Data-Driven Filtering of the Spherical Harmonics Method

Benjamin Plumridge[†], Cory Hauck[†‡], AND Steffen Schotthöfer[‡]

Abstract

We investigate a data-driven approach for tuning the filtered spherical harmonics method (FP_N) when solving the radiation transport equation (RTE). The FP_N method extends the classical spherical harmonics approach (P_N) by introducing regularization through a filter operator, which mitigates spurious oscillations caused by Gibbs' phenomenon. This filter includes a tunable parameter, the filter strength, that controls the degree of smoothing applied to the solution. However, selecting an optimal filter strength is nontrivial, often requiring inaccessible information such as the true or a high-order reference solution. To overcome this limitation, we model the filter strength as a neural network whose inputs include local state variables and material cross-sections. The optimal filter strength is formulated as the solution to a PDE-constrained optimization problem, and the neural network is trained using a discretize-then-optimize formulation in PyTorch. We evaluate the learned filter strength across a suite of test problems and compare the results to those from a simple, but tunable constant filter strength. In all cases, the neural-network-driven filter substantially improves the accuracy of the P_N approximation. For 1-D test cases, the constant filter outperforms the neural-network filter in some cases, but in 2-D problems, the neural-network filter generally performs better.

Funding

This work is sponsored by the National Science Foundation under DMS-1913277 and by the Applied Mathematics Program at the Office of Advanced Scientific Computing Research, U.S. Department of Energy. Work was performed at the Oak Ridge National Laboratory, which is managed by UT-Battelle, LLC under Contract No. DE-AC05-00OR22725 with the U.S. Department of Energy. The United States Government retains and the publisher, by accepting the article for publication, acknowledges that the United States Government retains a non-exclusive, paid-up, irrevocable, world-wide license to publish or reproduce the published form of this manuscript, or allow others to do so, for United States Government purposes. The Department of Energy will provide public access to these results of federally sponsored research in accordance with the DOE Public Access Plan (http://energy.gov/downloads/doe-public-access-plan).

1 Introduction

The radiation transport equation (RTE) is an integro-partial differential equation (PDE) used to model the movement of particles that interact with a background material medium via emission, absorption, and scattering processes. The RTE has applications in high-energy density physics and nuclear engineering, including core-collapse supernovae [31], thermal radiative transfer [21], general radiation hydrodynamics [32, 35], and nuclear reactor theory [4, 29]. One of the challenges in approximating solutions of the RTE is dealing with its high-dimensional phase space, which can include as many as six dimensions: three spatial variables, two angular variables, and one energy variable. If transients are desired, then the RTE will also depend on time. Due to limitations in computational resources, coarse approximations of the RTE are often necessary. In many cases, angular discretization is the primary target for such approximations.

One well known approach for discretizing the RTE in angle is with a truncated spherical harmonics expansion [6, 9], where the expansion coefficients are functions of space and time that satisfy a system of hyperbolic

[†]Department of Mathematics, University of Tennessee, Knoxville
[‡]Computer Science and Mathematics Division, Oak Ridge National Laboratory

balance laws. These equations are referred to as the P_N equations, where the subscript N refers to the polynomial degree at which the expansion is truncated. For smooth solutions of the RTE, the P_N equations converge spectrally to the RTE as $N \to \infty$ [13]; this is one the key advantages of the method. Another advantage is that the operators in the P_N equations are all sparse and therefore cheap to evaluate. In particular, because the spherical harmonics are eigenfunctions of the scattering operator in the RTE, the corresponding operator in the P_N equations appears as a diagonal matrix [29]. Meanwhile, the advection operator of the RTE is converted into a set of banded matrices whose exact form follows from known recursion relations for the spherical harmonics [6].

Despite the efficiency in approximating smooth solutions to the RTE, the P_N equations do have drawbacks. Like all spectral methods [19], they exhibit oscillatory behavior when approximating non-smooth functions, particularly near discontinuities and large gradients [7]. In some cases, these oscillations can lead to solutions with negative particle densities [7] unless some type of numerical limiter is applied [18]. Because scattering induces smooth solutions, these issues are more prevalent when the amount of scattering is relatively small.

The filtered spherical harmonics (FP_N) equations are a modification of the standard P_N equations that induce regularization via an artificial scattering operator. The key advantage of the FP_N approach is that it effectively damps numerical oscillations at a computational cost that remains comparable to that of the original P_N method. Filtering the spherical harmonic expansion was first proposed in [30], where the filter was implemented as an post-processing step after every time step of the P_N system. Later in [36], the FP_N equations were derived as the continuous limit as $\Delta t \to 0$ of the filtering scheme presented in [30]. In [13], the convergence properties of the FP_N equations were established. Roughly speaking, it was shown that the FP_N equations converge to the exact RTE solution at a rate that is equal to the filter order for smooth solutions and (almost) equal to the convergence order of the standard P_N equations for non-smooth solutions. Although filtering alone does not ensure positivity, additional limiting strategies can be employed [26, 27].

For a given filter order, the amount of regularization in the FP_N equations is determined by the filter strength, a single parameter that does not affect the convergence behavior when N is large but can have dramatic effects when N is small [25], particularly in regimes with relatively low scattering. As with Tikhonov regularization when solving noisy inverse problems, tuning the filter strength is a matter of balancing stability with consistency [16]: while increasing the filter strength improves the stability of the method, it also introduces additional consistency error in the angular discretization. It remains an open question how to best tune the filter, but currently the most viable approach seems to be a combination of physics-based scaling arguments and experimental trial and error using relatively cheap, coarse-mesh simulations [25, 30].

The goal of the current work is to design a data-driven model for the filter strength in the FP_N equations. Specifically, we design a parametrized neural network that takes local material properties and the solution state as inputs and outputs a suitable filter strength. As a consequence, the filter strength will vary both spatially and temporally. We train this data-driven filter via PDE-constrained optimization. We use a discretize-then-optimize approach using a second-order, finite-volume scheme for space-time discretization. The training loss is the L^2 discrepancy of the particle concentration for the data-driven FP_N solution and a high order P_N reference simulation. To generate training data, we sample various test-problems with different source terms, initial conditions and material properties.

The learned models parameters are used to predict the optimal filter strength in several test problems and compared to the original P_N solution. We test the neural network filter over a set of problems that challenge the approximation with discontinuities and steep gradients in the material properties and the solution state. Our results show substantial error reductions. Because the raw P_N solutions are often so poor that almost any filter yields improvements, we introduce a stronger baseline: a constant filter strength, realized as a degenerate neural network with only a bias term. In many instances, even this constant filter markedly improves the accuracy and quality of the P_N approximation. In several one-dimensional cases, it outperforms the neural-network filter, especially in regions of material vacuum.

The remainder of the paper is organized as follows. In Section 2, we describe the mono-energetic RTE, as well as the P_N and FP_N equations. In Section 3, we formulate the details of the data driven filter. In Section 4, we show numerical results. Some of the details of the P_N discretization are provided in the Appendix.

2 Background

In this section, we briefly recall the radiation transport equation (RTE) for a collection of mono-energetic particles. We also present the P_N and FP_N equations.

2.1 Radiation transport equation

The radiation transport equation (RTE) governs the evolution of a kinetic distribution ψ of particles moving through a background medium. Let $X \subset \mathbb{R}^3$ have a piecewise smooth boundary $\partial\Omega$ and let $\mathbb{S}^2$ be the unit sphere in $\mathbb{R}^3$. In this paper, ψ is defined such that $\psi(\boldsymbol{x}, \boldsymbol{\Omega}, t)$ gives the density of particles, with respect to the measure $d\boldsymbol{\Omega}dx$, located at position $\boldsymbol{x} = (x, y, z)^\top \in X$, moving in direction $\boldsymbol{\Omega} \in \mathbb{S}^2$ at time $t \geq 0$. For particles with unit speed, the RTE takes the form

$$\partial_t \psi(\boldsymbol{x}, \boldsymbol{\Omega}, t) + \boldsymbol{\Omega} \cdot \nabla_{\boldsymbol{x}} \psi(\boldsymbol{x}, \boldsymbol{\Omega}, t) + \sigma_{\mathrm{a}}(\boldsymbol{x})\psi(\boldsymbol{x}, \boldsymbol{\Omega}, t) + \sigma_{\mathrm{s}}(\boldsymbol{x})\mathcal{Q}\psi(\boldsymbol{x}, \boldsymbol{\Omega}, t) = S(\boldsymbol{x}, \boldsymbol{\Omega}, t), \tag{1a}$$

$$\psi(\boldsymbol{x}, \boldsymbol{\Omega}, t = 0) = \psi_0(\boldsymbol{x}, \boldsymbol{\Omega}), \tag{1b}$$

$$\psi(\boldsymbol{x}, \boldsymbol{\Omega}, t) = \psi_{\mathrm{b}}(\boldsymbol{x}, \boldsymbol{\Omega}), \qquad \forall (\boldsymbol{x}, \boldsymbol{\Omega}) \in \Gamma^-, \tag{1c}$$

where $S = S(\boldsymbol{x}, \boldsymbol{\Omega}, t)$ is a non-negative source and the absorption cross-section σ_{a} and scattering cross-section σ_{s} are non-negative functions of $\boldsymbol{x}$. The scattering operator $\mathcal{Q}$, which describes the change in direction of the particles due to collisions with the background medium, is an integral operator in $\boldsymbol{\Omega}$:

$$\mathcal{Q}\psi(\boldsymbol{x}, \boldsymbol{\Omega}, t) = \psi(\boldsymbol{x}, \boldsymbol{\Omega}, t) - \int_{\mathbb{S}^2} g(\boldsymbol{\Omega} \cdot \boldsymbol{\Omega}')\psi(\boldsymbol{x}, \boldsymbol{\Omega}', t) d\boldsymbol{\Omega}', \tag{2}$$

where the scattering kernel $g : [-1, 1] \to \mathbb{R}$ is normalized so that $\int_{\mathbb{S}^2} g(\boldsymbol{\Omega} \cdot \boldsymbol{\Omega}')d\boldsymbol{\Omega}' = 1$. In the numerical results of this work, we focus on isotropic scattering kernels, in which case $g = \frac{1}{4\pi}$. However, we maintain $\mathcal{Q}$ as a general operator in the derivations and proofs below. Inflow data is provided on the set

$$\Gamma^- = \left\{ (\boldsymbol{x}, \boldsymbol{\Omega}) \in \partial X \times \mathbb{S}^2 : \pm \mathbf{n}(\boldsymbol{x}) \cdot \boldsymbol{\Omega} \geq 0 \right\}, \tag{3}$$

where $\mathbf{n}(\boldsymbol{x})$ is the outward unit normal to X at $\boldsymbol{x} \in \partial X$, is the inflow boundary in phase space.

The existence of a unique semi-group solution for (1) is given in [12, Chapter XXI], under standard assumptions on the data. If all sources, initial conditions, and boundary conditions are non-negative, then the semi-group solution remains non-negative for all time.

2.2 P_N equations

The P_N equations are based on a spectral Galerkin approximation of the RTE in the angular variable in terms of a truncated spherical harmonic expansion. Let $\theta \in [0, \pi]$ and $\varphi \in [0, 2\pi)$ be the polar and azimuthal angles associated to $\boldsymbol{\Omega}$. Then

$$\boldsymbol{\Omega} = \begin{pmatrix} \Omega^{(1)} \\ \Omega^{(2)} \\ \Omega^{(3)} \end{pmatrix} = \begin{pmatrix} \sin\theta\cos\varphi \\ \sin\theta\sin\varphi \\ \cos\theta \end{pmatrix} = \begin{pmatrix} \sqrt{1-\mu^2}\cos\varphi \\ \sqrt{1-\mu^2}\sin\varphi \\ \mu \end{pmatrix} \tag{4}$$

where $\mu = \cos\theta$. Spherical harmonics [3], which form an orthonormal basis for $L^2(\mathbb{S}^2)$, are defined for for each $\ell \in \{0, 1, 2, ...\}$ and $m \in \{-\ell, ..., \ell\}$ by

$$Y_\ell^m(\boldsymbol{\Omega}) = (-1)^m \sqrt{\frac{(2\ell+1)}{4\pi}\frac{(\ell-m)!}{(\ell+m)!}} P_\ell^m(\mu) e^{im\varphi}, \tag{5}$$

where P_ℓ^m represents the Legendre function [1] of degree ℓ and order m, and μ and φ are related to $\boldsymbol{\Omega}$ via (4). The motivation for using this basis is that spherical harmonics form a complete set of eigenfunctions for Q [9]. More specifically,

$$\mathcal{Q}Y_\ell^m = (1 - g_\ell)Y_\ell^m, \tag{6}$$

where

$$g_\ell = 2\pi \int_{-1}^{1} P_\ell(\eta) g(\eta) d\eta \tag{7}$$

and P_ℓ is the degree ℓ Legendre polynomial, normalized such that $\int_{-1}^{1} |P_\ell(\eta)|^2 d\eta = \frac{2}{2\ell+1}$. With this normalization, $P_0(\eta) = 1$ and $|P_\ell(\eta)| < 1$ for $\ell > 1$ and almost every $\eta \in [-1, 1]$ [3]; it follows that $g_0 = 1$ and $g_\ell < 1$ for $\ell > 1$.

For convenience, we use real-valued, orthonormal spherical harmonics

$$r_\ell^m = \begin{cases} \frac{1}{\sqrt{2}} \left(Y_\ell^{-m} + (-1)^m Y_\ell^m \right) & \text{if } m > 0, \\ Y_\ell^0 & \text{if } m = 0, \\ \frac{i}{\sqrt{2}} \left(Y_\ell^{|m|} - (-1)^{|m|} Y_\ell^{-|m|} \right) & \text{if } m < 0. \end{cases} \tag{8}$$

Let $\mathbf{r}_\ell : \mathbf{\Omega} \longrightarrow \mathbb{R}^{2\ell+1}$ be a vector-valued function containing all of the spherical harmonic functions of degree ℓ, i.e.,

$$\mathbf{r}_0 = r_0^0, \quad \mathbf{r}_1 = [r_1^{-1}, r_1^0, r_1^1]^\top, \quad \mathbf{r}_\ell = [r_\ell^{-\ell}, \ldots, r_\ell^0 \ldots r_\ell^\ell]^\top; \tag{9}$$

and let $\mathbf{r} : \mathbf{\Omega} \longrightarrow \mathbb{R}^n$, where $n = (N+1)^2$, be a single vector-valued function containing all the spherical harmonic functions up to degree N; that is, $\mathbf{r} = [\mathbf{r}_0^\top \ldots \mathbf{r}_N^\top]^\top$.

The P_N approximation is given by

$$\psi_{\mathrm{P}_N}(\boldsymbol{x}, \mathbf{\Omega}, t) = \mathbf{u}_{\mathrm{P}_N}^\top(\boldsymbol{x}, t) \mathbf{r}(\mathbf{\Omega}) = \sum_{\ell=0}^{N} \sum_{m=-\ell}^{\ell} [u_{\mathrm{P}_N}]_\ell^m(\boldsymbol{x}, t) r_\ell^m(\mathbf{\Omega}), \tag{10}$$

where [1]

$$\langle \mathbf{r} \partial_t \psi_{\mathrm{P}_N} \rangle + \langle \mathbf{r} \mathbf{\Omega} \cdot \nabla_{\boldsymbol{x}} \psi_{\mathrm{P}_N} \rangle + \sigma_\mathrm{a} \langle \mathbf{r} \psi_{\mathrm{P}_N} \rangle + \sigma_\mathrm{s} \langle \mathbf{r} \mathcal{Q} \psi_{\mathrm{P}_N} \rangle = \langle \mathbf{r} S \rangle, \tag{11a}$$

$$\langle \mathbf{r} \psi_{\mathrm{P}_N}|_{t=0}) \rangle = \langle \mathbf{r} \psi_0 \rangle. \tag{11b}$$

Substituting the expansion in (10) into (11) yields the P_N equations for the coefficient vector $\mathbf{u}_{\mathrm{P}_N} : X \times [0, \infty) \to \mathbb{R}^n$:

$$\partial_t \mathbf{u}_{\mathrm{P}_N}(\boldsymbol{x}, t) + \mathbf{A} \cdot \nabla_{\mathbf{x}} \mathbf{u}_{\mathrm{P}_N}(\mathbf{x}, \mathbf{t}) + \sigma_\mathrm{a}(\mathbf{x}) \mathbf{u}_{\mathrm{P}_N}(\mathbf{x}, \mathbf{t}) + \sigma_\mathrm{s}(\mathbf{x}) \mathbf{G} \mathbf{u}_{\mathrm{P}_N}(\mathbf{x}, \mathbf{t}) = \mathbf{s}(\boldsymbol{x}, t), \tag{12a}$$

$$\mathbf{u}_{\mathrm{P}_N}(\boldsymbol{x}, 0) = \langle \mathbf{r} \psi_0 \rangle. \tag{12b}$$

Here $\mathbf{A}$ is a three-tensor such that

$$\mathbf{A} \cdot \nabla_{\boldsymbol{x}} \mathbf{u}_{\mathrm{P}_N} = \mathbf{A}^{(1)} \partial_x \mathbf{u}_{\mathrm{P}_N} + \mathbf{A}^{(2)} \partial_y \mathbf{u}_{\mathrm{P}_N} + \mathbf{A}^{(3)} \partial_z \mathbf{u}_{\mathrm{P}_N}, \quad \text{with} \quad \mathbf{A}^{(m)} = \langle \mathbf{\Omega}^{(m)} \mathbf{r} \mathbf{r}^\top \rangle, \tag{13}$$

The matrices $\mathbf{A}^{(m)} \in \mathbb{R}^{n \times n}$ are sparse and symmetric; they can be computed with a recurrence relationship for spherical harmonics, which for real-valued harmonics, can be found in the appendix of [13]; see also Appendix A. Meanwhile, due to (6), the matrix $\mathbf{G} = \langle \mathbf{r} Q \mathbf{r}^\top \rangle \in \mathbb{R}^{n \times n}$ is diagonal with components $G_{(\ell,m),(\ell,m)} = (1 - g_\ell)$. The source vector $\mathbf{s} = \langle \mathbf{r} S \rangle$ and initial condition $\langle \mathbf{r} \psi_0 \rangle$ are known. We write (12) in the compact form

$$\partial_t \mathbf{u}_{\mathrm{P}_N}(\boldsymbol{x}, t) + \mathcal{T} \mathbf{u}_{\mathrm{P}_N}(\boldsymbol{x}, t) = \mathbf{s}(\boldsymbol{x}, t), \tag{14a}$$

$$\mathbf{u}_{\mathrm{P}_N}(\boldsymbol{x}, 0) = \langle \mathbf{r} \psi_0 \rangle, \tag{14b}$$

where $\mathcal{T} \mathbf{v} = \mathbf{A} \cdot \nabla_{\mathbf{x}} \mathbf{v} + \sigma_\mathrm{a} \mathbf{v} + \sigma_\mathrm{s} \mathbf{G} \mathbf{v}$.

