

## Highlights

### **Application of Deep Operator Networks for Data Reduction in Deterministic Neutron Calculations**

Ben Whewell, Ryan G. McClarren, Nathan Gibson, Ajeeta Khatiwada

- Deep operator networks effectively predict the matrix-vector product of the scattering term in criticality neutron transport simulations.
- The trunk network is able to represent different material compositions of either stainless steel or a mixed oxide fuel.
- Applying deep operator networks to one-dimensional spherical test problems reduced memory requirements by a factor of 70 and accelerated transport simulation convergence by a factor of approximately 7 to 16.

# Application of Deep Operator Networks for Data Reduction in Deterministic Neutron Calculations

Ben Whewell<sup>a,\*</sup>, Ryan G. McClarren<sup>b</sup>, Nathan Gibson<sup>a</sup> and Ajeeta Khatiwada<sup>a</sup>

<sup>a</sup>Los Alamos National Laboratory, P.O. Box 1663, Los Alamos, 87545, NM, USA

<sup>b</sup>University of Notre Dame, Department of Aerospace and Mechanical Engineering, Notre Dame, 46556, IN, USA

## ARTICLE INFO

### Keywords:

Deep Operator Networks  
Data Reduction  
Machine Learning  
Neutron Transport

## ABSTRACT

High-fidelity neutron transport simulations require large memory footprints which can become a limiting factor. A significant portion of this memory footprint comes from the large scattering matrices, which are needed for every material, enrichment level, scattering moment, and temperature. Deep operator networks (DeepONets) can replace the scattering term in the criticality neutron transport problem by predicting the matrix-vector product using a machine learning model. These models eliminate the need for scattering matrices in the multigroup source iteration. DeepONets are generalizable, as the trunk network can encode the composition of different materials without having to retrain the model. One example model uses the weight percentages of eight elements found in stainless steel (iron, chromium, carbon, silicon, manganese, molybdenum, nickel, and copper) as inputs to the trunk network. A second example models a mixed oxide (MOX)fissionable material with varying percentages of uranium-235 and plutonium dioxide. Both models are shown to be accurate for different one-dimensional spherical  $k$ -eigenvalue criticality problems that vary in material layout and composition. This was accomplished while simultaneously reducing memory requirements by a factor of 70 and accelerating source iteration convergence by a factor of approximately 7 to 16. Future work involves adding more materials to the trunk network, incorporating anisotropic scattering, and including temperature-dependent scattering cross section matrices.

## 1. Introduction

Deterministic methods for solving the neutron transport equation (NTE) discretize seven dimensions to approximate the neutron population in a medium [10]. To ensure sufficient accuracy, each dimension must be adequately discretized. Techniques to reduce the computational cost of neutron transport solvers have received continued interest because they enable the use of high-fidelity grids. A significant amount of focus has been on computational speedup [14, 20, 19, 23], but given the need to reduce the memory-to-compute ratio, data reduction is becoming increasingly important. Previous work on data reduction in transport solvers has focused either on low-rank solutions to reduce the memory footprint [15] or on applying data reduction techniques to simulation output [2]. In this work, we introduce an approach that embeds data reduction directly in the transport solver by predicting the scattering term using machine learning (ML).

Machine learning has previously been used to predict the scattering term [22]. In that approach, Deep Jointly-Informed Neural Networks (DJINN) [5] were used to tune and train neural networks for both fissionable and reflector materials. This was effective for predicting the scattering and fission terms for fissionable materials, but it had difficulty predicting the scattering rate density for a generalized reflector model. Deep operator networks (DeepONets) [11] address this deficiency by allowing for flexibility in material composition.

DeepONets modify the neural network architecture to learn an operator, or a mapping between functions. For use with the scattering term, the spatially dependent neutron flux is the input function and the scattering source is the output function, such that

$$\mathcal{M}_\theta : \phi \mapsto \sigma_\theta^s \phi \quad (1)$$

where  $\theta$  represents the material properties, expressed as weight percentages of the constituent elements or compounds. This formulation is well suited to the neutron transport problem because the scattering term fundamentally represents a family of linear operators acting on the flux, one for each material composition. Rather than storing an explicit scattering cross section matrix for each material, composition, or retraining a model for every new composition, the DeepONet learns the underlying operator itself, parameterized by  $\theta$  through the trunk network. This enables generalization across a continuous range of material compositions, which motivates the memory reduction and addresses the single-material limitation of previous work.

For this application, DeepONets are a more suitable architecture than fully connected neural networks or physics-informed neural networks (PINNs) [16]. Fully connected neural networks learn a fixed mapping between vectors and would encode a single material composition per trained model, similar to the DJINN models in [22]. PINNs, on the other hand, embed the governing equations in the loss function rather than replacing and compressing the scattering data. They would neither reduce the scattering matrix memory footprint nor provide a drop-in operator for the source iteration.

 whewell@lanl.gov (B. Whewell)

ORCID(s): 0000-0001-7826-5525 (B. Whewell); 0000-0002-8342-6132 (R.G. McClarren); 0000-0001-7717-0364 (N. Gibson); 0000-0002-1270-078X (A. Khatiwada)

The paper is organized as follows. The neutron transport equation is introduced in Section 2. Deep operator networks and their implementation with the NTE scattering term are presented in Section 3. One-dimensional spherical results for fissionable and reflector problems are presented in Section 4. Conclusions and future work are discussed in Section 5.

## 2. The Neutron Transport Equation

The neutron transport equation comprises seven dimensions: three spatial  $\mathbf{x} \in D \subset \mathbb{R}^3$ , two angular  $\Omega \in \mathbb{S}^2$ , energy  $E > 0$ , and time  $t$  [10]. The NTE describes the angular neutron flux  $\Psi(\mathbf{x}, \Omega, E, t)$  as neutrons interact with the surrounding medium. The NTE can be reformulated as a  $k$ -eigenvalue problem, which removes the time dependence and estimates the change in neutron population from one generation to the next through  $k_{\text{eff}}$ . The continuous  $k$ -eigenvalue NTE,

$$\begin{aligned} \Omega \cdot \nabla \Psi(\mathbf{x}, \Omega, E) + \sigma^t(\mathbf{x}, E) \Psi(\mathbf{x}, \Omega, E) = \\ \int_0^\infty dE' \sigma^s(\mathbf{x}, E' \rightarrow E) \Phi(\mathbf{x}, E') \\ + \frac{\chi(\mathbf{x}, E)}{k_{\text{eff}}} \int_0^\infty dE' \nu(\mathbf{x}, E') \sigma^f(\mathbf{x}, E') \Phi(\mathbf{x}, E') \end{aligned} \quad (2)$$

