Nuclear Science and Engineering

ISSN: (Print) (Online) Journal homepage: https://www.tandfonline.com/loi/unse20

Physics-Informed Neural Network Method and Application to Nuclear Reactor Calculations: A Pilot Study

Mohamed H. Elhareef & Zeyun Wu

To cite this article: Mohamed H. Elhareef & Zeyun Wu (2023) Physics-Informed Neural Network Method and Application to Nuclear Reactor Calculations: A Pilot Study, Nuclear Science and Engineering, 197:4, 601-622, DOI: 10.1080/00295639.2022.2123211

To link to this article: https://doi.org/10.1080/00295639.2022.2123211

NUCLEAR SCIENCE AND ENGINEERING · VOLUME 197 · 601–622 · APRIL 2023

DOI: https://doi.org/10.1080/00295639.2022.2123211

Physics-Informed Neural Network Method and Application to Nuclear Reactor Calculations: A Pilot Study

Mohamed H. Elhareef[a,b] and Zeyun Wu[a]*

[a]*Virginia Commonwealth University, Department of Mechanical and Nuclear Engineering, 401 West Main Street, Richmond, Virginia 23284-3015*
[b]*Alexandria University, Department of Nuclear and Radiation Engineering, Faculty of Engineering, Alexandria 11432, Egypt*

Received June 10, 2022
Accepted for Publication September 6, 2022

Abstract — *In this paper, the physics-informed neural network (PINN) method is investigated and applied to nuclear reactor physics calculations with neutron diffusion models. The reactor problems were introduced with both fixed-source and eigenvalue modes. For the fixed-source problem, the loosely coupled reactor model was solved with the forward PINN approach, and then, the model was used to optimize the neural network hyperparameters. For the k-eigenvalue problem, which is unique for reactor calculations, the forward PINN approach was modified to expand the capability of solving for both the fundamental eigenvalue and the associated eigenfunction. This was achieved by using a free learnable parameter to approximate the eigenvalue and a novel regularization technique to exclude null solutions from the PINN framework. Both single-energy-group and multiple-energy-group diffusion models were examined in the work to demonstrate the PINN capabilities of solving systems of coupled partial differential equations in reactor problems. A series of numerical examples was tested to demonstrate the viability of the PINN approach. The PINN solution was compared against the finite element method solution for the neutron flux and the power iteration solution for the k-eigenvalue. The error in the predicted flux ranged from 0.63% for simple fixed-source problems up to about 15% for the two-group k-eigenvalue problem. The deviations in the predicted k-eigenvalues from the power iteration solver ranged from 0.13% to 0.92%. These generally acceptable results preliminarily justified the feasibility of PINN applications in reactor problems. The advantageous application potentials as well as the observable computational deficits of the PINN approaches are discussed along with the pilot study.*

Keywords — *Physics-informed neural network, deep learning, multigroup neutron diffusion equation, k-eigenvalue diffusion model.*

Note — *Some figures may be in color only in the electronic version.*

I. INTRODUCTION

Modeling a nuclear reactor involves solving a system of partial differential equations (PDEs) that describes the various phenomena in the reactor core. There is no universal PDE solver that can be applied to all design and analysis purposes. Conventional numerical methods [e.g., the finite element method (FEM)] for solving the PDEs are commonly used thanks to their rigorous numerical performance. Nevertheless, these methods more or less suffer some drawbacks including computation complexity, intense manpower efforts, need of prior assumptions, and so on. On the other hand, data-driven modeling techniques can be computationally effective and relatively easy to implement. However, the performance of data-driven methods relies on the quantity and quality of the available data, which is always a major challenge for disciplines where the cost of data acquisition is prohibitive.

*E-mail: zwu@vcu.edu

Theory guided data science (TGDS) is an emerging paradigm in data science that aims to provide an alternative framework for embedding scientific knowledge (e.g., PDEs) in data-driven approaches.[1] Integrating scientific knowledge with data-driven techniques enables a new scientific prediction paradigm that allows the generation of physically consistent models while reducing the demand for training data. This paradigm fills the gap between well-established theoretical approaches and data-driven approaches. Within the paradigm, several efforts have been made by different research groups to develop frameworks for data-driven solving and data-driven discovery of PDEs (Refs. 2 through 7).

Compared to the "hard constraint" physics-informed neural network (PINN) framework introduced by Lagaris et al.[2] years ago, the "soft constraint" PINN framework recently developed by Raissi et al.[3] has lately gained wider recognition in the TGDS community. Simply put, the PINN approach uses a neural network (NN) to approximate the solution of a PDE that can be obtained by training the NN model to fit the training data (e.g., boundary conditions) while imposing the PDE model on the model predictions. The PINN framework is applicable for both forward and inverse PDE-based problems. This framework relies on recent enhancements in the Machine Learning (ML) toolbox, in particular, automatic differentiation (AD), to directly tackle general nonlinear PDEs without the need for prior assumptions or domain discretization.

Generally, NNs can be considered as differentiable universal function approximators, which implies a large number of computational devices (neurons) arranged in successive layers.[8] In NN models, a weighted sum of the output of each layer is transferred through an activation function to the next layer. The NN model can be trained to map its input parameters to the target outputs with an arbitrary accuracy desired by adjusting the weights and biases (learnable parameters) of the weighted sum operation. The training process involves iteratively transferring the training inputs through the NN to obtain the outputs, which are then compared to the target outputs to compute the cost function (loss function). Loss functions are differentiable functions that measure the mismatch between the predicted outputs and the target outputs. The values of the loss function and its derivative with respect to the learnable parameters are used by means of gradient decent to update the weights and biases in the direction that minimizes the loss function. The training process terminates when a predefined criterion is satisfied. In the PINN approach, a physics-informed component is added to the loss function. The construction of this component is done by first differentiating the NN model with respect to its input parameters according to the PDE model. The predictions of the physics-informed component (residuals of the PDE) at some collocation points are used as training data to construct the loss function. Minimizing the physics-informed component of the loss function enforces the predictions to satisfy the PDE model.

In their original work, Raissi et al.[3] solved two main classes, namely, data-driven solution (forward problem) and data-driven discovery (inverse problem) of PDEs. The forward problem is solved by training on a data set of the known data (e.g., boundary and/or initial conditions) and a data set of collocation points sampled from the solution domain to evaluate the PDE model residuals. In the inverse problem, the unknown parameters in a parameterized PDE model are extracted from the known solution by training on a data set sampled from the known solution. For each of the two classes, they devised two distinct solution schemes: continuous time domain and discrete time domain with implicit Runge-Kutta time stepping schemes.

Since the introduction, many continued efforts have been made on the subject to resolve various issues encountered in the PINN implementations, such as computational complexity and uncertainty quantification, and to extend the method to various domains. Variations of the conventional PINN approach include probabilistic PINN for uncertainty quantification,[9] arbitrary polynomial chaos PINN for solving stochastic differential equations,[10] conservative PINN for discrete domains,[11] parareal PINN for long-time integration,[12] nonlocal PINN for integral equations,[13] and Bayesian PINN for large-noise data scenarios and uncertainty quantification.[14] Various methods have been proposed to reduce the computational complexity of the PINN approach including adaptive activation functions,[15] transfer learning,[16] variational PINN with domain decomposition,[17] and extreme theory of functional connections.[18]

On the application side, the PINN approach thus far has been successfully applied to various science and engineering disciplines including fluid dynamics,[3,19–21] quantum mechanics,[3] cardiac activation mapping,[22] diffusion systems,[10,23–25] nano-optics and metamaterials,[26] material mechanics,[27–29] power systems,[30] heat transfer,[31,32] three-dimensional surfaces,[33] subsurface transport,[34] biological tissues,[35] chemical kinetics,[36] and so on.

Recently, several publications reported applications of physics-guided data science in nuclear engineering.[37–41] However, to the authors' knowledge, at the time of the

writing this paper, the PINN has been rarely applied to reactor physics calculations besides the preliminary work reported by the same research group in recent academic conferences.[23–25] Very recently, Xie et al.[42] used the PINN to solve the two-dimensional (2D), one-energy-group, and time-dependent diffusion equation for reactors with square and circular geometries. Instead of focusing on the physics model and methodology, Xie et al.'s work was based on the boundary conditions and proposed different approaches for treating the boundary conditions, namely, the boundary dependent method (BDM) and the boundary independent method (BIM). The BDM reached two orders of magnitude higher accuracy compared to the BIM. In the BDM, a trial function is used to give a continuous-symbolic solution of the boundary conditions. They also addressed the effect of various hyperparameters and activation functions on the solution accuracy.

It is well known that the principal model for reactor core level calculations is multigroup neutron diffusion equations, which essentially can be considered as a set of second-order PDEs. In this paper, we make efforts to carry out a pilot type of study to assess the feasibility of PINN applications in neutronics calculations based on multigroup diffusion models. In our previous work,[23–25] we preliminarily demonstrated the applicability of the PINN approach for reactor physics problems. We started with steady-state, 2D, one-group, fixed-source neutron diffusion models[23] and then extended the approach to two-group (2-G) fixed-source[24] and *k*-eigenvalue problems,[25] respectively. This paper extends these previous efforts and gives a systematic presentation of the PINN method to the reactor problems in both fixed-source and *k*-eigenvalue modes. For the fixed-source problem, the loosely coupled reactor model (LCRM) was solved with the forward PINN approach, and then, the model was used to optimize the NN hyperparameters. For the *k*-eigenvalue problem, which is unique for reactor calculations, the forward PINN approach was modified to expand the capability of solving for both the fundamental eigenvalue and the associated eigenfunction. This was achieved by using a free learnable parameter to approximate the eigenvalue and a novel regularization technique to exclude null solutions from the PINN framework. Both single-energy-group and multiple-energy-group diffusion models were examined in the work to demonstrate the PINN capabilities of solving systems of coupled PDEs in reactor problems. A series of numerical examples was tested to demonstrate the viability of the PINN approach. The PINN solution was compared against the FEM solution for the neutron flux and the power iteration solution for the *k*-eigenvalue. This paper thoroughly discusses the advantageous application potentials and the observable computational deficits of the PINN approaches in reactor calculations as well as the pilot investigation.

The remainder of the paper is organized as follows. Section II formulates the theoretical foundations of the forward PINN framework and establishes a clear pathway to connect the PINN approach to the neutron diffusion models. Sections III and IV demonstrate the PINN applications with a series of reactor physics test problems based on neutron diffusion theory in both fixed-source and *k*-eigenvalue modes, respectively. These test problems are designed purposely from the simplest to the most difficult levels. Both single-energy-group and multiple-energy-group models are considered. The last section (Sec. V) offers some conclusive remarks and future perspectives to the PINN approach on the reactor physics applications based on the outcome of the conducted research.

II. PINN METHODOLOGY AND NEUTRON DIFFUSION MODELS

In the original paper of Raissi et al.,[3] the PINN method is classified into two distinct categories: the forward PINN method and the inverse PINN method. The first category forms a new family of data-efficient solution approximators for PDEs while the second category allows the use of solution approximators to enable a data-driven discovery of the PDEs. For most reactor physics problems, the solution approximators are of priori interest; thus, this work focuses only on the forward PINN method in this paper, and the inverse PINN is left for future interests.

II.A. Forward PINN Approach

The forward PINN approach develops a NN model to approximate the unknown state function that satisfies the PDE model and any known values of the function. Consider a general nonlinear PDE model as follows:

$$F := \mathbb{N}(Y(x_1, x_2, \cdots, x_n)) = 0\,, \tag{1}$$

where

$x_1, x_2, \cdots, x_n$ = independent variables

$Y(x_1, x_2, \cdots, x_n)$ = desired state function that satisfies the PDE model

$\mathbb{N}$ = generic nonlinear differential operator

F = PDE residuals.

In this regard, Eq. (1) can be considered as a residual formulation of the PDE model.