[1] Boundary conditions for the P_N equations can be formulated in several ways. We leave them unspecified until needed in the numerical results.

2.3 FP_N equations

The FP_N equations are constructed by adding artificial scattering terms to the P_N equations. These terms appear in the form of a diagonal operator, whose components are expressed in terms of a filter function [19]. The precise definition of this function can vary in the literature; for our purposes, a filter function of order $\alpha \in \mathbb{N}$ is a real-valued, non-negative, and monotonically decreasing function $f \in C^\alpha([0,1])$ that satisfies the following properties:

$$(i)\ f(0) = 1, \qquad (ii)\ f^{(\ell)}(0) = 0,\ \forall \ell \leq \alpha - 1, \qquad (iii)\ f^{(\alpha)}(0) \neq 0. \tag{15}$$

In this paper, we will use an exponential filter of order four:

$$f(s) = \exp(-s^\alpha), \qquad \alpha = 4. \tag{16}$$

With a filter function in hand, the FP_N equations can be written as

$$\partial_t \mathbf{u}_{\mathrm{FP}_N}(\boldsymbol{x},t) + \mathcal{T}\mathbf{u}_{\mathrm{FP}_N}(\boldsymbol{x},t) + \sigma_\mathrm{f}(\boldsymbol{x},t)\mathbf{F}\mathbf{u}_{\mathrm{FP}_N}(\boldsymbol{x},t) = \mathbf{s}(\boldsymbol{x},t), \tag{17a}$$

$$\mathbf{u}_{\mathrm{FP}_N}(\boldsymbol{x},0) = \langle \mathbf{r}\psi_0 \rangle, \tag{17b}$$

where $\sigma_\mathrm{f}(\boldsymbol{x},t) \geq 0$ is the filter strength and $\mathbf{F}$ is a diagonal matrix with components

$$\mathbf{F}_{(\ell,m),(\ell,m)} = -\log\left(f\left(\frac{\ell}{N+1}\right)\right). \tag{18}$$

In particular, the component $\mathbf{F}_{(0,0),(0,0)} = 0$, which implies that the filter does not affect the total particle concentration. The user must select the filter function and strength appropriately for a given setting. In previous work, σ_f was chosen to be constant.

Given a solution of (17), the FP_N approximation of ψ is $\psi_{\mathrm{FP}_N} = \mathbf{r}^\top \mathbf{u}_{\mathrm{FP}_N}$. The convergence of the filtered approximation to the true solution depends on the order of the filter function f, independent of the filter strength. Rigorous results [13] suggest that the order of the filter should be chosen based on the expected regularity of the RTE solution. Roughly speaking, when the filter order α is greater than the angular regularity of ψ, the convergence is still spectral but the rate drops by a small constant factor (less than one). When α is less than the angular regularity of ψ, then the convergence rate is given by α.

Numerical experiments in [25, 30] show that the potential benefits of filtering are greater for small values of N. Moreover, practical constraints typically limit realistic transport simulations with the P_N equations to $N \leq \mathcal{O}(10)$. Unfortunately, the convergence results in [13] provide little guidance about how to select the filter strength σ_f in order to ensure quality results when N is small. These facts motivate our data-driven approach.

3 Data-driven FP_N Equations

We construct a data-driven filter for the P_N equations that depends locally in space and time on $\mathbf{s}$ and on the solution itself. Specifically, we let

$$\sigma_\mathrm{f}(\boldsymbol{x},t) = \Sigma_\mathrm{f}\big(\boldsymbol{\xi}(\mathbf{u}_{\mathrm{FP}_N}(\boldsymbol{x},t)), \mathbf{s}(\boldsymbol{x},t); \boldsymbol{w}_{\mathrm{FP}_N}\big), \tag{19}$$

where the ansatz $\Sigma_\mathrm{f} \colon (\boldsymbol{\xi}, \mathbf{s}; \boldsymbol{w}) \mapsto \mathbb{R}^{\geq 0}$ takes a feature vector $\boldsymbol{\xi}$, the source $\mathbf{s}$, and a parameter vector $\boldsymbol{w}$ and outputs the corresponding filter strength. The parameter vector $\boldsymbol{w}_{\mathrm{FP}_N}$ in (19) is determined by a training procedure. The ansatz Σ_f can take on a variety of functional forms. Here we use neural networks with weights $\boldsymbol{w}$ that are trained based on the L^2 error in the particle concentration $\phi = \langle \psi \rangle$ at some final time t_final. For the input features in $\boldsymbol{\xi}$, we choose each of the terms in the operator $\mathcal{T}$ in the P_N equations (12). Specifically, for any sufficiently smooth function $\mathbf{v} : X \to \mathbb{R}^n$, $\boldsymbol{\xi}(\mathbf{v}) : X \to \mathbb{R}^{n\times 3}$ is given by

$$\boldsymbol{\xi}(\mathbf{v}) = [\mathbf{A} \cdot \nabla_{\boldsymbol{x}}\mathbf{v} \quad \sigma_\mathrm{t}\mathbf{v} \quad \sigma_\mathrm{s}(\mathbf{G} - \mathbf{I})\mathbf{v}]. \tag{20}$$

Let the random variable γ parameterize the distribution of sources, initial conditions, and cross-sections that are used for training. Let $\phi^\gamma = \langle \psi^\gamma \rangle$, where $\langle \psi^\gamma \rangle$ solves (1) with source S^γ, initial condition ψ_0^γ, and

cross-sections $\sigma_{\mathrm{a}}^{\gamma}$ and $\sigma_{\mathrm{s}}^{\gamma}$. We assume that for all parameterizations γ, there exists a unique solution of the FP_N equations

$$\partial_t \mathbf{u}_{\mathrm{FP}_N}^{\gamma} + \mathcal{T}\mathbf{u}_{\mathrm{FP}_N}^{\gamma} + \Sigma_{\mathrm{f}}(\boldsymbol{\xi}, \langle \mathbf{r} S^{\gamma} \rangle; \boldsymbol{w}) \mathbf{F} \mathbf{u}_{\mathrm{FP}_N}^{\gamma} = \langle \mathbf{r} S^{\gamma} \rangle \tag{21a}$$

$$\mathbf{u}_{\mathrm{FP}_N}^{\gamma}(\boldsymbol{x}, 0) = \langle \mathbf{r} \psi_0^{\gamma} \rangle \tag{21b}$$

and define the map

$$\Phi^{\gamma} : \boldsymbol{w} \mapsto \phi_{\mathrm{FP}_N}^{\gamma}(\cdot, t_{\mathrm{final}}), \tag{22}$$

which takes as input the parameter $\boldsymbol{w}$ and returns the particle concentration $\phi_{\mathrm{FP}_N}^{\gamma} = \langle \psi_{\mathrm{FP}_N}^{\gamma} \rangle = \sqrt{4\pi}[u_{\mathrm{FP}_N}^{\gamma}]_0^0$ at the time t_{final}. [2]

We formulate the training as a PDE-constrained minimization problem

$$\min_{\boldsymbol{w}} J(\boldsymbol{w}), \quad \text{where } J(\boldsymbol{w}) = \mathbb{E}_{\gamma}[j(\Phi^{\gamma}(\boldsymbol{w}); \phi^{\gamma}(\cdot, t_{\mathrm{final}}))] \tag{23}$$

and for any functions $a, b \in L^2(X)$

$$j(a; b) = \frac{1}{4\pi} \left\| a - b \right\|_{L^2(X)}^2 . \tag{24}$$

In practice the reference solution ϕ^{γ} is itself approximated in angle with a P_N solution with N very large.

3.1 Discretization of the PDE Constraint

In practice, the FP_N equations must be discretized in space and time. To this end, we employ a second-order finite-volume scheme in space and Heun's method in time. This discretization yields an approximate solution $\bar{\mathbf{v}}^{\gamma}$ that is constant in each spatial mesh cell and defined on a finite set of temporal points $\{0 = t_0, t_1, \ldots, t_K = t_{\mathrm{final}}\}$ such that $\bar{\mathbf{v}}^{\gamma}(\boldsymbol{x}, t_k) \approx \mathbf{u}_{\mathrm{FP}_N}^{\gamma}(\boldsymbol{x}, t_k)$ for all $\boldsymbol{x} \in X$.

Let $\vec{\mathbf{v}}^{\gamma,k}$ be an array that contains the cell averages of $\bar{\mathbf{v}}^{\gamma}(\cdot, t_k)$ on each spatial mesh cell. For each k, $\vec{\mathbf{v}}^{\gamma,k}$ evolves according to an update rule of the form

$$\vec{\mathbf{v}}^{\gamma,k+1}(\boldsymbol{w}) = \mathbf{H}^{\gamma,k}(\vec{\mathbf{v}}^{\gamma,k}(\boldsymbol{w}); \boldsymbol{w}), \quad k \in \{0, \ldots, K-1\}, \tag{25}$$

where $\vec{\mathbf{v}}^{\gamma,0}$ is computed from the cell-wise averages $\mathbf{u}_{\mathrm{FP}_N}^{\gamma}(\cdot, 0)$. The details of the operator $H^{\gamma,k}$ come from the the finite-volume discretization that is described in Appendix B.3. Define the discrete solution operators $\bar{\Phi}^{\gamma}$ and $\vec{\Phi}^{\gamma}$ by

$$\bar{\Phi}^{\gamma} : \boldsymbol{w} \mapsto \bar{\phi}_{\mathrm{FP}_N}^{\gamma}(\cdot, t_{\mathrm{final}}) \qquad \text{and} \qquad \vec{\Phi}^{\gamma} : \boldsymbol{w} \mapsto \vec{\phi}_{\mathrm{FP}_N}^{\gamma,K}, \tag{26}$$

where, for each $k \in \{0, \ldots, K\}$, $\bar{\phi}_{\mathrm{FP}_N}^{\gamma}(\cdot, t_k)$ is the piecewise-constant (in space) approximation of $\phi_{\mathrm{FP}_N}^{\gamma}(\cdot, t_k)$ and $\vec{\phi}_{\mathrm{FP}_N}^{\gamma,K}$ is the corresponding vector of cell-averages computed using the same finite volume method in (25). The operators in (26) are related by a linear, invertible reconstruction operator $\mathcal{E}$; that is, $\bar{\Phi}^{\gamma}(\boldsymbol{w}) = \mathcal{E}\vec{\Phi}^{\gamma}(\boldsymbol{w})$.

From (23), we obtain the discrete objective function

$$\bar{J}(\boldsymbol{w}) = \mathbb{E}_{\gamma}[j(\bar{\Phi}^{\gamma}(\boldsymbol{w}); \bar{\phi}^{\gamma}(\cdot, t_{\mathrm{final}}))] = \mathbb{E}_{\gamma}[j(\mathcal{E}\vec{\Phi}^{\gamma}(\boldsymbol{w}); \mathcal{E}\vec{\phi}^{\gamma,K})]. \tag{27}$$

where the piece-wise constant function $\bar{\phi}^{\gamma}(\cdot, t_k)$ with cell averages in $\vec{\phi}^{\gamma,t_k}$ approximates $\phi^{\gamma}(\cdot, k)$ with the same finite volume method. We expect that $\bar{J}^{\gamma}$ is differentiable with respect to $\boldsymbol{w}$ and thus we choose a gradient descent-based optimizer to find optimal parameters $\boldsymbol{w}$.

Remark 1. *The approach here is called discretize, then optimize in PDE-constrainted optimization literature, see, e.g. [20]. Alternatively, one can formally derive the Fréchet derivative of J with respect to $\boldsymbol{w}$ which yields an adjoint set of equations. The adjoint FP_N equations then need to be discretized to facilitate the optimization. This approach is called optimize, then discretize. In general the approaches are not equivalent.*

[2] The difference in the concentration ϕ_{FP_N} and the moment $[u_{\mathrm{P}_N}]_0^0$ is due to the normalization of $r_0^0 = 1/\sqrt{4\pi}$.

3.2 Gradient-based Optimization

At each iteration, we approximate the gradient of $\bar{J}$ using a batch B of training data with realizations $\gamma_b \in B$: [3]

$$\bar{J}(\boldsymbol{w}) \approx \bar{J}^B(\boldsymbol{w}) = \frac{1}{|B|} \sum_{\gamma_b \in B} j(\bar{\Phi}^\gamma(\boldsymbol{w}); \bar{\phi}^\gamma(\cdot, t_{\text{final}})). \tag{28}$$

The gradient descent procedure is then

$$\boldsymbol{w}^{\ell+1} = \boldsymbol{w}^\ell - \lambda \nabla_{\boldsymbol{w}} \bar{J}^B(\boldsymbol{w}^\ell), \tag{29}$$

where $\lambda > 0$ is the learning rate and the initial weights $\boldsymbol{w}^{\ell=0}$ are normally distributed.

In practice, the gradient in (29) is computed efficiently via automatic differentiation. For the sake of completeness, we describe the underlying components of the computation. First, $\nabla_{\boldsymbol{w}} \bar{J}^B = [\frac{\partial \bar{J}^B}{\partial \boldsymbol{w}}]^\top$ is computed via the chain rule

$$\frac{\partial \bar{J}^B}{\partial \boldsymbol{w}}(\boldsymbol{w}) = \frac{1}{|B|} \sum_{\gamma_b \in B} \mathcal{D}j(\bar{\Phi}^{\gamma_b}(\boldsymbol{w}); \bar{\phi}^\gamma(\cdot, t_{\text{final}})) \left[\frac{\partial \bar{\Phi}^{\gamma_b}}{\partial \boldsymbol{w}}(\boldsymbol{w}) \right] = \frac{1}{|B|} \sum_{\gamma_b \in B} \mathcal{D}j(\bar{\Phi}^{\gamma_b}(\boldsymbol{w}); \bar{\phi}^\gamma(\cdot, t_{\text{final}})) \left[\mathcal{E} \frac{\partial \vec{\Phi}^{\gamma_b}}{\partial \boldsymbol{w}}(\boldsymbol{w}) \right], \tag{30}$$

where $\mathcal{D}j$ is the Fréchet derivative of j, which in (30) is evaluated at $\bar{\Phi}^{\gamma_b}(\boldsymbol{w})$ and acts on $\mathcal{E} \frac{\partial \vec{\Phi}^{\gamma_b}}{\partial \boldsymbol{w}}(\boldsymbol{w})$. For any functions $a, b, c \in L^2(X)$

$$Dj(a; b)[c] = \lim_{\varepsilon \to 0^+} \frac{d}{d\varepsilon} j(a + \varepsilon c; b) = \frac{1}{2\pi} \int_X (a(\boldsymbol{x}) - b(\boldsymbol{x}))\, c(\boldsymbol{x}) d\boldsymbol{x}. \tag{31}$$

The remaining derivative needed in (30) is

$$\frac{\partial \vec{\Phi}^{\gamma_b}}{\partial \boldsymbol{w}} = \sqrt{4\pi} \left[\frac{\partial \vec{\mathbf{v}}^{\gamma_b, K}}{\partial \boldsymbol{w}} \right]_0^0, \tag{32}$$

which is computed recursively in k, starting with $k = K$ and using (25), as described at the end of Appendix B.3

3.3 Feature Selection and Rotational Invariance

In this subsection, we introduce the notion of rotational invariance for the RTE and P_N equations and establish sufficient conditions on Σ_{f} for the FP_N equations to preserve this property.

