Physics Informed Neural Networks for Simulating Radiative Transfer

S. Mishra and R. Molinaro

Research Report No. 2020-62
September 2020

Physics Informed Neural Networks for Simulating Radiative Transfer

Siddhartha Mishra, Roberto Molinaro *

September 25, 2020

Abstract

We propose a novel machine learning algorithm for simulating radiative transfer. Our algorithm is based on physics informed neural networks (PINNs), which are trained by minimizing the residual of the underlying radiative tranfer equations. We present extensive experiments and theoretical error estimates to demonstrate that PINNs provide a very easy to implement, fast, robust and accurate method for simulating radiative transfer. We also present a PINN based algorithm for simulating inverse problems for radiative transfer efficiently.

1 Introduction

The study of radiative transfer is of vital importance in many fields of science and engineering including astrophysics, climate dynamics, meteorology, nuclear engineering and medical imaging [30]. The fundamental equation describing radiative transfer is a *linear partial integro-differential equation*, termed as the *radiative transfer equation*. Under the assumption of a static underlying medium, it has the following form [30],

$$\frac{1}{c}u_t + \omega \cdot \nabla_x u + ku + \sigma\left(u - \frac{1}{s_d}\int\limits_{\Lambda}\int\limits_{S} \Phi(\omega,\omega',\nu,\nu')u(t,x,\omega',\nu')d\omega' d\nu'\right) = f, \tag{1.1}$$

with time variable $t \in [0,T]$, space variable $x \in D \subset \mathbb{R}^d$ (and $D_T = [0,T] \times D$), *angle* $\omega \in S = \mathbb{S}^{d-1}$ i.e. the d-dimensional sphere and *frequency* (or group energy) $\nu \in \Lambda \subset \mathbb{R}$. The constants in (1.1) are the speed of light c and the surface area s_d of the d-dimensional unit sphere. The unknown of interest in (1.1) is the so-called *radiative intensity* $u : D_T \times S \times \Lambda \mapsto \mathbb{R}$, while $k = k(x,\nu) : D \times \Lambda \mapsto \mathbb{R}_+$ is the *absorption coefficient* and $\sigma = \sigma(x,\nu) : D \times \Lambda \mapsto \mathbb{R}_+$ is the *scattering coefficient*. The integral term in (1.1) involves the so-called *scattering kernel* $\Phi : S \times S \times \Lambda \times \Lambda \mapsto \mathbb{R}$, which is normalized as $\int_{S\times\Lambda} \Phi(\cdot,\ \omega',\ \cdot,\ \nu')d\omega' d\nu' = 1$, in order to account for the conservation of photons during scattering. The dynamics of radiative transfer are driven by a source (emission) term $f = f(x,\nu) : D \times \Lambda \mapsto \mathbb{R}$.

Although the radiative transfer equation (1.1) is linear, explicit solution formulas are only available in very special cases [30]. Hence, numerical methods are essential for the simulation of the radiative intensity in (1.1). However, the design of efficient numerical methods is considered to be very challenging [15, 16, 30]. This is on account of the *high-dimensionality* of the radiative transfer equation (1.1), where in the most general case of three space dimensions, the radiative intensity is a function of 7 variables (4 for space-time, 2 for angle and 1 for frequency). Traditional grid-based numerical methods such as finite elements or finite differences, which involve N^ℓ degrees of freedom (for ℓ dimensions, with N being the number of points in each dimension), require massive computational resources to be able to simulate radiative transfer accurately [15, 16]. Moreover in practice, one has to encounter media with very different optical properties characterized by different scales in the absorption and scattering coefficients and in the emission term in (1.1), which further complicates the design of robust and efficient numerical methods.

*Seminar for Applied Mathematics (SAM), D-Math
ETH Zürich, Rämistrasse 101.

In spite of the aforementioned challenges, several types of numerical methods have been proposed in the literature for simulating radiative transfer, see [9, 12, 15, 30] and references therein for a detalied overview. These include Monte Carlo ray-tracing type particle methods [41], which do not suffer from the curse of dimensionality and are easy to parallelize but are characterized by slow convergence (with respect to number of particles) and are mostly limited to media with fairly uniform optical properties. *Discrete Ordinate Methods* (DOM), are based on the discretization of the angular domain S with a number of fixed directions and the resulting systems of spatio-temporal PDEs is solved by finite element or finite difference methods. Although easy to implement, these methods can be very expensive and also suffer from the so-called ray effects in optically thin media [20]. Spherical harmonics, based on a series expansion in the angle, have been widely used in radiative transfer [30]. Although shown to exhibit spectral convergence for smooth solutions [13], these methods are well-known to still suffer from the curse of dimensionality, see [12, 31]. A particular variant of the spherical harmonics, the so-called P_1 method, is an example of a class of flux limited diffusion methods [9], which are widely used for optically thick media.

Moment based methods lead to another class of numerical methods for simulating radiative transfer, see [9] and references therein. For these methods, one derives a PDE for the so-called *incident radiation* by integrating (1.1) over the angular domain S. The evolution of the incident radiation is determined by the *heat (radiation) flux*, which is also the first angular moment. The evolution of the heat flux has to be determined from the second angular moment of u, which is termed as the *pressure tensor*. These hierarchy of moments have to be closed by suitable closure relations (see [9] and references therein) and the resulting PDEs are discretized by finite elements or finite differences. These moment based methods can lead to inaccurate approximation of the incident radiation, particularly when suitable closures are not available. Finally in recent years, several attempts have been made to design efficient finite element methods for the radiative transfer equation, such as those based on sparse grids [43] or sparse tensor product finite element spaces [15, 43]. Although these methods can alleviate the curse of dimensionality in certain cases, they are rather complicated to implement and can still be computationally expensive, particularly when higher-order elements are used [12].

Summarizing the above discussion, it is fair to conclude that all the proposed methods have some deficiencies, in particular in their computational cost for simulating realistic problems. Thus, there is a pressing need to design a numerical method that is accurate, fast (in terms of computational time), easy to use and able to deal with the high dimensions and optical heterogeneity of the underlying radiative transfer equation (1.1). We aim to propose such a numerical method in this article.

Our proposed numerical method is based on deep neural networks [10], i.e. functions formed by concatenated compositions of affine transformations and scalar non-linear activation functions. Deep neural networks have been extremely successful at diverse tasks in science and engineering [21] such as at image and text classification, computer vision, text and speech recognition, autonomous systems and robotics, game intelligence and even protein folding [7]. They are being increasingly used in the context of scientific computing, particularly for different aspects of numerical solutions of PDEs [6, 14, 24, 25] and references therein.

As deep neural networks possess the so-called *universal approximation property* or ability to accurately approximate any continuous (even measurable) function [1], they can be used as ansatz (search) spaces for solutions of PDEs. This property lays the foundation for the so-called *Physics informed neural networks* (PINNs) which collocate the PDE residual on *training points* of the approximating deep neural network. First proposed in [18, 19], PINNs been revived and developed in significantly greater detail recently in the pioneering contributions of Karniadakis and collaborators. PINNs have been successfully applied to simulate a variety of forward and inverse problems for PDEs, see [5, 22, 23, 26, 33, 36, 37, 38] and references therein.

In recent papers [27] (for the forward problem) and [28] (for the inverse problem), the authors analyzed PINNs and provide a rigorous explanation for the efficiency of PINNs, based on stability of the underlying PDEs. A surprising observation in [27, 28] was the ability of PINNs to overcome the curse of dimensionality, at least for some PDEs. This observation, together with the well-documented ability of PINNs to approximate PDEs is a starting point of this paper where we adapt PINNs to solve the radiative transfer equation (1.1). The main contributions of the current paper are as follows,

- We present a novel algorithm for approximating the radiative transfer equation (1.1) in a very general setting. Our algorithm is based on suitable physics informed neural networks (PINNs).

- We analyze the proposed algorithm by rigorously proving an estimate on the so-called generalization error of the PINN. This estimate shows that as long as the PINN is trained well, it approximates the solution of (1.1) to high accuracy.
- We present a suite of numerical experiments to illustrate the accuracy and efficiency of the proposed algorithm.
- A major advantage of PINNs is their ability to approximate inverse problems (with the same level of complexity as the forward problem). Hence, we will also modify PINNs to approximate an inverse problem for radiative transfer, namely determining the unknown absorption or scattering coefficients in (1.1) from measurements of moments of the radiative intensity.

Thus, we present a novel, fast, robust, accurate and easy to code and implement algorithm for simulating the general form of the radiative transfer equations (1.1) and provide analysis and numerical experiments to demonstrate that this algorithm efficiently approximates both forward and inverse problems for radiative transfer. The rest of the paper is organized as follows, in section 2, we describe the PINNs algorithm and provide an estimate on the underlying generalization error. Numerical experiments are presented in section 3, PINNs for inverse problems are described in section 4 and the proposed method and results are discussed in section 5.

2 Physics informed neural networks for approximating (1.1)

In this section, we describe the PINNs algorithm for simulating radiative transfer. We start by elaborating on the underlying PDE (1.1).

2.1 The model.

We model radiative transfer in a static medium by the evolution equation (1.1) for the radiative intensity u. This partial integro-differential equation is supplemented with the initial condition,

$$u(0, x, \omega, \nu) = u_0(x, \omega, \nu), \quad (x, \omega, \nu) \in D \times S \times \Lambda, \tag{2.1}$$

for some initial datum $u_0 : D \times S \times \Lambda \mapsto \mathbb{R}$.

Given that the radiative transfer equation (1.1) is a *transport equation*, the boundary conditions are imposed on the so-called *inflow boundary* given by,

$$\Gamma_- = \{(t, x, \omega, \nu) \in [0, T] \times \partial D \times S \times \Lambda : \omega \cdot n(x) < 0\} \tag{2.2}$$

with $n(x)$ denoting the unit outward normal at any point $x \in \partial D$ (the boundary of the spatial domain D). We specific the following boundary condition,

$$u(t, x, \omega, \nu) = u_b(t, x, \omega, \nu), \quad (t, x, \omega, \nu) \in \Gamma_-, \tag{2.3}$$

for some boundary datum $u_b : \Gamma_- \mapsto \mathbb{R}$.

Given that the radiative intensity is a function of $2d + 1$-variables, it is essential to find suitable low-dimensional functionals (observables) to visualize and interpret it. To this end, one often considers physically interesting angular-moments such as the *incident radiation* (zeroth angular moment) and *heat flux* (first angular moment) given by,

$$G(t, x, \nu) = \int\limits_S u(t, x, \omega, \nu) d\omega \tag{2.4}$$

$$F(t, x, \nu) = \int\limits_S u(t, x, \omega, \nu) \omega d\omega \tag{2.5}$$

We note that for many applications of radiative transfer, it is common to consider the steady (time-independent) version of the radiative transfer equation (1.1), which formally results from setting $c \to \infty$ and dropping the time-derivative term in the left hand side of (1.1).

2.2 Quadrature rules and Training points

Quadrature i.e numerical approximation of integrals, is essential for simulating the radiative transfer equation (1.1) with PINNs. It is needed for approximating the integral with the scattering kernel in (1.1). Moreover, we follow [27], where quadrature points were used as training points for PINNs.

Given any domain $\mathbb{D}$ and an integrable function $g : \mathbb{D} \mapsto \mathbb{R}$, we need to specify quadrature points $z_i \in \mathbb{D}$ for $1 \leqslant i \leqslant N$, and quadrature weights w_i in order to perform the following approximation,

$$\int_{\mathbb{D}} g(z)dz \approx \sum_{i=1}^{N} w_i g(z_i). \tag{2.6}$$

For our specific integrals, we consider *Gauss-Legendre* quadrature rules [42] for approximating the scattering kernel integral in (1.1). To this end, we choose points $z_i^S = (\omega_i^S, \nu_i^S)$, for $1 \leqslant i \leqslant N_S$, with $\omega_i^S \in S$ and $\nu_i^S \in \Lambda$ as the Gauss-Legendre quadrature points and the weights w_i^S are the corresponding quadrature weights for a Gauss-Legendre quadrature rule of order $s \geqslant 1$.

We also need the following training points for the PINNs algorithm,

2.2.1 Interior training points.

We set $\mathcal{S}_{int} = \{z_j^{int}\}$, for $1 \leqslant j \leqslant N_{int}$, and $z_j^{int} = (t_j^{int}, x_j^{int}, \omega_j^{int}, \nu_j^{int})$ with $t_j^{int} \in [0,T], x_j^{int} \in D, \omega_j^{int} \in S, \nu_j^{int} \in \Lambda$, for all j. These points are the quadrature points of a suitable quadrature rule with weights w_j^{int}.

If the underlying spatial domain $D \subset \mathbb{R}^d$ can be mapped to a d-dimensional rectangle, either entirely or in patches, then we can set the training points z_j^{int} as a low-discrepancy Sobol sequence [40] in $[0,1]^{2d+1}$, by rescaling the relevant domains. Sobol sequences arise in the context of Quasi-Monte Carlo integration [2] and the corresponding quadrature weights are $w_j^{int} \equiv \frac{1}{N_{int}}$, for all j. Note that the QMC quadrature rule does not suffer from the curse of dimensionality (see section 2.5 for details). In case the geometry of the domain is very complicated, one has simply choose random points, independent and identically distributed with the underlying uniform distribution, as training points.

