

KiT-RT: An extendable framework for radiative transfer and therapy

JONAS KUSCH, University of Innsbruck, Austria

STEFFEN SCHOTTHÖFER, Karlsruhe Institute of Technology, Germany

PIA STAMMER, Karlsruhe Institute of Technology, Germany, German Cancer Research Center - DKFZ, Germany, and HIDSS4Health - Helmholtz Information and Data Science School for Health, Germany

JANNICK WOLTERS, Karlsruhe Institute of Technology, Germany

TIANBAI XIAO, Karlsruhe Institute of Technology, Germany

In this paper we present KiT-RT (Kinetic Transport Solver for Radiation Therapy), an open-source C++ based framework for solving kinetic equations in radiation therapy applications. The aim of this code framework is to provide a collection of classical deterministic solvers for unstructured meshes that allow for easy extendability. Therefore, KiT-RT is a convenient base to test new numerical methods in various applications and compare them against conventional solvers. The implementation includes spherical-harmonics, minimal entropy, neural minimal entropy and discrete ordinates methods. Solution characteristics and efficiency are presented through several test cases ranging from radiation transport to electron radiation therapy. Due to the variety of included numerical methods and easy extendability, the presented open source code is attractive for both developers, who want a basis to build their own numerical solvers and users or application engineers, who want to gain experimental insights without directly interfering with the codebase.

CCS Concepts: • **Mathematics of computing** → **Solvers**; • **Applied computing** → **Physics**.

ACM Reference Format:

Jonas Kusch, Steffen Schotthöfer, Pia Stammer, Jannick Wolters, and Tianbai Xiao. 2022. KiT-RT: An extendable framework for radiative transfer and therapy. *ACM Trans. Math. Softw.* 1, 1 (May 2022), 28 pages. <https://doi.org/10.1145/nnnnnnn.nnnnnnn>

1 INTRODUCTION

Personalized medicine in radiation oncology has been an important research topic in the last decades. To allow for accurate, reliable and efficient treatment planning tailored towards individual patient needs, there is a growing desire to undertake direct numerical simulation for radiation therapy. High-fidelity numerical solutions enable quantitative estimation of the dose received by the tumor as well as the surrounding tissue, while allowing for an automated generation of optimal treatment plans. Besides the aim to ensure sufficient accuracy, such simulations are required to run on limited computational resources such as workstation PCs.

Traditional methods to predict dose distributions in radiation oncology largely rely on simplified models, such as pencil beam models based on the Fermi-Eyges theory [18]. While such models are computationally efficient, they often lack the required accuracy, especially in cases including air cavities or other inhomogeneities [29, 35].

Authors' addresses: Jonas Kusch, University of Innsbruck, Technikerstraße 13, Innsbruck, Austria, jonas.kusch1@gmail.com; Steffen Schotthöfer, Karlsruhe Institute of Technology, Englerstraße 2, Karlsruhe, Germany, steffen.schotthoefer@kit.edu; Pia Stammer, Karlsruhe Institute of Technology, Englerstraße 2, Karlsruhe, Germany and German Cancer Research Center - DKFZ, Heidelberg, Germany and HIDSS4Health - Helmholtz Information and Data Science School for Health, Karlsruhe/Heidelberg, Germany, pia.stammer@kit.edu; Jannick Wolters, Karlsruhe Institute of Technology, Englerstraße 2, Karlsruhe, Germany, jannick.wolters@rwth-aachen.de; Tianbai Xiao, Karlsruhe Institute of Technology, Englerstraße 2, Karlsruhe, Germany, tianbai.xiao@kit.edu.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

© 2022 Association for Computing Machinery.

0098-3500/2022/5-ART \$15.00

<https://doi.org/10.1145/nnnnnnn.nnnnnnn>

On the other hand, Monte Carlo (MC) algorithms, which simulate individual interacting particles, achieve a satisfactory accuracy [7]. However, despite ongoing research to accelerate MC methods, their high computational costs currently renders them impractical for clinical usage [20, 32]. To obtain a computationally feasible model with comparable accuracy, radiation particles are described on a mesoscopic level through the deterministic linear Boltzmann equation [10, 61–63]. An efficient and accurate numerical approximation to the linear Boltzmann equation can be achieved through the construction of grid-based macroscopic approximations [17, 25, 64, 65]. Variants of grid based methods for radiation therapy can be found in for example [8, 28, 30, 37, 39, 54].

The available grid-based methods employ various macroscopic approximations to the linear Boltzmann equation which all exhibit certain advantages and shortcomings.

In [17], the modal entropy method called M_N is used as a macroscopic model. This method preserves positivity of particle distributions while yielding accurate results with little spurious oscillations. However, the need to solve a possibly ill-posed optimization problem in every spatial cell and time step results in increased computational costs. While analytic solutions to the optimization problems are available at small truncation order, they cannot capture all physical effects accurately. Further approaches aim at mitigating the challenge of the problems ill-condition by regularization [4] or reducing the associated computational costs, e.g. using neural networks [58].

In [54] the use of computationally cheap nodal discretizations, known as the S_N method has been proposed. In this case, the solution remains positive and it is shown that the expensive scattering terms can be approximated efficiently with a Fokker-Planck approximation. Furthermore, the solution can exhibit non-physical artifacts, known as ray-effects [41, 50, 52], which reduce the approximation quality. While methods to mitigate ray-effects exist, see e.g. [2, 12, 21, 42, 60], they commonly require picking problem-dependent tuning parameters.

In [37], the modal P_N method has been employed to derive a macroscopic model for radiation treatment planning. While it does not preserve positivity of the solution and can potentially yield oscillatory approximations, it allows for an efficient numerical treatment of scattering terms. In [39] a combination of S_N and P_N methods which reduces computational costs through a dynamical low rank approximation [33] has been proposed. The efficiency of this method relies on the ability to describe the solution through a low-rank function.

The variety of different methods allows for individual method choices tailored to different settings. The comparability of different methods in a uniform framework is not only interesting for clinical usage, but also for future research in computational radiation therapy. Our aim of the open-source C++ *Kinetic Transport for radiation therapy* (KiT-RT) framework is therefore to provide a collection of available deterministic methods. Special focus is put on extendability by the use of polymorphism to simplify the implementation of novel solution methods. The methods provided by our framework are optimized for an application on workstation PCs. This meets the typical requirements in radiation therapy applications: For clinical usage the computational resources are often limited and the time between recording the CT image and the actual treatment must not exceed a certain time period. Hence, radiation therapy codes that are applicable for clinical use should run efficiently on workstation PCs. Moreover, conventional codes often require structured grids [24, 36, 56, 59, 67], leading to inaccurate representations of structures on CT images. While accuracy in practice is also limited by the CT density values which are given on a structured grid, these are often downsampled to a lower resolution for dose computations. Further, a recomputation of the CT values for unstructured grids is feasible if it improves the quality of dose computations. Therefore, our framework provides functionalities for both unstructured meshes which preserve organ outlines on CT images, as well as standard rectangular grids.

This paper aims at presenting the developed framework and its functionality while providing the necessary mathematical and physical background on principles the software is based on. In Section 2 we provide the underlying physical model as well as its reformulation as a time-dependent partial differential equation. Section 3 discusses different macroscopic models as well as advantages and disadvantages of the individual underlying directional discretizations. Sections 4 and 5 focus on the used discretizations and software architecture, respectively. Lastly, we validate our implementations and analyse their performance for different test cases in Section 7.

2 PHYSICAL MODEL

Let us recap the main physical model that is used for computational radiotherapy treatment planning. The main goal is to compute the radiation dose distribution

$$D(\mathbf{x}) = \frac{1}{\rho(\mathbf{x})} \int_0^\infty \int_{\mathbb{S}^2} S(E, \mathbf{x}) \psi(E, \mathbf{x}, \Omega) \, d\Omega \, dE, \quad (1)$$

that results from a given treatment plan. Here, $E \in \mathbb{R}_+$ is the energy, $\mathbf{x} \in \mathbf{X} \subset \mathbb{R}^3$ denotes the spatial domain, and $\Omega \in \mathbb{S}^2$ is the flight direction of particles. Moreover, $\psi : \mathbb{R}_+ \times \mathbb{R}^3 \times \mathbb{S}^2 \rightarrow \mathbb{R}$ denotes the radiation flux density and $\rho : \mathbb{R}^3 \rightarrow \mathbb{R}$ is the patient tissue density. The stopping power $S : \mathbb{R}_+ \times \mathbb{R}^3 \rightarrow \mathbb{R}$ models the continuous energy loss of particles due to scattering with tissue and is defined as

$$S(E, \mathbf{x}) = \int_0^\infty E' \int_{-1}^1 \Sigma(E, E', \mathbf{x}, \mu) \, d\mu \, dE' \quad (2)$$

with the scattering cross section $\Sigma : \mathbb{R}_+ \times \mathbb{R}_+ \times \mathbb{R}^3 \times [-1, 1] \rightarrow \mathbb{R}$. The radiation flux density, which describes the probability of finding a particle at a certain region in phase space, can be computed from the continuous slowing down (CSD) approximation [40] of the energy dependent linear Boltzmann equation

$$\begin{aligned} & -\partial_E (S(E, \mathbf{x}) \psi(E, \mathbf{x}, \Omega)) + \Omega \cdot \nabla_{\mathbf{x}} \psi(E, \mathbf{x}, \Omega) + \Sigma_t(E, \mathbf{x}) \psi(E, \mathbf{x}, \Omega) \\ & = \int_{\mathbb{S}^2} \Sigma_s(E, \mathbf{x}, \Omega \cdot \Omega') \psi(E, \mathbf{x}, \Omega') \, d\Omega'. \end{aligned} \quad (3)$$

This model assumes a continuous energy loss of particles traveling through a background material, which is modeled using the stopping power S . The scattering cross section $\Sigma_s(E, \mathbf{x}, \Omega \cdot \Omega')$ denotes the probability of particles at position \mathbf{x} with energy E changing their flight direction from Ω' to Ω due to a collision with the patient tissue. The total cross section Σ_t is given by

$$\Sigma_t(E, \mathbf{x}) = \Sigma_{s,0}(E, \mathbf{x}) = 2\pi \int_{-1}^1 \Sigma_s(E, \mathbf{x}, \mu) \, d\mu. \quad (4)$$

To simplify the evaluation of material properties, we follow the common assumption that all materials are water-equivalent and differ only in density [e.g. 37, 54, 68], i.e.,

$$\begin{aligned} S(E, \mathbf{x}) &= S^{H_2O}(E) \rho(\mathbf{x}), \\ \Sigma_t(E, \mathbf{x}) &= \Sigma_t^{H_2O}(E) \rho(\mathbf{x}), \\ \Sigma_s(E, \mathbf{x}, \Omega \cdot \Omega') &= \Sigma_s^{H_2O}(E, \Omega \cdot \Omega') \rho(\mathbf{x}), \end{aligned} \quad (5)$$

where we leave out the superscript H_2O in the following. Cross-sections for water are taken from the ICRU database [26]. Having defined the prerequisites of our physical model, we can focus on bringing it into a form that allows for computing numerical approximations efficiently. It turns out that the energy variable in (3) can be treated as a pseudo-time, which facilitates solving the CSD equation. For a given maximal energy E_{\max} let us define the transformed energy as

$$\tilde{E}(E) := \int_0^{E_{\max}} \frac{1}{S(E')} \, dE' - \int_0^E \frac{1}{S(E')} \, dE' \quad (6)$$

and the transformed particle density as

$$\tilde{\psi}(\tilde{E}, \mathbf{x}, \Omega) := S(E) \rho(\mathbf{x}) \psi(E(\tilde{E}), \mathbf{x}, \Omega). \quad (7)$$

Then, multiplying (3) with $S(E)$ and plugging in the defined transformation gives

$$\partial_{\tilde{E}} \tilde{\psi}(\tilde{E}, \mathbf{x}, \Omega) + \Omega \cdot \nabla_{\mathbf{x}} \frac{\tilde{\psi}(\tilde{E}, \mathbf{x}, \Omega)}{\rho(\mathbf{x})} + \tilde{\Sigma}_t(\tilde{E}) \tilde{\psi}(\tilde{E}, \mathbf{x}, \Omega) = \int_{\mathbb{S}^2} \tilde{\Sigma}_s(\tilde{E}, \Omega \cdot \Omega') \tilde{\psi}(\tilde{E}, \mathbf{x}, \Omega') d\Omega', \quad (8)$$

where we define $\tilde{\Sigma}_t(\tilde{E}) := \Sigma_t(E(\tilde{E}))$ and $\tilde{\Sigma}_s(\tilde{E}, \Omega \cdot \Omega') := \Sigma_s(E(\tilde{E}), \Omega \cdot \Omega')$. Dropping the tilde notation and treating \tilde{E} as a pseudo-time t gives a slightly modified version of the classical linear Boltzmann equation

$$\begin{aligned} \partial_t \psi(t, \mathbf{x}, \Omega) + \Omega \cdot \nabla_{\mathbf{x}} \frac{\psi(t, \mathbf{x}, \Omega)}{\rho(\mathbf{x})} + \Sigma_t(t) \psi(t, \mathbf{x}, \Omega) &= \int_{\mathbb{S}^2} \Sigma_s(t, \Omega \cdot \Omega') \psi(t, \mathbf{x}, \Omega') d\Omega' \\ \psi(t = 0, \mathbf{x}, \Omega) &= S(E_{\max}) \rho(\mathbf{x}) \psi(E_{\max}, \mathbf{x}, \Omega). \end{aligned} \quad (9)$$

Hence, the CSD equation can be treated numerically with classical closure methods and space-time discretizations. Let us first discuss methods to discretize the directional domain, yielding macroscopic evolution equations.

3 MACROSCOPIC MODELS

This section discusses macroscopic models to (9). These models are derived from nodal and modal discretizations of the directional domain.

3.1 Modal discretizations

Modal discretizations of (9) can be interpreted as a closure problem [44, 45]. To present the derivation of different closures, we first formulate the moment equations which are not closed. Second, we close these equations with the P_N closure and third, we derive the M_N closure.

