



**HAL**  
open science

# Deterministic radiative transfer equation solver on unstructured tetrahedral meshes: Efficient assembly and preconditioning

Pierre Jolivet, Mohd Afeef Badri, Y. Favenec

## ► To cite this version:

Pierre Jolivet, Mohd Afeef Badri, Y. Favenec. Deterministic radiative transfer equation solver on unstructured tetrahedral meshes: Efficient assembly and preconditioning. *Journal of Computational Physics*, 2021, 437, pp.110313. 10.1016/j.jcp.2021.110313 . hal-03196806

**HAL Id: hal-03196806**

**<https://hal.science/hal-03196806v1>**

Submitted on 24 Apr 2023

**HAL** is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.



Distributed under a Creative Commons Attribution - NonCommercial 4.0 International License

# Deterministic Radiative Transfer Equation Solver on Unstructured Tetrahedral Meshes: Efficient Assembly and Preconditioning

P. Jolivet<sup>a,\*</sup>, M. A. Badri<sup>b</sup>, Y. Favennec<sup>c</sup>

<sup>a</sup>CNRS, Institut de Recherche en Informatique de Toulouse, 31062 Toulouse, France

<sup>b</sup>Université Paris-Saclay, CEA, Service de Thermo-hydraulique et de Mécanique des Fluides, 91191 Gif-sur-Yvette, France

<sup>c</sup>CNRS, Laboratoire de Thermique et Énergie de Nantes, 44306 Nantes, France

---

## Abstract

Due to its integro-differential nature, deriving schemes for numerically solving the radiative transfer equation (RTE) is challenging. Most solvers are efficient in specific scenarios: structured grids, simulations with low-scattering materials... In this paper, a full solver, from the discretization of the steady-state monochromatic RTE to the solution of the resulting system, is derived.

Using a mixed matrix-ready/matrix-free approach, our solver is able to discretize and solve a 45.7 billion unknown problem on 27 thousand processes in three minutes for a full physics involving scattering, absorption, and reflection. Because of the high dimensionality of the continuous equation, the linear system would have had more than  $6 \times 10^{15}$  nonzero entries if assembled explicitly. Our approach allows for large memory gains by only storing lower dimension reference matrices.

The finite element-based solver is wrapped around open-source software, FreeFEM for discretization, PETSc for linear algebra, and *hypra* for the algebraic multigrid infrastructure. Overall, deterministic results are presented on arbitrarily-decomposed unstructured grids for radiative transfer problems with scattering, absorbing, and reflecting heterogeneities on up to 27 thousand processes.

*Keywords:* radiative transfer, matrix-free solver, algebraic multigrid

---

## 1. Introduction

Radiative transfer is the phenomenon of energy transfer in the form of electromagnetic radiation, often characterized by its radiative intensity  $I$ . This intensity within any participating medium is affected by absorption, emission, reflection, and scattering processes. The radiative transfer equation (RTE) models such processes at the continuous level. The RTE is commonly used in the fields of thermal radiation [1, 2], neutronics [3, 4], astrophysics [5, 6], and optical imaging [7, 8], to name a few. This equation, in its steady-state monochromatic form, is stated as the following integro-differential equation:

$$(\vec{s} \cdot \nabla + \beta(\vec{x})) I(\vec{x}, \vec{s}) = \sigma(\vec{x}) \oint_S I(\vec{x}, \vec{s}') \phi(\vec{s}, \vec{s}') d\vec{s}' + \kappa(\vec{x}) I_b(\vec{x}), \quad (1)$$

where  $\vec{x} = (x, y, z) \in \Omega \subset \mathbb{R}^3$  and  $d\vec{s}$  is the differential solid angle around the direction  $\vec{s} = [\cos \varphi \sin \theta, \sin \varphi \sin \theta, \cos \theta]^T$ ,  $(\varphi, \theta) \in [0, 2\pi] \times [0, \pi]$  being the azimuthal and the zenith angles, respectively.  $\beta$  is the extinction coefficient computed as the sum  $\kappa + \sigma$ , where  $\kappa$  (resp.  $\sigma$ ) is the absorption (resp. scattering) coefficient. These three coefficients are piecewise constant in  $\Omega$  and their values depend on the various participating media in the domain.  $\phi$  is a scattering phase function that quantifies the probability of a photon traveling in  $\vec{s}'$  being scattered towards  $\vec{s}$ . Finally,  $I_b$  is the radiation from a black body specified by Planck's law.

