International Journal of Engineering Works
https://www.ijew.io/
ISSN-p: 2521-2419
ISSN-e: 2409-2770
Vol. 13, Issue 02, PP. 13-21, February 2026
https://doi.org/10.5281/zenodo.18753514

Adaptive Gradient-Enhanced PINNs for Numerical Solution of Convection-Diffusion Equation

Javeed Ahmad
Department of Basic Sciences and Islamiat, University of Engineering and Technology, Peshawar, Pakistan.
javeedbashir@uetpeshawar.edu.pk
Received: 23 January, Revised:15 February, Accepted: 22 February

***Abstract*—**Physics-Informed Neural networks (PINNs) are mesh-free Deep Learning (DL) framework to solve Partial Differential equations (PDEs). This technique embeds physical laws directly into the training process, enabling the solution of forward and inverse problems governed by PDEs. Unlike traditional neural networks, PINNs incorporate the governing equations, initial conditions, and boundary conditions directly into the loss function. Automatic differentiation in PINNs avoids truncation errors and ensures high precision enforcing the governing equations. Despite their advantages, PINNs face several challenges. PINNs struggle to solve Convection-Diffusion Equations (CDEs), particularly at the region where the convection term dominated. To overcome this problem, an extended form of PINNs is discussed here. Adaptive Gradient-enhanced PINNs (AG-PINNs) are extensions of PINNs, where Residual-based Adaptive Refinement (RAR) and the derivatives of the governing equations are also enforced during training. However, adding gradient constraints leads to over-constraining the network, increased computational cost, and inefficient learning in smooth regions. This motivates RAR, which improves the solution accuracy while avoiding over-constraining the neural network in smooth regions. In this paper we discuss convection-diffusion equation with high Péclet number. As Pé increased the convection terms dominated so it become challenging for standard PINNs, to mitigate these challenges AG-PINNs is used. AG-PINNs is better than standard PINNs which is shown in this paper by comparing results of AG-PINNs with standard PINNs technique. This work is carried out through Python Jupiter Notebook in a deepXDE library.

***Keywords*—** Physics-informed neural networks, Deep learning, Automatic differentiation, Residual-based adaptive refinement, Covection dominated PDEs.

I. INTRODUCTION

Partial differential equations (PDEs) are typically used as an essential tool in mathematical modeling since they are often used to describe the evolution of physical phenomena by derivatives [1]. The PDEs are widely use to study and predict processes, such as heat conduction [2], wave propagation [3], fluids flow[4], electricity, and solid mechanics [5]. Besides, they are crucial in simulating chemical reactions and diffusion processes [6]. In addition to physical sciences, PDEs can be applied in population dynamics, epidemiology, as well as in the biological transport process, which demonstrates their relevance to various scientific and engineering fields [7].
Hence, both analytical and numerical methods can be used to solve the PDEs. PDEs that are relatively simple can be solved by Lie symmetry method, variational method, variational iteration method, eigenfunction expansion method, method of fundamental solution etc. However, these traditional techniques are seriously challenged in the application to PDEs which have complicated boundary conditions and nonlinearities. More importantly, they turn out to be ever more difficult to apply to high-dimensional PDEs, where computer complexity and efficiency are a significant constraint [8].

To overcome these limitations of traditional techniques, Raissi introduced an outstanding framework which is known as Physics-Informed Neural Networks (PINNs). This neural network is trained by embedding the physical laws of governing equation into the training process, since it learn directly from the physics of given problem [9].

The idea of neural network models was initially introduced by a neurophysiologist, Warren McCulloch and a mathematician, Walter Pitts, who studied how biological neurons actually act [10]. Later in 1944, Joseph Erlanger and Herbert Gasser made a systematic classification of neurons (group A, B, and C) and defined a correlation between conduction velocity of action potentials and the nerve fiber diameter [11]. Frank Rosenblatt, a psychologist, came up with the perceptron model in 1958. The perceptron answered some basic questions on the sense of physical world information, storage, and recognition, thus making a conceptual connection between biophysics and psychology [12]. Generally, neural networks came to prominence in 1969, with a seminal book by Marvin Minsky and Seymour Papert applying neural networks to computational geometry problems. Their contribution mainly emphasized the shortcomings of single-layer perceptrons, showing that such models are capable only linearly separable problems and fail execute the exclusive-OR (XOR) operation. They were aware that more powerful architectures would be possible with multi-layer perceptrons, and that the original feedforward perceptron described by Rosenblatt was a

three-layer architecture, howover, their analysis mostly focused on two-layer networks. This shortcoming made neural network studies slow down considerably in the next few years [13]. Essential concepts of learning presented in artificial neurons were introduced by Klopf in 1972 based on biological learning processes. The work of this research was further developed in 1974 when Paul Werbos introduced the backpropagation algorithm. Nevertheless, his work did not receive the importance that it deserved until 1986 when backpropagation became widely known and popular [14]. In the meantime, in 1975 Kunihiko Fukushima contributed noteworthy by coming up with a multilayer neural network that was trained using a two step learning sequence to recognize written characters. This was recorded in his famous and influential paper and formed significant groundwork on the subsequent developments of neural network architectures [15].

The 1980s were a renaissance period in the field of neural networks with a number of developments. Teuvo Kohonen proposed the concept of a self-organizing neural networks, commonly referred to as the Kohonen map [16]. In 1982, a historic article by John Hopkins of Caltech was published, in which he described a recurrent neural network as memory system based on pattern matching [17]. The work of Hopfield attracted the attention of many researchers of various fields. In the meantime, the backpropagation algorithm which had been introduced by Paul Werbos in 1974 came back into the limelight in 1986. In the same year, Rumelhart, Hanto, and Williams published landmark paper articles that showed how multilayer neural networks could be trained using error backpropagation, and that the algorithm was able to learn highly intricate internal representations [18]. Algorithms based on gradient descent quickly gain popularity in the reduction of errors in neural networks. In 1985, the American Institute of Physics launched the aritificial neural networks for computing initiative, that took a big step in formalizing the field. The greatest breakthrough was in 1987, when the first IEEE International Conference on Neural Networks was held, and the International Neural Network Society (INNS) was formed which further encouraged international cooperation and research in neural networks [19]. Over the next years the International Neural Network Society (INNS) had formed a number of journals with influence, such as Neural Networks (1988), Neural Computation (1989), and IEEE Transactions on Neural Networks (1990). In the year 1987, Carpenter and Grossberg came up with a neural architecture called ART1 a self-organizing system which mimics the way the brain works and which is especially created to detect binary patterns [20].