2.2.2 Temporal boundary training points.

We denote $\mathcal{S}_{tb} = \{z_j^{tb}\}$, for $1 \leqslant j \leqslant N_{tb}$, and $z_j^{tb} = (x_j^{tb}, \omega_j^{tb}, \nu_j^{tb})$ with $x_j^{tb} \in D, \omega_j^{tb} \in S, \nu_j^{tb} \in \Lambda$, for all j. These points are the quadrature points of a suitable quadrature rule with weights w_j^{tb}. We can choose Sobol points for logically rectangular domains D or random points to constitute this training set.

2.2.3 Spatial boundary training points.

We denote $\mathcal{S}_{sb} = \{z_j^{sb}\}$, for $1 \leqslant j \leqslant N_{sb}$, and $z_j^{sb} = (t_j^{tb}, x_j^{tb}, \omega_j^{tb}, \nu_j^{tb})$ with $t_j^{tb} \in [0,T], x_j^{tb} \in \partial D, \omega_j^{tb} \in S, \nu_j^{tb} \in \Lambda$, for all j. These points are the quadrature points of a suitable quadrature rule with weights w_j^{sb}. As before, we can choose Sobol points for logically rectangular domains D or random points to constitute this training set.

2.3 Neural Networks

PINNs are neural networks i.e. given an input $y = (t, x, \omega, \nu) \in \mathbb{D} = D_T \times S \times \Lambda$, a feedforward neural network (also termed as a multi-layer perceptron), shown in figure 1, transforms it to an output, through a layers of units (neurons) which compose of either affine-linear maps between units (in successive layers) or scalar non-linear activation functions within layers [10], resulting in the representation,

$$\mathbf{u}_\theta(y) = C_K \circ A \circ C_{K-1} \ldots\ldots\ldots \circ A \circ C_2 \circ A \circ C_1(y). \tag{2.7}$$

Here, $\circ$ refers to the composition of functions and A is a scalar (non-linear) activation function (applied to vectors componentwise). A large variety of activation functions have been considered in the machine learning literature [10]. Popular choices for the activation function A in (2.7) include the sigmoid function, the hyperbolic tangent function and the *ReLU* function.

The affine map in the k-layer is given by,

$$C_k z_k = W_k z_k + b_k, \quad \text{for } W_k \in \mathbb{R}^{d_{k+1} \times d_k}, z_k \in \mathbb{R}^{d_k}, b_k \in \mathbb{R}^{d_{k+1}}. \tag{2.8}$$

For consistency of notation, we set $d_1 = \bar{d} = 2d + 1$, for d-space dimensions and $d_K = 1$.

Thus in the terminology of machine learning (see also figure 1), our neural network (2.7) consists of an input layer, an output layer and $(K-1)$ hidden layers for some $1 < K \in \mathbb{N}$. The k-th hidden layer (with d_k neurons) is given an input vector $z_k \in \mathbb{R}^{d_k}$ and transforms it first by an affine linear map C_k (2.8) and then by a nonlinear (component wise) activation σ. A straightforward addition shows that our network contains $\left(2d + 2 + \sum\limits_{k=2}^{K-1} d_k\right)$ neurons. We also denote,

$$\theta = \{W_k, b_k\}, \; \theta_W = \{W_k\} \quad \forall\, 1 \leqslant k \leqslant K, \tag{2.9}$$

to be the concatenated set of (tunable) weights for our network. It is straightforward to check that $\theta \in \Theta \subset \mathbb{R}^M$ with

$$M = \sum_{k=1}^{K-1} (d_k + 1) d_{k+1}. \tag{2.10}$$

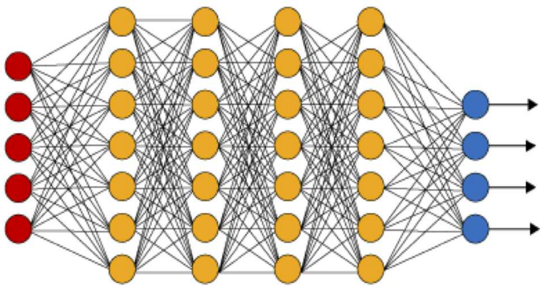


Figure 1: An illustration of a (fully connected) deep neural network. The red neurons represent the inputs to the network and the blue neurons denote the output layer. They are connected by hidden layers with yellow neurons. Each hidden unit (neuron) is connected by affine linear maps between units in different layers and then with nonlinear (scalar) activation functions within units.

2.4 Training PINNs: Loss functions and optimization

The neural network u_θ (2.7) depends on the tuning parameter $\theta \in \Theta$ of weights and biases. Within the standard paradigm of *deep learning* [10], one *trains* the network by finding tuning parameters θ such that the loss (error, mismatch, regret) between the neural network and the underlying target is minimized. Our target is the solution u of the radiative transfer equation (1.1) and we wish to find the tuning parameters θ such that the resulting neural network u_θ approximates u.

To do so, we follow [19, 36, 27] and define the following *PDE residual* $\mathcal{R}_{int,\theta} = \mathcal{R}_{int,\theta}(t, x, \omega, \nu)$, for all $(t, x, \omega, \nu) \in D_T \times S \times \Lambda$,

$$\mathcal{R}_{int,\theta} := \frac{1}{c}\partial_t u_\theta + \omega \cdot \nabla_x u_\theta + k u_\theta + \sigma\left(u_\theta - \frac{1}{s_d}\sum_{i=1}^{N_S} w_i^S \Phi(\omega, \omega_i^S, \nu, \nu_i^S) u_\theta(t, x, \omega_i^S, \nu_i^S)\right) - f. \tag{2.11}$$

Here, k, σ, f are defined from (1.1) and (ω_i^S, ν_i^S) are the Gauss-Legendre quadrature points, with quadrature weights w_i of order S.

We also need the following residuals for the initial and boundary conditions,

$$\begin{aligned} \mathcal{R}_{tb} = \mathcal{R}_{tb,\theta} &:= u_\theta - u_0, \qquad \forall (x, \omega, \nu) \in D \times S \times \Lambda, \\ \mathcal{R}_{sb} = \mathcal{R}_{sb,\theta} &:= u_\theta - u_b, \qquad \forall (t, x, \omega, \nu) \in \Gamma_-. \end{aligned} \tag{2.12}$$

The strategy of PINNs, following [36, 27], is to minimize the *residuals* (2.11) (2.12), simultaneously over the admissible set of tuning parameters $\theta \in \Theta$ i.e

$$\text{Find } \theta^* \in \Theta : \quad \theta^* = \arg\min_{\theta\in\Theta} \left(\|\mathcal{R}_{int,\theta}\|^2_{L^2(D_T\times S\times\Lambda)} + \|\mathcal{R}_{sb,\theta}\|^2_{L^2(\Gamma_-)} + \|\mathcal{R}_{tb,\theta}\|^2_{L^2(D\times S\times\Lambda)} \right). \tag{2.13}$$

However, the L^2 norms in (2.13) involve integrals that cannot be computed exactly and need to be approximated by suitable quadrature rules. It is exactly the place to recall the different training sets, introduced in section 2.2. As these precisely correspond to the quadrature points of an underlying quadrature rule, we approximate the integrals in (2.13) with the corresponding quadrature rule to define the following loss function,

$$J(\theta) := \sum_{j=1}^{N_{sb}} w_j^{sb} |\mathcal{R}_{sb,\theta}(z_j^{sb})|^2 + \sum_{j=1}^{N_{tb}} w_j^{tb} |\mathcal{R}_{tb,\theta}(z_j^{tb})|^2 + \lambda \sum_{j=1}^{N_{int}} w_j^{int} |\mathcal{R}_{int,\theta}(z_j^{int})|^2 \tag{2.14}$$

with the residuals $\mathcal{R}_{sb}, \mathcal{R}_{tb}$ and $\mathcal{R}_{int}$ defined in (2.12), (2.11), and w^{sb}, z^{sb}, w^{sb}, z^{sb}, w^{int}, z^{int} being the quadrature weights and training points, defined in section 2.2. Furthermore, λ is a hyperparameter for balancing the residuals, on account of the PDE and the initial and boundary data, respectively.

It is common in machine learning [10] to regularize the minimization problem for the loss function i.e we seek to find,

$$\theta^* = \arg\min_{\theta\in\Theta} \left(J(\theta) + \lambda_{reg} J_{reg}(\theta) \right). \tag{2.15}$$

Here, $J_{reg} : \Theta \to \mathbb{R}$ is a *weight regularization* (penalization) term. A popular choice is to set $J_{reg}(\theta) = \|\theta_W\|_q^q$ for either $q = 1$ (to induce sparsity) or $q = 2$. The parameter $0 \leqslant \lambda_{reg} \ll 1$ balances the regularization term with the actual loss J (2.14).

The above minimization problem amounts to finding a minimum of a possibly non-convex function over a subset of $\mathbb{R}^M$ for possibly very large M. We will follow standard practice in machine learning and solving this minimization problem approximately by either (first-order) stochastic gradient descent methods such as ADAM [17] or even higher-order optimization methods such as different variants of the LBFGS algorithm [8].

For notational simplicity, we denote the (approximate, local) minimum in (2.15) as θ^* and the underlying deep neural network $u^* = u_{\theta^*}$ will be our physics-informed neural network (PINN) approximation for the solution u of the PDE (1.1). We summarize the PINN algorithm for approximating radiative transfer below,

Algorithm 2.1. *Finding a physics informed neural network (PINN) to approximate the radiative intensity u solving* (1.1).

Inputs: *Underlying domain $D_T \times S \times \Lambda$, coefficients and data for the radiative transfer equation* (1.1), *quadrature points and weights for underlying quadrature rules, non-convex gradient based optimization algorithms.*

Goal: *Find PINN $u^* = u_{\theta^*}$ for approximating the solution u of* (1.1) .

Step 1: *Choose the training sets as described in section 2.2.*

Step 2: *For an initial value of the weight vector $\overline{\theta} \in \Theta$, evaluate the neural network $u_{\overline{\theta}}$* (2.7), *the PDE residual* (2.11), *the boundary residuals* (2.12), *the loss function* (2.15) *and its gradients to initialize the underlying optimization algorithm.*

Step 3: *Run the optimization algorithm till an approximate local minimum θ^* of* (2.15) *is reached. The map $u^* = u_{\theta^*}$ is the desired PINN for approximating the solution u of the radiative transfer equation.*

2.5 Estimates on the generalization error

For the sake of definiteness and simplicity, we consider the spatial domain as $D = [0,1]^d$, with d being the spatial dimension. Any rectangular domain $\prod_{i=1}^{d} [a_i, b_i]$, with $a_i < b_i$, for any $a_i, b_i \in \mathbb{R}$ can be mapped to $[0,1]^d$ by rescaling. Similarly, logically (patch or block) cartesian domains can be transformed to $(0,1)^d$ by combinations of coordinate transforms. We also rescale time and frequency to set $T = 1$ and $\Lambda = [0,1]$. Finally, the angular domains can be mapped onto to $[0,1]^{d-1}$ by rescaling the underlying polar coordinates. Hence, the underlying domain is $\mathbb{D} = D_T \times S \times \Lambda = [0,1]^{2d+1}$. Thus, we can choose our interior training points $\mathcal{S}_{int}$, temporal boundary training points $\mathcal{S}_{tb}$ and spatial boundary training points $\mathcal{S}_{sb}$ as low-discrepancy Sobol points [2].

Our aim in this section is to derive a rigorous estimate on the so-called *generalization error* (or approximation error) for the trained neural network $u^* = u_{\theta^*}$, which is the output of the PINNs algorithm 2.1. This error is of the form,

$$\mathcal{E}_G = \mathcal{E}_G(\theta^*) := \left(\int\limits_{\mathbb{D}} |u(t,x,\omega,\nu) - u^*(t,x,\omega,\nu)|^2 dz \right)^{\frac{1}{2}}, \tag{2.16}$$

with $dz = dxdtd\omega d\nu$ denoting the volume measure on $\mathbb{D}$.

We follow the recent paper [27] and estimate the generalization error (2.16), in terms of *training errors*,

$$\mathcal{E}_T^{sb} := \left(\sum_{j=1}^{N_{sb}} w_j^{sb} |\mathcal{R}_{sb,\theta^*}(z_j^{sb})|^2 \right)^{\frac{1}{2}}, \quad \mathcal{E}_T^{tb} := \left(\sum_{j=1}^{N_{tb}} w_j^{tb} |\mathcal{R}_{tb,\theta^*}(z_j^{tb})|^2 \right)^{\frac{1}{2}}, \quad \mathcal{E}_T^{int} := \left(\sum_{j=1}^{N_{int}} w_j^{int} |\mathcal{R}_{int,\theta^*}(z_j^{int})|^2 \right)^{\frac{1}{2}} \tag{2.17}$$

Note that the training errors, defined above, correspond to a local minimizer θ^* of (2.15) and are readily computable from the loss function (2.15), during and at the end of the training process.