shows the streaming and removal terms on the left-hand side and the scattering and fission terms on the right-hand side. The macroscopic cross sections  $\sigma^t$ ,  $\sigma^s$ , and  $\sigma^f$  are the probabilities per unit path length of total, scattering, and fission interactions, respectively. The quantity  $\nu$  is the average number of neutrons emitted per fission event, and  $\chi$  is the energy distribution of neutrons emitted by fission, where  $\int_0^\infty \chi(\mathbf{x}, E) dE = 1$ . The scalar flux  $\Phi(\mathbf{x}, E)$  is calculated by integrating the angular flux over the angular dimension  $\Omega$ . The effective multiplication factor  $k_{\text{eff}}$  describes the change in neutron population between generations, where  $k_{\text{eff}} = 1$  corresponds to a critical system, and  $k_{\text{eff}} > 1$  and  $k_{\text{eff}} < 1$  correspond to supercritical and subcritical systems, representing an increase or a decrease in neutron population, respectively.

### 2.1. Multigroup Discrete Ordinates Neutron Transport

The angular dimension is discretized using the discrete ordinates ( $S_N$ ) method [1] to solve for the angular flux at specific angles. Quadrature sets approximate the integral over angle using directions  $\Omega_n$  and weights  $w_n$ , where  $n \in \mathcal{N} := \{1, \dots, N\}$ . The multigroup method discretizes the energy dimension by converting the energy-dependent cross sections into piecewise-constant values, where the group boundaries are  $E_{\min} = E_0 < E_1 < \dots < E_g < E_{g+1} < \dots < E_G = E_{\max}$  for energy group  $g \in \mathcal{G} := \{1, \dots, G\}$ .

The multigroup total cross section is defined as

$$\sigma_g^t(\mathbf{x}) \approx \frac{\int_{E_{g-1}}^{E_g} dE \sigma^t(\mathbf{x}, E) \Phi(\mathbf{x}, E)}{\int_{E_{g-1}}^{E_g} dE \Phi(\mathbf{x}, E)} \quad (3)$$

which uses the scalar flux  $\Phi(\mathbf{x}, E)$  as a weighting function. Nuclear data processing codes, such as NJOY [12], facilitate the energy discretization process.

The continuous NTE in Eq. (2) can be reformulated as the  $k$ -eigenvalue multigroup discrete ordinates NTE,

$$\begin{aligned} \Omega \cdot \nabla \psi_{n,g}(\mathbf{x}) + \sigma_g^t \psi_{n,g}(\mathbf{x}) = \\ \sum_{g'=1}^G \sigma_{g' \rightarrow g}^s \phi_{g'}(\mathbf{x}) + \frac{\chi_g}{k_{\text{eff}}} \sum_{g'=1}^G \nu_{g'} \sigma_{g' \rightarrow g}^f \phi_{g'}(\mathbf{x}) \end{aligned} \quad (4)$$

where the discretized angular and scalar fluxes are represented as  $\psi_{n,g}(\mathbf{x})$  and  $\phi_g(\mathbf{x})$ , respectively.

A common multigroup NTE solver for the  $k$ -eigenvalue problem uses two loops. The inner loop uses Gauss-Seidel iterations to converge the scattering term while keeping the fission term fixed. This loop involves “sweeping” [10] over the spatial dimension for each energy group and angle. The outer loop is the power iteration, which converges the fission term and calculates the largest  $k$ -eigenvalue [10].

### 2.2. Memory Reduction Motivation

The formulation of the energy grid is important for estimating the flux accurately, as poor grid structures lead to inaccurate flux approximations. Although accuracy does not increase monotonically with the number of energy groups, it is often desirable to use as many groups as computational constraints allow. The scattering matrix  $\sigma^s$  is of size  $G \times G$  and is needed for every material, enrichment level, temperature, and scattering moment. Because of the desire to use as many energy groups as possible, combined with the requirement to store these cross section matrices, memory storage can be a limiting factor when running neutron transport simulations [13, 6, 7]. In this work, machine learning is used to reduce the need for the scattering matrices when running transport simulations.

## 3. Deep Operator Networks

Deep operator networks (DeepONets) [11] are designed to learn complex nonlinear operators that map between infinite-dimensional function spaces. The operator  $M$  takes the function  $\phi(\mathbf{x})$  as input and produces the output function  $M(\phi)$ , which is evaluated at  $\theta$  to give the discrete output value  $M(\phi)(\theta)$ . This relationship is split between a branch network, which takes  $\phi(\mathbf{x})$  as input, and a trunk network, which takes  $\theta$  as input. An unstacked DeepONet with one trunk network and one branch network is used in this work. The outputs of these networks are combined to predict the matrix-vector product. The unstacked architecture in [11] was chosen because the flux spectrum is represented as a single output function, and the shared branch network jointly produces the coefficients to reconstruct this function. In contrast, a stacked DeepONet uses multiple independent branch networks that each output a single value. This would require significantly more parameters and would not exploit the shared features among the coefficients.

### 3.1. DeepONets and the NTE

DeepONets can be applied to the scattering term in the NTE, modifying Eq. (4) as

$$\Omega \cdot \nabla \psi_{n,g}(\mathbf{x}) + \sigma_g^t \psi_{n,g}(\mathbf{x}) = M(\phi)(\theta) + \frac{\chi_g}{k_{\text{eff}}} \sum_{g'=1}^G \nu_{g'} \sigma_{g' \rightarrow g}^f \phi_{g'}(\mathbf{x}) \quad (5)$$

where  $M(\phi)(\theta)$  is the operator replacing the matrix-vector multiplication in the Gauss-Seidel iteration. Here,  $\phi$  is the scalar flux calculated at each iteration, and  $\theta$  represents the inputs to the trunk network. The  $\theta$  values represent the elemental composition of the material used in the given simulation, with different variations discussed in detail in Section 4. By training the trunk network on a range of material compositions, we avoid storing a new scattering matrix for each material variation when running neutron transport simulations.