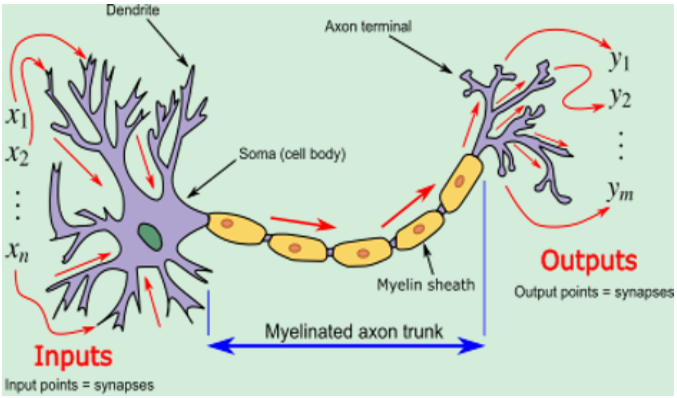
Nowadays, the field of neural networks has entered the era of deep learning, which is gaining popularity among scholars, practitioners, and the general audience. This increased interest is a sign of both curiosity about the commercial potential of the technology as well as its wider adoption in solving different real-world problems. Research is still being conducted to expand the field, such as the construction of special hardware and novel applications derived using the neural network theory. The developments are a major turning point in the history and influence of the neural network technology. The results of the research group headed by Juergen Schmidhuber between 2009 and 2012 brought noteworthy discoveries that gave the bases to current recurrent neural networks (RNNs) and deep feedforward neural networks [21]. As these developments continued to happen, various neural network models have also been created to tackle numerous issues in many areas. The basic example of these is the simple feedforward neural network, which will be described below.

- **Feedforward neural networks (FNNs)**

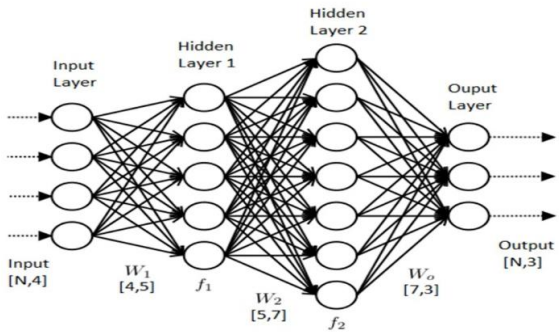
Feedforward Neural Network (FNN) is very simple neural network in which connections between neurons strictly acyclic. There are three major types of FNNs layers. The original data or features are first sent at the input layer and it is referred to as the input layer. The input data is actually fed through one or more hidden layers, where weighted operation perform the transformation. Lastly, the output layer generates network prediction that can be in form of a classification label or regression value or any other target output.

- **Architecture and components**

A feedforward neural network is organized in the form of a series of interconnected layers. The layers have a predetermined number of nodes also known as neurons or perceptrons, that are based on the human brain structure. These nodes have numerical values called activations, that indicate the state of the network. The values of the previous layer are calculated to obtain activations in each layer, which flow forward through the network to the output layer starting with the input layer.



Mathematically, if k neurons are in the layer it show k-dimensional vector. The neurons of one layer have connections to the neurons of the next layer and the connections are defined by weights. These are real-valued numbers denoting the strength of the connections. Each neuron in a given layer receives input from all neurons of preceeding layers, and in this way, the network is able to propagate and transform information through its layers. The inputs to each neuron are multiplied by the respective connection weights and the weighted result is then processed by an activation function to give the final output. The structure of FNNs shown in below diagram.



Forward Propogation in Neural Network

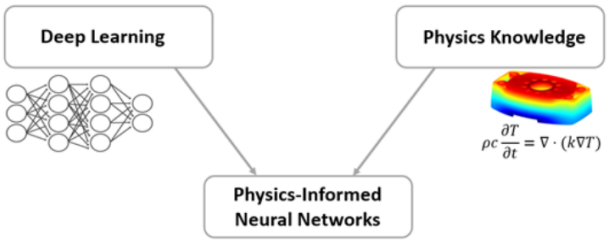
The process through which the information given as an input flows through the layers of a neural network to give an approximation final output is called forward propagation. It involves the first stage of the training process and it is then followed by backpropagation whereby the network weights are modified as per the difference between the output which is actually created and the real target values [22].

Backward propagation

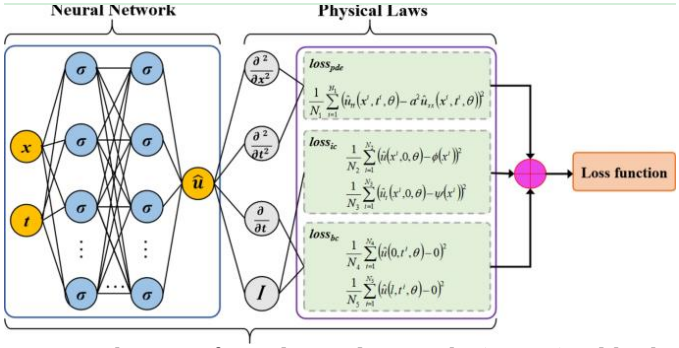
A neural network consists of several layers of neurons with each neuron being a weighted sum of its inputs and a activation function is applied to the neuron to give the neuron its output. In prediction, the difference between the real target and the network output is measured with the help of a loss function. This loss is minimized by backpropagation that then optimizes the network parameters by changing the weights [23].

II. Literature Review

Deep learning techniques have become an effective approach to the computations of PDEs with high accuracy and efficiency, especially because they are able to solve complex and high-dimensional PDEs. Raissi was the first to propose a new method called Physics-Informed Neural Networks (PINNs), which directly integrate the equations governing the system into the loss function of neural networks [24].



