Physics-informed Neural Networks with Adaptive Dynamic Adjustment for Neutron Transport in simple Geometry

Jeongmin Kang, Minseop Song*
Department of Nuclear engineering, Hanyang Univ., 222 Wangsimni-ro, Seongdong-gu, Seoul
*Corresponding author: hysms@hanyang.ac.kr

Keywords : Neutron transport equation; Physics-informed neural network; Adaptive Dynamic Adjustment; Deep learning; Ray effect

1. Introduction

The neutron transport equation (NTE) governs neutron behavior in materials and is essential for predicting neutron flux in reactor cores. Due to its seven variables, solving NTE analytically is highly complex, necessitating numerical methods such as deterministic (e.g., finite difference, MOC, spherical harmonics) and non-deterministic (Monte Carlo) approaches. However, these methods face challenges like high computational costs, implementation complexity, and sensitivity to dimensionality.

To address these issues, this study introduces a deep learning-based approach using Physics-Informed Neural Networks (PINN). Unlike traditional methods, PINN is mesh-free, mitigating the curse of dimensionality and adapting well to complex geometries while providing continuous solutions without requiring high-quality meshes.

This study presents Adaptive Dynamic Adjustment PINN (ADA-PINN), a newly developed model for solving NTE. In conventional PINN, as the geometry becomes more complex, numerous loss terms arise, requiring users to manually set weights for each loss function to achieve optimal total loss. However, as the number of losses increases, it becomes more difficult for users to grasp the priority of losses. Therefore, we developed an ADA technique that utilizes the second derivative of the loss function using curvature-based loss adjustment. As the result, ADA-PINN overcomes this limitation by automatically adjusting loss weights during training, leading to more efficient and stable convergence without manual hyperparameter tuning. Additionally, ADA-PINN applies optimal domain decomposition and multi-group equation solving. It also incorporates a training dataset repositioning technique to reduce errors near boundaries, further enhancing computational efficiency and accuracy. These advancements establish ADA-PINN as a promising alternative for solving NTE compared to conventional numerical methods.

The paper is structured as follows: Section 2 introduces NTE, ADA-PINN, and the training techniques used; Section 3 presents test cases for 1D and 2D NTE problems, comparing ADA-PINN with existing data; and Section 4 summarizes key conclusions.

2. Methods

The neutron transport equation (NTE) differs from the neutron diffusion equation (NDE), requiring distinct PINN approaches. As an integro-differential equation, NTE involves complex integrations and directional-energy relationships, leading to intricate matrix transformations.

Its non-smoothness near interfaces introduces errors, while angular discretization causes the Ray effect, reducing accuracy. To address these challenges in multi-energy NTE, this study applies four key techniques. This section provides an overview of NTE and PINN, explains the model structure, and details the proposed techniques.

2.1 Neutron transport equation

The multi-group NTE is expressed as follows:

$$\frac{\partial \psi_g}{\partial t} + \vec{\Omega} \cdot \nabla_{\vec{r}} \psi_g + \Sigma_{t,g} \psi_g = Q_g(\psi_g) \qquad (1)$$

where $\psi_g(\vec{r}, \vec{\Omega}, t)$represents the spatial vector $\vec{r}$, the directional vector $\vec{\Omega}$, and the time variable t, defining a specific energy group, g. $\Sigma_{t,g}\vec{r}$ is the macroscopic total cross-section for a given energy group, g, and $Q_g(\psi)$ represents the source term, which includes the scattering term, fission term, and external source. Fig. 1. Direction variable in the unit circle. The direction variable $\vec{\Omega}$ and its components μ, η and ξ in unit sphere is illustrated with Fig.1. μ, η and ξ are projections of $\vec{\Omega}$ on the x, y and z axes, respectively.