The detailed estimate on the generalization error in Lemma A.1, together with the assumptions on the underlying coefficients, functions and neural network, is presented and proved in Appendix A. We direct the interested reader to the appendix and focus on the following form of the error estimate (A.3),

$$\begin{aligned}(\mathcal{E}_G)^2 \leqslant\, & C_1 \left((\mathcal{E}_T^{tb})^2 + c(\mathcal{E}_T^{sb})^2 + c(\mathcal{E}_T^{int})^2 \right) \\ & + C_2 \left(\frac{(\log(N_{tb}))^{2d}}{N_{tb}} + c\frac{(\log(N_{sb}))^{2d}}{N_{sb}} + c\frac{(\log(N_{int}))^{2d+1}}{N_{int}} + cN_S^{-2s} \right),\end{aligned} \tag{2.18}$$

with finite constants $C_1 = C$, $C_2 = CC^*$ defined in (A.4). The following remarks about the bound (2.18) are in order,

Remark 2.2. The estimate (2.18) bounds the generalization error in terms of the training errors defined in (2.17) and the number of training points $N_{int,sb,tb}$ as well as quadrature points N_S for approximating the scattering integral in (1.1). Although we have no apriori estimate on the training errors, as argued in [27], these errors can readily calculated after the training process has completed. Thus, the estimate (2.18) tells us that under the assumptions that the constants appearing in (2.18) are finite, *as long as the PINN is trained well, it generalizes well.* This is exactly in the spirit of generalization results in theoretical machine learning [32]. ■

Remark 2.3. We see from the right hand side of the bound (2.18) that the dimensional dependence of the upper bound is only a logarithmic factor. This is not a severe restriction in this case, as the spatial dimension d is atmost 3. It is well known [2] that the logarithmic factor in the rhs of (2.18) starts affecting the rate of decay only when $N_{int} < 2^{2d+1}$. Thus as long as $N_{int} > 128$ and $N_{tb}, N_{tb} > 64$, we should see a linear decay in the error contributions of the Sobol points in (2.18). Hence, we claim that as long as the training errors do not depend on the underlying dimension, the estimate (2.18) suggests that the PINNs algorithm 2.1 will *not suffer from a curse of dimensionality.* ■

Remark 2.4. The estimate (2.18) brings out the role of the speed of light c very clearly. As long as c is finite, we can rescale time to set $c = 1$. Nevertheless, the constant C in (2.18) grows exponentially with the rescaled time, deteriorating the control on the error provided by the bound (2.18). Thus, this bound is not suitable for steady-state problems (formally) obtained by letting $c \to \infty$. Nevertheless, a modified error estimate can be derived for the steady state case and we present it in the appendix B. ■

3 Numerical Experiments

3.1 Implementation

The PINNs algorithm 2.1 has been implemented within the PyTorch framework [34] and the code can be downloaded from `https://github.com/mroberto166/RadiativeTransportPinns`. As is well documented [36, 37, 27], the coding and implementation of PINNs is extremely simple, particularly when compared to standard methods such as finite elements. Only a few lines of Python code suffice for this purpose. All the numerical experiments were performed on a single GeForce GTX1080 GPU.

The PINNs algorithm has the following hyperparameters, the number of hidden layers $K-1$, the width of each hidden layer $d_k \equiv \tilde{d}$ in (2.7), the specific activation function A, the parameter λ in the loss function (2.14), the regularization parameter λ_{reg} in the cumulative loss function (2.15) and the specific gradient descent algorithm for approximating the optimization problem (2.15). We use the hyperbolic tangent tanh activation function, thus ensuring that all the smoothness hypothesis on the resulting neural networks, as required in lemmas A.1 and B.1 are satisfied. Moreover, we use the second-order LBFGS method [8] as the optimizer. We follow the ensemble training procedure of [25] in order to choose the remaining hyperparameters. To this end, we consider a range of values, shown in Table 1, for the number of hidden layers, the depth of each hidden layer, the parameter λ and the regularization parameter λ_{reg}. For each configuration in the ensemble, the resulting model is retrained (in parallel) n_θ times with different random starting values of the trainable weights in the optimization algorithm and the one yielding the smallest value of the training loss is selected.

	$K-1$	$\tilde{d}$	λ	λ_{reg}	n_θ
Example 3.2, 3.3	4, 8	16, 20, 24	0.1, 1, 10	0	5
Example 3.4	4, 8	16, 20	0.1, 1	0, 10^{-6}, 10^{-5}	10
Example 3.5	4, 8, 12, 16, 20	16, 20, 24, 28, 32, 36, 40	0.1, 1	0	20
Example 4.2	4, 8	16, 20, 24	1, 10	0	5

Table 1: Hyperparameter configurations and number of retrainings employed in the ensemble training of PINNs for the radiative transfer equation (1.1)

3.2 Monochromatic stationary radiative transfer in one space dimension

We begin with the much simpler case of steady state radiative transfer in the one space dimension, also referred to as slab geometry [9]. In this case, the radiative transfer equations (1.1) simplify to,

$$\mu\frac{\partial}{\partial x}u(x,\mu) + \Big(\sigma(x)+k(x)\Big)u(x,\mu) = \frac{\sigma(x)}{2}\int_{-1}^{1}\Phi(\mu,\mu')u(z,\mu)d\mu', \quad \mu=\cos(\theta), \quad (x,\mu)\in[0,1]\times[-1,1]. \tag{3.1}$$

We follow the setup of [35] where the authors benchmarked least squares finite element methods for one-dimensional radiative transfer on this problem. As in [35], the following *inflow* boundary conditions are imposed:

$$\begin{aligned} u(0,\mu) &= 1, \quad \mu\in(0,1], \\ u(1,\mu) &= 0, \quad \mu\in[-1,0). \end{aligned} \tag{3.2}$$

Note that the boundary conditions allow for possible discontinuities at $\mu = 0$. The coefficients and scattering kernel are,

$$\sigma(x) = x, \quad k(x)=0, \quad \Phi(\mu',\mu) = \sum_{\ell=0}^{L} d_\ell P_\ell(\mu)P_\ell(\mu'), \quad d_0 = 1, \tag{3.3}$$

with $P_\ell(\mu)$ denoting the Legendre polynomial of order ℓ. We employ the sequence of coefficients $d_\ell = \{1.0, 1.98398, 1.50823, 0.70075, 0.23489, 0.05133, 0.00760, 0.00048\}$, proposed in [35]. Although only in 2

N_{int}	N_{sb}	$K-1$	$\tilde{d}$	λ	$\mathcal{E}_T$	$\lvert\lvert u_- - u_-^* \rvert\rvert_{L^2}$	$\lvert\lvert u_+ - u_+^* \rvert\rvert_{L^2}$	Training Time
8192	2048	8	24	0.1	0.00016	0.07 %	0.29 %	57 min

Table 2: Results for monochromatic stationary radiative transfer in one space dimension.

dimensions, this problem is nevertheless considered rather challenging on account of the possible presence of discontinuities.

We use the PINNs algorithm 2.1 to approximate (3.1), with Sobol points for the interior training set $\mathcal{S}_{int}$ and spatial boundary training set $\mathcal{S}_{sb}$. Similarly, a Gauss-Legendre quadrature rule of order 20 is used for approximating the integral with the scattering kernel. We set $N_{int} = 8192$, $N_{sb} = 2048$ and $N_S = 10$, for this experiment. The hyperparameters that resulted from the ensemble training are presented in Table 2. As seen from the table, a very low training error is obtained in this case. A contour plot of the resulting radiative intensity in (x, μ)-plane is presented in figure 2. The results are very similar to those obtained with a least squares finite element method in [35] (compare figure 2 with figure 3 of [35]). It is interesting to note that this very good qualitative match with the least-squares finite element method is obtained with a training time of slightly less than 1 hour.

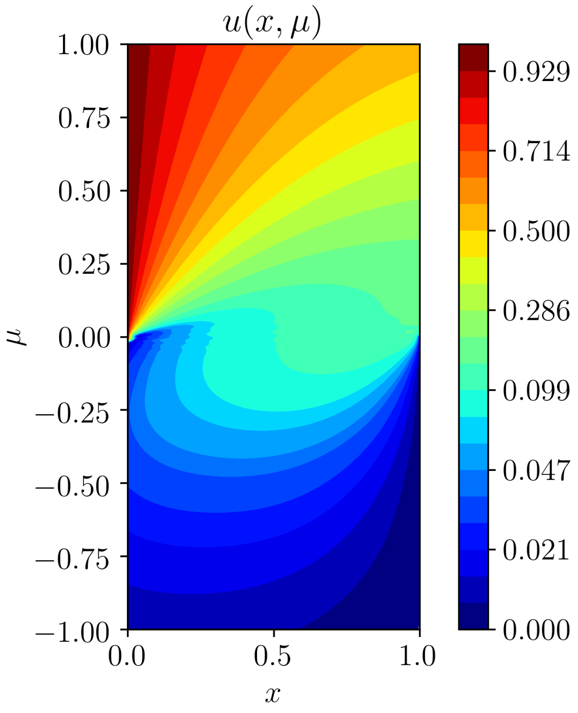


Figure 2: Contour plot of the radiative intensity $u(x, \mu)$ for the 1D monochromatic experiment

Another attractive feature of this simplified problem lies in the fact that the authors in [4] obtained an exact analytical solution for it. Although it is very complicated to evaluate this solution for the whole (x, μ)-plane, its values on the boundaries can be readily evaluated i.e. we can readily compute $u_-(\mu) = u(0, \mu)$ and $u_+(\mu) = u(1, \mu)$. We do so and compare the exact solution with the trained PINN, denoted by $u_\pm^*$. These results are plotted in figure 3. We see from this figure that the PINN is able to very accurately approximate the discontinuous exact solution at the boundary. A quantitative comparison in performed by computed the errors $u_\pm - u_\pm^*$ in L^2-norm. These errors, presented in table 2, are very small for both boundaries and further demonstrate that the PINN is able to approximate the underlying discontinuous solution to high-accuracy, at very low computational cost.

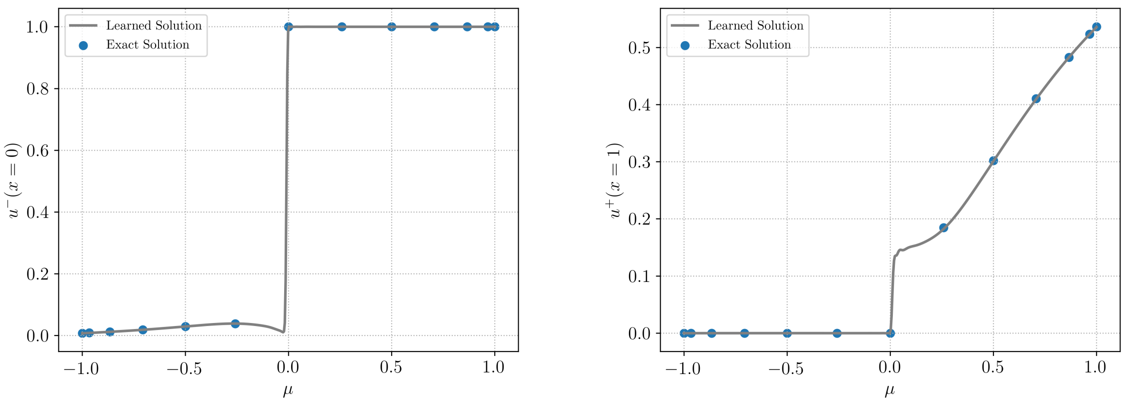


Figure 3: Comparison of the analytical and PINNN radiative intensity at the physical domain boundaries for the stationary monochromatic radiative transfer in one-space dimension.

3.3 Monochromatic stationary radiative transfer in three space dimensions

Next, we consider a monochromatic and stationary version of the general radiative transfer equations (1.1), but in three space dimensions. Already, this problem is in 5 dimensions and is challenging on account of possibly high computational cost. We use the same setup as in [12] (section 8.2, experiment 3) and consider the problem in the unit cube $D = [0, 1]^3$ where a source, located at the center $c = (0.5, 0.5, 0.5)$, radiates into the surrounding medium. We consider no further radiation entering the domain (zero Dirichlet boundary conditions). The source term f is given by

$$f(x) = k(x)I_b(x), \quad I_b(x) = \begin{cases} 0.5 - r, & r \leqslant 0.5 \\ 0, & \text{otherwise} \end{cases} \tag{3.4}$$

with $r = |x - c|$. The absorption coefficient is $k(x) = I_b(x)$ and isotropic scattering $\Phi = 1$, with unit scattering coefficient $\sigma(x) = 1$ is considered.