Machine learning as a division of artificial intelligence has developed and grown quite fast over the last years, which is facilitated by the presence of massive amounts of data and the growing strength of the modern computing resources. Such advancements have resulted in a revolutionary change in many areas, which have included image recognition, natural language processing, cognitive modeling, and genomics [25]. Despite the notable achievements, scientific and engineering issues are associated with systems whose data gathering processes are costly, difficult, or even impossible in large volumes. In this case, traditional machine learning systems which depend so much on massive data sets usually cannot give credible predictions or theoretical assurances [26]. In order to overcome this drawback, scholars have incorporated physical knowledge into learning models. Sevral physical systems physical systems that are governed by well-known laws such as conservation principles, symmetries, and empirically conformed rules. Such domain knowledge can be incorporated into learning algorithms, enabling models to produce physically consistent and practically meaningful solutions even when only limited data are available, this is base of PINNs. PINNs method is a type of neural network, where observational data are used along with the PDEs of the physical system. Such networks are automatically trained to calculate the necessary derivatives and they directly include physical constraints in the loss function. The key concept of this framework is to be consistent with physical laws but learn a model through data and this leads to both accurate and generalizable models, especially in data-scarce regimes.



Physics-Informed Neural Networks (PINNs) unlike the previous types of models, especially Gaussian process-based models do not involve the linearization and simplification of nonlinear terms, which is often required in other traditional methods. Such an advantage is due to the expressive power of deep neural networks as well as the availability of state-of-the-art machine learning systems like TensorFlow. The PINNs framework is used in the current study to get approximate solutions of the governing equations. The neural network loss is developed by adding the residuals of the partial differential equations together with error as caused by the initial and boundary conditions, which are all computed at randomly distributed collocation points. Recent deep learning developments have seen data-driven models solve complex mathematical, scientific, and engineering problems. Here, PINNs have already proved to be extremely promising in addressing both forward and inverse problems of PDEs through direct physical laws being encoded into the training process, with the learned solutions then able to meet the observational and physical aspects of the science [27].

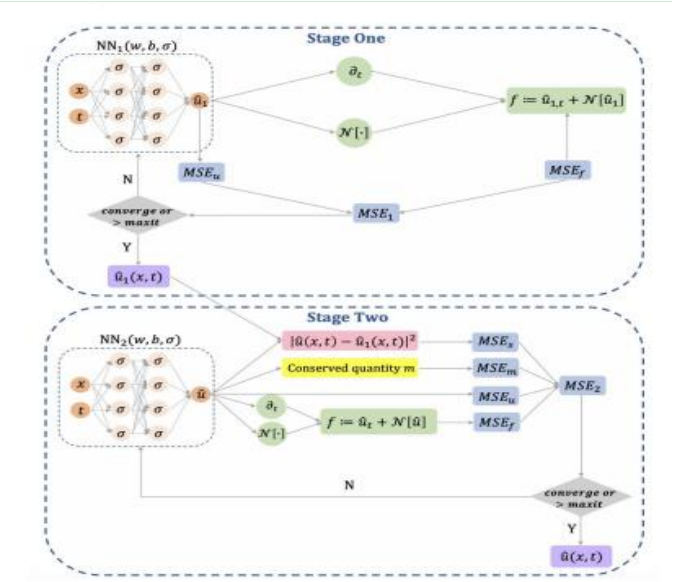
PINNs have a number of shortcomings in some cases, such as slow convergence and decreased accuracy, especially on high-dimensional PDEs. The difficulties of these standard PINNs are that they only impose the PDE residual at collocation points, which is in most cases inadequate to resolve the gradient

and fine-scale details of the solution. In order to overcome these restrictions, a gradient-enhanced method called gradient-enhanced Physics-Informed Neural Networks (gPINNs) was introduced. This approach uses gradient data on the governing PDEs in the training framework and allows the network to enhance local behaviors around sharp gradients and thus enhances convergence in addition to the accuracy of the solution [28]. Both forward and inverse PDEs can be solved precisely by using gPINNs. gPINNs loss function for both forward and inverse problems is composed of the following components.

- PDE residual loss
- Initial condition loss
- Boundary condition loss
- Gradient of residual loss

The gPINN formulation is specifically good at minimizing generalization error and, hence, improving predictive accuracy in areas with a thin layer of training data. Numerical experiments show that, gPINNs are better in forward and inverse problems compared to standard PINNs. As an illustration, in the case of Poisson equations, diffusion-reaction equations, PINNs, gPINNs display lower L2 errors than the traditional counterparts. In addition, gPINNs provide more precise derivative predictions that can be important to most physical and engineering applications [29].

Residual-based Adaptive Refinement (RAR) uses PINNs with training points that are added in areas where there are large PDE residuals and sampling strategies can be uniform or adaptive [30]. Two improved Neural Net-based techniques (rad and rar-d) coupled with gPINNs (RAR+gPINNs), give improved accuracy with reduced training points [31]. Other deep learning methods, including CNNs of spatial features and complex geometries, and RNNs of time-dependent PDEs have also been used. Moreover, two-stage PINNs are those that include the loss function with conserved quantities, which go a long way in enhancing prediction accuracy and generalization [32].



Recent improvements in PINN deal with steep boundary-layers of convection-dominated convection diffusion equations. Guan and Elman (2024) came up with TPINNs, through coordinate changes and baseline corrections to minimise oscillations, and enhance convergence in singularly perturbed problems [33]. As pointed out by Beguinet et al. (2022), sharp gradients, and low-regularity solutions pose a challenge to standard PINNs, and that the selection of network architecture, sampling, and loss function should be done with great greatness in convection-driving regimes to guarantee accuracy and generalization [34]. Lin and Chen generalized gPINNs to inverse problems by introducing Transfer Learning (TL-gPINNs) that adds gradient residuals to the loss function and also uses pre-trained neural networks as network initializations. The method enhances high robustness and allows the detection of PDEs with non-steady-state coefficients even in noisy conditions [35]. Fine-grained adaptive sampling and strong collocation techniques were applied to convection diffusion reaction PDEs by Khan et al. (2024) which have the same accuracy as classical solvers and can handle steep gradients and source terms [36]. Frerichs-Mihov, Henning and John (2024) proposed a new loss term to the convection-dominated PDEs, which penalises misalignment with layers and crosswind residuals, resulting in L_2 errors that are 17% and 5% lower than vanilla and hp-variational PINNs [37].