### 3.2. Preserving Reaction Rates

The application of DeepONets to the scattering term can effectively predict the matrix-vector product, but it does not necessarily preserve the reaction rate. Reaction rate errors can negatively impact the approximation of the flux and the  $k$ -effective solution to the  $k$ -eigenvalue criticality equation [9]. As presented in [22], a scaling factor is introduced to enforce conservation and correct the DeepONet output. The matrix-vector product  $\sigma^s \phi$  for a single energy group  $g$  represents the scattering rate into group  $g$ ,

$$(\sigma^s \phi)_g = \sum_{g'=1}^G \sigma_{g' \rightarrow g}^s \phi_{g'}. \quad (6)$$

Summing over the energy groups and exchanging the order of summation yields

$$\sum_{g=1}^G (\sigma^s \phi)_g = \sum_{g=1}^G \sum_{g'=1}^G \sigma_{g' \rightarrow g}^s \phi_{g'} = \sum_{g'=1}^G \sigma_{g'}^s \phi_{g'}, \quad (7)$$

meaning that only the total scattering cross section vector needs to be stored rather than the full scattering matrix. The scaling factor is then defined as

$$m = \frac{\sum_{g'=1}^G \sigma_{g'}^s \phi_{g'}}{\sum_{g=1}^G (M(\phi)(\theta))_g} \quad (8)$$

where  $\sigma_{g'}^s$  is the total scattering cross section for group  $g'$ . This scaling factor does not ensure that the group-specific values are correct, but it does preserve the total scattering rate density.

## 4. One-Dimensional Results

To demonstrate the effectiveness of DeepONets, two models are presented: one for a fissionable material and one

for a reflector material. The fissionable material is a MOX-like material that varies the  $^{235}\text{U}$  enrichment of the  $\text{UO}_2$  and the fraction of  $\text{PuO}_2$  relative to  $\text{UO}_2$ . The reflector material varies the concentrations of eight elements commonly found in stainless steel (iron, chromium, carbon, silicon, manganese, molybdenum, nickel, and copper). These ML models are presented separately in Sections 4.1 and 4.2. To demonstrate the full memory and computational savings of applying DeepONets to the scattering term in the NTE, these models are combined in a single simulation in Section 4.3. The memory reduction and computational speedup are discussed in Section 4.4.

The data collection process involved two steps. The first step involved sampling the subspace of material compositions for either MOX or stainless steel. The second step required running multiple criticality simulations with different material orientations and compositions. The scalar flux was saved at every source iteration for each material region in order to provide realistic inputs for the ML models. Each sampled material composition was then paired with every saved scalar flux, forming the Cartesian product of the two sets. For example, 25 material compositions and 400 scalar fluxes would produce 10,000 training samples. The target for each pair is the matrix-vector product  $\sigma^s(\theta) \phi$ , computed using the reference scattering matrix for composition  $\theta$ . The resulting dataset was divided using a 60%/20%/20% train/validation/test split.

Each criticality simulation used the same discretization scheme:  $S_8$  discrete ordinates, a 70-group energy structure ( $G = 70$ ), and isotropic scattering. The six material layouts, referred to as orientations and shown in Fig. 1, are one-dimensional spherical problems solved with the diamond difference spatial discretization method [10]. The spatial resolution remained constant for each orientation, with  $h = 0.01$  cm for orientations 1, 3, and 4,  $h = 0.02$  cm for orientations 2 and 6, and  $h = 0.025$  cm for orientation 5.

The reference model with full scattering cross section matrices is compared to the DeepONet models in two ways. The first is the absolute  $k$ -effective difference, measured in per cent mille (pcm), as is customary for reactivity. The second uses the scattering rate density, which accounts for the number of scattering interactions occurring per unit volume per unit time and is calculated as

$$\text{SRD} = \sum_{g=1}^G \sum_{g'=1}^G \sigma_{g' \rightarrow g}^s \phi_{g'}, \quad (9)$$

with units of  $\text{cm}^{-3}\text{s}^{-1}$ . The root mean squared error (RMSE) is used to compare the scattering rate density between the reference and ML models. The RMSE was also normalized by the number of spatial cells to account for the different material orientations.

### 4.1. Fissionable Material Model

The DeepONet is first shown to predict the scattering matrix-vector product for a fissionable material. A MOX-type material was used, varying both the  $^{235}\text{U}$  enrichment

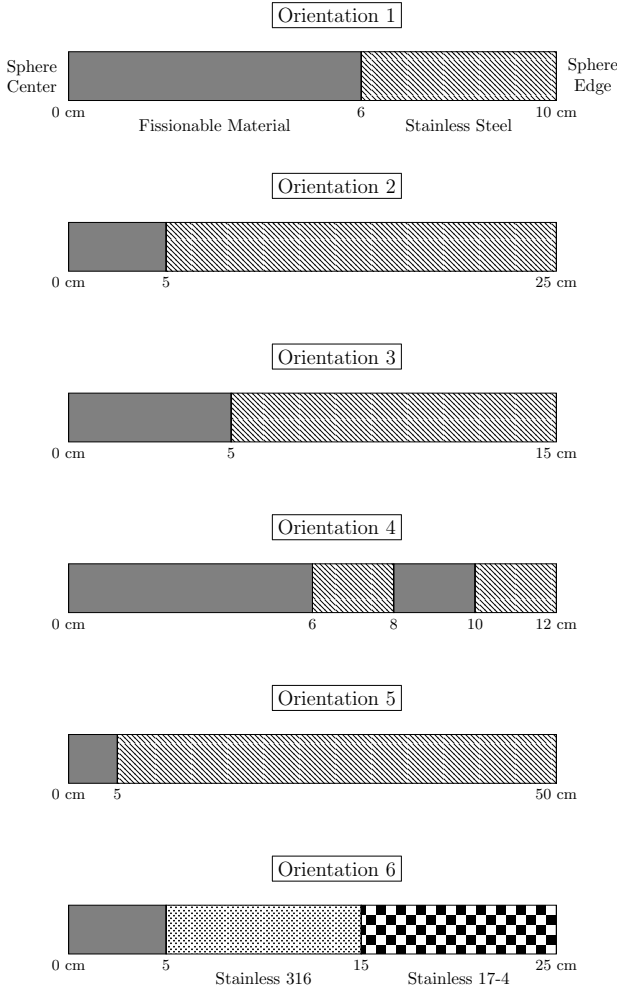


**Figure 1:** Material orientations used in this work. The spatial resolution remained constant when the material composition was modified within the same orientation. The spatial cell widths were  $h = 0.01$  cm for orientations 1, 3, and 4,  $h = 0.02$  cm for orientations 2 and 6, and  $h = 0.025$  cm for orientation 5.

of the  $\text{UO}_2$  ( $w_{235}$ ) and the fraction of  $\text{PuO}_2$  relative to  $\text{UO}_2$ . The MOX model trunk network input is represented as  $\theta = [w_{235}, \text{PuO}_2]$ .