As before, we use Sobol points for the interior training set $\mathcal{S}_{int}$ and boundary training set $\mathcal{S}_{sb}$. Quadrature points, corresponding to a Gauss quadrature rule of order 20 are also used. We set $N_{int} = 16384$, $N_{sb} = 12288$ and $N_S = 100$. The hyperparameters, corresponding to the best performing networks, that result from ensemble training are presented in Table 3. We see from this table that this hyperparameter configuration resulted in a very low (total) training error of 4.4×10^{-4}, which is comparable to those obtained in the one-space dimension case (see table 2).

N_{int}	N_{sb}	$K-1$	$\tilde{d}$	λ	$\mathcal{E}_T$	Training Time
16384	12288	8	24	0.1	0.00044	1 hr 9 min

Table 3: Results of the ensemble training for the stationary monochromatic radiative transfer in three space dimensions.

As there is no analytical solution available for the radiative intensity in this case, we cannot compute generalization errors. However, based on the theory (see estimate (A.3)) and on the comparison with the one-dimensional case, we expect very low generalization errors when the training errors are this low. Moreover, we can perform qualitative comparisons with the results obtained in [12] with an efficient discrete ordinate method. To this end, we plot three-dimensional volume plot for the *incident radiation* $G(x)$ (see the first equation in (2.4) for definition) in figure 4. We see from this figure that the results with PINN are very similar to the results with the discrete ordinate method, shown in [12] (figure 8.12, page 126). Thus, we are able to approximate the incident radiation to the same accuracy as a discrete ordinate method. The main differences lies in the simplicity of implementation and very low computational cost. We observe from table 3 that the PINN was trained in approximately 70 minutes on a single GPU. This should be contrasted with the very intricate parallel algorithm of [12], which required considerably more computational time as the method resulted in very number of degrees of freedom ranging from $200000 - 600000$.

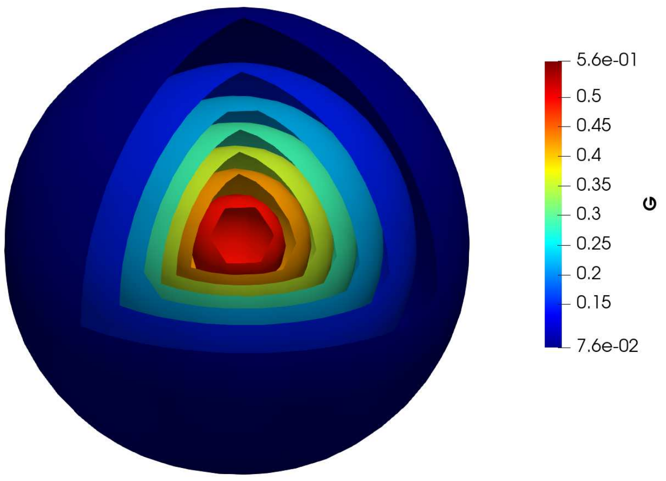


Figure 4: Contour plot of the incident radiation $G(x)$ for the 3D monochromatic experiment

3.4 Polychromatic stationary radiative transfer in three space dimensions

Next, we consider the most general case of the steady state radiative transfer equation (B.1) by following the setup of [39] and references therein, where (B.1) is considered in the unit cube $D = [0,1]^3$ and in the frequency domain $\Lambda = [-6,6]$, with normalization of energy groups. Furthermore, we consider a simple case of zero absorption, isotropic kernel, zero Dirichlet boundary conditions and spherical symmetry. Under the assumptions, by integrating equation (B.1) over the unit sphere S, we arrive at the following ordinary differential equation for the radial flux i.e the incident heat flux (2.4) along the radius,

$$\nabla \cdot F_r = \frac{1}{r^2}\frac{d}{dr}r^2 F_r = 4\pi f(r,\nu) \tag{3.5}$$

with $r = |x - (0.5, 0.5, 0.5)|$ (see also [39] and references therein).

An exact solution for the above ODE can be easily obtained. In particular, with the source term:

$$f(x,\nu) = \begin{cases} \sqrt{\pi}\varphi(\nu)\Big(1-2r\Big) & \text{if } r \leqslant 0.5, \\ 0 & \text{otherwise,} \end{cases} \qquad \varphi(\nu) = \frac{1}{\sqrt{\pi}}\exp\big(-\nu^2\big), \tag{3.6}$$

the radial flux F_r results in

$$F_r = \begin{cases} 4\sqrt{\pi^3}\varphi(\nu)\Big(\frac{r}{3} - \frac{r^2}{2}\Big) & \text{if } r \leqslant 0.5, \\ 4\sqrt{\pi^3}\varphi(\nu)\frac{1}{96r^2} & \text{otherwise.} \end{cases} \tag{3.7}$$

As in the previous numerical experiment, we use Sobol points for the interior and boundary training sets and Gauss quadrature points for integrating the scattering kernel, with $N_{int} = 16384$, $N_{sb} = 12288$ and $N_S = 100$. The hyperparameters used in the ensemble trainig are reported in table 1 and the resulting best performing configuration is shown in table 4. We observe from this table that the resulting training error is 1.6×10^{-3}, which is about three times higher than the training error with the monochromatic experiment (see table 3). This is not surprising as the underlying problem is more complicated on account of introducing frequency as an additional variable and resulting in a 6-dimensional problem.

As no analytical solution is available for the radiative intensity, we cannot compute the generalization error (B.4). However, we can compute the error between the analytical radial flux (3.7) and the PINN approximation (computed from the intensity with a Gauss-Legendre quadrature rule). We show the resulting L^2-norm of the error in table 4. We see from this table that the error for the flux is quite low at approximately 2% relative error, even for this rather complicated underlying problem. Moreover, the training time is only one hour. Thus, PINNs are able to approximate the underlying solution to high accuracy at low computational cost for this 6-dimensional problem.

N_{int}	N_{sb}	$K-1$	$\tilde{d}$	λ	$\mathcal{E}_T$	$\|\|F_r - F_r^*\|\|_{L^2}$	Training Time
16384	12288	8	20	0.1	0.0016	2.1 %	1 hr 6 min

Table 4: Results for steady polychromatic radiative transfer in three space dimensions
.

3.5 Polychromatic time-dependent Radiative transfer in three space dimensions

For the final numerical experiment, we consider the configuration proposed in [11], which is widely used in benchmarking the radiative transport modules in production codes for radiation-(magneto)hydrodynamics, in the context of Astrophysics [44]. The setup is as follows; a sphere with radius R_i and fixed temperature T_S is surrounded by a cold static medium at temperature $T_m < T_S$. The experiment might represent, for instance, the model of a star radiating in the surrounding atmosphere. It is assumed that the sphere, as well as the surrounding medium, are emitting with a Planckian distribution

$$B(T,\nu) = \frac{2h\nu^3}{c^2}\frac{1}{e^{\frac{h\nu}{k_bT}}-1} \tag{3.8}$$

with h and k_b being the Planck and Boltzmann constant, and c the speed of light.

To make the problem tractable, the authors of [11] neglect scattering entirely by setting $\sigma \equiv 0$. Moreover, the absorption coefficient is modeled by $k(x,\nu) = k_\nu$, with ν being the frequency. The emission term is modeled by $f(x,\nu) = k_\nu B(T_m,\nu)$, resulting in the following form of the radiative transfer equation (1.1),

$$\frac{1}{c}\frac{\partial u}{\partial t} + \omega\cdot\nabla_x u = k_\nu(B(T_m,\nu) - u), \quad (t,x,n,\nu) \in D_T \times S \times \Lambda. \tag{3.9}$$

In the context of radiation-(magneto)hydrodynamics, one is mostly interested in the angular moments of the radiative intensity that naturally arise in calculating the contridbution of radiation to the total energy of the fluid (plasma). Hence, it is customary to integrate (3.9) over the sphere S to derive the following PDE for incident radiation (2.4):

$$\frac{1}{c}\frac{\partial}{\partial t}G + \nabla_x\cdot F = k_\nu\Big(b(T_m,\nu) - G\Big), \quad (t,x,\nu) \in D_T \times \Lambda. \tag{3.10}$$

with $b(T,\nu) = 4\pi B(T,\nu)$.

However, the PDE (3.10) is not closed and one needs a closure for the flux F in terms of the incident radiation G. It is common practice in astrophysics to use the so-called *diffusion approximation* of the flux [3]:

$$F(t,x,\nu) = -\frac{1}{3k_\nu}\nabla G(t,x,\nu), \tag{3.11}$$

resulting in the following PDE,

$$\frac{1}{c}\frac{\partial}{\partial t}G - \frac{1}{3k_\nu}\Delta G = k_\nu\Big(b(T_m,\nu) - G\Big), \quad (t,x,\nu) \in D_T \times \Lambda. \tag{3.12}$$

Defining the *Knudsen number* $K = Lk_\nu$ (with L being a characteristic length scale), it is well known that the diffusion approximation is justified in the limit of $K \to \infty$.

k_ν	N_{int}	N_{sb}	N_{tb}	$K-1$	$\tilde{d}$	λ	$\mathcal{E}_T$	Training Time
1	16384	12288	12288	4	40	0.1	0.0028	3 hr 25 min
10	16384	12288	12288	4	40	0.1	0.012	2 hr 15 min

Table 5: Results for polychromatic time-dependent radiative transfer in three space dimensions.

Although the PDE (3.12) is simpler than the full radiative transfer equation (1.1), efficient numerical approximation of (3.12) is still quite challenging as the incident radiation is a function of 5 variables. As it happens, PINNs provide an efficient method for approximating high-dimensional parabolic equations such as (3.12), see [27] section 3 for details.

However, by assuming radial symmetry and with the flux approximation given in (3.11), the differential equation (3.12) admits an analytical solution satisfying the initial and boundary conditions

$$\begin{aligned} G(0,r,\nu) &= b(T_m,\nu), \\ G(t,r\to\infty,\nu) &= b(T_m,\nu), \\ G(0,R_i,\nu) &= b(T_s,\nu). \end{aligned} \tag{3.13}$$

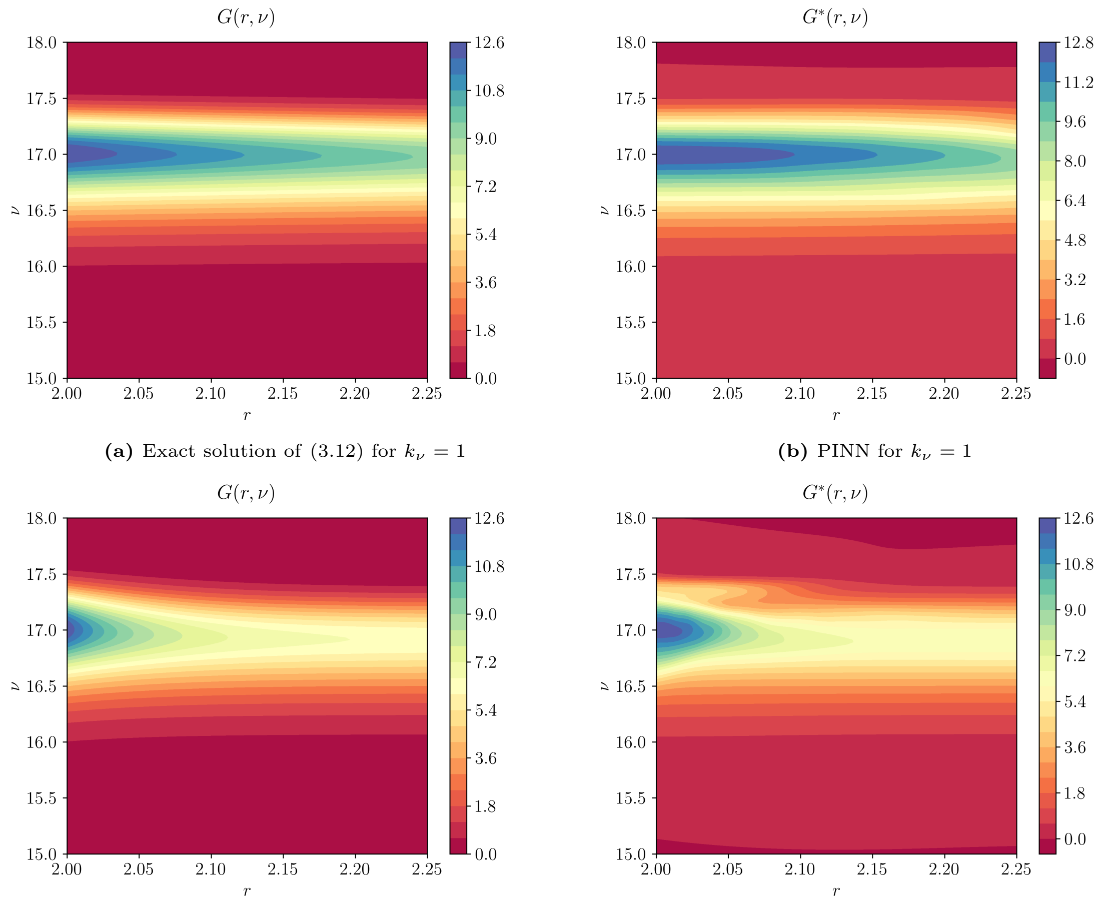
The exact solution for (3.12) then reads [11],

$$\begin{aligned} G(t,r,\nu) =& b(T_m,\nu) + \frac{R_i}{r}\Big(b(T_s,\nu) - b(T_m,\nu)\Big)F(t,r,\nu), \\ F(t,r,\nu) =& \frac{1}{2}\exp\left(-3k_\nu(r-R)\right)\left\{Erfc\left(\sqrt{\frac{3k_\nu}{4ct}}(r-R) - \sqrt{k_\nu ct}\right) + Erfc\left(\sqrt{\frac{3k_\nu}{4ct}}(r-R) + \sqrt{k_\nu ct}\right)\right\}. \end{aligned} \tag{3.14}$$

For this numerical experiment, we will approximate the full time-dependent radiative transfer equations (3.9) with the PINNs algorithm 2.1. To this end, we consider (3.9) in the spatial domain D enclosed between two spheres with radii $R_i = 2$ and $R_e = 4$. Moreover, we introduce an auxiliary temporal variable $\tau = ct$ to rescale time to $[0,1]$, whereas the energy group ranges between 10^{15} and 10^{18}. We set $T_s = 150eV$ and $T_m = 120eV$.