III. Methodology

In this section we discussed the methodology of AG-PINNs. To understand the methodology of AG-PINNs we also discuss standard PINNs and gPINNs methodology.

A. Physics-Informed Neural Networks (PINNs)

Generally time dependent problems with initial conditions and boundary condition is written as:

$$\begin{aligned} &\mathcal{R}[\mathrm{w}(\mathrm{x},\mathrm{y},\mathrm{t})] = 0, && (\mathrm{x},\mathrm{y},\mathrm{t}) \in \mathcal{D} \\ &\mathrm{w}(\mathrm{x},\mathrm{y},0) = 0, && (\text{initial condition}) \\ &\mathrm{w}(\mathrm{x},\mathrm{y},\mathrm{t})|_{\partial\mathcal{D}} = \mathrm{g}(\mathrm{x},\mathrm{y},\mathrm{t}) && (\text{Boundary conditions}) \end{aligned}$$

$\boldsymbol{\mathcal{R}}[.]$ is the non-linear differential operator and $\boldsymbol{\mathcal{D}}$ denotes spartiotemporal domains. $\hat{\mathrm{w}}(\mathrm{x},\mathrm{y},\mathrm{z},\theta)$ is the solution predicted by neural networks with trainable parameter θ. PINNs directly embed the governing partial differential equation into the loss function of the training process. Loss function of PINNs include:

- ***PDE residual loss*** :

$$\mathcal{L}_{PDE} = \frac{1}{N_r}\sum_{i=1}^{N_r} |\mathcal{R}[\hat{\mathrm{w}}(x_i, y_i, t_i, \theta)]|^2 ,$$

- ***Initial condition los***:

$$\mathcal{L}_{IC} = \frac{1}{N_0}\sum_{i=1}^{N_0} |\hat{\mathrm{w}}(x_i, y_i, 0) - w(x_i, y_i)|^2 ,$$

- ***Boundary condition loss***:

$$\mathcal{L}_{BC} = \frac{1}{N_b}\sum_{i=1}^{N_b}|\hat{w}(x_i, y_i, t_i, \theta) - g(x_i, y_i, t_i)|^2.$$

\
The total loss function of PINNs method is:

$$\mathcal{L}_{total} = \mathcal{L}_{PDE} + \mathcal{L}_{IC} + \mathcal{L}_{BC}$$

Neural networks is trained to minimize this total loss ($\mathcal{L}_{total}$) using Adam or L-BFGS or other optimizer. In case of exact solution given, accuracy is assessed by using L_2 relative error.

$$Relative\ L_2\ error = \frac{(\sum_{i=1}^{N}|w(x_i, y_i, t_i) - \hat{w}(x_i, y_i, t_i)|^2)^{\frac{1}{2}}}{(\sum_{i=1}^{N}|w(x_i, y_i, t_i)|^2)^{\frac{1}{2}}}$$

B. Gradienent-Enhanced Physics-Informed Neural Networks (gPINNs)

In standard PINNs the derivative of residual of PDE is not include, therefore it has poor convergence particularly at the regions of steep gradients. PINNs also face challenges for PDEs with complex dynamics. gPINNs improve the predicted solution by adding the gradient of residual of PDE in the loss function. So in gPINNs, the loss function minimize not only residual of PDE but also the gradient of residual. Hence the total loss in gPINNs is:

$$\mathcal{L}_{total} = \mathcal{L}_{PDE} + \lambda_x\mathcal{L}_{\partial_x\mathcal{R}} + \lambda_y\mathcal{L}_{\partial_y\mathcal{R}} + \lambda_t\mathcal{L}_{\partial_t\mathcal{R}} + \mathcal{L}_{IC} + \mathcal{L}_{BC}$$

Where,

$$\mathcal{L}_{\partial_x\mathcal{R}} = \frac{1}{N_r}\sum_{i=1}^{N_r}\left|\frac{\partial\mathcal{R}[\hat{w}(x_i, y_i, t_i)]}{\partial_x}\right|^2,$$

$$\mathcal{L}_{\partial_y\mathcal{R}} = \frac{1}{N_r}\sum_{i=1}^{N_r}\left|\frac{\partial\mathcal{R}[\hat{w}(x_i, y_i, t_i)]}{\partial_y}\right|^2,$$

$$\mathcal{L}_{\partial_t\mathcal{R}} = \frac{1}{N_r}\sum_{i=1}^{N_r}\left|\frac{\partial\mathcal{R}[\hat{w}(x_i, y_i, t_i)]}{\partial_t}\right|^2.$$

$\mathcal{L}_{total}$ is the total loss of gPINNs, $\mathcal{L}_{\partial_x\mathcal{R}}, \mathcal{L}_{\partial_y\mathcal{R}}\ and\ \mathcal{L}_{\partial_t\mathcal{R}}$ are the gradient loss of PDE with respect to x, y, and t dimensions respectively. $\lambda_x, \lambda_y, and\ \lambda_t$ are weights, which are defined by the user. gPINNs method is useful for convection dominated problems or stiff problems, where the standard PINNs struggle to capture the sharp gradients [38].

C. Adaptive Gradient-enhanced Physics-Informed Neural Networks (AG-PINNs)

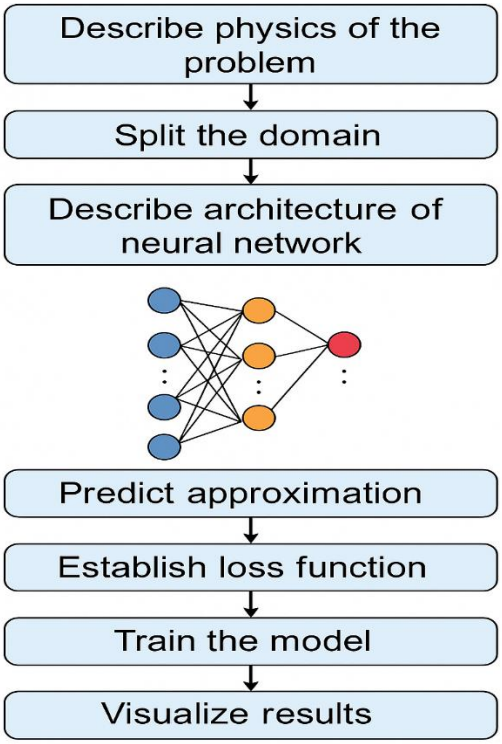
AG-PINNs is a fusion of gPINNs are RAR. The loss term improve the learning rate of the neural network while RAR control overfitting or underfitting of data. In this way AG-PINNs does't waste effort on easy regions which accelerate the convergence of solution. RAR also guide the training toward steep gradients (residual is large) which make AG-gPINNs accurate and stable.