Within the forward PINN framework, the state function $Y(x_1, x_2, \cdots, x_n)$ is approximated by a NN $net\_Y(x_1, x_2, \cdots, x_n)$ that takes independent variables $(x_1, x_2, \cdots, x_n)$ as the input vector and predicts the state value at this vector. The NN model is differentiable with respect to its input variables. Thus, the residuals of the PDE can also be approximated to a NN $net\_F(x_1, x_2, \cdots, x_n)$ by differentiating $net\_Y(x_1, x_2, \cdots, x_n)$ according to the original PDE form, namely,

$$net\_F := \mathbb{N}(net\_Y(x_1, x_2, \cdots, x_n)). \tag{2}$$

These two NN models, net_Y and net_F, share the same learnable parameters. These shared learnable parameters can be optimized to meet two goals: Reproduce the known values of the state function, and minimize the predicted residuals. The optimization goals of the learnable parameters can be accomplished by minimizing a loss function that penalizes the model predictions. The loss function generally consists of two terms: One term accounts for the mismatch between the predictions and the known data (explicit values of the state function), and a second term accounts for the predicted residuals $net\_F(x_1, x_2, \cdots, x_n)$. Thus, the physics-informed loss function can be defined as

$$Loss = \frac{1}{N_b}\sum_{i=1}^{N_b}\left[net\_Y\left(x_1^i, x_2^i, \ldots x_n^i\right) - Y\left(x_1^i, x_2^i, \ldots x_n^i\right)\right]^2 + \frac{1}{N_f}\sum_{j=1}^{N_f}\left[net\_F\left(x_1^j, x_2^j, \ldots x_n^j\right)\right]^2, \tag{3}$$

where the first term represents the mean-squared error (MSE) between the model predictions and the target values for a set of N_b training points at which the target values are explicitly known (e.g., the boundary or initial conditions of the PDE model). The second term in the loss function represents the mean-squared value of the predicted residuals for a set of N_f training points randomly sampled from the solution domain. For a general case in which the boundary conditions are given in terms of the state function and its derivatives (e.g., Neumann boundary condition), the contribution to the first loss term will be shown in a similar manner to the residual loss. In this case, the loss function can be defined as

$$Loss = \frac{1}{N_b}\sum_{i=1}^{N_b}\left[net\_B\left(x_1^j, x_2^j, \ldots x_n^j\right)\right]^2 + \frac{1}{N_f}\sum_{j=1}^{N_f}\left[net\_F\left(x_1^j, x_2^j, \ldots x_n^j\right)\right]^2, \tag{4}$$

where net_B is the NN predictions for the boundary conditions.

Figure 1 depicts the training scheme of the forward PINN approach. It also explains the numerical implementation of the forward PINN approach. The training scheme starts by defining a training data set for each term in the loss function to be evaluated. The NN learnable parameters (weights and biases) are randomly initialized at the starting point. Each training data set is then transferred to the NN model to evaluate the corresponding residuals function and add it to the total loss. An optimization algorithm [e.g., L-BFGS (Ref. 43)] is used to iteratively update the NN learnable parameters until

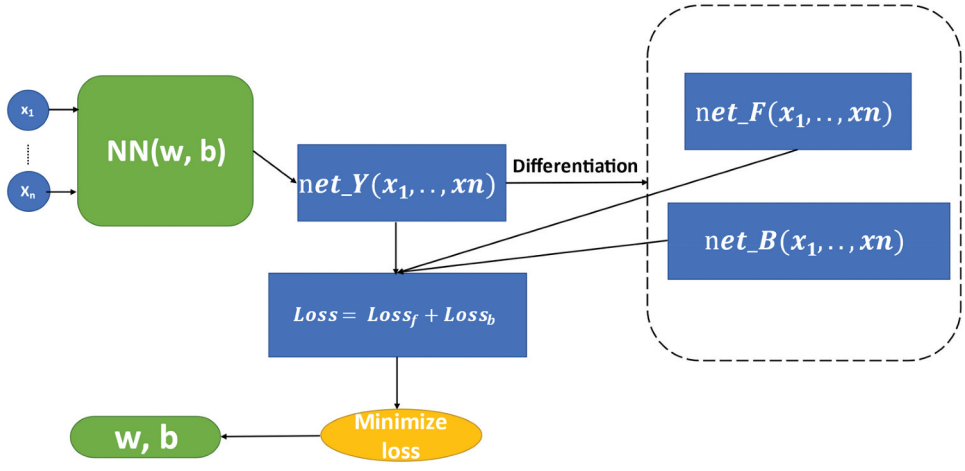

Fig. 1. Training scheme for the forward PINN approach.

the prescribed convergence criteria are achieved to output the satisfactory predictions.

II.B. Fixed-Source Diffusion Model

To apply the PINN to reactor problems, the neutron diffusion model is used for demonstration. We focus on fixed-source problems in this section and extend to *k*-eigenvalue models later. Without loss of generality, the PINN methodology is illustrated with a simplified one-energy-group 2D diffusion equation described by

$$-\left[\frac{\partial}{\partial x}\left(D(x,y)\frac{\partial\phi}{\partial x}\right)+\frac{\partial}{\partial y}\left(D(x,y)\frac{\partial\phi}{\partial y}\right)\right]+\Sigma_a(x,y)\phi(x,y)-S(x,y)=0, \tag{5}$$

where S is the distributed external source, $\Sigma_a$is the macroscopic absorption cross section, and D is the diffusion coefficient given by

$$D=\frac{1}{3(\Sigma_a+\Sigma_s)}, \tag{6}$$

where Σ_s is the macroscopic scattering cross section.

To apply the PINN approach to the fixed-source diffusion problem, we first construct a NN model to approximate the flux distribution of the diffusion equation: $net\_\phi(x,y)\approx\phi(x,y)$. Relying on AD, $net_\phi(x,y)$is then differentiated based on the PDE model to construct the residuals NN $net\_f(x,y)$ as

$$net\_f(x,y)=-\left[\frac{\partial}{\partial x}\left(D\frac{\partial net\_\phi}{\partial x}\right)+\frac{\partial}{\partial y}\left(D\frac{\partial net\_\phi}{\partial y}\right)\right]+\Sigma_a net\_\phi(x,y)-S(x,y). \tag{7}$$

Similarly, a NN approximator is constructed to the boundary conditions at each surface of the system. For instance,

$$net\_B(x,y)=\frac{1}{4}net\_\phi(x,y)-\frac{1}{2}D\frac{dnet\_\phi}{dx}, \tag{8}$$

where a generic Robin type of boundary condition is assumed at the boundary surface. Finally, the shared learnable parameters of the NN models are trained to minimize the loss function:

$$Loss=\frac{1}{N_f}\sum_{i=1}^{N_f}\left[net\_f(x_i,y_i)\right]^2+\frac{1}{N_b}\sum_{j=1}^{N_b}\left[net\_B(x_j,y_j)\right]^2+\cdots. \tag{9}$$

The additional terms in the loss function are constructed to approximate the boundary conditions at the remaining surfaces. Each of the terms in the loss function is evaluated using a set of training points sampled from the corresponding domain. All training sets can be generated using the Latin hypercube sampling (LHS) strategy.[44,45]

II.C. PINN Approach in Eigenvalue Problems

Eigenvalue problems are quite special and very important for reactor calculations. For eigenvalue problems, the eigenfunction of a general linear differential operator $\mathcal{D}$ may be defined by the nontrivial solution of Eq. (10):

$$\mathcal{D}Y(x_1,x_2,\cdots,x_n)=\lambda Y(x_1,x_2,\cdots,x_n), \tag{10}$$

where λ is a scaler. The problem of finding the principal eigenvalue and the associated eigenfunction can be solved in the PINN framework by considering the residuals equation:

$$F:=\mathcal{D}(Y(x_1,x_2,\cdots,x_n))-\lambda Y(x_1,x_2,\cdots,x_n)=0. \tag{11}$$

Equation (11) is a parameterized PDE with an unknown parameter λ. The principal solution of this equation $(Y(x_1,x_2,\cdots,x_n)|\lambda_1)$ can be obtained by approximating the eigenfunction by a NN model that is restricted to predict nontrivial solutions and approximates the principal eigenvalue with a free learnable parameter that can be learned from minimizing the PDE residuals. Although in theory this approach can converge to any of the eigensolutions of the system, the gradient decent optimization most likely converges to the fundamental mode. This is due to the spectral bias in the fully connected NNs making these models partially incapable of learning high-frequency solutions.[46]

One unique challenge posed to the PINN application to eigenvalue problems is that the obtained solution approximators are prone to a trivial solution. In order to eliminate convergence to a trivial solution, we introduce an additional regularization term to the loss function of the original forward PINN approach. The additional regularization term is an integral term that enforces a predefined value

for the integration of the eigenfunction over its input space. The predefined value of the integration can be directly related to some physical quantity of interest, or it can be simply taken to be one. In the latter case, the predicted eigenfunction is normalized over the solution domain. The condition for a nontrivial solution can be written as

$$\ldots \; Y(x_1, \cdots, x_n) dx_1 \ldots dx_n = C \,, \tag{12}$$

where C is the predefined value for the integration. The regularization term can be defined as the squared difference between the mean of the predicted values at the training points and the target mean as follows:

$$R = \left[\frac{1}{N_f} \left(\sum_{i=1}^{N_f} net\_Y\left(x_1^i, x_2^i, \ldots x_n^i\right) \right) - \frac{C}{N_f} \right]^2 . \tag{13}$$

The total loss in the eigenvalue PINN approach is defined by

$$Loss = Loss_f + Loss_b + R. \tag{14}$$

The training scheme for the PINN approach in eigenvalue problems is shown in Fig. 2.

II.D. *k*-Eigenvalue Diffusion Model

For illustration, the one-group, 2D, *k*-eigenvalue diffusion model is given as

$$-\left[\frac{\partial}{\partial x}\left(D\frac{\partial \phi}{\partial x}\right) + \frac{\partial}{\partial y}\left(D\frac{\partial \phi}{\partial y}\right)\right] + \Sigma_a(x,y)\phi(x,y) = \frac{1}{k} \nu \Sigma_f(x,y)\phi(x,y) \,, \tag{15}$$

where Σ_f is the macroscopic fission cross section and ν is the average number of neutrons emitted per fission. All other terms are similar to the ones in Eq. (5).

The most well-known numerical scheme to solve for the principal *k*-eigenvalue in the equation is the power iteration method.[47] In this approach, the eigenvalue problem is literally reduced to the fixed-source problem that is solved iteratively by updating the source term in each iteration. First, an initial guess of the flux distribution and k is used to calculate the source term. The source term is then used as a fixed term in the diffusion model, which is to be solved for the flux distribution. The calculated flux distribution is then used to estimate the value of k and to calculate the source term for the next iteration. This process terminates after a convergence criterion is achieved. A flow diagram of the power iteration scheme applied to the *k*-eigenvalue diffusion problem is shown in Fig. 3. For the sake of verifying the PINN solution for this problem, the power iteration scheme was implemented and used the FEM integrated in COMSOL Multiphysics[48] as the flux solver.

The implementation of the PINN approach for *k*-eigenvalue problems starts by defining the NN models similarly as discussed earlier. Here, we assume

$$net\_\phi(x,y) \approx \phi(x,y) \tag{16}$$

and

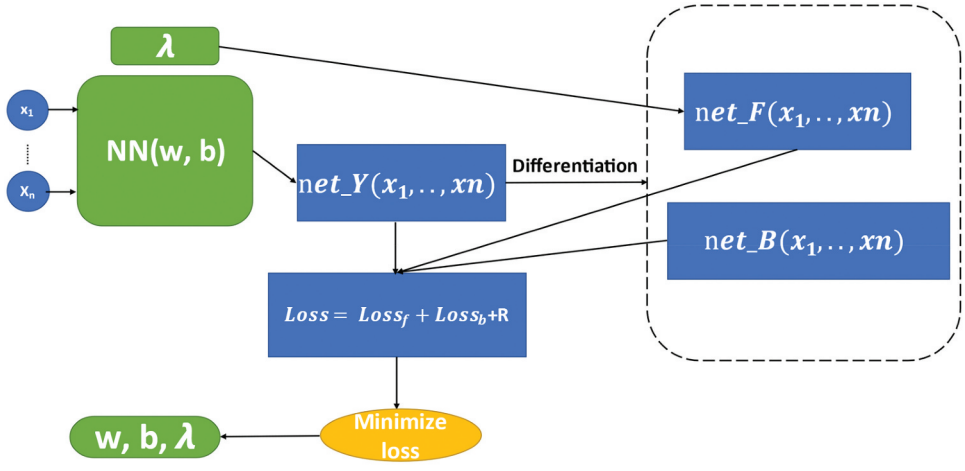

Fig. 2. Training schematic of PINN for the principal solution of eigenvalue problems.

