)) + c(x,y)u(x,y) = f(x,y), \quad (x,y) \in D,$$

with the Neumann boundary condition

$$\mathcal{B}u := \sigma(x,y)\frac{\partial u(x,y)}{\partial \nu} = g(x,y), \quad (x,y) \in \partial D.$$

Here, $\sigma(x,y)$ and $c(x,y)$ are given smooth functions (with $\sigma(x,y) \geq \sigma_0 > 0$ and $c(x,y)$ bounded) and ν is the outward normal vector on ∂D, and for a specified source $f(x,y)$ and boundary data $g(x,y)$, the forward problem is well posed and admits a unique solution $u \in H^1(D)$. The source term $f(x,y)$ is not directly measurable in many practical applications. Instead, one often acquires internal measurement data on a subdomain $D_s \subset D$. In our noise-free setting, we assume that

$$u(x,y) = u_s(x,y), \quad (x,y) \in D_s,$$

where $u_s(x,y)$ represents the exact measurement of the solution within D_s. Due to the unique continuation property of elliptic equations, the internal data u_s uniquely determine the solution u throughout D. Consequently, the source term can be formally computed via

$$f(x,y) = -\nabla \cdot (\sigma(x,y)\nabla u(x,y)) + c(x,y)u(x,y).$$

However, because the continuation and the differentiation are unstable, the inverse problem is inherently unstable, even in the noise-free case. For our numerical experiment, we consider a two-dimensional model with $D = [0,1]^2$ and specify the exact solution as

$$u(x,y) = \sin(\pi x) + y^3, \quad (x,y) \in D.$$

We choose the coefficients as

$$c(x,y) = \sigma(x,y) = \exp(-30[(x-0.5)^2 + (y-0.5)^2]).$$

A straightforward calculation then yields the exact source term:

$$\begin{aligned} f(x,y) = \exp(-30[(x-0.5)^2 + (y-0.5)^2] \times [(\pi^2+1)\sin(\pi x) + \\ 60\pi(x-0.5)\cos(\pi x) + 180(y-0.5)y^2 + y^3 - 6y]) \end{aligned}$$

The internal measurement $u_s(x,y)$ is taken as the restriction of $u(x,y)$ to the subdomain D_s; for instance, we may choose

$$D_s = [\frac{1}{4}, \frac{3}{4}] \times [\frac{1}{2}, \frac{3}{4}].$$

Figure 7 illustrates the domain D and the observation subdomain D_s along with the distribution of training points. Training points are sampled uniformly over D for the interior residual, along ∂D for boundary residuals, and within D_s for observation data. Spatial causal weights are constructed to propagate the influence of the observation subdomain D_s and the boundary conditions throughout the domain.

In our reconstruction approach, both $u(x,y)$ and $f(x,y)$ are approximated by deep neural networks $u_\theta(x,y)$ and $f_\eta(x,y)$, respectively. Since recovering f from u involves differentiation, which is unstable, the inverse problem is ill posed. We augment our loss function with additional regularization terms (detailed in Section 5.4) that penalize large gradients in the network outputs to stabilize the inversion. This regularization enforces a Lipschitz condition on the approximations, thereby mitigating the instability and improving reconstruction accuracy. Our numerical experiments demonstrate that, with this regularized framework, the unknown source $f(x,y)$ and the solution $u(x,y)$ can be recovered accurately over the entire domain.