- **Important steps of AG-PINNs**

1. Select an initial collocation points $\mathcal{T}$ and initialize a neural networks $\hat{w}(x_i, y_i, t_i)$.
2. Start training by minimizing the loss function $\mathcal{L}_{total}$

$$\mathcal{L}_{total} = \mathcal{L}_{PDE} + \lambda_x\mathcal{L}_{\partial_x\mathcal{R}} + \lambda_y\mathcal{L}_{\partial_y\mathcal{R}} + \lambda_t\mathcal{L}_{\partial_t\mathcal{R}} + \mathcal{L}_{IC} + \mathcal{L}_{BC}$$

3. Identify residual of PDE and its gradients over the dense set of candidates points.
4. Adaptively refine those points where the residual is large.
5. Retrain the neural networks by using the updated collocation points.
6. Repeat the training process and adaptive refinement until the residual become less than of equal to some threshold.



IV. MATERIAL AND DATASET

To evaluate the performance of our proposed AG-PINNs, the neural network consists of five hidden layers with 100 neurons per each hidden layer. The Swish activation function, which is defined as x.sigmoid(x), is used. The Adam optimizer is used to optimize model parameters which are then further optimized with L-BFGS (Limited-memory Broyden-Fletcher-Goldferb-Shanno) optimizer. The network is then trained with 10,000 collocation points in the domain and 2,000 points of the initial and boundary conditions. Preliminary training is done with 5,000 epochs, followed by Residual Adaptive Refinement

(RAR), that include five refinement iteration and each iteration include 3000 epochs. This deep learning technique follow three main steps during training process.

Step 1. Select the initial residual points T∈$\mathcal{D}$×[0, t_{max}] and train the neural networks for the limited number of iterations.

Step 2. Estimate the total loss terms, which include the PDE residual loss, initial conditions loss, boundary condition loss, and gradient of PDE loss.

$$\mathcal{L}_{total} = \mathcal{L}_{PDE} + \lambda_x \mathcal{L}_{\partial_x \mathcal{R}} + \lambda_y \mathcal{L}_{\partial_y \mathcal{R}} + \lambda_t \mathcal{L}_{\partial_t \mathcal{R}} + \mathcal{L}_{IC} + \mathcal{L}_{BC}$$

Step 3.

- If $\mathcal{L}_{total} < \delta$ (where δ is some threshold), the training stop.
- Otherwise, select some points from the region where the residual is large and add them to the training process
- Retrain the model and repeat ***step 2.***

V. Experimental Results And Discussion

In this section we discuss the solution plots two dimensional convection-diffusion equation.

A. Governing Equation

$$\frac{\partial T}{\partial x} + u\frac{\partial T}{\partial x} + v\frac{\partial T}{\partial x} = \Gamma\left(\frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2}\right)$$

with spatial domain [0, L]×[0, H] = [0, 20]×[0, 20] and the temporal domain is t∈[0, 1].

- ***Initial conditions***

$$T(x, y, 0) = e^{\left(\frac{u}{2\Gamma}(x+y)\right)} \sin\left(\frac{\pi x}{L}\right) \sin\left(\frac{\pi y}{H}\right)$$

- ***boundary condition***

$$T(0, y, t) = T(0, y, t) = 0,$$
$$T(0, y, t) = T(0, y, t) = 0.$$

B. Dimentionless form of Governing Equation

$$\frac{\partial T^*}{\partial t^*} + \frac{\partial T^*}{\partial x^*} + \frac{u\partial T^*}{v\partial t^*} = \frac{1}{Pé}\left(\frac{\partial^2 T^*}{\partial x^{*2}} + \frac{\partial^2 T^*}{\partial y^{*2}}\right)$$

Here Pé is Péclet number. Dimensionless form is more general and reduces the number of parameters.

C. Péclet number

Péclet number (Pé) is the dimensionless number which is very important in solving Convection-Diffusion Equations (CDEs). Pé define the rate of advection and diffusion term so it vital role in solving such type of equations. It can be defined as:

$$Pé = \frac{uL}{\Gamma}$$

u=flow velocity, L=characteristic length, Γ=diffusion coeficient

The Péclet number shows whether the equation is convection dominated or not. If Pé is less than one (ie Pé<1) diffusion term dominate, in convection dominated case Pé>1. In this paper we solve convection dominated PDEs, because their solution is challenging for standard PINNs.

D. Solution plots

The in following section we have solution plots:
(a) for PINNs,
(b) for PINNs+RAR,
and (c) for AG-PINNs.