The PINNs algorithm 2.1 employs Sobol points in the interior, spatial and temporal boundary training sets and we set $N_{int} = 16384$, $N_{sb} = N_{tb} = 12228$. Moreover, we solve this problem for two different values of the (constant in frequency) absorption coefficient i.e $k_\nu = 1$ and $k_\nu = 10$, resulting in two different Knudsen numbers of $K = 2$ and $K = 20$, respectively. Given the challenging nature of this problem, we choose slightly different ranges of the hyperparameters, presented in tabel 1 for ensemble training and also use 20 retrainings, corresponding to different random starting values for the weights and biases in the training procedure. The resulting best performing configurations are reported in table 5. We observe from this table that PINNs provide a very low training error of 2.8×10^{-3}, for the $K = 2$ case. This training error is comparable to the training errors for the previous two examples. The training error increases by a factor of 4 for the $K = 20$ case, but still remains relatively low.

As we do not have exact analytical formulas for the full radiative intensity, it is not possible to compute generalization errors. However to ascertain the quality of the solution, we compare with the exact solution (3.14) of the diffusion equation (3.12) for the incident radiation. This comparison is shown as contour plots for the incident radiation in the (r,ν)-plane (with r denoting the radial direction) in figure 5 as well as one-dimensional cross-sections for different values of the radius r in figure 6. As seen from both these figures, there is good agreement between the incident radiation, computed by a Gauss quadrature of the PINN approximation to the radiative intensity in (3.9), and the analytical solution of the diffusion approximation (3.12) for the $K = 20$ case. This is not unexpected as the diffusion approximation is accurate for large Knudsen numbers. On the other hand, there is a significant difference between the the incident radiation, computed by a Gauss quadrature of the PINN approximation to the radiative intensity in (3.9), and the analytical solution of the diffusion approximation (3.12) for the $K = 2$ case. This follows from the fact that the diffusion approximation will provide a poor approximation of (3.9) for low Knudsen numbers. On the other hand, given the relatively low training error as well as the error estimate (A.3), coupled with the results of the previous numerical experiments, we argue that the PINN



(a) Exact solution of (3.12) for $k_\nu = 1$

(b) PINN for $k_\nu = 1$

(c) Exact solution of (3.12) for $k_\nu = 10$

(d) PINN at $\tau = 1$ for $k_\nu = 10$

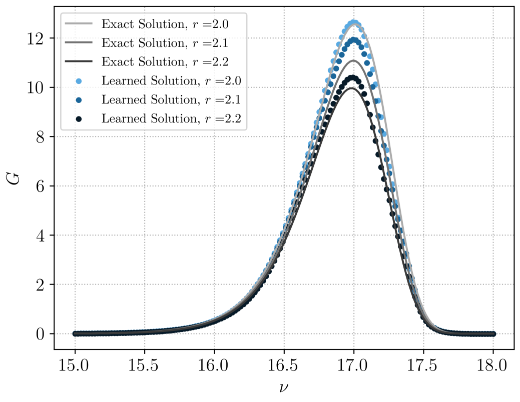
Figure 5: Comparison of incident radiationn with respect to the exact solution of the diffusion approximation (3.12) and PINN approximation of the full radiative transfer equation (3.9) at rescaled time $\tau = 1$ for two different values of the absorption coefficient $k_\nu = 1, 10$

provides a much more accurate approximation to the underlying radiative intensity (and its moments) than the diffusion approximation will do, atleast for low to moderate Knudsen numbers. Hence, PINNs provide a viable and accurate method for competing radiative transfer in media with different optical properties. Moreover, the runtime for even this very complicated problem was reasonably small, ranging from two to three and half hours on a single GPU.

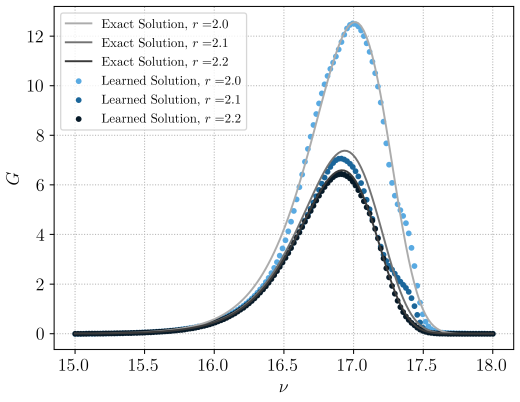
4 PINNs for the Inverse problem for radiative transfer

One of the most notable features of PINNs is their ability to approximate solutions of *inverse problems*, with the same accuracy and computational cost as that of forward problems for PDEs. Moreover, the code for inverse problems ends up being a very minor modification to the code for forward problems, which makes PINNs extremely attractive for various applications [37, 38] and references therein.

Here, we focus on the following inverse problem for radiative transfer. We consider the full time-dependent version on the radiative transfer equation (1.1), with initial and boundary conditions (2.3). The inverse problem is to compute unknown absorption coefficients k, scattering coefficients σ, scattering kernel Φ or emission term f, given measurements of either the full radiative intensity u or its angular moments, such as the incident radiation G or heat flux F (2.4). For simplicity of exposition, we choose the following concrete inverse problem;
Given measurements of the incident radiation $G(t, x, \nu)$, find the unknown absorption coefficient k =



(a) Solution of (3.9) and (3.10) for $k_\nu = 1$



(b) Solution of (3.9) and (3.10) for $k_\nu = 10$

Figure 6: Comparison of exact solutions of the diffusion approximation (3.12) with the PINN approximation of the full radiative transfer equation (3.9) for two different values of the absorption coefficient $k_\nu = 1, 10$ at different radial locations and at rescaled time $\tau = 1$

$k(x, \nu)$ *and the resulting radiative intensity* $u(t, x, \omega, \nu)$ *which solves the radiative transfer equation* (1.1) *with initial and boundary conditions* (2.3).

Other combinations of the measured and unknown quantities can be similarly considered. Clearly this inverse problem is ill-posed as multiple absorption coefficients might lead to the same incident radiation. However, we will aim to obtain one of the possible absorption coefficients, consistent with the measured incident radiation.

To this end, we slightly modify the PINNs algorithm as described below,

4.1 PINNs for the inverse problem

Following [37, 28], we seek to find the deep neural networks $k_{\theta_k} : D \times \Lambda \mapsto \mathbb{R}_+$ and $u_{\theta_u} : D_T \times S \times \Lambda \mapsto \mathbb{R}$, with the concatenated parameter vector $\theta = \{\theta_k, \theta_u\} \in \Theta$, approximating the absorption coefficient and radiative intensity, respectively.

In addition to the interior training set $\mathcal{S}_{int}$, spatial boundary training set $\mathcal{S}_{sb}$ and temporal boundary training set $\mathcal{S}_{tb}$, defined in section 2, we also required the so-called *data training set* $\mathcal{S}_d = \{y_j^d\}$, for $1 \leqslant j \leqslant N_d$, and $y_j^d \in D_T \times \Lambda$.

The residuals for initial and boundary conditions are given by $\mathcal{R}_{tb}, \mathcal{R}_{sb}$ (2.12). We slightly modify the PDE residual (2.11) to,

$$\overline{\mathcal{R}}_{int,\theta} := \frac{1}{c}\partial_t u_{\theta_u} + \omega \cdot \nabla_x u_{\theta_u} + k_{\theta_k} u_{\theta_u} + \sigma \left(u_{\theta_u} - \frac{1}{s_d} \sum_{i=1}^{N_S} w_i^S \Phi(\omega, \omega_i^S, \nu, \nu_i^S) u_{\theta_u}(t, x, \omega_i^S, \nu_i^S) \right) - f. \quad (4.1)$$

We also need the *data residual*,

$$\mathcal{R}_{d,\theta} := G\left(u_{\theta_u}\right) - \bar{G}(t, x, \nu), \quad \forall (t, x, \nu) \in D_T \times \Lambda, \quad (4.2)$$

with G being the incident radiation calculated from (2.4) with a Gauss quadrature approximation of the angular integral and $\bar{G}$ being the measured incident radiation.

The resulting loss function is,

$$J(\theta) := \sum_{j=1}^{N_d} w_j^d |\mathcal{R}_{d,\theta}(y_j^d)|^2 + \sum_{j=1}^{N_{sb}} w_j^{sb} |\mathcal{R}_{sb,\theta}(z_j^{sb})|^2 + \sum_{j=1}^{N_{tb}} w_j^{tb} |\mathcal{R}_{tb,\theta}(z_j^{tb})|^2 + \lambda \sum_{j=1}^{N_{int}} w_j^{int} |\overline{\mathcal{R}}_{int,\theta}(z_j^{int})|^2 \quad (4.3)$$

with the residuals $\mathcal{R}_d, \mathcal{R}_{sb}, \mathcal{R}_{tb}$ and $\overline{\mathcal{R}}_{int}$ defined in (4.2), (2.12) ,(2.11), and w^d, y^d, w^{sb}, z^{sb}, w^{sb}, z^{sb}, w^{int}, z^{int} being the quadrature weights and training points, corresponding to the data, boundary and interior training sets.

The PINNs algorithm for the inverse problem is summarized as,

Algorithm 4.1. *Finding physics informed neural network (PINNs) to approximate the absorption coefficient k and radiative intensity u solving the radiative transfer equation (1.1), and consistent with measured data $\bar{G}$ for the incident radiation*

Inputs: *Underlying domain $D_T \times S \times \Lambda$, coefficients and data for the radiative transfer equation (1.1), measured incident radiation G, quadrature points and weights for underlying quadrature rules, non-convex gradient based optimization algorithms.*

Goal: *Find PINN $(k^*, u^*) = \left(k_{\theta_k^*}, u_{\theta_u^*}\right)$ for approximating the inverse problem for radiative transfer*

Step 1: *Choose the training sets as described in section 2.2 and in this subsection.*

Step 2: *For an initial value of the weight vector $\overline{\theta} \in \Theta$, evaluate the neural networks $u_{\overline{\theta}_u}$, $k_{\overline{\theta}_k}$ (2.7), the PDE residual (4.1), the data residual (4.2), the boundary residuals (2.12), the loss function (4.3) and its gradients to initialize the underlying optimization algorithm.*

Step 3: *Run the optimization algorithm till an approximate local minimum θ^* of (4.3) is reached. The map $u^* = u_{\theta_u^*}$ is the desired PINN for approximating the solution u of the radiative transfer equation and the map $k^* = k_{\theta_k^*}$ is the corresponding absorption coefficient.*

N_{int}	N_d	$K-1$	$\tilde{d}$	λ	$\mathcal{E}_T$	$\lvert\lvert u-u^*\rvert\rvert_{L^2}$	$\lvert\lvert k-k^*\rvert\rvert_{L^2}$	$\lvert\lvert G-G^*\rvert\rvert_{L^2}$	Training Time
16384	4096	8	20	1.0	0.00094	0.65 %	2.8 %	0.073%	1 hr 44 min

Table 6: Results for the inverse problem for radiative transfer.