Efficient solution strategy for the monochromatic steady-state RTE is still an active area of research for cases that involve scattering and reflections in complex geometries [9, 10]. The main challenge, for methods presented in this article, is the presence of in-scattering from the surface integral in eq. (1), which leads to coupled systems that are difficult to solve due to memory and convergence issues. Associated with the multigroup approximation [11, 12] or spectral models [13, 14], monochromatic RTE solvers are a stepping stone for designing multi-frequency RTE solvers in different scientific communities, e.g., combustion [15, 16].

In most of this paper, the boundary  $\partial\Omega$  is assumed to be nonreflective. If  $\vec{n}$  is the outward normal vector at a given point of  $\partial\Omega$ , the following inflow boundary condition is used to provide closure to eq. (1):

$$I = I_{\text{inflow}} \quad \forall (\vec{x}, \vec{s}) \in \Gamma^- = \{(\vec{x}, \vec{s}) : \vec{x} \in \partial\Omega, \vec{s} \cdot \vec{n} < 0\}. \quad (2)$$

---

\*Corresponding author: pierre.jolivet@enseeiht.fr

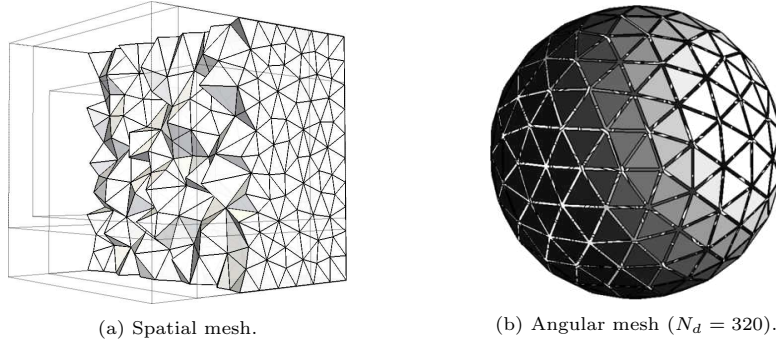


Figure 1: DOM and FEM coupling strategy, each spatial unknown is associated with a set of  $N_d$  angular unknowns.

22 One needs two schemes for discretizing eq. (1) both in space  $\vec{x}$  and angles  $\vec{s}$ . In this work, we couple the  
 23 finite element method (FEM) for the spatial discretization with the discrete ordinates method (DOM) based  
 24 on the sphere triangulation for the angular discretization [17, 18]. The terminology DOM is used throughout  
 25 the paper.

26 *Angular discretization.* The DOM, here performed according to the unit sphere triangulation, either based on  
 27 Thurgood's method [19] or icosahedron refinements [5], cf. fig. 1b, provides a set of directions and weights  
 28  $\{(\vec{s}_k, \omega_k)\}_{k=1}^{N_d}$ . Such a scheme is second-order accurate when combined with piecewise linear finite elements in  
 29 space [20]. The continuous intensity is now approximated by a set of discrete intensities  $\{I_k\}_{k=1}^{N_d}$ . The surface  
 30 integral in eq. (1) is thus evaluated as, for all discrete directions  $k \in \llbracket 1; N_d \rrbracket$ :

$$\oint_{\mathcal{S}} I(\vec{x}, \vec{s}) \phi(\vec{s}_k, \vec{s}) d\vec{s} \approx \sum_{j=1}^{N_d} \omega_j I_j(\vec{x}) \phi_{k,j}.$$

31 The integro-differential eq. (1) may now be stated as a set of  $N_d$  continuous partial differential equations:

$$\forall k \in \llbracket 1; N_d \rrbracket, (\vec{s}_k \cdot \nabla + \beta(\vec{x})) I_k(\vec{x}) = \sigma(\vec{x}) \sum_{j=1}^{N_d} \omega_j I_j(\vec{x}) \phi_{k,j} + \kappa(\vec{x}) I_b(\vec{x}). \quad (3)$$

32 In a similar fashion the inflow boundary condition from eq. (2) has to be rewritten in discrete angular form.  
 33 In this paper, collimated beams of intensity  $I_{\text{inflow}}$  impinge the boundary along prescribed entrant directions in  
 34  $L \subset \llbracket 1; N_d \rrbracket$ . Hence,  $\forall k \in \llbracket 1; N_d \rrbracket$ :

$$I_k = \begin{cases} I_{\text{inflow}} & \forall \vec{x} \in \partial\Omega : \vec{s}_k \cdot \vec{n} < 0, \text{ if } k \in L, \\ 0 & \text{otherwise.} \end{cases} \quad (4)$$