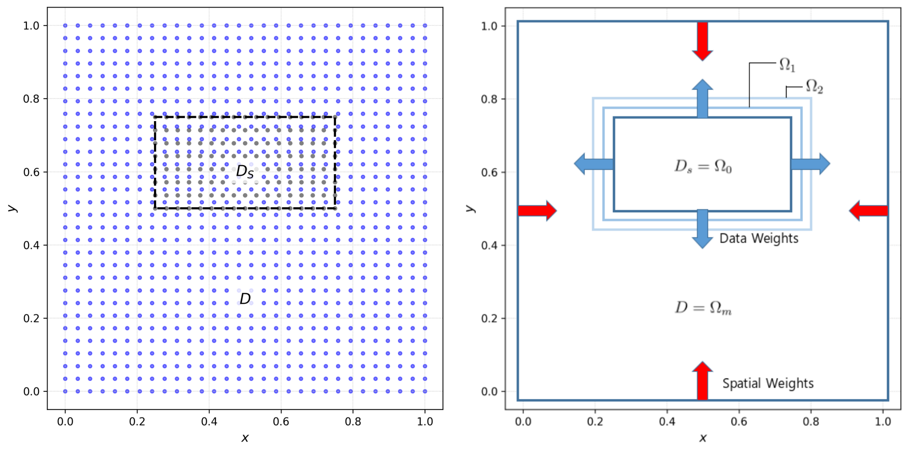


Figure 7. (**Left**) Visualization of the domain D and observation subdomain D_s for the elliptic inverse problem. Blue points indicate interior training points, while gray points represent observation data points. (**Right**) A schematic illustration of how $\Omega_0 = D_s$ expands layer by layer ($\Omega_1, \Omega_2, \ldots$ toward the full domain $\Omega_m = D$). The blue arrows represent data weights propagating outward from the observation subdomain D_s, while the red arrows indicate spatial weights moving inward from the ∂D.

5.2. Residuals and Network Smoothness

Both $u(x,y)$ and $f(x,y)$ are approximated by deep neural networks $u_\theta(x,y)$ and $f_\eta(x,y)$, respectively. The PDE residual is defined as

$$\mathcal{R}_{\theta,\eta}(x,y) = -\nabla \cdot (\sigma \nabla u_\theta(x,y)) + c(x,y)u_\theta - f_\eta(x,y), \quad (x,y) \in D.$$

For the boundary data, the residual is

$$\mathcal{R}_{sb,\theta}(x,y) = \sigma \frac{\partial u_\theta}{\partial \nu} - g(x,y), \quad (x,y) \in \partial D.$$

The data residual corresponding to observation data is

$$\mathcal{R}_{d,\theta}(x,y) = u_\theta(x,y) - u_s(x,y), \quad (x,y) \in D_s.$$

5.3. Causal Weights

The elliptic problem differs from the wave and parabolic cases in that the observation data are confined to a specific subdomain D_s, and the boundary conditions are provided on ∂D. We construct data weights that expand outward to propagate the influence of the observation data and boundary conditions on the entire domain. Specifically, we divide the domain D into a sequence of expanding regions:

$$D_s = \Omega_0 \subset \Omega_1 \subset \Omega_2 \subset \cdots \subset \Omega_m = D.$$

(See Figure 7 for a schematic illustration of how these regions Ω_j expand outward from D_s).

Data weights: For points x_i within Ω_0, the causal weight is set to 1. For points in each subsequent layer Ω_j (for $j \geq 1$), the weight is defined as

$$w_d(x_i, y_i) = \begin{cases} 1, & \text{if } (x_i, y_i) \in D_s = \Omega_0, \\ \exp\Big(-\epsilon \sum_{(x_k, y_k) \in \Omega_{j-1}} \mathcal{L}_r((x_k, y_k), \theta, \eta)\Big), & \text{if } (x_i, y_i) \in \Omega_j \setminus \Omega_{j-1}, \quad j = 1, 2, \ldots m. \end{cases}$$