Moment equations. Let us derive an evolution equation to describe the moments of the radiation flux with respect to the real-valued spherical harmonics basis functions. These are defined as the real parts of the spherical harmonics

$$Y_\ell^k(\Omega) = \sqrt{\frac{2\ell + 1}{4\pi} \frac{(\ell - k)!}{(\ell + k)!}} e^{ik\varphi} P_\ell^k(\mu),$$

where P_ℓ^k are the associated Legendre polynomials. Then, the real spherical harmonics are given as

$$m_\ell^k(\Omega) = \begin{cases} \frac{(-1)^k}{\sqrt{2}} (Y_\ell^k(\Omega) + (-1)^k Y_\ell^{-k}(\Omega)), & k > 0, \\ Y_\ell^0(\Omega) & k = 0, \\ -\frac{(-1)^k i}{\sqrt{2}} (Y_\ell^{-k}(\Omega) - (-1)^k Y_\ell^k(\Omega)), & k < 0, \end{cases}$$

where i is the imaginary unit. Collecting all basis functions up to degree N in a vector

$$\mathbf{m} = \left(m_0^0, m_1^{-1}, m_1^0, m_1^1, \dots, m_N^N \right)^T \in \mathbb{R}^{(N+1)^2}$$

yields the so-called moments

$$u_\ell^k(t, \mathbf{x}) := \int_{\mathbb{S}^2} \psi(t, \mathbf{x}, \Omega) m_\ell^k(\Omega) d\Omega.$$

Evolution equations for the moments can be derived by testing (9) against $\mathbf{m}_\ell = (m_\ell^{-\ell}, \dots, m_\ell^\ell)$, which gives

$$\begin{aligned} \partial_t \mathbf{u}_\ell(t, \mathbf{x}) + \nabla_{\mathbf{x}} \cdot \int_{\mathbb{S}^2} \Omega \mathbf{m}_\ell(\Omega) \frac{\psi(t, \mathbf{x}, \Omega)}{\rho(\mathbf{x})} d\Omega + \Sigma_\ell(t) \mathbf{u}_\ell(t, \mathbf{x}) \\ = \int_{\mathbb{S}^2} \int_{\mathbb{S}^2} \mathbf{m}_\ell(\Omega) \Sigma_s(t, \Omega \cdot \Omega') \psi(t, \mathbf{x}, \Omega') d\Omega' d\Omega . \end{aligned} \quad (10)$$

To rewrite this equation, we use the spherical harmonics recursion relation

$$\Omega_i \mathbf{m}_\ell = \mathbf{a}_\ell^i \mathbf{m}_{\ell-1} + \mathbf{a}_{\ell+1}^i \mathbf{m}_{\ell+1} \quad \text{with } \mathbf{a}_\ell^i \in \mathbb{R}^{(2\ell-1) \times (2\ell+1)}$$

as well as the fact that there exists a diagonal matrix $\Sigma_\ell(t)$ with entries $\Sigma_{\ell, kk} = \Sigma_\ell^k := 2\pi \int_{[-1,1]} P_\ell(\mu) \Sigma_s(t, \mu) d\mu$ such that

$$\int_{\mathbb{S}^2} \int_{\mathbb{S}^2} \mathbf{m}_\ell(\Omega) \Sigma_s(t, \Omega \cdot \Omega') \psi(t, \mathbf{x}, \Omega') d\Omega' d\Omega = \Sigma_\ell(t) \mathbf{u}_\ell(t, \mathbf{x}) .$$

Then, the moment equations at degree ℓ become

$$\begin{aligned} \partial_t \mathbf{u}_\ell(t, \mathbf{x}) + \sum_{i=1}^3 \partial_{x_i} (\mathbf{a}_\ell^i \mathbf{u}_{\ell-1}(t, \mathbf{x}) + \mathbf{a}_{\ell+1}^i \mathbf{u}_{\ell+1}(t, \mathbf{x})) / \rho(\mathbf{x}) + \Sigma_\ell(t) \mathbf{u}_\ell(t, \mathbf{x}) \\ = \Sigma_\ell(t) \mathbf{u}_\ell(t, \mathbf{x}) . \end{aligned} \quad (11)$$

Note that the equations for degree ℓ depend on the moments of degree $\ell + 1$. Hence, to obtain a closed system of moments up to a fixed degree N , we need to define a closure relation

$$\mathbf{u}_{N+1}(t, \mathbf{x}) \simeq \mathcal{U}(\mathbf{u}_0(t, \mathbf{x}), \dots, \mathbf{u}_N(t, \mathbf{x})) . \quad (12)$$

P_N closure. The most commonly used closure is the P_N closure which leads to the spherical harmonics (P_N) method [14]. It expands the solution by spherical harmonics, i.e.,

$$\psi(t, \mathbf{x}, \Omega) \approx \psi_{P_N}(t, \mathbf{x}, \Omega) := \mathbf{u}(t, \mathbf{x})^T \mathbf{m}(\Omega) , \quad (13)$$

where $\mathbf{u} \in \mathbb{R}^{(N+1)^2}$ collects all moments according to $\mathbf{u} = (u_0^0, u_1^{-1}, u_1^0, u_1^1, \dots, u_N^N)^T \in \mathbb{R}^{(N+1)^2}$. Hence, the P_N closure is simply given as $\mathcal{U}_{P_N} \equiv \mathbf{0}$. In this case, the moment equations read

$$\partial_t \mathbf{u}(t, \mathbf{x}) = -\mathbf{A} \cdot \nabla_{\mathbf{x}} \frac{\mathbf{u}(t, \mathbf{x})}{\rho(\mathbf{x})} - \Sigma_\ell(t) \mathbf{u}(t, \mathbf{x}) + \Sigma \mathbf{u}(t, \mathbf{x}) ,$$

where $\mathbf{A} \cdot \nabla_{\mathbf{x}} := \mathbf{A}_1 \partial_x + \mathbf{A}_2 \partial_y + \mathbf{A}_3 \partial_z$ with $\mathbf{A}_i := \int_{\mathbb{S}^2} \mathbf{m} \mathbf{m}^T \Omega_i d\Omega$ and $\Sigma = \text{diag}(\Sigma_0^0, \Sigma_1^{-1}, \Sigma_1^0, \Sigma_1^1, \dots, \Sigma_N^N)$. While P_N is a computationally efficient method (especially for scattering terms), it does not preserve positivity of the radiation flux approximation and can lead to spurious oscillations. A closure which mitigates oscillations and preserves positivity at significantly increased computational costs is the M_N closure.

M_N closure. The M_N closure [44, 45] employs the principle of minimal mathematical, i.e., maximal physical entropy to close the moment system. To this end, we define the twice differentiable, strictly convex kinetic entropy density $\eta : \mathbb{R}_+ \rightarrow \mathbb{R}$. Different, problem specific entropy densities can be defined, e.g. the Maxwell-Boltzmann entropy $\eta(g) = g \log(g) - g$, or a quadratic entropy $\eta(g) = g^2$, which recovers the P_N method. Thus, one can close the system by choosing the reconstructed radiation flux density $\psi_{\mathbf{u}}$ out of the set of all possible functions

$$F_{\mathbf{m}} = \left\{ g(t, \mathbf{x}, \Omega) > 0 : u = \int_{\mathbb{S}^2} \mathbf{m} g d\Omega < \infty \right\} , \quad (14)$$

that fulfill the moment condition $\mathbf{u}(t, \mathbf{x}) = \langle \mathbf{m}g \rangle$ as the one with minimal entropy. The modal basis \mathbf{m} can be chosen arbitrarily. Common choices consist of spherical harmonics or other polynomial basis functions. The minimal entropy closure can be formulated as a constrained optimization problem for a given vector of moments \mathbf{u} ,

$$\min_{g \in F_{\mathbf{m}}} \int_{\mathbb{S}^2} \eta(g) \, d\Omega \quad \text{s.t. } \mathbf{u} = \int_{\mathbb{S}^2} \mathbf{m}g \, d\Omega. \quad (15)$$

The minimal value of the objective function is denoted by $h(\mathbf{u}) = \langle \eta(\psi_{\mathbf{u}}) \rangle$ and describes the systems minimal entropy depending on time and space. $\psi_{\mathbf{u}}$ is the minimizer of Eq. (15), which we use to close the moment system

$$\partial_t \mathbf{u}_\ell(t, \mathbf{x}) + \nabla_{\mathbf{x}} \cdot \int_{\mathbb{S}^2} \Omega \mathbf{m}_\ell(\Omega) \frac{\psi_{\mathbf{u}}(t, \mathbf{x}, \Omega)}{\rho(\mathbf{x})} \, d\Omega + \Sigma_\ell(t) \mathbf{u}_\ell(t, \mathbf{x}) = \Sigma_\ell \mathbf{u}_\ell(t, \mathbf{x}); \quad (16)$$

The minimal entropy method preserves important properties of the underlying equation [4, 45], i.e., positivity of the solution, hyperbolicity of the moment system, dissipation of mathematical entropy and the H-Theorem. The minimal entropy closed moment system is invariant under Galilean transforms. Lastly, if collision invariants of the Boltzmann equations are used as modal basis functions, then the moment system yields a local conservation law.

The set of all moments corresponding to a radiation flux density $\psi_{\mathbf{u}} > 0$ is called the realizable set

$$\mathcal{R} = \left\{ \mathbf{u} : \int_{\mathbb{S}^2} \mathbf{m}g \, d\Omega = \mathbf{u}, g \in F_{\mathbf{m}} \right\}. \quad (17)$$

Outside \mathcal{R} the minimal entropy closure problem has no solution. At the boundary $\partial\mathcal{R}$, the optimization problem becomes singular and $\psi_{\mathbf{u}}$ consists of a linear combination of dirac distributions. Near $\partial\mathcal{R}$ the entropy closure becomes ill conditioned and thus, a numerical optimizer requires a large amount of iterations to compute a solution.

To mitigate this issue, a regularized version of the entropy closure problem has been proposed by [4],

$$\inf_{g \in F_{\mathbf{m}}} \int_{\mathbb{S}^2} \eta(g) \, d\Omega + \frac{1}{2\gamma} \left\| \int_{\mathbb{S}^2} \mathbf{m}g \, d\Omega - \mathbf{u} \right\|_2^2, \quad (18)$$

where γ is the regularization parameter. Generally, moments of the regularized reconstructed radiation flux density $\int_{\mathbb{S}^2} \mathbf{m}\psi_{\mathbf{u}} \, d\Omega$ deviate from the non-regularized moments. For $\gamma \rightarrow 0$, we recover the original entropy closure of Eq. (15) and the moments coincide again. The regularized entropy closure is solvable for any $\mathbf{u} \in \mathbb{R}^{(N+1)^2}$ and preserves all structural properties of the non-regularized entropy closure [4]. One can also choose to regularize only parts of the entropy closure, e.g. to preserve moments of specific interest. Then the partially regularized entropy closure reads

$$\inf_{g \in F_{\mathbf{m}}} \int_{\mathbb{S}^2} \eta(g) \, d\Omega + \frac{1}{2\gamma} \left\| \int_{\mathbb{S}^2} \mathbf{m}^r g \, d\Omega - \mathbf{u}^r \right\|_2^2 \quad \text{s.t. } \mathbf{u}^{nr} = \int_{\mathbb{S}^2} \mathbf{m}^{nr} g \, d\Omega, \quad (19)$$

where \mathbf{u}^{nr} denotes non-regularized moment elements and \mathbf{u}^r denotes regularized elements of the moment vector. Recently, structure preserving deep neural networks have been successfully employed to approximate the entropy closure [58] to accelerate the M_N method. The authors leverage convexity of the optimization problem and use corresponding input convex neural networks [6] to preserve structural properties of the closure in the neural network based approximation.

3.2 Nodal discretizations

The S_N method [46] employs a nodal discretization for the directional domain. To facilitate the computation of integral terms that arise due to scattering, the nodal point sets are commonly chosen according to a quadrature rule. In the application case of radiative transport, the directional domain is assumed to be the unit sphere $\mathbb{S}^2 \subset \mathbb{R}^3$, thus a suitable parametrization is given by spherical coordinates

$$\mathbb{S}^2 = \left\{ \begin{pmatrix} \sqrt{1-\mu^2} \sin(\varphi) \\ \sqrt{1-\mu^2} \cos(\varphi) \\ \mu \end{pmatrix} : \mu \in [-1, 1], \varphi \in [0, 2\pi) \right\}. \quad (20)$$

Note, that we can allow different particle velocities by scaling the unit sphere with a given maximum velocity. The implementation assumes a slab geometry setting, i.e., lower dimensional velocity spaces are described by a projection of \mathbb{S}^2 onto \mathbb{R}^2 and \mathbb{R} , respectively. Thus, the parametrization of the two-dimensional slab geometry is given by

$$P_{\mathbb{R}^2}(\mathbb{S}^2) = \left\{ \begin{pmatrix} \sqrt{1-\mu^2} \sin(\varphi) \\ \sqrt{1-\mu^2} \cos(\varphi) \end{pmatrix} : \mu \in [-1, 1], \varphi \in [0, 2\pi) \right\} \quad (21)$$

and the one dimensional case is described by

$$P_{\mathbb{R}}(\mathbb{S}^2) = \{\mu : \mu \in [-1, 1]\} \quad (22)$$

Hence the task is to derive a quadrature formula for the direction of travel. The perhaps most common approach is the product quadrature rule. Here, a Gauss quadrature is used for μ and equally weighted and spaced points for φ , i.e., when using N_q points, we have

$$\varphi_i = i\Delta\varphi \quad \text{for } i = 1, \dots, N_q \quad \text{and} \quad \Delta\varphi = \frac{2\pi}{N_q}. \quad (23)$$

If the Gauss quadrature for μ uses N_q points, then we obtain a total of $Q = N_q^2$ possible directions. Denoting the Gauss weights as w_k^G with $k = 1, \dots, N_q$, we obtain the product quadrature weights

$$w_{k \cdot N_q + \ell} = \frac{2\pi w_k^G}{N_q}$$

and points

$$\Omega_{k \cdot N_q + \ell} = \begin{pmatrix} \sqrt{1-\mu_k^2} \sin(\varphi_\ell) \\ \sqrt{1-\mu_k^2} \cos(\varphi_\ell) \\ \mu_k \end{pmatrix}. \quad (24)$$

Other implemented quadrature methods include spherical Monte Carlo, Levelsymmetric [48], LEBEDEV [49] and LDFESA [31]. A comparison of different quadrature sets and their approximation behaviour for S_N methods can be found in [13].

The evolution equations for $\psi_q(t, \mathbf{x}) := \psi(t, \mathbf{x}, \Omega_q)$ are then given by

$$\partial_t \psi_q(t, \mathbf{x}) + \Omega_q \cdot \nabla_{\mathbf{x}} \frac{\psi_q(t, \mathbf{x})}{\rho(\mathbf{x})} + \Sigma_t(t) \psi_q(t, \mathbf{x}) = \sum_{p=1}^Q w_p \Sigma_s(t, \Omega_q \cdot \Omega_p) \psi_p(t, \mathbf{x}). \quad (25)$$

A main disadvantage of S_N methods are so called ray-effects [41, 50, 52], which are spurious artifacts that stem from the limited number of directions in which particles can travel. Moreover, radiation therapy applications exhibit forward-peaked scattering, which cannot be well-captured by classical quadrature rules.