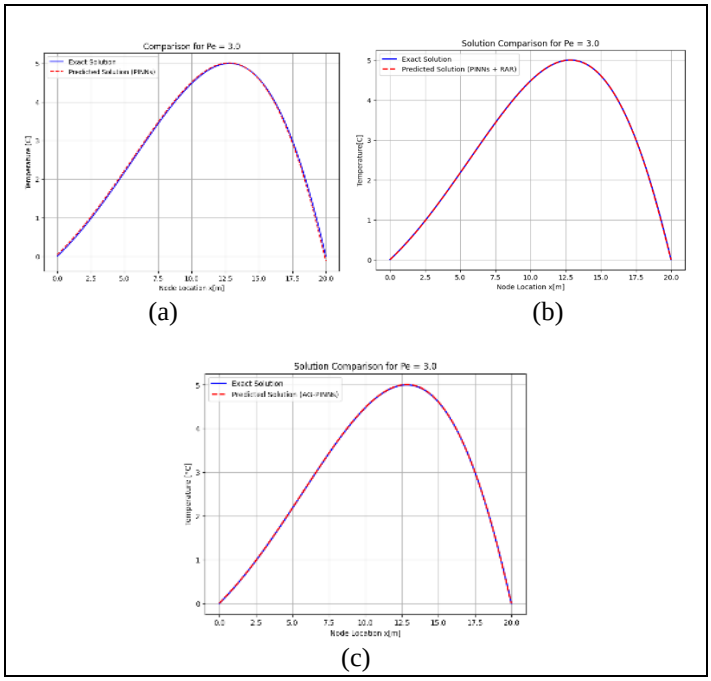


figure 5.1 show the solution of CDE for Pé = 3

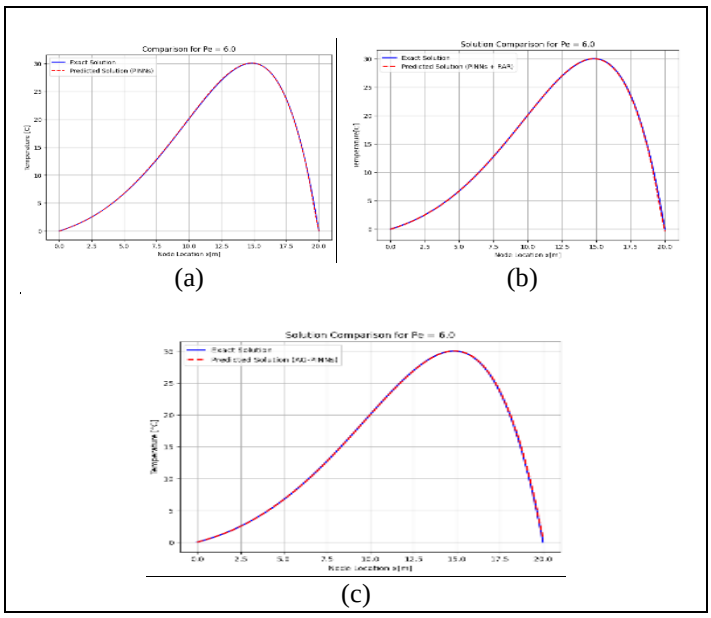
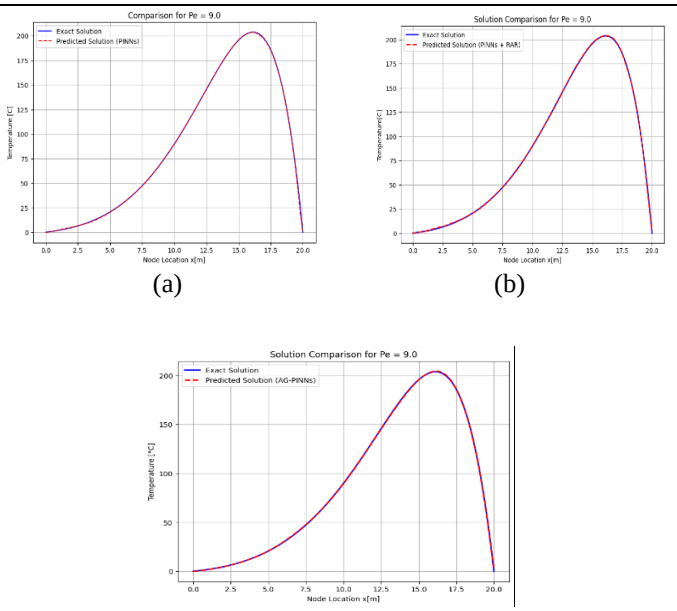


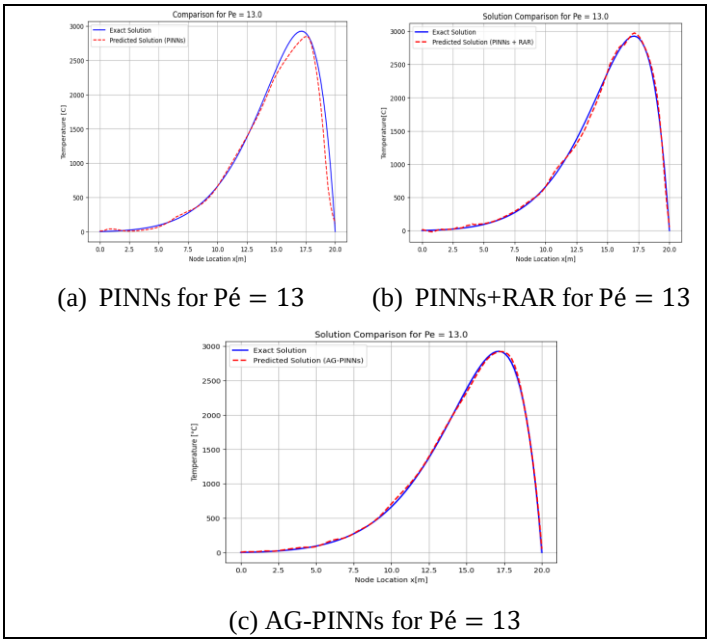
figure 5.2 show the solution of CDE for Pé = 6



(a) (b)

(c)

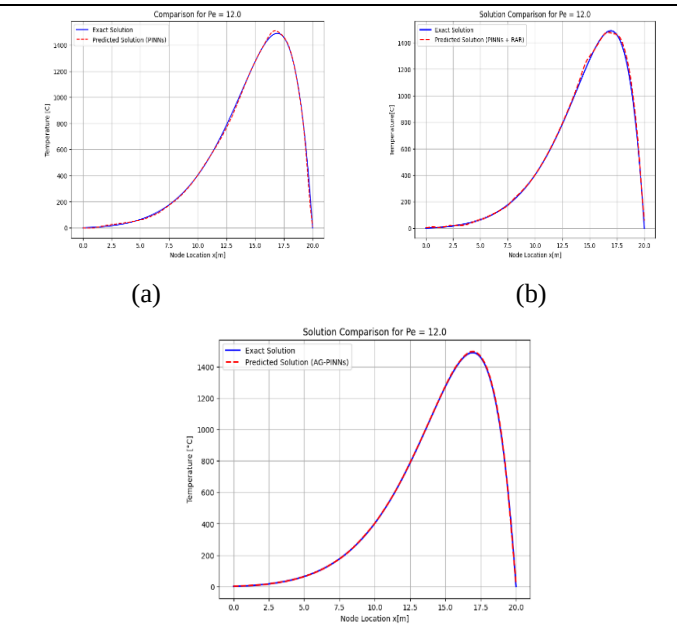
figure 5.3 show the solution of CDE for Pé = 9



