Gradient Enhanced Self-Training Physics-Informed Neural Network (gST-PINN) for Solving Nonlinear Partial Differential Equations

Narayan S Iyer[1*], Bivas Bhaumik[2*], Ram S Iyer[3] and Satyasaran Changdar[4]

[1]Department of Physics and Astronomy, National Institute of Technology Rourkela, 769008, Odisha, India.
[2]Department of Mathematics, National Institute of Technology Rourkela, 769008, Odisha, India.
[3]Department of Electronics Engineering, Rajiv Gandhi Institute of Petroleum Technology, 229304, Uttar Pradesh, India.
[4]Department of Food Science, University of Copenhagen, Copenhagen, Denmark.

*Corresponding author(s). E-mail(s): narayansiyer7@gmail.com; bhaumikbivas@gmail.com;
Contributing authors: ramsiyer77@gmail.com; sach@di.ku.dk;

Abstract

Partial differential equations (PDEs) provide a mathematical foundation for simulating and understanding intricate behaviors in both physical sciences and engineering. With the growing capabilities of deep learning, data-driven approaches like Physics-Informed Neural Networks (PINNs) have been developed, offering a mesh-free, analytic type framework for efficiently solving PDEs across a wide range of applications. However, traditional PINNs often struggle with challenges such as limited precision, slow training dynamics, lack of labeled data availability, and inadequate handling of multi-physics interactions. To overcome these challenging issues of PINNs, we proposed a Gradient Enhanced Self-Training PINN (gST-PINN) method that specifically introduces a gradient based pseudo point self-learning algorithm for solving PDEs. We tested the proposed method on three different types of PDE problems from various fields, each representing distinct scenarios. The effectiveness of the proposed method is evident, as the PINN approach for solving the Burgers' equation attains a mean square error

($\boldsymbol{MSE}$) on the order of $\mathbf{10^{-3}}$, while the diffusion–sorption equation achieves an MSE on the order of $\mathbf{10^{-4}}$ after $\mathbf{12,500}$ iterations, with no further improvement as the iterations increase In contrast, the MSE for both PDEs in the gST-PINN model continues to decrease, demonstrating better generalization and reaching an MSE on the order of $\mathbf{10^{-5}}$ after 18,500 iterations. Furthermore, the results show that the proposed purely semi-supervised gST-PINN consistently outperforms the standard PINN method in all cases, even when solution of the PDEs are unavailable. It generalizes both PINN and Gradient-enhanced PINN (gPINN), and can be effectively applied in scenarios prone to low accuracy and convergence issues, particularly in the absence of labeled data.

Keywords: Physics Informed Neural Network, Non-Linear PDEs, Gradient Enhanced Pseudo Points, Gradient Residual Filtering, Self Semi-supervised Training

1 Introduction

1.1 Scientific And Physics Informed Machine Learning For Solving Non Linear PDEs

Computational efficiency of traditional numerical methods in addressing partial differential equations (PDEs) arising in non trivial situations such as Riemannian manifolds, complex manifolds (i.e. specifically non Euclidean spaces), high dimensional spaces and inverse problem solving have always been a topic of concern. The advent of deep learning has facilitated data-driven models in PDE solving, yet their precision remains inherently tied to the size, distribution, and representativeness of the training data. Physics informed machine learning using physics informed neural networks [1–6] has been successful in tacking the issue till an extent, which unlike traditional numerical solvers and data driven methods solves the PDE by encoding the physics information such as the PDE itself, boundary conditions, initial conditions, any other constraints such as symmetries etc. into the loss function, thereby converting numerical analysis problem into an optimization problem. The accuracy of PINNs over other traditional solvers can be attributed to mainly two of its charecteristics: firstly they are mesh free, due to which they can randomly sample points in the domain, and secondly they ideally do not require any labeled data for functioning, nevertheless a small amount of labeled data can help in better convergence to the solution. Despite theoretical guarantees, the accuracy and efficiency of ordinary PINNs remain insufficient for practical applications [7, 8]. A sparse amount of work targeted in this direction have been conducted in recent years. Numerous modifications to standard PINNs have been researched, yielding promising results in some cases [9]. However, robust models that consistently perform well across a broad range of PDE types with diverse applications in science and engineering remain elusive.

1.2 Self Training

Self training approaches [10–12], which is a branch of semi-supervised learning [13, 14] have become increasingly popular in recent years due to their ability in harnessing limited labeled data and plentiful unlabeled datasets, thereby combining them to enhance predictive performance across various tasks. The core concept of self training revolves around repeatedly assigning pseudo labels to unlabeled samples having confidence scores exceeding a specified threshold, augmenting the labeled dataset and retraining. Some of the popular pseudo-labeling strategies include threshold based methods, proportion based methods, curriculum learning based methods majority vote classifiers and adaptive thresholding methods. A mathematical description of self training is provided in [10].

1.3 Related works

To resolve the challenges related to PINNs (in terms of accuracy, efficiency, solving PDEs in complex manifolds, solving higher dimensional PDEs, etc.), various modifications to the standard PINN framework have been proposed by several researchers, thereby yielding novel algorithms and models. For instance, the Bayesian physics-informed neural network (B-PINNs) was proposed by L Yang et al. [15] which integrates Bayesian neural networks with traditional PINNs to solve forward and inverse PDEs with noisy measurements as input data, which enabled uncertainty quantification and more accurate predictions compared to PINNs, especially in extremely noisy scenarios. A similar augmentation to the standard PINN framework was formulated by A D Jagtap et al. [16], the conservative physics-informed neural network (cPINN), which functions by decomposing the computational domain into discrete sub-domains, each with its own deep or shallow neural network (which is dependent on the complexity of the solution in a particular sub-domain) trained separately and by enforcing flux continuity at the sub-domain interfaces for preserving the conservation laws. cPINN has been analyzed to have multiple advantages such as reduced spread of error across the domain, efficient fine-tuning of hyper parameters and enhanced representation ability of the network. Albeit exceptionally accurate across a wide range of PDEs, cPINN fails to perform in situations involving complex-geometrical and highly irregular domains. To address this shortcoming of cPINN, G E Karniadakis et al. presented the Extended physics-informed neural network (XPINN) [17], which is based on a generalized space-time computational domain decomposition algorithm for PINNs. In addition to preserving the advantages of cPINN over the classical PINNs, XPINN was proved to be highly applicable to multi-physics and dynamic interface problems, and also hold certain exclusive benefits while addressing the same. Moreover, the domain decomposition algorithm based on XPINN is extendable to any type of PDEs, regardless of its physical nature. Another non-trivial limitation of PINNs which have been less researched on is the Unbalanced Prediction Problem (UPP) which primarily arises due to the equal treatment of easy (where solutions are relatively smooth) and hard (points in regions with steep gradients, points near boundaries, etc. where approximation of solution is relatively harder) sample points by PINNs, thereby leading to unstable learning of the network. To overcome this problem, S Duan et al. proposed cognitive physics-informed

neural network (CoPINN) [18], which operates by implicitly evaluating the hardness of every sample during training, based on the PDE residual gradient. Apart from these proposals, numerous modifications on the standard PINN structure have been researched on, tailored at specifically finding applications in fluid dynamics [19, 20], power systems [21] and financial mathematics [22, 23].

1.4 Overview of the paper and our contribution

The article is organized as follows: In section (2), a concise and mathematically streamlined overview of PINN is presented followed by the gPINN methodology for solving non-linear PDEs, concluding with the discussion on formulation of the total loss function for the gPINN model. In section (3), we introduce our proposed gST-PINN model. Subsequently, in the subsections (3.1) and (3.2), the overview of the framework, the gST-PINN model architecture and the gradient enhanced pseudo point generation algorithm is discussed in detail. In section (4), various experimental results (section (3.1): Burgers' equation, section (3.2): diffusion reaction equation, section (3.3): diffusion sorption equation) have been demonstrated to support the arguments made in section (3). Consequently, in section (3.4), a detailed comparison of the gST-PINN model's performance with the other state-of-the-art models have been discussed. In section (5) (appendix section), a mathematical overview of semi-supervised learning and self training is provided. Finally, the concluding remarks and potential future research trajectories have been summarized in section (6).
Our key contributions include:

1. We formulate a novel gradient enhanced pseudo point generation algorithm which utilizes the gradient information of the PDE residual under consideration for sorting and selecting the collocation points for pseudo labeling.
2. We propose a novel gradient enhanced self training physics informed neural network model (gST-PINN) based on our novel pseudo labeling algorithm.
3. We provide an evaluation of the accuracy our model in terms of the MSE and L_2 relative errors of the predicted solution to various non-linear PDEs and perform a comparative analysis with a wide range of scientific machine learning settings.

2 Methodology

2.1 Physics Informed Neural Network (PINN) for solving PDEs

We first consider the general form of a PDE for the solution $u(\mathbf{x})$ defined on a domain $\Omega \subset \mathbb{R}^N$ with a parametrization $\zeta = (\zeta_1,\ \zeta_2, ...)$ and $\mathbf{x} = (x_1,\ x_2, ...,\ x_N) \in \Omega$:

$$G(\mathbf{x}; \frac{\partial u}{\partial x_1}, ..., \frac{\partial u}{\partial x_N}; \frac{\partial^2 u}{\partial x_1 \partial x_1}, ..., \frac{\partial^2 u}{\partial x_1 \partial x_N}, ..., \frac{\partial^2 u}{\partial x_N \partial x_N}, ...; \zeta) = 0 \tag{1}$$

along with the boundary condition (Dirichlet, Periodic, etc.) defined on $\partial\Omega$ and additional constraints (such as initial condition, symmetries, etc.) respectively:

$$D(u; \mathbf{x}) = 0 \quad \text{and} \quad S(u, \mathbf{x}) = 0. \tag{2}$$

The PDE can be solved by constructing a neural network $\hat{u}(\mathbf{x}, \tilde{\theta})$ using trainable parameters $\tilde{\theta}$ to estimate the solution $u(\mathbf{x})$ and consecutively optimizing $\tilde{\theta}$ using the constraints dictated by the PDE (through equations (1) and (2)). A set of clustered points $\mathcal{T} = \mathcal{T}_G \cup \mathcal{T}_D \cup \mathcal{T}_S = \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_{|\mathcal{T}|}\}$ of size $|\mathcal{T}|$ represents the collocation points chosen randomly to train the network where $\mathcal{T}_G \subset \Omega$ of size $|\mathcal{T}_G|$, $\mathcal{T}_D \subset \partial\Omega$ of size $|\mathcal{T}_D|$ and $\mathcal{T}_S \subset \Omega$ of size $|\mathcal{T}_S|$ separately denotes the collocation points on the domain, boundary and for additional constraints respectively. To optimize the parameters $\tilde{\theta}$, we utilize the loss function $\mathcal{L}(\tilde{\theta}, \mathcal{T})$ defined as

$$\mathcal{L}(\tilde{\theta}; \mathcal{T}) = w_G \mathcal{L}_G(\tilde{\theta}; \mathcal{T}_G) + w_D \mathcal{L}_D(\tilde{\theta}; \mathcal{T}_D) + w_S \mathcal{L}_S(\tilde{\theta}; \mathcal{T}_S) \tag{3}$$

where w_G, w_D, w_S are the weights and the individual losses $\mathcal{L}_G$, $\mathcal{L}_D$ and $\mathcal{L}_D$ in accordance with equations (1) and (2) can be defined respectively as

$$\mathcal{L}_G(\tilde{\theta}; \mathcal{T}_G) = \frac{1}{|\mathcal{T}_G|} \sum_{\mathbf{x} \in \mathcal{T}_G} \left\| G(\mathbf{x}; \frac{\partial \hat{u}}{\partial x_1}, ...; \frac{\partial^2 \hat{u}}{\partial x_1 \partial x_1}, ...; \zeta) \right\|_2^2 \tag{4}$$

and

$$\mathcal{L}_D(\tilde{\theta}; \mathcal{T}_D) = \frac{1}{|\mathcal{T}_D|} \sum_{x \in \mathcal{T}_D} \|\mathcal{D}(\hat{u}, \mathbf{x})\|_2^2 \tag{5}$$

and

$$\mathcal{L}_S(\tilde{\theta}; \mathcal{T}_S) = \frac{1}{|\mathcal{T}_S|} \sum_{x \in \mathcal{T}_S} \|\mathcal{S}(\hat{u}, \mathbf{x})\|_2^2 \tag{6}$$