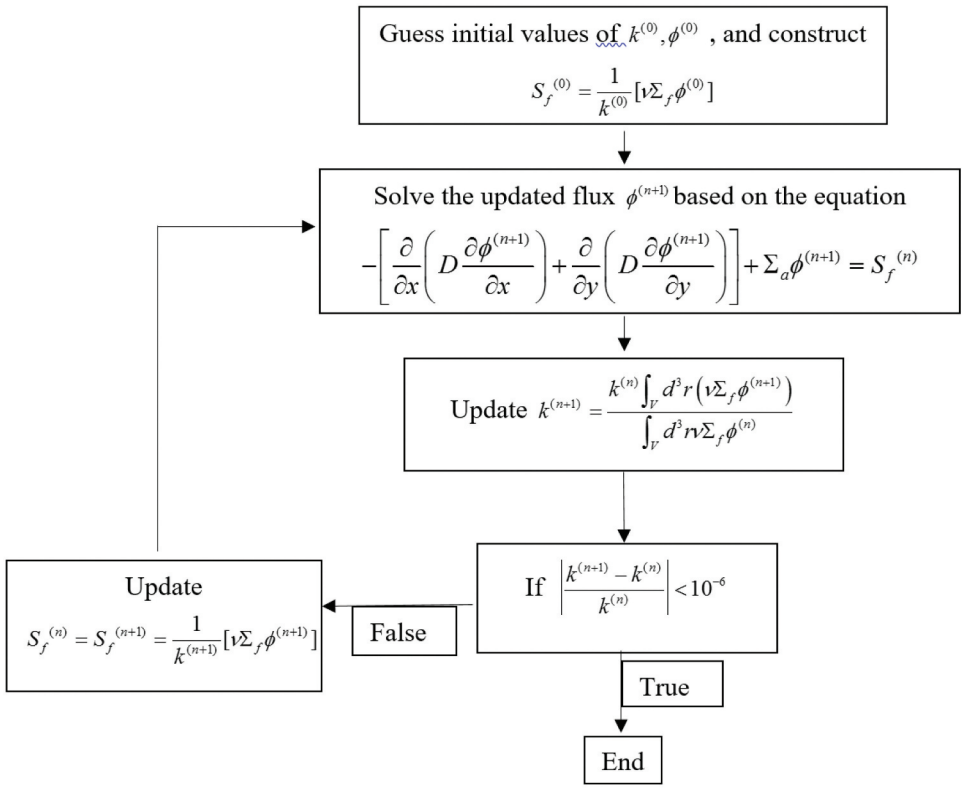

Fig. 3. Power iteration scheme for solving the *k*-eigenvalue diffusion problem.

$$net\_f(x,y) = -\left[\frac{\partial}{\partial x}\left(D\frac{\partial net\_\phi}{\partial x}\right)+\frac{\partial}{\partial y}\left(D\frac{\partial net\_\phi}{\partial y}\right)\right] + \Sigma_a net\_\phi(x,y) - \frac{1}{k}\nu\Sigma_f net\_\phi(x,y). \quad (17)$$

The NN models for the boundary conditions are identical to the fixed-source case while the regularization term is defined by

$$R = \left[\frac{1}{N_f}\left(\sum_{i=1}^{N_f} net\_\phi(x_i,y_i)\right) - \frac{C}{N_f}\right]^2. \quad (18)$$

The shared learnable parameters of the NN models are learned by minimizing the loss function defined by

$$Loss = \frac{1}{N_f}\sum_{i=1}^{N_f}[net\_f(x_i,y_i)]^2 + \frac{1}{N_b}\sum_{j=1}^{N_b}\left[net\_B(x_j,y_j)\right]^2 + R. \quad (19)$$

II.E. PINN Approach in Multigroup Diffusion Models

In this section, the PINN approach is extended to multigroup diffusion models. For illustration, we consider the 2-G, *k*-eigenvalue diffusion equations defined by

$$\begin{cases} -\ (D_1\ \ \phi_1) + \Sigma_{r,1}\phi_1 = \frac{1}{k}\left(\nu\Sigma_{f,1}\phi_1 + \nu\Sigma_{f,2}\phi_2\right) \\ -\ (D_2\ \ \phi_2) + \Sigma_{a,2}\phi_2 = \Sigma_{s,1\to 2}\phi_1 \end{cases}, \quad (20)$$

where

ϕ_g, D_g, $\Sigma_{a,g}$, $\Sigma_{f,g}(g = 1, 2)$ = flux, diffusion coefficient, macroscopic absorption cross section, and macroscopic fission cross section of the group g neutrons, respectively

$\Sigma_{s,1\to 2}$ = macroscopic downscattering cross section from the fast group neutrons to the thermal group neutrons

$\Sigma_{r,1} = \Sigma_{a,1} + \Sigma_{s,1\to 2}$ = removal cross section for the fast group neutrons.

The PINN approach can be readily applied to the 2-G diffusion problems by using a NN model with two outputs, with each output representing one component of the flux. This NN model can then be used to construct the residuals of each group equation, and a loss function that accounts for the magnitude of the combined residuals can be used as a physical constraint on the NN predictions in the training phase. Moreover, for the k-eigenvalue problems, the fission rate can be used as a regularization term to avoid the trivial solution for the homogeneous PDEs. For the 2-G case, the total fission rate is defined by

$$fission\ rate = \int_V \left(\nu\Sigma_{f,1}\phi_1 + \nu\Sigma_{f,2}\phi_2\right) dV. \tag{21}$$

By using a predefined value of the fission rate, and enforcing this value on the model predictions, the NN model is restricted to predict the nonzero-flux distribution. The solution can be learned by minimizing the following loss function:

$$Loss = \sum_{i=1}^{N_f} \left|\begin{matrix} f_1 \\ f_2 \end{matrix}\right|_i^2 + \sum_{k=1}^{N_b} \left|\begin{matrix} f_{B1} \\ f_{B2} \end{matrix}\right|_k^2 + \left[\sum_{j=1}^{N_i} \left(\nu\Sigma_{f,1}\phi_1 + \nu\Sigma_{f,2}\phi_2\right)_j - C\right]^2, \tag{22}$$

where

ϕ_1, ϕ_2, f_1, f_2 = NN predictions of fast flux, thermal flux, fast flux equation residuals, and thermal flux equation residuals

C = predefined value of fission rate

f_B = NN predictions of the boundary conditions.

It is worth noting that we have defined the loss function as the sum of residuals instead of the mean of residuals. Although this change in the way of defining the loss function does not essentially affect the final outcome of the learning process, we find that it accelerates the convergence of the solution in all test cases. This is due to the larger weight put on the residuals of the PDE model compared to the other components of the loss function (i.e., the boundary conditions and the regularization term).

In the following sections, a series of numerical examples is provided to cover problems of interest in reactor physics arranged in ascending order of problem complexity. Figure 4 arranges the problems discussed in this work into four categories showing the significance of each example as a demonstration of PINN applicability to reactor physics problems.

III. ONE-GROUP EXAMPLES

The one-energy-group examples discussed in this section were presented at recent American Nuclear

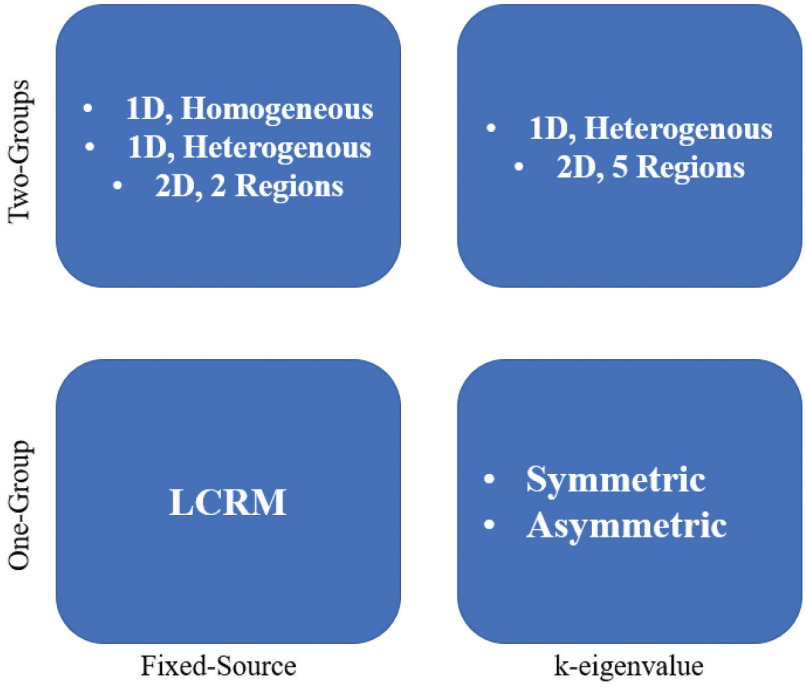

Fig. 4. Summary of the numerical demonstration of PINN for reactor physics problems.

Society–sponsored topical meetings.[23,25] We briefly repeat the results here to offer a complete story of PINN applications. After all, the one-group example can serve as a leading case to facilitate the PINN extension to the multigroup examples that will be discussed in Sec. IV. Because of the distinct nature of the governing equations, both in this section and in Sec. IV, the numerical examples are categorized into two classes: fixed-source problems and *k*-eigenvalue problems.

III.A. Fixed-Source Problems

For the fixed-source case, the LCRM problem described in the work of Rokrok et al.[49] is solved. The configuration of the LCRM is shown in Fig. 5. The material properties of the essentially two regions are given in Table I.

Zero-incoming fluxes are assumed for all boundary surfaces of the problem, which can be expressed as a Robin type of boundary condition as follows:

At the surface x = 0:

$$\frac{1}{4}\phi(0,y) - \frac{1}{2}D\frac{d\phi}{dx}\bigg|_{x=0} = 0\,; \tag{23a}$$

At the surface x = 100:

$$\frac{1}{4}\phi(100,y) + \frac{1}{2}D\frac{d\phi}{dx}\bigg|_{x=100} = 0\,; \tag{23b}$$

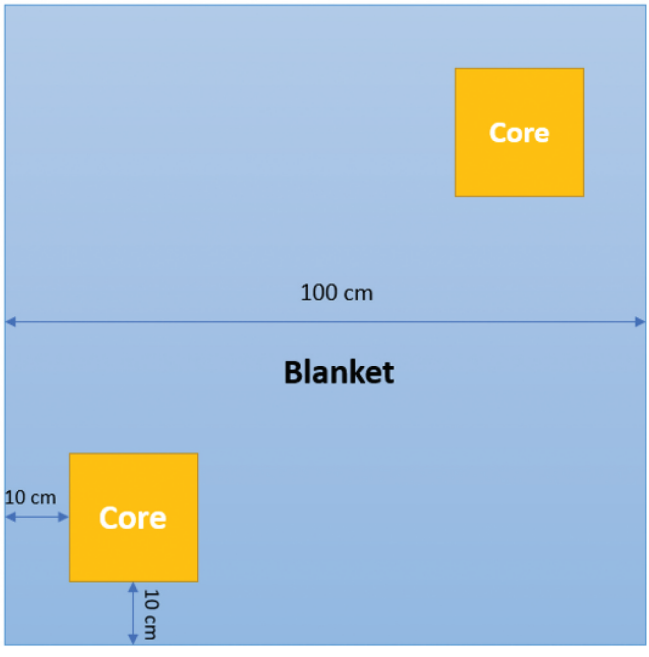

Fig. 5. Geometry of the LCRM problem.