(a) PINNs for Pé = 13 (b) PINNs+RAR for Pé = 13

(c) AG-PINNs for Pé = 13

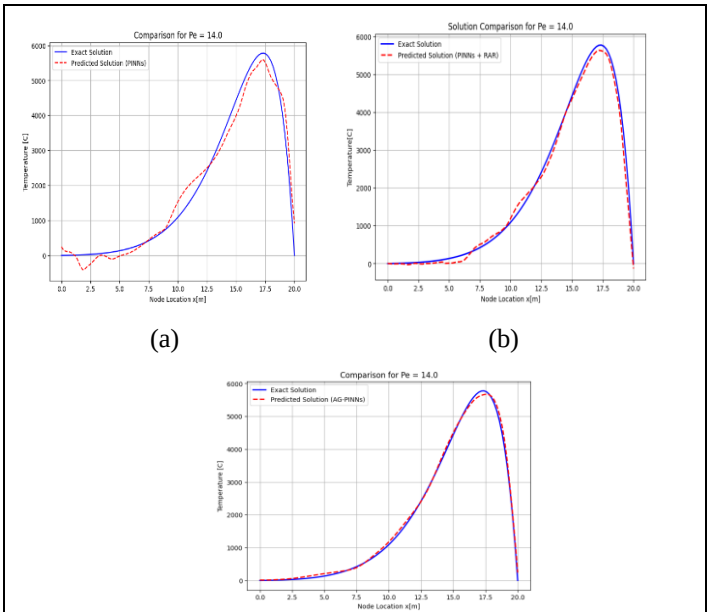
figure 5.5 show the solution of CDE for Pé = 13



(a) (b)

(c)

figure 5.4 show the solution of CDE for Pé = 12



(a) (b)

(c)

figure 5.6 show the solution of CDE for Pé = 14

E. Quantitative Evaluation

In the following tabme the L_2 relatve error of three neural networks are given:

Pé	PINNs	PINNs+RAR	AG-PINNs
3	$1.841e-3$	$3.004e-3$	$1.086e-3$
6	$3.259\,e-3$	$3.0345-3$	$1.656e-3$
9	$1.675\,e-2$	$2.007e-2$	$4.982e-3$
12	$5.734\,e-2$	$2.128e-2$	$1.387e-2$
13	$1.015\,e-1$	$3.028e-2$	$2.112e-2$
14	$6.784\,e-1$	$5.821e-2$	$4.619e-2$

VI. CONCLUSION AND FUTURE WORK

This paper introduces the Adaptive Gradient-enhanced Physics-Informed Neural Networks (AG-PINNs) model of numerical solution of partial differential equations. AG-PINNs is fusion of residual-based adaptive refinement method and gradient of residual of PDE, thus enforce the governing PDE and its spatio-temporal gradient into the loss function while dynamically refining its collocation points. Hence, this proposed formulation improving the numerical solution, especially at the regions of large residual or steep gradients or convection dominated parts, where the standard PINNs struggle. The numeric studies show that AG-PINNs are more accurate and faster converging than the standard PINNs and PINNs with RAR. Specifically, gradient residuals are important in improving the capability of the network to represent local solution characteristics and adaptive refinement is used to optimally redistribute collocation points. The corresponding decrease in relative L_2-errors, particularly in high regimes of Péclet number, is a validation of the efficacy and strength of the AG-PINNs in forward and inverse problems of the PDE under limited availability of data.

The mathematical studies indicate that AG-PINNs are more accurate and converge faster than standard PINNs and PINNs enhanced with RAR. Specifically, gradient residuals are important to the extent that they greatly improve the capacity of the network to retain the local solution characteristics and adaptive refinement provides efficient distribution of the collocation points [48]. The observe results in L_2- relative error and high Péclet number conditions, in particular, vindicate the strengths and efficacy of AG-PINNs to the forward and inverse PDE solves where constrained data is accessible.

It is possible that in the future, the AG-PINNs framework can be extended to high-dimensional, highly nonlinear PDEs and coupled and multiphysics problems [49]. Subsequent analytical treatment of convergence and generalization properties especially with respect to gradient based loss terms is also an area of significance [50]. By adaptive optimization strategy numerical stability and efficiency can be enhanced. Moreover, the combination of AG-PINNs and more sophisticated neural structures, e.g., multi-scale or operator-learning models, and the applicability of the approach to real-world applications with messy or incomplete data are promising directions of the future research.

REFERENCES

[1] J. Cadena-Morales, C. L´opez-Castro, J. Alba-Maldonado, Applications of differential equations to model the physical phenomenon of heat transfer with an internal energy source, in: Journal of Physics: Conference Series, Vol. 2102, IOP Publishing, 2021, p. 012018.

[2] H. Nguyen, R. Tsai, Numerical wave propagation aided by deep learning, Journal of Computational Physics 475 (2023) 111828.

[3] J. W. Sanders, A. C. DeVoria, N. J. Washuta, G. A. Elamin, K. L. Skenes, J. C. Berlinghieri, A canonical hamiltonian formulation of the navier–stokes problem, Journal of Fluid Mechanics 984 (2024) A27.

[4] H. Lhachemi, R. Shorten, Boundary output feedback stabilization of state delayed reaction–diffusion pdes, Automatica 156 (2023) 111188.

[5] S. Ghosh-Dastidar, H. Adeli, Spiking neural networks, International journal of neural systems 19 (04) (2009) 295–308.

[6] V. Davydovych, V. Dutka, R. Cherniha, Reaction–diffusion equations in mathematical models arising in epidemiology, Symmetry 15 (11) (2023) 2025.

[7] J. H. Lagergren, J. T. Nardini, G. Michael Lavigne, E. M. Rutter, K. B. Flores, Learning partial differential equations for biological transport models from noisy spatio-temporal data, Proceedings of the Royal Society A 476 (2234) (2020) 20190800.

[8] X. Yu, K. Lan, J. Wu, Green's functions, linear second-order differential equations, and one-dimensional diffusion advection models, Studies in Applied Mathematics 147 (1) (2021) 319–362.