To allow for moderate computational costs when computing scattering terms and to efficiently treat forward-peaked scattering, we transform the nodal solution to a modal description and apply the more efficient P_N methodology for scattering terms. For this, we define a truncation order N and construct the matrices $\mathbf{O} \in \mathbb{R}^{Q \times (N+1)^2}$ which maps the modal onto its nodal representation and $\mathbf{M} \in \mathbb{R}^{(N+1)^2 \times Q}$ which maps the nodal onto its modal representation. Such matrices can be constructed by

$$\mathbf{O} = (\mathbf{m}(\Omega_k))_{k=1}^Q, \text{ and } \mathbf{M} = (w_k \mathbf{m}(\Omega_k))_{k=1}^Q.$$

In this case, we can replace the scattering term on the right-hand side of (25) by its P_N counterpart. Collecting the nodal solution in the vector $\boldsymbol{\psi}$ then gives

$$\partial_t \boldsymbol{\psi}(t, \mathbf{x}) + \Omega_q \cdot \nabla_{\mathbf{x}} \frac{\boldsymbol{\psi}(t, \mathbf{x})}{\rho(\mathbf{x})} + \Sigma_t(t) \boldsymbol{\psi}(t, \mathbf{x}) = \mathbf{O} \Sigma \mathbf{M} \boldsymbol{\psi}. \quad (26)$$

4 DISCRETIZATION METHODS

4.1 Spatial Discretization

The KiT-RT framework is based on unstructured, cell-centered grids as spatial discretization. In the following, we restrict ourselves to a two-dimensional spatial grid, however, the notations are straight forwardly extended to three or one spatial dimensions. A unstructured grid $\tilde{\mathbf{X}} = \{\mathbf{X}_i\}_{i \in I}$ is a partition of a bounded spatial domain $\mathbf{X} \subset \mathbb{R}^d$. A grid cell \mathbf{X}_i holds information about the coordinates of its centroid \mathbf{x}_i , its measure A_i , the indices of its boundary vertices, indices of its neighbor cells $N(i)$ and cell faces. The information of the cell faces is encoded in the unit-normal vector of the face dividing cell i and its neighbor $j \in N(i)$, multiplied with the measure of the face and is denoted by $\mathbf{n}_{i,j}$. The grids used in this work are either triangular or quadrilateral, unstructured grids in two spatial dimensions.

4.2 Finite volume methods

The nodal and modal methods are different approaches to discretize the velocity space of the Boltzmann equation, however all of them result in a system of transport equations, that can be solved using a finite volume scheme. Thus, we first describe a method agnostic finite volume scheme and afterwards point out the differences of the S_N , P_N and M_N based implementations. We denote the temporal variable by t , however the results hold for energy interpreted as pseudo-time as well. Let $\mathbf{g}(t, \mathbf{x}) \in \mathbb{R}^m$ be the vector of conserved variables of a system of transport equations

$$\partial_t \mathbf{g}(t, \mathbf{x}) + \nabla_{\mathbf{x}} \cdot \mathbf{F}(\mathbf{g}(t, \mathbf{x})) = \mathbf{R}(t, \mathbf{x}, \mathbf{g}(t, \mathbf{x})), \quad \mathbf{x} \in \mathbf{X}, t \in [0, t_f] \quad (27)$$

where \mathbf{F} is the general flux function describing the solution transport, and \mathbf{R} is a general right hand side, containing velocity discretizations of collision terms, sources and absorption terms. The main discretization strategy is to divide the spatial domain into an unstructured grid with N_x cells and the time domain into N_t discrete values $0 = t_0 < \dots < t_{N_t-1}$. We consider the solution as an average over one space-time cell

$$\mathbf{g}_i^n = \frac{1}{A_i} \int_{\mathbf{X}_i} \mathbf{g} \, d\mathbf{x} \quad (28)$$

and average Eq. (27) over one space-time cell

$$\begin{aligned} & \frac{1}{\Delta t A_i} \int_{\mathbf{X}_i} \int_{t_n}^{t_{n+1}} \partial_t \mathbf{g}(t, \mathbf{x}) \, dt \, d\mathbf{x} + \\ & \frac{1}{\Delta t A_i} \int_{\mathbf{X}_i} \int_{t_n}^{t_{n+1}} \nabla_{\mathbf{x}} \cdot \mathbf{F}(\mathbf{g}(t, \mathbf{x})) \, dt \, d\mathbf{x} = \frac{1}{\Delta t A_i} \int_{\mathbf{X}_i} \int_{t_n}^{t_{n+1}} \mathbf{R}(t, \mathbf{x}, \mathbf{g}(t, \mathbf{x})) \, dt \, d\mathbf{x}, \end{aligned} \quad (29)$$

where $\Delta t = t_{n+1} - t_n$. Solving the integrals using the Gauss theorem for the advection term and an explicit Euler scheme for the time derivative yields

$$\frac{1}{\Delta t} (\mathbf{g}_i^{n+1} - \mathbf{g}_i^n) + \frac{1}{\Delta t A_i} \int_{t_n}^{t_{n+1}} \sum_{j \in N(i)} F(\mathbf{g}(t, \mathbf{x}_{i,j})) \cdot \mathbf{n}_{i,j} dt = \frac{1}{\Delta t} \int_{t_n}^{t_{n+1}} R(t, \mathbf{x}, \mathbf{g}(t, \mathbf{x}_i)) dt, \quad (30)$$

where $g(t, \mathbf{x}_i)$ is the conserved variable evaluated at cell i and $g(t, \mathbf{x}_{i,j})$ is the conserved variable evaluated at the interface between cell i and its neighbor j . To compute the actual value of \mathbf{g}_j^{n+1} , one needs to find approximations for the flux integral. A common ansatz is of the form

$$F(\mathbf{g}_j^n, \mathbf{g}_i^n) \approx \frac{1}{\Delta t} \int_{t_n}^{t_{n+1}} F(\mathbf{g}(t, \mathbf{x}_{i,j})) \cdot \mathbf{n}_{i,j} dt, \quad (31)$$

where the numerical flux $F(\mathbf{g}_j^n, \mathbf{g}_i^n)$ at face (i, j) is approximated using the cell averaged conserved variable at cell i and j . For transport equations, a well known numerical flux is given by the Upwind scheme [43]

$$F(\mathbf{g}_j^n, \mathbf{g}_i^n)_{up} = F(\mathbf{g}_i^n) \cdot \mathbf{n}_{i,j} H(\mathbf{n}_{i,j} \cdot \mathbf{v}) + F(\mathbf{g}_j^n) \cdot \mathbf{n}_{i,j} (1 - H(\mathbf{n}_{i,j} \cdot \mathbf{v})), \quad (32)$$

where H is the heaviside step function and \mathbf{v} is the transport velocity vector. Finally, we approximate the source, absorption and collision terms using the current cell average. Thus the explicit solution iteration of a first order scheme reads

$$\mathbf{g}_i^{n+1} = \mathbf{g}_i^n - \frac{\Delta t}{A_i} \sum_{j \in N(i)} F(\mathbf{g}_j^n, \mathbf{g}_i^n)_{up} + \Delta t R(t, \mathbf{x}_i, \mathbf{g}_i^n). \quad (33)$$

Due to the fact that the scattering term R is commonly stiff, implicit-explicit (IMEX) schemes can be used to remove influences of scattering from time step restrictions. If we assume a linear scattering term, that is with a given matrix \mathbf{R}_i^{n+1} we have $R(t^{n+1}, \mathbf{x}_i, \mathbf{g}_i^n) = \mathbf{R}_i^{n+1} \mathbf{g}_i^n$, the IMEX scheme reads

$$(\mathbf{I} - \Delta t \mathbf{R}_i^{n+1}) \mathbf{g}_i^{n+1} = \mathbf{g}_i^n - \frac{\Delta t}{A_i} \sum_{j \in N(i)} F(\mathbf{g}_j^n, \mathbf{g}_i^n)_{up}. \quad (34)$$

4.3 Second order finite volume schemes

The KiT-RT solver provides the option to evaluate the space and time discretizations using second order accurate schemes. To this end, we use a Heun scheme for the temporal discretization and a second order upwind flux for the numerical flux [9]. Whereas, first order spatial fluxes assume a constant solution value \mathbf{g}_i^n in a cell i , a second order upwind scheme is based on a linear reconstruction of the conserved variable. Therefore the inputs \mathbf{g}_i^n and \mathbf{g}_j^n to the numerical flux of Eq. (32) are replaced by

$$\tilde{\mathbf{g}}_i^n = \mathbf{g}_i^n + \Psi_i (\nabla_{\mathbf{x}} \mathbf{g}_i^n \cdot \mathbf{r}_{i,j}), \quad (35)$$

$$\tilde{\mathbf{g}}_j^n = \mathbf{g}_j^n + \Psi_j (\nabla_{\mathbf{x}} \mathbf{g}_j^n \cdot \mathbf{r}_{j,i}), \quad (36)$$

where $\mathbf{r}_{i,j}$ is the vector pointing from cell centroid \mathbf{x}_i of cell i to the interface midpoint between cells i and j , and Ψ_i is the flux limiter for cell i . This reconstruction is formally second order accurate on regular grids [3] assuming exact evaluation of the gradient $\nabla_{\mathbf{x}} \mathbf{g}_i^n$. The gradient of the conserved variable \mathbf{g}_i^n is evaluated using the Green-Gauss theorem with interpolated solution values at the cell interfaces,

$$\nabla_{\mathbf{x}} \mathbf{g}_i^n \approx \frac{1}{A_i} \sum_{j \in N(i)} \frac{1}{2} (\mathbf{g}_i^n + \mathbf{g}_j^n) \cdot \mathbf{n}_{i,j}. \quad (37)$$

Second or higher order upwind spatial discretizations require the use of flux limiters in order to prevent the generation of oscillations in shock regions and to achieve a monotonicity preserving scheme. In the KiT-RT package, the Barth and Jespersen [9] limiter as well as the Venkatakrishnan limiter [66] are implemented. Exemplary, we show the computation of the Barth and Jespersen limiter at cell i

$$\Psi_i = \min_j \begin{cases} \min(1, \frac{\mathbf{g}_{\max} - \mathbf{g}_i}{\Delta_2}), & \text{if } \Delta_2 > 0 \\ \min(1, \frac{\mathbf{g}_{\min} - \mathbf{g}_i}{\Delta_2}), & \text{if } \Delta_2 < 0, \\ 1, & \text{else} \end{cases} \quad (38)$$

where we have

$$\Delta_2 = \frac{1}{2} \nabla_{\mathbf{x}} \mathbf{g}_i^n \cdot \mathbf{r}_{i,j}, \quad (39)$$

$$\mathbf{g}_{\max} = \max(\mathbf{g}_i, \mathbf{g}_j), \quad (40)$$

$$\mathbf{g}_{\min} = \min(\mathbf{g}_i, \mathbf{g}_j). \quad (41)$$

The second order Heun scheme for temporal discretization is a two step Runge-Kutta scheme with the iteration formula

$$\begin{aligned} \mathbf{g}_i^* &= \mathbf{g}_i^n - \frac{\Delta t}{A_i} \sum_{j \in N(i)} F(\mathbf{g}_j^n, \mathbf{g}_i^n)_{up} + \frac{\Delta t}{A_i} R(\mathbf{g}_i^n), \\ \mathbf{g}_i^{**} &= \mathbf{g}_i^* - \frac{\Delta t}{A_i} \sum_{j \in N(i)} F(\mathbf{g}_j^*, \mathbf{g}_i^*)_{up} + \frac{\Delta t}{A_i} R(\mathbf{g}_i^*), \\ \mathbf{g}_i^{n+1} &= \frac{1}{2} (\mathbf{g}_i^n + \mathbf{g}_i^{**}), \end{aligned} \quad (42)$$

which is based on the implicit trapezoidal integration method.

4.4 Numerical Fluxes

In the following we adapt the introduced numerical methods to the method specific notation and present the detailed implementation. The space and time averaged conservative variables \mathbf{g}_i^n for the nodal discretization are given by the vector of the radiation flux $\boldsymbol{\psi} = [\psi_1, \dots, \psi_{N_q}]^T$ evaluated at the quadrature points and for the modal discretization by the moment vector \mathbf{u} . The different methods are distinguishable by their numerical flux function. The corresponding numerical flux for the S_N method is given by

$$F(\boldsymbol{\psi}_i^n) = \boldsymbol{\Omega} \otimes \frac{\boldsymbol{\psi}_i^n}{\rho(\mathbf{x}_i)} \quad (43)$$

and the corresponding upwind flux reads

$$F(\boldsymbol{\psi}_j^n, \boldsymbol{\psi}_i^n)_{up} = \boldsymbol{\Omega} \cdot \mathbf{n}_{i,j} \frac{\boldsymbol{\psi}_i^n}{\rho(\mathbf{x}_i)} H(\mathbf{n}_{i,j} \cdot \boldsymbol{\Omega}) + \boldsymbol{\Omega} \cdot \mathbf{n}_{i,j} \frac{\boldsymbol{\psi}_j^n}{\rho(\mathbf{x}_j)} (1 - H(\mathbf{n}_{i,j} \cdot \boldsymbol{\Omega})). \quad (44)$$

The numerical flux for the P_N method reads

$$F(\mathbf{u}_i^n) = [\mathbf{A}_1 \mathbf{u}_i^n, \mathbf{A}_2 \mathbf{u}_i^n, \mathbf{A}_3 \mathbf{u}_i^n]^T, \quad (45)$$

where A_l are the flux Jacobians emerging from the spherical harmonics recursion scheme. To evaluate the numerical flux with an upwind scheme, we decompose the flux Jacobians in their positive and negative definite parts,

$$A_l = A_l^+ + A_l^-, \quad l = 1, \dots, d \quad (46)$$

Then the numerical flux is given by

$$F(\mathbf{u}_i^n, \mathbf{u}_j^n)_{up} = \sum_{l=1}^d \left(A_l^+ \frac{\mathbf{u}_i^n}{\rho(\mathbf{x}_i)} + A_l^- \frac{\mathbf{u}_j^n}{\rho(\mathbf{x}_j)} \right) n_l H(n_l) + \left(A_l^- \frac{\mathbf{u}_i^n}{\rho(\mathbf{x}_i)} + A_l^+ \frac{\mathbf{u}_j^n}{\rho(\mathbf{x}_j)} \right) n_l (1 - H(n_l)). \quad (47)$$

In contrast to the P_N method, the flux function of the M_N method cannot be expressed as a matrix multiplication, but reads

$$F(\mathbf{u}_i^n) = \int_{\mathbb{S}} \Omega \otimes \mathbf{m}(\Omega) \psi_{\mathbf{u}_i^n}(\Omega) d\Omega, \quad (48)$$

where $\psi_{\mathbf{u}_i^n}$ is the reconstructed radiation flux density of the minimal entropy closure at the cell averaged moment \mathbf{u}_i^n . Using a quadrature rule for the velocity integral discretization and a numerical flux for every quadrature point, we arrive at the kinetic numerical flux

$$F(u_j^n, u_i^n)_{up} = \sum_{q=1}^Q w_q \mathbf{m}_q \Omega_q \cdot \mathbf{n}_{i,j} \left[\frac{\psi_{\mathbf{u}_i^n, q}}{\rho(\mathbf{x}_i)} H(\Omega_q \cdot \mathbf{n}_{i,j}) + \frac{\psi_{\mathbf{u}_j^n, q}}{\rho(\mathbf{x}_j)} (1 - H(\Omega_q \cdot \mathbf{n}_{i,j})) \right]. \quad (49)$$

Note, that the updated solution of the M_N method must still be a feasible moment for the minimal entropy closure of Eq (15). To ensure this, one must either employ a flux-limiter [36], construct a realizability reconstruction [38] or employ the regularized entropy closure formulation.