TABLE I

Material Properties of the LCRM

Region Material	Σ_a (cm^{-1})	Σ_s (cm^{-1})	S (n/cm^3)
Core	0.062158	0.089302	0.01048083
Blanket	0.064256	0.094853	0.00214231

At the surface y = 0:

$$\frac{1}{4}\phi(x,0) - \frac{1}{2}D\frac{d\phi}{dy}\bigg|_{y=0} = 0\,; \tag{23c}$$

At the surface y = 100:

$$\frac{1}{4}\phi(x,100) + \frac{1}{2}D\frac{d\phi}{dy}\bigg|_{y=100} = 0. \tag{23d}$$

To solve the LCRM problem using the PINN approach, we use a NN model to approximate the flux distribution: $net\_\phi(x,y) \approx \phi(x,y)$. Then, AD is used to differentiate $net\_\phi(x,y)$ according to the PDE model to construct the residuals NN $net\_f(x,y)$:

$$net\_f(x,y) = -\left[\frac{\partial}{\partial x}\left(D\frac{\partial net\_\phi}{\partial x}\right) + \frac{\partial}{\partial y}\left(D\frac{\partial net\_\phi}{\partial y}\right)\right] + \Sigma_a net\_\phi(x,y) - S. \tag{24}$$

Similarly, we construct four functions, $net\_B_L(x,y)$, $net\_B_R(x,y)$, $net\_B_B(x,y)$, and $net\_B_T(x,y)$, to evaluate the NN predictions for the boundary conditions at the system surfaces, where the subscripts *L*, *R*, *B*, and *T* refer to the surfaces: left, right, bottom, and top, respectively. For instance,

$$net\_B_L(x,y) = \frac{1}{4}net\_\phi(x,y) - \frac{1}{2}D\frac{dnet\_\phi}{dx}. \tag{25}$$

The shared learnable parameters of the NN models are trained to minimize the loss function:

$$Loss = \frac{1}{N_f}\sum_{i=1}^{N_f}\left[net\_f(x_i,y_i)\right]^2 + \frac{1}{N_b}\sum_{j=1}^{N_b}\left[net\_B_L(x_j,y_j)\right]^2 + \cdots. \tag{26}$$

Each of the five terms in the loss function was evaluated using a set of training points sampled from the corresponding domain. All training sets were generated using the LHS strategy.

For the application of the PINN to the fixed-source example described in Sec. II.B, the tanh function is used as the activation function in the NN structure. The NN learnable parameters were trained using the Adam optimizer[50] for a fixed number of iterations (10^5), and then, the L-BFGS (Ref. 43) algorithm was used to complete the training until the convergence criterion (the maximum component of the loss function gradient is ≤1E-11). TensorFlow 1.0 (Ref. 51) software was used to implement all models. We used a high-order FEM solution obtained using COMSOL Multiphysics[48] as the reference solution. The reference solution was averaged to a 100 × 100 grid to achieve the pointwise comparison with the PINN predictions.

A systematic parametric study was conducted to optimize the NN architecture (number of hidden layers, number of neurons per layer) and to understand the accuracy of the PINN predictions for different numbers of training points (N_f, N_b).

Table II shows the mean relative error in the PINN predictions compared with the reference solution for a fixed number of training points ($N_f = 10^4, N_b = 25$) and different NN architectures. Table III shows the mean relative error in the PINN predictions compared with the reference solution for the fixed architecture (8 hidden layers and 40 neurons per layer) and different numbers of collocation and boundary points.

The results of the systematic study showed that increasing the number of hidden layers and/or the number of neurons per layer increases the accuracy of the PINN predictions. This result was expected as increasing these parameters increases the approximation capacity of the NN model. Results also showed that the PINN prediction accuracy is stable over a wide range of training data volumes. Note that the parametric optimization procedure is a typical practice for any NN method related applications. Throughout this paper, only the parameter optimization procedure for this very first numerical was detailed as an illustration. For the rest of the examples, the optimization was also performed, but only the results with the best NN architecture were presented.

Based on the findings of the parametric optimization, the PINN model with 8 hidden layers and 40 neurons per layer was chosen for the LCRM problem (indicated by italic print in Table II). The chosen model was trained using 10 000 collocation points and 100 boundary points per surface (indicated by italic print in Table III). The mean relative error for this model is 0.69%, and the maximum error is 6.9%. Figure 6 shows the predicted flux $net\_\phi(x,y)$ and the relative percentage pointwise errors compared to the reference solutions obtained by the FEM. The convergence curve of the loss function for the selected PINN model is shown in Fig. 7.

III.B. *k*-Eigenvalue Problems

For the *k*-eigenvalue case, we solve the diffusion models with two different geometries that are variant from the LCRM. The geometric configurations of these two problems are shown in Fig. 8. The material properties of each region are given in Table IV. We again assume zero-incoming fluxes at all boundary surfaces of the problem.

For the application of the PINN to the *k*-eigenvalue example described in Sec. II.D, we used the same NN architecture, training data set volumes, and optimization algorithm as the ones in the fixed-source example. In the first numerical example, we solve for the configuration given in Fig. 8a. The predicted value of $k = 0.96266$, and the power iteration algorithm converged at $k = 0.96395$. The relative percentage error in the predicted value is0.13%. Figure 9 shows the predicted flux with the pointwise difference between

TABLE II

Mean Relative Error Between PINN Prediction and the Reference Solution for Different NN Architectures*

Layers \ Neurons	10	20	40
2	25.04	11.04	47.69
4	11.24	5.15	1.56
6	2.15	0.79	0.81
8	1.2	0.96	*0.73*

*Mean relative error is in units of percent. N_f = 10 000 and N_b = 25.

TABLE III

Mean Relative Error Between PINN Prediction and the Reference Solution for N_f and N_b with Fixed NN Architecture*

N_b \ N_f	2000	5000	10 000
25	1.06	0.72	0.73
50	0.95	1.04	0.72
100	1.39	0.82	*0.69*
300	1.13	0.76	0.84
1000	0.91	0.74	0.69

Mean relative error is in units of percent. Eight hidden layers and 40 neurons per layer.

the power iteration solution and the PINN predicted one. The mean absolute error (MAE) in the predicted flux is 2.9E-6. For the second numerical example, we solve for the configuration given in Fig. 8b. The predicted value of $k = 0.95894$, and the power iteration algorithm converged at$k = 0.96321$. The relative error in the predicted value is0.44%. Figure 10 shows the predicted flux with the pointwise difference between the power iteration solution and the PINN predicted flux. The lower peak flux at the top right region of the core is nearly invisible in the 2D view shown in Fig. 10a. The overall MAE in the predicted flux is ~1.2E-6. These results provide a preliminary justification of the successful implantation of the PINN approach for k-eigenvalue problems. Further justification is continued in Sec. IV by applying the PINN to multigroup diffusion problems.

IV. MULTIGROUP EXAMPLES

To demonstrate PINN applications in the multigroup case, several numerical examples are provided, covering both one-dimensional (1D) and 2D geometries, as well as

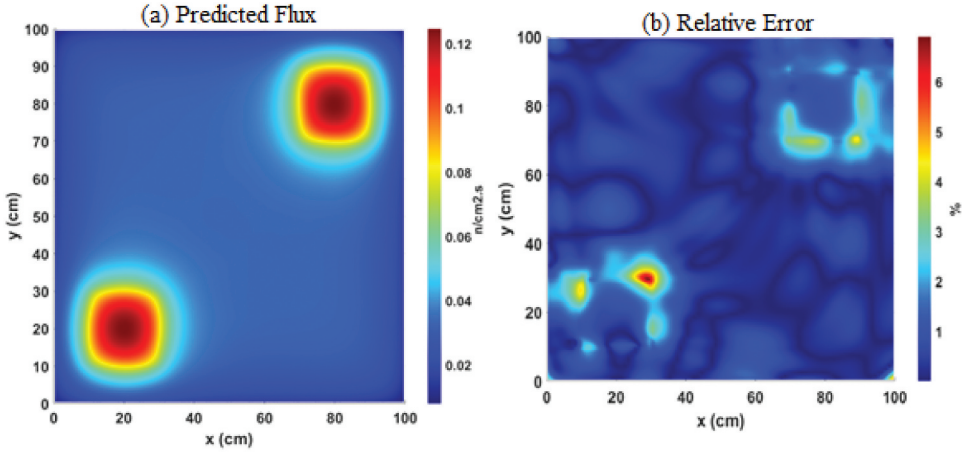

Fig. 6. Heat map view of the PINN predicted flux distribution (a) in whole domain and relative percentage error distribution compared to the FEM solution (b).

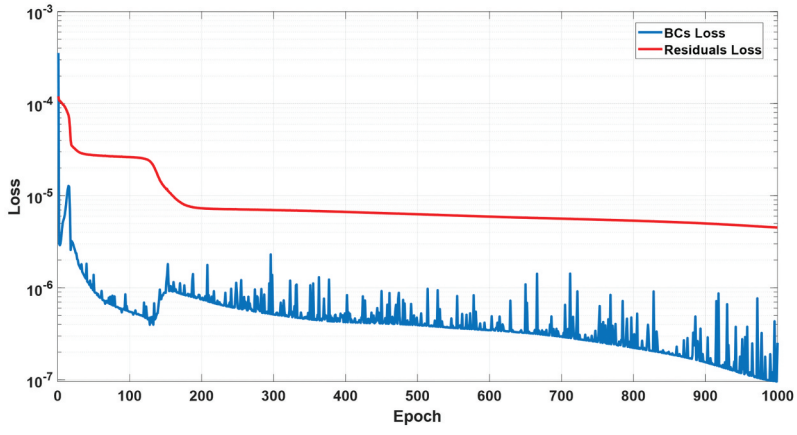

Fig. 7. Convergence curve of the PINN model for the LCRM.

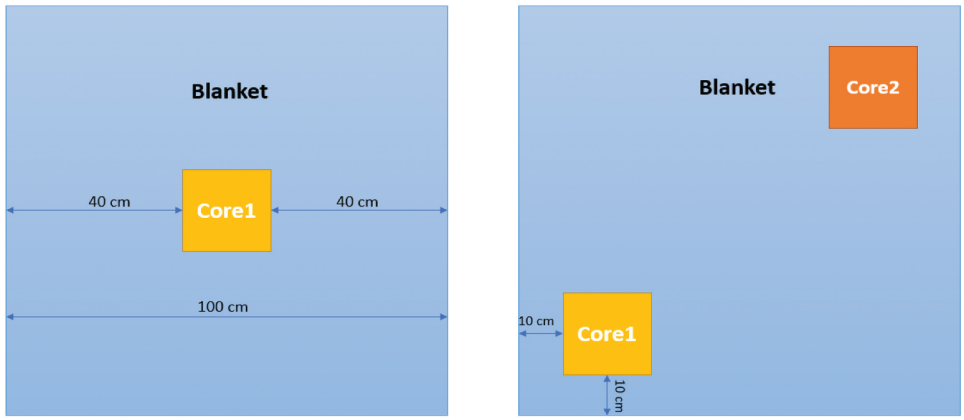

Fig. 8. Geometric configuration of the two k-eigenvalue numerical examples: (a) the configuration of Example 1 and (b) the configuration of Example 2.

TABLE IV

Material for the Two Numerical Examples

Region Material	Σ_a (cm^{-1})	D (cm)	$\nu\Sigma_f$ (cm^{-1})
Core 1	0.062158	2.2008	0.107622
Core 2	0.062158	2.2008	0.102622
Blanket	0.064256	2.0950	0.0

fixed-source and k-eigenvalue problems. For all the test cases, we used a common NN architecture that embraced 8 hidden layers, 40 neurons per hidden layer, and tanh activation function. We used the Adam optimizer for 10^5 iterations followed by the L-BFGS optimization algorithm for each example. All calculations were performed on the Google Collab GPU machine.

IV.A. Fixed-Source Problems

IV.A.1. Fixed-Source 1D Homogenous Slab

The fixed-source 2-G diffusion equations may be given by

$$\begin{cases} -\ (D_1\ \ \phi_1) + \Sigma_{r,1}\phi_1 = Q_1 \\ -\ (D_2\ \ \phi_2) + \Sigma_{a,2}\phi_2 = \Sigma_{s,1\to 2}\phi_1 + Q_2 \end{cases}, \tag{27}$$

where Q_1 and Q_2 represent the fast and neutron source strength, respectively. In practice, we may assume that the external source existed only in the fast energy range (i.e., $Q_2 = 0$). We first solved Eq. (27) for an 80-cm slab subjected to a reflection boundary condition on the left side and zero flux on the right side. The material properties of the slab are given in Table V.