Definition 1. *Given any orthogonal matrix $O \in \mathbb{R}^{3\times 3}$, the operator $\mathcal{R}_O$ induced by O is given by*

$$(\mathcal{R}_O f)(\boldsymbol{x}, \boldsymbol{\Omega}) = f(Ox, O\boldsymbol{\Omega}), \quad \text{for any } f : X \times \mathbb{S}^2 \to \mathbb{R}. \tag{33}$$

The operator $\mathcal{W}_O$ acting on the corresponding moment $\mathbf{v} = \langle \mathbf{r} f \rangle$ is given by

$$(\mathcal{W}_O \mathbf{v})(\boldsymbol{x}) = \langle \mathbf{r}\, (\mathcal{R}_O f)(\boldsymbol{x}, \cdot) \rangle\,. \tag{34}$$

Remark 2. *In what follows, we may apply $\mathcal{R}_O$ to a function of $\boldsymbol{x}$ only, in which case $\mathcal{R}_O f(\boldsymbol{x}) = f(O\boldsymbol{x})$. For vector-valued functions, $\mathcal{R}_O$ is applied component-wise.*

Lemma 1. *Let $O \in \mathbb{R}^{3\times 3}$ be an orthogonal matrix. Then for each $\ell \in \{0, \dots, N\}$, there exists an orthogonal matrix $\mathbf{Z}_O^\ell \in \mathbb{R}^{(2\ell+1)\times(2\ell+1)}$ such that*

$$\mathbf{r}_\ell(\boldsymbol{\Omega}) = \mathbf{Z}_O^\ell \mathbf{r}_\ell(O\boldsymbol{\Omega}). \tag{35}$$

Thus $\mathbf{r}(\boldsymbol{\Omega}) = \mathbf{Z}_O \mathbf{r}(O\boldsymbol{\Omega})$, where $\mathbf{Z} = \mathrm{diag}(\mathbf{Z}_O^0, \dots, \mathbf{Z}_O^N) \in \mathbb{R}^{n\times n}$. Moreover,

$$(\mathcal{W}_O \mathbf{v})(\boldsymbol{x}) = (\mathbf{Z}_O \mathbf{v})(O\boldsymbol{x}). \tag{36}$$

[3] The details for this sampling will be discussed in the numerical results.

Proof. We follow the proof from [14, Lemma 1]. For each polynomial order ℓ, $\text{span}\{r_\ell^{-\ell}, \dots, r_\ell^0, \dots, r_\ell^\ell\}$ is invariant under orthogonal transformations in $\mathbf{\Omega}$ [11, Theorem 1.1.7]. Thus there exists a matrix $\mathbf{Z}_O^\ell \in \mathbb{R}^{(2\ell+1)\times(2\ell+1)}$ such that (35) holds. Let $\hat{\mathbf{\Omega}} = O\mathbf{\Omega}$ so that $d\hat{\mathbf{\Omega}} = d\mathbf{\Omega}$. Since the components of $\mathbf{r}_\ell$ are orthonormal,

$$I = \int_{\mathbb{S}^2} \mathbf{r}_\ell(\mathbf{\Omega})\mathbf{r}_\ell(\mathbf{\Omega})^\top d\mathbf{\Omega} = \int_{\mathbb{S}^2} \mathbf{Z}_O^\ell \mathbf{r}_\ell(\hat{\mathbf{\Omega}})\mathbf{r}_\ell(\hat{\mathbf{\Omega}})^\top (\mathbf{Z}_O^\ell)^\top d\hat{\mathbf{\Omega}} = \mathbf{Z}_O^\ell (\mathbf{Z}_O^\ell)^\top. \tag{37}$$

It follows that $\mathbf{Z}_O^\ell$ is orthogonal and hence so is $\mathbf{Z}_O$. To show (36), direct application of (34) and (35) gives

$$(\mathcal{W}_O \mathbf{v})(\boldsymbol{x}) = \int_{\mathbb{S}^2} \mathbf{r}(\mathbf{\Omega}) f(O\boldsymbol{x}, O\mathbf{\Omega})\, \mathrm{d}\mathbf{\Omega} = \mathbf{Z}_O \int_{\mathbb{S}^2} \mathbf{r}(\hat{\mathbf{\Omega}}) f(O\boldsymbol{x}, \hat{\mathbf{\Omega}})\, \mathrm{d}\hat{\mathbf{\Omega}} = \mathbf{Z}_O \mathbf{v}(O\boldsymbol{x}). \tag{38}$$

□

Remark 3. *Recursive algorithms to compute $\mathbf{Z}_O$ are given in [5, 10, 23], and an explicit example for $N = 1$ is provided in [15, Eq. (11)].*

Proposition 1. *The RTE* (1) *is rotationally invariant in the sense that ψ satisfies* (1) *with source S if and only if, for any orthogonal matrix $O \in \mathbb{R}^{3\times3}$, $\mathcal{R}_O\psi$ satisfies* (1) *with a source $\mathcal{R}_O S$ and cross-sections $\mathcal{R}_O\sigma_\mathrm{a}$ and $\mathcal{R}_O\sigma_\mathrm{s}$. Similarly, the P_N equations* (12) *are rotationally invariant in the sense that $\mathbf{u}_{\mathrm{P}_N}$ satisfies* (12) *with source $\mathbf{s}$, if and only if for any orthogonal matrix $O \in \mathbb{R}^{3\times3}$, $\mathcal{W}_O\mathbf{u}_{\mathrm{P}_N}$ satisfies* (12) *with a source $\mathcal{W}_O\mathbf{s}$ and cross-sections $\mathcal{R}_O\sigma_\mathrm{a}$ and $\mathcal{R}_O\sigma_\mathrm{s}$.*

Proof. These result are well known; see e.g., [2]. We provide a proof here for completeness. For the RTE, the result amounts to proving that $\mathcal{R}_O$ commutes with the operators $\mathbf{\Omega}\cdot\nabla_{\boldsymbol{x}}$ and $\mathcal{Q}$. Let $\hat{\boldsymbol{x}} = O\boldsymbol{x}$, $\hat{\mathbf{\Omega}} = O\mathbf{\Omega}$, and $\hat{\mathbf{\Omega}}' = O\mathbf{\Omega}'$. Direct calculation gives

$$\mathcal{R}_O(\mathbf{\Omega}\cdot\nabla_x\psi)(\boldsymbol{x},\mathbf{\Omega},t) = \hat{\mathbf{\Omega}}\cdot\nabla_{\hat{\boldsymbol{x}}}\psi(\hat{\boldsymbol{x}},\hat{\mathbf{\Omega}},t) = O\mathbf{\Omega}\cdot O\nabla_{\boldsymbol{x}}\psi(\hat{\boldsymbol{x}},\hat{\mathbf{\Omega}},t) = \mathbf{\Omega}\cdot\nabla_{\boldsymbol{x}}(\mathcal{R}_O\psi)(\boldsymbol{x},\mathbf{\Omega},t) \tag{39}$$

and, since $d\hat{\mathbf{\Omega}}' = d\mathbf{\Omega}'$,

$$\begin{aligned}\mathcal{R}_O\left(\int_{\mathbb{S}^2} g(\mathbf{\Omega}\cdot\mathbf{\Omega}')\psi(\boldsymbol{x},\mathbf{\Omega}',t)d\mathbf{\Omega}'\right) &= \int_{\mathbb{S}^2} g(\hat{\mathbf{\Omega}}\cdot\mathbf{\Omega}')\psi(\hat{\boldsymbol{x}},\mathbf{\Omega}',t)d\mathbf{\Omega}' = \int_{\mathbb{S}^2} g(\hat{\mathbf{\Omega}}\cdot\hat{\mathbf{\Omega}}')\psi(\hat{\boldsymbol{x}},\hat{\mathbf{\Omega}}',t)d\hat{\mathbf{\Omega}}' \\ &= \int_{\mathbb{S}^2} g(\mathbf{\Omega}\cdot\mathbf{\Omega}')\psi(\hat{\boldsymbol{x}},\hat{\mathbf{\Omega}}',t)d\mathbf{\Omega}' = \int_{\mathbb{S}^2} g(\mathbf{\Omega}\cdot\mathbf{\Omega}')(\mathcal{R}_O\psi)(\boldsymbol{x},\mathbf{\Omega}',t)d\mathbf{\Omega}'.\end{aligned} \tag{40}$$

Similarly, for the P_N equations, the result amounts to proving that the operator $\mathcal{W}_O$ commutes with the operators $\mathbf{A}\cdot\nabla_{\boldsymbol{x}}$ and $\mathbf{G}$. Direction calculation, The using the relation $\mathbf{Z}_O\mathbf{r}(O\mathbf{\Omega})$, gives

$$\begin{aligned}\mathcal{W}_O(\mathbf{A}\cdot\nabla_{\boldsymbol{x}}\mathbf{u}_{\mathrm{P}_N})(\boldsymbol{x},t) &= \mathcal{W}_O\left(\int_{\mathbb{S}^2}\mathbf{r}(\hat{\mathbf{\Omega}})\mathbf{r}(\hat{\mathbf{\Omega}})^\top\hat{\mathbf{\Omega}}\cdot\nabla_{\boldsymbol{x}}\mathbf{u}_{\mathrm{P}_N}(\boldsymbol{x},t)d\hat{\mathbf{\Omega}}\right) = \mathbf{Z}_O\int_{\mathbb{S}^2}\mathbf{r}(\hat{\mathbf{\Omega}})\mathbf{r}(\hat{\mathbf{\Omega}})^\top\hat{\mathbf{\Omega}}\cdot\nabla_{\hat{\boldsymbol{x}}}\mathbf{u}_{\mathrm{P}_N}(\hat{\boldsymbol{x}},t)d\hat{\mathbf{\Omega}} \\ &= \int_{\mathbb{S}^2}\mathbf{r}(\mathbf{\Omega})\mathbf{r}^\top(\mathbf{\Omega})\mathbf{\Omega}\cdot\nabla_x\mathbf{Z}_O\mathbf{u}_{\mathrm{P}_N}(\hat{\boldsymbol{x}},t)d\mathbf{\Omega} = \mathbf{A}\cdot\nabla_{\boldsymbol{x}}(\mathcal{W}_O\mathbf{u}_{\mathrm{P}_N})(\boldsymbol{x},t),\end{aligned} \tag{41}$$

and, because $\mathbf{Z}_O$ and $\mathbf{G}$ have the same block structure and each block of $\mathbf{G}$ is a multiple of the identity, $\mathbf{Z}_O$ and $\mathbf{G}$ commute. Hence

$$\mathcal{W}_O(\mathbf{G}\mathbf{u}_{\mathrm{P}_N})(\boldsymbol{x},t) = \mathbf{Z}_O\mathbf{G}\mathbf{u}_{\mathrm{P}_N}(\hat{\boldsymbol{x}},t) = \mathbf{G}\mathbf{Z}_O\mathbf{u}_{\mathrm{P}_N}(\hat{\boldsymbol{x}},t) = \mathbf{G}(\mathcal{W}_O\mathbf{u}_{\mathrm{P}_N})(\boldsymbol{x},t). \tag{42}$$

□

When the filter strength is constant, the FP_N solution inherits the rotational invariance of the underlying P_N solution. However, when the filter strength depends on the FP_N solution, rotational invariance must be explicitly enforced.

Theorem 1. *Suppose that for any $\mathbf{v} : X \to \mathbb{R}^n$ sufficiently smooth, Σ_f satisfies*

$$\Sigma_\mathrm{f}(\boldsymbol{\xi}(\mathbf{v}), \mathbf{s}; \boldsymbol{w}) = \Sigma_\mathrm{f}(\mathbf{Z}_O\boldsymbol{\xi}(\mathbf{v}), \mathbf{Z}_O\mathbf{s}; \boldsymbol{w}). \tag{43}$$

Then the FP_N equations (17) are rotationally invariant in the sense that $\mathbf{u}_{\mathrm{FP}_N}$ satisfies (17) with source $\mathbf{s}$ if and only if for any orthogonal matrix $O \in \mathbb{R}^{3\times 3}$, $\mathcal{W}_O\mathbf{u}_{\mathrm{FP}_N}$ satisfies (17) with a source $\mathcal{W}_O\mathbf{s}$ and cross-sections $\mathcal{R}_O\sigma_\mathrm{a}$ and $\mathcal{R}_O\sigma_\mathrm{s}$.

Proof. The proof is similar to the proof for the P_N equations, but with the additional filter term. As before, let $\hat{\boldsymbol{x}} = O\boldsymbol{x}$. Then as with $\mathbf{G}$, $\mathbf{Z}_O$ and $\mathbf{F}$ commute. Hence

$$\mathcal{W}_O\big(\sigma_\mathrm{f}(\boldsymbol{x},t)\mathbf{F}\mathbf{u}_{\mathrm{FP}_N}(\boldsymbol{x},t)\big) = \sigma_\mathrm{f}(\hat{\boldsymbol{x}},t)\mathbf{F}\mathbf{Z}_O\mathbf{u}_{\mathrm{FP}_N}(\hat{\boldsymbol{x}},t) = \sigma_\mathrm{f}(\hat{\boldsymbol{x}},t)\mathbf{F}(\mathcal{W}_O\mathbf{u}_{\mathrm{FP}_N})(\boldsymbol{x},t), \tag{44}$$

where σ_f is given by (19). By the assumption on Σ_f in (43), for any parameter vector $\boldsymbol{w}$,

$$\sigma_\mathrm{f}(\hat{\boldsymbol{x}},t) = \Sigma_\mathrm{f}\big(\boldsymbol{\xi}(\mathbf{u}_{\mathrm{FP}_N})(\hat{\boldsymbol{x}},t), \mathbf{s}(\hat{\boldsymbol{x}}); \boldsymbol{w}\big) = \Sigma_\mathrm{f}(\mathbf{Z}_O\boldsymbol{\xi}(\mathbf{u}_{\mathrm{FP}_N})(\hat{\boldsymbol{x}},t), \mathbf{Z}_O\mathbf{s}(\hat{\boldsymbol{x}}); \boldsymbol{w}). \tag{45}$$

However, for any sufficiently smooth function $\mathbf{v}\colon X \to \mathbb{R}^n$

$$\begin{aligned} Z_O\boldsymbol{\xi}(\mathbf{v})(\hat{\boldsymbol{x}}) &= [Z_O\mathbf{A}\cdot\nabla_{\boldsymbol{x}}\mathbf{v}(\hat{\boldsymbol{x}}) \quad \sigma_\mathrm{t}(\hat{\boldsymbol{x}})Z_O\mathbf{v}(\hat{\boldsymbol{x}}) \quad \sigma_\mathrm{s}(\hat{\boldsymbol{x}})Z_O(\mathbf{G}-\mathbf{I})\mathbf{v}(\hat{\boldsymbol{x}})] \\ &= [\mathcal{W}_O(\mathbf{A}\cdot\nabla_{\boldsymbol{x}}\mathbf{v})(\boldsymbol{x}) \quad \mathcal{W}_O(\sigma_\mathrm{t}\mathbf{v})(\boldsymbol{x}) \quad \mathcal{W}_O(\sigma_\mathrm{s}(\mathbf{G}-\mathbf{I})\mathbf{v}))(\boldsymbol{x})] \end{aligned} \tag{46}$$

where the last line in (46) follows from the calculations in (41) and (42). In addition, $\mathbf{Z}_O\mathbf{s}(\hat{\boldsymbol{x}}) = (\mathcal{W}_O\mathbf{s})(\boldsymbol{x})$ by definition. Hence (45) becomes

$$\sigma_\mathrm{f}(\hat{\boldsymbol{x}},t) = \Sigma_\mathrm{f}(\boldsymbol{\xi}(\mathbf{Z}_O\mathbf{u}_{\mathrm{FP}_N})(\hat{\boldsymbol{x}},t), \mathbf{Z}_O\mathbf{s}(\hat{\boldsymbol{x}}); \boldsymbol{w}) = \Sigma_\mathrm{f}(\boldsymbol{\xi}(\mathcal{W}_O\mathbf{u}_{\mathrm{FP}_N})(\boldsymbol{x},t), (\mathcal{W}_O\mathbf{s})(\boldsymbol{x}); \boldsymbol{w}) \tag{47}$$

Thus the proof is complete. □

Given $\mathbf{u} = [\mathbf{u}_0, \ldots, \mathbf{u}_\ell, \ldots, \mathbf{u}_N]^\top \in \mathbb{R}^1 \times \cdots \times \mathbb{R}^{2\ell+1} \times \cdots \times \mathbb{R}^{2N+1} = \mathbb{R}^n$, let $|\mathbf{u}_\ell| = \sqrt{\mathbf{u}_\ell^\top\mathbf{u}_\ell}$ be the Euclidean norm of $\mathbf{u}_\ell$. To satisfy the condition in (43), we pre-process the features $\boldsymbol{\xi}(\mathbf{u}_{\mathrm{FP}_N})$ using the operator $\mathbf{P}\colon \mathbb{R}^n \to \mathbb{R}^{N+1}$, given by

$$\mathbf{P}\mathbf{u} = [|\mathbf{u}_0|, \ldots, |\mathbf{u}_N|]^\top. \tag{48}$$

Proposition 2. *For any neural network $\mathcal{N}$ with parameters $\boldsymbol{w}$, the ansatz*

$$\Sigma_\mathrm{f}(\boldsymbol{\xi}(\cdot), \mathbf{s}; \boldsymbol{w}) = \mathcal{N}(\mathbf{P}\boldsymbol{\xi}(\cdot), \mathbf{P}\mathbf{s}; \boldsymbol{w}) \tag{49}$$

fulfills (43).