Spatial weights: To ensure that the boundary conditions are accurately enforced, we define causal weights for the domain boundaries in the x and y directions as

$$w_x(x_i) = \begin{cases} 1, & x_i \in \partial D \\ \max\left[\exp\left(-\epsilon \sum_{k=1}^{i-1} \mathcal{L}_r(x_k, \theta, \eta)\right), \right. & \\ \qquad \left. \exp\left(-\epsilon \sum_{k=1}^{N_x - i} \mathcal{L}_r(x_{N_x - k+1}, \theta, \eta)\right)\right], & x_i \in D \setminus \partial D \end{cases}$$

and similarly for y:

$$w_y(y_i) = \begin{cases} 1, & y_i \in \partial D \\ \max\left[\exp\left(-\epsilon \sum_{k=1}^{i-1} \mathcal{L}_r(y_k, \theta, \eta)\right), \right. & \\ \qquad \left. \exp\left(-\epsilon \sum_{k=1}^{N_y - i} \mathcal{L}_r(y_{N_y - k+1}, \theta, \eta)\right)\right], & y_i \in D \setminus \partial D \end{cases}$$

The final causal weight for a point x_i is then defined by combining the weights from the observation and boundary considerations:

$$w_{xyd}(x_i, y_i) = \max(w_d(x_i, y_i), w_x(x_i), w_y(y_i))$$

As shown in Figure 7, the domain D is decomposed into nested regions Ω_0, $\Omega_1, \ldots, \Omega_m$. The data weights (blue arrows in the right panel) propagate outward from the observation subdomain $D_s = \Omega_0$, while the spatial weights (red arrows) move inward from the boundary of D. Consequently, the influence of observation data and boundary conditions gradually converges toward the domain's interior. This ensures that the entire domain is guided by critical constraints—observation data in D_s and boundary conditions on ∂D—ultimately leading to more stable and accurate reconstructions. The flow of causal weights depicted in the right panel of Figure 7 thus matches the final distribution of weights observed during training.

5.4. Loss Function

The overall loss function for the elliptic inverse problem is constructed by combining the weighted residual losses from the PDE, the boundary conditions, and the internal measurement data:

$$
\begin{aligned}
\mathcal{L}(\theta,\eta) = & \sum_{\substack{i=1\\(x_i,y_i)\in D}}^{N_{int}} w_{xyd}(x_i,y_i)|\mathcal{R}_{\theta,\eta}(x_i,y_i)|^2 + \\
& \sum_{\substack{i=1\\(x_i,y_i)\in \partial D}}^{N_{sb}} |\mathcal{R}_{sb,\theta}(x_i,y_i)|^2 + \sum_{\substack{i=1\\(x_i,y_i)\in D_s}}^{N_{d}} |\mathcal{R}_{d,\theta}(x_i,y_i)|^2.
\end{aligned}
$$

Here, w_{xyd} are the causal weights, which are computed so that critical regions (such as the boundary or observation subdomain) exert a greater influence during training. However, to address the ill-posed nature inherent in differentiating u to recover the source f, we augment this basic loss function with Lipschitz-based regularization terms:

$$
\begin{aligned}
\mathcal{L}_R(\theta,\eta) = & \mathcal{L}(\theta,\eta) + \lambda \max_{1\le i\le N_{int}} \left\|\nabla f_\eta(x_i,y_i)\right\|^2 + \lambda \max_{1\le i\le N_d} \|\nabla u_\theta(x_i,y_i)\|^2 + \\
& \lambda \max_{1\le i\le N_{int}} \|\nabla \mathcal{P}[u_\theta](x_i,y_i)\|^2 + \lambda \max_{1\le i\le N_{sb}} \|\nabla B[u_\theta](x_i,y_i)\|^2.
\end{aligned}
$$

where $\mathcal{P}[u_\theta]$ represents the PDE operator $(-\nabla\cdot(\sigma(\nabla u_\theta) + cu_\theta)$ and $\mathcal{B}[u_\theta]$ corresponds to the boundary residual (e.g., the Neumann boundary condition). The parameter $\lambda = 10^{-4}$ scales these gradient-based penalties. By penalizing large gradients in the source network f_η, the solution network u_θ, the PDE residual $\mathcal{P}[u_\theta]$, and the boundary residual $\mathcal{B}[u_\theta]$, this Lipschitz regularization effectively enforces higher regularity on the network outputs. Such regularity is crucial for stabilizing the differentiation process needed to recover f, thereby mitigating the ill-posed nature of the inverse source problem.