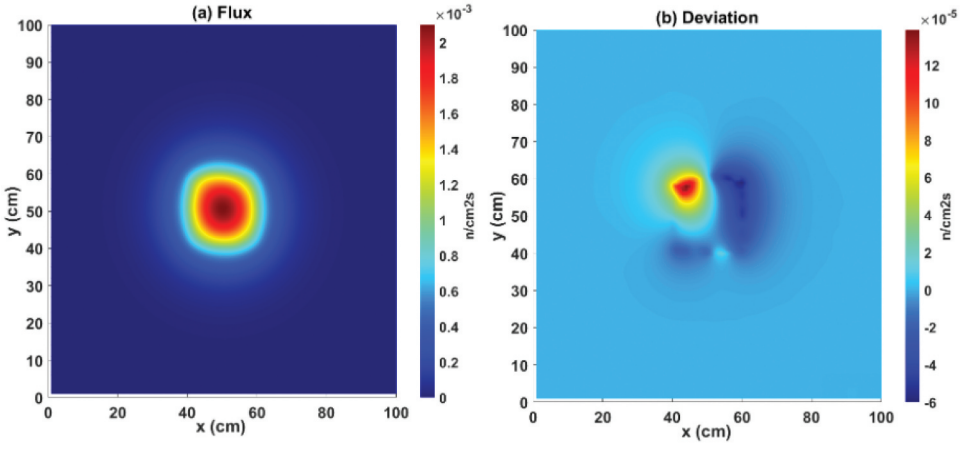

Fig. 9. Flux solution for Example 1: (a) heat map of the predicted flux and (b) difference between the reference solution and the PINN solution.

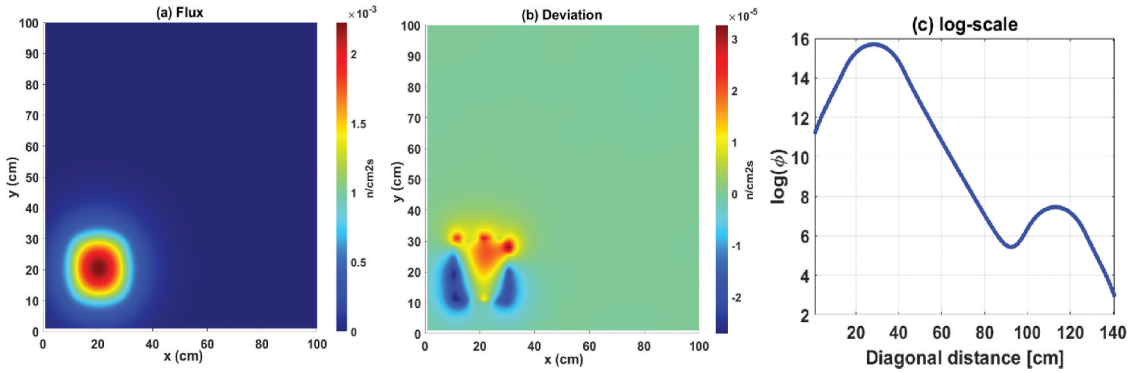

Fig. 10. Flux solution for Example 2: (a) heat map of the predicted flux, (b) difference between the reference solution and PINN solution, and (c) predicted log-scale flux plot across the diagonal of the reactor domain.

TABLE V

Material Properties of the Fixed-Source Homogenous Slab Example

D_1 (cm)	D_2 (cm)	$\Sigma_{r,1}$ (cm^{-1})	$\Sigma_{a,2}$ (cm^{-1})	$\Sigma_{s,1\to 2}$ (cm^{-1})	Q_1 $(\text{cm}^{-3}\cdot\text{s}^{-1})$
1.2	0.4	0.03	0.1	0.02	1.5

For the fixed-source problem, the loss function defined in the PINN method does not include a regularization term. The significance of this example is that it has the analytical solution given by

$$\begin{cases} \phi_1(x) = \frac{Q_1}{\Sigma_{r,1}}\left(1 - \dfrac{\cosh\left(x/L_1\right)}{\cosh\left(a/L_1\right)}\right) \\ \phi_2(x) = \frac{Q_1\Sigma_{s,1\to 2}}{\Sigma_{r,1}\Sigma_{a,2}}\left(1 - \dfrac{L_1^2\cosh\left(x/L_1\right)}{\left(L_1^2-L_2^2\right)\cosh\left(a/L_1\right)} + \dfrac{L_2^2\cosh\left(x/L_2\right)}{\left(L_2^2-L_1^2\right)\cosh\left(a/L_2\right)}\right) \end{cases}, \tag{28}$$

where the diffusion lengths L_1 and L_2 are given by

$$L_1 = \sqrt{\frac{D_1}{\Sigma_{r,1}}} \text{ and } L_2 = \sqrt{\frac{D_2}{\Sigma_{a,2}}}. \tag{29}$$

For the PINN implementation, the solution domain was sampled using 1000 points generated with the LHS strategy. The training time for this simple example is below 1 min. A comparison between the analytical and the PINN solution for this example is shown in Fig. 11, which clearly shows that the PINN obtained nearly identical flux solutions as the analytic ones. Two commonly used error metrics, MAE and MSE, can be used to have a direct quantitative inspection of the results. The MAE and the MSE are defined by

$$\text{MAE} = \frac{1}{N}\sum_{i=1}^{N}\left|\phi_{pred}(x_i) - \phi_{ref}(x_i)\right| \tag{30}$$

and

$$\text{MSE} = \frac{1}{N}\sum_{i=1}^{N}\left(\phi_{pred}(x_i) - \phi_{ref}(x_i)\right)^2, \tag{31}$$

where ϕ_{pred} and $\phi_{ref}(x_i)$ are the PINN solution and the reference analytic solution at point x_i, respectively. The MAE and the MSE for the predicted fast flux are 0.0043 and 3.599E-5, respectively. The MAE and the MSE for the thermal flux are 0.0014 and 2.593E-6, respectively. All these results indicate the correct implementation of the PINN approach for the 2-G problem.

IV.A.2. Fixed-Source 1D Heterogeneous Slab

In this example, we solved the fixed-source 2-G diffusion model for seven 100-cm slabs with three

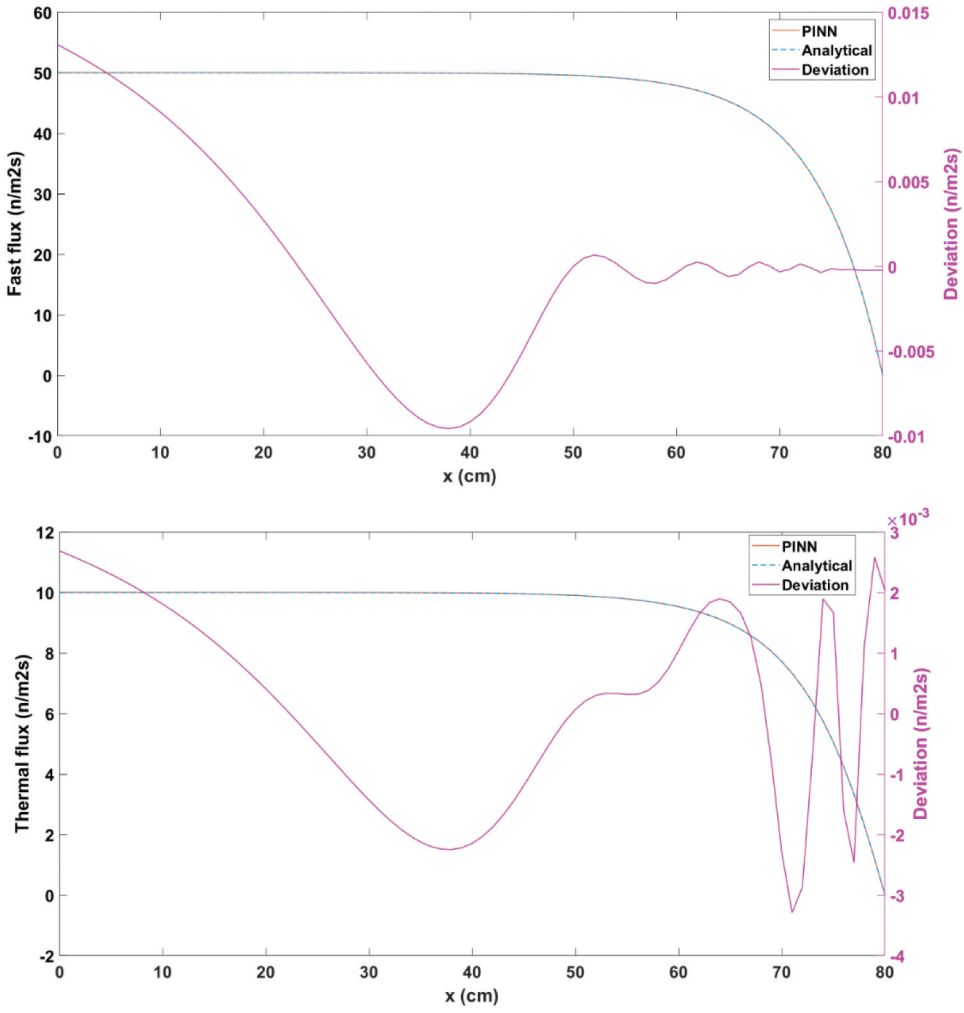

Fig. 11. Comparison between the PINN solution and the analytical solution for the fixed-source homogenous slab problem (flux on the left y-axis and error on the right y-axis).

types of materials arranged in the 1-2-3-2-3-3-2 manner, with each number representing the corresponding material type. The properties and volumetric source strength for each region are given in Table VI. Both the left and the right sides are assumed with the zero-flux boundary condition. This example was brought to us through Ref. 52. A FEM solution was used as a reference solution to assess the performance of the PINN solution. The comparison of the PINN solutions to the reference ones is shown in Fig. 12. The MAE and the MSE for the predicted fast flux are 0.1542 and 0.8403, respectively. The MAE and the MSE for the thermal flux are 0.0152 and 0.0037, respectively. All these results further justify the accuracy of the PINN solutions to the multigroup multiregion problems.

TABLE VI

Material Properties for the Fixed-Source Heterogeneous Slab Example

Material	D_1 (cm)	D_2 (cm)	$\Sigma_{r,1}$ (cm^{-1})	$\Sigma_{a,2}$ (cm^{-1})	$\Sigma_{s,1\to2}$ (cm^{-1})	Q_1 ($cm^{-3}\cdot s^{-1}$)
1	1.2	0.4	0.03	0.1	0.02	0
2	1.2	0.4	0.03	0.2	0.015	1.5
3	1.2	0.4	0.03	0.25	0.015	1.8

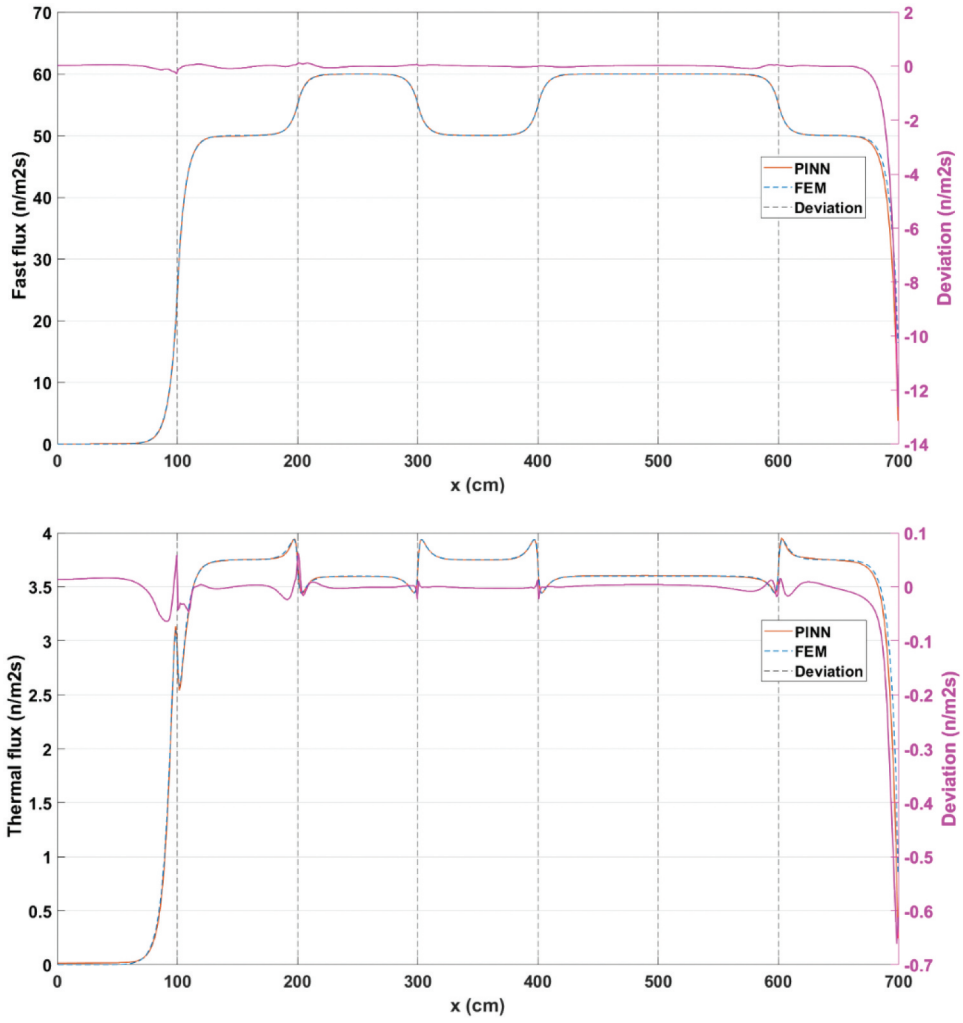

Fig. 12. Comparison between the PINN solution and the FEM solution for the fixed-source heterogeneous slab problem (flux on the left y-axis, and error on the right y-axis).