35 *Spatial discretization.* With the FEM, given a spatial mesh  $\Omega^h$ , cf. fig. 1a, and a set of appropriate functional  
 36 spaces  $\{\mathcal{V}_k^h\}_{k=1}^{N_d}$ , the following variational equations may be defined using the Galerkin method. The finite  
 37 element trial functions for the discrete radiative intensities  $\{I_k^h \in \mathcal{V}_k^h\}_{k=1}^{N_d}$  have to satisfy for all test functions  
 38  $\{J_k^h \in \mathcal{V}_k^h\}_{k=1}^{N_d}$ :

$$\forall k \in \llbracket 1; N_d \rrbracket, \int_{\Omega^h} \left( (\vec{s}_k \cdot \nabla + \beta(\vec{x})) I_k^h(\vec{x}) - \sigma(\vec{x}) \sum_{j=1}^{N_d} \omega_j I_j^h(\vec{x}) \phi_{k,j} \right) J_k^h(\vec{x}) d\vec{x} = \int_{\Omega^h} \kappa(\vec{x}) I_b(\vec{x}) J_k^h(\vec{x}) d\vec{x},$$

40 with the boundary conditions defined in eq. (4). In this paper, it is assumed that for all directions, the same  
 41 functional space  $\mathcal{V}^h$  is used to search for all the discrete radiative intensities, i.e., for all  $k \in \llbracket 1; N_d \rrbracket$ ,  $\mathcal{V}_k^h = \mathcal{V}^h$ .  
 42 In particular,  $\mathcal{V}^h$  is the space of piecewise linear functions throughout the paper. Because of the convection-  
 43 dominant term in eq. (1), an artificial diffusivity is introduced, and the following test functions derived from  
 44 the SUPG method [21] are used:  $J_k^h + \gamma \vec{s}_k \cdot \nabla J_k^h$ . An effective piecewise constant value of  $\gamma$  was determined in  
 45 prior works [22]:  $\gamma(\vec{x}) = \left( \frac{4}{h^2} + \beta(\vec{x})^2 + \frac{4\sigma(\vec{x})}{hN_d} \right)^{-1/2}$ , where  $h$  is the size of each spatial element. By integrating

46 by parts, the system of variational formulations becomes:  $\forall k \in \llbracket 1; N_d \rrbracket$ ,

$$\begin{aligned}
& \int_{\Omega^h} \left( \left( \beta(\vec{x}) I_k^h(\vec{x}) - \sigma(\vec{x}) \sum_{j=1}^{N_d} \omega_j I_j^h(\vec{x}) \phi_{k,j} \right) (J_k^h(\vec{x}) + \gamma(\vec{x}) \vec{s}_k \cdot \nabla J_k^h(\vec{x})) \right. \\
& \quad \left. - (\vec{s}_k \cdot \nabla J_k^h(\vec{x})) I_k^h(\vec{x}) + \gamma(\vec{x}) (\vec{s}_k \cdot \nabla I_k^h(\vec{x})) (\vec{s}_k \cdot \nabla J_k^h(\vec{x})) \right) d\vec{x} \\
& = \int_{\Omega^h} \kappa(\vec{x}) I_b(\vec{x}) (J_k^h(\vec{x}) + \gamma(\vec{x}) \vec{s}_k \cdot \nabla J_k^h(\vec{x})) d\vec{x} \\
& \quad - \int_{\Gamma^-} \vec{s}_k \cdot \vec{n} J_k^h(\vec{x}) I_{\text{inflow}}(\vec{x}) d\vec{x} - \int_{\partial\Omega^h \setminus \Gamma^-} \vec{s}_k \cdot \vec{n} I_k^h(\vec{x}) J_k^h(\vec{x}) d\vec{x}.
\end{aligned} \tag{5}$$

47 Now that the RTE has been discretized both in space and angles, it is possible to highlight the major  
48 contributions of this paper. Indeed, we present:

- 49 • an efficient way to assemble linear systems arising from the discretization of eq. (5),
- 50 • a scalable approach for solving such linear systems using a mixed matrix-ready/matrix-free approach,
- 51 • how to handle physics involving specular reflections.