4.2 A Numerical Experiment.

The monochromatic stationary version of the radiative transfer equation (1.1) in three space dimensions, is used in this numerical experiment. The spatial domain is the unit cube $D = [0,1]^3$, with scattering coefficient $\sigma = 0.5$, scattering kernel $\Phi \equiv 1$. The source term f and boundary term u_b are generated using the following synthetic absorption coefficient and exact solution,

$$k(x) = \prod_{i=1}^{3} x_i^2, \quad u(x,\omega) = \frac{3}{16\pi}(1 + (\omega \cdot \omega')^2) \prod_{i=1}^{3} x_i(x_i - 1), \quad n' = \Big(\frac{1}{\sqrt{3}}, \frac{1}{\sqrt{3}}, \frac{1}{\sqrt{3}}\Big)^T \tag{4.4}$$

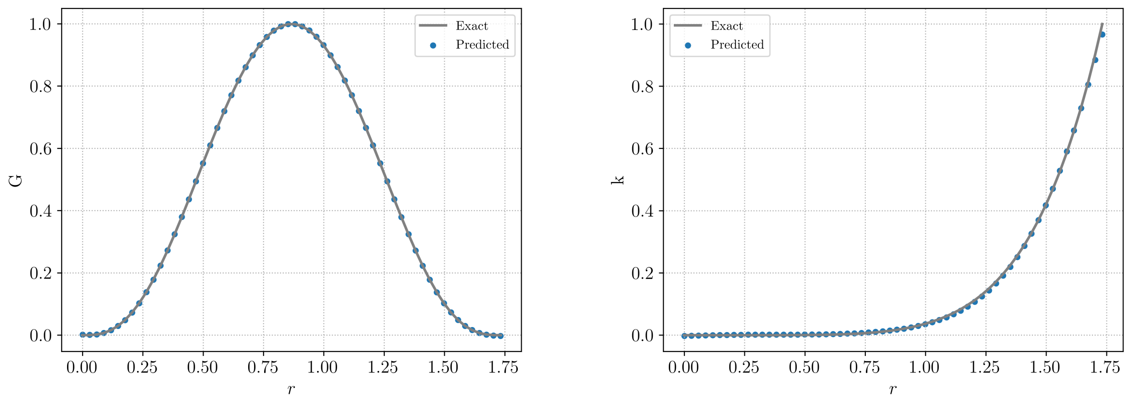
The measured incident radiation $\bar{G}$ in (4.2) is calculated from the radiative intensity u above by using the formula (2.4).

For this numerical experiment, we also impose boundary conditions on the neural network approximating the absorption coefficient k_{θ_k} to approximately match the values of k, defined in (4.4) on the boundary of D, leading to an additional term in (4.3). Finally, in order to ensure uniqueness of the absorption coefficient, we include in the loss function, the so-called Tikhonov regularization:

$$J_T(\theta) = \lambda_k ||\nabla k_\theta||_2^2, \quad \lambda_k = 0.001. \tag{4.5}$$

We use Sobol points for the interior training points and uniformly distributed random points are used as data training points, with $N_{int} = 16384$, $N_d = 4096$. The resulting best performing hyperparameter configuration after ensemble training is presented in Table 6.

In figure 7 we plot the incident radiation G and the absorption coefficient k, along the diagonal of the unit cube, computed with the PINNs algorithm 4.1. As observed from this figure, the incident radiation is almost identical to the measured data $\bar{G}$. This is further verified from table 6, from which we observe a very low L^2-error for the incident radiation. On the other hand, the absorption coefficient agrees reasonably well with the ground truth in (4.4), with an error of less than 3%. Also the radiative intensity is approximated to very high accuracy, with a generalization error below 1%. This is even more impressive if one consider that the problem is solved with a computational time of approximately 100 minutes.



(a) Incident radiation G **(b)** Absorption Coefficient k

Figure 7: Results for the PINNs algorithm 4.1 for the inverse problem for radiative transfer. The PINNs approximation to the incident radiation and absorption coefficient are plotted along the diagonal of the unit cube and compared with the measured data $\bar{G}$ and ground truth absorption coefficient k given by (4.4).

5 Discussion

Accurate numerical approximation of the radiative transfer equations (1.1) is considered very challenging as the underlying problem is high-dimensional, with 7-dimensions in the most general case. Moreover, the presence of different physical effects such as emission, absorption and scattering as well as varying optical parameters in the surrounding medium further complicates design of efficient numerical algorithms. As discussed in the introduction, existing numerical methods can suffer from the so-called *curse of dimensionality* and require very large amount of computational resources to achieve desired accuracy.

Hence, there is a need for designing algorithms for simulating radiative transfer, that are easy to implement and fast (in terms of computational run time) while still being accurate. We proposed such an algorithm in this article. Our algorithm 2.1 is based on *physics informed neural networks* (PINNs) i.e. deep neural networks for approximating the radiative intensity in (1.1). The deep neural network is trained by using a gradient descent method to minimize a loss function (2.15), that consists of the *PDE residual*, resulting from the neural network being plugged into the radiative transfer equation (1.1). Mismatches with respect to the initial and boundary conditions also contribute to the loss function. The residuals are collocated at *training points*, which correspond to quadrature points with respect to an underlying quadrature rule. We chose Sobol low-discrepancy sequences as training points in order to alleviate the curse of dimensionality.

The resulting algorithm is extremely simple to code within standard machine learning frameworks such as TensorFlow and Pytorch. We presented a suite of numerical experiments ranging from the simplest monchromatic stationary radiative transfer in one space dimension (3.1) to the most general time-dependent polychromatic radiative transfer in three space dimensions. PINNs performed very well on all the numerical experiments, leading to low errors with small run (training) times. In particular, the results were qualitatively and quantitatively comparable to published results, but possibly at a fraction of the cost. The experimental results were supplemented with rigorous error estimates that bounded the generalization error (2.16) in terms of computable training errors (2.17) and number of quadrature points, independent of the underlying dimension, see bounds (A.3) for details. The predictions of the error estimates were validated by the experiments.

Hence, we claim that the PINNs algorithm 2.1 is a general purpose, simple to implement, fast and accurate simulator for radiative transfer. Moreover, we also presented a (very) slighly modified version of the PINN algorithm to efficiently simulate a class of *inverse problems* for radiative transfer. In this problem, the objective is to compute the unknown absorption coefficient from measurements of the incident radiation. To this end, we proposed the deep neural network based algorithm 4.1, which added a data fidelity term (4.2) to the underlying loss function (4.3). This algorithm was also found to be both fast and accurate in numerical experiments. Thus, we provide *novel machine learning algorithms* which are fast, easy to implement and accurate for efficiently simulating different aspects of radiative transfer.

This article should be considered as a first step in adapting machine learning algorithms for simulating radiative transfer. The proposed algorithm lends itself readily to be extended in the following directions,

- The quantities of interest in many applications of radiative transfer, particularly in astrophysics, are angular moments such as the incident radiation G and heat flux F, defined in (2.4) as these quantities define the contribution of radiation to the total energy. Thus, in radiation hydrodynamics, one often resorts to using moment models. However, these moment models require closure relations, which might either be expensive to compute and/or inaccurate (see section 3.5 for a simple diffusion based closure). Given the computational speed of PINNs, one can readily employ algorithm 2.1 as the radiation module in a hydrodynamics code, with angular moments and the resulting energy being computed from the neural network approximation of the radiative intensity. Another option would be to leverage the ability of PINNs to directly perform hydrodynamic simulations (see [38, 27]) and have a full PINN simulation of radiation hydrodynamics. Both approaches should be developed and tested.
- We covered the inverse problem for radiative transfer briefly in section 4. Algorithm 4.1 can be readily extended to many other inverse problems involving radiative transfer, for instance finding emission and scattering coefficients from measurements of incident radiation and heat fluxes at a few points. A careful exposition and analysis of the resulting algorithms and their application in practical contexts will be the basis of future papers.

Acknowledgements

The research of SM and RM was partially supported by the European Research Council (ERC) under ERCCoG 770880 COMANFLO. SM thanks Dr. Roger Käppeli of SAM, ETH Zürich for his inputs on radiation and astrophysics.

A Estimates on the generalization error for the radiative transfer equation (1.1)

In order to derive an error estimate for the PINNs algorithm, we need to make some assumptions on the scattering kernel Φ in (1.1). We follow standard practice and assume that it is symmetric $\Phi(\omega, \omega', \nu, \nu') = \Phi(\omega', \omega, \nu', \nu)$. Moreover, the following function,

$$\Psi(\omega, \nu) = \int\limits_{S \times \Lambda} \Phi(\omega, \omega', \nu, \nu') d\omega' d\nu', \tag{A.1}$$

is essentially bounded i.e. $\Psi \in L^\infty(S \times \Lambda)$. We have the following estimate on the generalization error (2.16),

Lemma A.1. *Let $u \in L^2(\mathbb{D})$ be the unique weak solution of the radiative transfer equation* (1.1)*, with absorption coefficient $0 \leqslant k \in L^\infty(D \times \Lambda)$, scattering coefficient $0 \leqslant \sigma \in L^\infty(D \times \Lambda)$ and a symmetric scattering kernel $\Phi \in C^\ell(S \times \Lambda \times S \times \Lambda)$, for some $\ell \geqslant 1$, such that the function Ψ (defined in* (A.1)*) is in $L^\infty(S \times \Lambda)$. Let $u^* = u_{\theta^*} \in C^\ell(\mathbb{D})$ be the output of the PINNs algorithm 2.1 for approximating the radiative transfer equation* (1.1)*, such that*

$$\max\{V_{HK}(u^*), V_{HK}\left(\mathcal{R}_{int,\theta^*}\right)\} < +\infty, \tag{A.2}$$

with V_{HK} being the so-called Hardy-Krause variation (see [2, 29] for the precise definition). We also assume that the initial data u_0 and boundary data u_b are of bounded Hardy-Krause variation. Then, under the assumption that Sobol points are used as the training points $\mathcal{S}_{int}, \mathcal{S}_{sb}, \mathcal{S}_{tb}$ in algorithm 2.1 and

Guass-quadrature rule of order $s = s(\ell)$ is used in approximating the scattering kernel in the residual (2.11)*, we have the following estimate on the generalization error,*

$$
\begin{aligned}
(\mathcal{E}_G)^2 \leqslant\, & C\left((\mathcal{E}_T^{tb})^2 + c(\mathcal{E}_T^{sb})^2 + c(\mathcal{E}_T^{int})^2\right) \\
& + CC^*\left(\frac{(\log(N_{tb}))^{2d}}{N_{tb}} + c\frac{(\log(N_{sb}))^{2d}}{N_{sb}} + c\frac{(\log(N_{int}))^{2d+1}}{N_{int}} + cN_S^{-2s}\right)
\end{aligned}
\tag{A.3}
$$

with constants defined as,

$$
\begin{aligned}
C &= T + c\hat{C}T^2 e^{c\hat{C}T}, \quad \hat{C} = 2 + \frac{2(\|\sigma\|_{L^\infty} + \|\Psi\|_{L^\infty})}{s_d} \\
C^* &= \max\left\{V_{HK}\left((\mathcal{R}_{tb}^*)^2\right), V_{HK}\left((\mathcal{R}_{sb}^*)^2\right), V_{HK}\left((\mathcal{R}_{int}^*)^2\right), \overline{C}\right\} \\
\overline{C} &= \overline{C}\left(|\mathbb{D}|, \|\Phi\|_{C^\ell}, \|u^*\|_{C^\ell}\right)
\end{aligned}
\tag{A.4}
$$

Proof. We drop the θ^* dependence in the residuals (2.11), (2.12), for notational convenience and denote the residuals as $\mathcal{R}_{int}^*, \mathcal{R}_{sb}^*, \mathcal{R}_{tb}^*$. Define,

$$
E(u^*, \Phi) := \sum_{i=1}^{N_S} w_i^S \Phi(\omega, \omega_i^S, \nu, \nu_i^S) u^*(t, x, \omega_i^S, \nu_i^S) - \int\limits_\Lambda \int\limits_S \Phi(\omega, \omega', \nu, \nu') u^*(t, x, \omega', \nu') d\omega' d\nu'. \tag{A.5}
$$

It is straightforward to derive from the radiative transfer equation (1.1) and the definition of residuals (2.11), (2.12), that the error $\hat{u} = u^* - u$, satisfies the following integro-differential equation,

$$
\begin{aligned}
\frac{1}{c}\hat{u}_t + \omega \cdot \nabla_x \hat{u} &= -(k+\sigma)\hat{u} + \frac{\sigma}{s_d}\int\limits_\Lambda \int\limits_S \Phi(\omega, \omega', \nu, \nu')\hat{u}(t, x, \omega', \nu') d\omega' d\nu' \\
&\quad + \mathcal{R}_{int}^* + E(u^*, \Phi). \\
\hat{u}(0, x, \omega, \nu) &= \mathcal{R}_{tb}^*, \quad (x, \omega, \nu) \in D \times S \times \Lambda, \\
\hat{u}(t, x, \omega, \nu) &= \mathcal{R}_{sb}^*, \quad (t, x, \omega, \nu) \in \Gamma_- \times \Lambda.
\end{aligned}
\tag{A.6}
$$

Multiplying $\hat{u}$ on both sides of the first equation in (A.6), we obtain,

$$
\begin{aligned}
\frac{1}{2c}\frac{d(\hat{u}^2)}{dt} + \omega \cdot \nabla_x\left(\frac{\hat{u}^2}{2}\right) &= -(k+\sigma)\hat{u}^2 + \frac{\sigma}{s_d}\int\limits_\Lambda \int\limits_S \Phi(\omega, \omega', \nu, \nu')\hat{u}(t, x, \omega', \nu')\hat{u}(t, x, \omega, \nu) d\omega' d\nu' \\
&\quad + \mathcal{R}_{int}^*\hat{u} + E(u^*, \Phi)\hat{u}
\end{aligned}
\tag{A.7}
$$

Integrating the above over $D \times S \times \nu$, integrating by parts and using the Cauchy's inequality and the fact that $k, \sigma \geqslant 0$, we obtain for any $t \in (0, T]$,

$$
\begin{aligned}
\frac{1}{2c}\frac{d}{dt}\int_{D\times S\times\Lambda} \hat{u}^2(t, x, \omega, \nu) dx d\omega d\nu \leqslant & \int_{D\times S\times\Lambda} \hat{u}^2(t, x, \omega, \nu) dx d\omega d\nu - \int\limits_{(\partial D\times S\times\Lambda)_-} (\omega \cdot n(x))\frac{\hat{u}^2(t, x, \omega, \nu)}{2} ds(x) d\omega d\nu \\
& + \int\limits_{D\times S\times\Lambda} \frac{\sigma}{s_d}\int\limits_\Lambda \int\limits_S \Phi(\omega, \omega', \nu, \nu')\hat{u}(t, x, \omega', \nu')\hat{u}(t, x, \omega, \nu) d\omega' d\nu' d\nu d\omega dx, \\
& + \int\limits_{D\times S\times\Lambda} \frac{(\mathcal{R}_{int}^*(t, x, \omega, \nu))^2}{2} d\nu d\omega dx + \int\limits_{D\times S\times\Lambda} \frac{(E(u^*, \Phi)(t, x, \omega, \nu))^2}{2} d\nu d\omega dx
\end{aligned}
\tag{A.8}
$$

Here $ds(x)$ denotes the surface measure on ∂D and we define

$$
(\partial D \times S \times \Lambda)_- := \{(x, \omega, \nu) \in \partial D \times S \times \Lambda : \omega \cdot n(x) \leqslant 0\},
$$

with $n(x)$ being the unit outward normal at $x \in \partial D$.