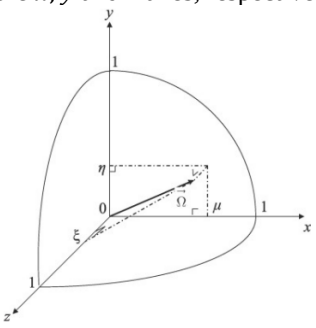

Fig. 1. Direction variable in the unit circle. [1]

In this study, scattering is assumed to be isotropic, and the fission term is neglected. Therefore, the source term is given as follows:

$$Q_g(\psi_g) = \frac{Q_{ext,g} + \sum_{g'=1}^{G} \sum_{s,g' \to g} \phi_{g'}}{4\pi}, \phi_g = \int_S \psi_g \, d\vec{\Omega} \quad (2)$$

Here, $Q_{ext,g}$ represents the external source term, which is independent of ψ_g in group g. The term $\sum_{s,g' \to g} \vec{r}$ denotes the scattering cross-section, and ϕ_g corresponds to the integral of ψ_g over all directions.

2.2 Neural networks

The Adaptive Dynamic Adjustment PINN (ADA-PINN) is a multi-layer feed-forward neural network that dynamically adjusts learning parameters. Building upon the concept of Physics-Informed Neural Networks (PINNs) proposed in Raissi et al. [2], a new variant called ADA-PINN has been developed by incorporating an adaptive weighting strategy (ADA) into the original framework. Neural networks process inputs through multiple hidden layers to achieve non-linear mappings for output predictions. Common activation functions in machine learning include sigmoid, hyperbolic tangent (tanh), and ReLU functions. Based on previous research by Xie et al. (2023) [3], the hyperbolic tangent function demonstrated superior performance, and thus was chosen for all two case studies in this research. Multi-group NTE is calculated based on single-group problems, and this section explains how ADA-PINN is applied to solve single-group NTE problems.

ADA-PINN optimizes single-group NTE calculations using loss functions and tuning parameters. During this process, tuning parameters, weights, and biases are defined. The schematic diagram of ADA-PINN is shown in Fig. 2. The process begins with the construction of the *NN*(Neural Network), followed by inputting spatial vectors, directional vectors, and time variables, which pass through the multi-layer network to generate an output. The output represents ψ at the corresponding point, expressed as:

$$\psi(\vec{r}, \vec{\Omega}, t) = NN(\vec{r}, \vec{\Omega}, t, \vec{p}) \quad (3)$$

After this step, we can get values of $\partial t, \nabla_{\vec{r}}, Q$ from neural networks. By substituting these values into the governing equation, the ideal condition is for the equation's value to be zero. This condition defines the loss function:

$$loss = \frac{\partial \psi_g}{\partial t} + \vec{\Omega} \cdot \nabla_{\vec{r}} \psi_g + \sum_{t,g} \psi_g - Q_g(\psi_g) \quad (4)$$

However, since this method approximates the solution iteratively, it never exactly reaches zero. Instead, the loss function represents the deviation from zero, which is minimized during training. Several loss terms are defined, but only the physics-based loss derived from the governing equation is mentioned here; additional loss functions will be introduced in Section 2.6. The training set comprises randomly sampled discrete points within the computational domain, each associated with velocity vectors. If the size of the training set is N_{train}, the loss function derived from L^2norm is defined as:

$$loss_{train}(\vec{p}) = \sum_{i=1}^{N_{train}} \left(\frac{\partial \psi_g}{\partial t}(\vec{r}_i, \vec{\Omega}_i, t_i, \vec{p}) + \vec{\Omega}_i \cdot \nabla_{\vec{r}} \psi_g(\vec{r}_i, \vec{\Omega}_i, t_i, \vec{p}) + \sum_{t,g}(\vec{r}) \psi_g(\vec{r}_i, \vec{\Omega}_i, t_i, \vec{p}) - Q_g(\psi_g(\vec{r}_i, \vec{\Omega}_i, t_i, \vec{p})) \right)^2 \quad (5)$$