52 The paper is organized as follows. In section 2, related work and limitations of current implementations of  
53 radiative transfer equation solvers are presented. In section 3, the flexibility of modern finite element domain  
54 specific languages is leveraged to efficiently discretize eq. (5) numerically. In section 4, methods to iteratively  
55 solve the large systems generated in the previous section on thousands of processes are presented. Validation  
56 and scaling analysis of our solver is presented in section 5. The capabilities of our solver when dealing with  
57 specular reflections are shown in section 6.

## 58 2. Related work

59 Most RTE solvers are either physics-based (stochastic) or deterministic. Concerning the former, Monte  
60 Carlo [23] or ray-tracing methods [24] are well-suited to parallel computing [25]. However, computations in-  
61 volving optically thick media are numerically challenging for such methods [26]. Moreover, coupling these with  
62 other physics, e.g., fluid mechanics, is also demanding [27].

63 For deterministic solvers, historically, the surface integral from eq. (1) was dealt with by the introduction  
64 of the DOM [28]. Though new approaches for discretizing in angles exist [29], it is still nowadays the most  
65 used technique for semi-discretizing the integro-differential RTE into a set of coupled PDEs on which standard  
66 spatial discretization schemes are applicable. In the field of radiative transfer, the DOM is often combined with  
67 the finite volume method (FVM) [1].

68 As an alternative, one may also use the FEM. Many recent works in the radiative transfer community support  
69 its use together with the DOM to improve the efficiency of radiative transport equation solvers [5, 30, 31, 9], or  
70 to deal with complex geometries [32, 33] or coupled physics [2].

71 Regardless of the spatial discretization, the fully discrete set of linear equations is commonly solved by a  
72 direction-by-direction iterative scheme known as source iteration [34, 35, 36]. It provides a memory-efficient  
73 strategy needed to handle the high-dimensionality of RTE in its discrete form eq. (5). Then, an optimal  
74 sweeping sequence is performed to compute solution of linear systems. Source iteration is widely popular in  
75 parallel deterministic solvers: Uintah [37] from University of Utah, Denovo [38] from ORNL, Ardra [39] from  
76 LLNL, PDT [40] from TAMU. However, they are most often optimized for regular geometries that have to be  
77 (semi-)structurally meshed to facilitate optimal marching (sweeping sequence) [41], which is also sensitive to  
78 boundary conditions, **unless, e.g., orthogonality conditions between reflective surfaces hold** [42]. These schemes  
79 may be extended to unstructured grids [43, 44, 45, 46, 47] but with understandably lower parallel efficiencies.

80 Krylov methods have been often used with source iteration. They are particularly suited to situations  
81 where there is a tight coupling between the discrete directions, e.g., strongly scattering or reflecting media [48].  
82 Currently, there are different ways to leverage Krylov methods for solving the discrete RTE, e.g.:

- 83 • direct application on the full linear system yielded by the spatial and DOM discretizations [49];
- 84 • in conjunction with sweep-based iterative schemes recasted as preconditioned operators [50].

85 The latter choice is often less memory-demanding since the workload can be implemented direction-by-direction.  
86 Contrary to direction-by-direction schemes, all the discrete radiative intensities are considered simultaneously  
87 in the present work. Consequently, RTE solvers based on direct application of Krylov methods are most often  
88 limited by the high dimensionality of eq. (5) and thus require a large amount of memory to store the according

89 linear system. As such, these solvers usually tackle small-scale problems [51, 49] (up to 512 MPI processes on  
90 a regularly-decomposed Cartesian grid of 262 thousand elements with up to 120 directions) [52] (up to four  
91 directions in 2D) or have to rely on tools such as adaptive mesh refinement in space [53] or in angles [54], which  
92 may complexify a distributed solver due to the need of proper load balance. To deal with larger problems, at  
93 least two modern ways to solve the RTE have emerged. First, the use of reduced order models [55] or sparse  
94 grids may be used to deal efficiently with the “curse of dimensionality” [56, 57] (up to  $3 \cdot 10^6$  unknowns solved  
95 sequentially). Second, Jacobian-free Newton–Krylov methods [58] have been extended to the RTE [59] (up  
96 to 512 MPI processes and 8 threads per MPI process on a Cartesian grid of 65 thousand elements with up  
97 to 120 directions). These two methods are based on simplifications of the RTE by approximating either the  
98 interactions between the unknowns in space and in angles for the latter, or the global operator for the former.