The numerical framework supports the usual Neumann and Dirichlet boundary conditions.

5 SOFTWARE ARCHITECTURE

The design principle of the KiT-RT software package is focused on efficient implementation, high re-usability of its components and ease of extension. It contains a set of efficient numerical solvers for radiation transport, which are constructed of basic, reusable building blocks. These building blocks can be freely arranged to implement new solvers or tools for completely different applications. On the other hand, KiT-RT is equipped with an easy to use command line interface based on readable configuration files, which allow easy manipulation of the solvers. Thus the software is attractive for developers, who want to experiment with the framework and build their own numerical solvers as well as users and application engineers, who want to gain experimental insights without directly interfering with the codebase.

KiT-RT is implemented in modern C++ and uses mainly polymorphism for its construction. In the following, we present the class structures used to build the numerical solvers and explain the used building blocks, which are displayed in Fig. 1. Most building blocks consist of a virtual base class, which contains a static factory method to build an instance of the concrete derived class, defined by the given configuration details. Furthermore, the virtual base class defines the interface of this building block with other parts of the KiT-Framework.

5.1 Solver Class

The virtual SolverBase class is the basic blueprint for all time or energy dependent finite volume solvers of the KiT-RT framework. It holds an instance of the Config, NumericalFlux, ProblemBase, QuadratureBase and Mesh class.

It controls the screen, log and volume output of the solver. The screen output provides instantaneous feedback of the solver state via the command line and gives information on the current iteration, the total mass of the system, the residual of the radiation flux as well as the flow field and whether logs and volume outputs have

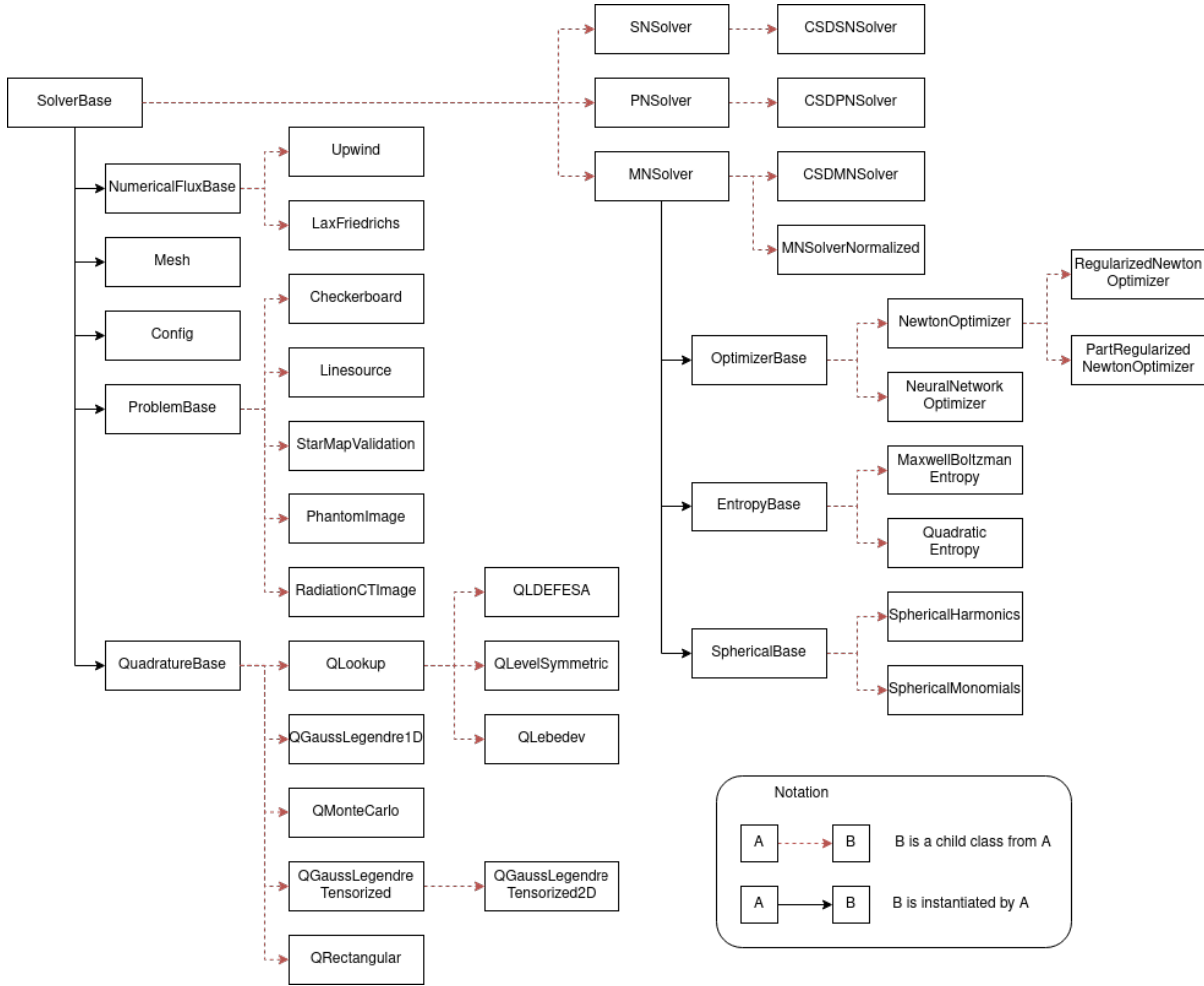


Fig. 1. Class and inheritance structure of the virtual SolverBase Class. Each instantiated Solver has class members and routines specific to its numerical structure.

been written to file. The file log carries the same information as the screen output in a tabular format. Lastly, the volume output consists of vtk files with solver and problem specific solution data. The output data can be specified in the solver configuration.

The method Solve() of the SolverBase class drives the execution of all derived solvers by iterating over the time discretization of the numerical methods described in Section 4. This main time-iteration is displayed in Algorithm 1. Each command is specified in the derived solver classes such as the P_N solver and does not induce any additional communication overhead for the parallelization architecture. The class PNSolver inherits from SolverBase and does not own additional instances of other custom building blocks and overwrites the sub-routines of Algorithm 1 for the P_N equation specific numerical method, which allows for runtime solver assembly. It's child class is the CSDPNSolver, which is the implementation of the P_N based continuous slowing down solver, that

Algorithm 1: Execution of the solver

```

Prepare outputs for Screen, Log and Volume
Solver specific pre-processing
for  $i = 1, \dots, n_{tf}$  do
  for  $l = 1, \dots, k_{rk}$  do
    Runge-Kutta pseudo iteration pre-processing
    Update numerical fluxes
    Perform finite volume step
    Update Runge-Kutta intermediate solution
  end
  Iteration post-processing
  Compute Runge-Kutta solution
  Write Screen, Log and Volume output as configured
end
Post-processing of output

```

overwrites the solver-preprocessing routines for the continuous slowing down specific energy transformation. The P_N based solvers produce radiation flux and moments as output.

The class SNSolver adapts the sub-routines of Algorithm 1 for the ordinate based numerical methods and is the parent class of the CSDNSolver. S_N based solvers produce the radiation flux as output.

Lastly, the MNSolver class contains the implementation of the M_N numerical method and holds the module SphericalBase, which controls the choice of basis functions $m(v)$ of the velocity space, the module EntropyBase, that controls the choice of entropy functional for the entropy closure and lastly the module OptimizerBase, which controls the choice of numerical optimizer used to compute the entropy closure. The class CSDMNSolver inherits from the MNSolver class and analogously overwrites the sub-routines of Algorithm 1 for the continuous-slowing down equations. M_N based solvers produce the radiation flux, moments and dual variable of the entropy closure as output.

5.2 Mesh Class

The mesh class handles the computational meshes of the spatial discretization of the underlying differential equation. It can handle 1D meshes and 2D unstructured triangular and quadrilateral meshes in the SU2 [55] mesh format. The mesh class keeps a record of all geometry and adjacency information required for the finite volume methods with first and second order fluxes.

5.3 Computational problem class

The problem class handles the setup of computational problems and test cases. It sets the initial conditions for the solution of the numerical solver and manages space, time or energy dependent material properties for the solver. The abstract class problem base holds pointers to the Mesh and Config classes and creates instances of specific problems depending on the chosen configurations. Each implemented problem has two child classes, one for the ordinate based and one for moment based solvers. The moment based problem classes compute the moments of the initial condition and sources of the corresponding kinetic densities specified in the ordinate based problem class.

The solver framework comes with a number of pre-implemented test cases and functionalities. This includes

standard 1D and 2D test cases such as line source and checkerboard for the radiative transfer solvers, as well as isotropic and directed sources with different background media which can be loaded from a user-supplied image file, for the continuous slowing down solvers. Custom test cases can be easily added by the user, based on the provided examples and our modular approach.

5.4 Quadrature Class

The virtual quadrature base class creates instances of specific numerical quadratures using its static factory method. The quadratures are intended to integrate over the velocity space of the Boltzmann equation, however, they can be applied to other use-cases as well. The implemented quadratures are distinguished by the dimension of the integrated velocity space and the integration area. Each quadrature has a specifiable order and manages the integration points in Cartesian and spherical coordinates as well as the corresponding quadrature weights. By default, the quadrature rules integrate over the unit sphere, but the integration region can be scaled.

5.5 IO/Use of Config Files

The KiT-RT solver is a command line interface based program and takes one argument, the configuration file. This file is parsed and the specified modules of the KiT-RT framework are arranged for a solver instance or another custom tool. The configuration file is a document containing option specifications of the form `CONFIG_OPTION=VALUE`. A solver configuration contains information about file input and output, where the location of the mesh file, the volume output files and the log files are specified. Then, the computational problem and the problem specific parameters, e.g. scattering coefficient, final time, spatial dimension and boundary conditions, are set. Next, the solver specific options are set. In the example of an M_N solver, the choice of velocity basis, the maximal degree of the basis functions, the CFL number, spatial integration order, entropy functional, optimizer, quadrature and quadrature order are set. Finally, quantities for screen, volume and log output are specified along with their output-frequency. Example configuration files for the numerical results of Section 7 can be found in the Github Repository <https://github.com/CSMMLab/KiT-RT>.

5.6 Practices of modern software development

The entire solver and associated documentation is put under the version control system git [15] to greatly enhance collaborative workflows. Additionally, the web-hosting service GitHub is used to provide global access to the code which is licensed under the open-source MIT license.

To further improve collaborations, the service also acts as a central host for progress and issue tracking, deployment and maintaining code integrity. The latter is obtained through automated testing in terms of unit test, which ensure the validity of smaller code instances such as functions or classes by testing predefined inputs against their expected result, or regression tests which validate the correctness of the solver as a whole based on small test problems and compare obtained results to reference solutions. These test are automatically executed every time code changes are submitted to the main development branches or if a merge request is opened.

If any of the automated tests fail for a new submit it is rejected for merging, ensuring code integrity and quality on the important development streams at all times. Combining the test information, we can further define metrics such as a test coverage describing the percentage of code lines validated by any form of testing and ultimately helps building trust in the code framework. For the KiT-RT framework the test coverage is reported to the coveralls.io service at <https://coveralls.io/github/CSMMLab/KiT-RT>. The KiT-RT framework features relatively modest software dependencies, but nevertheless being able to build and run the code correctly can be troublesome on many systems. To circumvent this issue, we provide a pre-configured build environment through the containerization engine Docker [51]. These so-called Docker containers have been developed for consistent software development and deployment and work as isolated instances with a minimal software stack

comparable to lightweight virtual machines. The respective specialized docker image is also publicly available at <https://hub.docker.com/r/kitrt/test>.

As also mentioned previously, GitHub can also be used for the deployment of precompiled software packages and the associated documentation. The documentation is automatically generated as part of a complete software build. It is written in the reStructuredText Markup language and uses the documentation framework Sphinx [11], which compiles the Markup files to a series of linked HTML files or in other words a local website. To make the website itself publicly available it is hosted by ReadTheDocs¹ service under the URL <https://kit-rt.readthedocs.io>.

With all these tools in mind, the development workflow can be described as follows:

Starting on GitHub, each developer can create a new branch based on the development branch or fork the entire KiT-RT framework to obtain a personal workspace. After the developers have added their changes, they can file a merge request that will automatically be tested by the continuous integration processes and a core developer will perform a code review of all changes. Provided all test succeed and the core developer is satisfied with the added/changed code quality, it will be merged into the development branch. If enough new features have been added into the development branch, it will be merged to the master branch and the software will obtain a new version number (major or minor).

6 PARALLEL SCALING

In the following we investigate the parallel performance of the three base solver implementations S_N , P_N and M_N , where we follow [53] for the brief review of parallel scalings. The speedup of a parallel algorithm is defined as

$$S(n, p) = \frac{T^*(n)}{T(n, p)}, \quad (50)$$

where $T^*(n)$ is the execution time of the best inherently serial algorithm with input size n and $T(n, p)$ the time for the parallel implementation with p processing workers and input size n . In general the best serial algorithm may be different than the parallel algorithm, however in our application case, the finite-volume discretization scheme does not change for serial implementation.

In theory the best possible speedup is linear [19], i.e., $S(n, p) = p$, thus the measure of parallel efficiency is

$$E(n, p) = \frac{S(n, p)}{p}. \quad (51)$$

In practise the speedup and parallel efficiency of an algorithm is limited by spawning and communication overhead of the parallel workers as well as the fraction of inherently serial code f , that exists in any algorithm. Thus the upper bound for the speedup is given by Amdahl [27]

$$S(n, p) \leq \frac{1}{f + (1 - f)/p} \quad (52)$$

For larger input sizes, the fraction of inherently serial code f typically decreases, which enables the use of highly parallel implementations. Two common approaches to measure the parallel performance are given by the strong and weak scaling approach, where the former describes an experiment, where for fixed input size n the amount of parallel workers p is increased and the their timing is measured, which directly results in the speedup of Eq. (50). The latter increases the input size n proportionally to the worker count p . A perfectly parallel algorithm would have a constant parallel time $T(n, p)$.