The uranium in the uranium dioxide consists of  $^{234}\text{U}$ ,  $^{235}\text{U}$ , and  $^{238}\text{U}$ , where the  $^{234}\text{U}$  concentration is determined from  $w_{235}$  using

$$w_{234} = 0.0015 + 0.0058 w_{235} + 0.000054 w_{235}^2 \quad (10)$$

as taken from [17]. The plutonium mixture was held constant at 80%  $^{239}\text{Pu}$  and 20%  $^{240}\text{Pu}$ . Allowing for variations in the plutonium isotopic composition is an area of future work.

The material composition training data used 30 samples, varying  $w_{235} \in [0.5\%, 95\%]$  and  $\text{PuO}_2 \in [0\%, 95\%]$ . The scalar flux training data were collected by running three separate criticality problems with varying MOX fuel compositions. Two of the criticality simulations used orientation 1 in Fig. 1 with  $\theta = [0.5\%, 20\%]$  and  $[95\%, 90\%]$ . These were combined with one simulation of orientation 2 using

$\theta = [0.5\%, 20\%]$ . Saving the scalar flux at every source iteration resulted in 700,700 scalar flux training data points. The stainless steel 304 composition listed in Table 1 was used as the reflector material for all three simulations.

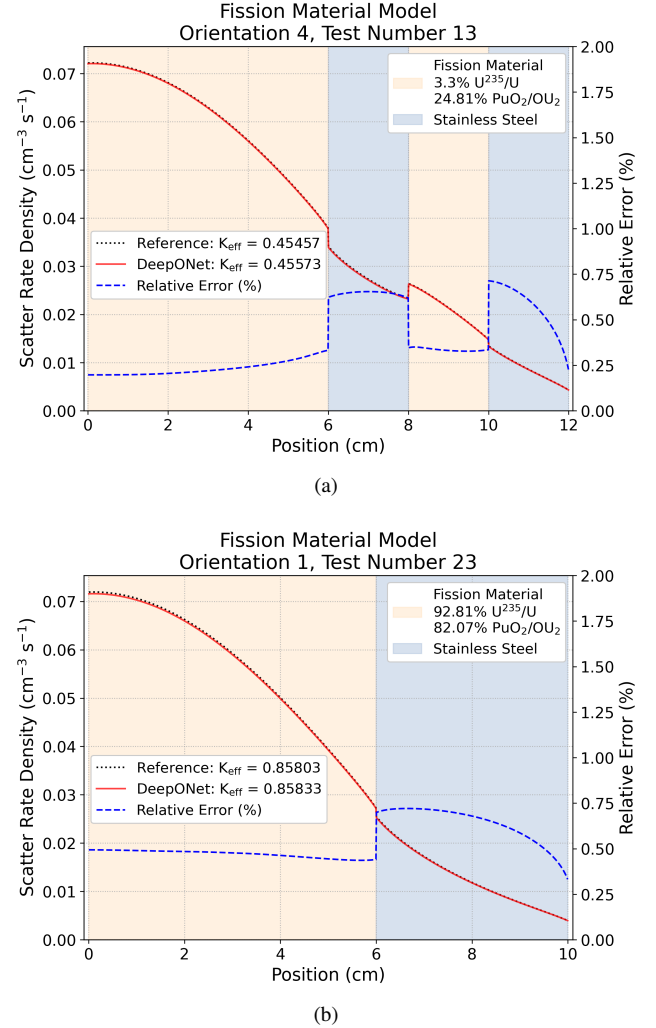


**Figure 2:** Test results for the fissionable material ML model. The highest  $k$ -effective error among all test problems is shown in (a), and the highest root mean squared scattering rate density error among all test problems is shown in (b).

The branch network has one hidden layer with 144 nodes. The trunk network also has one hidden layer, with 252 nodes, followed by a dropout layer. The two networks are combined in a layer with 224 nodes, and the ReLU activation function is used throughout the model. The training used 80 epochs, a batch size of 256, and a learning rate of  $5 \times 10^{-4}$ . The DeepONet training used the Huber loss function [4] and the Adam optimizer [8].

The trained model was tested with 25 randomly sampled trunk inputs and four different material orientations (orientations 1, 2, 3, and 4 in Fig. 1). Varying the material compositions and layouts did not affect the accuracy of the DeepONet model. For these 100 test cases, the average  $k$ -effective error was 23.7 pcm, with a maximum of 116.1 pcm,

**Table 1**

Weight percentage compositions of stainless steel alloys with their AISI designations, inspired by [21, 18].

| AISI | Fe    | Cr   | C    | Si  | Mn  | Mo   | Ni   | Cu  |
|------|-------|------|------|-----|-----|------|------|-----|
| 304  | 70.9  | 18.0 | 0.1  | 1.0 | 2.0 | 0.0  | 8.0  | 0.0 |
| 308  | 65.9  | 20.0 | 0.1  | 1.0 | 2.0 | 0.0  | 11.0 | 0.0 |
| 316  | 63.9  | 16.0 | 0.1  | 1.0 | 2.0 | 3.0  | 14.0 | 0.0 |
| 321  | 68.9  | 18.0 | 0.1  | 1.0 | 2.0 | 0.0  | 10.0 | 0.0 |
| 410  | 85.35 | 12.5 | 0.15 | 1.0 | 1.0 | 0.0  | 0.0  | 0.0 |
| 420  | 85.8  | 12.0 | 0.2  | 1.0 | 1.0 | 0.0  | 0.0  | 0.0 |
| 430  | 79.9  | 17.0 | 0.1  | 1.0 | 1.0 | 0.0  | 1.0  | 0.0 |
| 440  | 78.25 | 17.0 | 1.0  | 1.0 | 1.0 | 0.75 | 1.0  | 0.0 |
| 17-4 | 73.9  | 15.0 | 0.1  | 1.0 | 1.0 | 3.0  | 3.0  | 3.0 |

when compared to a reference solution that did not use ML. This case is shown in Fig. 2(a). Although the model was not trained on multiple shells of MOX and stainless steel, the relative scattering rate density error is less than 1%.

The RMS scattering rate density error for the test set averaged  $4.024 \times 10^{-8} \text{ cm}^{-3} \text{ s}^{-1}$ , with a maximum of  $2.229 \times 10^{-7} \text{ cm}^{-3} \text{ s}^{-1}$ . This case is shown in Fig. 2(b), with the error most noticeable near the center of the sphere. The DeepONet result has a relative error of less than 0.75% and a  $k$ -effective difference of 30 pcm when compared to the reference solution. Both of the results in Fig. 2 represent the worst cases for the fissionable material DeepONet model. However, they still represent accurate approximations of the reference solution for orientations and material compositions not seen in the training data.