### 99 3. Efficient assembly

100 Nowadays, the finite element method is available in a large number of professional or academic domain  
101 specific (embedded) languages: deal.II [60], FEniCS [61], FreeFEM [62], etc. The goal of the next section is to  
102 show that these software typically cannot cope with the dimension complexity of the RTE if the corresponding  
103 linear system is assembled explicitly. However, we will use one of them, FreeFEM, to discretize auxiliary  
104 problems and show how this relates to the original PDE.

#### 105 3.1. Vectorial finite elements

106 Naively,  $N_d$  linear systems need to be assembled corresponding to the  $N_d$  variational formulations of eq. (5).  
107 However, by introducing the vectorial finite element space  $\mathbb{V}^h$  defined as the Cartesian product of  $N_d$  finite  
108 element spaces  $\mathcal{V}^h$ , eq. (5) may be formulated in terms of a single vectorial unknown  $\mathbb{I}^h$  and a single vectorial  
109 test function  $\mathbb{J}^h$ . Let  $\mathbb{I}^h = [I_1^h \cdots I_{N_d}^h]^T$ ,  $\mathbb{J}^h = [J_1^h \cdots J_{N_d}^h]^T$ ,  $\mathbb{S} = [\vec{s}_1 \cdots \vec{s}_{N_d}]^T$ , and  $\{\Phi_{i,j} = \omega_j \phi_{i,j}\}_{i,j=1}^{N_d}$ . If we  
110 denote by  $\otimes$  (resp.  $\odot$ ) the entrywise product (resp. scalar product) of two vectorial functions, and by  $\diamond$  the  
111 product of a matrix by a vectorial function, then, the bilinear part of eq. (5) becomes, for all  $\mathbb{J}^h \in \mathbb{V}^h$ :

$$\begin{aligned}
a(\mathbb{I}^h, \mathbb{J}^h) &= \int_{\Omega^h} \left( (\beta(\vec{x})\mathbb{I}^h(\vec{x}) - \sigma(\vec{x})\Phi \diamond \mathbb{I}^h(\vec{x})) \otimes (\mathbb{J}^h(\vec{x}) + \gamma(\vec{x})\mathbb{S} \odot \nabla \mathbb{J}^h(\vec{x})) \right. \\
&\quad \left. + \gamma(\vec{x})(\mathbb{S} \odot \nabla \mathbb{I}^h(\vec{x})) \otimes (\mathbb{S} \odot \nabla \mathbb{J}^h(\vec{x})) - (\mathbb{S} \odot \nabla \mathbb{J}^h(\vec{x})) \otimes \mathbb{I}^h(\vec{x}) \right) d\vec{x} \\
&\quad + \int_{\partial\Omega^h \setminus \Gamma^-} \left( (\mathbb{S} \odot \vec{n}) \otimes \mathbb{J}^h(\vec{x}) \right) \otimes \mathbb{I}^h(\vec{x}) d\vec{x}.
\end{aligned} \tag{6}$$

112 The same goes for the linear part of eq. (5) such that one may write for all  $\mathbb{J}^h \in \mathbb{V}^h$ :  $a(\mathbb{I}^h, \mathbb{J}^h) = l(\mathbb{J}^h)$ .

#### 113 3.2. Matrix-ready assembly

114 Let the linear system yielded by the discretization of the previous equation be written as:

$$\mathbb{A}\mathbb{X} = \mathbb{B}. \tag{7}$$

115 If the number of degrees of freedom in  $\mathcal{V}^h$  is  $n$ , since the number of components in the vectorial finite element  
116 space  $\mathbb{V}^h$  is equal to the number of directions  $N_d$  in the angular mesh,  $\mathbb{A}$  is of order  $N_d \cdot n$ . Furthermore, if it  
117 is assumed that each component of  $\mathbb{V}^h$  is interleaved in  $\mathbb{A}$ ,  $\mathbb{X}$ , and  $\mathbb{B}$ , then:

- 118 • the sparse matrix  $\mathbb{A}$  is made of dense  $N_d \times N_d$  blocks,
- 119 •  $\mathbb{B} = [b_{1,1}, \dots, b_{N_d,1}, b_{1,2}, \dots, b_{N_d,2}, \dots, b_{1,n}, \dots, b_{N_d,n}]^T$ .