5.5. Numerical Experiments

This section presents the numerical experiments for the inverse source problem for the elliptic equation. All experiments follow the setup described in Section 5.1, and we provide additional details on the network architectures and hyper-parameters used. In our implementation, two deep neural networks are employed: one to approximate the solution $u(x,y)$ with parameters θ, and the other to approximate the unknown source term $f(x,y)$ with parameters η. The network $u_\theta(x,y)$ consists of 2 hidden layers with 10 neurons per layer, while the network $f_\eta(x,y)$ is deeper, comprising 14 hidden layers with 10 neurons per layer. Both networks utilize the *tanh* activation function. Training is performed using the Adam optimizer with a learning rate of 1×10^{-4} for 5×10^5 epochs.

Figure 8 presents the reconstruction results. The top row compares the exact solution $u(x)$ (left) with the CI-PINN prediction (middle), along with the corresponding absolute error (right). The bottom row shows the exact source term $f(x)$ (left), its reconstruction (middle), and the absolute error (right).

Figure 9 illustrates the evolution of the spatial causal weights during training. Initially, the weights are concentrated in the observation subdomain D_s and along the domain boundaries. As training progresses, the weights gradually propagate into the interior, ensuring the model captures the solution's local and global features.

Figure 10 presents the L^2 errors during training for both u and f. Since this problem does not involve a temporal domain, the Causal PINN approach introduced in Section 2.3 [23] is not applicable here. Therefore, we compare only conventional PINNs and CI-PINNs. The smoother convergence of CI-PINNs, in contrast to the oscillatory behavior observed in conventional PINNs, further demonstrates the effectiveness of the causal weighting strategy in improving stability and accuracy. Quantitatively, the L^2 relative error for the solution u of conventional PINNs is 0.06823, while CI-PINNs achieve an error of 0.033448—a reduction of over 50%. For the source term f, conventional PINNs yield an L^2 error of 0.08981, whereas CI-PINNs reduce the error to 0.06434, corresponding to an improvement of approximately 28%. Moreover, conventional PINNs exhibit significant oscillations in the error for f during the training process, indicating unstable convergence behavior. In contrast, CI-PINNs show a much smoother and more monotonic decrease in error, reflecting improved stability and more consistent propagation from the supervised region.

