Transactions of the Korean Nuclear Society Autumn Meeting
Changwon, Korea, October 24-25, 2024

Physics-informed Neural Network for Neutron Transport Simulation

Byoungil Jeon[a]*, Hogeon Seo[a], Heesun Choi[a], and Yonggyun Yu[a]
[a)]Applied Artificial Intelligence Section, Korea Atomic Energy Research Institute, Republic of Korea
E-mail: bijeon@kaeri.re.kr*

***Keywords : machine learning, physics-informed neural network, neutron transport, diffusion equation**

1. Introduction

Calculating various interactions of neutrons with matters is an essential task for the reactor physics. This requires the distribution of neutron flux and their energy within the reactor. These distributions are acquired by solving neutron transport problems. Neutron transport follows the Boltzmann equation, solved using particle-based approaches, making it impossible to acquire the solution directly. With several assumptions, we can derive the neutron diffusion equation, which has better computational efficiency than the Boltzmann equation.

Even with the neutron diffusion theory, calculating neutron flux distributions with high spatial resolution requires a huge computational load. For several of next-generation nuclear reactors or cutting edge techniques for nuclear safety (e.g. digital twin), it is required that rapid neutronics analysis with high resolutions. Therefore, in these cases, we need to accelerate the neutronics simulations.

Among simulation acceleration techniques, physics-informed machine learning is spotlighted, because it can achieve better performance even with small dataset than traditional deep learning methods through embedding the physics information into the problem domains. In this study, we introduce an example of the physics-informed neural network (PINN) for neutron transport simulation. We developed a PINN based on the diffusion equation, and compared its performance with a numerical solver [1] and an artificial neural network (ANN) for the Reed's problem [2].

2. Neutron diffusion problem

The physics governing this problem is the neutron diffusion equation. The steady-state neutron diffusion equation for a slab geometry is represented by the following equation:

$$-\nabla\cdot D(x)\nabla\phi(x)+\Sigma_a(x)\phi(x) = \nu\Sigma_f(x)\phi(x)+Q(x) \quad (1)$$

where, D is the diffusion coefficient, Σ_a is the macroscopic absorption cross-section, ν is the average number of neutrons per fission, Σ_f is macroscopic fission cross-section, and the Q is the external source term.

We also have two general boundary conditions:

$$\left.\frac{d\phi}{dx}\right|_{x=0} = 0 \quad (2)$$

$$\mathrm{A}(L)\phi(L)+\mathrm{B}(L)\left.\frac{d\phi}{dx}\right|_{x=L} = C(L) \quad (3)$$

The Reed's problem is a test problem for neutron transport codes. It is a heterogeneous reactor problem that has several region. The problem we set up consists of following 5 regions:

- strong absorber with a strong source from x=0~2,
- strong absorber without source from x=2~3,
- void from x=3~4,
- scatter with source from x=5~7, and
- scatter without source from x= 7~10.

Fig. 1 depicts material properties of the Reed' problem.

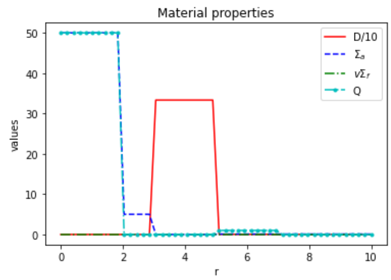


Fig. 1. Material properties of the Reed's problem

3. Physics-informed neural network

We can embed the physics of the system using the PINN technique. The equation and conditions above are not considered during the design of the neural network structure but are considered during the design of the loss function. For this problem, we implemented a simple artificial neural network, which consists three hidden layers with 64 nodes each and an output layer. The neural network takes a value 'x' in one-dimension as an input and outputs the neutron flux.

If the neural network is trained to minimize the error between true and estimated neutron fluxes, also called the fitting loss, it is just an ANN. To embed the physics information in neutronics analysis, we implemented the diffusion equation and two boundary conditions as loss functions. In PINN, the neural network is trained to minimize three loss functions: fitting loss, equation loss, and boundary condition losses. Fig. 2 depicts the difference between ANN and PINN.