IV.A.3. Fixed-Source 2D Example

For the 2D fixed-source example, we applied the PINN approach to the X-Y Cartesian geometry shown in Fig. 13. The domain is subject to zero-flux boundary conditions on the left and bottom sides, and reflective boundary conditions on the right and top boundaries. The material properties of the two regions of the problem are given in Table VII. This example was brought to us originally through Ref. 53.

For the PINN implementation, the solution domain was sampled using $N_f = 50000, N_b = 200$ per side. COMSOL Multiphysics[48] was used to obtain a high-order FEM for solution verification. The predicted flux and the associated error distribution are shown in Fig. 14. The MAEs are 2.1382 and 3.9886 for the fast flux and the thermal flux, respectively. Considering the average flux value for the fast and thermal groups, their MAE values are within the 3% and 2% range, respectively.

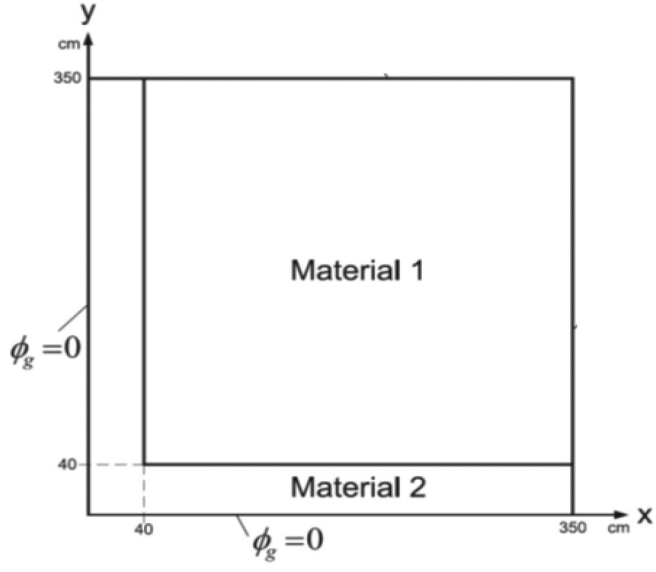

Fig. 13. Geometry of the 2D fixed-source problem.

TABLE VII

Materials Properties of the 2D Fixed-Source Problem

	Material 1	Material 2
D_1 (cm)	1.269	1.31
D_2 (cm)	0.9328	0.8695
$\Sigma_{a,1}$ (cm^{-1})	7.86E-4	0
$\Sigma_{a,2}$ (cm^{-1})	4.1E-3	2.117E-4
$\Sigma_{s,1\to2}$ (cm^{-1})	7.368E-3	1.018E-2
$\nu\Sigma_{f,2}$ (cm^{-1})	4.562E-3	0
Q_1 ($cm^{-3}\cdot s^{-1}$)	0.01	0

IV.B. *k*-Eigenvalue Problems

IV.B.1. k-Eigenvalue 1D Example

For the multigroup *k*-eigenvalue problems, we first solve the 1D 2-G slab example consisting of three regions, and each region is 21.42 cm thick. The material properties for each region are given in Table VIII. The slab domain is subjected to a reflective boundary condition on the left side and a zero-flux boundary condition on the right side.

For the PINN solution, we used 1000 points to sample the solution domain and used $C = 10$ for the regularization term. The normalized predicted flux compared to a FEM solution is shown in Fig. 15. The predicted *k*-eigenvalue is $k = 0.96764$, and the reference value is 0.96243 with a relative error less than 0.54%. For the fast flux, the MAE and the MSE in the predictions are 0.0011 and 1.846E-6, respectively. For the thermal flux, the MAE and the MSE are 0.0025 and 1.087E-5, respectively. Since all these errors are shown in absolute deviations between the PINN predictions to the reference solutions, these values are acceptable to some extent. The training time for this example is about 30 min. All models were trained on the Google Colab GPU machine.[54]

IV.B.2. k-Eigenvalue 2D Example

For the 2D case, we solve the 2D 2-G *k*-eigenvalue diffusion equation for the geometry given by Fig. 16. The

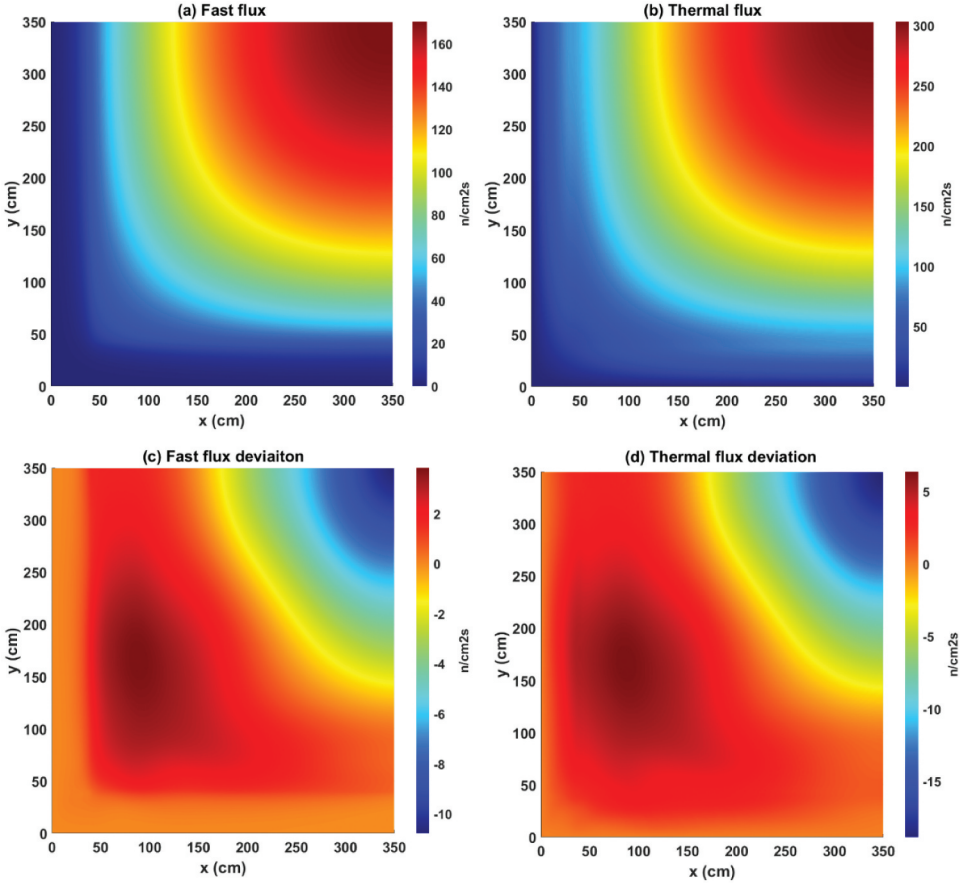

Fig. 14. Predicted PINN flux and the deviation from the reference solution for the 2D fixed-source problem.

TABLE VIII

Materials Properties of the k-Eigenvalue 1D Problem

	Material 1	Material 2	Material 3
D_1 (cm)	1.2	1.2	1.2
D_2 (cm)	0.4	0.4	0.2
$\Sigma_{r,1}$ (cm^{-1})	0.03	0.03	0.051
$\Sigma_{a,2}$ (cm^{-1})	0.3	0.25	0.04
$\Sigma_{s,1\rightarrow 2}$ (cm^{-1})	0.015	0.015	0.05
$\nu\Sigma_{f,1}$ (cm^{-1})	0.0075	0.0075	0
$\nu\Sigma_{f,2}$ (cm^{-1})	0.45	0.375	0

left and bottom sides are assumed to be reflective while the right and top sides have zero flux. The materials given in Table VIII are also used for this example. This example was a simplified version of the C5G7 benchmark problem.[55]

We sampled the solution domain using $N_f = 15000, N_b = 100$ per side. We used $C = 400$ for the regularization term. COMSOL Multiphysics was used to obtain a high-order FEM solution for solution verification. The predicted value of k = 0.93620, and the reference value from the FEM solution is 0.92764. The relative error of the k value is ~ 0.92%. The normalized flux and the associated error distribution are shown in Fig. 17. The MAEs are 2.16E-5 and 3.67E-5 for the fast flux and the thermal flux, respectively. Considering the average flux value for the fast and thermal groups, their MAE values are within the 8% and 15% range, respectively.

V. SUMMARY AND CONCLUSIONS

In this pilot study, we applied the forward PINN approach to solve the neutron diffusion equations for nuclear

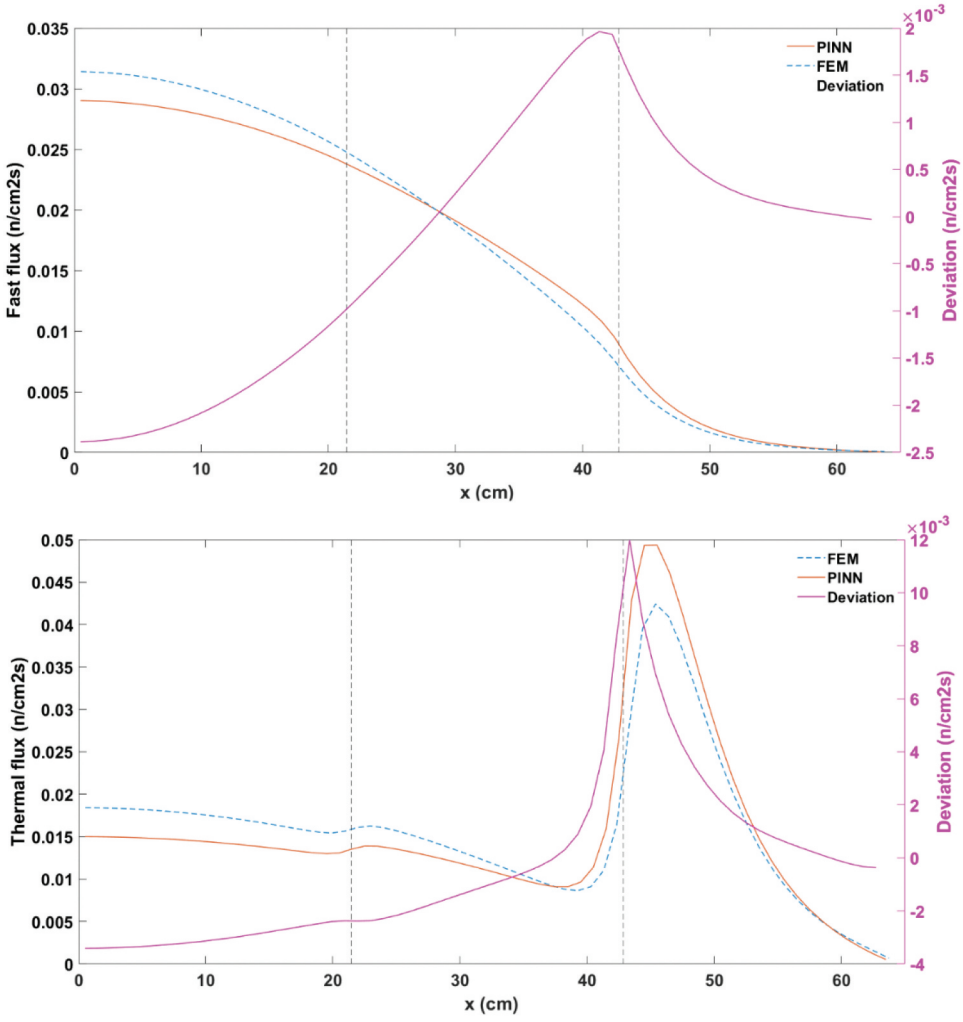

Fig. 15. Comparison between the PINN solution and the FEM solution for the 1D k-eigenvalue example (flux on the left y-axis and error on the right y-axis).