#### 4.2. Reflector Material Model

A second DeepONet model was trained by varying the material composition of stainless steel. The weight percentages of eight common elements found in stainless steel were chosen as inputs to the trunk network: iron, chromium, carbon, silicon, manganese, molybdenum, nickel, and copper. Varying the percentages within given ranges enables the DeepONet model to predict the scattering matrix-vector product for different grades of stainless steel. These ranges are 10% to 30% for chromium, 0% to 1% for carbon, 0% to 2% for silicon, 0% to 15% for manganese, 0% to 10% for molybdenum, 3% to 20% for nickel, and 0% to 6% for copper, with iron making up the remaining balance. These ranges were based on the compositions of standard stainless steel grades reported in [21, 18] to provide realistic stainless steel alloys. The trunk network input is represented as  $\theta = [\text{Fe}, \text{Cr}, \text{C}, \text{Si}, \text{Mn}, \text{Mo}, \text{Ni}, \text{Cu}]$ , where  $\sum \theta = 1$ .

A collection of 40 stainless steel alloys was used for the material composition training data. The scalar flux training data were collected using two simulations of orientation 5 in Fig. 1. The fissionable material consisted of enriched uranium, with either 10% or 90%  $^{235}\text{U}$ , to provide subcritical and supercritical training data. The stainless steel used in these simulations consisted of 79% iron, 18% chromium, 1% carbon, 1% silicon, and 1% manganese. Saving the scalar

flux at every source iteration for the stainless steel region resulted in 471,600 scalar flux training data points.

The trunk network had three hidden layers with 126, 76, and 164 nodes, with dropout rates of 10% and 15% after the second and third hidden layers, respectively. The branch network had one hidden layer with 206 nodes and a dropout layer with a 5% dropout rate. The trunk network used ReLU activation functions, while the branch network used a hyperbolic tangent activation function. These networks were combined in a layer with 160 nodes. The training used 90 epochs, a batch size of 256, and a learning rate of  $5 \times 10^{-4}$ . The Huber loss function and Adam optimizer were also used.

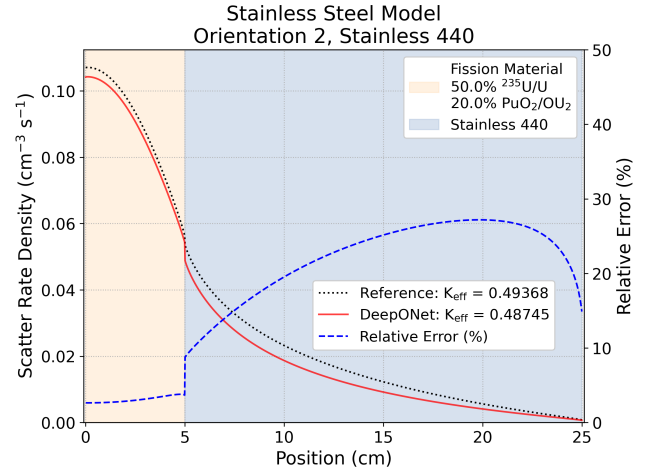


**Figure 3:** Test result for the reflector material ML model. This case has the highest absolute  $k$ -effective error and the highest root mean squared scattering rate density error among all 45 test problems.

The model was tested using the nine stainless steel alloys listed in Table 1 and the first five orientations in Fig. 1. The fissionable material was held constant as a 20%/80%  $\text{PuO}_2/\text{UO}_2$  MOX fuel with 50% enriched uranium. The average absolute  $k$ -effective error over the 45 test cases was 167.3 pcm, with a maximum of 623.4 pcm. The average RMS scattering rate density error normalized by spatial cell was  $5.223 \times 10^{-7} \text{ cm}^{-3} \text{ s}^{-1}$ , and the maximum was  $2.579 \times 10^{-6} \text{ cm}^{-3} \text{ s}^{-1}$ . The maximum  $k$ -effective error and the maximum RMS error were both obtained for the case shown in Fig. 3, which uses a 20 cm reflecting region and stainless steel 440. The 400-series steels have nickel concentrations ( $\leq 1\%$ ) outside the training range of [3%, 20%]. However, other 400-series steels perform well for large reflecting regions, as seen in Fig. 4(a), which uses a 45 cm reflector. Stainless steel 440 differs from the other test cases, including the other 400-series steels, in its carbon content: 1% versus  $\leq 0.2\%$ . This is most likely the primary cause of the poor performance of the stainless steel DeepONet model for this case. The combination of out-of-range compositions and larger reflecting regions exacerbates the deficiencies of the model.

Overall, the stainless steel DeepONet performed well, as most absolute  $k$ -effective errors were less than 250 pcm.

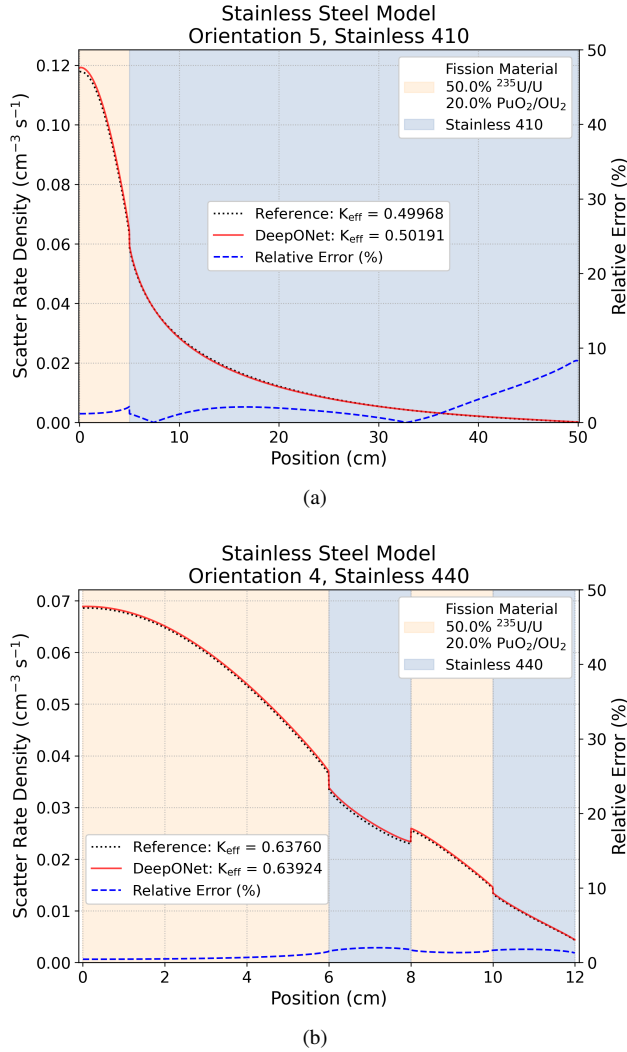


**Figure 4:** Test results for the stainless steel DeepONet model. Both tests use a 400-series stainless steel with either (a) 45 cm of reflector material or (b) multiple shells of stainless steel.