Here, $\|.\|_2$ indicates the L^2 norm. The optimal $\tilde{\theta}$ upon training of the network $u(\mathbf{x}, \tilde{\theta})$ using the above loss function is $\tilde{\theta}^* = \arg\min_{\tilde{\theta}} \mathcal{L}(\tilde{\theta}; \mathcal{T})$. Alternatively, the value of function G in equation (1) is called the PDE residual and $\mathcal{T}_G$ the set of residual points.

2.2 Gradient enhanced physics informed neural network (gPINN) for solving PDEs

gPINN [24] improves the traditional PINN framework by leveraging the derivative information of the PDE residual and using it to enrich the loss function. Referring to the PDE given in equation (1) again and considering the fact that G vanishes for all $\mathbf{x} \in \Omega$, a direct inference can be made that the partial derivative of G with respect to each component of x (i.e $x_1, x_2, ..., x_N$) is also zero for all $x \in \Omega$.

$$\therefore \nabla G(\mathbf{x}) = \left(\frac{\partial G}{\partial x_1}, \frac{\partial G}{\partial x_2}, ..., \frac{\partial G}{\partial x_N} \right) = \mathbf{0}, \quad \forall \mathbf{x} \in \Omega. \tag{7}$$

The additional loss term as a consequence of the above information is

$$\mathcal{L}_{g_i}(\tilde{\theta}; \mathcal{T}_{g_i}) = \frac{1}{|\mathcal{T}_{g_i}|} \sum_{x \in \mathcal{T}_{g_i}} \left\| \frac{\partial G}{\partial x_i} \right\|_2^2 \tag{8}$$

where $\mathcal{T}_{g_i}$ represents the set of collocation points for the partial derivative $\frac{\partial G}{\partial x_i}$, with size $|\mathcal{T}_{g_i}|$. Upon using appropriate weights w_{g_i}, the total loss function of equation (3) modifies to

$$\mathcal{L} = w_G\mathcal{L}_G + w_D\mathcal{L}_D + w_S\mathcal{L}_S + \sum_{i=1}^{N} w_{g_i}\mathcal{L}_{g_i}(\tilde{\theta}; \mathcal{T}_{g_i}). \tag{9}$$

3 Gradient Enhanced Self Training Physics Informed Neural Network (gST-PINN): Proposed Approach

This section outlines the proposed gST-PINN framework in detail, which implicitly combines gPINN framework with the pseudo point generation strategy, thus developing a novel gradient enhanced self training model that is physics aware.

3.1 Overview of the framework

The architecture of the proposed framework is illustrated in figure (1). The network takes in $(x_1,\ x_2,\ x_3, ...,\ x_N) \in \mathbb{R}^N$ (which in most cases will be coordinates in 3 dimensions constituting of the time coordinate t and the regular spatial coordinates $x,\ y$) as inputs and estimates $(v_1,\ v_2, ...,\ v_M) \in \mathbb{R}^M$ (which in most cases will be the vector fields, tensor fields and physical quantities such as velocity fields in two dimensions (u, v), density ρ, etc.) as outputs. Each iteration comprises two key steps: network training and gradient enhanced pseudo label generation which are represented in green and red colored lines respectively in figure (1).

1. *Network Training* : This part of the framework is similar to the normal PINNs. For the prediction of the solution $\hat{u}$ for the governing PDE, three sets of training points (or training data), namely residual points, boundary points and auxiliary constraint points such as collocation points for initial conditions, symmetry conditions, etc. are fed as inputs to the network as already mentioned in the above sections as $\mathcal{T}_G$, $\mathcal{T}_D$ and $\mathcal{T}_S$ respectively. The residual points are referred to as equation points in certain articles. The residual points are unsupervised points randomly sampled from the N-dimensional domain used to address the task of learning physics information through the minimization of residual loss (4) and thereby training the network by standard back propagation algorithm. The boundary points and the auxiliary constraint points can be either non-labeled or labeled data depending on whether the boundary and auxiliary conditions are given explicitly in analytic form (as in equation (2)) or not. If given in non-labelled form, the loss function can be modeled as given in equations (5) and (6) to learn the corresponding information. Ideally, PINNs do not require any supervised data for learning the PDE solution. But the provision of few intra-domain labeled data can help in improving accuracy and efficiency in terms of convergence speed.

2. *Gradient Enhanced Pseudo Label Generation* : This part of the iteration is what distinguishes the proposed framework from ordinary PINN and gPINN, thereby improving its accuracy. We introduce a novel gradient-enhanced pseudo-label generation technique that incorporates pseudo-labels, generated based on the PDE residual and its gradient information, into the training process. In gST-PINN, following a certain number of iterations, the neural network generates predictions for all sample points and

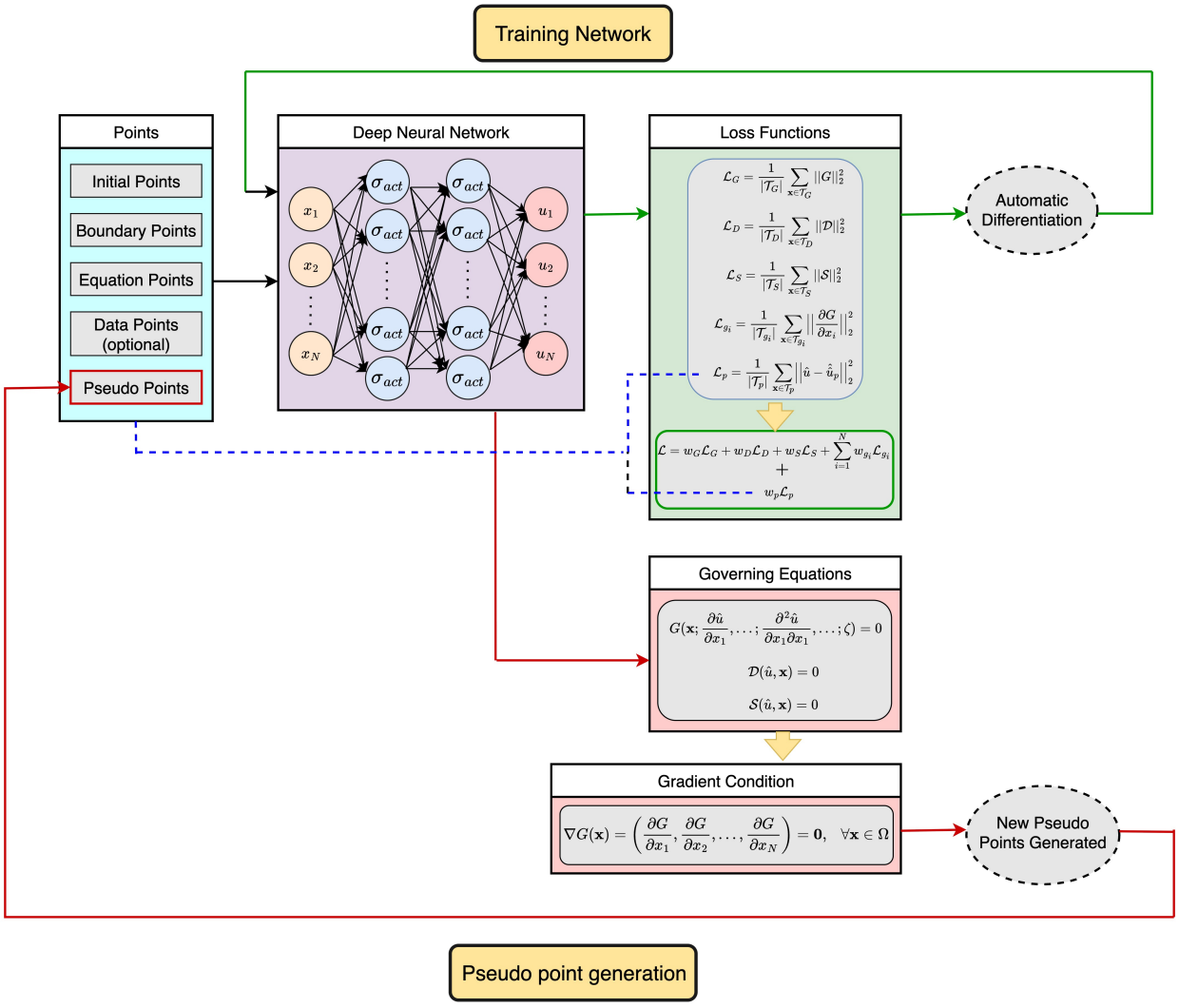


Fig. 1: Pipeline/Architecture of the gST-PINN model **(section (3))**. The green line depicts the neural network training process fulfilled by minimizing the loss function given by equation (9) **(section (2.1))** and the red line represents the gradient enhanced pseudo point generation algorithm/strategy accomplished by filtering and labeling the collocation points with considerably small PDE and gradient residuals given by equations (1) and (7) **(section(3.2), Algorithm 1**. Both the processes, apart from the incorporation of the physics based information (PDE residual, initial and boundary conditions, additional constraints and symmetries, etc.), also utilizes the gradient details **(section (2.2))** for escalating the overall accuracy of the model.

employs them to compute the governing equation's residual as well as partial derivative

of residual with respect to all the N coordinates $x_1,\ x_2, ...,\ x_N$ (or in a compact way, the gradient of residual, i.e. $\nabla \equiv (\frac{\partial}{\partial x_1},\ \frac{\partial}{\partial x_2}, ..., \frac{\partial}{\partial x_N})$. For $(x_1,\ x_2,\ x_3) = (x,\ y,\ t)$, $\frac{\partial}{\partial x},\ \frac{\partial}{\partial y},\ \frac{\partial}{\partial t}$ are computed).