Proof. Due to the block structure of $\mathbf{Z}_O$ and orthogonality of each block (see Lemma 1) it follows that $\mathbf{P}\mathbf{u} = \mathbf{P}\mathbf{Z}_O\mathbf{u}$ for any $\mathbf{u} \in \mathbb{R}^n$. Hence $\mathbf{P}\boldsymbol{\xi}(\cdot) = \mathbf{P}\mathbf{Z}_O\boldsymbol{\xi}(\cdot)$ and

$$\Sigma_\mathrm{f}(\boldsymbol{\xi}(\cdot), \mathbf{s}; \boldsymbol{w}) = \mathcal{N}(\mathbf{P}\boldsymbol{\xi}(\cdot), \mathbf{P}\mathbf{s}; \boldsymbol{w}) = \mathcal{N}(\mathbf{P}\mathbf{Z}_O\boldsymbol{\xi}(\cdot), \mathbf{P}\mathbf{Z}_O\mathbf{s}; \boldsymbol{w}) = \Sigma_\mathrm{f}(\mathbf{Z}_O\boldsymbol{\xi}(\cdot), \mathbf{Z}_O\mathbf{s}; \boldsymbol{w}), \tag{50}$$

which is exactly (43). □

4 Numerical Results

In the following, we perform numerical tests with FP_N approximations in 1-D and 2-D geometries, which are both described in Appendix B. We explore several test cases to evaluate the performance of the data-driven FP_N equations. All numerical results can be reproduced using the GitHub repository `https://github.com/benplumridge/NN_FPN`.

We model the data-driven filter Σ_f by applying the map $\mathbf{P}$ from (48) to each component of the feature vector in (20), obtaining the preprocessed features

$$\mathbf{P}\boldsymbol{\xi}(\mathbf{v}) = [\mathbf{P}\mathbf{A}\cdot\nabla_{\boldsymbol{x}}\mathbf{v} \quad \sigma_\mathrm{t}\mathbf{P}\mathbf{v} \quad \sigma_\mathrm{s}\mathbf{P}(\mathbf{G}-\mathbf{I})\mathbf{v}]. \tag{51}$$

In general there are $3N+3$ features ($N+1$ for each column of $\boldsymbol{\xi}(\mathbf{v})$). However, for isotropic scattering, which is assumed in all of the problems below, only the $\ell = 0$ component of the vector $\sigma_\mathrm{s}\mathbf{P}(\mathbf{G}-\mathbf{I})\mathbf{v}$ is nonzero.

Any neural network architecture with non-negative scalar output and the preprocessor $\mathbf{P}$ is a feasible choice for the proposed framework. Thus as a baseline comparison, we train a scalar filter, i.e. $\Sigma_\mathrm{f} = w \in \mathbb{R}_+$, which is independent of the state of the moment system. The corresponding filter values are displayed in Table 1. For the neural network based and constant filter, each ansatz is trained for a fixed moment order N; once trained it can be used in any simulation with the same moment order N and same space-time discretization.

	1-D			2-D			
N	3	7	9	3	5	7	9
Σ_f	7.74	6.47	4.76	31.75	27.27	16.52	13.62

Table 1: Optimal scalar constant $\Sigma_f = w \in \mathbb{R}^+$ for 1-D and 2-D. Values are obtained using a degenerate network consisting of a single bias, trained on the same data as the neural network ansatz. For testing in 1-D the network is trained multiple times, and the values above are one representative instance.

As validation metrics we measure relative error of the particle concentration to the reference solution for the P_N solution (e_N), the FP_N solution with neural network filter strength (e_N^nn), and the FP_N solution with constant, but trainable, filter strength (e_N^const):

$$e_N = \frac{\|\phi_{\mathrm{P}_N} - \phi\|_{L^2(X)}}{\|\phi\|_{L^2(X)}}, \qquad e_N^\mathrm{nn} := \frac{\left\|\phi_{\mathrm{FP}_N}^\mathrm{nn} - \phi\right\|_{L^2(X)}}{\|\phi\|_{L^2(X)}}, \quad \text{and} \quad e_N^\mathrm{const} := \frac{\left\|\phi_{\mathrm{FP}_N}^\mathrm{const} - \phi\right\|_{L^2(X)}}{\|\phi\|_{L^2(X)}}, \tag{52}$$

where the reference ϕ is computed using a high-order P_N solution and the same space-time discretization and the other solutions. We also report the ratios

$$r_N^\mathrm{nn} = \frac{e_N^\mathrm{nn}}{e_N} \qquad \text{and} \qquad r_N^\mathrm{const} = \frac{e_N^\mathrm{const}}{e_N} \tag{53}$$

to quantity the improvement provided by the filtering over the standard P_N method.

4.1 1-D Tests

In slab geometries, the P_N and FP_N equations can be reduced to equations in one spatial variable $z \in [z_\mathrm{L}, z_\mathrm{R}]$. In this setting, the kinetic distribution $\psi = \psi(z,\mu)$, where $\mu = \Omega^{(3)}$ (see (4)), is approximated by the expansion $\psi_{\mathrm{FP}_N}(z,\mu) = \tilde{\mathbf{p}}^\top(\mu)\mathbf{v}(z,t)$, where $\tilde{\mathbf{p}} : [-1,1] \to \mathbb{R}$ is a vector-valued functions whose components are the Legendre polynomials normalized with respect to $L^2[-1,1]$ and $\mathbf{v} = [v_0, \ldots, v_N]^\top$ satisfies the 1-D FP_N equations

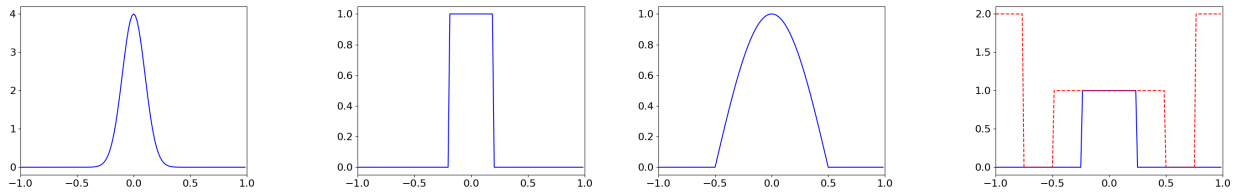
$$\partial_t \mathbf{v} + \tilde{\mathbf{A}}\partial_z \mathbf{v} + \sigma_\mathrm{a}\mathbf{v} + \sigma_\mathrm{s}\tilde{\mathbf{G}}\mathbf{v} + \sigma_\mathrm{f}\tilde{\mathbf{F}}\mathbf{v} = \tilde{\mathbf{s}}, \tag{54}$$

with matrices $\tilde{\mathbf{A}}$, $\tilde{\mathbf{G}}$, and $\tilde{\mathbf{F}}$ that are defined in Appendix B.1. The physical cross-sections and isotropic source $\tilde{\mathbf{s}} = [\tilde{s}_0, \ldots, \tilde{s}_N]^\top$ are all functions of z, while the filter strength σ_f depends on z and t through the state $\mathbf{v}$ and source $\tilde{\mathbf{s}}$.

We test the data-driven FP_N solver for $N \in \{3,7,9\}$. In both training and test cases, the initial condition and source are isotropic; that is $v_\ell|_{t=0} = \tilde{s}_\ell = 0$ for all $\ell > 0$. Thus we only need to specify $v_0|_{t=0}$ and $\tilde{s}_0$. Moreover, except for Reed's Problem, all components of v are zero on the boundary (vacuum boundary condition).

4.1.1 Data Sampling Strategy

All simulation setups to generate training data use the spatial domain $[z_\mathrm{L}, z_\mathrm{R}] = [-1,1]$ with zero boundary conditions for all components of $\mathbf{v}$ on both sides of the domain. We specify the random variable γ that constitutes the data sampling strategy to train the data-driven filter for the 1-D test cases. Specifically, γ describes the choice of initial condition for v_0 and source s_0, selecting from



(a) Gaussian (b) Square-Wave (c) Bump function (d) Discontinuous Source

Figure 1: Simulation setups for the 1-D data generation. The blue solid line denotes the initial data and the red dotted line denotes the source.

(a) a centered Gaussian with standard deviation 0.1 as initial condition:

$$v_0(z, t = 0) = \frac{1}{\sqrt{2\pi\varsigma^2}} \exp\left(-\frac{z^2}{2\varsigma^2}\right), \quad \varsigma = 0.1 \tag{55}$$

and zero source;

(b) the indicator function on the interval $[-0.25, 0.25]$ as initial condition:

$$v_0(z, t = 0) = \begin{cases} 0, & |z| > 0.25, \\ 1, & |z| \leq 0.25, \end{cases} \tag{56}$$

and zero source;

(c) a bump function centered at 0 as initial condition:

$$v_0(z, t = 0) = \begin{cases} 0, & |z| > 0.5, \\ \cos(\pi z), & |z| \leq 0.5, \end{cases} \tag{57}$$

and zero source;

(d) the indicator function on the interval $[-0.25, 0.25]$ and a source term given by

$$\tilde{s}_0(z) = 2\mathbb{1}_{z<-0.75} + 2\mathbb{1}_{z>0.75} + \mathbb{1}_{-0.25<z<0.25}. \tag{58}$$

These simulation setups are illustrated in Figure 1. All setups are simulated until final time $t_{\text{final}} = 0.5$. The value of γ also indicates the value for the scattering and absorption cross section. For each of the four initial condition and source setups above, constant values of σ_{s} and σ_{a} are chosen randomly:

$$\sigma_{\text{s}} \sim \mathcal{U}(0, 1), \quad \sigma_{\text{a}} \sim \mathcal{U}(0, 1 - \sigma_{\text{s}}), \tag{59}$$

where $\mathcal{U}(a, b)$ denotes the uniform distribution on the interval (a, b).

4.1.2 Neural Network Architecture and Training

We train the filter strength Σ_{f} for $N \in \{3, 7, 9\}$, modeling the filter strength with a fully connected feed-forward neural network with a single hidden layer and scalar output. ReLu activation functions are applied to both the hidden and output layers. The features are preprocessed and normalized before being fed into the network. The weights and biases of the hidden and output layers, $\boldsymbol{W}^{(1)}, \boldsymbol{b}^{(1)}$ and $\boldsymbol{W}^{(2)}, \boldsymbol{b}^{(2)}$ are initialized from uniform distributions:

$$\boldsymbol{W}^{(1)} \sim \mathcal{U}(-n_{\text{f}}^{-1/2}, n_{\text{f}}^{-1/2}), \quad \boldsymbol{b}^{(1)} \sim \mathcal{U}(-n_{\text{f}}^{-1/2}, n_{\text{f}}^{-1/2}), \tag{60}$$

$$\boldsymbol{W}^{(2)} \sim \mathcal{U}(-n_{\text{h}}^{-1/2}, n_{\text{h}}^{-1/2}), \quad \boldsymbol{b}^{(2)} \sim \mathcal{U}(-n_{\text{h}}^{-1/2}, n_{\text{h}}^{-1/2}), \tag{61}$$

where n_f is the number of input features (which includes $2N+3$ solution features and one source feature) and n_h is the number of hidden neurons. Table 2 shows the network and training parameters. The networks are trained using the Adam optimizer [24] in with weight decay between the layers.

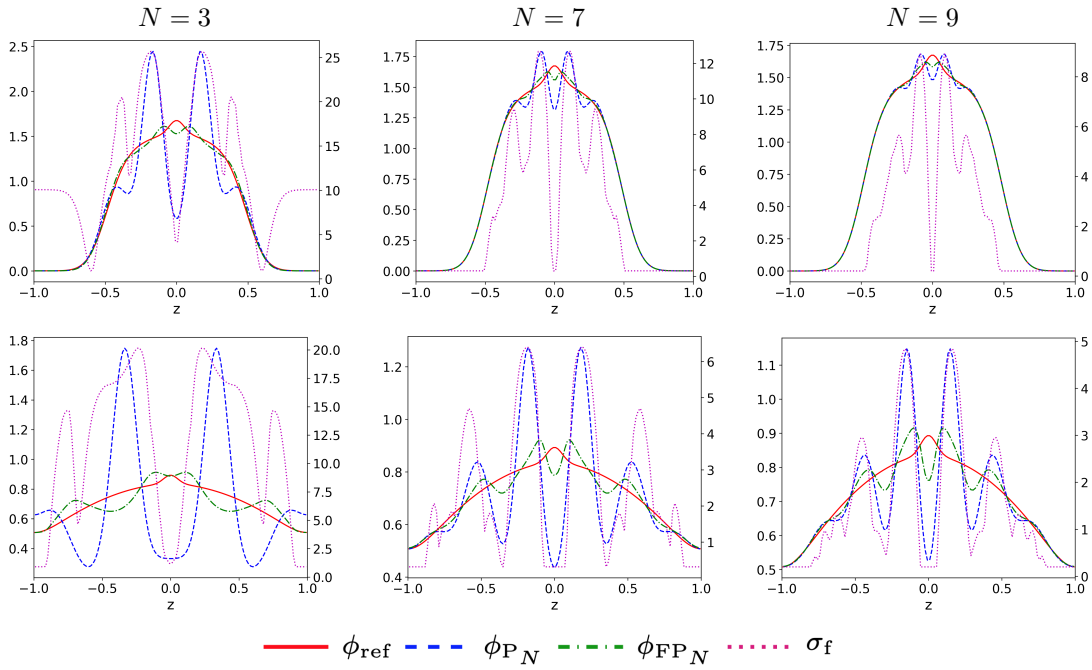
input features (n_f)	hidden layers	hidden width (n_h)	learning rate	batch size	number of epochs	weight decay
2N + 4	1	50	0.1	64	200	1e-5

Table 2: The table shows parameters used in 1-D training. The learning rates are chosen by an initial hyperparameter search.

In each iteration, each of the simulation setups shown in Figure 1 is assigned $\frac{\text{batch size}}{4}$ independently sampled realizations of σ_a and σ_s. For each of these configurations, we compute a high-order P_{127} reference solution and an FP_N solution using the data-driven filter strength. Both FP_N and the reference P_{127} system use grid spacings of $\Delta z = \frac{1}{64}$ and $\Delta t = \frac{1}{128}$ for the space-time discretization. We repeat this process 10 times, producing 10 models for each $N \in \{3, 7, 9\}$. In the following tests, we plot particle concentrations from one representative iteration and report the sample mean of the relative errors r_N^{nn} over all 10 runs.

4.1.3 Test Case: Gaussian

The Gaussian test case features a localized initial condition in a purely scattering medium. The spatial domain is [-1,1] and solutions are run to $t_\mathrm{final} = 0.5$ and $t_\mathrm{final} = 1.0$. The cross-sections are $\sigma_\mathrm{s} = 0.5$ and $\sigma_\mathrm{a} = 0$. The initial condition is a Gaussian of the form in (55), but with $\varsigma = 0.05$. We compare the data-driven FP_N simulation for $N \in \{3, 7, 9\}$ to a P_{127} reference solution. We observe in Figure 2 that the data-driven FP_N equation with the neural network-based filter Σ_f yields a final time solution for $\phi^{\mathrm{nn}}_{\mathrm{FP}_N}(\cdot, t_{t_\mathrm{final}})$ that matches the high-order reference simulation well for $N \in \{3, 7, 9\}$, whereas the low-order P_N simulations exhibit oscillatory behavior. While the constant filter reduces the P_N error significantly in all cases, the neural network-based filter solution achieves greater improvements.



(a) Particle concentrations at $t_{\text{final}} = 0.5$ (top row) and $t_{\text{final}} = 1.0$ (bottom row) for $N = 3, 7, 9$. In each plot, the scale for the particle concentrations is shown on the left and the scale for σ_{f} is shown on the right.