The philosophy of the parallel implementation of the solvers of this work are based on the independence of the performed computations of the grid cells. As Algorithm 1 displays, during one (pseudo) time iteration of a solver,

¹<https://readthedocs.org/>

a set of instructions are calculated. Each instruction can be carried out independently for each grid cell and only in between two instruction sets, communication between parallel workers need to be established. Therefore, the parallel implementation spawns a set of parallel workers with shared memory access and distributes the spatial grid among them to carry out the current instruction. The input size n is thus given by the number of grid cells of the spatial discretization. We perform a strong parallel scaling study for the implementation of the M_N , S_N and

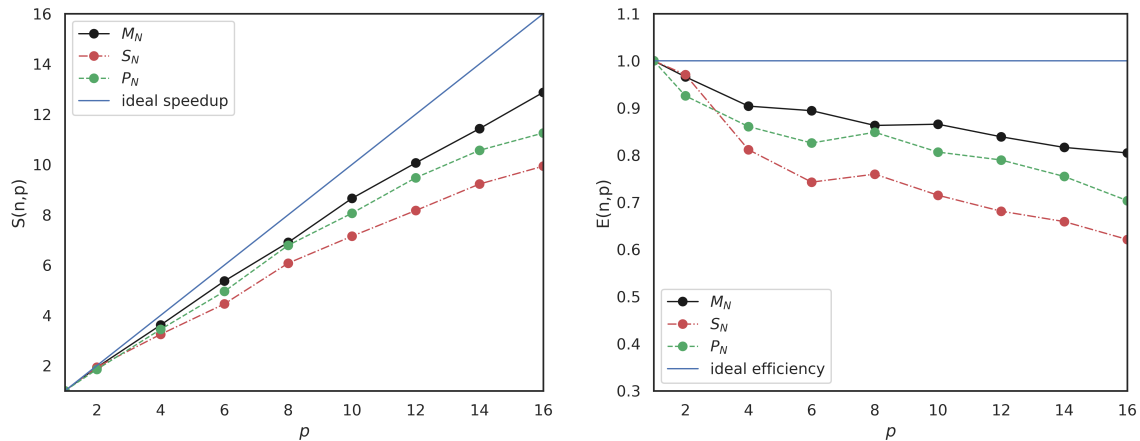


Fig. 2. Strong parallel scaling (left) and parallel efficiency (right) for the M_N , S_N and P_N solver

P_N solver on the Linesource test case, as described in Section 7.1, with a fixed unstructured triangular mesh of size $n = 578290$ and varying number of shared memory parallel workers p . We choose $p = 1, \dots, 16$, furthermore, the solvers allocated memory does not exceed the systems memory. Figure. 2 shows a comparison of the solvers parallel scalings and efficiency. It is apparent, that the M_N solver enjoys the highest speedup even for a high parallel worker count, while the P_N and S_N solver experience diminishing returns for more than $p = 12$ workers. The performance of the continuous slowing down solver implementations follows the corresponding base solver performance, since the same spatial, velocity and (pseudo) temporal discretizations are used.

7 VALIDATION

For validation and a comparison of the implemented solvers, we consider a selection of the test cases provided within the problem class (5.3). The S_N solver is validated with a comparison with Kinetic.jl [69, 70]. The continuous slowing down solvers are further compared to a reference Monte Carlo solution computed using TOPAS [56] as well as the validated spherical harmonics solver StaRMAP [59].

7.1 Inhomogeneous linesource

In the following, we compare the numerical results of our framework to a Monte Carlo solution computed using TOPAS [56] as well as the staggered-grid spherical harmonics solver StaRMAP [59]. The problem considered is an inhomogeneous linesource test case, which extends the classical linesource benchmark [22, 23] to a steady-state but energy dependent setting. As background density, a piece-wise constant function $\rho(\mathbf{x}) = 1 + 4 \cdot \mathbb{1}_{X_u}(\mathbf{x})$ for $\mathbf{x} \in [0, 1]^2$ is chosen. Here $\mathbb{1}_A : X \rightarrow \mathbb{R}$ denotes the indicator function for the set A . The upper part of the spatial domain on which we prescribe a reduced density is defined as $X_u = [0, 1] \times [0.56, 1]$. At a maximal energy of

$E_{\max} = 1$, a particle beam is positioned in the center of the spatial domain $\mathbf{x}_0 = \frac{1}{2}(1, 1)^T$, which is modelled as

$$\psi(E_{\max}, \mathbf{x}, \Omega) = \frac{1}{2\pi\sigma^2} \exp\left(-\frac{\|\mathbf{x} - \mathbf{x}_0\|^2}{2\sigma^2}\right)$$

$$\psi(E_{\max}, \mathbf{x}, \Omega) = 0 \quad \text{for } \mathbf{x} \in \partial X.$$

Here a standard deviation of $\sigma = 0.01$ is chosen to obtain a sharp particle beam in the center. The spatial grid for all deterministic methods is a structured rectangular grid with 300^2 cells. Due to the functionality of the Monte-Carlo software, we use a three-dimensional grid and project the x_3 -domain onto the $x_1 - x_2$ plane. To allow for feasible costs, the Monte-Carlo method uses a coarser grid resolution of 100 spatial cells per dimension and 100000 Monte-Carlo runs are computed to reduce statistical noise. The S_N solver uses a product quadrature rule of order 20 for the streaming step and spherical moments up to order 8 to compute scattering terms. Similarly, the P_N solver employs spherical moments up to order 8. The time step restriction of all deterministic methods picks a CFL number of 0.7. All methods are second order in space and first order in time.

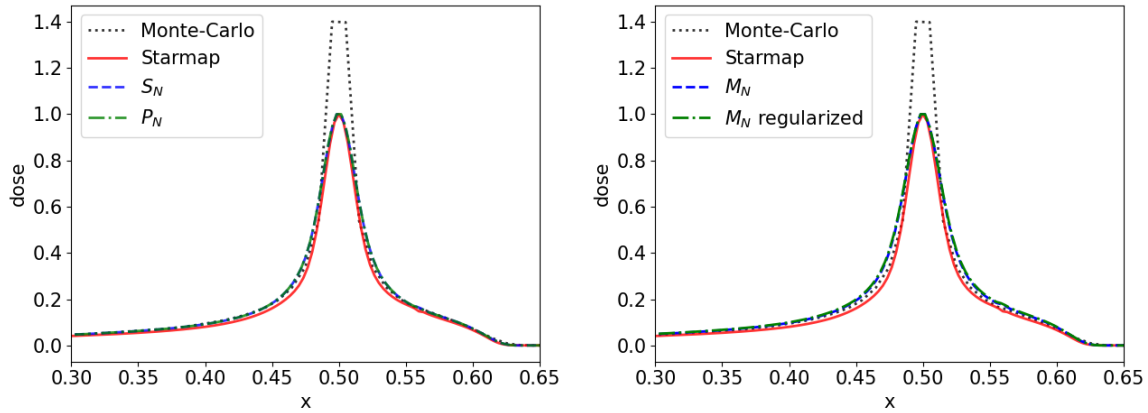


Fig. 3. Comparison of simulation results of deterministic and stochastic methods.

The resulting dose profile is plotted in Figure 3 along the x_2 -axis in the interval $x_2 \in [0.3, 0.65]$. It is observed that the dose drops in the transition between densities 1 and 5. All methods show a similar behaviour and (except for the center region) agree well with the Monte-Carlo results. Note that the increased Monte-Carlo dose results from the coarser mesh, leading to a coarser initial particle distribution. Moreover, it is observed that the regularized M_N method seems to coincide with its non-regularized counterpart.

7.2 Checkerboard

The checkerboard test case mimics a nuclear reactor block with a strong radiative source in the domain center, which is denoted by \mathbf{X}_q , and several highly absorptive regions \mathbf{X}_a placed in a checkerboard pattern it. The spatial layout of this two dimensional test case can be found in Figure 4, where the source region \mathbf{X}_q is marked orange and the absorption regions \mathbf{X}_a are marked black. The corresponding time dependent linear Boltzmann equation reads

$$\partial_t \psi + \Omega \nabla_{\mathbf{x}} \psi + \Sigma_t = \int_{\mathbb{S}^2} \Sigma_s(t, \mathbf{x}, \Omega, \Omega_*) \psi(\Omega_*) d\Omega_* + q(t, \mathbf{x}, \Omega) \quad (53)$$

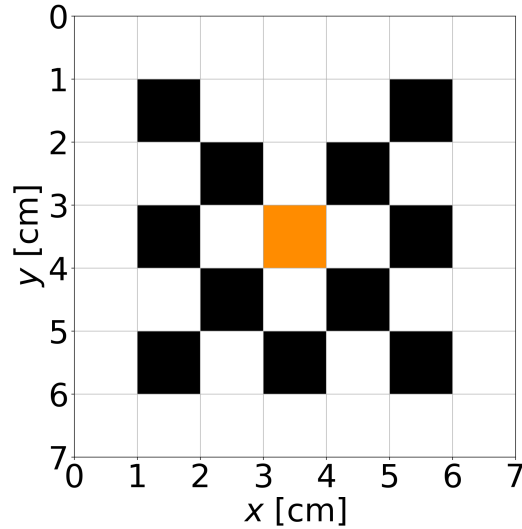


Fig. 4. Layout of the two dimensional checkerboard test case.

for $\mathbf{x} \in [0, 7]^2$, $t \in [0, 10)$ and $\Omega \in P_{\mathbb{R}^2}(\mathbb{S}^2)$. This corresponds to Eq. (9) with $\rho(\mathbf{x}) = 1$. We equip the equation with Dirichlet boundary conditions and initial condition

$$\psi(t, \mathbf{x}, \Omega) = 0, \quad \mathbf{x} \in \partial\mathbf{X} \quad (54)$$

$$\psi(t, \mathbf{x}, \Omega) = 0, \quad t = 0 \quad (55)$$

to obtain a well posed problem. Furthermore, the scattering kernel k and source term q are assumed to be isotropic and constant in time. The scattering cross and absorption cross sections are given by

$$\Sigma_s(\mathbf{x}) = \begin{cases} 0 & \mathbf{x} \in X_a \\ 1 & \text{else} \end{cases}, \quad \Sigma_t(\mathbf{x}) = \begin{cases} 10 & \mathbf{x} \in X_a \\ 1 & \text{else} \end{cases}, \quad (56)$$

and the isotropic source is given by

$$q(\mathbf{x}, \Omega, t) = \begin{cases} 1 & \mathbf{x} \in X_q \\ 0 & \text{else} \end{cases}. \quad (57)$$

We create a unstructured triangular mesh with 25000 cells to discretize the spatial domain with regard to the absorption and source regions, such that the region boundaries coincide with the mesh faces. The simulation is computed until final time $t_f = 10$ using various solver configurations. All employed solvers use a second order upwind flux as the spatial discretization and a second order explicit Runge Kutta scheme for temporal discretization with CFL number equals 0.45, since M_N solvers with non-regularized entropy closure require a CFL number smaller than 0.5 for stability [5, 36, 38]. The solution computed at final time $t_f = 10$ is displayed in Fig. 5, where we can see the scalar flux

$$\Psi(\mathbf{x}, t) = \int_{\mathbb{S}} \psi \, d\Omega, \quad (58)$$

in the contour plot. The radiation flux is highest at the source region X_q and almost zero in the absorption region X_a for all solvers. Towards the top of the domain, the radiation travels freely, whereas towards the left, right and bottom, the radiation expansion is damped by absorption regions. Figure 5 shows the solution computed

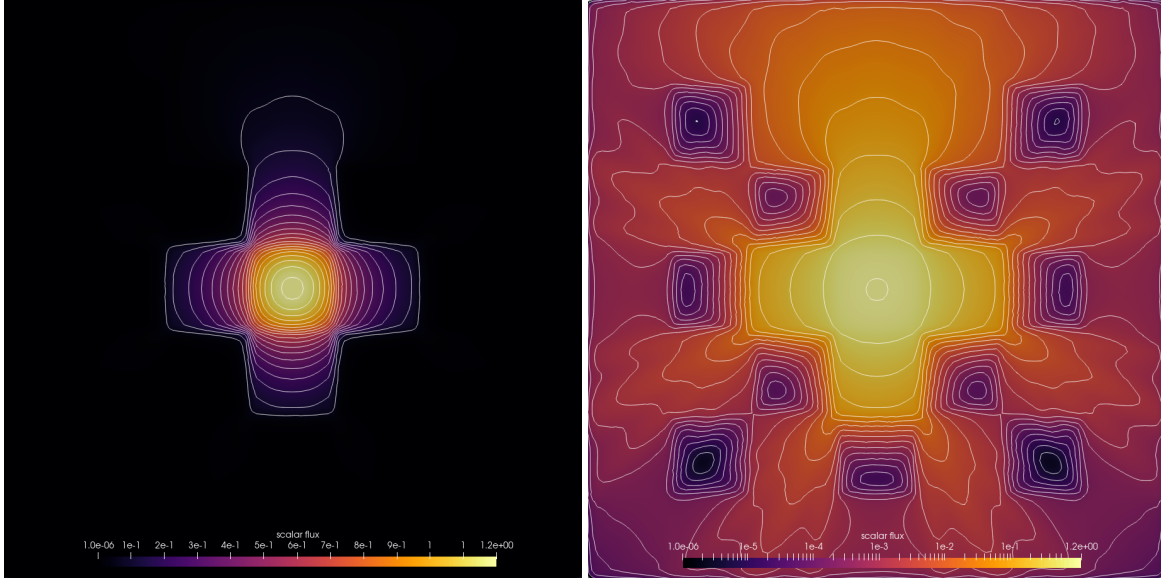


Fig. 5. Simulation results for the S_{10} solver in linear scale and log scale.

with the S_{10} solver with an order 10 tensorized Gauss Legendre Quadrature. On the logarithmic scale plot, one can see ray effects right outside the chokepoints between the absorption regions, which are typical artifacts for S_N [12]. We validate the S_{10} solver against the implementation of the kinetic.jl framework [70], and display the vertical cross section of the KiT-RT and kinetic.jl bases solution at final time t_f in Fig. 6 using the same triangular mesh. We can see, that the deviation between implementations is below the $1e-3$, which is the characteristic length of a grid cell.

Figure 7 displays the solution computed with the P_5 solver using an order 5 Spherical Harmonics basis. Compared to the S_{10} solution, there are no ray effects visible at the chokepoints between absorption regions.