3.2 Pseudo Point Generation Strategy

Generation of pseudo labels from selection followed by filtration of unsupervised data is a pivotal procedure in self training. However, the conventional pseudo-labeling method in classification, which relies on confidence thresholds to sort unlabeled data, cannot be directly applied to PINNs since solving a PDE is fundamentally different from solving a classification problem.

The pseudo point generation and labeling strategy of gST-PINN revolves around three crucial hyperparameters: q, the pseudo-label candidate rate or maximum rate, r, the pseudo-label selection threshold or the stable coefficient and p, the (pseudo-label) update frequency. The algorithm adopted by gST-PINN for the pseudo point selection and labeling is conspicuous through algorithm 1. Primarily, upon the initialization of the pseudo point coordinates as empty sets, the network predicts solutions $\hat{u}$ (of the equation (1) with constraints given by equation (2)) corresponding to each of the $|\mathcal{T}_G|$ collocation points fed to it and the PDE residual denoted by R_G based on the predictions are computed (lines 1-5). Now, the algorithm ranks the collocation points based on the residual and the top $q\%$ points with smallest residuals are sorted out for next filtration (line 6). This completes the initial phase of the algorithm.
The subsequent phase commences with the computation of the gradients of the PDE residual G with respect to each of the N coordinates x_0 to x_{N-1}, denoted by $g_i = \frac{\partial \hat{G}}{\partial x_i}$. Now, the residuals corresponding to each of the N PDEs represented by equation (7) for each of the initially sorted out top $q\%$ points in the first phase are evaluated. Furthermore, the gST-PINN generates N rankings of the $|\mathcal{T}_G| \times q$ collocation points (termed as idx_i; i = 0 to $N-1$), each corresponding to the N coordinates, on the basis of the respective gradient residual (equation (7)) and the top $q\%$ points among the available sample points with the smallest gradient residuals are promoted as candidates for pseudo points. This concludes the second phase of the algorithm (lines 7-13).
In the third phase, the algorithm keeps track of N flag accumulators, each corresponding to the N coordinates for the selection and labeling of pseudo points. The flag accumulators are updated (line 16) and if all the N flag accumulators flag_i to flag_{i+N-1} exceeds the pseudo label selection threshold r (line 18), the double filtered $|\mathcal{T}_G| \times q^2$ collocation points are elected as pseudo points. This summarizes the third phase of the gST-PINN algorithm (lines 14- 23). In the last phase, the flag counters of the $|\mathcal{T}_G|(1-q^2)$ collocation points which fail to qualify as pseudo points are reset to 0, which is substantial in order to disqualify those candidate points with low residuals, but inconsistent, thereby leveraging the overall accuracy of the gST-PINN model (lines 24-28).

4 Experimental Results

To assess the performance of our proposed gST-PINN approach, we conduct a comprehensive set of experiments on various PDEs representing diverse fields and scenarios.

Algorithm 1 Pseudo point generation

$x_{0_p}, x_{1_p}, x_{2_p}, ..., x_{N-1_p}, u_p \leftarrow \emptyset$
for $i = 0$ to $|\mathcal{T}_G| - 1$ **do**
  $\hat{u}^i \leftarrow \mathcal{N}^{(k)}(x_0^i, x_1^i, ..., x_{N-1}^i)$
  $R_G^i \leftarrow G(x_0^i, x_1^i, ..., x_{N-1}^i, \frac{\partial \hat{u}^i}{\partial x_0^i}, \frac{\partial \hat{u}^i}{\partial x_1^i}, ..., \frac{\partial \hat{u}^i}{\partial x_{N-1}^i}, \frac{\partial^2 \hat{u}^i}{\partial {x_0^i}^2}, \frac{\partial^2 \hat{u}^i}{\partial x_0^i \partial x_1^i}, ..., \frac{\partial^2 \hat{u}^i}{\partial {x_{N-1}^2}^i}, ..., \hat{u}^i, \zeta)$
end for
idx = argPartition$(R_G, x_0, x_1, ..., x_{N-1}, \hat{u}, q)$
for $i = 0$ to $N - 1$ **do**
  $g_i = \frac{\partial \hat{G}}{\partial x_i}$
  for $j = 0$ to $|\mathcal{T}_G| \times q - 1$ **do**
    $R_{g_i}^{\ j} \leftarrow g_i(x_0^{idx[j]}, x_1^{idx[j]}, .., x_{N-1}^{idx[j]}, \frac{\partial \hat{u}^{idx[j]}}{\partial x_0^{idx[j]}}, \frac{\partial \hat{u}^{idx[j]}}{\partial x_1^{idx[j]}}, .., \frac{\partial \hat{u}^{idx[j]}}{\partial x_{N-1}^{idx[j]}}, \frac{\partial^2 \hat{u}^{idx[j]}}{\partial {x_0^{idx[j]}}^2}, \frac{\partial^2 \hat{u}^{idx[j]}}{\partial x_0^{idx[j]} \partial x_1^{idx[j]}}, .., \frac{\partial^2 \hat{u}^{idx[j]}}{\partial {x_{N-1}^2}^{idx[j]}}, ..$
    $.., \hat{u}^{idx[j]}, \zeta)$
  end for
  $\text{idx}_i = \text{argPartition}(R_{g_i}, x_0, x_1, ..., x_{N-1}, \hat{u}, q)$
end for
for $i = 0$ to $N - 1$ **do**
  for $j = 0$ to $|\mathcal{T}_G| \times q^2 - 1$ **do**
    $\text{flag}_i[\text{idx}_i[j]] \mathrel{+}= 1$
  end for
  if $\text{flag}_i[\text{idx}_i[i]],\ \text{flag}_{i+1}[\text{idx}_{i+1}[i]], ...,\ \text{flag}_{i+N-1}[\text{idx}_{i+N-1}[i]] > r$ **then**
    $x_{0_p} \leftarrow x_{0_p} \cup x_0^i$
    $x_{1_p} \leftarrow x_{1_p} \cup x_1^i$
    $u_p \leftarrow u_p \cup \hat{u}^i$
  end if
end for
for $i = 0$ to $N - 1$ **do**
  for $j = |\mathcal{T}|_G \times q^2$ to $|\mathcal{T}_G| - 1$ **do**
    $\text{flag}_i[\text{idx}[j]] = 0$
  end for
end for

A comparative performance analysis with the standard PINN method is also presented in Table 1.

4.1 Burgers Equation

For a field $u(x, t)$, the general form of Burgers equation (also known as viscous Burgers equation or Bateman-Burgers equation) [25, 26] is given as

$$\frac{\partial}{\partial t}(u(t,x)) + \frac{\partial}{\partial x}(u^2(t,x)/2) = (\eta_v/\pi)\frac{\partial^2}{\partial x^2}, (u(t,x)) \qquad x \in (0,1),\ t \in [0,2]. \tag{10}$$

Here, η_v (*kinematic viscosity*) $\rightarrow 0.01$.
The initial condition for the above given Burgers equation (10), represented as a linear

combination of sinusoidal functions is provided in equation (11) as

$$u(0,x) = u_0(x) = \sum_{i=1}^{N} \lambda_i \sin(2\pi f_i/L_x)x + \Psi_i \approx \tilde{I}_1(x) \qquad x \in (0, L_x) \tag{11}$$

λ_i (*amplitude of the* i^{th} *sinusoidal function*) $\in$ [0,1]
Ψ_i (*phase shift of* i^{th} *sinusoidal function* $\in$ (0,2π)
f_i (*mode number determining the number of oscillations over the domain*) $\in \mathbb{N} := \{1, 2, 3, ...\}$
L_x (*domain length in the spatial direction* x) $\rightarrow 1$
The boundary condition considered is the periodic boundary condition, i.e.

$$u(t,0) = u(t,1) \approx \tilde{B}_1(t) \ \forall t \in [0,2] \tag{12}$$

The plots of $\tilde{B}_1(t)$ and $\tilde{I}_1(x)$ are represented in figure (2).
The derivatives of the Burgers' equation (10) with respect to the variable x and t (corresponding to x_1 and x_2 in equation (1)) can be evaluated as

$$\frac{\partial^2}{\partial x \partial t}(u(t,x)) + \frac{\partial^2}{\partial x^2}(u^2(t,x)/2) = (\eta_v/\pi)\frac{\partial^3}{\partial x^3}(u(t,x)) \tag{13}$$

and

$$\frac{\partial^2}{\partial t^2}(u(t,x)) + \frac{\partial^2}{\partial t \partial x}(u^2(t,x)/2) = (\eta_v/\pi)\frac{\partial^2}{\partial x^2}(u(t,x)). \tag{14}$$

Subsequently, the loss function for training the neural network can be formulated deploying equation (9) from equations (13) and (14) as

$$\begin{aligned}
\mathcal{L}_{Burgers'} = {} & \frac{w_G}{|\mathcal{T}_G|} \sum_{(x,t)\in\mathcal{T}_G} \left\| \frac{\partial(\hat{u}(t,x))}{\partial t} + \frac{\partial(\hat{u}^2(t,x)/2)}{\partial x} - (\eta_v/\pi)\frac{\partial^2(\hat{u}(t,x))}{\partial x^2} \right\|_2^2 \\
& + \frac{w_D}{|\mathcal{T}_D|} \sum_{(x,t)\in\mathcal{T}_D} \|\hat{u}(t,0) - \hat{u}(t,1)\|_2^2 + \frac{w_I}{|\mathcal{T}_I|} \sum_{(x,t)\in\mathcal{T}_I} \left\| \hat{u}(0,x) - \sum_{i=1}^{N} \lambda_i \sin(2\pi f_i/L_x)x - \Psi_i \right\|_2^2 \\
& + \frac{w_{g_1}}{|\mathcal{T}_{g_1}|} \sum_{(x,t)\in\mathcal{T}_{g_1}} \left\| \frac{\partial^2(\hat{u}(t,x))}{\partial x \partial t} + \frac{\partial^2(\hat{u}^2(t,x)/2)}{\partial x^2} - (\eta_v/\pi)\frac{\partial^3(\hat{u}(t,x))}{\partial x^3} \right\|_2^2 + \frac{w_{g_2}}{|\mathcal{T}_{g_2}|} \sum_{(x,t)\in\mathcal{T}_{g_2}} \\
& \left\| \frac{\partial^2(\hat{u}(t,x))}{\partial t^2} + \frac{\partial^2}{\partial t \partial x}(\hat{u}^2(t,x)/2) - (\eta_v/\pi)\frac{\partial^2(\hat{u}(t,x))}{\partial x^2} \right\|_2^2 + \frac{w_p}{|\mathcal{T}_p|} \sum_{(x,t)\in\mathcal{T}_p} \left\| \hat{u}(t,x) - \hat{\hat{u}}(t,x)_p \right\|_2^2.
\end{aligned} \tag{15}$$

where $\hat{\hat{u}}_p$ is the pseudo points generated in the previous iterations. If there is supervised dataset available, then an extra loss term $\frac{w_d}{|\mathcal{T}_d|}\sum_{(x,t)\in\mathcal{T}_d} \|\hat{u}(t,x) - u(t,x)_d\|_2^2$ can be added to the overall loss function (15).
Analyzing the results obtained for the approximated solution $\hat{u}$ of the solution u to the