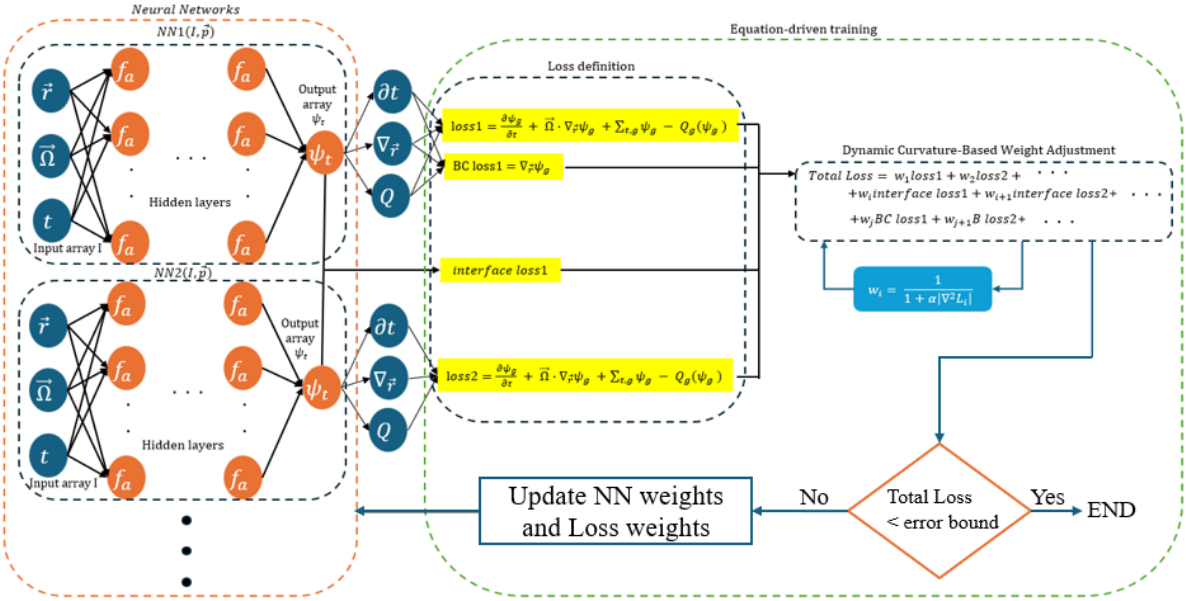

Fig. 2. Schematic diagram of ADA-PINN for NTE

where i denotes the discrete points in the training set, and tuning parameter $\vec{p}$ is used to minimize the loss toward zero. The total loss combines multiple loss functions, each assigned a different weight based on its significance. Training iterations include two main optimization steps, with loss weighting dynamically adjusted during training. The detailed explanation of dynamic loss weighting is provided in Section 2.3.

The optimization of the loss function in machine learning is typically solved using first-order gradient descent methods such as Adam (Kingma and Ba, 2014) [4] or second-order methods like LBFGS (Liu and Nocedal, 1989) [5]. In this study, Adam was used.

2.3 Adaptive dynamic adjustment training for losses

Adaptive Dynamic Adjustment Training for Losses (ADATL) dynamically adjusts the loss function's contribution during training to enhance learning efficiency. Traditional loss functions remain fixed throughout training, which can lead to unstable learning rates—some loss terms may decay too quickly or too slowly. To address this issue, a curvature-based loss adjustment technique is introduced. This technique analyzes the curvature of the loss function and adjusts its weight accordingly: reducing the weight when the loss changes rapidly and increasing it when the loss changes slowly. This ensures balanced learning across different training stages.

The core principle of curvature-based loss adjustment is utilizing the second-order derivative (curvature) of the loss function. A high curvature of the loss function indicates that the loss value is changing rapidly, which may cause the model to overreact and lead to unstable learning. Conversely, when the curvature is low, it suggests that learning has stagnated in that region, requiring the enhancement of the loss influence to encourage more active learning. This can be mathematically expressed as follows, where the loss weight w_i is defined as:

$$w_i = \frac{1}{1+\alpha|\nabla^2 L_i|} \tag{6}$$

where:
- $\nabla^2 L_i$ represents the curvature of the loss function,
- α is the adjustment coefficient.

Higher curvature $\nabla^2 L_i$ (rapid loss changes) results w_i in a lower , while lower curvature (slow loss changes) increases w_i . This prevents the model from overreacting to abrupt loss variations while reinforcing learning in stable regions.

This method enhances learning stability and optimizes convergence speed. Conventional deep learning training suffers from excessive fluctuations in learning rates when loss functions change rapidly. Curvature-based adjustment prevents extreme reactions during sharp loss changes while accelerating convergence in slow-learning regions. Additionally, this approach mitigates local minimum traps by dynamically adjusting loss weighting, increasing the likelihood of achieving a global minimum.