These include results for 400-series stainless steels with large reflecting regions, such as the case in Fig. 4(a), which uses a 45 cm stainless steel 410 reflector. The  $k$ -effective error was 223 pcm, and the only visible difference in the scattering rate density is at the sphere center. In addition, while the stainless steel 440 test case did not perform as well as the others for large reflecting regions, the performance improved when the reflector size was reduced, as shown in Fig. 4(b). In this case, there were multiple shells of stainless steel, which differs from the orientations used to generate the scalar flux training data. This demonstrates the generalizability and versatility of using DeepONets to predict the matrix-vector product for a family of reflector materials.

### 4.3. Simulations with Multiple Models

In Sections 4.1 and 4.2, each DeepONet model replaced only one scattering matrix in the transport test simulations. The models are now combined in one simulation to test

realistic use cases and to demonstrate the full capabilities of using DeepONets for matrix-vector predictions. These multiple-model simulations were tested on the six material orientations in Fig. 1, using the nine stainless steel compositions in Table 1 for the first five orientations. Orientation 6 uses a fixed set of stainless steel compositions. The fissionable material was also varied between two combinations,  $w_{235} = 10\%$  with  $\text{PuO}_2 = 5\%$  and  $w_{235} = 80\%$  with  $\text{PuO}_2 = 50\%$ , to demonstrate effectiveness at both low and high enrichment.

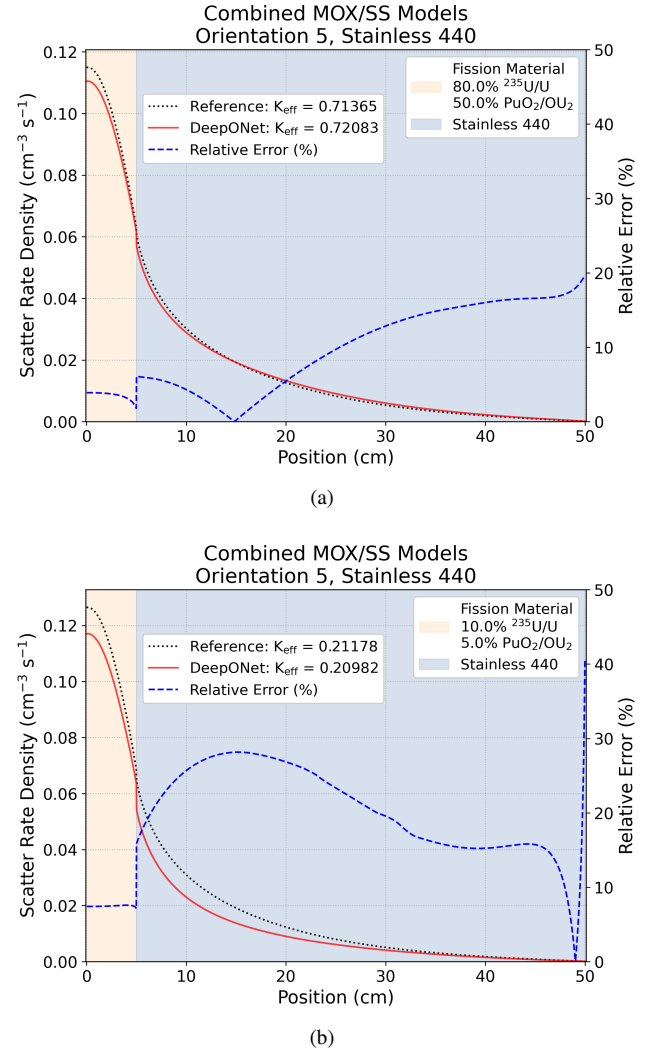


**Figure 5:** Test results for the combined models, showing (a) the highest  $k$ -effective error and (b) the highest RMS scattering rate density error. Both of these examples have a large reflector region and use stainless steel 440. The combination of a higher carbon percentage and a large reflector region may cause the degradation of the stainless steel model.

The 92 test cases showed an average  $k$ -effective error of 129.7 pcm and an average RMS scattering rate density error of  $5.614 \times 10^{-7} \text{ cm}^{-3} \text{s}^{-1}$ . The worst-case  $k$ -effective error is shown in Fig. 5(a), with an absolute  $k$ -effective difference of 718.1 pcm; this case uses stainless steel 440 and a 45 cm reflector region. The maximum RMS scattering rate

density error among the test cases,  $3.234 \times 10^{-6} \text{ cm}^{-3} \text{ s}^{-1}$ , also occurred for a case using stainless steel 440 and a 45 cm reflector region, as seen in Fig. 5(b). Although these results use different fissionable material compositions, the deficiencies are likely due to the combination of the material layout and the stainless steel 440 composition. As in Fig. 3, the combination of a large reflector region and the higher carbon content of stainless steel 440 relative to the other steels in Table 1 likely caused these poor results. Comparing the two results, the scattering rate density error is higher in Fig. 5(b), which is attributed to that case being more deeply subcritical than the case in Fig. 5(a).

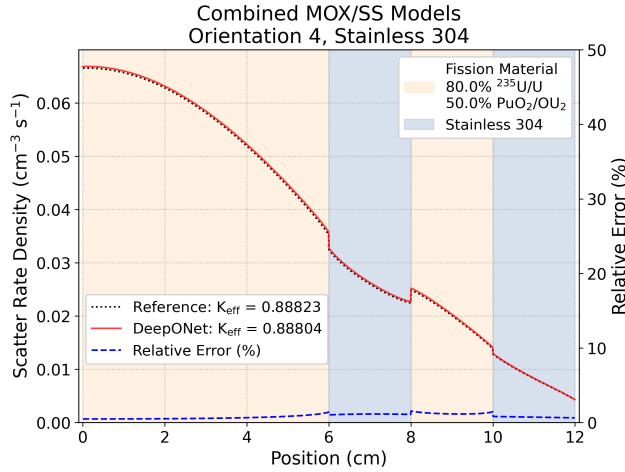


**Figure 6:** Test result for the combined models with a multiple-shell layout. This demonstrates the flexibility of DeepONet models for material orientations not included in the training data.