Burgers' equation using the gST-PINN algorithm, we obtain an L^2 relative error and MSE error in $\hat{u}$ respectively $7.732166 \times 10^{-2} (\sim< 10\%)$ and 5.421603×10^{-5}, which are exceptional when compared with the results of standard classical PINN, which demonstrates the corresponding values $4.051458 \times 10^{-1} (\sim> 10\%)$ and 1.488497×10^{-3}. Figure (4) shows a detailed performance analysis of the gST-PINN model in solving Burgers' equation in terms of predicted solution as a two variable function of space and time variables defined on a rectangular domain, i.e. $\hat{u}\ vs\ (t, x)$ and the absolute error with respect to the true value of the solution from the input labeled data, i.e., $|\hat{u} - u|(x, t)$. The figures not only clearly depict that the solutions estimated by the gST-PINN model are profoundly accurate, but also demonstrates its superiority compared to the standard PINN model in terms of prediction accuracy. Figure (5) exhibits the $\hat{u}vsx$ plots at three distinct fixed time values $t = 0.3,\ 0.6,\ 0.9$. At $t = 0.3$, both gST-PINN and PINN provides comparable predictions, nevertheless gST-PINN provides more accurate predictions closely matching the sinusoidal form of solutions. But at higher time values $t = 0.6$ and 0.9, the pre-eminence of gST-PINN becomes conspicuous. While gST-PINN preserves its estimation accuracy over higher time points, it can be seen that the solutions predicted by PINN start to deviate significantly, especially near the extrema as we move to higher time points. For a more precise analysis of accuracy, an inspection of $\hat{u}(t, x)$ from the plots in figure (5) shows that $u(0.3,\ 0.4) \approx 00.1489$, $\hat{u}_{gST-PINN}(0.3,\ 0.4) \approx 0.1501$, $\hat{u}_{PINN}(0.3,\ 0.4) \approx 0.1802$. Similar observations can also be made for $t = 0.6$ and $t = 0.9$.

Both PINN and gST-PINN models implemented fully connected neural networks, structured with 4 neurons of 32 neurons each and the network structure was selected by Bayesian optimization. The network training was done over a spatio-temporal domain spanning $[0, 1] \times [0, 2]$ discretized into a grid of $N_x \times N_t = 512 \times 201 = (102,912$ total points). The number of initial, boundary and intra-domain data points used were 512, 402 and respectively. Adam optimizer was used to train both the networks over 20,000 iterations, configured with a learning rate of 10^{-3} and the tanh activation function.

4.2 Diffusion-Reaction Equation

The diffusion-reaction equation in one spatial dimension (also known as Kolmogorov–Petrovsky–Piskunov equation) [27, 28] in plane geometry can be expressed as given in equation (16):

$$\frac{\partial}{\partial t}(u(t, x)) = \xi_v \frac{\partial^2}{\partial x^2}(u(t, x)) + \rho_m A(u(t, x)) \tag{16}$$

where u represents the solute concentration. The values of the various parameters in equation (16) have been considered by us as
ρ_m (*mass density*) $\rightarrow 1.5$
ξ_v (*diffusion coefficient*) $\rightarrow 0.5$.
In this article, we consider a specific diffusion-reaction equation, the Fisher's equation

by setting

$$A(u(t,x)) = (u(t,x))((1-u(t,x)). \tag{17}$$

Similar to the Burger's equation, we consider a periodic boundary condition ($\tilde{B}_2(t)$) (12) and the initial condition ($\tilde{I}_2(x)$) (12) as a superposition of sinusoidal functions (figure (2)).

Computing the gradients with respect to the x and t variables, we get the corresponding equations

$$\frac{\partial^2}{\partial x \partial t}(u(t,x)) = \xi_v \frac{\partial^3}{\partial x^3}(u(t,x)) + \rho_m\Big(\frac{\partial}{\partial x}(u(t,x))\Big)(1-2u(t,x)) \tag{18}$$

and

$$\frac{\partial^2}{\partial t^2}(u(t,x)) = \xi_v \frac{\partial^3}{\partial t \partial x^2}(u(t,x)) + \rho_m\Big(\frac{\partial}{\partial t}(u(t,x))\Big)(1-2u(t,x)). \tag{19}$$

Subsequently, we model the loss function for training the network corresponding to solving the diffusion-reaction equation making use of equations (18) and (19) as

$$\begin{aligned}
\mathcal{L}_{diffusion-reaction} = {} & \frac{w_G}{|\mathcal{T}_G|}\sum_{(x,t)\in\mathcal{T}_G}\left\|\frac{\partial(\hat{u}(t,x))}{\partial t} - \xi_v\frac{\partial^2(\hat{u}(t,x))}{\partial x^2} - \rho_m\hat{u}(t,x)(1-\hat{u}(t,x))\right\|_2^2 \\
+\frac{w_D}{|\mathcal{T}_D|}\sum_{(x,t)\in\mathcal{T}_D}\|\hat{u}(0,t)-\hat{u}(1,t)\|_2^2 + & \frac{w_I}{|\mathcal{T}_I|}\sum_{(x,t)\in\mathcal{T}_I}\left\|\hat{u}(0,x) - \sum_{i=1}^{N}\chi_i\sin(2\pi k_i/L_x)x - \Phi_i\right\|_2^2 \\
+ \frac{w_{g_1}}{|\mathcal{T}_{g_1}|}\sum_{(x,t)\in\mathcal{T}_{g_1}}\left\|\frac{\partial^2(\hat{u}(t,x))}{\partial x\partial t} - \xi_v\frac{\partial^3(\hat{u}(t,x))}{\partial x^3} - \rho_m\Big(\frac{\partial(\hat{u}(t,x))}{\partial x}\Big)(1-2\hat{u}(t,x))\right\|_2^2 \\
+ \frac{w_{g_2}}{|\mathcal{T}_{g_2}|}\sum_{(x,t)\in\mathcal{T}_{g_2}}\left\|\frac{\partial^2(\hat{u}(t,x))}{\partial t^2} - \xi_v\frac{\partial^3(\hat{u}(t,x))}{\partial t\partial x^2} - \rho_m\Big(\frac{\partial(\hat{u}(t,x))}{\partial t}\Big)(1-2\hat{u}(t,x))\right\|_2^2 \\
\frac{w_p}{|\mathcal{T}_p|}\sum_{(x,t)\in\mathcal{T}_p}\left\|\hat{u}(t,x) - \hat{\hat{u}}(t,x)_p\right\|_2^2.
\end{aligned} \tag{20}$$

Investigating the results obtained for the estimation of solution $\hat{u}$ of the diffusion reaction equation using gST-PINN model, the L^2 relative error and MSE errors in $\hat{u}$ are correspondingly $6.110305\times10^{-2}(<6.2\%)$ and 1.067104×10^{-3} while the respective values obtained using standar PINN algorithm are $5.218521\times10^{-1}(\sim>10\%)$ and 7.783521, thereby gST-PINN displaying better accuracy. From figure (6), it is clearly visible that the diffusion reaction equation solution estimates are much closer to the actual u values over the entire rectangular (t,x) domain, thus reducing the absolute error over the domain with respect to the actual solution, i.e., $|\hat{u}_{gST-PINN}(t,x)-u(t,x)|\approx\epsilon_1$, $|\hat{u}_{PINN}(t,x)-u(t,x)|\approx\epsilon_2 \Rightarrow \epsilon_1 \lesssim \epsilon_2 \;\; \forall(t,x)\in\Omega_{diffusion-reaction}$.

4.3 Diffusion-Sorption Equation

The one dimensional diffusion sorption equation can be represented by equation (21) as

$$\frac{\partial}{\partial t}(u(t,x)) = \frac{C_d}{R(u(t,x))}\frac{\partial^2}{\partial x^2}(u(t,x)) \qquad x \in (0,1),\ t \in (0,500] \tag{21}$$

where $R(u(x,t))$ is the retardation factor which is a non-linear function of $u(t,x)$:

$$R(u(t,x)) = 1 + \frac{1-\psi}{\psi}\rho_s k_f n_f u^{n_f-1}. \tag{22}$$

The values of the various parameters in equations (21) and (22) has been considered to be
ψ (*porosity* of the medium for which the PDE is defined) $\rightarrow 0.31$
k_f (*Freundlich's parameter*) $\rightarrow 3.5 \times 10^{-4}$
n_f (*Freundlich's exponent*) $\rightarrow 0.875$
ρ_s (*bulk density*) $\rightarrow 2875$
C_s (*diffusion coefficient*) $\rightarrow 4.5 \times 10^{-4}$
At the lower boundary ($x = 0$), we have the uniform Dirichlet boundary condition given by

$$u(t,0) = 1 \approx \tilde{B}_3(t) \tag{23}$$

and at the higher boundary, we have a Robin boundary condition:

$$u(t,1) = C_s\frac{\partial u}{\partial x}(t,1). \tag{24}$$

The initial condition is formulated using a uniform random distribution:

$$u(0,x) = 0.16395 \quad \forall x \in (0,1) \approx \tilde{I}_3(x). \tag{25}$$

The plot of initial and boundary conditions are displayed in figure (2).
The derivatives (or gradients) with respect to the space and time variables can be computed respectively as

$$\frac{\partial^2}{\partial x \partial t}u(t,x) = \frac{C_d}{R(u(t,x))}\Big\{\frac{\partial^3}{\partial x^3}u(t,x) - \Big(\frac{1}{R}\Big)\Big(\frac{\partial}{\partial x}u(t,x)\Big)\Big(\frac{\partial^2}{\partial x^2}u(t,x)\Big)\Big(\frac{1-\psi}{\psi}\Big) (\rho_s n_f k_f (n_f - 1))u^{n_f-2}\Big\} \tag{26}$$

and

$$\frac{\partial^2}{\partial t^2}u(t,x) = \frac{C_d}{R(u(t,x))}\Big\{\frac{\partial^3}{\partial t \partial x^2}u(t,x) - \Big(\frac{1}{R}\Big)\Big(\frac{\partial}{\partial t}u(t,x)\Big)\Big(\frac{\partial^2}{\partial x^2}u(t,x)\Big)\Big(\frac{1-\psi}{\psi}\Big) (\rho_s n_f k_f (n_f - 1))u^{n_f-2}\Big\}. \tag{27}$$