Additionally, curvature-based loss adjustment can demonstrate strong performance in multi-loss learning environments. In models where multiple loss functions are combined, certain losses may become disproportionately large or small compared to others, leading to imbalanced training. In the case of the neutron transport equation (NTE), loss functions are not only derived from the governing equation but also from various boundaries and initial conditions, and the number of these losses increases as the geometry becomes more complex. By applying curvature-based loss adjustment, the weights of each loss function can be dynamically adjusted based on their rate of change, enabling more balanced learning.

2.4 Rearrangement of training set

In the process of determining the training dataset, ADA-PINN follows a mesh-free approach, making it free from conventional constraints. ADA-PINN randomly samples data points, where in the case of the neutron transport equation (NTE), these random values include spatial vectors, directional vectors, and time variables. However, when multiple models are used, the fit between models must be guaranteed at the physical boundary, which requires denser allocation of data points near the region, as used in existing numerical solution techniques.

The reason for this is that while a single model can produce smooth results, NTE solutions only maintain continuity and can still be non-smooth in certain regions. To address this, dataset rearrangement was applied. Although it is possible to densely allocate data points across the entire domain for training, this approach is computationally inefficient. Instead, the goal is to strategically assign more data points to critical regions where learning needs to be emphasized, thereby achieving maximum efficiency with minimal computational cost.

In this study, this technique was applied in the final case to enhance the efficiency of the training process.

2.5 Multi-group iteration

In the case of multi-group NTE, the source term includes neutrons scattered from other energy groups. That is, when there are two groups, they are coupled through an integral-differential equation. This can be approximated using the source iteration method (Hébert, 2010) [6], which is a conventional numerical approach.

The process of applying this method to PINN is as follows: First, each source term is initially assigned a

random value. Then, the governing equation for each group is solved using the techniques described in Sections 2.2 and 2.3. Subsequently, the updated source term is obtained, and this process is repeated in each training iteration until convergence is achieved.

2.6 Area decomposition

As geometrical complexity increases, the benefits of training data rearrangement diminish. A viable alternative is area decomposition, inspired by Wang et al. (2022) [7], who applied PINN to solve the neutron diffusion equation (NDE). Unlike NDE, NTE lacks a second-order derivative term, so continuity conditions for first-order derivatives do not apply. The loss function incorporating inter-region continuity is defined as follows:

$$Loss = \sum_{i=1}^{M} loss_{train,i} + \sum_{i=1}^{P} loss_{interface,i} + \sum_{i=1}^{R} loss_{BC,i} \quad (7)$$

where M is the number of regions, P is the number of interfaces, $loss_{interface,i}$ represents inter-region continuity error, R is the number of boundary conditions, and $loss_{BC,i}$ represents boundary condition errors. The $loss_{interface,i}$ is given by:

$$loss_{interface,i} = \sum_{j=1}^{N_{interface,i}} (\psi_m(\vec{r}_j, \vec{\Omega}_j, t_j, \overrightarrow{p_j}) - \psi_n(\vec{r}_j, \vec{\Omega}_j, t_j, \overrightarrow{p_j}))^2 \quad (8)$$

where $N_{interface,i}$ denotes the number of discrete points at the interface, and ψ_m and ψ_n represent ψ values on either side of the interface. Similarly, boundary condition loss $loss_{BC,i}$ is defined as:

$$loss_{BC,i} = \sum_{j=1}^{N_{interface,i}} \left(\nabla\psi_{symmetry}(\vec{r}_j, \vec{\Omega}_j, t_j, \overrightarrow{p_j})\right)^2 + \sum_{j=1}^{N_{interface,i}} (\psi_{inlet}(\vec{r}_j, \vec{\Omega}_j, t_j, \overrightarrow{p_j}))^2 \quad (9)$$

For $\nabla\psi_{symmetry}(\vec{r}_j, \vec{\Omega}_j, t_j, \overrightarrow{p_j})$, by using Dirichlet boundary conditions, where the gradient should be zero, the loss is minimized as the gradient approaches zero. For $\psi_{inlet}(\vec{r}_j, \vec{\Omega}_j, t_j, \overrightarrow{p_j})$, from vacuum boundary conditions, where ψ_{inlet} values should be zero at the boundary. To be more specific, there are zero neutrons coming from outside to geometry, so the loss is defined as above.