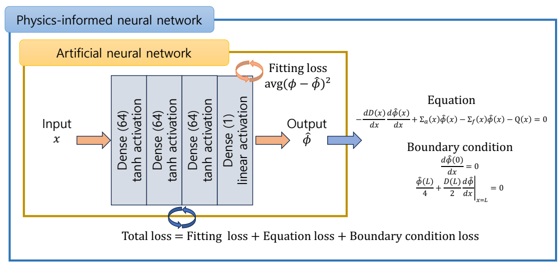
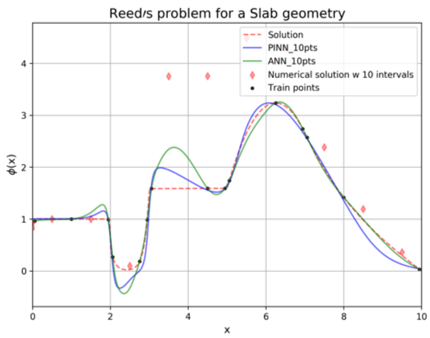


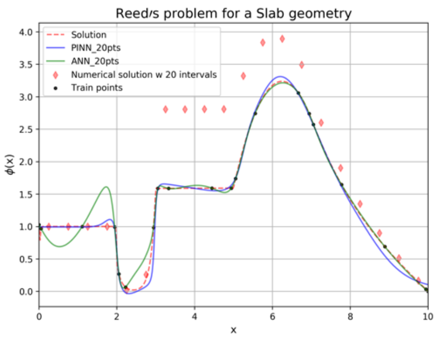
Fig. 2 Physics-informed neural network for neutron diffusion simulation

4. Experimental results

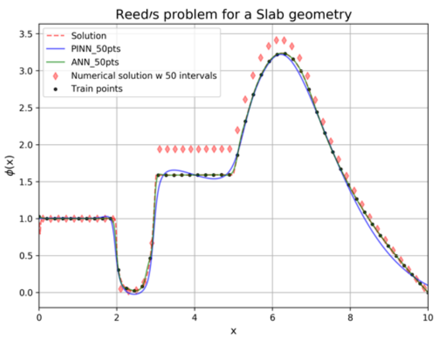
To test one of the benefits of the PINN, its strong performance with a small dataset, we trained both ANN and PINN with 10, 20, and 50 datapoints. Each network was implemented in the Python environment with the Tensorflow library [3]. Each network was trained during 50,000 epochs, and the model whose loss was the minimum was used as final model. Fig. 3 depicts performance comparison results for the numerical solver, ANN, and PINN. For the numerical solver, intervals of numerical analysis were set to the number of datapoints used to train neural networks. As shown in Fig. 3, numerical solver showed the poorest results in all cases. PINN showed better results than ANN with small training points (see in Fig. 3 (a) and (b)). However, PINN showed slightly poorer results than ANN, when trained with 50 data points (see in Fig. 3 (c)).



(a)



(b)



(c)

Fig. 3 Experimental results of numerical solver, ANN, and PINN

5. Conclusion

In this study, we developed a PINN and compared its performance with numerical solver and ANN for a neutron transport problem. Experimental results showed that the PINN performed better than the ANN with a small dataset, but its performance decreased than the ANN as the dataset size increased. This may be due to one of the drawbacks of the PINN, having poor performance on the discontinuity, which could be improved through the further researches.

Acknowledgements

This research was supported by a grant from the Korea Atomic Energy Research Institute (KAERI) R&D program (No. KAERI-524540-24).

REFERENCES

[1] R. G. McClarren, Computational nuclear engineering and radiological science using python, Academic press, pp. 340-342, 2018.
[2] W. H. Reed, New difference schemes for the neutron transport equation, Nuclear Science and Engineering, Vol. 46, No. 2, pp. 309-314, 1971.
[3] Abadi, Martín, et al. "{TensorFlow}: a system for {Large-Scale} machine learning." 12th USENIX symposium on operating systems design and implementation (OSDI 16). 2016.