Hence, from equations (21), (23), (24), (25), (26) and (27), the loss function for training the network can be constructed as

$$\mathcal{L}_{diffusion-sorption} = \frac{w_G}{|\mathcal{T}_G|}\sum_{(x,t)\in\mathcal{T}_G}\left\|\frac{\partial(\hat{u}(t,x))}{\partial t} - \frac{C_d}{R(\hat{u}(t,x))}\frac{\partial^2(\hat{u}(t,x))}{\partial x^2}\right\|_2^2$$
$$+ \frac{w_{D_1}}{|\mathcal{T}_{D_1}|}\sum_{(x,t)\in\mathcal{T}_{D_1}}\left\|\hat{u}(t,1) - C_d\frac{\partial\hat{u}}{\partial x}(t,1)\right\|_2^2 + \frac{w_{D_2}}{|\mathcal{T}_{D_2}|}\sum_{(x,t)\in\mathcal{T}_{D_2}}\|\hat{u}(t,0)-1\|_2^2 + \frac{w_I}{|\mathcal{T}_I|}$$
$$\sum_{(x,t)\in\mathcal{T}_I}\|\hat{u}(0,x) - 0.16395\|_2^2 + \frac{w_{g_1}}{|\mathcal{T}_{g_1}|}\sum_{(x,t)\in\mathcal{T}_{g_1}}\left\|\frac{\partial^2}{\partial x\partial t}\hat{u}(t,x) - \frac{C_d}{R(\hat{u}(t,x))}\Big\{\frac{\partial^3\hat{u}(t,x)}{\partial x^3}\right.$$
$$\left. - \Big(\frac{1}{R}\Big)\Big(\frac{\partial\hat{u}(t,x)}{\partial x}\Big)\Big(\frac{\partial^2\hat{u}(t,x)}{\partial x^2}\Big)\Big(\frac{1-\psi}{\psi}\Big)(\rho_s n_f k_f(n_f-1))\hat{u}^{n_f-2}\Big\}\right\|_2^2 + \frac{w_{g_2}}{|\mathcal{T}_{g_2}|}\sum_{(x,t)\in\mathcal{T}_{g_2}}$$
$$\left\|\frac{\partial^2}{\partial t^2}\hat{u}(t,x) - \frac{C_d}{R(u(t,x))}\Big\{\frac{\partial^3}{\partial t\partial x^2}\hat{u}(t,x) - \Big(\frac{1}{R}\Big)\Big(\frac{\partial}{\partial t}\hat{u}(t,x)\Big)\Big(\frac{\partial^2}{\partial x^2}\hat{u}(t,x)\Big)\Big(\frac{1-\psi}{\psi}\Big)\right.$$
$$\left.(\rho_s n_f k_f(n_f-1))\hat{u}^{n_f-2}\Big\}\right\|_2^2 + \frac{w_p}{|\mathcal{T}_p|}\sum_{(x,t)\in\mathcal{T}_p}\left\|\hat{u}(t,x) - \hat{\hat{u}}(t,x)_p\right\|_2^2. \quad (28)$$

When considering the diffusion-sorption equation (21), for the predicted solution $\hat{u}$, the gST-PINN model yields L^2 relative error and MSE errors respectively $1.831367 \times 10^{-2}(\sim< 2\%)$ and 5.348408×10^{-5}. Alike for the case of Burgers' equation, the gST-PINN model shows superior performance also for diffusion-sorption equation compared to PINNs which yields the L^2 relative errors and MSE errors of $2.330058\times10^{-2}(\sim> 2\%)$ and 8.657794×10^{-5} respectively for the predicted solution $\hat{u}$. Further performance analysis of the gST-PINN and its comparison with the standard PINN model can be examined from the figures (7) and (8), which respectively are the $\hat{u}$ vs (x,t) information across the 2-D rectangular domain and prediction of the form of solution as a function of x at three distinct fixed times $t = 150,\ 250,\ 350$, i.e., $\hat{u}(x,150),\ \hat{u}(x,250),\ \hat{u}(x,350)$.

4.4 Comparison with other scientific machine learning models

For portraying the dominance of the novel gST-PINN model over other existing scientific machine learning models for solving non-linear partial differential equations, we provide a performance comparison table which depicts the performance of various state-of-the-art models in terms of L_2 error and MSE error in the solution u for solving Burgers' and Diffusion Reaction equations. For our analysis, we consider the models PINN (Physics Informed Neural Network), ST-PINN (Self Training Physics Informed Neural Network) [29] and our newly proposed gST-PINN (Gradient Enhanced Self Training Physics Informed Neural Network). For both the equations, in accordance with the trivial expectations, augmentation of a few labeled data to the input enhances the model efficiency for all the models. For the Burgers' equation, PINN with labeled dataset gives an L_2 and MSE error of 4.051458×10^{-1} and 1.488497×10^{-3} respectively. ST-PINN without labeled data also shows approximately similar results with negligible

improvement with the values 4.056303×10^{-1} and 1.492060×10^{-3} respectively. ST-PINN with the addition of a few labeled dataset in the input shows substantial improvement in performance metrics with corresponding L_2 and MSE erros in u 8.154244×10^{-2} and 6.029660×10^{-5}. Now, analysing the figures of the gST-PINN model, it exhibits profound escalation in performance when compared with the other two models. gST-PINN without labeled data shows better results than both PINN and ST-PINN without labeled data with the value of L_2 and MSE errors in u equal to 3.979561×10^{-1} and 1.436137×10^{-3}. gST-PINN with labeled data depicts the best accuracy among all the models considered with comparatively exceptional figures of 7.732166×10^{-2} and 5.421603×10^{-5}. For the diffusion reaction equation also, the trend is similar to Burgers' equation as depicted by table (9). PINN with the inclusion of labeled data incorporates L_2 and MSE errors in u respectively 5.218521×10^{-1} and 7.783521×10^{-2} where as ST-PINN model with and without labeled data shows corresponding errors 6.194710×10^{-2}, 1.096789×10^{-3}, 5.236130×10^{-1} and 7.836138×10^{-2}. The gST-PINN model without labeled data shows L_2 and MSE errors in u 5.236974×10^{-1} and 7.838666×10^{-2} while the analogous values for gST-PINN model inclusive of labeled data are 6.110305×10^{-2} and 1.067104×10^{-3}.
All these above mentioned statistics indicate the supremacy of gST-PINN model for approximating the solution of highly non linear PDEs, in terms of both accuracy and efficiency in comparison with other models. It is noteworthy that Some of the models like the gPINN [24], gw-PINN [30], PINN without labeled data etc. have not been considered for the comparisons as these models exhibit substantially low results with respect to the models taken into consideration for the analysis.

5 Conclusion and future outlook

In the current research article, we have proposed a novel gradient enhanced self training physics informed neural network (gST-PINN) model, which is based on the semi-supervised learning paradigm. The core functioning of the model is based on our formulated gradient enhanced pseudo labeling algorithm which utilizes the information on the PDE residual and its derivative with respect to each independent variable to generate supervised data from unsupervised collocation points (gradient enhanced pseudo labeling) upon satisfying certain threshold filtration criteria. Furthermore, the algorithm is characterized by two crucial hyperparameters, the pseudo label selection threshold r and the psuedo label candidate rate q. We have utilized the gST-PINN model to solve some of the most important non-linear PDEs including the Burgers' equation, diffusion-reaction equation and diffusion-sorption equation to demonstrate the superiority of gST-PINN model compared to the classical PINN in terms of accuracy in estimating the solution. For respectively the Burgers' equation, diffusion-reaction equation and diffusion-equation, the gST-PINN model achieves a MSE error in the approximated solution $\hat{u}$ equal to 5.421603×10^{-5}, 1.067104×10^{-3} and 5.348408×10^{-5} compared to the figures 1.488497×10^{-3}, 7.783521×10^{-2} and 8.657794×10^{-5} achieved by the classical PINN model. Subsequently, it has also been demonstrated that the performance of the gST-PINN model without the inclusion of the labeled input data (with 1.436137×10^{-3} and 7.838666×10^{-2} being the MSE errors in $\hat{u}$ for Burgers'

equation and diffusion reaction equation respectively) is comparable to that of the performance portrayed by the classical PINN model assisted with inclusion of supervised input data. Furthermore, we also show the superiority of our proposed gST-PINN model with the inclusion of a small amount of supervised input data over the state-of-the-art models including the classical PINN, ST-PINN and gPINN, all considered with the provision of labeled input data.
One of the primary scopes of for future research as an extension to this research work include the modification of the gST-PINN model to address the geometry and topology of the PDE domain in order to solve high dimentional PDEs defined on non-trivial domains (such as moving boundaries) and Riemannian manifolds. Another noteworthy scope for future research is the enhancement of the gST-PINN model by augmenting it with the ability to solve fractional differential equations, as currently the number of scientific machine learning models specifically designed to exhibit high accuracy in solving fractional differential equations is scarce.

References

[1] 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. Journal of Computational physics **378**, 686–707 (2019)

[2] Cai, S., Mao, Z., Wang, Z., Yin, M., Karniadakis, G.E.: Physics-informed neural networks (pinns) for fluid mechanics: A review. Acta Mechanica Sinica **37**(12), 1727–1738 (2021)

[3] Toscano, J.D., Oommen, V., Varghese, A.J., Zou, Z., Ahmadi Daryakenari, N., Wu, C., Karniadakis, G.E.: From pinns to pikans: Recent advances in physics-informed machine learning. Machine Learning for Computational Science and Engineering **1**(1), 1–43 (2025)

[4] Wang, S., Sankaran, S., Wang, H., Perdikaris, P.: An expert's guide to training physics-informed neural networks. arXiv preprint arXiv:2308.08468 (2023)

[5] Karniadakis, G.E., Kevrekidis, I.G., Lu, L., Perdikaris, P., Wang, S., Yang, L.: Physics-informed machine learning. Nature Reviews Physics **3**(6), 422–440 (2021)

[6] Meng, C., Griesemer, S., Cao, D., Seo, S., Liu, Y.: When physics meets machine learning: A survey of physics-informed machine learning. Machine Learning for Computational Science and Engineering **1**(1), 20 (2025)

[7] Farea, A., Yli-Harja, O., Emmert-Streib, F.: Understanding physics-informed neural networks: Techniques, applications, trends, and challenges. AI **5**(3), 1534–1557 (2024)

[8] Klapa Antonion, X.W., Raissi, M., Joshie, L.: Machine learning through physics–informed neural networks: Progress and challenges. Academic Journal of Science

and Technology **9**(1), 2024 (2024)

[9] Changdar, S., Bhaumik, B., Sadhukhan, N., Pandey, S., Mukhopadhyay, S., De, S., Bakalis, S.: Integrating symbolic regression with physics-informed neural networks for simulating nonlinear wave dynamics in arterial blood flow. Physics of Fluids **36**(12) (2024)

[10] Amini, M.-R., Feofanov, V., Pauletto, L., Hadjadj, L., Devijver, E., Maximov, Y.: Self-training: A survey. Neurocomputing **616**, 128904 (2025)

[11] Triguero, I., García, S., Herrera, F.: Self-labeled techniques for semi-supervised learning: taxonomy, software and empirical study. Knowledge and Information systems **42**(2), 245–284 (2015)

[12] Zou, Y., Yu, Z., Liu, X., Kumar, B., Wang, J.: Confidence regularized self-training. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 5982–5991 (2019)

[13] Hady, M.F.A., Schwenker, F.: Semi-supervised learning. Handbook on neural information processing, 215–239 (2013)

[14] Van Engelen, J.E., Hoos, H.H.: A survey on semi-supervised learning. Machine learning **109**(2), 373–440 (2020)