2.7 *Random uniform dataset for ray effect*

The ray effect that occurs when solving the NTE arises from directional bias in fixed-direction methods such as s16 and s32, where information is concentrated along specific directions. To mitigate this issue, using random uniform sampling in PINN can help distribute the solution more evenly by reducing directional bias. While traditional methods confine data flow to specific directions, exacerbating the ray effect, random sampling allows for learning from various directions, improving generalization performance and smoothing the loss landscape, leading to more stable gradient updates. However, random sampling alone does not completely resolve the problem, and strategies such as proper boundary condition handling and importance sampling must also be considered. For instance, when dealing with boundary conditions like vacuum conditions, where neutrons only enter from specific directions, the range of random sampling values can be adjusted to ensure that only outgoing neutrons exist at the boundary.

3. Results and discussion

In this study, two cases were analyzed, where ADA-PINN was applied to NTE and compared with conventional numerical methods that utilize iteration. Case 1 was applied to the 1D-1G Reed's problem (William, 1971) [8], while Case 2 was applied to the 2D-2G TWIGL(Hoffman and Lee, 2016) [9] problem. For both cases, parallel computation was performed using a single NVIDIA GeForce 4060 card. The structure of the neural network and the training set size are listed in Table 1.

Table 1. Summary of important parameters used in cases.

	Number of hidden layers	Number of neurons in each layer	Training set size
Case 1	3	20	1000
Case 2	12	512	14000

3.1 Case 1: 1D transport problem with scattering

Case 1 is a one-dimensional mono-energy neutron transport equation (NTE) problem that includes a scattering term. In this case, Eq. (1) can be simplified as follows:

$$\mu \frac{\partial \psi}{\partial x} + \sum_t \psi = \frac{Q_{ext} + \sum_s \phi}{4\pi} \quad (10)$$

Here, $\psi(x, \mu)$ represents the neutron angular flux, while ϕ denotes the neutron scalar flux, which is defined from the equation (2).

This case study was inspired by Reed's problem, which is commonly used as a benchmark test problem for transport codes. The structure consists of heterogeneous materials, including strong absorbers, vacuum regions, and scattering regions, as illustrated in Fig. 3.

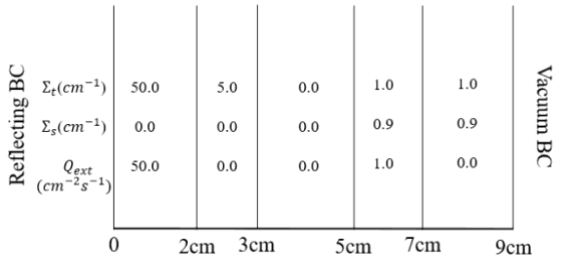

Fig. 3 Multi-region transient Reed's problem

These regions are useful for testing various aspects of numerical discretization. For example, the vacuum region presents challenges when applying the second-order form of the transport equation.

By applying the boundary conditions (BC) provided in the problem, we can rewrite the BC as:

$$\nabla\psi(x=0,\mu)=0 \quad (11)$$
$$\psi(x=8,\mu<0)=0 \quad (12)$$

In this case, there are a total of five regions, so five separate models were assigned accordingly. Additionally, training set rearrangement was applied at the boundaries between regions to generate data points, as shown in Fig. 4. In this context, the x-axis represents the position of a neutron within a one-dimensional geometry, while the y-axis indicates the angular direction of the neutron.

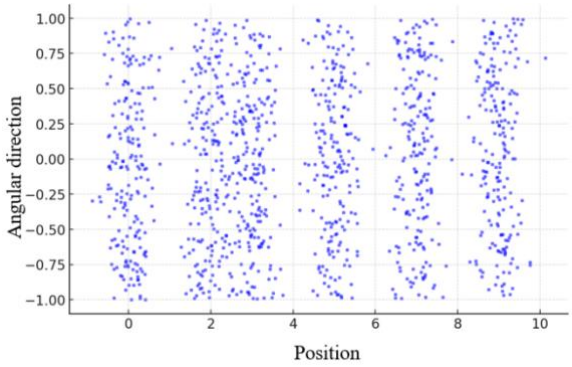

Fig. 4. Train set of Case 1

The comparison between the results obtained using the above dataset and the dynamically weighted total loss assignment and the reference values is shown in Fig. 5. Additionally, the loss history recorded during training is presented in Fig. 6.