	Neural Network (r_N^{nn})			Constant (r_N^{const})		
t_{final}	$N = 3$	$N = 7$	$N = 9$	$N = 3$	$N = 7$	$N = 9$
0.5	$\mathbf{0.100 \pm 0.006}$	$\mathbf{0.240 \pm 0.001}$	$\mathbf{0.388 \pm 0.002}$	0.592 ± 0.002	0.643 ± 0.002	0.748 ± 0.000
1.0	$\mathbf{0.151 \pm 0.004}$	$\mathbf{0.198 \pm 0.000}$	$\mathbf{0.295 \pm 0.002}$	0.338 ± 0.003	0.357 ± 0.002	0.464 ± 0.000

(b) Ratio of the FP_N relative to the P_N error for the neural network filter strength (r_N^{nn}) and constant filter strength (r_N^{const}). Smaller numbers are better, and bold denotes which trainable filter that does better for each choice of N and t_{final}.

Figure 2: (Gaussian) The neural network ansatz consistently outperforms the constant ansatz in all cases.

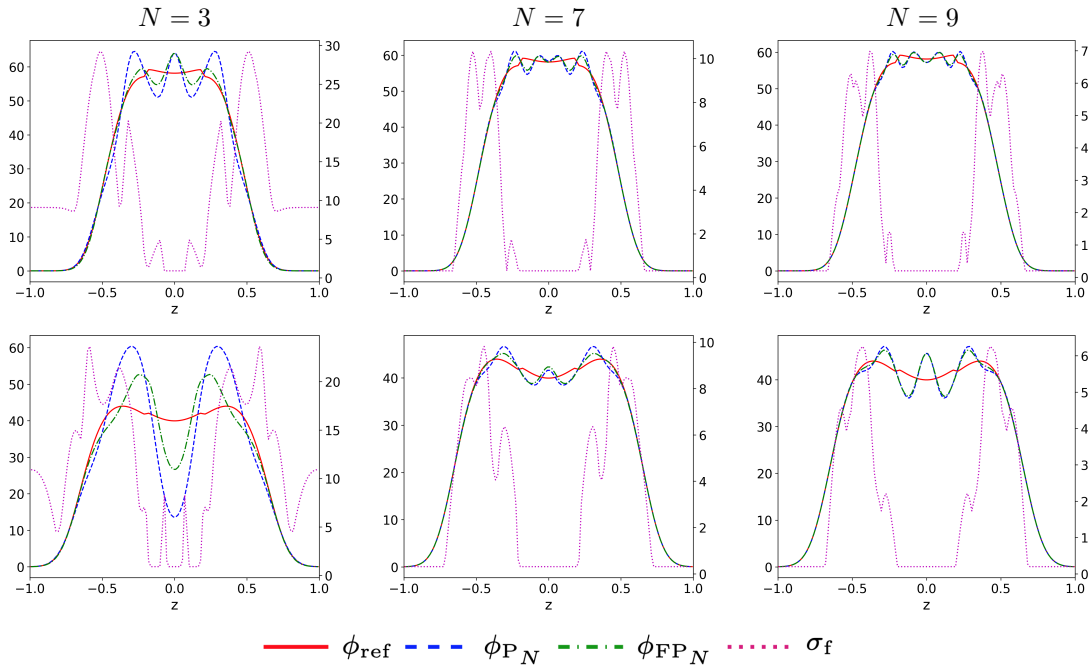
4.1.4 Test Case: Vanishing Cross-Section

The vanishing cross-section problem features non-smooth initial data in a purely scattering medium with smooth, but large variations in σ_{s}. In particular, σ_{s} is close to zero in the center of the spatial domain $[-1, 1]$ and large at the boundaries. The initial data and scattering cross-sections are [8]

$$v_0(z) = \begin{cases} 1, & z \in (-0.2, 0.2), \\ 0, & z \in [-1, 0, -0.2] \cup [0.2, 1.0] \end{cases} \tag{62a}$$

$$\sigma_{\mathrm{s}}(z) = 100 z^4. \tag{62b}$$

We compare the data-driven FP_N simulation for $N \in \{3, 7, 9\}$ with the P_{127} reference solution. The results, which are given in Figure 3, show that the constant filter produces the smallest error in all but one case. While the neural network filter does reduce the error with respect to the P_N solution, the error in the center of the domain, where the σ_{s} is small is still significant.



(a) Particle concentrations at $t_{\text{final}} = 0.5$ (top row) and $t_{\text{final}} = 1.0$ (bottom row) for $N = 3, 7, 9$. The scale for the particle concentration is shown on the left and the scale for σ_{f} is shown on the right in each plot.

	Neural Network (r_N^{nn})			Constant (r_N^{const})		
t_{final}	$N = 3$	$N = 7$	$N = 9$	$N = 3$	$N = 7$	$N = 9$
0.5	$\mathbf{0.437 \pm 0.001}$	0.678 ± 0.000	0.772 ± 0.001	0.646 ± 0.002	$\mathbf{0.649 \pm 0.002}$	$\mathbf{0.728 \pm 0.000}$
1.0	0.497 ± 0.046	0.748 ± 0.000	0.908 ± 0.000	$\mathbf{0.323 \pm 0.003}$	$\mathbf{0.446 \pm 0.002}$	$\mathbf{0.465 \pm 0.001}$

(b) Ratio of the FP_N relative to the P_N error for the neural network filter strength (r_N^{nn}) and constant filter strength (r_N^{const}). Smaller numbers are better, and bold denotes which trainable filter that does better for each choice of N and t_{final}.

Figure 3: (Vanishing Cross-Section) The constant ansatz yields the smaller error in all but one case.

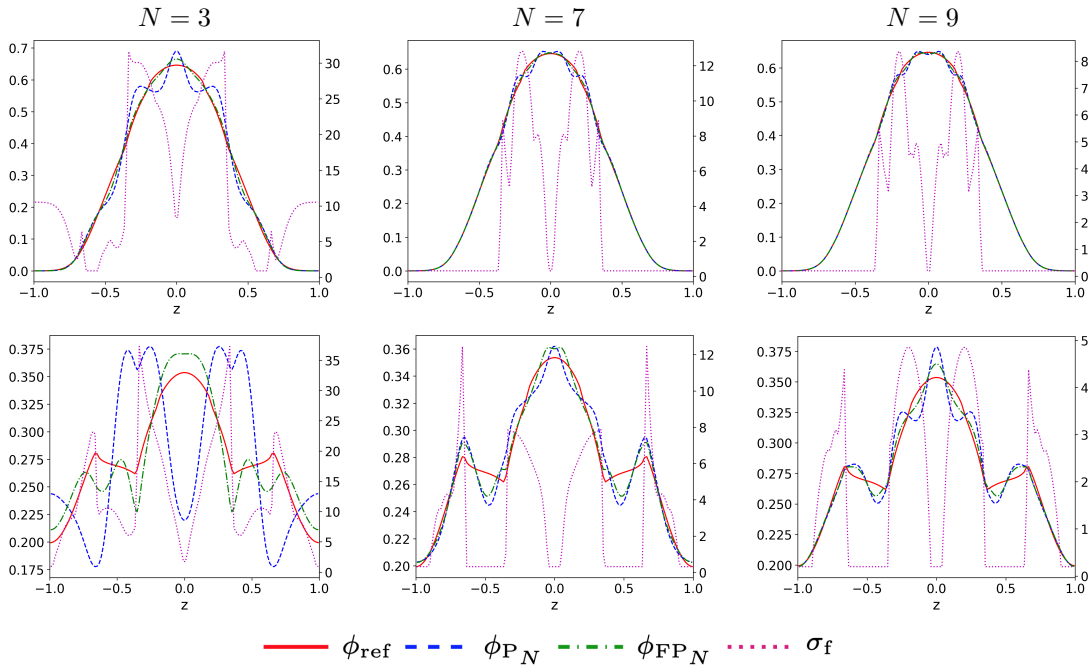
4.1.5 Test Case: Discontinuous Cross-Section

The discontinuous cross-section problem contains sharp discontinuities in σ_{s} and the initial data. This scenario is common in physical models with multiple materials. The problem domain is [-1,1]. The initial data and scattering cross-sections are [17]

$$v_0(z) = \begin{cases} 1, & z \in (-0.2, 0.2), \\ 0, & z \in [-1, -0.2] \cup [0.2, 1.0] \end{cases} \tag{63a}$$

$$\sigma_{\text{s}}(z) = \begin{cases} 0.2, & z \in [-0.65, -0.35] \cup [0.35, 0.65], \\ 1, & z \in [-1, -0.65) \cup (-0.35, 0.35) \cup (0.65, 1]. \end{cases} \tag{63b}$$

We compare the data-driven FP_N simulation for $N \in \{3, 7, 9\}$ with the P_{127} reference solution. We observe in Figure 4 that the low-order P_N simulation yields a final time solution with oscillatory behavior. The neural network-based filter Σ_{f} suppresses these oscillations, improving accuracy. While the constant filter reduces the P_N error significantly over all cases, the neural network-based filter solution achieves greater improvements in all but two cases.



(a) Particle concentrations at $t_{\text{final}} = 0.5$ (top row) and $t_{\text{final}} = 1.0$ (bottom row) for $N = 3, 7, 9$. The scale for the particle concentration ϕ is shown on the left and the scale for σ_{f} is shown on the right in each plot.

	Neural Network (r_N^{nn})			Constant (r_N^{const})		
t_{final}	$N = 3$	$N = 7$	$N = 9$	$N = 3$	$N = 7$	$N = 9$
0.5	$\mathbf{0.309 \pm 0.012}$	$\mathbf{0.325 \pm 0.001}$	$\mathbf{0.492 \pm 0.002}$	0.624 ± 0.001	0.649 ± 0.002	0.729 ± 0.000
1.0	$\mathbf{0.246 \pm 0.021}$	0.633 ± 0.000	0.487 ± 0.001	0.312 ± 0.003	$\mathbf{0.346 \pm 0.002}$	$\mathbf{0.447 \pm 0.001}$

(b) Ratio of the FP_N relative to the P_N error for the neural network filter strength (r_N^{nn}) and constant filter strength (r_N^{const}). Smaller numbers are better, and bold denotes which trainable filter that does better for each choice of N and t_{final}.

Figure 4: (Discontinuous Cross-Section) The neural network ansatz yields the smallest error in all but one case.

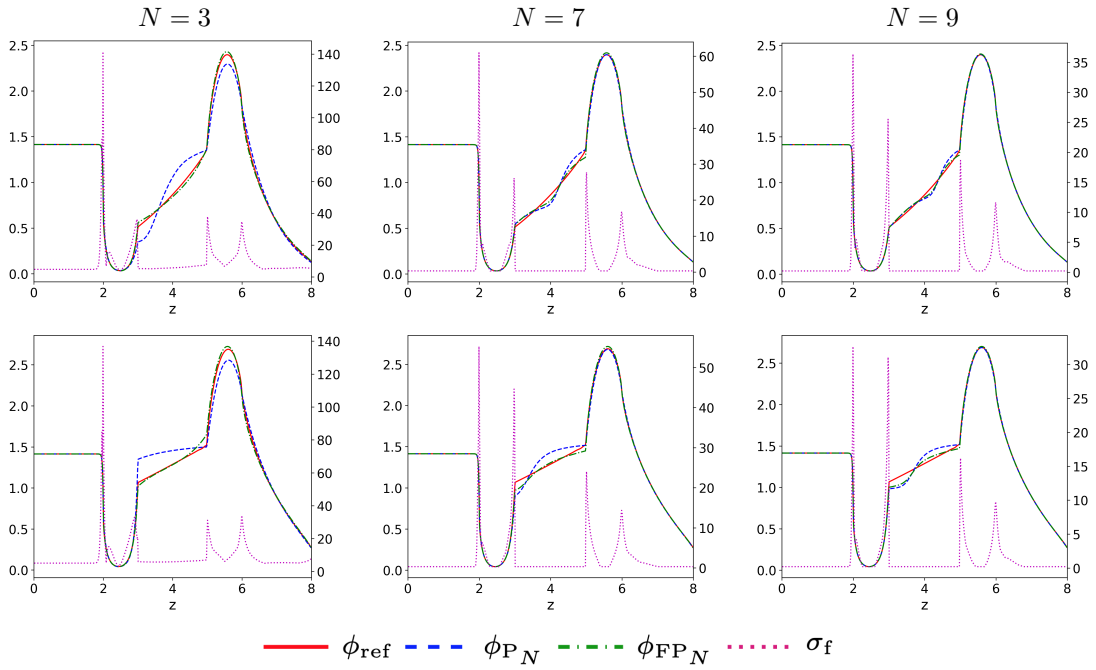
4.1.6 Test Case: Reed's Problem

Reed's problem involves transport through a domain with multiple materials characterized by different cross-sections and sources. The problem layout is described in Table 3. The left boundary condition is reflecting, while the right boundary is vacuum.

	Region 1	Region 2	Region 3	Region 4	Region 5
x	$[0, 2)$	$[2, 3)$	$[3, 5)$	$[5, 6)$	$[6, 8]$
S	50	0	0	1	0
σ_{s}	0	0	0	0.9	0.9
σ_{t}	50	5	0	1	1

Table 3: Spatial configuration for Reed's problem.

Solutions at $t_{\text{final}} = 5$ and $t_{\text{final}} = 10$ are presented in Figure 5. For $N = 3$, the neural network filter yields smaller errors than the constant filter, whereas for $N = 7$ and $N = 9$, the constant filter solutions achieve the



(a) Particle concentrations at $t_{\text{final}} = 5$ (top row) and $t_{\text{final}} = 10$ (bottom row) for $N = 3, 7, 9$. The scale for the particle concentration is shown on the left and the scale for σ_{f} is shown on the right in each plot.

	Neural Network (r_N^{nn})			Constant (r_N^{const})		
t_{final}	$N = 3$	$N = 7$	$N = 9$	$N = 3$	$N = 7$	$N = 9$
5	$\mathbf{0.224 \pm 0.001}$	0.593 ± 0.000	0.625 ± 0.000	0.273 ± 0.000	$\mathbf{0.282 \pm 0.000}$	$\mathbf{0.281 \pm 0.000}$
10	$\mathbf{0.289 \pm 0.030}$	0.438 ± 0.000	0.602 ± 0.000	0.555 ± 0.000	$\mathbf{0.423 \pm 0.001}$	$\mathbf{0.316 \pm 0.000}$

(b) Ratio of the FP_N relative to the P_N error for the neural network filter strength (r_N^{nn}) and constant filter strength (r_N^{const}). Smaller numbers are better, and bold denotes which trainable filter that does better for each choice of N and t_{final}.

Figure 5: (Reed's Problem) The neural network ansatz yields the smallest errors for $N = 3$, while the constant filter strength yields the smallest errors for $N = 7$ and $N = 9$.

smallest errors. As in the vanishing cross-section test, the largest errors occur in the streaming region (Region 3) where the σ_{f} is small.

4.1.7 Ablation Study

We assess the sensitivity of the neural network ansatz through single-feature and leave-one-out ablation tests on Reed's problem with $N = 3$ and $N = 7$ at $t_{\text{final}} = 5$. In the single-feature tests, each feature is tested in isolation by setting all others to zero. In the leave-one-out test, one feature is removed at a time while keeping the rest. The error ratios r_N^{nn} for the particle concentration are reported in Table 4. The data is presented as a mean and standard deviation over ten runs.

For $N = 3$, feature $\sigma_{\text{t}}\mathbf{v}$ gives the smallest error ratio in the single-feature tests and when it is omitted in the leave-one-out tests, the error increases significantly, indicating the importance of this feature in the model. For $N = 7$, feature $\tilde{\mathbf{A}}\partial_x\mathbf{v}$ gives the smallest error ratio in the single-feature tests, while omitting this feature increases the error ratio significantly in the leave-one-out tests.

N	Feature	$\tilde{\mathbf{A}}\partial_x\mathbf{v}$	$\sigma_\mathrm{t}\mathbf{v}$	$\sigma_\mathrm{s}(\tilde{\mathbf{G}}-I)\mathbf{v}$	$\tilde{\mathbf{s}}$	None	All	Interpretation
3	Single-feature	0.374 ± 0.016	0.193 ± 0.005	0.267 ± 0.007	0.270 ± 0.001	0.272 ± 0.001	0.224 ± 0.001	Only this feature included
	Leave-one-out	0.162 ± 0.013	0.290 ± 0.001	0.302 ± 0.014	0.225 ± 0.011	0.224 ± 0.001	–	This feature omitted
7	Single-feature	0.184 ± 0.000	0.419 ± 0.000	0.323 ± 0.000	0.339 ± 0.000	0.334 ± 0.000	0.438 ± 0.000	Only this feature included
	Leave-one-out	0.417 ± 0.000	0.150 ± 0.000	0.438 ± 0.000	0.438 ± 0.000	0.438 ± 0.000	–	This feature omitted

Table 4: Errors from single-feature and leave-one-out tests for Reed's problem with $N = 3$ and $N = 7$ at $t_\mathrm{final} = 5$ are presented. For the single feature tests, the "None" column shows the error with no features (bias only), while "All" includes all features. Feature $\sigma_\mathrm{t}\mathbf{v}$ yields the smallest relative error for $N = 3$ and feature $\tilde{\mathbf{A}}\partial_x\mathbf{v}$ yields the smallest relative error for $N = 7$. In the leave-one-out tests, each column indicates the omitted feature, with "None" referring to the full model. Omitting feature $\tilde{\mathbf{A}}\partial_x\mathbf{v}$ results in the smallest error for $N = 3$ and omitting feature $\sigma_\mathrm{t}\mathbf{v}$ results in the smallest error for $N = 7$.