[15] Yang, L., Meng, X., Karniadakis, G.E.: B-pinns: Bayesian physics-informed neural networks for forward and inverse pde problems with noisy data. Journal of Computational Physics **425**, 109913 (2021)

[16] 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. Computer Methods in Applied Mechanics and Engineering **365**, 113028 (2020)

[17] Jagtap, A.D., Karniadakis, G.E.: Extended physics-informed neural networks (xpinns): A generalized space-time domain decomposition based deep learning framework for nonlinear partial differential equations. Communications in Computational Physics **28**(5) (2020)

[18] Duan, S., Wu, W., Hu, P., Ren, Z., Peng, D., Sun, Y.: Copinn: Cognitive physics-informed neural networks. In: Forty-second International Conference on Machine Learning

[19] Zhao, C., Zhang, F., Lou, W., Wang, X., Yang, J.: A comprehensive review of advances in physics-informed neural networks and their applications in complex fluid dynamics. Physics of Fluids **36**(10) (2024)

[20] Choi, S., Jung, I., Kim, H., Na, J., Lee, J.M.: Physics-informed deep learning

for data-driven solutions of computational fluid dynamics. Korean Journal of Chemical Engineering **39**(3), 515–528 (2022)

[21] Stiasny, J., Zhang, B., Chatzivasileiadis, S.: Pinnsim: A simulator for power system dynamics based on physics-informed neural networks. Electric Power Systems Research **235**, 110796 (2024)

[22] Nuugulu, S.M., Patidar, K.C., Tarla, D.T.: A physics informed neural network approach for solving time fractional black-scholes partial differential equations. Optimization and Engineering, 1–30 (2024)

[23] Bai, Y., Chaolu, T., Bilige, S.: The application of improved physics-informed neural network (ipinn) method in finance. Nonlinear Dynamics **107**(4), 3655–3667 (2022)

[24] Yu, J., Lu, L., Meng, X., Karniadakis, G.E.: Gradient-enhanced physics-informed neural networks for forward and inverse pde problems. Computer Methods in Applied Mechanics and Engineering **393**, 114823 (2022)

[25] Ortiz Ortiz, R.D., Martínez Núñez, O., Marín Ramírez, A.M.: Solving viscous burgers' equation: hybrid approach combining boundary layer theory and physics-informed neural networks. Mathematics **12**(21), 3430 (2024)

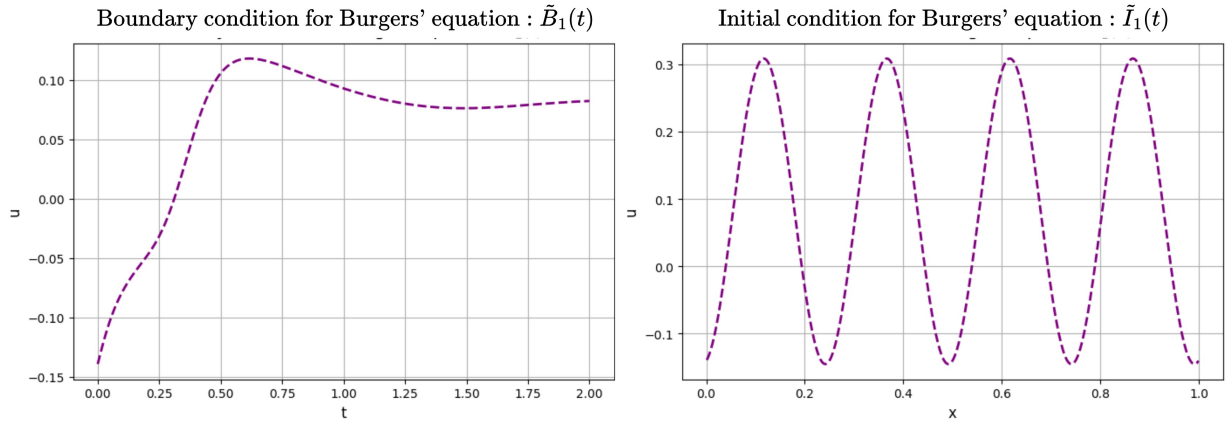
[26] Li, C.: Assessing the performance of pinn and cnn approaches in solving the 1d burgers' equation with deep learning architectures. In: Proceedings of the 2023 4th International Conference on Machine Learning and Computer Application, pp. 888–892 (2023)

[27] Zhang, W.-g., Hu, X.-k., Ling, X.-q., Li, W.-x.: Approximate analytical solution of the generalized kolmogorov-petrovsky-piskunov equation with cubic nonlinearity. Acta Mathematicae Applicatae Sinica, English Series **39**(2), 424–449 (2023)

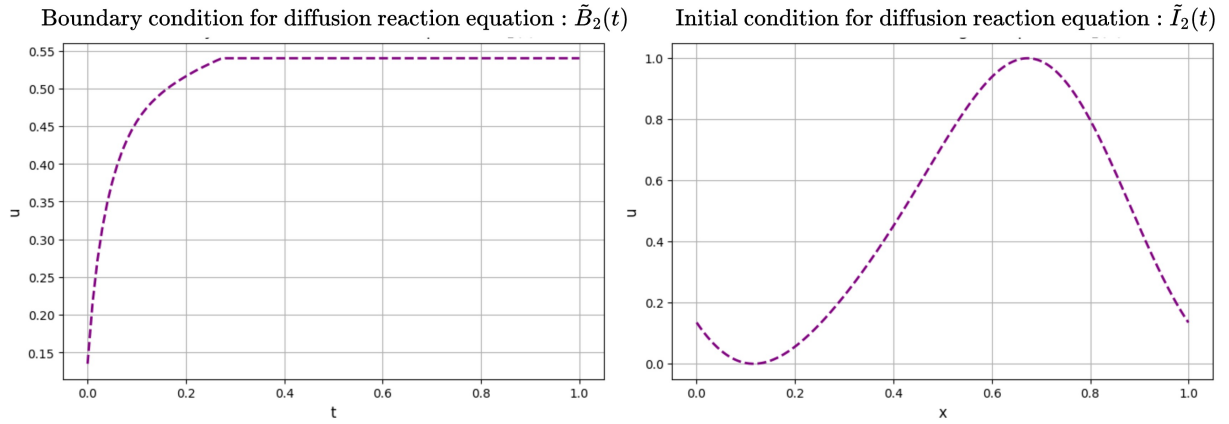
[28] Wongsaijai, B., Aydemir, T., Ak, T., Dhawan, S.: Analytical and numerical techniques for initial-boundary value problems of kolmogorov–petrovsky–piskunov equation. Numerical Methods for Partial Differential Equations **40**(1), 22693 (2024)

[29] Yan, J., Chen, X., Wang, Z., Zhoui, E., Liu, J.: St-pinn: A self-training physics-informed neural network for partial differential equations. In: 2023 International Joint Conference on Neural Networks (IJCNN), pp. 1–8 (2023). IEEE

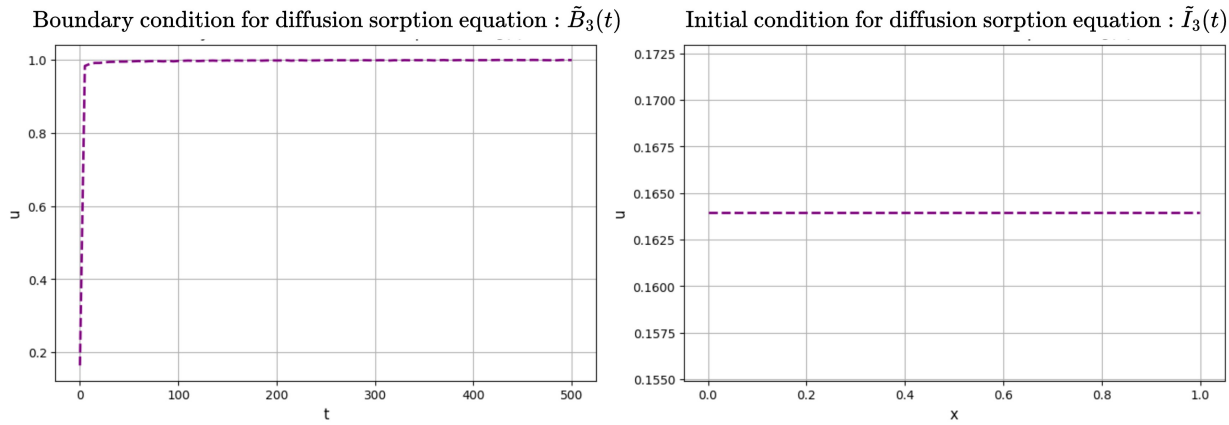
[30] Xiong, F., Liu, L., Liu, S., Wang, H., Yong, H.: Gradient-weighted physics-informed neural networks for one-dimensional euler equation. Authorea Preprints (2023)



(a)



(b)



(c)

Fig. 2: Functional form of initial and boundary conditions of **(a)**: Burgers' equation **(section(4.1))**, **(b)**: diffusion reaction equation **(section (4.2))**, **(c)**: diffusion sorption equation **(section(4.3))**.

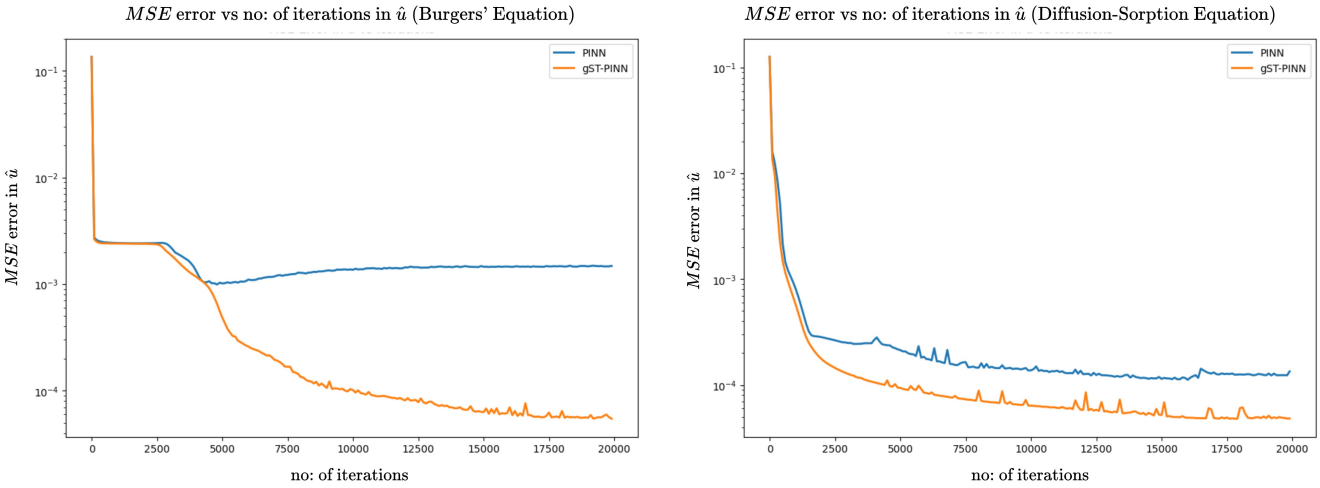


Fig. 3: Plots of MSE error in $\hat{u}$ vs number of iterations in gST-PINN and PINN models for the approximation of solution to the **Burgers' equation** (4.1) and **diffusion-sorption equation** (4.3). It is conspicuous that the MSE errors in PINN model converge very quickly (for example, approximately after 7500 iterations in case of Burgers' equation, eventually achieving the $\mathcal{O}(10^{-3})$ and 12500 iterations in case of diffusion-sorption equation, eventually achieving $\mathcal{O}(10^{-4})$), while the MSE errors in gST-PINN model continue to decrease till the last iteration.