We fix any $\bar{T} \in (0,T]$ and integrate (A.8) over $(0,\bar{T})$ and estimate the result to obtain,

$$
\begin{aligned}
\int\limits_{D\times S\times\Lambda} \hat{u}^2(\bar{T},x,\omega,\nu)dxd\omega d\nu &\leqslant \int\limits_{D\times S\times\Lambda} \hat{u}^2(0,x,\omega,\nu)dxd\omega d\nu + 2c\int\limits_0^{\bar{T}}\int_{D\times S\times\Lambda} \hat{u}^2(t,x,\omega,\nu)dtdxd\omega d\nu \\
&+ c\int\limits_{\Gamma_-} |\omega\cdot n|\hat{u}^2(t,x,\omega,\nu)dtds(x)d\omega d\nu + I + c\int\limits_{\mathbb{D}}(\mathcal{R}^*_{int})^2dz + c\int\limits_{\mathbb{D}}(E(u^*,\Phi))^2dz.
\end{aligned}
\tag{A.9}
$$

Here, the term I in (A.9), is defined and estimated by successive applications of Cauchy-Schwatrz inequality as,

$$
\begin{aligned}
I &= 2c\int\limits_0^{\bar{T}}\int\limits_{D\times S\times\Lambda}\frac{\sigma}{s_d}\int\limits_{\Lambda}\int\limits_{S}\Phi(\omega,\omega',\nu,\nu')\hat{u}(t,x,\omega',\nu')\hat{u}(t,x,\omega,\nu)d\omega'd\nu'd\nu d\omega dxdt, \\
&\leqslant \frac{2c(\|\sigma\|_{L^\infty}+\|\Psi\|_{L^\infty})}{s_d}\int\limits_0^{\bar{T}}\int_{D\times S\times\Lambda}\hat{u}^2(t,x,\omega,\nu)dtdxd\omega d\nu.
\end{aligned}
$$

By identifying constant $\hat{C}$ from (A.4), we obtain from (A.9) and (A.6) that,

$$
\begin{aligned}
\int\limits_{D\times S\times\Lambda}\hat{u}^2(\bar{T},x,\omega,\nu)dxd\omega d\nu &\leqslant \int\limits_{D\times S\times\Lambda}(\mathcal{R}^*_{tb})^2dxd\omega d\nu + c\int\limits_{\Gamma_-}(\mathcal{R}^*_{sb})^2dtds(x)d\omega d\nu \\
&+ c\int\limits_{\mathbb{D}}(\mathcal{R}^*_{int})^2dz + c\int\limits_{\mathbb{D}}(E(u^*,\Phi))^2dz \\
&+ c\hat{C}\int\limits_0^{\bar{T}}\int\limits_{D\times S\times\Lambda}\hat{u}^2(t,x,\omega,\nu)dtdxd\omega d\nu.
\end{aligned}
\tag{A.10}
$$

Applying the integral form of Grönwall's inequality to (A.10), we obtain for any $0<\bar{T}\leqslant T$,

$$
\begin{aligned}
\int\limits_{D\times S\times\Lambda}\hat{u}^2(\bar{T},x,\omega,\nu)dxd\omega d\nu &\leqslant \left(1+c\hat{C}\bar{T}e^{c\hat{C}\bar{T}}\right)\left(\int\limits_{D\times S\times\Lambda}(\mathcal{R}^*_{tb})^2dxd\omega d\nu + c\int\limits_{\Gamma_-}(\mathcal{R}^*_{sb})^2dtds(x)d\omega d\nu\right) \\
&+ \left(1+c\hat{C}\bar{T}e^{c\hat{C}\bar{T}}\right)\left(c\int\limits_{\mathbb{D}}(\mathcal{R}^*_{int})^2dz + c\int\limits_{\mathbb{D}}(E(u^*,\Phi))^2dz\right)
\end{aligned}
\tag{A.11}
$$

Integrating (A.11) over $(0,T)$ yields,

$$
\begin{aligned}
(\mathcal{E}_G)^2 := \int\limits_{\mathbb{D}}\hat{u}^2(t,x,\omega,\nu)dz &\leqslant \left(T+c\hat{C}T^2e^{c\hat{C}T}\right)\left(\int\limits_{D\times S\times\Lambda}(\mathcal{R}^*_{tb})^2dxd\omega d\nu + c\int\limits_{\Gamma_-}(\mathcal{R}^*_{sb})^2dtds(x)d\omega d\nu\right) \\
&+ \left(T+c\hat{C}T^2e^{c\hat{C}T}\right)\left(c\int\limits_{\mathbb{D}}(\mathcal{R}^*_{int})^2dz + c\int\limits_{\mathbb{D}}(E(u^*,\Phi))^2dz\right)
\end{aligned}
\tag{A.12}
$$

As the training points in $\mathcal{S}_{tb}$ are the Sobol quadrature points, we realize that the training error $(\mathcal{E}^{tb}_T)^2$ (2.17) is the quasi-Monte Carlo quadrature for the first integral in (A.12). Hence by the well-known Koksma-Hlawka inequality [2], we obtain the following estimate,

$$
\int\limits_{D\times S\times\Lambda}(\mathcal{R}^*_{tb})^2dxd\omega d\nu \leqslant (\mathcal{E}^{tb}_T)^2 + V_{HK}\left((\mathcal{R}^*_{tb})^2\right)\frac{(\log(N_{tb}))^{2d}}{N_{tb}}. \tag{A.13}
$$

By a similar argument, we can estimate,

$$\int\limits_{\Gamma_-} (\mathcal{R}^*_{sb})^2 dt ds(x) d\omega d\nu \leqslant (\mathcal{E}_T^{sb})^2 + V_{HK}\left((\mathcal{R}^*_{sb})^2\right) \frac{(\log(N_{sb}))^{2d}}{N_{sb}},$$
$$\int\limits_{\mathbb{D}} (\mathcal{R}^*_{int})^2 dz \leqslant (\mathcal{E}_T^{int})^2 + V_{HK}\left((\mathcal{R}^*_{int})^2\right) \frac{(\log(N_{int}))^{2d+1}}{N_{int}}, \tag{A.14}$$

As ω_i^S, ν_i^S, for $1 \leqslant i \leqslant N_S$ are Gauss-quadrature points, we follow [42] and readily estimate E defined in (A.5) by the error for an s-th order accurate Gauss quadrature rule with $s = s(\ell)$ as,

$$\int\limits_{\mathbb{D}} (E(u^*, \Phi))^2 dz \leqslant \overline{C} N_S^{-2s}, \tag{A.15}$$

with constant $\overline{C}$ defined in (A.4) By plugging in the estimates (A.13), (A.14), (A.15) in (A.12) and identifying constants, we derive the desired estimate (A.3) on the generalization error (2.16). □

B Estimates on the generalization error in the steady case

The steady-state (time-independent) version of the radiative transfer equation (1.1) is obtained by letting the speed of light $c \to \infty$ and resulting in,

$$(k+\sigma)u = -\omega \cdot \nabla_x u + \frac{\sigma}{s_d} \int\limits_{\Lambda} \int\limits_{S} \Phi(\omega, \omega', \nu, \nu') u(x, \omega', \nu') d\omega' d\nu' + f, \tag{B.1}$$

with all the coefficients and sources as defined before. We also impose the *inflow* boundary condition,

$$u(x, \omega, \nu) = u_b(x, \omega, \nu), \quad (t, x, \omega, \nu) \in \Gamma^s_, \tag{B.2}$$

with inflow boundary defined by,

$$\Gamma^s_- = \{(x, \omega, \nu) \in \partial D \times S \times \Lambda : \omega \cdot n(x) < 0\} \tag{B.3}$$

with $n(x)$ denoting the unit outward normal at any point $x \in \partial D$.

The PINNs algorithm 2.1 can be readily adpated to this case by simply (formally) neglecting the temporal dependence in the residuals (2.11), (2.12) and loss functions and the underlying definitions of neural networks. We omit detailing this procedure here. Our objective is to bound the resulting generalization error,

$$\mathcal{E}_G^s = \mathcal{E}_G^s(\theta^*) := \left(\int\limits_{D \times S \times \Lambda} |u(x, \omega, \nu) - u^*(x, \omega, \nu)|^2 dz \right)^{\frac{1}{2}}, \tag{B.4}$$

with $dz = dx d\omega d\nu$ denoting the underlying volume measure. As in lemma A.1, we will bound the generalization error in terms of the training errors,

$$\mathcal{E}_T^{sb} := \left(\sum_{j=1}^{N_{sb}} w_j^{sb} |\mathcal{R}_{sb,\theta^*}(z_j^{sb})|^2 \right)^{\frac{1}{2}}, \quad \mathcal{E}_T^{int} := \left(\sum_{j=1}^{N_{int}} w_j^{int} |\mathcal{R}_{int,\theta^*}(z_j^{int})|^2 \right)^{\frac{1}{2}} \tag{B.5}$$

Here, z_j^{int} and z_j^{sb} are the interior and spatial boundary training points.

We have the following estimate on the generalization error,

Lemma B.1. *Let $u \in L^2(D \times S \times \Lambda)$ be the unique weak solution of the radiative transfer equation* (B.1)*, with absorption coefficient $0 < k_{min} \leqslant k(x,\nu) \leqslant k_{max} < \infty$, scattering coefficient $0 < \sigma_{min} \leqslant \sigma(x,\nu) \leqslant \sigma_{max} < \infty$, for almost every $x \in D, \nu \in \Lambda$ and a symmetric scattering kernel $\Phi \in C^\ell(S \times \Lambda \times S \times \Lambda)$, for some $\ell \geqslant 1$, such that the function Ψ (defined in* (A.1)*) is in $L^\infty(S \times \Lambda)$. We further assume that the absorption and scattering coefficients are related in the following manner, there exists a $\kappa > 0$, such that*

$$k_{min} + \sigma_{min} - \frac{\sigma_{max} + \|\Psi\|_{L^\infty}}{s_d} \geqslant \kappa \tag{B.6}$$

Let $u^* = u_{\theta^*} \in C^\ell(D \times S \times \Lambda)$ be the output of the PINNs algorithm 2.1 for approximating the stationary radiative transfer equation (B.1)*, such that*

$$\max\{V_{HK}(u^*), V_{HK}\left(\mathcal{R}_{int,\theta^*}\right)\} < +\infty, \tag{B.7}$$

with V_{HK} being the Hardy-Krause variation. We also assume that the boundary data u_b is of bounded Hardy-Krause variation. Then, under the assumption that Sobol points are used as the training points $\mathcal{S}_{int}, \mathcal{S}_{sb}$ in algorithm 2.1 and Guass-quadrature rule of order $s = s(\ell)$ is used in approximating the scattering kernel in the residual (2.11)*, we have the following estimate on the generalization error,*

$$(\mathcal{E}_G^s)^2 \leqslant C\left((\mathcal{E}_T^{sb})^2 + (\mathcal{E}_T^{int})^2 + \frac{(\log(N_{sb}))^{2d-1}}{N_{sb}} + \frac{(\log(N_{int}))^{2d}}{N_{int}} + N_S^{-2s}\right) \tag{B.8}$$

with constants defined as,

$$C = \max\left\{\frac{2}{\kappa}, \frac{2}{\kappa}V_{HK}\left((\mathcal{R}_{sb}^*)^2\right), \frac{2C_\varepsilon}{\kappa}\left((\mathcal{R}_{int}^*)^2\right), \frac{2C_\varepsilon}{\kappa}\overline{C}N_S^{-2s}\right\}, \tag{B.9}$$

where $\overline{C}$ is defined in (A.4)*. Here, C_ε is a constant that depends on κ and is defined in* (B.14)*.*