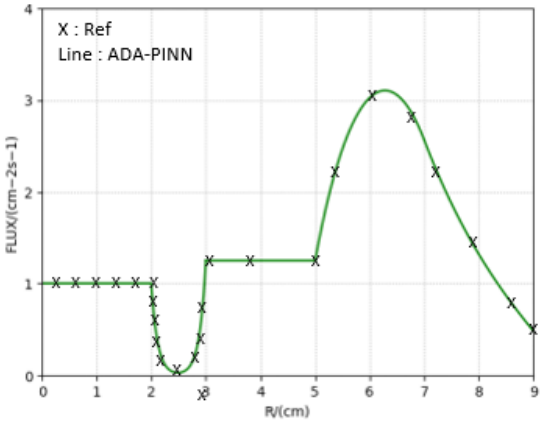

Fig. 5 Predicted solution comparison between Reference & ADA-PINN

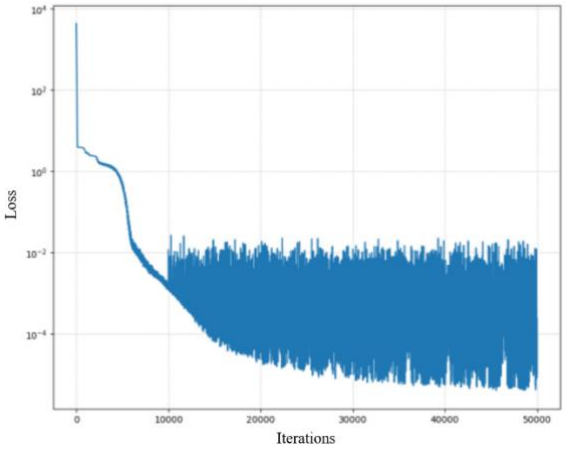

Fig. 6 Loss history of ADA-PINN for Case1

For the 1D-1G case, a comparison between the reference and ADA-PINN results shows that high-accuracy results were obtained despite using a relatively small number of assigned nodes and hidden layers.

Examining the loss history, fluctuations in the loss values begin around the 10,000th iteration. However, the minimum loss continues to decrease steadily, indicating that the training process proceeded normally. By approximately the 50,000th iteration, the loss converges to 10^{-4}. The training time took a total of 6 hours and 32 minutes. And when using the completed learning model, it took 0.32 seconds to derive the result.

3.2 Case 2: 2D transport problem with multi-groups

Case 2 primarily aims to validate ADA-PINN in a 2D multi-group neutron transport problem. This problem involves a stationary neutron transport equation for a two-group system with absorption and scattering materials. In this case, the multi-group iteration described in Section 2.5, the area decomposition method discussed in Section 2.6 and

random uniform dataset for ray effect described in Section 2.7 are applied.
Eq. (1) for this case can be simplified as follows:

$$\begin{cases} \mu\frac{\partial\psi_1}{\partial x}+\eta\frac{\partial\psi_1}{\partial y}+\Sigma_{t,1}\psi_1=\frac{Q_{ext,1}+\Sigma_{s,1\to1}\phi_1+\Sigma_{s,2\to1}\phi_2}{4\pi} \\ \mu\frac{\partial\psi_2}{\partial x}+\eta\frac{\partial\psi_2}{\partial y}+\Sigma_{t,2}\psi_2=\frac{Q_{ext,2}+\Sigma_{s,2\to1}\phi_1+\Sigma_{s,2\to2}\phi_2}{4\pi} \end{cases} \quad (13)$$

Here, ψ_1 and ψ_2 represent the neutron angular flux for the fast group and thermal group, respectively, while $\phi_1$and ϕ_2 denote the neutron scalar flux for the fast and thermal groups.

Case 2 was inspired by TWIGL. The geometry and boundary conditions for this problem are illustrated in Fig. 7, while the physical parameters are listed in Table 2. In the steady-state condition of TWIGL, the external source term $Q_{ext,1}$ is shown in Fig. 8, and $Q_{ext,2}$=0.