We show now the solution for different entropy based moment, i.e., M_N solvers, that use an order 10 tensorized Gauss Legendre quadrature to evaluate the kinetic flux. Figure 8 displays a M_3 solution using an order 3 spherical harmonics moment basis computed with a Newton optimizer with line-search configured to accuracy $1e-7$. The same solver configuration is displayed in Figure 9, where the entropy closure is replaced by the partial regularized minimal entropy problem of Eq. (19) with the regularizer γ set to $1e-3$ and a non-regularized moment of order zero. While on the linear scale plots, both solutions are similar, on the log scale plots, one can see that the regularized M_N solver computes much lower values within the absorption regions. Furthermore, the log plot shows that the regularized M_N solution oscillates slightly in regions with small scalar fluxes. Figure 10 shows a direct comparison of a vertical cross section using all four presented solvers. In the vast majority of the spatial domain, the solutions correspond well, with the exception of absorption regions, i.e., $x \in [1, 2]$, where the M_3 solution computes slightly bigger values and the regularized M_3 solution captures the absorption region slightly better than the reminding solvers.

Next we compare a non-regularized M_1 solver using a monomial basis of order 1 and a Newton solver based

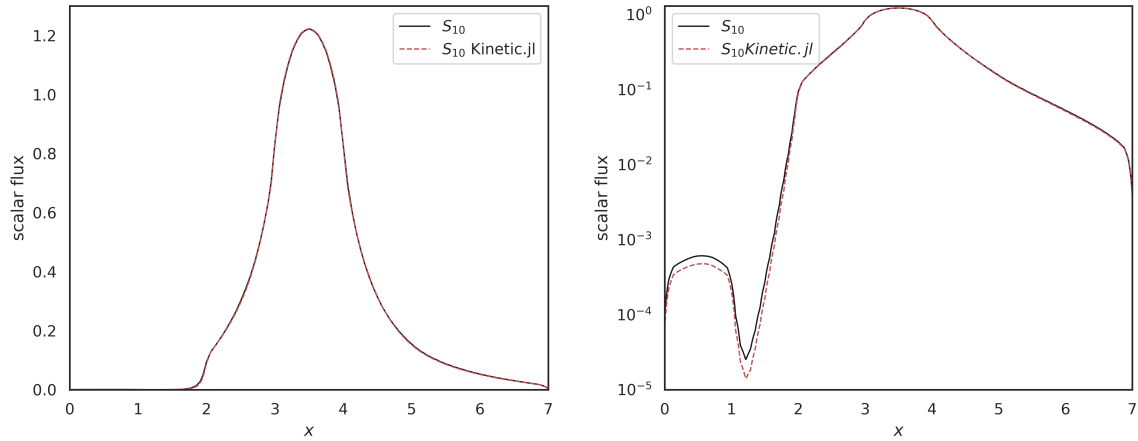


Fig. 6. Vertical cross section through the solution of the S_{10} method. Comparison of the KiT-RT and kinetic.jl packages.

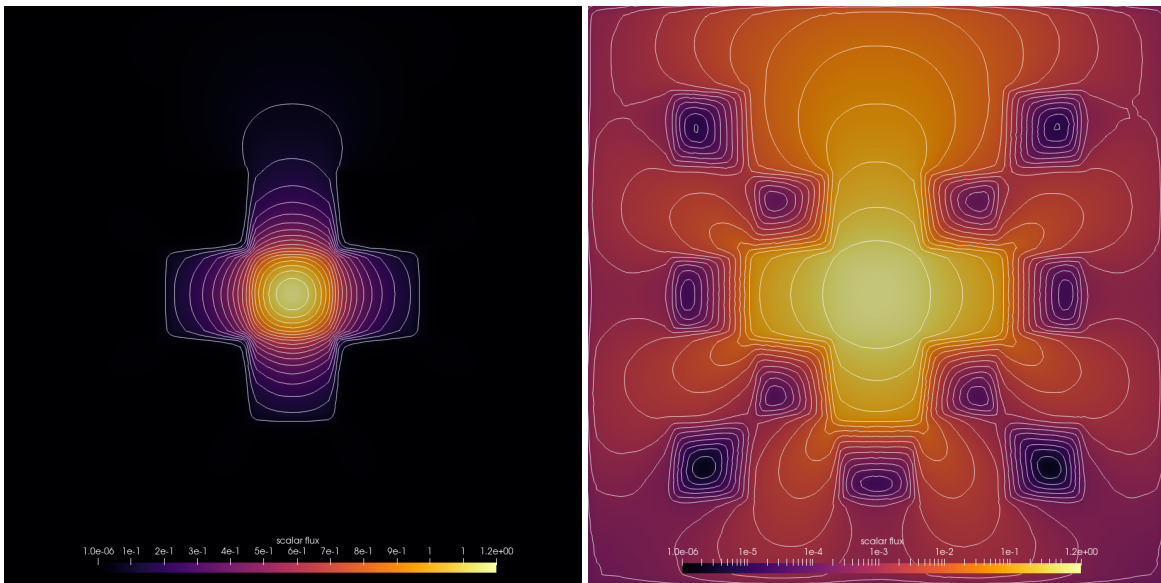


Fig. 7. Simulation results for the P_5 solver in linear scale and log scale.

entropy computation, shown in Fig. 11, with a similar solver configuration, which employs a neural network to compute the entropy closure, shown in Fig 12. Various data driven entropy closures have been introduced by [57, 58]. In the KiT-RT package we have compiled the trained networks by [58] to C++ and have implement a fast tensorflow [1] backend for seamless integration into the KiT-RT framework. The neural network used in the simulation of Fig 12 is an input-convex architecture. Both, the Newton based and neural network based solution correspond well. As Fig. 13 shows, only in at the top of the domain, i.e., at $x \approx 6.5$ in the cross section plot, we see a small deviation between the methods. In this region at the wave-front of the radiative transport, the moments

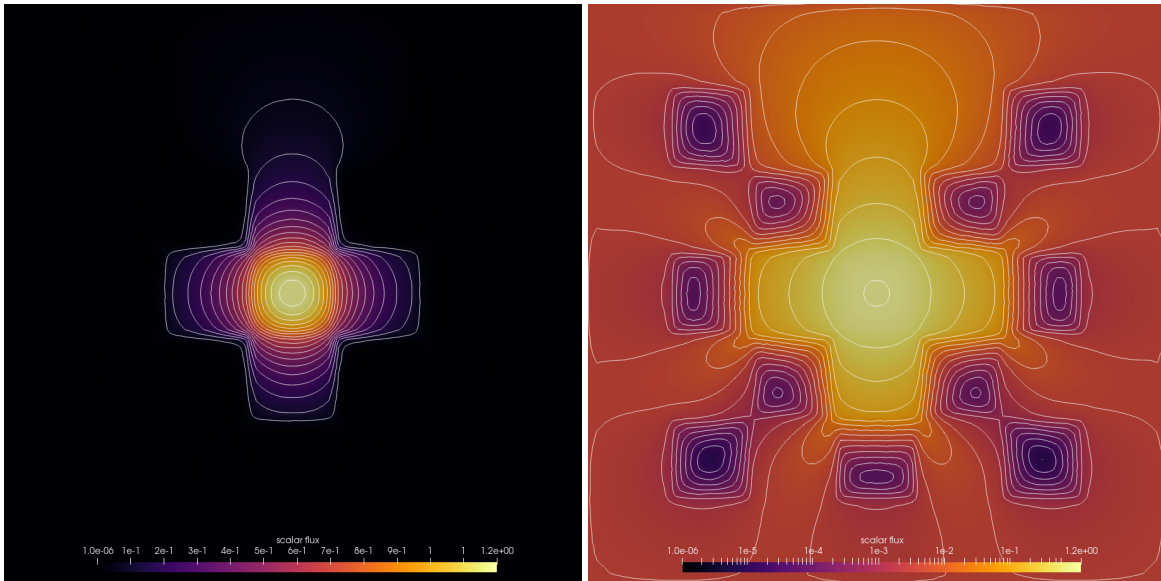


Fig. 8. Simulation results for the M_3 solver with spherical harmonics basis in linear scale and log scale.

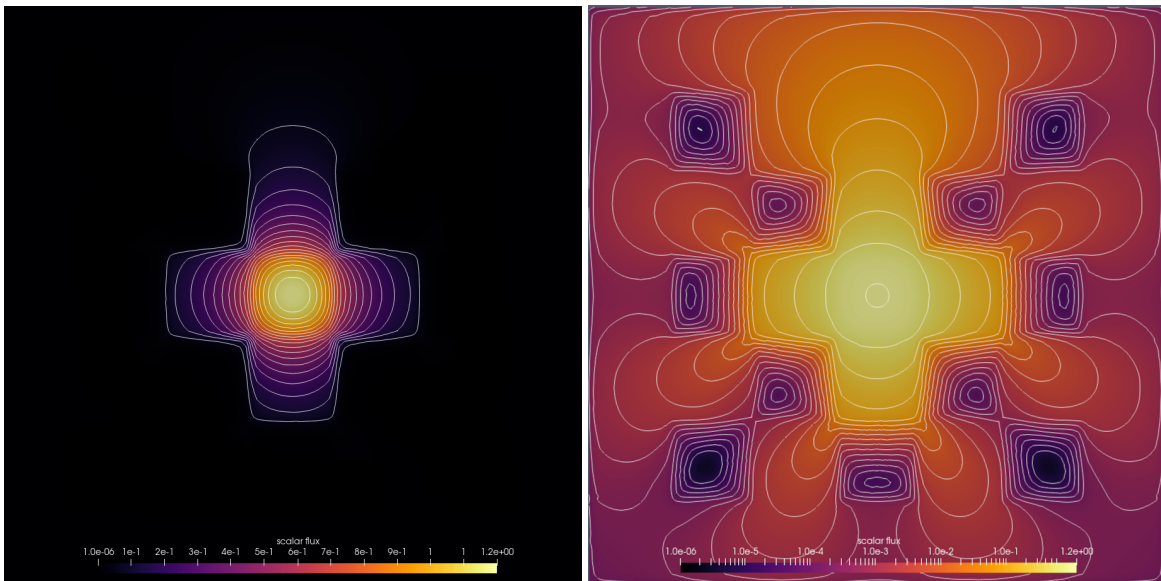


Fig. 9. Simulation results for the M_3 solver with spherical harmonics basis and regularized entropy in linear scale and log scale.

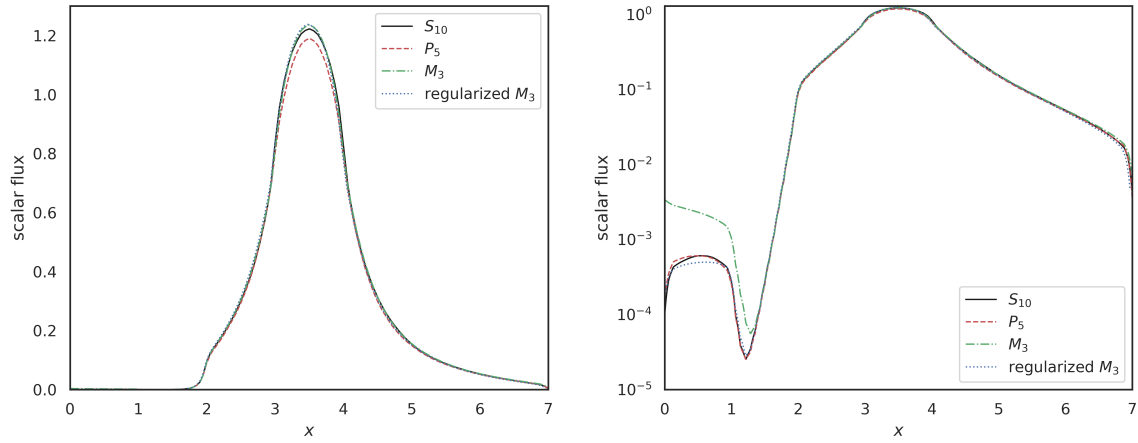


Fig. 10. Vertical cross section through the solution of the checkerboard test case. Comparison of the S_{10} , P_5 , M_3 and regularized M_3 solver .

of the kinetic equation are close to the boundary of the realizable set, where on the one hand, the Newton based solver needs more iterations and thus more wall time to compute the solution to the optimization problem, and on the other hand, the neural network accuracy declines.

Neural network based entropy closures are constructed to accelerate the time consuming M_N method. In the

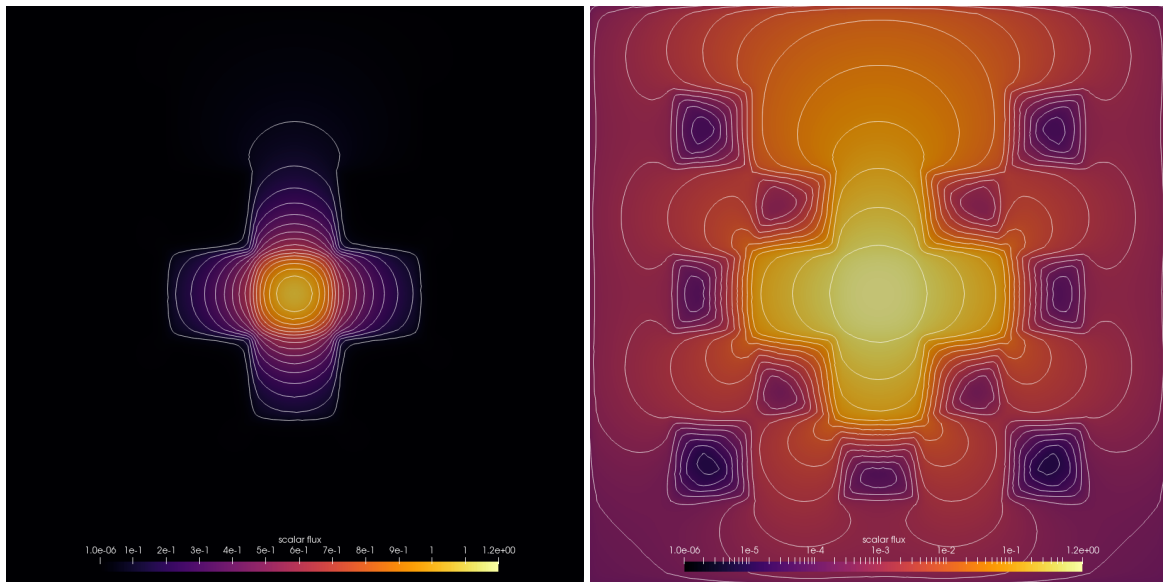


Fig. 11. Simulation results for the M_1 solver with monomial basis in linear scale and log scale.

following we validate the speedup through the neural network entropy closure using a larger mesh of 700000 grid