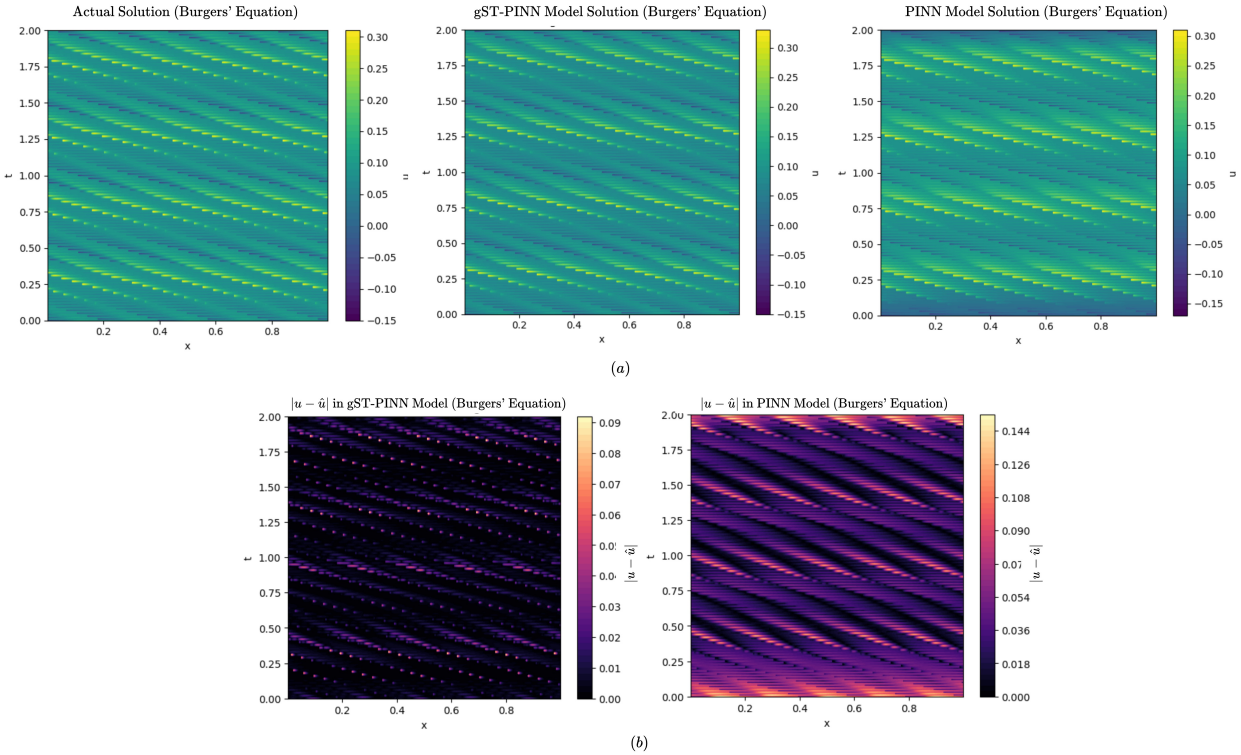


Fig. 4: **(Section (4.1), Burgers' equation) (a)**: Comparison of the actual solution u of Burgers' equation with its predicted solutions $\hat{u}$ using gST-PINN and PINN models. **(b)**: Comparison of the absolute error in u, i.e., $|u - \hat{u}|$ predicted by the gST-PINN and PINN model. From both **(a)** and **(b)**, it can be clearly observed that the prediction $\hat{u}$ of the solution u for Burgers' equation performed by the gST-PINN model is more accurate in comparison with the PINN model, i.e., $|u(x) - \hat{u}(x)|_{gST-PINN} \leq |u(x) - \hat{u}(x)|_{PINN}$, $\forall x \in \Omega_{Burgers'}$, where $\Omega_{Burgers'}$ denotes the domain on which the Burgers' equation is defined.

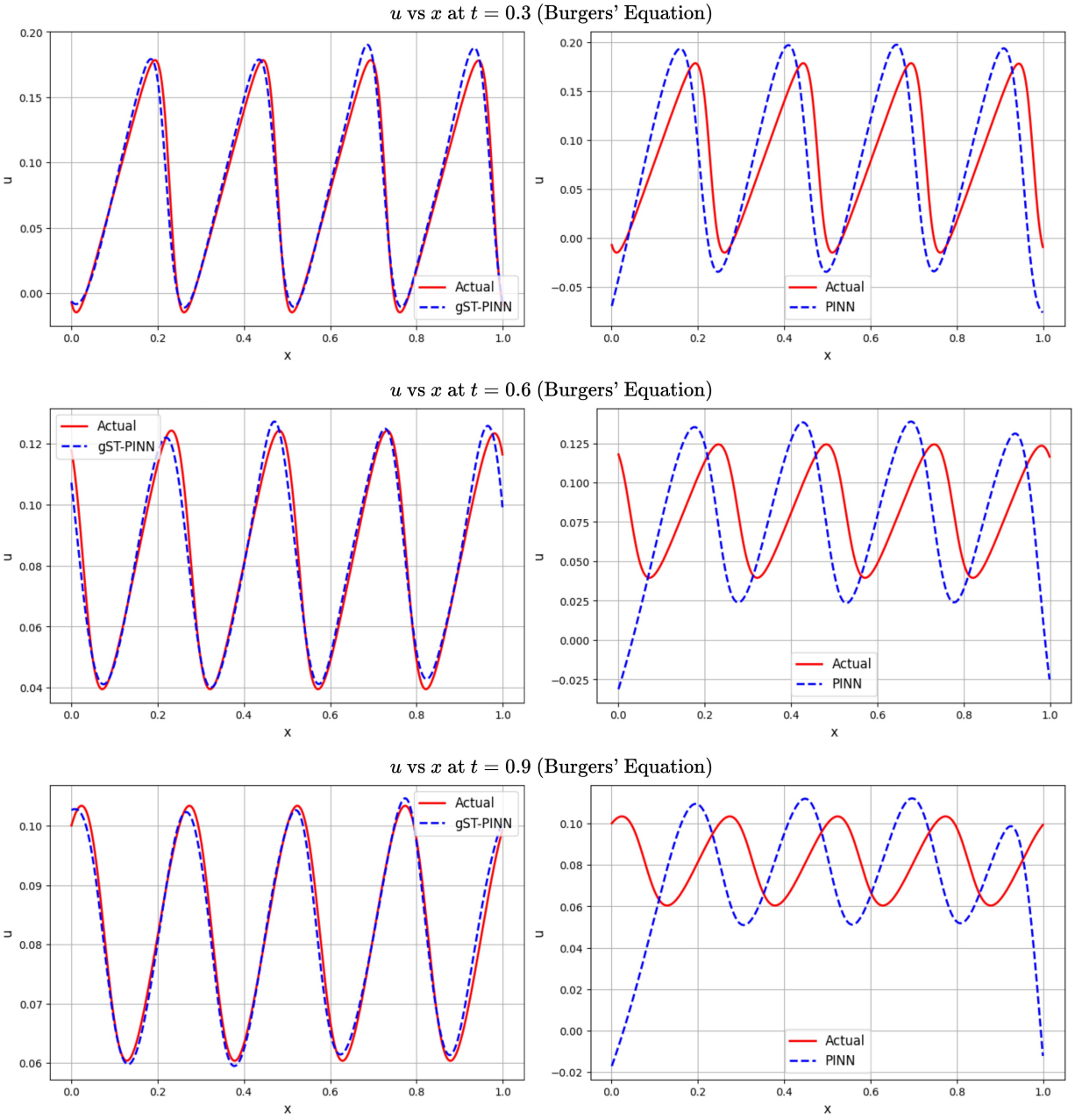


Fig. 5: **(Section (4.1), Burgers' equation)** Performance comparison analysis of gST-PINN model and the normal PINN model for solving the Burgers' equation. The analysis has been done by studying the "**u** vs **x**" plots at three different time instances, **t=0.3, 0.6, 0.9**. It has been examined from the plots that, initially **(t=0.3)**, both the models exhibit roughly identical accuracy, but at later intervals **(t=0.6, 0.9)**, the accuracy of the PINN model considerably deteriorates whereas the gST-PINN model succeeds in preserving the accuracy level exhibited initially, thereby proving to be more consistent compared to the PINN model.

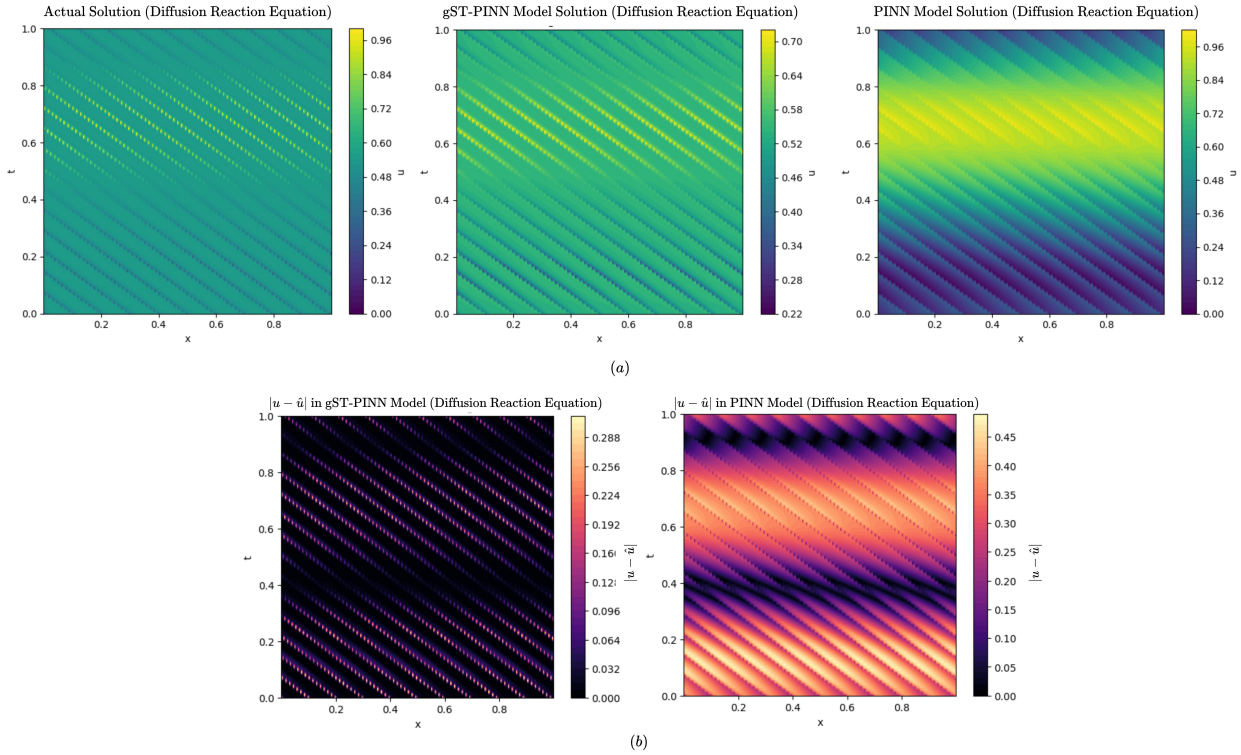


Fig. 6: **(Section (4.2), Diffusion-reaction equation) (a)**: Comparison of the actual solution u of diffusion reaction equation with its predicted solutions $\hat{u}$ using gST-PINN and PINN models. **(b)**: Comparison of the absolute error in u, i.e., $|u - \hat{u}|$ predicted by the gST-PINN and PINN model. From both **(a)** and **(b)**, it can be clearly summarized that the performance of the gST-PINN model in predicting the solution u of the diffusion reaction equation is substantially better in comparison with the usual PINN model.