[9] M. Raissi, P. Perdikaris, G. E. Karniadakis, Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations, Journal of Computational physics 378 (2019) 686–707.

[10] W. S. McCulloch, W. Pitts, A logical calculus of the ideas immanent in nervous activity, The bulletin of mathematical biophysics 5 (1943) 115–133.

[11] H. S. Gasser, J. Erlanger, The role played by the sizes of the constituent fibers of a nerve trunk in determining the form of its action potential wave, American Journal of Physiology-Legacy Content 80 (3) (1927) 522–547.

[12] F. Rosenblatt, The perceptron: a probabilistic model for information storage and organization in the brain., Psychological review 65 (6) (1958) 386.

[13] M. Minsky, S. Papert, An introduction to computational geometry, Cambridge tiass., HIT 479 (480) (1969) 104.

[14] P. Werbos, Beyond regression: New tools for prediction and analysis in the behavioral sciences, PhD thesis, Committee on Applied Mathematics, Harvard University, Cambridge, MA (1974).

[15] K. Fukushima, Cognitron: A self-organizing multilayered neural network, Biological cybernetics 20 (3) (1975) 121–136.

[16] T. Kohonen, Self-organized formation of topologically correct feature maps, Biological cybernetics 43 (1) (1982) 59–69.

[17] J. J. Hopfield, Neural networks and physical systems with emergent collective computational abilities., Proceedings of the national academy of sciences 79 (8) (1982) 2554–2558.

[18] D. E. Rumelhart, G. E. Hinton, R. J. Williams, et al., Learning internal representations by error propagation (1985).

[19] G. A. Carpenter, S. Grossberg, A massively parallel architecture for a selforganizing neural pattern recognition machine, Computer vision, graphics, and image processing 37 (1) (1987) 54–115.

[20] M. Liu, H. Dong, Y. Fang, Y. Zhang, Lie symmetry analysis of burgers equation and the euler equation on a time scale, Symmetry 12 (1) (2019) 10.

[21] J. Schmidhuber, Deep learning in neural networks: An overview, Neural networks 61 (2015) 85–117

[22] M. A. Nielsen, Neural Networks and Deep Learning, Determination Press, 2015, http://neuralnetworksanddeeplearning.com/.

[23] D. E. Rumelhart, G. E. Hinton, R. J. Williams, Learning representations by back-propagating errors, nature 323 (6088) (1986) 533–536.

[24] M. Raissi, P. Perdikaris, G. E. Karniadakis, Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations, Journal of Computational physics 378 (2019) 686–707.

[25] Y. LeCun, Y. Bengio, G. Hinton, Deep learning, nature 521 (7553) (2015) 436–444.

[26] A. Krizhevsky, I. Sutskever, G. E. Hinton, Imagenet classification with deep convolutional neural networks, Advances in neural information processing systems 25 (2012).

[27] B. M. Lake, R. Salakhutdinov, J. B. Tenenbaum, Human-level concept learning through probabilistic program induction, Science 350 (6266) (2015) 1332–1338.

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

[29] Z. Long, Y. Lu, X. Ma, B. Dong, Pde-net: Learning pdes from data, in: Proceedings of the 35th International Conference on Machine Learning, PMLR, 2018, pp. 3208–3216.

[30] Y. Khoo, J. Lu, L. Ying, Solving parametric pde problems with artificial neural networks, European Journal of Applied Mathematics 32 (3) (2021) 421–435.

[31] J. Sirignano, K. Spiliopoulos, Dgm: A deep learning algorithm for solving partial differential equations, Journal of Computational Physics 375 (2018) 1339–1364.

[32] S. Lin, Y. Chen, A two-stage physics-informed neural network method based on conserved quantities and applications in localized wave solutions, Journal of Computational Physics 457 (2022) 111053.

[33] J. Guan, H. Elman, Transformed physics-informed neural networks for the convection-diffusion equation, arXiv preprint arXiv:2409.07671Submitted September 2024 (2024).

[34] A. Beguinet, H. Owhadi, C. Schwab, Deep learning for singularly perturbed convection–diffusion equations, arXiv preprint arXiv:2205.04779 (2022). URL https://arxiv.org/abs/2205.04779

[35] Y. Lin, W. Chen, Gradient-enhanced physics-informed neural networks based on transfer learning for inverse problems of the variable coefficient differential equations, Engineering Analysis with Boundary Elements 152 (2023) 58–72. doi:10.1016/j.enganabound.2023.03.011.

[36] M. S. Khan, K. M. Abualnaja, A. Sagheer, M. A. Memon, A. Fenta, Neuralnetwork-based numerical analysis of some convection–diffusion–reaction equations, AIP Advances 14 (12) (2024) 125224. doi:10.1063/5.0239079

[37] D. Frerichs-Mihov, L. Henning, V. John, On loss functionals for physicsinformed neural networks for steady-state convection-dominated convectiondiffusion problems, Communications on Applied Mathematics and Computation (2024). doi:10.1007/s42967-024-00433-7

[38] I. Hariri, A. Radid, K. Rhofir, Physics-informed neural networks methodology for the resolution of some turing's type models., Advanced Mathematical Models & Applications 10 (1) (2025).

[39] A. Pervez, E. Gavves, F. Locatello, Mechanistic pde networks for discovery of governing equations, arXiv preprint arXiv:2502.18377 (2025).

[40] V. Volpert, S. Petrovskii, Reaction-diffusion waves in biology: new trends, recent developments, Physics of Life Reviews 52 (2025) 1–20.

[41] K. V. Park, Towards a foundation model for physics-informed neural networks: Multi-pde learning with active sampling, arXiv preprint arXiv:2502.07425 (2025).

Javeed Ahmad, born on 15 july 1994, is a postgraduate student in the Department of Basic Sciences and Islamiat, University of Egineering and Technology, Peshawar, Pakistan.

How to cite this article:

Javeed Ahmad "Adaptive Gradient-Enhanced PINNs for Numerical Solution of Convection-Diffusion Equation" International Journal of Engineering Works, Vol. 13, Issue 02, PP. 13-21, February 2026. https://doi.org/10.5281/zenodo.18753514.