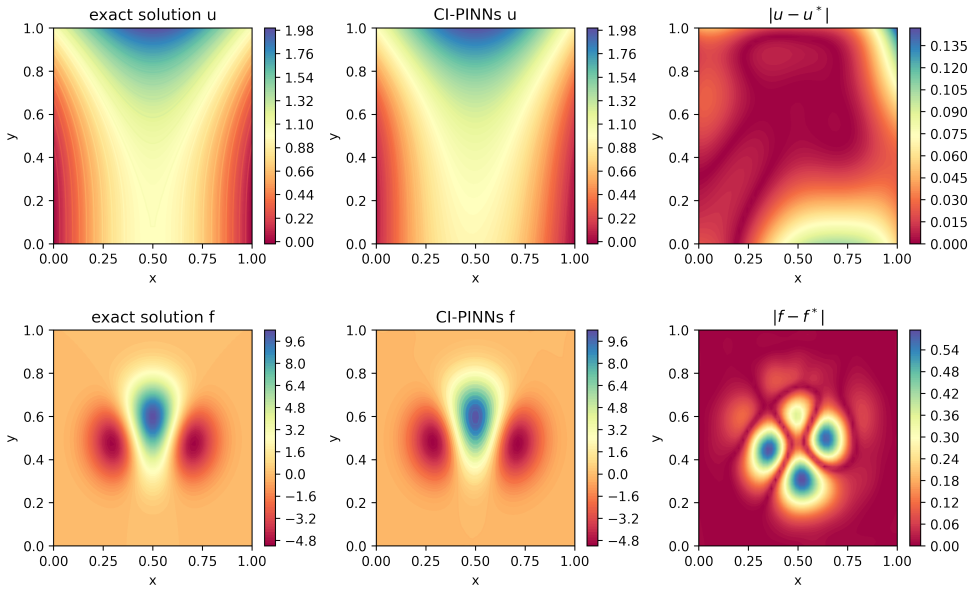


Figure 8. Visualization of results for the elliptic inverse problem. The top row compares the exact solution u (**upper left**) with the CI-PINN prediction (**upper middle**) and their absolute difference $|u - u^*|$ (**upper right**). The bottom row illustrates the exact source term f (**upper left**), the CI-PINNs' reconstruction (**upper middle**), and their absolute difference $|f - f^*|$ (**upper right**).

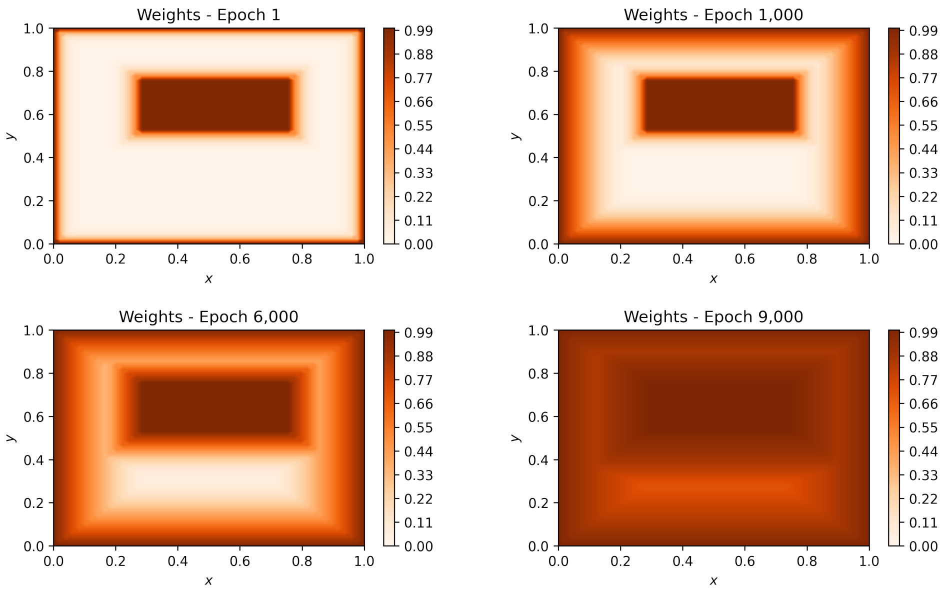


Figure 9. Evolution of spatial causal weights during the training process for the elliptic problem. The central rectangular region corresponds to the observation domain D_s, where data are available. The edges represent the domain boundaries ∂D. The weights progressively spread outward as training advances, enabling accurate reconstruction throughout the domain.

Table 1 presents the number of iterations and iteration times for vanilla PINNs and CI-PINNs. While both methods exhibit similar iteration times for the wave and parabolic problems, CI-PINNs run slower in the elliptic case. This slowdown is primarily due to the additional computational overhead required to define and process the expanding subdomains, Ω_k, which propagate the influence of the observation data and boundary conditions. In contrast, for other inverse problems, CI-PINNs can be applied without such subdomain constructions.

These results demonstrate that the CI-PINN framework significantly improves reconstruction accuracy for the elliptic inverse problem. The method reduces the final L^2 errors for u and f and ensures smoother convergence during training, thereby providing a more robust and reliable solution.