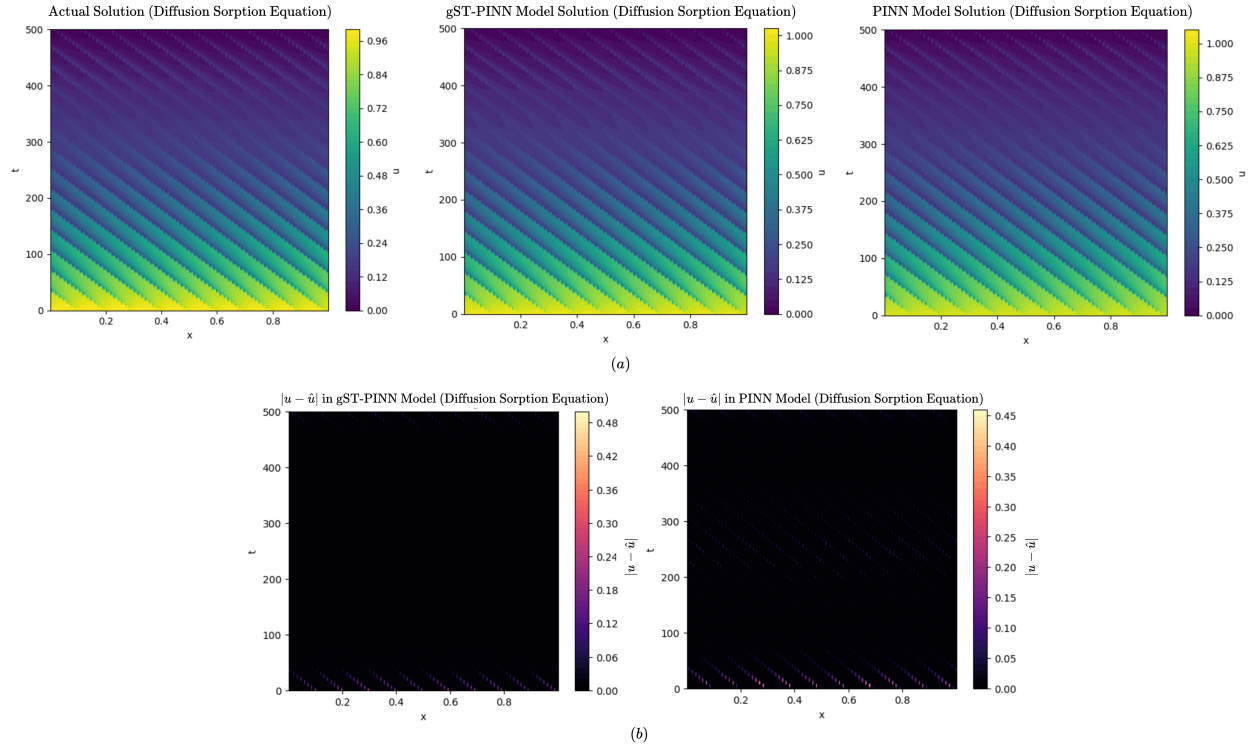


Fig. 7: **(Section (4.3), Diffusion sorption equation) (a)**: Comparison of the actual solution u of diffusion sorption equation with its predicted solutions $\hat{u}$ using gST-PINN and PINN models. **(b)**: Comparison of the absolute error in u, i.e., $|u - \hat{u}|$ predicted by the gST-PINN and PINN model. From both **(a)** and **(b)**, it can be clearly concluded that the gST-PINN model exhibits superior performance in comparison with the usual PINN model for the prediction $\hat{u}$ of the solution u of the diffusion sorption equation.

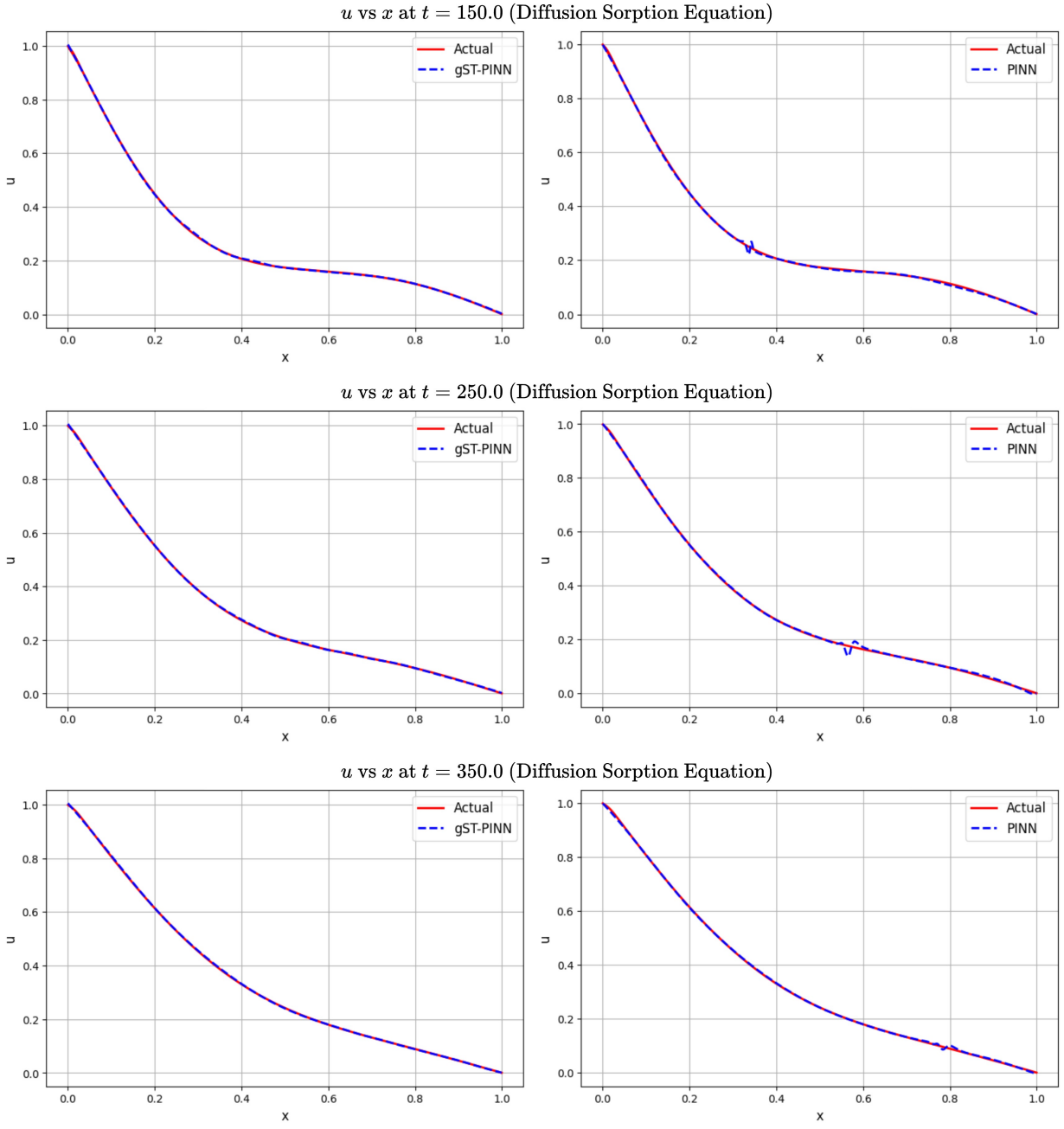


Fig. 8: **(Section (4.3), Diffusion sorption equation)** "**u** vs **x**" plots considered at three distinct time points **t=150, 250, 350**, of gST-PINN and PINN models for the estimation of solution to diffusion sorption equation. From the plots, it can be clearly examined that PINN portrays lesser accuracy in certain small ranges of x compared to other regions (for example, at $t = 150$, $t = 250$ and $t = 350$ respectively, PINN shows deteriorated performance when $x \in (0.3,\ 0.35)$, $x \in (0.55,\ 0.6)$ and $x \in (0.78,\ 0.81)$) whereas the gST-PINN model portrays exceptional accuracy over the entire range of $x \in [0, 1]$ over all the time instances under consideration.

Table 1: Burgers' Equation		
	L_2 error in u	MSE error in u
PINN with labelled data	4.051458×10^{-1}	1.488497×10^{-3}
ST-PINN with labelled data	8.154244×10^{-2}	6.029660×10^{-5}
ST-PINN without labelled data	4.056303×10^{-1}	1.492060×10^{-3}
gST-PINN with labelled data	7.732166×10^{-2}	5.421603×10^{-5}
gST-PINN without labelled data	3.979561×10^{-1}	1.436137×10^{-3}

Table 2: Diffusion Reaction Equation		
	L_2 error in u	MSE error in u
PINN with labelled data	5.218521×10^{-1}	7.783521×10^{-2}
ST-PINN with labelled data	6.194710×10^{-2}	1.096789×10^{-3}
ST-PINN without labelled data	5.236130×10^{-1}	7.836138×10^{-2}
gST-PINN with labelled data	6.110305×10^{-2}	1.067104×10^{-3}
gST-PINN without labelled data	5.236974×10^{-1}	7.838666×10^{-2}

Fig. 9: **Section (4.4)** Comparison of gST-PINN **(Table 1: Burgers' equation, Table 2: diffusion reaction equation)** with some state-of-the-art scientific machine learning models. The gST-PINN model with the incorporation of a few labelled data outperforms all the other models (PINN, gPINN, ST-PINN) in terms of accuracy in the prediction of solution to nonlinear PDEs.