4.2 2-D Tests

In geometries with no variation in the z direction, the P_N and FP_N equations can be reduced to equations in two spatial variables $(x, y) \in D \subset \mathbb{R}^2$. In this setting, the kinetic distribution $\psi = \psi(x, y, \mathbf{\Omega})$ is approximated by the expansion $\psi_{\mathrm{FP}_N}(x, y, \mathbf{\Omega}) = \tilde{\mathbf{r}}^\top(\mathbf{\Omega})\mathbf{v}(x, y, t)$, where $\tilde{\mathbf{r}} : \mathbb{S}^2 \to \mathbb{R}^{(N+1)(N+2)/2}$ is a vector-valued function containing components of $\mathbf{r}$ with respect to spherical harmonic components that are even with respect to μ and $\mathbf{v}$ satisfies the 2D FP_N equations

$$\partial_t\mathbf{v} + \tilde{\mathbf{A}}^{(1)}\partial_x\mathbf{v} + \tilde{\mathbf{A}}^{(2)}\partial_y\mathbf{v} + \sigma_\mathrm{a}\mathbf{v} + \sigma_\mathrm{s}\tilde{\mathbf{G}}\mathbf{v} + \sigma_\mathrm{f}\tilde{\mathbf{F}}\mathbf{v} = \tilde{\mathbf{s}}, \tag{64}$$

with matrices $\tilde{\mathbf{A}}^{(1)}$, $\tilde{\mathbf{A}}^{(2)}$, $\tilde{\mathbf{G}}$, and $\tilde{\mathbf{F}}$ that are described in Appendix B.2. Because the kinetic distribution ψ in this case is an even function of μ, moments with respect to spherical harmonics that are odd with respect to μ are identically zero and not considered. The physical cross-sections and source $\tilde{\mathbf{s}}$ are all function of x and y, while the filter strength σ_f depends on x, y, and t through the state $\mathbf{v}$ and source $\tilde{\mathbf{s}}$.

Below we describe the data-sampling strategy, architectural details and numerical evaluation of the data driven filter in several 2-D test cases. We test the data-driven FP_N solver for $N \in \{3, 5, 7, 9\}$. In both training and testing scenarios, the initial condition and source are isotropic; that is $v_\ell^m = \tilde{s}_\ell^m = 0$ for all $\ell > 0$; thus we need only specify $v_0^0|_{t=0}$ and $\tilde{s}_0^0$. Moreover all components of v are zero on the boundary (vacuum boundary condition).

4.2.1 Data Sampling Strategy

All simulation setups to generate training data take the domain $D = [-1, 1] \times [-1, 1]$. We specify the random variable γ that constitutes the data sampling strategy to train the data-driven filter for the 2-D test cases. Specifically, γ describes the choice of initial condition for v_0 and sources $\tilde{s}_0$. The initial conditions are selected from

(a) a zero centered Gaussian with standard deviation 0.1:

$$v_0^0(x, y, t = 0) = \frac{1}{\sqrt{2\pi\varsigma^2}} \exp\left(-\frac{x^2 + y^2}{2\varsigma^2}\right), \quad \varsigma = 0.1; \tag{65}$$

(b) a piecewise linear function with maximum at the origin:

$$v_0^0(x, y) = \max\Big(0,\ 1 - |x| - |y|\Big) \tag{66}$$

(c) the indicator function on the interval $[-0.2, 0.2] \times [-0.2, 0.2]$,

(d) the indicator function on the interval $[-0.4, 0.4] \times [-0.4, 0.4]$,

(e) a bump function centered at the origin:

$$v_0^0(x, y) = \begin{cases} 10 \cdot \cos(2\pi r), & \text{if } \cos(2\pi r) > 0 \text{ and } r < \frac{3}{8}, \\ 0, & \text{otherwise} \end{cases} \qquad r = \sqrt{x^2 + y^2}, \tag{67}$$

The sources are selected from

(a) zero

(b) the indicator function on a circle with center $z = 0$ and radius 0.05,

(c) two plates

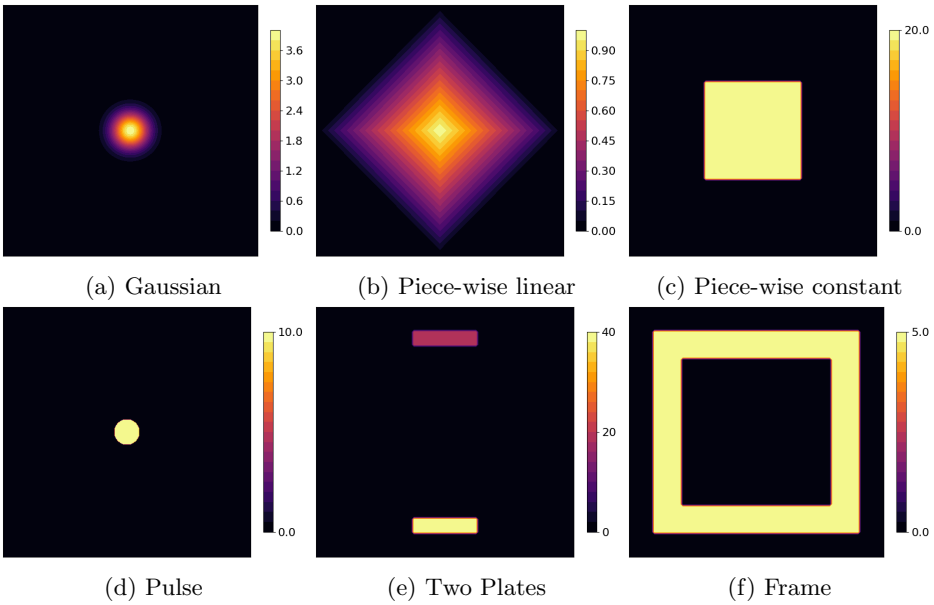
$$v_0^0(x,y) = \begin{cases} 20, & \text{if } x \in [-0.25, 0.25] \text{ and } y \in [0.7, 0.8], \\ 40, & \text{if } x \in [-0.25, 0.25] \text{ and } y \in [-0.8, -0.7], \\ 0, & \text{otherwise,} \end{cases} \tag{68}$$

(d) the indicator function of a frame with width 0.1:

$$v_0^0(x,y) = \begin{cases} 5, & \text{if } \big(|x| \in [0.6, 0.8] \text{ and } |y| \leq 0.8\big) \text{ or } \big(|y| \in [0.6, 0.8] \text{ and } |x| \leq 0.8\big), \\ 0, & \text{otherwise,} \end{cases} \tag{69}$$

(e) a zero centered Gaussian (65) with standard deviation 0.03.

Several of the initial conditions and sources are shown in Figure 6. All setups are simulated until final time $t_{\text{final}} = 0.5$.



(a) Gaussian (b) Piece-wise linear (c) Piece-wise constant

(d) Pulse (e) Two Plates (f) Frame

Figure 6: Figures (a) - (c) show initial conditions for training data. While the Gaussian is smooth, steep gradients are challenging to resolve with the P_N method. The irregularities in piece-wise linear and piece-wise constant functions bring their own challenges. The other initial conditions that are not shown are a bump function and a narrow piece-wise constant function. Figures (d) - (f) show isotropic sources used to train the model. The other training sources that are not shown are a Gaussian and a zero function.

Furthermore, γ describes the values for the scattering and absorption cross section, i.e. σ_{s} and σ_{a} which are distributed as follows

$$\sigma_{\mathrm{s}} \sim \mathcal{U}(0,1), \quad \sigma_{\mathrm{a}} \sim \mathcal{U}(0, 1 - \sigma_{\mathrm{s}}),$$

where $\mathcal{U}$ denotes the uniform distribution in an interval.

4.2.2 Neural Network Architecture and Training

We train filters Σ_{f} for $N \in \{3, 5, 7, 9\}$, where the preprocessed features are put into a feedforward neural network. The network is fully connected with tanh activations. Residual connections and LayerNorm (see the PyTorch documentation[33]) are applied after each hidden layer to stabilize training. A final ReLU-activated output layer maps to a single scalar per input sample. The first layer is equipped with a batch-normalization layer [22] that estimates mean and variance of the features across the grid and batch. The weights and biases of the hidden and output layers, $\boldsymbol{W}^{(i)}$ and $\boldsymbol{b}^{(i)}$, are initialized from uniform distributions. For the first layer, the initialization depends on the number of input features n_{f}:

$$\boldsymbol{W}^{(1)}, \boldsymbol{b}^{(1)} \sim \mathcal{U}\Big(-n_{\mathrm{f}}^{-1/2}, n_{\mathrm{f}}^{-1/2}\Big). \tag{70}$$

For subsequent layers $i \in \{2, \ldots, 5\}$, the initialization is based on the number of hidden neurons n_{h_i} in layer i:

$$\boldsymbol{W}^{(i)}, \boldsymbol{b}^{(i)} \sim \mathcal{U}\Big(-n_{\mathrm{h}_i}^{-1/2}, n_{\mathrm{h}_i}^{-1/2}\Big). \tag{71}$$

Table 5 shows the network and training parameters. The networks are trained using the Adam optimizer [24] in PyTorch.

input features (n_{f})	hidden layers	hidden width (n_{h_i})	learning rate	batch size	number of epochs	weight decay
2N + 4	4	N+2	1e-2	5	200	0

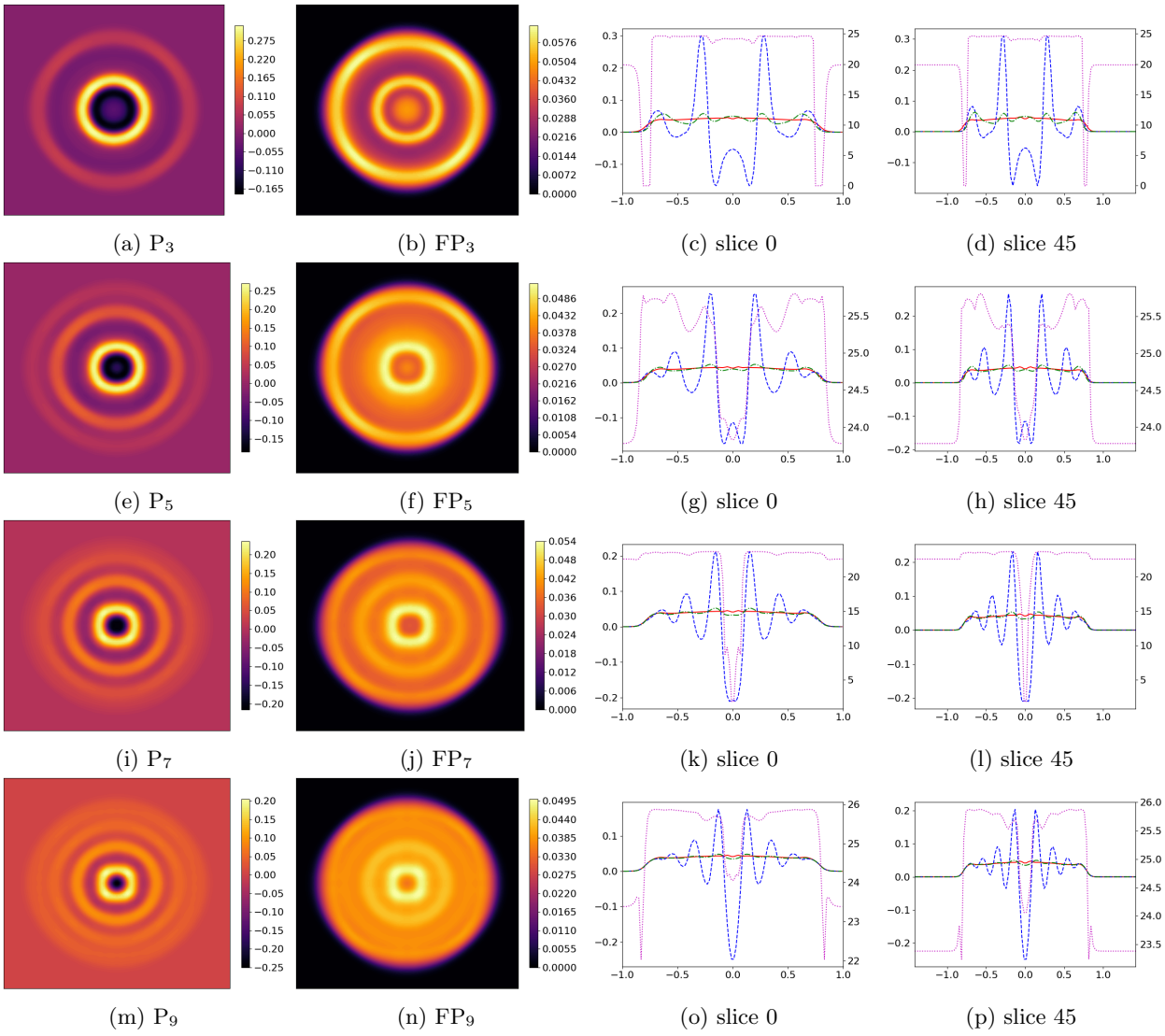
Table 5: The table shows parameters used in 2-D training. The learning rate is chosen by an initial hyperparameter search.

In each iteration of the training process, we draw five realizations of γ, uniformly sampling and pairing initial conditions and sources described in Section 4.2.1. Each pairing is assigned independently sampled realizations of σ_{a} and σ_{s}. Both FP_N and the reference P_{37} system use grid spacings of $\Delta x = \frac{1}{50}$ and $\Delta t = \frac{1}{100}$ for the space-time discretization.

4.2.3 Test Case: Line-source

The line-source problem [7] presents a challenging benchmark due to the presence of steep gradients in the scalar flux and the emergence of large high-order moments in the angular flux. Physically, the initial condition corresponds to a highly localized source at the center of the domain, which is typically modeled as a Dirac delta function. In our setup, we approximate this source using a smooth Gaussian (65) centered at $(x, y) = (0, 0)$ with standard deviation $\varsigma = 0.03$.

We consider a sequence of polynomial orders $N \in \{3, 5, 7, 9\}$ and compare the classical P_N approximation with the FP_N approximation. Representative solution profiles of the particle density for different values of N are shown in Figure 7. The FP_N method exhibits significantly reduced oscillations in the scalar flux, particularly near the source, where Gibbs-like artifacts dominate in the P_N approximation. For smaller N, the neural network filter performs marginally better than the constant filter. However, for moderate N, the benefit of the neural network filter over the constant is significant.



(a) P_3 (b) FP_3 (c) slice 0 (d) slice 45

(e) P_5 (f) FP_5 (g) slice 0 (h) slice 45

(i) P_7 (j) FP_7 (k) slice 0 (l) slice 45

(m) P_9 (n) FP_9 (o) slice 0 (p) slice 45

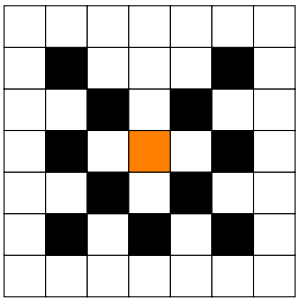
ϕ_{ref} ϕ_{P_N} ϕ_{FP_N} σ_{f}

N	Plot Ref	e_N	e_N^{nn}	e_N^{const}	r_N^{nn}	r_N^{const}
3	(a) - (d)	2.3141	**0.3276**	0.3859	**0.1416**	0.1668
5	(e) - (h)	1.7282	**0.1333**	0.1389	**0.0771**	0.0804
7	(i) - (l)	1.3168	**0.0822**	0.1599	**0.0624**	0.1214
9	(m) - (p)	0.9664	**0.0408**	0.1632	**0.0422**	0.1688

Figure 7: (Line-Source) The figure shows particle concentrations for P_N and FP_N at $t_{\mathrm{final}} = 0.75$. The first row shows the $N = 3$ profiles and increases sequentially to $N = 9$ in the bottom row. From left to right, the plots show P_N, FP_N, a slice at $y = 0$ and a slice at 45 degrees. In the plots of the 1-D slices, the scale for particle concentration is shown on the left axis and the scale for σ_{f} is shown on the right axis. The horizontal axis measures the signed distance from the origin. As N increases, the error of the FP_N solution decreases and the oscillations associated with the P_N approximation are significantly dampened. The values shown in bold in the table represent the smallest errors between the neural network and constant filter solutions.