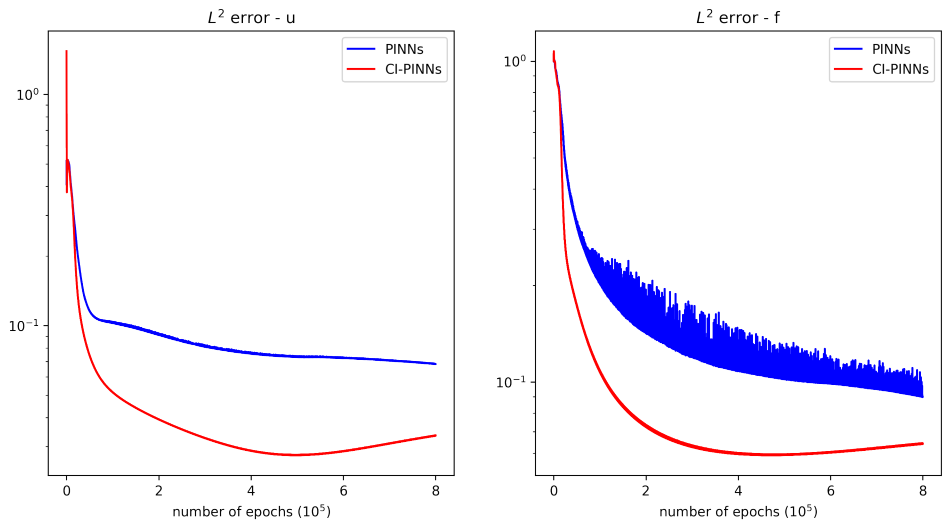


Figure 10. Convergence comparison for the elliptic inverse problem. The left panel shows the L^2 error for the solution u, while the right panel illustrates the L^2 error for the source term f.

6. Conclusions

In this paper, we introduced a novel framework called Causal Inverse PINNs (CI-PINNs), marking a significant advancement in physics-informed machine learning for solving inverse problems governed by partial differential equations. By integrating multidirectional causal weights into the loss function, our approach prioritizes learning in supervised regions to capture physical systems' inherent causal structure, such as initial conditions, boundary conditions, and observational data. This innovative strategy directly addresses one of the key limitations of conventional PINNs, resulting in enhanced stability and improved reconstruction accuracy.

Comprehensive numerical experiments on the wave equation and inverse source problems for the parabolic and elliptic equation demonstrated that CI-PINNs significantly outperform standard methods in convergence behavior and solution fidelity. The framework's ability to enforce the natural causal progression within the system enables more accurate prediction of unknown parameters and solutions, even in sparse data scenarios. These results underscore the robustness and generalizability of our proposed method across various types of inverse problems.

While incorporating causal weights represents a major innovation, the development and application of CI-PINNs also attain several practical challenges. In particular, careful tuning of the causality parameter is essential, as suboptimal settings can impede effective supervision propagation or lead to premature convergence. Additionally, the extra computational overhead of calculating multidimensional causal weights poses a challenge when extending the approach to high-dimensional or more complex systems. Another notable difficulty arises from applying the method to diverse domain geometries and configurations. The performance and stability of CI-PINNs can vary significantly depending on the domain shape, boundary conditions, and data distribution, indicating that further investigation is needed to adapt and optimize the framework for a broader range of practical scenarios.

In summary, CI-PINNs not only establish a new benchmark for stability and accuracy in solving inverse problems but also offer a flexible and scalable framework that can be adapted to a wide range of scientific and engineering applications. We believe this work paves the way for diverse future research directions, including developing adaptive weighting strategies, integrating uncertainty quantification techniques, and extending the framework to tackle nonlinear multi-physics systems, more complex domain geometries, and realistic noisy data scenarios.