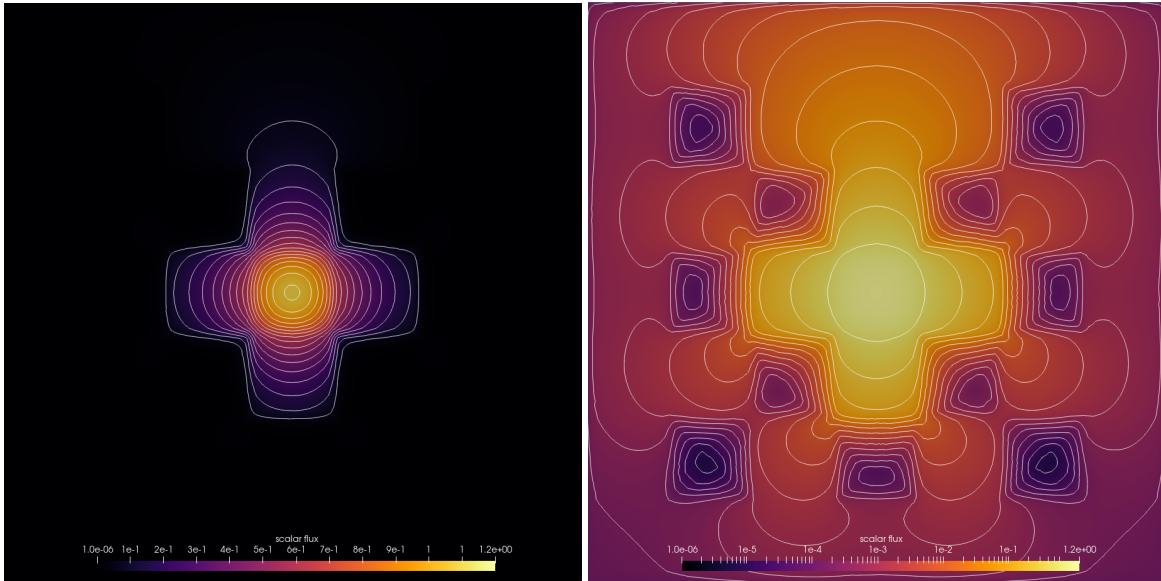


Fig. 12. Simulation results for the M_1 solver with monomial basis and neural network based entropy computation in linear scale and log scale.

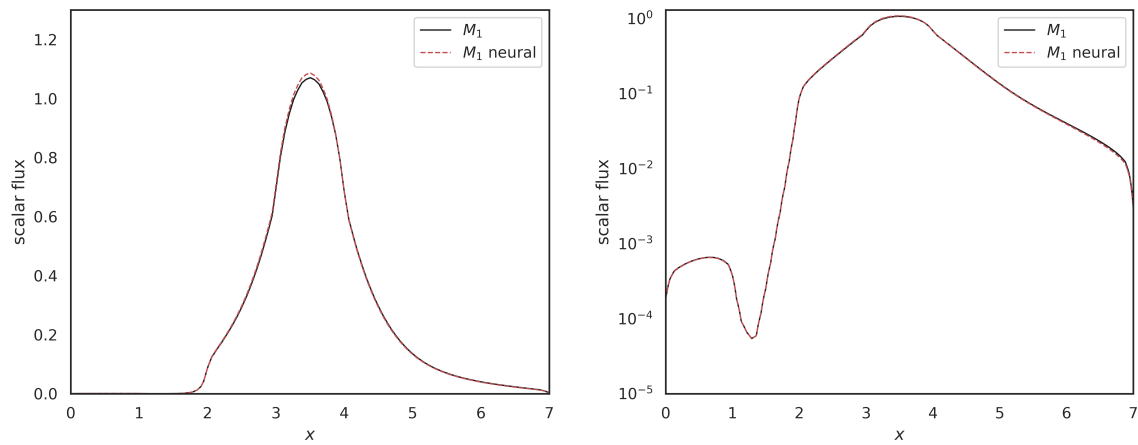


Fig. 13. Cross section comparison of the Checkerboard test case using the M_1 and neural network based M_1 method.

cells, since in [58] it is shown, that the due to data transformation to tensorflow tensors, the speedup is faster for higher data-set sizes. The time consumption of the M_1 and neural network based M_1 solver is illustrated in Table 1, where one can see that the neural network based closure accelerates the computational time by 89.33-87.01%.

Table 1. Timings of the neural network based M_1 solver compared to the Newton based M_1 solver

	M_1 Newton	M_1 neural network	runtime reduction
4 cores	757.88979	80.810305	89.33%
12 cores	258.6485	33.606152	87.01%

7.3 Beam in 2D patient CT

Having validated the CSD solvers against StarMAP and a Monte Carlo framework in section 7.1, we now examine a realistic 2D CT scan of a lung patient as a proof of concept for the application of our framework to radiation therapy computations. The patient data was retrieved from an open source data set [47] in The Cancer Imaging Archive (TCIA) [16]. The patient is irradiated with an electron beam of $E_{\max} = 20$ MeV. We model this beam as the initial condition

$$\psi(E_{\max}, \mathbf{x}, \Omega) = \frac{1}{(2\pi)^{3/2} \sigma_{\Omega_2} \sigma_x \sigma_y} \cdot \exp(-(\mu_{\Omega_2} - \Omega_2)^2 / 2\sigma_{\Omega_2}^2) \cdot \exp(-(\mu_{x_1} - x_1)^2 / 2\sigma_{x_1}^2) \cdot \exp(-(\mu_{x_2} - x_2)^2 / 2\sigma_{x_2}^2),$$

where $(\mu_{x_1}, \mu_{x_2}) = (2.5\text{cm}, 5.8\text{cm})$ is the beam position within the $6\text{cm} \times 6\text{cm}$ domain and $\mu_{\Omega_2} = \frac{\pi}{2}$ rad is the beam direction. The remaining parameters are chosen as $\sigma_{x_1} = \sigma_{x_2} = \sigma_{\Omega_2} = 0.1$. To determine a tissue density ρ for given gray-scale values of the CT image, we set the maximum density, represented by white pixels, to the density of bone $\rho_{\text{bone}} = 1.85 \text{ g/cm}^3$. The remaining tissue is scaled such that the minimum pixel value of zero corresponds to a minimal density of $\rho_{\min} = 0.05 \text{ g/cm}^3$. This corresponds approximately to the lower bound of observed lung densities [34].

Figure 14 compares the normalized dose for a CSD S_{13} , P_{13} and M_5 solver. While all methods show similar behaviour and are able to capture the effects of heterogeneities in the patient density, some differences e.g. in the maximum depth of the S_{13} solution compared to P_{13} and M_5 or the shape of the lowest two isolines can be observed. The cross sections in figure 15 further show that the S_{13} dose has a lower maximum and higher minimum value than the M_5 and to a lesser extent also P_{13} solutions.

8 CONCLUSION

In this work, we have presented a collection of deterministic transport solvers for radiation therapy applications. The methods agree well with results obtained with conventional radiation therapy codes. Due to the use of polymorphism, we are able to guarantee a straight forward extension to further numerical methods, which facilitates the investigation of novel radiation therapy solvers and their comparison to conventional methods.

ACKNOWLEDGEMENTS

All authors of this work have contributed equally to this project. The authors would like to thank Thomas Camminady for his help with implementing spherical quadrature rules. Jonas Kusch has been funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) – 491976834. Pia Stammer is supported by the Helmholtz Association under the joint research school HIDSS4Health – Helmholtz Information and Data Science School for Health. The work of Steffen Schotthöfer is funded by the Priority Programme "Theoretical Foundations of Deep Learning (SPP2298)" by the Deutsche Forschungsgemeinschaft.

REFERENCES

- [1] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas,

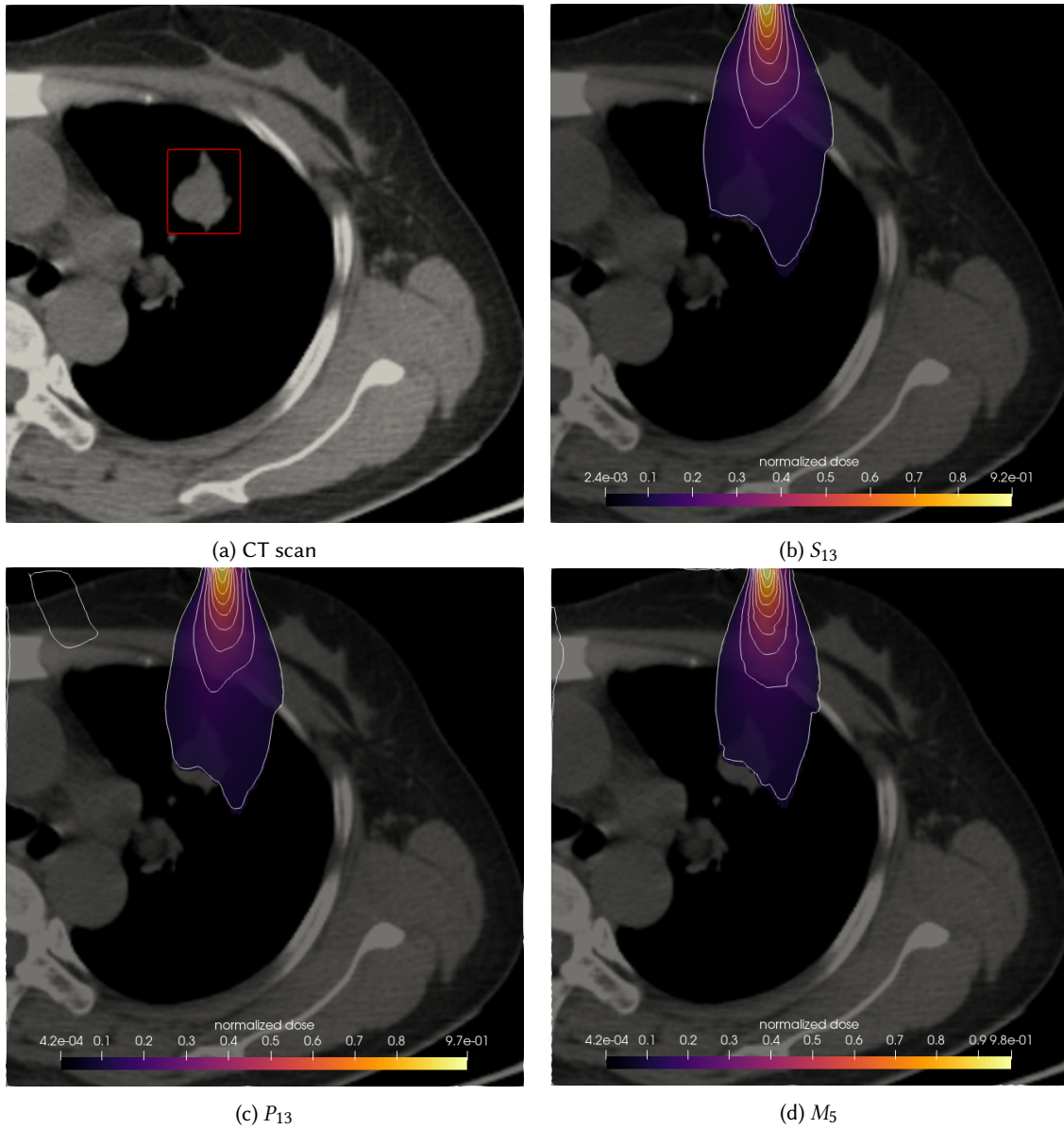


Fig. 14. Patient CT scan with lung tumor (red box) (a) as well as corresponding simulation results for the S_{13} , P_{13} and M_5 solver with spherical harmonics basis and partially regularized entropy.

Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. 2015. TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems. <https://www.tensorflow.org/> Software available from tensorflow.org.

[2] IK Abu-Shumays. 2001. Angular quadratures for improved transport computations. *Transport Theory and Statistical Physics* 30, 2-3 (2001), 169–204.

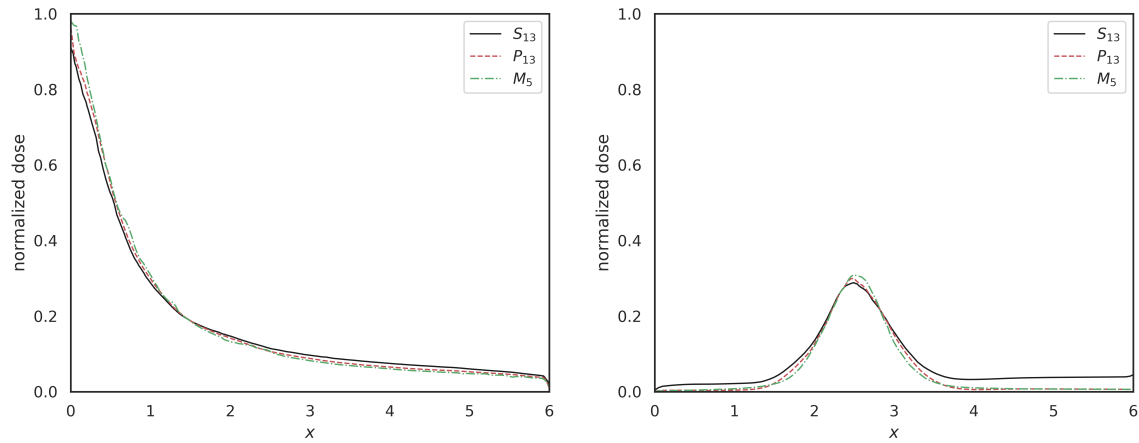


Fig. 15. Vertical (at $x = 2.5\text{cm}$) and horizontal (at $y = 5\text{cm}$) cross section through the normalized dose in the patient CT. Comparison of the S_{13} , P_{13} and partially regularized M_5 solver.