4.2.4 Test Case: Lattice

The lattice problem [7] poses a challenging benchmark due to the presence of sharp discontinuities in the material cross-sections. The layout of the material and a P_{37} reference solution are given in Figure 8. Numerical simulations are performed using a uniform spatial resolution of $\Delta x = \Delta y = \frac{1}{40}$ and a temporal step size of $\Delta t = \frac{1}{160}$.



Region	σ_t	σ_s	S
white	1	1	0
orange	10	0	1
black	10	0	0

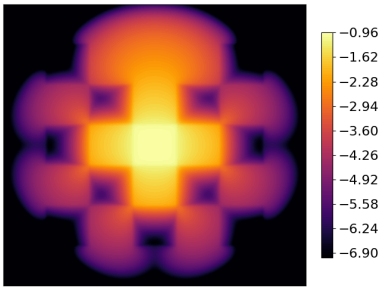
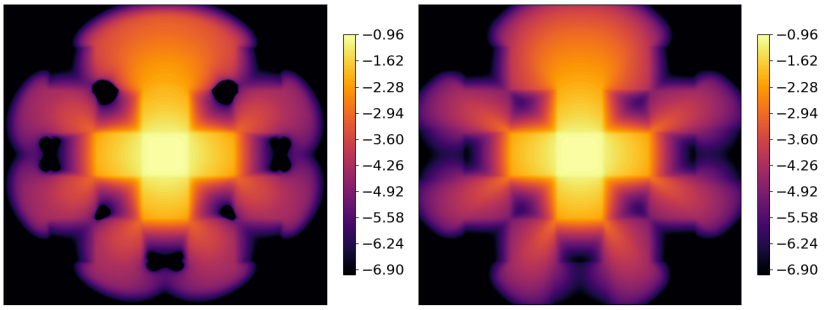


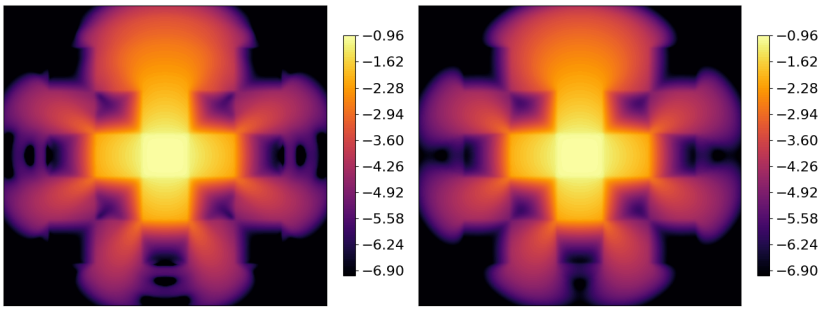
Figure 8: (Lattice) The left plot shows the geometric layout of the lattice problem. The spatial domain is $[0,7]\times[0,7]$ and each square has an area of 1. The intensity is initially zero everywhere and radiation is introduced by an isotropic source in the orange region. The orange and black regions are purely absorbing, while the white regions are purely scattering. The boundary conditions are vacuum. The values of the material cross-sections and isotropic source are shown in the center table. The right figure shows the P_{37} solution at $t_{\text{final}} = 3.2$ with $\log_{10}$ scaling.

Representative solution profiles for both the P_N and FP_N methods at $t_{\text{final}} = 3.2$ are shown in Figure 9. The profiles show the improved resolution of the filtered solutions in regions with steep gradients and material heterogeneities. The behavior of the learned filter strength is further illustrated in Figure 10, which shows the filter strength for $N = 5$ at both $t_{\text{final}} = 1.6$ and $t_{\text{final}} = 3.2$. The filter accurately identifies regions of high anisotropy and sharp features, thereby enabling more accurate and efficient propagation of radiative transport features throughout the domain.

Table 6 reports the L^2 errors and relative errors at times $t_{\text{final}} = 1.6$ and $t_{\text{final}} = 3.2$ for varying angular resolutions. The boldface entries highlight the more accurate result between the neural-network-based filter and the constant filter baseline. Both filter provide substantial accuracy gains, with the neural network performing slightly better in each case.



(a) P_3 (b) FP_3



(c) P_9 (d) FP_9

Figure 9: (Lattice) Particle concentrations at $t_{\text{final}} = 3.2$ for the P_N and FP_N methods are shown. The corresponding errors with respect to the P_{37} solution are reported in Table 6

.

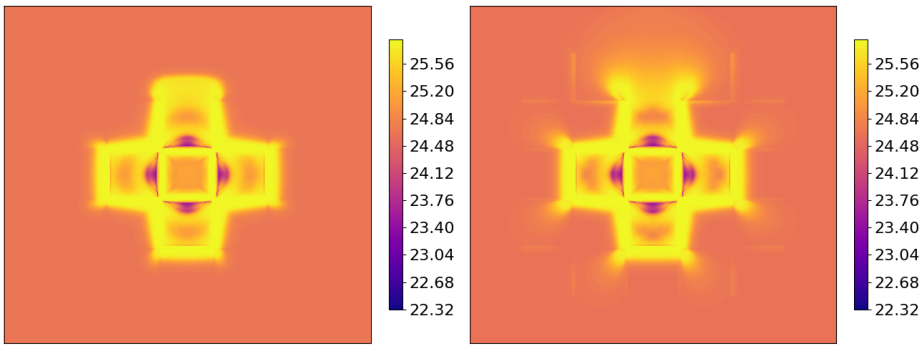


Figure 10: (Lattice) The figure shows the filter strength for $N = 9$ at $t = 1.6$ and $t = 3.2$. The filter detects the geometric features of the test case domain where the solution has large gradients.

		$t_{\text{final}} = 1.6$					$t_{\text{final}} = 3.2$				
N	Plot Ref	e_N	e_N^{nn}	e_N^{const}	r_N^{nn}	r_N^{const}	e_N	e_N^{nn}	e_N^{const}	r_N^{nn}	r_N^{const}
3	(a) - (b)	0.0426	**0.0126**	0.0146	**0.2959**	0.3435	0.0274	**0.0127**	0.0129	**0.4642**	0.4695
5	-	0.0225	**0.0063**	0.0066	**0.2792**	0.2905	0.0156	**0.0065**	0.0068	**0.4182**	0.4358
7	-	0.0140	**0.0038**	0.0042	**0.2728**	0.3021	0.0102	**0.0040**	0.0042	**0.3913**	0.4096
9	(c) - (d)	0.0097	**0.0029**	0.0031	**0.2953**	0.3160	0.0072	**0.0029**	0.0029	**0.4090**	0.4103

Table 6: (Lattice) Errors and relative errors for the lattice problem at $t = 1.6$ and $t = 3.2$. Bold values indicate the smaller error between neural network and constant filters. Plot references for $N = 3$ and $N = 9$ are shown in the second column; solution profiles are given in Figure 9.

5 Conclusion

In this work, we establish a strategy for adaptively filtering in FP_N based on the local state of the system. We model the filter strength as a neural network that depends on features that are terms in the original P_N equations. These features are pre-processed to ensure rotational invariance.

We train 1-D and 2-D models for several values of N and compare the neural network based filtered solutions with those obtained using a constant filter strength that is trained with the same data. The resulting low-resolution FP_N solutions yield significant improvements in quality at a comparable cost to the original P_N approximation. The training data can be generated from simple initial conditions and cross-section configurations. Naturally, the quality of the training data is crucial; the corresponding P_N solutions should be representative of typical behaviors associated with irregular solutions. We train the model with problems evolved until final time $t_{\text{final}} = 0.5$, which provides a reasonable balance between computational cost and sufficient training.

Across all test cases, filtering reduces the error with respect to the original P_N approximation, regardless of whether the filter strength is a constant or determined by a neural network ansatz. In 1-D tests, this reduction can vary from $10 - 90\%$. The neural network filter outperforms the constant filter in several cases, but under-performs in others. We observe that the constant filter outperforms the neural network filter for problems with streaming regimes (when the material is vacuum). In 2-D tests, the error reduction can vary from $70 - 96\%$. The neural network filter reduces the error more than the constant filter in all test cases and in some cases, the error reduction is significant.

This strategy is flexible and can accommodate various forms of the filter strength ansatz, including a constant form, which offers the simplest and often most practical implementation. However, the constant form is limited by construction, as it does not adapt with the solution over time and space. This limitation necessitates tuning the filter based on the known (or estimated) regularity of the solution and the final time. For short-time simulations, a large filter strength may improve the accuracy of the approximation significantly, but that same filter strength applied to a long-time simulation over-smooths the solution and accuracy may be lost. In contrast, the neural network ansatz does not have this limitation, since the filter strength evolves dynamically with the state.

In future work, it would be beneficial to better understand and address the shortcomings of the neural network models in 1-D test cases, which under-perform the constant filter models in several cases. In application spaces, we believe that improvements could be made with a dedicated set of training problems and hyperparameter optimization strategies. In addition, we conjecture that filtering could include extended to other physically significant problems like thermal radiative transfer [34].

References

[1] Milton Abramowitz and Irene A. Stegun. *Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables*. Vol. 55. Applied Mathematics Series. New York: Dover Publications, 1972.

[2] Graham W Alldredge, Martin Frank, and Cory D Hauck. "A regularized entropy-based moment method for kinetic equations". In: *SIAM Journal on Applied Mathematics* 79.5 (2019), pp. 1627–1653.

[3] K. Atkinson and W. Han. *Spherical Harmonics and Approximations on the Unit Sphere: An Introduction*. Lecture Notes in Mathematics. Springer, 2012.

[4] George I Bell and Samuel Glasstone. *Nuclear reactor theory*. Tech. rep. US Atomic Energy Commission, Washington, DC (United States), 1970.

[5] Miguel A Blanco et al. "Evaluation of the rotation matrices in the basis of real spherical harmonics". In: *Journal of Molecular Structure: THEOCHEM* 419 (1997), pp. 19–27.

[6] Thomas A Brunner and James Paul Holloway. "Two-dimensional time dependent Riemann solvers for neutron transport". In: *Journal of Computational Physics* 210.1 (2005), pp. 386–399.

[7] Thomas A. Brunner. *Forms of Approximate Radiation Transport*. Tech. rep. United States, 2002, p. 43.

[8] Christophe Buet and Stephane Cordier. "Asymptotic preserving scheme and numerical methods for radiative hydrodynamic models". In: *Comptes Rendus Mathematique* 338.12 (2004), pp. 951–956.

[9] Kenneth M Case and Paul F. Zweifel. *Linear transport theory*. Addison-Wesley, 1967.

[10] Cheol Ho Choi et al. "Rapid and stable determination of rotation matrices between spherical harmonics by direct recursion". In: *The Journal of Chemical Physics* 111.19 (1999), pp. 8825–8831.

[11] Feng Dai. *Approximation theory and harmonic analysis on spheres and balls*. Springer, 2013.

[12] R. Dautray and J.-L. Lions. *Mathematical Analysis and Numerical Methods for Science and Technology: Volume 6 Evolution Problems 2*. Berlin, Heidelberg: Springer-Verlag, 2000.

[13] M. Frank, C. Hauck, and Kerstin Küpper. "Convergence of filtered spherical harmonic equations for radiation transport". In: *Communications in Mathematical Sciences* 14 (2016), pp. 1443–1465.

[14] C Kristopher Garrett and Cory D Hauck. "On the eigenstructure of spherical harmonic equations for radiative transport". In: *Computers & Mathematics with Applications* 72.2 (2016), pp. 264–270.

[15] Robin Green. "Spherical harmonic lighting: The gritty details". In: *Game Developers Conference* (2003).

[16] Per Hansen and Dianne O'leary. "The Use of the L-Curve in the Regularization of Discrete Ill-Posed Problems". In: *SIAM J. Sci. Comput.* 14 (Nov. 1993), pp. 1487–1503.

[17] Cory Hauck and Robert Lowrie. "Temporal regularization of the PN equations". In: *Multiscale Modeling and Simulation* 7 (Sept. 2008).

[18] Cory Hauck and Ryan McClarren. "Positive P_N closures". In: *SIAM Journal on Scientific Computing* 32.5 (2010), pp. 2603–2626.

[19] Jan S. Hesthaven, Sigal Gottlieb, and David Gottlieb. *Spectral Methods for Time-Dependent Problems*. Cambridge Monographs on Applied and Computational Mathematics. Cambridge University Press, 2007.

[20] M. Hinze et al. *Optimization with PDE Constraints*. Mathematical Modelling: Theory and Applications. Springer Netherlands, 2008.

[21] John R Howell et al. *Thermal radiation heat transfer*. CRC press, 2020.

[22] Sergey Ioffe and Christian Szegedy. *Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift*. 2015. arXiv: `1502.03167 [cs.LG]`.

[23] Joseph Ivanic and Klaus Ruedenberg. "Rotation Matrices for Real Spherical Harmonics, Direct Determination by Recursion". In: *The Journal of Physical Chemistry A* 100 (1996), pp. 6342–6347.

[24] Diederik P. Kingma and Jimmy Ba. *Adam: A Method for Stochastic Optimization*. 2017. arXiv: `1412.6980 [cs.LG]`.

[25] Vincent M Laboure, Ryan G McClarren, and Cory D Hauck. "Implicit filtered PN for high-energy density thermal radiation transport using discontinuous Galerkin finite elements". In: *Journal of Computational Physics* 321 (2016), pp. 624–643.

[26] M Paul Laiu and Cory D Hauck. "Positivity limiters for filtered spectral approximations of linear kinetic transport equations". In: *Journal of Scientific Computing* 78 (2019), pp. 918–950.

[27] M Paul Laiu et al. "Positive filtered p _n moment closures for linear kinetic equations". In: *SIAM Journal on Numerical Analysis* 54.6 (2016), pp. 3214–3238.

[28] Randall J. LeVeque. *Finite Volume Methods for Hyperbolic Problems*. Cambridge Texts in Applied Mathematics. Cambridge: Cambridge University Press, 2002.

[29] Elmer Eugene Lewis and Warren F Miller. "Computational methods of neutron transport". In: (1984).

[30] Ryan G. McClarren and Cory D. Hauck. "Robust and accurate filtered spherical harmonics expansions for radiative transfer". In: *Journal of Computational Physics* 229.16 (2010), pp. 5597–5614.

[31] Anthony Mezzacappa et al. "Physical, numerical, and computational challenges of modeling neutrino transport in core-collapse supernovae". In: *Living Reviews in Computational Astrophysics* 6 (2020), pp. 1–174.

[32] Dimitri Mihalas and BW Mihalas. "Radiation hydrodynamics". In: *Computational methods for astrophysical fluid flow. Saas-Fee advanced course* 27 (1984), pp. 161–261.

[33] Adam Paszke et al. "Automatic differentiation in PyTorch". In: (2017).

[34] Benjamin Plumridge. “Filtered Angular Discretizations in Radiative Transfer”. PhD Thesis. University of Tennessee, Knoxville, 2025.

[35] Gerald C Pomraning. *The equations of radiation hydrodynamics*. Courier Corporation, 2005.

[36] David Radice et al. “A new spherical harmonics scheme for multi-dimensional radiation transport I. Static matter configurations”. In: *Journal of Computational Physics* 242 (2013), pp. 648–669.

[37] Endre Süli and David F. Mayers. *An Introduction to Numerical Analysis*. Cambridge University Press, 2003.