Author Contributions: Conceptualization, H.S.; Methodology, J.K.; Writing—original draft, J.K.; Visualization, J.K.; Supervision, H.S. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: All data used in this study can be generated following the procedures described in the paper.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. LeCun, Y.; Bengio, Y.; Hinton, G. Deep learning. *Nature* **2015**, *521*, 436–444. [PubMed]
2. Schmidhuber, J. Deep learning in neural networks: An overview. *Neural Netw.* **2015**, *61*, 85–117. [PubMed]
3. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. Imagenet classification with deep convolutional neural networks. In Proceedings of the Advances in Neural Information Processing Systems 25 (NIPS 2012): 26th Annual Conference on Neural Information Processing Systems 2012, Lake Tahoe, NV, USA, 3–6 December 2012.
4. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778.
5. Sutskever, I. Sequence to Sequence Learning with Neural Networks. *arXiv* **2014**, arXiv:1409.3215.
6. Vaswani, A. Attention is all you need. In Proceedings of the Advances in Neural Information Processing Systems 2017, Long Beach, CA, USA, 4–9 December 2017.
7. Devlin, J. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv* **2018**, arXiv:1810.04805.
8. Xu, X.; Pu, X.; Zhang, J. Asymptotic limit of the Navier-Stokes-Poisson-Korteweg system in the half-space. *J. Differ. Equ.* **2022**, *335*, 201–243.
9. Wang, W.; Zeng, W.; Chen, W. Global exponential stability of periodic solutions for inertial delayed BAM neural networks. *Commun. Nonlinear Sci. Numer. Simul.* **2025**, *145*, 108728.
10. Guo, B.; Huang, D.; Zhang, J. Decay of solutions to a two-layer quasi-geostrophic model. *Anal. Appl.* **2017**, *15*, 595–606.
11. Wang, W.; Wu, J.; Chen, W. The characteristics method to study global exponential stability of delayed inertial neural networks. *Math. Comput. Simul.* **2025**, *232*, 91–101.
12. Raissi, M.; Perdikaris, P.; Karniadakis, G.E. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *J. Comput. Phys.* **2019**, *378*, 686–707. [CrossRef]
13. Karniadakis, G.E.; Kevrekidis, I.G.; Lu, L.; Perdikaris, P.; Wang, S.; Yang, L. Physics-informed machine learning. *Nat. Rev. Phys.* **2021**, *3*, 422–440. [CrossRef]
14. Lu, L.; Meng, X.; Mao, Z.; Karniadakis, G.E. DeepXDE: A deep learning library for solving differential equations. *SIAM Rev.* **2021**, *63*, 208–228. [CrossRef]
15. Sirignano, J.; Spiliopoulos, K. DGM: A deep learning algorithm for solving partial differential equations. *J. Comput. Phys.* **2018**, *375*, 1339–1364. [CrossRef]
16. Zhu, Y.; Zabaras, N.; Koutsourelakis, P.S.; Perdikaris, P. Physics-constrained deep learning for high-dimensional surrogate modeling and uncertainty quantification without labeled data. *J. Comput. Phys.* **2019**, *394*, 56–81. [CrossRef]
17. Chen, Y.; Lu, L.; Karniadakis, G.E.; Dal Negro, L. Physics-informed neural networks for inverse problems in nano-optics and metamaterials. *Opt. Express* **2020**, *28*, 11618–11633. [CrossRef]