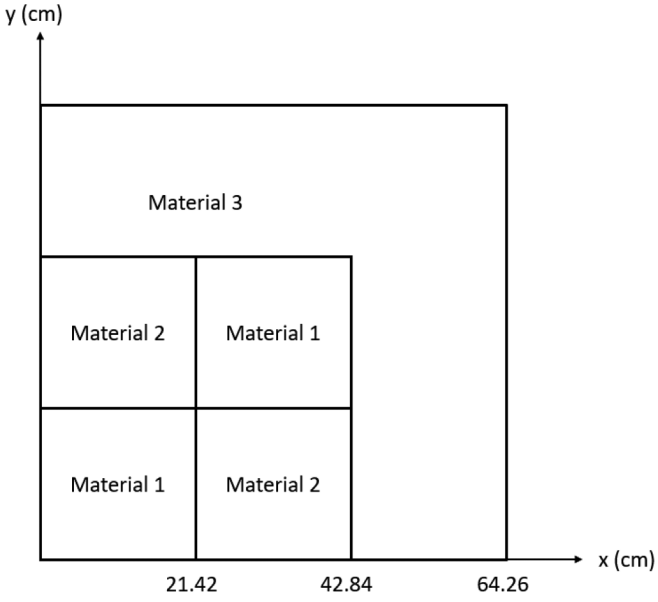

Fig. 16. Geometry of the 2D *k*-eigenvalue problem.

reactor calculations. We started the study by the implementation of the forward PINN approach to solve the fixed-source mode one-group diffusion problem. We then modified and extended the approach by adding the capability of solving the *k*-eigenvalue mode problem. The methods were then eventually extended to the 2-G diffusion models.

For the one-group, 2D, steady-state, fixed-source diffusion equation with zero-incoming flux boundary conditions, we used a simple example with the LCRM configuration to demonstrate the feasibility of the PINN method and to optimize the NN hyperparameters to be applied for the following examples. For the LCRM example, a systematical parameter study was first performed to address the behavior of the PINN for different NN architectures and training point sets. A FEM solution obtained from COMSOL Multiphysics was used as the reference solution. The mean relative error of the predicted flux was ~0.69%. The pointwise relative error was uniformly distributed across the solution domain; however, maximum errors were observed to appear mostly at the core-blanket interface. These results indicate the successful implementation of the PINN approach for fixed-source diffusion problems.

Eigenvalue problems are of special importance in reactor neutronic calculations, and the conventional

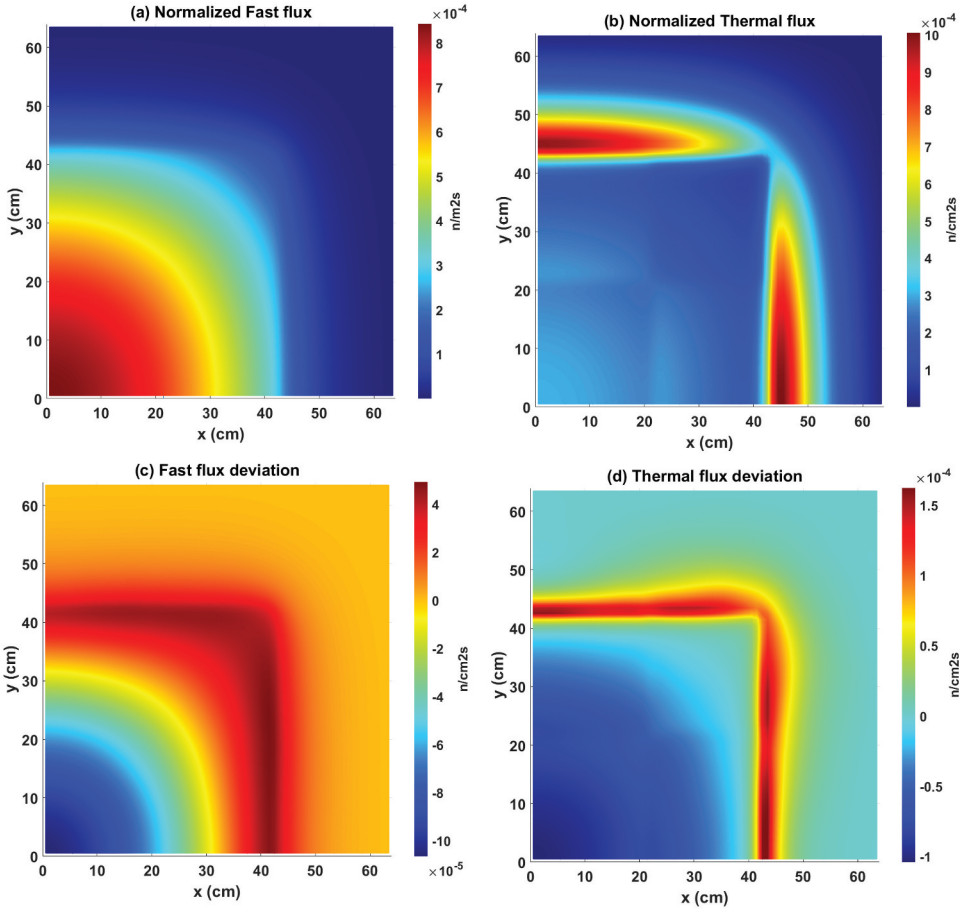

Fig. 17. Predicted PINN flux and the deviation from the reference solution for the 2D *k*-eigenvalue problem.

forward PINN approach is not capable for this class of problems. To overcome this limitation, we modified the PINN framework to enable the solving capability for the principal eigenvalue and the associated eigenfunction. This was achieved by first introducing a free learnable parameter to approximate the eigenvalue and a novel regularization term to enforce a nonzero solution by constraining the total fission rate to a user provided value. The viability of these ideas was demonstrated by solving the *k*-eigenvalue neutron diffusion equation in two different geometries. The solution was compared to the result yielded from the conventional power iteration solution for error quantification. For the first example with symmetric geometry and material settings, the predicted *k*-eigenvalue was 0.96266 compared to the reference solution 0.96395. The deviation of the PINN solution was 0.13% (−129 pcm). For the second example with nonsymmetric settings, the predicted eigenvalue was 0.95894, and the reference value was 0.96321 with a deviation of 0.44% (−427 pcm). No further sophisticated optimization efforts for the NN solutions were pursued, and the PINN estimated *k*-eigenvalues are acceptable at this moment.

We continued the study of the performance of the PINN by considering multigroup diffusion problems in both fixed-source and *k*-eigenvalue modes. The PINN approach can be naturally extended to solve coupled PDEs by devoting an output node for each unknown field and constrain it by the corresponding PDE. This capability was leveraged for solving the two coupled PDEs representing the 2-G neutron balance equations. Five numerical examples were examined for this class of problems. We started by solving a simple 1D homogenous fixed-source problem and compared the solution to the analytical solution. The MAEs in the PINN solution were 0.0043 and 0.0014 for the fast flux and the thermal group flux, respectively. Next, we solved for a 1D heterogeneous slab reactor problem with seven regions consisting of three different materials. The errors in the PINN solution compared to the FEM were 0.1542 and 0.0152 for fast flux and the thermal flux, respectively. The last fixed-source example was a 2D two-region test problem. The MAEs were 2.1382 and 3.9886 for the fast flux and the thermal flux, respectively. Finally, we solved two *k*-eigenvalue problems. The 1D example is a heterogeneous slab consisting of two different fuels and one reflector material. The predicted eigenvalue for this example was 0.96764 with the deviation from the reference solution of 0.54% (521 pcm). The 2D example consisted of five regions containing three different materials. The predicted eigenvalue was 0.93620 with a deviation of 0.92% (856 pcm). All the results from these test examples indicate the feasibility of the PINN approach for diffusion model–based reactor problems.

Based on the computation experience of these numerical examples, we have gained some first-hand understanding of the PINN approach on practical engineering applications. The PINN method apparently offers many calculation advantages over conventional numerical methods. Some of these advantages are summarized as follows:

1. It generates purely continuous and mesh-free solutions, which offers great flexibility for numerical solutions because rigorous mesh generation itself could be a big manpower burden for conventional numerical methods such as the FEM.

2. It precludes the need for special treatment for the boundary conditions or material interface, which also gives much convenience in the implementation of PDE-type problems.

3. It is readily applicable to complex geometries and versatile boundary conditions with very little sacrifice to the accuracy.

4. For eigenvalue problems, as demonstrated by the examples, it eliminates the iteration-based convergence scheme and greatly simplifies the solution procedure, which reduces the manpower needed to develop the conventional methods–based solvers.

5. It also holds the merit of being capable of solving coupled systems of PDEs, which makes it a good candidate for Multiphysics solvers, which are at the heart of the advanced nuclear reactor design process.

Given the above-mentioned advantages, the PINN suffers from some inherent drawbacks including the computation complexity, the different convergence rates for the various components of the loss function, and the possibility of convergence to a local minimum resulting in large error in the predicted solutions. All these drawbacks are related to the optimization algorithm, which searches for the optimum learnable parameters that minimize the loss function.

Since the main purpose of this paper was to demonstrate the feasibility of the PINN for reactor problems, much less attention was paid to the aspect of PINN optimization at this time. For future work, we are tackling PINN solution accuracy and efficiency by considering recent developments in this approach including optimizing the sampling strategy, using adaptive activation functions, applying the dynamic weighting of the loss function, and examining advanced NN architectures and

PINN variations. Following this optimization phase, we are considering expanding this framework to the Multiphysics domain by coupling the neutron diffusion equations with the thermal-hydraulic equation to develop a solver capable of tackling problems for the advanced reactor designs. We are also aiming to conduct a systematic study for the effect of dominance ratio on eigenvalue PINN solutions.

Acknowledgments

The authors are grateful to Yu Ma from the Sun Yat-sen University for bringing to our attention the PINN method and initializing fruitful discussions about this topic at the early stage of this research.

Disclosure Statement

No potential conflict of interest was reported by the author(s).

ORCID

Mohamed H. Elhareef http://orcid.org/0000-0001-5643-3591
Zeyun Wu http://orcid.org/0000-0002-6114-0352