- [3] Michael Aftosmis, Datta Gaitonde, and T. Sean Tavares. 1995. Behavior of linear reconstruction techniques on unstructured meshes. *AIAA Journal* 33, 11 (1995), 2038–2049. <https://doi.org/10.2514/3.12945> arXiv:<https://doi.org/10.2514/3.12945>
- [4] Graham W. Alldredge, Martin Frank, and Cory D. Hauck. 2018. A regularized entropy-based moment method for kinetic equations. arXiv:1804.05447 [math.NA]
- [5] Graham W. Alldredge, Cory D. Hauck, and André L. Tits. 2012. High-Order Entropy-Based Closures for Linear Transport in Slab Geometry II: A Computational Study of the Optimization Problem. *SIAM Journal on Scientific Computing* 34, 4 (2012), B361–B391. <https://doi.org/10.1137/11084772X> arXiv:<https://doi.org/10.1137/11084772X>
- [6] Brandon Amos, Lei Xu, and J. Zico Kolter. 2016. Input Convex Neural Networks. *CoRR* abs/1609.07152 (2016). arXiv:1609.07152 <http://arxiv.org/abs/1609.07152>
- [7] P. Andreo. 1991. Monte Carlo techniques in medical radiation physics. 36, 7 (jul 1991), 861–920. <https://doi.org/10.1088/0031-9155/36/7/001>
- [8] Richard Barnard, Martin Frank, and Michael Herty. 2012. Optimal radiotherapy treatment planning using minimum entropy models. *Appl. Math. Comput.* 219, 5 (2012), 2668–2679.
- [9] TIMOTHY BARTH and DENNIS JESPERSEN. [n. d.]. *The design and application of upwind schemes on unstructured meshes*. <https://doi.org/10.2514/6.1989-366> arXiv:<https://arc.aiaa.org/doi/pdf/10.2514/6.1989-366>
- [10] Christoph Börgers. 1998. Complexity of Monte Carlo and deterministic dose-calculation methods. *Physics in Medicine & Biology* 43, 3 (1998), 517.
- [11] Georg Brandl. 2021. Sphinx documentation. URL <http://sphinx-doc.org/> (2021).
- [12] Thomas Camminady, Martin Frank, Kerstin Küpper, and Jonas Kusch. 2019. Ray effect mitigation for the discrete ordinates method through quadrature rotation. *J. Comput. Phys.* 382 (2019), 105–123.
- [13] Thomas Camminady, Martin Frank, and Jonas Kusch. 2021. Highly uniform quadrature sets for the discrete ordinates method. *Nuclear Science and Engineering* (2021).
- [14] Kenneth M Case and Paul Frederick Zweifel. 1967. *Linear transport theory*. (1967).
- [15] Scott Chacon and Ben Straub. 2014. *Pro git*. Springer Nature.
- [16] K Clark, B Vendt, K Smith, J Freymann, J Kirby, P Koppel, S Moore, S Phillips, D Maffitt, M Pringle, L Tarbox, and F Prior. 2013. The Cancer Imaging Archive (TCIA): Maintaining and Operating a Public Information Repository. *Journal of Digital Imaging* 26, 6 (2013), 1045–1057. <https://doi.org/10.1007/s10278-013-9622-7>
- [17] Roland Duclous, Bruno Dubroca, and Martin Frank. 2010. A deterministic partial differential equation model for dose calculation in electron radiotherapy. *Physics in Medicine & Biology* 55, 13 (2010), 3843.
- [18] Leonard Eyges. 1948. Multiple scattering with energy loss. *Physical Review* 74, 10 (1948), 1534.
- [19] V. Faber, Olaf M. Lubeck, and Andrew B. White. 1986. Superlinear speedup of an efficient sequential algorithm is not possible. *Parallel Comput.* 3, 3 (1986), 259–260. [https://doi.org/10.1016/0167-8191\(86\)90024-4](https://doi.org/10.1016/0167-8191(86)90024-4)

- [20] Matthias Fippel and Martin Soukup. 2004. A Monte Carlo dose calculation algorithm for proton therapy. *Medical physics* 31, 8 (2004), 2263–2273.
- [21] Martin Frank, Jonas Kusch, Thomas Camminady, and Cory D Hauck. 2020. Ray effect mitigation for the discrete ordinates method using artificial scattering. *Nuclear Science and Engineering* 194, 11 (2020), 971–988.
- [22] BD Ganapol. 1999. Homogeneous infinite media time-dependent analytic benchmarks for X-TM transport methods development. *Los Alamos National Laboratory* (1999).
- [23] BD Ganapol. 2008. Analytical benchmarks for nuclear engineering applications. *Case Studies in Neutron Transport Theory* (2008).
- [24] C. Kristopher Garrett and Cory D. Hauck. 2013. A Comparison of Moment Closures for Linear Kinetic Transport Equations: The Line Source Benchmark. *Transport Theory and Statistical Physics* 42, 6-7 (2013), 203–235.
- [25] Kent A Gifford, John L Horton, Todd A Wareing, Gregory Failla, and Firas Mourtada. 2006. Comparison of a finite-element multigroup discrete-ordinates code with Monte Carlo for radiotherapy calculations. *Physics in Medicine & Biology* 51, 9 (2006), 2253.
- [26] V Grégoire and TR Mackie. 2011. State of the art on dose prescription, reporting and recording in Intensity-Modulated Radiation Therapy (ICRU report No. 83). *Cancer/Radiothérapie* 15, 6-7 (2011), 555–559.
- [27] John L. Gustafson. 2011. *Amdahl's Law*. Springer US, Boston, MA, 53–60. https://doi.org/10.1007/978-0-387-09766-4_77
- [28] Hartmut Hensel, Rodrigo Iza-Teran, and Norbert Siedow. 2006. Deterministic model for dose calculation in photon radiotherapy. *Physics in Medicine & Biology* 51, 3 (2006), 675.
- [29] Kenneth R Hogstrom, Michael D Mills, and Peter R Almond. 1981. Electron beam dose calculations. *Physics in Medicine & Biology* 26, 3 (1981), 445.
- [30] Mi Huang. 2015. *Application of deterministic 3D SN transport driven dose kernel methods for out-of-field dose assessments in clinical megavoltage radiation therapy*. Ph.D. Dissertation. Georgia Institute of Technology.
- [31] J.J. Jarrel and M.L. Adams. 2011. Discrete-ordinates quadrature sets based on linear discontinuous finite elements. *Proc. International Conference on Mathematics and Computational Methods applied to Nuclear Science and Engineering* (2011).
- [32] Xun Jia, Jan Schümann, Harald Paganetti, and Steve B Jiang. 2012. GPU-based fast Monte Carlo dose calculation for proton therapy. *Physics in Medicine & Biology* 57, 23 (2012), 7783.
- [33] O. Koch and C. Lubich. 2007. Dynamical low-rank approximation. *SIAM J. Matrix Anal. Appl.* 29, 2 (2007), 434–454. <https://doi.org/10.1137/050639703>
- [34] Ehiichi Kohda and Naoyuki Shigematsu. 1989. Measurement of lung density by computed tomography: implication for radiotherapy. *The Keio Journal of Medicine* 38, 4 (1989), 454–463.
- [35] Thomas Krieger and Otto A Sauer. 2005. Monte Carlo- versus pencil-beam-/collapsed-cone-dose calculation in a heterogeneous multi-layer phantom. 50, 5 (feb 2005), 859–868. <https://doi.org/10.1088/0031-9155/50/5/010>
- [36] C. Kristopher Garrett, Cory Hauck, and Judith Hill. 2015. Optimization and large scale computation of an entropy-based moment closure. *J. Comput. Phys.* 302 (2015), 573 – 590.
- [37] Kerstin Kuepper. 2016. *Models, numerical methods, and uncertainty quantification for radiation therapy*. Ph.D. Dissertation. Universitätsbibliothek der RWTH Aachen.
- [38] Jonas Kusch, Graham W Alldredge, and Martin Frank. 2019. Maximum-principle-satisfying second-order intrusive polynomial moment scheme. *The SMAI journal of computational mathematics* 5 (2019), 23–51.
- [39] Jonas Kusch and Pia Stammer. 2021. A robust collision source method for rank adaptive dynamical low-rank approximation in radiation therapy. *arXiv preprint arXiv:2111.07160* (2021).
- [40] Edward W Larsen, Moyed M Miften, Benedick A Fraass, and Ian AD Bruinvis. 1997. Electron dose calculations using the method of moments. *Medical physics* 24, 1 (1997), 111–125.
- [41] Kaye D Lathrop. 1968. Ray effects in discrete ordinates equations. *Nuclear Science and Engineering* 32, 3 (1968), 357–369.
- [42] Kaye D Lathrop. 1971. Remedies for ray effects. *Nuclear Science and Engineering* 45, 3 (1971), 255–268.
- [43] Randall J. LeVeque. 1992. *Numerical methods for conservation laws (2. ed.)*. Birkhäuser. 1–214 pages.
- [44] C. Levermore. 1996. Moment closure hierarchies for kinetic theories. *Journal of Statistical Physics* 83 (1996), 1021–1065.
- [45] C. David Levermore. 1997. Entropy-based moment closures for kinetic equations. *Transport Theory and Statistical Physics* 26, 4-5 (1997), 591–606.
- [46] Elmer Eugene Lewis and Warren F Miller. 1984. Computational methods of neutron transport. (1984).
- [47] P Li, S Wang, T Li, J Lu, Y Huang Fu, and D Wang. 2020. A Large-Scale CT and PET/CT Dataset for Lung Cancer Diagnosis. *The Cancer Imaging Archive* (2020). <https://doi.org/10.7937/TCIA.2020.NNC2-0461>
- [48] Gianluca Longoni. 2004. *Advanced quadrature sets and acceleration and preconditioning techniques for the discrete ordinates method in parallel computing environments*. Ph.D. Dissertation. University of Florida.
- [49] G Marchuk and V I Lebedev. 1986. Numerical methods in the theory of neutron transport. (1 1986). <https://www.osti.gov/biblio/7057084>
- [50] Kirk A Mathews. 1999. On the propagation of rays in discrete ordinates. *Nuclear science and engineering* 132, 2 (1999), 155–180.
- [51] Dirk Merkel. 2014. Docker: lightweight linux containers for consistent development and deployment. *Linux journal* 2014, 239 (2014), 2.

- [52] JE Morel, TA Wareing, RB Lowrie, and DK Parsons. 2003. Analysis of ray-effect mitigation techniques. *Nuclear science and engineering* 144, 1 (2003), 1–22.
- [53] Kenneth Moreland and Ron Oldfield. 2015. Formal Metrics for Large-Scale Parallel Performance. In *High Performance Computing*, Julian M. Kunkel and Thomas Ludwig (Eds.). Springer International Publishing, Cham, 488–496.
- [54] Edgar Olbrant and Martin Frank. 2010. Generalized Fokker–Planck theory for electron and photon transport in biological tissues: application to radiotherapy. *Computational and mathematical methods in medicine* 11, 4 (2010), 313–339.
- [55] Francisco Palacios, Juan Alonso, Karthikeyan Duraisamy, Michael Colonna, Jason Hicken, Aniket Aranake, Alejandro Campos, Sean Copeland, Thomas Economon, Amrita Lonkar, Trent Lukaczyk, and Thomas Taylor. [n. d.]. *Stanford University Unstructured (SU<sup>2</sup>): An open-source integrated computational environment for multi-physics simulation and design*. <https://doi.org/10.2514/6.2013-287> arXiv:<https://arc.aiaa.org/doi/pdf/10.2514/6.2013-287>
- [56] J. Perl, J. Shin, J. Schumann, B. Faddegon, and H. Paganetti. 2012. TOPAS: An Innovative Proton Monte Carlo Platform for Research and Clinical Applications. *Medical Physics* 39, 11 (Nov. 2012), 6818–6837. <https://doi.org/10.1118/1.4758060>
- [57] William A. Porteous, M. Paul Lau, and Cory D. Hauck. 2021. Data-driven, structure-preserving approximations to entropy-based moment closures for kinetic equations. arXiv:2106.08973 [math.NA]
- [58] Steffen Schotthöfer, Tianbai Xiao, Martin Frank, and Cory D. Hauck. 2022. Neural network-based, structure-preserving entropy closures for the Boltzmann moment system. <https://doi.org/10.48550/ARXIV.2201.10364>
- [59] Benjamin Seibold and Martin Frank. 2014. StaRMAP—A Second Order Staggered Grid Method for Spherical Harmonics Moment Equations of Radiative Transfer. *ACM Trans. Math. Softw.* 41, 1, Article 4 (oct 2014), 28 pages. <https://doi.org/10.1145/2590808>
- [60] John Tencer. 2016. Ray effect mitigation through reference frame rotation. *Journal of Heat Transfer* 138, 11 (2016).
- [61] Jouko Tervo and Pekka Kolmonen. 2002. Inverse radiotherapy treatment planning model applying Boltzmann-transport equation. *Mathematical Models and Methods in Applied Sciences* 12, 01 (2002), 109–141.
- [62] J Tervo, P Kolmonen, M Vauhkonen, LM Heikkinen, and JP Kaipio. 1999. A finite-element model of electron transport in radiation therapy and a related inverse problem. *Inverse Problems* 15, 5 (1999), 1345.
- [63] J Tervo, M Vauhkonen, and E Boman. 2008. Optimal control model for radiation therapy inverse planning applying the Boltzmann transport equation. *Linear Algebra Appl.* 428, 5-6 (2008), 1230–1249.
- [64] Oleg N Vassiliev, Todd A Wareing, Ian M Davis, John McGhee, Douglas Barnett, John L Horton, Kent Gifford, Gregory Failla, Uwe Titt, and Firas Mourtada. 2008. Feasibility of a multigroup deterministic solution method for three-dimensional radiotherapy dose calculations. *International Journal of Radiation Oncology* Biology* Physics* 72, 1 (2008), 220–227.
- [65] Oleg N Vassiliev, Todd A Wareing, John McGhee, Gregory Failla, Mohammad R Salehpour, and Firas Mourtada. 2010. Validation of a new grid-based Boltzmann equation solver for dose calculation in radiotherapy with photon beams. *Physics in Medicine and Biology* 55, 3 (jan 2010), 581–598. <https://doi.org/10.1088/0031-9155/55/3/002>
- [66] V. VENKATAKRISHNAN. [n. d.]. *On the accuracy of limiters and convergence to steady state solutions*. <https://doi.org/10.2514/6.1993-880> arXiv:<https://arc.aiaa.org/doi/pdf/10.2514/6.1993-880>
- [67] Hans-Peter Wieser, Eduardo Cisternas, Niklas Wahl, Silke Ulrich, Alexander Stadler, Henning Mescher, Lucas-Raphael Müller, Thomas Klinge, Hubert Gabrys, Lucas Burigo, et al. 2017. Development of the open-source dose calculation and optimization toolkit matRad. *Medical physics* 44, 6 (2017), 2556–2568.
- [68] M. K. Woo and J. R. Cunningham. 1990. The validity of the density scaling method in primary electron transport for photon and electron beams. *Medical Physics* 17, 2 (1990), 187–194. <https://doi.org/10.1118/1.596497> arXiv:<https://aapm.onlinelibrary.wiley.com/doi/pdf/10.1118/1.596497>
- [69] Tianbai Xiao. 2021. A flux reconstruction kinetic scheme for the Boltzmann equation. *J. Comput. Phys.* 447 (2021), 110689.
- [70] Tianbai Xiao. 2021. Kinetic.jl: A portable finite volume toolbox for scientific and neural computing. *Journal of Open Source Software* 6, 62 (2021), 3060. <https://doi.org/10.21105/joss.03060>