Overall, the combined fissionable material and stainless steel DeepONet models performed well for every other material composition and layout. This is exemplified in Figs. 6 and 7. Figure 6 shows the multiple-shell layout using stainless steel 304 and a highly enriched fissionable material. This resulted in an absolute  $k$ -effective error of 19 pcm and a maximum relative scattering rate density error of 1.5%. The test case in Fig. 7 used 20 cm of reflector material consisting of two different stainless steels and a low-enriched fissionable material. The  $k$ -effective error was 23 pcm, and the maximum relative scattering rate density error was 1.2%.

#### 4.4. Memory Reduction and Simulation Speedup

The primary benefit of using DeepONets instead of standard matrix-vector multiplication in the neutron transport equation is that they eliminate the need to store large scattering matrices. The fissionable material and stainless steel DeepONet models effectively predict the matrix-vector product while introducing minimal error. To ensure that reaction rates are preserved, the total scattering vector of size  $G$  is stored. Assuming that the size of the DeepONet models is negligible (under 500 KB), the storage requirement is reduced from multiple  $G \times G$  matrices to vectors of size  $G$ . The current simulations used  $G = 70$ , meaning the memory

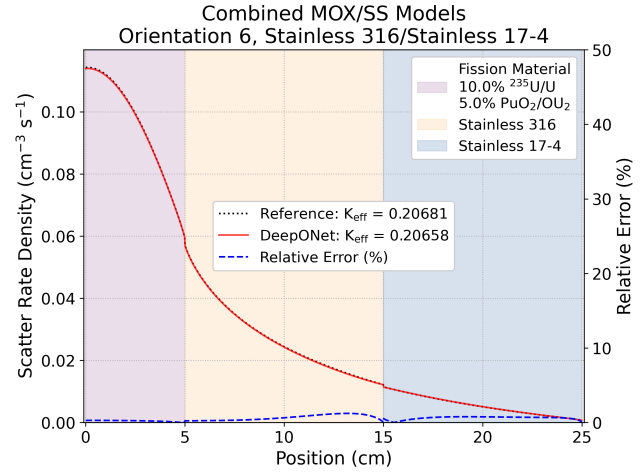


**Figure 7:** Test result for the combined models using two different stainless steels in the same problem. The stainless steel DeepONet accurately predicts the matrix-vector product for different stainless steel compositions.

**Table 2**

DeepONet speedup per power iteration (averaged over seven repetitions) compared to standard source iteration. Orientations 1 through 5 used stainless steel 304, and orientation 6 used stainless steels 316 and 17-4. Results are shown for both high- and low-enriched fissionable materials.

| Orientation | Speedup Factor               |                             |
|-------------|------------------------------|-----------------------------|
|             | High Enrichment <sup>a</sup> | Low Enrichment <sup>b</sup> |
| 1           | 9.96×                        | 11.38×                      |
| 2           | 15.94×                       | 15.73×                      |
| 3           | 13.19×                       | 13.61×                      |
| 4           | 6.80×                        | 9.18×                       |
| 5           | 16.15×                       | 16.09×                      |
| 6           | 14.04×                       | 15.10×                      |

<sup>a</sup>80%  $^{235}\text{U}/\text{U}$ , 50%  $\text{PuO}_2/\text{UO}_2$

<sup>b</sup>10%  $^{235}\text{U}/\text{U}$ , 5%  $\text{PuO}_2/\text{UO}_2$

footprint is reduced by a factor of 70, to approximately 1.43% of the original size. Considering that there are about 2.8 million possible stainless steel compositions when varying weight percentages in whole-number increments, this compression substantially reduces memory requirements without significantly degrading simulation accuracy.

A second benefit of using DeepONets instead of matrix-vector multiplication in the neutron transport equation is the speedup relative to standard source iteration. This acceleration stems from how the scattering term is computed in the source iteration. In standard multigroup calculations, computing the scattering term requires iterating over the energy groups, with each group requiring sweeps over the discrete angles until convergence, which is computationally expensive. In contrast, the DeepONet directly outputs the energy spectrum of the scattering source for each spatial cell, effectively achieving convergence in a single iteration. The use of DeepONets results in a speedup of 6.80 to 16.15

**Table 3**

Summary of DeepONet test results:  $k_{\text{eff}}$  absolute errors (pcm) and scattering rate density (SRD) root mean squared errors ( $\text{cm}^{-3}\text{s}^{-1}$ ) for each model configuration.

| Model       | Metric                 | Average                | Maximum                |
|-------------|------------------------|------------------------|------------------------|
| Fissionable | $k_{\text{eff}}$ (pcm) | 23.7                   | 116.1                  |
|             | SRD RMSE               | $4.024 \times 10^{-8}$ | $2.229 \times 10^{-7}$ |
| Reflector   | $k_{\text{eff}}$ (pcm) | 167.3                  | 623.4                  |
|             | SRD RMSE               | $5.223 \times 10^{-7}$ | $2.579 \times 10^{-6}$ |
| Both        | $k_{\text{eff}}$ (pcm) | 129.7                  | 718.1                  |
|             | SRD RMSE               | $5.614 \times 10^{-7}$ | $3.234 \times 10^{-6}$ |

times per power iteration, averaged over seven repetitions, as shown in Table 2 for the different material layouts and enrichment levels. The stainless steel composition was held constant, as the results did not change when the composition was modified. The variation among orientations is due to the reflector thickness, with greater speedups for thicker reflectors. Both high- and low-enriched fissionable materials were included to show that the speedup does not differ markedly with enrichment.

## 5. Conclusion and Future Work

The application of deep operator networks (DeepONets) to replace the scattering term matrix-vector multiplication is beneficial to neutron transport solvers. This work provides a new method of directly using machine learning models to predict the matrix-vector product without requiring large scattering cross section matrices. In this method, the trunk network takes material weight percentages as inputs, and the branch network takes the scalar flux. The output of the DeepONet is an approximation of the matrix-vector product of the scattering term. This prediction model is directly inserted into a neutron transport solver to approximate the neutron flux and  $k$ -effective value for one-dimensional spherical criticality problems. To help preserve reaction rates, the total scattering cross section is retained to compute a scaling factor.

Structuring the scattering term as a DeepONet allows greater generalization across material compositions, each of which corresponds to a different scattering matrix. To demonstrate this, two different DeepONet models were presented: a fissionable material model and a stainless steel model. The trunk network of the fissionable material model used the percentage of  $^{235}\text{U}$  in uranium and the percentage of  $\text{PuO}_2$  relative to  $\text{UO}_2$  as inputs. The stainless steel model used the weight percentages of eight common elements found in stainless steel (iron, chromium, carbon, silicon, manganese, molybdenum, nickel, and copper) as inputs. This structure allows simulations with many different material compositions without requiring additional training, which was necessary with other neural network structures [22].