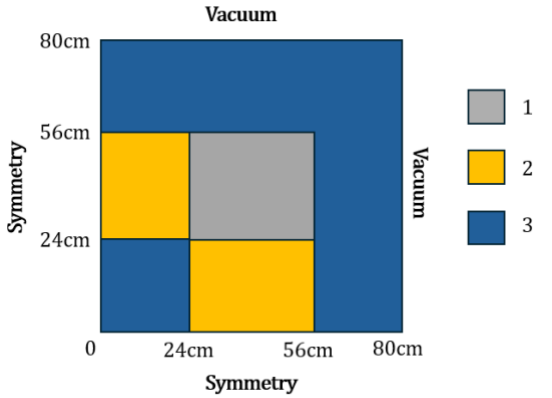

Fig. 7 Geometry of Case 2.

Table 2. Physical parameters of Case 2.

Region	Group, g	$\Sigma_t(cm^{-1})$	$\Sigma_{s,g\to g}(cm^{-1})$	$\Sigma_{s,g\to g'}(cm^{-1})$
1	1	0.238095	0.218095	0.01
	2	0.83333	0.68333	0
2	1	0.238095	0. 218095	0.01
	2	0.83333	0.68333	0
3	1	0.25641	0.23841	0.01
	2	0.66666	0.616667	0

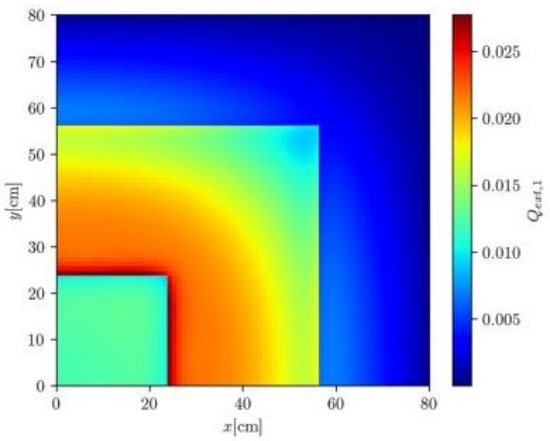

Fig. 8 $Q_{ext,1}$ of Case 2.

The solutions obtained using ADA-PINN and the Finite Element Method (FEM)-based BenchMark Solution(BMS) are shown in Fig. 9 and Fig. 10. For further comparison, the error values between BMS and ADA-PINN within the given geometry are presented in Fig. 11 For errors compared to BMS, ADA-PINN achieved an average error of 0.0017 for the fast group and 0.0011 for the thermal group. From these figures, it is evident that the ADA-PINN solution closely matches the FEM solution. This demonstrates that the proposed ADA-PINN effectively and flexibly simulates multi-group heterogeneous problems with high accuracy. The training time took a total of 18 hours and 29 minutes. And when using the completed learning model, it took 1.84 seconds to derive the result.

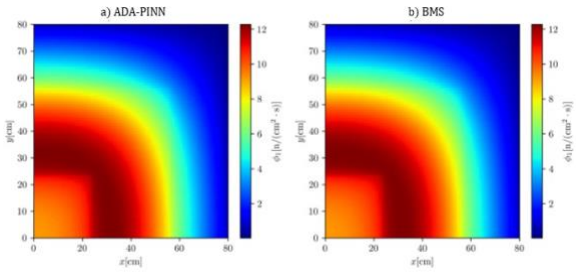

Fig. 9 Global comparison of ϕ_1 in Case 2.

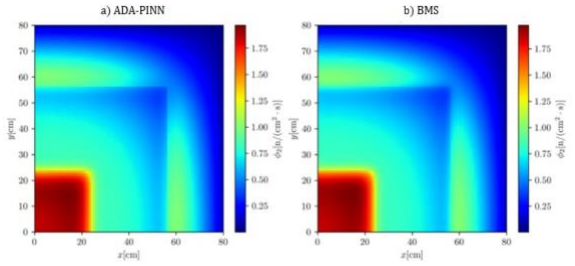

Fig. 10 Global comparison of ϕ_2 in Case 2.