18. Lu, L.; Pestourie, R.; Yao, W.; Wang, Z.; Verdugo, F.; Johnson, S.G. Physics-informed neural networks with hard constraints for inverse design. *SIAM J. Sci. Comput.* **2021**, *43*, B1105–B1132. [CrossRef]
19. Jagtap, A.D.; Mao, Z.; Adams, N.; Karniadakis, G.E. Physics-informed neural networks for inverse problems in supersonic flows. *J. Comput. Phys.* **2022**, *466*, 111402. [CrossRef]
20. Son, H.; Lee, M. A PINN approach for identifying governing parameters of noisy thermoacoustic systems. *J. Fluid Mech.* **2024**, *984*, A21. [CrossRef]
21. Yang, L.; Meng, X.; Karniadakis, G.E. B-PINNs: Bayesian physics-informed neural networks for forward and inverse PDE problems with noisy data. *J. Comput. Phys.* **2021**, *425*, 109913. [CrossRef]
22. Blalock, D.; Gonzalez Ortiz, J.J.; Frankle, J.; Guttag, J. What is the state of neural network pruning? *Proc. Mach. Learn. Syst.* **2020**, *2*, 129–146.
23. Wang, S.; Sankaran, S.; Perdikaris, P. Respecting causality for training physics-informed neural networks. *Comput. Methods Appl. Mech. Eng.* **2024**, *421*, 116813.
24. Son, H.; Cho, S.W.; Hwang, H.J. Enhanced physics-informed neural networks with augmented Lagrangian relaxation method (AL-PINNs). *Neurocomputing* **2023**, *548*, 126424.
25. Jung, J.; Kim, H.; Shin, H.; Choi, M. CEENs: Causality-enforced evolutional networks for solving time-dependent partial differential equations. *Comput. Methods Appl. Mech. Eng.* **2024**, *427*, 117036.
26. Li, Y.; Chen, S.; Shan, B.; Huang, S.J. Causality-enhanced discreted physics-informed neural networks for predicting evolutionary equations. In Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence, Jeju, Republic of Korea, 3–9 August 2024; pp. 4497–4505.
27. Jo, H.; Son, H.; Hwang, H.J.; Kim, E. Deep neural network approach to forward-inverse problems. *arXiv* **2019**, arXiv:1907.12925.
28. Han, J.; Jentzen, A.; E, W. Solving high-dimensional partial differential equations using deep learning. *Proc. Natl. Acad. Sci. USA* **2018**, *115*, 8505–8510.
29. E, Y.; Yu, B. The deep Ritz method: A deep learning-based numerical algorithm for solving variational problems. *Commun. Math. Stat.* **2018**, *6*, 1–12.
30. Jagtap, A.D.; Kharazmi, E.; Karniadakis, G.E. Conservative physics-informed neural networks on discrete domains for conservation laws: Applications to forward and inverse problems. *Comput. Methods Appl. Mech. Eng.* **2020**, *365*, 113028.
31. Mishra, S.; Molinaro, R. Estimates on the generalization error of physics-informed neural networks for approximating PDEs. *IMA J. Numer. Anal.* **2023**, *43*, 1–43.
32. Bardos, C.; Lebeau, G.; Rauch, J. Un exemple d'utilisation des notions de propagation pour le contrôle et la stabilisation de problemes hyperboliques. *Rend. Sem. Mat. Univ. Politec. Torino* **1988**, *46*, 11–31.
33. Le Rousseau, J.; Lebeau, G.; Terpolilli, P.; Trélat, E. Geometric control condition for the wave equation with a time-dependent observation domain. *Anal. PDE* **2017**, *10*, 983–1015.
34. Zhang, M.; Li, Q.; Liu, J. On stability and regularization for data-driven solution of parabolic inverse source problems. *J. Comput. Phys.* **2023**, *474*, 111769. [CrossRef]
35. Isakov, V. *Inverse Problems for Partial Differential Equations*; Springer: Berlin/Heidelberg, Germany, 2006.
36. Prilepko, A.I.; Kostin, A.B. On certain inverse problems for parabolic equations with final and integral observation. *Mat. Sb.* **1992**, *183*, 49–68. [CrossRef]
37. Prilepko, A.I.; Orlovsky, D.G.; Vasin, I.A. *Methods for Solving Inverse Problems in Mathematical Physics*; CRC Press: Boca Raton, FL, USA, 2000.
38. Zhang, H.; Liu, J. Solving an inverse source problem by deep neural network method with convergence and error analysis. *Inverse Probl.* **2023**, *39*, 075013.