Proof. We drop the θ^* dependence in the residuals (2.11), (2.12), for notational convenience and denote the residuals as $\mathcal{R}_{int}^*, \mathcal{R}_{sb}^*$. Define,

$$E_s(u^*, \Phi) := \sum_{i=1}^{N_S} w_i^S \Phi(\omega, \omega_i^S, \nu, \nu_i^S) u^*(x, \omega_i^S, \nu_i^S) - \int_\Lambda \int_S \Phi(\omega, \omega', \nu, \nu') u^*(x, \omega', \nu') d\omega' d\nu'. \tag{B.10}$$

It is straightforward to derive from the radiative transfer equation (B.1) and the definition of residuals (2.11), (2.12), that the error $\hat{u} = u^* - u$, satisfies the following integro-differential equation,

$$\begin{aligned}(k+\sigma)\hat{u} &= -\omega \cdot \nabla_x \hat{u} + \frac{\sigma}{s_d}\int_\Lambda \int_S \Phi(\omega, \omega', \nu, \nu')\hat{u}(t, x, \omega', \nu') d\omega' d\nu' + \mathcal{R}_{int}^* + E_s(u^*, \Phi), \\ \hat{u}(t, x, \omega, \nu) &= \mathcal{R}_{sb}^*, \quad (x, \omega, \nu) \in \Gamma_-^s\end{aligned} \tag{B.11}$$

Multiplying $\hat{u}$ on both sides of the first equation in (B.11), we obtain,

$$\begin{aligned}(k+\sigma)\hat{u}^2 &= -\omega \cdot \nabla_x(\frac{\hat{u}^2}{2}) + \frac{\sigma}{s_d}\int_\Lambda \int_S \Phi(\omega, \omega', \nu, \nu')\hat{u}(t, x, \omega', \nu')\hat{u}(t, x, \omega, \nu) d\omega' d\nu' \\ &+ \mathcal{R}_{int}^*\hat{u} + E_s(u^*, \Phi)\hat{u}\end{aligned} \tag{B.12}$$

Integrating the above over $D \times S \times \nu$, integrating by parts, using the assumed lower and upper bounds on k, σ, we obtain,

$$(k_{min} + \sigma_{min}) \int_{D \times S \times \nu} \hat{u}^2 dz \leqslant \int_{\Gamma_-^s} (\mathcal{R}_{sb}^*)^2 ds(x) d\omega d\nu + I + \int_{D \times S \times \nu} (\mathcal{R}_{int}^*\hat{u} + E_s(u^*, \Phi)\hat{u}) dz, \tag{B.13}$$

with term I defined and estimated by,

$$\begin{aligned} I &= \int\limits_{D\times S\times\Lambda} \frac{\sigma}{s_d} \int\limits_{\Lambda}\int\limits_{S} \Phi(\omega,\omega',\nu,\nu')\hat{u}(x,\omega',\nu')\hat{u}(x,\omega,\nu)d\omega' d\nu' d\nu d\omega dx, \\ &\leqslant \frac{\sigma_{max}+\|\Psi\|_{L^\infty}}{s_d} \int_{D\times S\times\Lambda} \hat{u}^2(x,\omega,\nu)dz. \end{aligned}$$

From the assumption (B.6), there exists an $\varepsilon > 0$ such that $k_{min}+\sigma_{min}-\frac{\sigma_{max}+\|\Psi\|_{L^\infty}}{s_d}-2\varepsilon > \frac{\kappa}{2}$, we use the ε-version of Cauchy's inequality,

$$ab \leqslant \varepsilon a^2 + C_\varepsilon b^2, \tag{B.14}$$

to further estimate (B.13) as,

$$\int\limits_{D\times S\times\Lambda} \hat{u}^2 dz \leqslant \frac{2}{\kappa}\int\limits_{\Gamma^s_-} (\mathcal{R}^*_{sb})^2 ds(x)d\omega d\nu + \frac{2C_\varepsilon}{\kappa}\left(\int\limits_{D\times S\times\nu} (\mathcal{R}^*_{int})^2 + (E_s(u^*,\Phi))^2 dz\right) \tag{B.15}$$

By using the estimates (A.14) and (A.15) and identifying constants, we obtain the desired bound (B.8) on the generalization error (B.4).

□

As for the time-dependent case, the bound (B.8) should be considered in the sense of *if the PINN is trained well, it generalizes well.* Moreover, the bound, and consequently, the PINN does not suffer from a curse of dimensionality by the same argument as in the time-dependent case. Infact, the logarithmic corrections to the linear decay of the rhs in (B.8) can be ignored at an even smaller number of training points.

The assumption (B.6) plays a key role in the derivation of the bound (B.8). A careful inspection of this assumption reveals that the scattering coefficient is not allowed to vary over a large range, unless there is enough absorption in the medium. However, there is no restriction on the range of scales over which the absorption coefficient can vary.

References

[1] A. R. Barron. Universal approximation bounds for superpositions of a sigmoidal function. *IEEE Trans. Inform. Theory.*, 39(3):930–945, 1993.

[2] R. E. Caflisch. Monte carlo and quasi-monte carlo methods. *Acta Numerica*, 7:1–49, 1998.

[3] J. I. Castor. *Radiation hydrodynamics.* Cambridge University Press, 2004.

[4] Y. Cengel, M. Özi, et al. Radiation transfer in an anisotropically scattering plane-parallel medium with space-dependent albedo ω(x). *Journal of Quantitative Spectroscopy and Radiative Transfer*, 34(3):263–270, 1985.

[5] Y. Chen, L. Lu, G. E. Karniadakis, and L. D. Negro. Physics-informed neural networks for inverse problems in nano-optics and metamaterials. Preprint, available from arXiv:1912.01085, 2019.

[6] W. E, J. Han, and A. Jentzen. Deep learning-based numerical methods for high-dimensional parabolic partial differential equations and backward stochastic differential equations. *Communications in Mathematics and Statistics*, 5(4):349–380, 2017.

[7] R. Evans, J. Jumper, J. Kirkpatrick, L. Sifre, T. Green, C. Qin, A. Zidek, A. Nelson, A. Bridgland, H. Penedones, et al. De novo structure prediction with deep-learning based scoring. *Annual Review of Biochemistry*, 77(363-382):6, 2018.

[8] R. Fletcher. *Practical methods of optimization.* John Wiley and sons, 1987.

[9] M. Frank. Approximate models for radiative transfer. *Bull. Inst. Math. Acad. Sinica (New Series)*, 2:409–432, 2007.

[10] I. Goodfellow, Y. Bengio, and A. Courville. *Deep learning.* MIT press, 2016.

[11] F. Graziani. The prompt spectrum of a radiating sphere: Benchmark solutions for diffusion and transport. In *Computational methods in transport: verification and validation, LNCSE-62*, pages 151–167. Springer, 2008.

[12] K. Grella. *Sparse tensor approximation for radiative transport.* PhD thesis, ETH Zurich, 2013.

[13] K. Grella and C. Schwab. Sparse tensor spherical harmonics approximation in radiative transfer. *J. Comput. Phys.*, 230:8452–8473, 2011.

[14] J. Han, A. Jentzen, and W. E. Solving high-dimensional partial differential equations using deep learning. *Proceedings of the National Academy of Sciences*, 115(34):8505–8510, 2018.

[15] G. Kanschat and et. al. *Numerical methods in multi-dimensional radiative transfer.* Springer, 2008.

[16] D. Kaushik, M. Smith, A. Wollaber, B. Smith, A. Siegel, and W. S. Yang. Enabling high fidelity neutron transport simulations on petascle architectures. In *Proceedings of the Conference on High Performance Computing Networking, Storage, and Analysis, volume 67, Portland, Oregon*, 2009.

[17] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015*, 2015.

[18] I. E. Lagaris, A. Likas, and P. G. D. Neural-network methods for bound- ary value problems with irregular boundaries. *IEEE Transactions on Neural Networks*, 11:1041–1049, 2000.

[19] I. E. Lagaris, A. Likas, and D. I. Fotiadis. Artificial neural networks for solving ordinary and partial differential equations. *IEEE Transactions on Neural Networks*, 9(5):987–1000, 1998.

[20] K. D. Lathrop. Ray effects in discrete ordinates equations. *Nucl. Sci. Eng*, 32:357, 1968.

[21] Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.

[22] Y. Liu, X. Meng, and G. E. Karniadakis. B-pinns: Bayesian physics-informed neural networks for forward and inverse pde problems with noisy data. Preprint, available from arXiv:2003.06097, 2020.

[23] L. Lu, X. Meng, Z. Mao, and G. E. Karniadakis. Deepxde: A deep learning library for solving differential equations. Preprint, available from arXiv:1907.04502, 2019.

[24] K. . O. Lye, S. Mishra, P. Chandrasekhar, and D. Ray. Iterative surrogate model optimization (ismo): An active learning algorithm for pde constrained optimization with deep neural networks. Preprint, available as arXiv:2008.05730, 2020.

[25] K. O. Lye, S. Mishra, and D. Ray. Deep learning observables in computational fluid dynamics. *Journal of Computational Physics*, page 109339, 2020.

[26] Z. Mao, A. D. Jagtap, and G. E. Karniadakis. Physics-informed neural networks for high-speed flows. *Computer Methods in Applied Mechanics and Engineering*, 360:112789, 2020.

[27] S. Mishra and R. Molinaro. Estimates on the generalization error of physics informed neural networks (pinns) for approximating pdes. Preprint, available from arXiv:2006:16144v1, 2020.

[28] S. Mishra and R. Molinaro. Estimates on the generalization error of physics informed neural networks (pinns) for approximating pdes ii: A class of inverse problems. Preprint, available from arXiv:2007:01138v1, 2020.

[29] S. Mishra and T. K. Rusch. Enhancing accuracy of deep learning algorithms by training with low-discrepancy sequences. Preprint, available as arXiv:2005.12564, 2020.

[30] M. F. Modest. *Radiative heat transfer.* Elsevier, 2003.

[31] M. F. Modest and J. Yang. Elliptic pde formulation and boundary conditions of the spherical harmonics method of arbitrary order for general three-dimensional geometries. *Journal of Quantitative Spectroscopy and Radiative Transfer*, 109:1641–1666, 2008.

[32] M. Mohri, A. Rostamizadeh, and A. Talwalkar. *Foundations of machine learning.* MIT press, 2018.

[33] G. Pang, L. Lu, and G. E. Karniadakis. fpinns: Fractional physics-informed neural networks. *SIAM journal of Scientific computing*, 41:A2603–A2626, 2019.

[34] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer. Automatic differentiation in pytorch. In *Workshop Proceedings of Neural Information Processing Systems*, 2017.

[35] J. Pontaza and J. Reddy. Least-squares finite element formulations for one-dimensional radiative transfer. *Journal of Quantitative Spectroscopy and Radiative Transfer*, 95(3):387–406, 2005.

[36] M. Raissi and G. E. Karniadakis. Hidden physics models: Machine learning of nonlinear partial differential equations. *Journal of Computational Physics*, 357:125–141, 2018.

[37] 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. *Journal of Computational Physics*, 378:686–707, 2019.

[38] M. Raissi, A. Yazdani, and G. E. Karniadakis. Hidden fluid mechanics: A navier-stokes informed deep learning framework for assimilating flow visualization data. *arXiv preprint arXiv:1808.04327*, 2018.

[39] S. Richling, E. Meinköhn, N. Kryzhevoi, and G. Kanschat. Radiative transfer with finite elements-i. basic method and tests. *Astronomy & Astrophysics*, 380(2):776–788, 2001.

[40] I. M. Sobol'. On the distribution of points in a cube and the approximate evaluation of integrals. *Zhurnal Vychislitel'noi Matematiki i Matematicheskoi Fiziki*, 7(4):784–802, 1967.

[41] D. Stamatellos and A. P. Whitworth. Probing the initial conditions for star formation with monte carlo radiative transfer simulations. In *Numerical Methods in Multidimensional Radiative Transfer, G. Kanschat, E. Meinköhn, R. Rannacher, and R. Wehrse, eds*, pages 289–298. Springer, 2008.

[42] J. Stoer and R. Bulirsch. *Introduction to numerical analysis.* Springer Verlag, 2002.

[43] G. Widmer. *Sparse Finite Elements for Radiative Transfer.* PhD thesis, ETH Zurich, 2009.

[44] W. Zhang, L. Howell, A. Almgren, A. Burrows, J. . Dolence, and J. Bell. Castro: A new compressible astrophysical solver. iii. multigroup radiation hydrodynamics. *Astrophysical Journal (supplement series)*, 204(7):27 pp, 2013.