References

1. A. KARPATNE et al., "Theory-Guided Data Science: A New Paradigm for Scientific Discovery from Data," *IEEE Trans. Knowl. Data Eng.*, **29**, *10*, 2318 (2017); https://doi.org/10.1109/TKDE.2017.2720168.
2. I. E. LAGARIS, A. LIKAS, and D. I. FOTIADIS, "Artificial Neural Networks for Solving Ordinary and Partial Differential Equations," *IEEE Trans. Neural Networks*, **9**, *5*, 987 (1998); https://doi.org/10.1109/72.712178.
3. M. RAISSI, P. PERDIKARIS, and G. E. KARNIADAKIS, "Physics-Informed Neural Networks: A Deep Learning Framework for Solving Forward and Inverse Problems Involving Nonlinear Partial Differential Equations," *J. Comput. Phys.*, **378**, 686 (2019); https://doi.org/10.1016/j.jcp.2018.10.045.
4. Y. ZHU et al., "Physics-Constrained Deep Learning for High-Dimensional Surrogate Modeling and Uncertainty Quantification Without Labeled Data," *J. Comput. Phys.*, **394**, 56 (2019); https://doi.org/10.1016/j.jcp.2019.05.024.
5. Y. LI, J. LU, and A. MAO, "Variational Training of Neural Network Approximations of Solution Maps for Physical Models," *J. Comput. Phys.*, **409**, 109338 (2020); https://doi.org/10.1016/j.jcp.2020.109338.
6. Z. LONG, Y. LU, and B. DONG, "PDE-Net 2.0: Learning PDEs from Data with a Numeric-Symbolic Hybrid Deep Network," *J. Comput. Phys.*, **399**, 108925 (2019); https://doi.org/10.1016/j.jcp.2019.108925.
7. Y. ZANG et al., "Weak Adversarial Networks for High-Dimensional Partial Differential Equations," *J. Comput. Phys.*, **411**, 109409 (2020); https://doi.org/10.1016/j.jcp.2020.109409.
8. S. SHALEV-SHWARTZ and S. BEN-DAVID, *Understanding Machine Learning: From Theory to Algorithms*, Cambridge University Press (2014).
9. Y. YANG and P. PERDIKARIS, "Adversarial Uncertainty Quantification in Physics-Informed Neural Networks," *J. Comput. Phys.*, **394**, 136 (2019); https://doi.org/10.1016/j.jcp.2019.05.027.
10. D. ZHANG et al., "Quantifying Total Uncertainty in Physics-Informed Neural Networks for Solving Forward and Inverse Stochastic Problems," *J. Comput. Phys.*, **397**, 108850 (2019); https://doi.org/10.1016/j.jcp.2019.07.048.
11. A. D. JAGTAP, E. KHARAZMI, and G. E. KARNIADAKIS, "Conservative Physics-Informed Neural Networks on Discrete Domains for Conservation Laws: Applications to Forward and Inverse Problems," *Comput. Methods Appl. Mech. Eng.*, **365**, 113028 (2020); https://doi.org/10.1016/j.cma.2020.113028.
12. X. MENG et al., "PPINN: Parareal Physics-Informed Neural Network for Time-Dependent PDEs," *Comput. Methods Appl. Mech. Eng.*, **370**, 113250 (2020); https://doi.org/10.1016/j.cma.2020.113250.
13. G. PANG et al., "nPINNs: Nonlocal Physics-Informed Neural Networks for a Parametrized Nonlocal Universal Laplacian Operator," *Algorithms Appl. J. Comput. Phys.*, **422**, 109760 (2020); https://doi.org/10.1016/j.jcp.2020.109760.
14. L. YANG, X. MENG, and G. E. KARNIADAKIS, "B-PINNs: Bayesian Physics-Informed Neural Networks for Forward and Inverse PDE Problems with Noisy Data," *J. Comput. Phys.*, **425**, 109913 (2021); https://doi.org/10.1016/j.jcp.2020.109913.
15. A. D. JAGTAP, K. KAWAGUCHI, and G. E. KARNIADAKIS, "Adaptive Activation Functions Accelerate Convergence in Deep and Physics-Informed Neural Networks," *J. Comput. Phys.*, **404**, 109136 (2020); https://doi.org/10.1016/j.jcp.2019.109136.
16. S. GOSWAMI et al., "Transfer Learning Enhanced Physics Informed Neural Network for Phase-Field Modeling of Fracture," *Theor. Appl. Fract. Mech.*, **106**, 102447 (2020); https://doi.org/10.1016/j.tafmec.2019.102447.
17. E. KHARAZMI, Z. ZHANG, and G. E. M. KARNIADAKIS, "hp-VPINNs: Variational

Physics-Informed Neural Networks with Domain Decomposition," *Comput. Methods Appl. Mech. Eng.*, **374**, 113547 (2021); https://doi.org/10.1016/j.cma.2020.113547.

18. E. SCHIASSI et al., "Extreme Theory of Functional Connections: A Fast Physics-Informed Neural Network Method for Solving Ordinary and Partial Differential Equations," *Neurocomputing*, **457**, 334 (2021); https://doi.org/10.1016/j.neucom.2021.06.015.
19. Z. MAO, A. D. JAGTAP, and G. E. KARNIADAKIS, "Physics-Informed Neural Networks for High-Speed Flows," *Comput. Methods Appl. Mech. Eng.*, **360**, 112789 (2020); https://doi.org/10.1016/j.cma.2019.112789.
20. X. JIN et al., "NSFnets (Navier-Stokes Flow Nets): Physics-Informed Neural Networks for the Incompressible Navier-Stokes Equations," *J. Comput. Phys.*, **426**, 109951 (2021); https://doi.org/10.1016/j.jcp.2020.109951.
21. H. WESSELS, C. WEIßENFELS, and P. WRIGGERS, "The Neural Particle Method—An Updated Lagrangian Physics Informed Neural Network for Computational Fluid Dynamics," *Comput. Methods Appl. Mech. Eng.*, **368**, 113127 (2020); https://doi.org/10.1016/j.cma.2020.113127.
22. F. SAHLI COSTABAL et al., "Physics-Informed Neural Networks for Cardiac Activation Mapping," *Front. Phys.*, **8** (2020); https://doi.org/10.3389/fphy.2020.00042.
23. M. H. ELHAREEF, Z. WU, and Y. MA, "Physics-Informed Deep Learning Neural Network Solution to the Neutron Diffusion Model," *Proc. Int. Conf. Mathematics and Computational Methods Applied to Nuclear Science and Engineering*, Virtual Meeting, October 3–7, 2021, American Nuclear Society (2021).
24. M. H. ELHAREEF and Z. WU, "Extending the Physics Informed Neural Network Model to Multigroup Neutron Diffusion Problems," presented at American Nuclear Society 2022 Student Conference, Urbana, Illinois, April 14, 2022.
25. M. H. ELHAREEF and Z. WU, "Extension of the PINN Diffusion Model to k-Eigenvalue Problems," *Proc. Int. Conf. Physics of Reactors (PHYSOR 2022)*, Pittsburgh, Pennsylvania, May 15–20, 2022, American Nuclear Society (2022).
26. Y. CHEN et al., "Physics-Informed Neural Networks for Inverse Problems in Nano-Optics and Metamaterials," *Opt. Express*, **28**, *8*, 11618 (2020); https://doi.org/10.1364/OE.384875.
27. K. SHUKLA et al., "Physics-Informed Neural Network for Ultrasound Nondestructive Quantification of Surface Breaking Cracks," *J. Nondestr. Eval.*, **39**, *3* (2020); https://doi.org/10.1007/s10921-020-00705-1.
28. Y. YUCESAN and F. VIANA, "A Physics-Informed Neural Network for Wind Turbine Main Bearing Fatigue," *Int. J. Progn. Health Manag.*, **11**, 17 (2020).
29. S. AMINI NIAKI et al., "Physics-Informed Neural Network for Modelling the Thermochemical Curing Process of Composite-Tool Systems During Manufacture," *Comput. Methods Appl. Mech. Eng.*, **384**, 113959 (2021); https://doi.org/10.1016/j.cma.2021.113959.
30. G. S. MISYRIS, A. VENZKE, and S. CHATZIVASILEIADIS, "Physics-Informed Neural Networks for Power Systems," presented at 2020 IEEE Power & Energy Society General Meeting (GM 2020), Virtual Event, August 3–6, 2020.
31. S. MISHRA and R. MOLINARO, "Physics Informed Neural Networks for Simulating Radiative Transfer," *J. Quant. Spectrosc. Radiat. Transfer*, **270**, 107705 (2021); https://doi.org/10.1016/j.jqsrt.2021.107705.
32. S. CAI et al., "Physics-Informed Neural Networks for Heat Transfer Problems," *J. Heat Transfer*, **143**, *6* (2021); https://doi.org/10.1115/1.4050542.
33. Z. FANG and J. ZHAN, "A Physics-Informed Neural Network Framework for PDEs on 3D Surfaces: Time Independent Problems," *IEEE Access*, **8**, 26328 (2020); https://doi.org/10.1109/ACCESS.2019.2963390.
34. Q. HE et al., "Physics-Informed Neural Networks for Multiphysics Data Assimilation with Application to Subsurface Transport," *Adv. Water Resour.*, **141**, 103610 (2020); https://doi.org/10.1016/j.advwatres.2020.103610.
35. M. LIU, L. LIANG, and W. SUN, "A Generic Physics-Informed Neural Network-Based Constitutive Model for Soft Biological Tissues," *Comput. Methods Appl. Mech. Eng.*, **372**, 113402 (2020); https://doi.org/10.1016/j.cma.2020.113402.
36. W. JI et al., "Stiff-PINN: Physics-Informed Neural Network for Stiff Chemical Kinetics," *J. Phys. Chem. A*, **125**, *36*, 8098 (2021); https://doi.org/10.1021/acs.jpca.1c05102.
37. H. BAO et al., "Computationally Efficient CFD Prediction of Bubbly Flow Using Physics-Guided Deep Learning," *Int. J. Multiphase Flow*, **131**, 103378 (2020); https://doi.org/10.1016/j.ijmultiphaseflow.2020.103378.
38. A. AKINS, Z. XIE, and X. WU, "Solving a System of Ordinary Differential Equations for Reactivity Insertion Accident with Artificial Neural Networks," *Trans. Am. Nucl. Soc.*, **125**, 472 (2021); https://doi.org/10.13182/T125-36879.
39. A. AKINS and X. WU, "Using Physics-Informed Neural Networks to Solve a System of Coupled ODEs for a Reactivity Insertion Accident," *Proc. Int. Conf. Physics of Reactors (PHYSOR 2022)*, Pittsburgh, Pennsylvania, May 15–20, 2022, American Nuclear Society (2022).
40. K. M. KIM, P. HURLEY, and J. P. DUARTE, "Prediction of Minimum Film Boiling Temperature Using a Physics-Informed Machine Learning-Aided Framework," *Proc. Topl. Mtg. Advances in Thermal Hydraulics (ATH 2022)*, Anaheim, California, June 12–16, 2022, p. 242, American Nuclear Society (2022).

41. K. PRANTIKOS, L. H. TSOUKALAS, and A. HEIFETZ, "Physics-Informed Neural Network Solution of Point Kinetics Equations for Development of Small Modular Reactor Digital Twin," *Trans. Am. Nucl. Soc.*, **126**, 219 (2022); https://doi.org/10.13182/T126-38405.

42. Y. XIE et al., "Neural Network Based Deep Learning Method for Multi-Dimensional Neutron Diffusion Problems with Novel Treatment to Boundary," *J. Nucl. Eng.*, **2**, *4*, 533 (2021); https://doi.org/10.3390/jne2040036.

43. D. C. LIU and J. NOCEDAL, "On the Limited Memory BFGS Method for Large Scale Optimization," *Math. Program.*, **45**, *1–3*, 503 (1989); https://doi.org/10.1007/BF01589116.

44. M. D. MCKAY, R. J. BECKMAN, and W. J. CONOVER, "A Comparison of Three Methods for Selecting Values of Input Variables in the Analysis of Output from a Computer Code," *Technometrics*, **42**, *1*, 55 (2000); https://doi.org/10.1080/00401706.2000.10485979.

45. C. LU and Z. WU, "Uncertainty Quantification of the 1-D SFR Thermal Stratification Model via Latin Hypercube Sampling Monte Carlo Method," *Nucl. Technol.*, **208**, *1*, 37 (Jan. 2022); https://doi.org/10.1080/00295450.2021.1874779.

46. S. WANG, X. YU, and P. PERDIKARIS, "When and Why PINNs Fail to Train: A Neural Tangent Kernel Perspective," *J. Comput. Phys.*, **449**, 110768 (2022); https://doi.org/10.1016/j.jcp.2021.110768.

47. J. DUDERSTADT and L. HAMILTON, *Nuclear Reactor Analysis*, Wiley & Sons (1976).

48. "COMSOL Multiphysics Reference Manual," COMSOL Multiphysics® (2014).

49. B. ROKROK, H. MINUCHEHR, and A. ZOLFAGHARI, "Element-Free Galerkin Modeling of Neutron Diffusion Equation in X–Y Geometry," *Ann. Nucl. Energy*, **43**, 39 (2012); https://doi.org/10.1016/j.anucene.2011.12.032.

50. D. P. KINGMA and J. BA, "Adam: A Method for Stochastic Optimization," *Proc. 3rd Int. Conf. Learning Representations*, San Diego, California, May 7–9, 2015.

51. M. ABADI et al., "TensorFlow: A System for Large-Scale Machine Learning," *Proc. 12th USENIX Conf. Operating Systems Design and Implementation*, Savannah, Georgia, November 2–4, 2016, p. 265, USENIX Association (2016).

52. Y. WANG, "HP-Mesh Adaptation for 1-D Multi-Group Neutron Diffusion Problems," Master Thesis, Texas A&M University (2006).

53. A. HÉBERT, "Applied Reactor Physics," Problem 5.7, Presses Internationales Polytechnique (2009).

54. E. BISONG, *Building Machine Learning and Deep Learning Models on Google Cloud Platform*, Apress, Berkeley California (2019).

55. E. E. LEWIS et al., "Benchmark Specification for Deterministic 2-D/3-D MOX Fuel Assembly Transport Calculations Without Spatial Homogenization (C5G7 MOX)," Organisation for Economic Co-operation and Development, Nuclear Energy Agency, Nuclear Science Committee (2001).