A Flux matrices in the spherical harmonic equations

The complex spherical harmonics satisfy the recursion relation [6]

$$\mathbf{\Omega}\overline{Y}_\ell^m = \frac{1}{2}i\begin{bmatrix} -c_{\ell-1}^{m-1}\overline{Y}_{\ell-1}^{m-1} + d_{\ell+1}^{m-1}\overline{Y}_{\ell+1}^{m-1} + e_{\ell-1}^{m+1}\overline{Y}_{\ell-1}^{m+1} - f_{\ell+1}^{m+1}\overline{Y}_{\ell+1}^{m+1} \\ i\left(c_{\ell-1}^{m-1}Y_{\ell-1}^{m-1} - d_{\ell+1}^{m-1}Y_{\ell+1}^{m-1} + e_{\ell-1}^{m+1}Y_{\ell-1}^{m+1} - f_{\ell+1}^{m+1}Y_{\ell+1}^{m+1}\right) \\ 2(a_{\ell-1}^m\overline{Y}_{\ell-1}^m + r_{\ell+1}^m\overline{Y}_{\ell+1}^m) \end{bmatrix} \tag{72}$$

where

$$a_\ell^m = \sqrt{\frac{(\ell-m+1)(\ell+m+1)}{(2\ell+3)(2\ell+1)}}, \quad b_\ell^m = \sqrt{\frac{(\ell-m)(\ell+m)}{(2\ell+1)(2\ell-1)}}, \qquad c_\ell^m = \sqrt{\frac{(\ell+m+1)(\ell+m+2)}{(2\ell+3)(2\ell+1)}}, \tag{73}$$

$$d_\ell^m = \sqrt{\frac{(\ell-m)(\ell-m-1)}{(2\ell+1)(2\ell-1)}}, \qquad e_\ell^m = \sqrt{\frac{(\ell-m+1)(\ell-m+2)}{(2\ell+3)(2\ell+1)}}, \quad f_\ell^m = \sqrt{\frac{(\ell+m)(\ell+m-1)}{(2\ell+1)(2\ell-1)}}. \tag{74}$$

When expressed in terms of the real spherical harmonics, the recursion relation (72) becomes [13]

$$\mathbf{\Omega}R_\ell^m = \frac{1}{2}\mathrm{sgn}(m)\begin{bmatrix} (1-\delta_{\ell,1})(\tilde{c}_{\ell-1}^{|m|-1}R_{\ell-1}^{|m|-1} - \tilde{d}_{\ell-1}^{|m|-1}R_{\ell-1}^{|m|-1}) - \tilde{e}_{\ell-1}^{|m|+1}R_{\ell-1}^{|m|+1} + \tilde{f}_{\ell-1}^{|m|+1}R_{\ell-1}^{|m|+1} \\ (1-\delta_{\ell,1})(-\tilde{c}_{\ell-1}^{|m|-1}R_{\ell-1}^{-|m|-1} + \tilde{d}_{\ell-1}^{|m|-1}R_{\ell-1}^{-|m|-1}) - \tilde{e}_{\ell-1}^{|m|+1}R_{\ell-1}^{-|m|-1} + \tilde{f}_{\ell-1}^{|m|+1}R_{\ell-1}^{-|m|+1} \\ 2(\tilde{a}_{\ell-1}^m R_{\ell-1}^m + \tilde{b}_{\ell+1}^m R_{\ell+1}^m) \end{bmatrix} \tag{75}$$

where $\delta_{i,j}$ denotes the Kronecker delta, and $\mathrm{sgn}(m)$ denotes the sign function (with abuse of notation in zero: $\mathrm{sgn}(0) \equiv 1$). The coefficients are given by

$$m^+ = m + \mathrm{sgn}(m), \qquad m^- = m - \mathrm{sgn}(m) \tag{76}$$

$$\tilde{c}_\ell^m = \begin{cases} 0, & m < 0 \\ \sqrt{2}c_\ell^m, & m = 0 \\ c_\ell^m, & m > 0 \end{cases}, \qquad \tilde{d}_\ell^m = \begin{cases} 0, & m < 0 \\ \sqrt{2}d_\ell^m, & m = 0 \\ d_\ell^m, & m > 0 \end{cases} \tag{77}$$

$$\tilde{e}_\ell^m = \begin{cases} \sqrt{2}e_\ell^m, & m = 1 \\ e_\ell^m, & m > 1 \end{cases}, \qquad \tilde{f}_\ell^m = \begin{cases} \sqrt{2}f_\ell^m, & m = 1 \\ f_\ell^m, & m > 1 \end{cases} \tag{78}$$

The recursion in (75) enables a direct evaluation of the tensor $\mathbf{A} = \langle \mathbf{\Omega}\mathbf{r}\mathbf{r} \rangle^\top$, which is simplified by the normalization condition $\langle R_\ell^m R_{\ell'}^{m'} \rangle = \delta_{ell,\ell'}\delta_{m,m'}$

B Reduction to one and two dimensions

B.1 Reduction to 1-D

In a slab geometry the spatial domain is the region between two infinite parallel plates $\{z = z_\mathrm{L}\}$ and $\{z = z_\mathrm{R}\}$; that is, $X = \mathbb{R}^2 \times (z_L, z_R)$. With appropriate initial and boundary conditions, ψ depends only on z and the angular variable $\mu = \cos\theta$.

As a result, the moments u_ℓ^m are identically zero for $m \neq 0$. In this setting we use moments with respect to the Legendre polynomials normalized on the interval $[-1, 1]$: [4]

$$\tilde{P}_\ell(\mu) = \sqrt{2\pi}r_\ell^0(\mu). \tag{79}$$

Let $\tilde{\mathbf{p}} : [-1, 1] \to \mathbb{R}^{N+1}$ be a vector-valued function of all orthonormal Legendre polynomials up to degree N such that $\tilde{\mathbf{p}}(\mu) = [\tilde{P}_0(\mu), \tilde{P}_1(\mu), \ldots, \tilde{P}_N(\mu)]^\top$. The FP_N solution in 1-D slab geometry is described by $\psi_{\mathrm{FP}_N}(z, \mu) = \tilde{\mathbf{p}}^\top(\mu)\mathbf{v}(z, t)$ where

[4] This normalization differs from the standard normalization in which $\int_{-1}^1 |P_\ell(\mu)|^2 d\mu = \frac{2\ell+1}{2}$.

$$\partial_t \mathbf{v} + \tilde{\mathbf{A}}\partial_z \mathbf{v} + \sigma_{\mathrm{a}}\mathbf{v} + \sigma_{\mathrm{s}}\tilde{\mathbf{G}}\mathbf{v} + \sigma_{\mathrm{f}}\tilde{\mathbf{F}}\mathbf{v} = \tilde{\mathbf{s}}, \tag{80}$$

and $\tilde{\mathbf{s}}$ is a 1-D source. The FP_N matrices are given by

$$\tilde{\mathbf{A}}_{\ell,\ell'} = a_\ell\,\delta_{\ell',\ell+1} + a_{\ell'}\,\delta_{\ell,\ell'+1} \qquad \text{where} \qquad a_\ell = \frac{\ell+1}{\sqrt{(2\ell+1)(2\ell+3)}}, \tag{81}$$

$$\tilde{\mathbf{G}}_{\ell,\ell'} = (1-g_\ell)\delta_{\ell,\ell'}, \qquad \text{and} \qquad \tilde{\mathbf{F}}_{\ell,\ell'}, = -\log\left(f\left(\frac{\ell}{N+1}\right)\right)\delta_{\ell,\ell'}. \tag{82}$$

B.2 Reduction to 2-D

In two-dimensional planar geometry, $X = D \times \mathbb{R}$, where $D \subset \mathbb{R}^2$. Under appropriate boundary and initial conditions, ψ is independent of z and an even function of μ. As a result, the fact that r_ℓ^m is an odd function of μ whenever $\ell + m$ is odd implies that moments the u_ℓ^m are identically zero whenever $\ell + m$ is odd. In such cases, the P_N equations take the form

$$\partial_t \mathbf{v} + \tilde{\mathbf{A}}^{(1)}\partial_x \mathbf{v} + \tilde{\mathbf{A}}^{(2)}\partial_y \mathbf{v} + \sigma_{\mathrm{a}}\mathbf{v} + \sigma_{\mathrm{s}}\tilde{\mathbf{G}}\mathbf{v} + \sigma_{\mathrm{f}}\tilde{\mathbf{F}}\mathbf{v} = \tilde{\mathbf{s}} \tag{83}$$

Here $\mathbf{v}$ contains the $n_2 = \frac{1}{2}(N+1)(N+2)$ nonzero components of $\mathbf{u}_{\mathrm{P}_N}$, $\tilde{\mathbf{s}}$ contains the corresponding components of $\mathbf{s}$ and the matrices $\tilde{\mathbf{A}}^{(1)}$, $\tilde{\mathbf{A}}^{(2)}$, $\tilde{\mathbf{G}}$, and $\tilde{\mathbf{F}}$, are formed by removing the appropriate rows and columns of $\mathbf{A}^{(1)}$, $\mathbf{A}^{(2)}$, $\mathbf{G}$, and $\mathbf{F}$, respectively.

B.3 Space time discretization

Now we discuss the spatial and temporal discretizations. We show the spatial discretization used in our 2-D tests. The 1-D discretization follows similarly. For ease of notation, we write $\mathbf{v} := \mathbf{u}_{\mathrm{FP}_N}$.

For simplicity, we assume that D is rectangle with side lengths L_x and L_y and is divided into $N_x \times N_y$ cells with centers $\boldsymbol{x}_{i,j} := (x_i, y_j)$ and dimension $\Delta x = L/N_x$ and $\Delta y = L/N_y$. Let $C_{i,j} = (x_{i-1/2}, x_{i+1/2}) \times (y_{j-1/2}, y_{j+1/2})$ be the cell centered at $\boldsymbol{x}_{i,j}$ where $i \in \{0, \ldots, N_x - 1\}$ and $j \in \{0, \ldots, N_y - 1\}$ are spatial indices. The physical cross-sections, source, and filter strength are assumed to be constant on each cell.

Since the matrices $\tilde{\mathbf{A}}^{(m)}$ are symmetric and diagonalizable, they can be decomposed as

$$\tilde{\mathbf{A}}^{(m)} = \mathbf{V}^{(m)}\boldsymbol{\Lambda}^{(m)}(\mathbf{V}^{(m)})^\top$$

where $\boldsymbol{\Lambda}^{(m)} \in \mathbb{R}^{n_2 \times n_2}$ is diagonal and $\mathbf{V}^{(m)} \in \mathbb{R}^{n_2 \times n_2}$ is orthogonal. With this decomposition, let

$$|\tilde{\mathbf{A}}^{(m)}| = \mathbf{V}^{(m)}|\boldsymbol{\Lambda}^{(m)}|(\mathbf{V}^{(m)})^\top.$$

Then the semi-discrete, second-order finite volume scheme [28] can be described as follows: Find

$$\mathbf{v}_{i,j}(t) \approx \frac{1}{\Delta x \Delta y}\int_{C_{i,j}} \mathbf{v}(x,y,t)dxdy \tag{84}$$

such that

$$\begin{aligned}
&\partial_t \mathbf{v}_{i,j} + \frac{1}{2\Delta x}\tilde{\mathbf{A}}^{(1)}\left[(\mathbf{v}^-_{i+1/2,j} + \mathbf{v}^+_{i+1/2,j}) - (\mathbf{v}^-_{i-1/2,j} + \mathbf{v}^+_{i-1/2,j})\right] \\
&\quad - \frac{1}{2\Delta x}|\tilde{\mathbf{A}}^{(1)}|\left[\mathbf{v}^+_{i+1/2,j} - (\mathbf{v}^-_{i+1/2,j} + \mathbf{v}^+_{i-1/2,j}) + \mathbf{v}^-_{i-i/2,j}\right] \\
&\quad + \frac{1}{2\Delta y}\tilde{\mathbf{A}}^{(2)}\left[(\mathbf{v}^-_{i,j+1/2} + \mathbf{v}^+_{i,j+1/2}) - (\mathbf{v}^-_{i,j-1/2} + \mathbf{v}^+_{i,j-1/2})\right] \\
&\quad - \frac{1}{2\Delta y}|\tilde{\mathbf{A}}^{(2)}|\left[\mathbf{v}^+_{i,j+1/2} - (\mathbf{v}^-_{i,j+1/2} + \mathbf{v}^+_{i,j-1/2}) + \mathbf{v}^-_{i,j-i/2})\right] \\
&\quad + (\sigma_{\mathrm{a}})_{i,j}\mathbf{v}_{i,j} + (\sigma_{\mathrm{s}})_{i,j}\tilde{\mathbf{G}}\mathbf{v}_{i,j} + (\sigma_{\mathrm{f}})_{i,j}\tilde{\mathbf{F}}\mathbf{v}_{i,j} = \tilde{\mathbf{s}}_{i,j}
\end{aligned} \tag{85}$$

where,

$$\mathbf{v}^-_{i+1/2,j} = \mathbf{v}_{i,j} + \frac{\Delta x}{2}\mathbf{p}^x_{i,j}, \qquad \mathbf{v}^+_{i+1/2,j} = \mathbf{v}_{i+1,j} - \frac{\Delta x}{2}\mathbf{p}^x_{i+1,j}, \tag{86}$$

$$\mathbf{v}^-_{i,j+1/2} = \mathbf{v}_{i,j} + \frac{\Delta y}{2}\mathbf{p}^y_{i,j}, \qquad \mathbf{v}^-_{i,j+1/2} = \mathbf{v}_{i,j+1} - \frac{\Delta y}{2}\mathbf{p}^y_{i,j+1} \tag{87}$$

and the slope limiters are defined as

$$\mathbf{p}^x_{i,j} = \operatorname{minmod}\left\{\mathbf{v}_{i+1,j} - \mathbf{v}_{i,j}, \mathbf{v}_{i,j} - \mathbf{v}_{i-1,j}\right\}, \qquad \mathbf{p}^y_{i,j} = \operatorname{minmod}\left\{\mathbf{v}_{i,j+1} - \mathbf{v}_{i,j}, \mathbf{v}_{i,j} - \mathbf{v}_{i,j-1}\right\} \tag{88a}$$

where

$$\operatorname{minmod}(a,b) := \begin{cases} 0, & ab < 0 \\ \operatorname{sgn}(a)\min\left(|a|,|b|\right), & ab > 0. \end{cases} \tag{89}$$

The approximation source is given by $\tilde{\mathbf{s}}_{i,j}(t) = \int_{C_{i,j}} \tilde{\mathbf{s}}(x,y,t)dxdy$ and the cross-sections are given by $(\sigma_\mathrm{a})_{i,j} := \sigma_\mathrm{a}(\boldsymbol{x}_{i,j})$, $(\sigma_\mathrm{s})_{i,j} := \sigma_\mathrm{s}(\boldsymbol{x}_{i,j})$, and $(\sigma_\mathrm{f})_{i,j}(t) = \Sigma_\mathrm{f}\big(\boldsymbol{\xi}(\mathbf{v}_{i,j}(t), \tilde{\mathbf{s}}_{i,j}(t); \boldsymbol{w}\big)$.

We collect the cell averages of the FP_N moments into an array $\vec{\mathbf{v}}$ and express (85) in matrix-vector form as

$$\partial_t \vec{\mathbf{v}} + \mathbf{L}(\Sigma_\mathrm{f}(\boldsymbol{\xi}(\vec{\mathbf{v}}), \vec{\mathbf{s}}; \boldsymbol{w})\vec{\mathbf{v}} = \vec{\mathbf{s}} \tag{90}$$

where $\mathbf{L}$ is the matrix representation of the spatially discretized operators that depends on $\vec{\mathbf{v}}, \vec{\mathbf{s}}$, and $\boldsymbol{w}$ via the filter strength ansatz Σ_f, and $\boldsymbol{\xi}$ are Σ_f act component-wise with respect to the mesh indices i and j.

For the temporal discretization, we employ Heun's method [37], an explicit second-order Runge-Kutta scheme. One time step of Heun's method can be written as

$$\vec{\mathbf{v}}^{(1)} = \vec{\mathbf{v}}^k - \Delta t \mathbf{L}(\Sigma_\mathrm{f}(\boldsymbol{\xi}(\vec{\mathbf{v}}^k), \vec{\mathbf{s}}^k; \boldsymbol{w})\vec{\mathbf{v}}^k + \Delta t \mathbf{s}^k, \tag{91a}$$

$$\vec{\mathbf{v}}^{(2)} = \vec{\mathbf{v}}^{(1)} - \Delta t \mathbf{L}(\Sigma_\mathrm{f}(\boldsymbol{\xi}(\vec{\mathbf{v}}^{(1)}), \vec{\mathbf{s}}^{k+1}; \boldsymbol{w})\vec{\mathbf{v}}^{(1)} + \Delta t \mathbf{s}^{k+1}, \tag{91b}$$

$$\vec{\mathbf{v}}^{k+1} = \frac{1}{2}\left(\vec{\mathbf{v}}^{(1)} + \vec{\mathbf{v}}^{(2)}\right), \tag{91c}$$

where $k \in \{0, \ldots, K-1\}$ is the temporal index. We combine these formula into an update rule for $\vec{\mathbf{v}}^{k+1} = \vec{\mathbf{v}}^{k+1}(\boldsymbol{w})$:

$$\vec{\mathbf{v}}^{k+1}(\boldsymbol{w}) = \mathbf{H}^k(\vec{\mathbf{v}}^k; \boldsymbol{w}), \quad k \in \{0, \ldots, K-1\} \tag{92}$$

where the index of $\mathbf{H}^k$ accounts for the sources $\vec{\mathbf{s}}^k$ and $\vec{\mathbf{s}}^{k+1}$. This update rule is used to compute derivative of $\vec{\mathbf{v}}^{k+1}$ recursively,

$$\frac{\partial \vec{\mathbf{v}}^{k+1}}{\partial \boldsymbol{w}}(\boldsymbol{w}) = \frac{\partial \mathbf{H}^k}{\partial \boldsymbol{w}}(\vec{\mathbf{v}}^k; \boldsymbol{w}) + \frac{\partial \mathbf{H}^k}{\partial \vec{\mathbf{v}}^k}(\vec{\mathbf{v}}^k)\frac{\partial \vec{\mathbf{v}}^k}{\partial \boldsymbol{w}}(\vec{\mathbf{v}}^k, \boldsymbol{w}), \quad k \in \{0, \ldots, K-1\}. \tag{93}$$

The derivatives of $\mathbf{H}^k$ are straight-forward, yet tedious, calculations that apply the chain rule to (91). These calculations involve state and parameter derivatives of Σ_f which are implemented via backpropagation through the neural network ansatz.