The trained models were shown to perform well on an assortment of test cases that varied the material layout and

composition, as summarized in Table 3. Both individually and combined in test criticality simulations, the DeepONets accurately predicted the scattering rate density and  $k_{\text{eff}}$  values. They achieved these results while using 70 times less memory and converging approximately 7 to 16 times faster than the standard multigroup solver.

The overall computational cost associated with data collection and model training was modest. The scalar flux was collected from five simulations (three fissionable and two reflector), while the material compositions were sampled. Training both models required less than two hours on an Apple MacBook Pro equipped with an Apple M3 Max chip (16-core CPU) and 64 GB of unified memory running Python 3.12. This cost is small considering the number of material compositions the DeepONet models can accurately predict.

A potential avenue for future work is expanding the trunk network inputs. Additional elements can be added to the trunk networks, such as beryllium, zirconium, and depleted uranium for the reflector material, or additional isotopes of uranium and plutonium for the fissionable material model. Expanding the inputs to include information about anisotropic scattering and temperature dependence would also be useful.

Additional methods could also expand the functionality of DeepONets for neutron transport simulations. One such method is the variational Bayes deep operator network [3], which could potentially improve the accuracy of DeepONets for simulations with large reflector regions. Verification and validation methods for the models could also be explored, which would allow users to run simulations with confidence that the model is sufficiently accurate. Lastly, the DeepONet models presented here were applied only to the scattering term, but they could easily be extended to the fission term as well.

## Acknowledgments

This work was supported by the U.S. Department of Energy through Los Alamos National Laboratory (LANL). LANL is operated by Triad National Security, LLC, for the National Nuclear Security Administration of the U.S. Department of Energy (contract no. 89233218CNA000001).

## CRedit authorship contribution statement

**Ben Whewell:** Data curation, formal analysis, investigation, methodology, software, validation, visualization, writing - original draft, writing - review and editing. **Ryan G. McClarren:** Conceptualization, writing - review and editing. **Nathan Gibson:** Funding acquisition, writing - review and editing. **Ajeeta Khatriwada:** Writing - review and editing.

## References

- [1] Chandrasekhar, S., 1950. Radiative Transfer. Dover, New York.

- [2] Cherezov, A., Jang, J., Lee, D., 2020. A PCA compression method for reactor core transient multiphysics simulation. *Progress in Nuclear Energy* 128, 103441.
- [3] Garg, S., Chakraborty, S., 2023. Vb-deeponet: A bayesian operator learning framework for uncertainty quantification. *Engineering Applications of Artificial Intelligence* 118, 105685.
- [4] Hastie, T., Tibshirani, R., Friedman, J., et al., 2009. *The Elements of Statistical Learning*. Springer.
- [5] Humbird, K.D., Peterson, J.L., McClarren, R.G., 2018. Deep neural network initialization with decision trees. *Institute of Electrical and Electronics Engineers Transactions on Neural Networks and Learning Systems* 30, 1286–1295.
- [6] Jeon, B.K., 2021. A Generalized Multigroup Cross Section Generation Procedure for High-fidelity Reactor Simulation of Advanced Reactor Designs. Ph.D. thesis. University of Michigan.
- [7] Kim, K.S., Williams, M.L., Wiarda, D., Clarno, K.T., 2019. Development of the multigroup cross section library for the CASL neutronics simulator MPACT: Method and procedure. *Annals of Nuclear Energy* 133, 46–58.
- [8] Kingma, D.P., Ba, J., 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.
- [9] Laboure, V.M., McClarren, R.G., Wang, Y., 2017. Globally conservative, hybrid self-adjoint angular flux and least-squares method compatible with voids. *Nuclear Science and Engineering* 185, 294–306.
- [10] Lewis, E.E., Miller, W.F., 1993. *Computational Methods of Neutron Transport*. American Nuclear Society Scientific Publications.
- [11] Lu, L., Jin, P., Pang, G., Zhang, Z., Karniadakis, G.E., 2021. Learning nonlinear operators via deeponet based on the universal approximation theorem of operators. *Nature machine intelligence* 3, 218–229.
- [12] Macfarlane, R., Muir, D.W., Boicourt, R.M., Kahler III, A.C., Conlin, J.L., 2017. The NJOY nuclear data processing system, version 2016. Technical Report. Los Alamos National Lab, Los Alamos, NM (USA).
- [13] Mattingly, J., Yi, C., Sjoden, G., Courau, T., 2009. *Computational Optimized Multi-group Cross Section Data Collapsing using the YGROUP Code*. Technical Report. Sandia National Lab, Albuquerque, NM (USA).
- [14] McClarren, R.G., Haut, T.S., 2022. Data-driven acceleration of thermal radiation transfer calculations with the dynamic mode decomposition and a sequential singular value decomposition. *Journal of Computational Physics* 448, 110756.
- [15] Peng, Z., McClarren, R.G., Frank, M., 2020. A low-rank method for two-dimensional time-dependent radiation transport calculations. *Journal of Computational Physics* 421, 109735.
- [16] Raissi, M., Perdikaris, P., Karniadakis, G., 2019. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics* 378, 686–707.
- [17] Rucker, T., Johnson Jr, C., 1998. Relationship between isotopic uranium activities and total uranium at various uranium enrichments. *Journal of Radioanalytical and Nuclear Chemistry* 235, 47–52.
- [18] Steels, A., 2013. *Stainless steel grade datasheets*. Atlas Steels Technical Department: Melbourne, Australia.
- [19] Sun, Y., Yang, J., Wang, Y., Li, Z., Ma, Y., 2020. A POD reduced-order model for resolving the neutron transport problems of nuclear reactor. *Annals of Nuclear Energy* 149, 107799.
- [20] Tano, M.E., Ragusa, J.C., 2021. Sweep-Net: An artificial neural network for radiation transport solves. *Journal of Computational Physics* 426, 109757.
- [21] Was, G.S., Ukai, S., 2019. Chapter 8 - austenitic stainless steels, in: Odette, G.R., Zinkle, S.J. (Eds.), *Structural Alloys for Nuclear Energy Applications*. Elsevier, Boston, pp. 293–347.
- [22] Whewell, B., McClarren, R.G., 2022. Data reduction in deterministic neutron transport calculations using machine learning. *Annals of Nuclear Energy* 176, 109276.
- [23] Whewell, B., McClarren, R.G., 2026. Collision-based hybrid method for two-dimensional neutron transport problems. *Nuclear Science and*