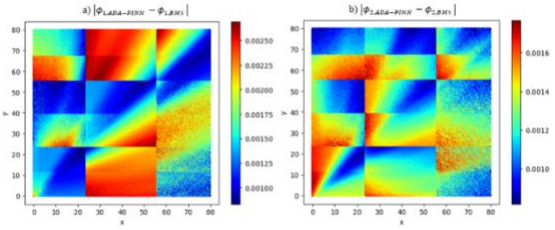

Fig. 11 Difference values between ADA-PINN and BMS over the geometry

4. Conclusions

In this study, the Adaptive Dynamic Adjustment Physics-Informed Neural Network (ADA-PINN) was used to numerically analyze the Neutron Transport Equation (NTE). To address the limitations of conventional deterministic and non-deterministic methods, a deep learning-based PINN approach was applied, incorporating Adaptive Dynamic Adjustment for Training Losses (ADATL) and a Training Set Rearrangement technique to enhance computational efficiency and accuracy.

For the first case study (Case 1), ADA-PINN was tested on the 1D-1G Reed's Problem, demonstrating high accuracy with fewer training nodes and hidden layers than traditional numerical methods. The dynamic loss weighting adjustment technique improved training stability and convergence speed, as confirmed by tracking loss function evolution.

In the second case study (Case 2), ADA-PINN was applied to the 2D-2G TWIGL problem, solving the multi-group neutron transport equation. The Multi-group Iteration and Area Decomposition techniques enabled a flexible computational approach while maintaining accuracy comparable to the FEM-based BMS method. The Fast and Thermal Groups achieved average error rates of 0.0017 and 0.0011, respectively, confirming the method's high computational speed and accuracy.

This study contributes to the field by mitigating the Ray-effect issue of conventional numerical methods, improving training stability and convergence speed with ADATL, reducing boundary errors through training dataset rearrangement, and applying area decomposition for solving multi-group equations and complex geometries with greater precision. Furthermore, the automated loss weighting adjustment technique eliminates the need for manual hyperparameter tuning, enhancing optimization efficiency.

Future research will focus on extending ADA-PINN to transient multi-group neutron transport problems in complex geometries for real reactor simulations. Additional verification is needed for cases with severe Ray-effects, such as anisotropic environments, and the proposed loss adjustment and dataset rearrangement techniques should be applied to various physical models to assess their generality and applicability.

Acknowledgement

This research was supported by the National Research Council of Science & Technology (NST) grant by the Korea government (MSIT) (No. GTL24031-000).

REFERENCES

[1] Altac, Z., Bao, H., Dang, Z., Florio, M. D., Gao, H., Gnudi, A., Hoffman, A. J., Holloway, J., Jin, X., Kim, T. K., Li, R., Lou, Q., Mao, Z., Mishra, S., Mostajeran, F., Raissi, M., Schiassi, E., Sinha, S., Sun, L., … Wang, J., *Boundary dependent physics-informed neural network for solving neutron transport equation*. Annals of Nuclear Energy. Vol.195, p.112, 2023.

[2] 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*, Vol. 378, p. 686–707, 2019.

[3] Xie, Y., & Wu, J., *HGTHP: A novel hyperbolic geometric transformer hawkes process for event prediction - applied intelligence*. Applied Intelligence, Vol. 54, p. 357-374, 2023.

[4] Kingma, D. P., & Ba, J., *Adam: A method for stochastic optimization*. arXiv.org. Vol. 1412.6980, 2014.

[5] Liu, D. C., & Nocedal, J., *On the limited m emory BFGS method for large scale optimization - mathematical programming*, Vol. 45, p. 503-52 8, 2021.

[6] Hébert, A. Multigroup Neutron Transport and Diffusion Computations. In: Cacuci, D.G. (eds) H andbook of Nuclear Engineering. Springer, Bosto n, MA., p. 751-911, 2010.

[7] Chen, Y., Wang, W., & Zhang, D., Surrogate modeling for neutron diffusion problems based on physics-informed neural networks. *Annals of Nuclear Energy*, Vol. 176, p. 109-234, 2022.

[8] Reed, W. H., New Difference Schemes for t he Neutron Transport Equation. *Nuclear Science a nd Engineering*, Vol. *46*(2), p. 309–314, 1971.

[9] Yasinsky, J B, et al., TWIGL: A PROGRAM TO SOLVE THE TWO-DIMENSIONAL, TWO-GROUP, SPACE-TIME NEUTRON DIFFUSION EQUATIONS WITH TEMPERATURE FEEDBACK, AEC Research and Development Report, WAPD-TM-743, 1968.