120 Increasing the number of directions for the DOM means that the number of nonzero coefficients in  $\mathbb{A}$  will  
121 increase as well. In practice, the efficiency of two traditional finite element languages, FEniCS and FreeFEM,  
122 is assessed in fig. 2. For this test case, the RTE is discretized in a unit two-dimensional square using a fixed  
123 unstructured spatial mesh made of 6,768 triangles. The resulting finite element space  $\mathcal{V}^h$  has 3,487 degrees of  
124 freedom. The time spent to assemble the corresponding linear systems, while increasing the number of directions  
125  $N_d$ , is displayed. The results have been averaged over five consecutive runs, performed after cache warming.  
126 With FEniCS, prior to any computations, a first run to JIT compile the solver is also performed. This is done  
127 in sequential, on a single core of an Intel Core i7-6567U. Clearly, these timings are not satisfactory since both  
128 the spatial and angular meshes are tiny. They could not be used to resolve any nontrivial physics.

### 129 3.3. Semi-matrix-free assembly

130 The two previous general-purpose languages do not succeed at exploiting the structure of eq. (6). Instead,  
 131 they assemble the matrix from eq. (7) coefficient by coefficient, which has thus a cost proportional to the  
 132 order of the linear system  $N_d \cdot n$ . If it is assumed that the sparse matrix  $\mathbb{A}$  does not have to be assembled  
 133 explicitly, this cost may be greatly reduced. Indeed, if one looks for example at the following term from the  
 134 bilinear form eq. (6):  $\int_{\Omega^h} \beta(\vec{x}) \mathbb{I}^h(\vec{x}) \otimes \mathbb{J}^h(\vec{x}) d\vec{x}$ , assembled as  $\mathbb{A}_1$ , one can instead discretize this bilinear form  
 135  $\int_{\Omega^h} \beta(\vec{x}) I^h(\vec{x}) J^h(\vec{x}) d\vec{x}$ , assembled as  $\mathcal{A}_1$ , where both  $I^h$  and  $J^h$  are in the finite element space  $\mathcal{V}^h$  (of dimension  
 136  $n$ ), and not in  $\mathbb{V}^h$  (of dimension  $N_d \cdot n$ ). The action of  $\mathbb{A}_1$  on a vector  $\mathbb{X}$  of order  $N_d \cdot n$ , is nothing else than  
 137 the action of  $\mathcal{A}_1$  on all  $N_d$  components of order  $n$  of  $\mathbb{X}$ :  $\forall(k, i) \in \llbracket 1; N_d \rrbracket \times \llbracket 1; n \rrbracket$ ,

$$(\mathbb{A}_1 \mathbb{X})_{k+N_d \cdot (i-1)} = (\mathcal{A}_1 [x_{k,1} \cdots x_{k,n}]^T)_i.$$

138 Now, looking at:

$$\begin{aligned} a_2(\mathbb{I}^h, \mathbb{J}^h) &= \int_{\Omega^h} \left( (\beta(\vec{x}) \mathbb{I}^h(\vec{x})) \otimes (\gamma(\vec{x}) \mathbb{S} \odot \nabla \mathbb{J}^h(\vec{x})) - (\mathbb{S} \odot \nabla \mathbb{J}^h(\vec{x})) \otimes \mathbb{I}^h(\vec{x}) \right) d\vec{x} \\ &= \int_{\Omega^h} \left( (\gamma(\vec{x}) \beta(\vec{x}) - 1) \mathbb{I}^h(\vec{x}) \otimes (\mathbb{S} \odot \nabla \mathbb{J}^h(\vec{x})) \right) d\vec{x}, \end{aligned}$$

139 assembled as  $\mathbb{A}_2$ , it is instead possible to assemble the operators  $\mathcal{A}_{2a}$ ,  $\mathcal{A}_{2b}$ , and  $\mathcal{A}_{2c}$ , corresponding to the  
 140 discretization of the following bilinear forms:

$$\begin{aligned} a_{2a}(I^h, J^h) &= \int_{\Omega^h} (\gamma(\vec{x}) \beta(\vec{x}) - 1) I^h(\vec{x}) \frac{\partial J^h(\vec{x})}{\partial x} d\vec{x} \\ a_{2b}(I^h, J^h) &= \int_{\Omega^h} (\gamma(\vec{x}) \beta(\vec{x}) - 1) I^h(\vec{x}) \frac{\partial J^h(\vec{x})}{\partial y} d\vec{x} \\ a_{2c}(I^h, J^h) &= \int_{\Omega^h} (\gamma(\vec{x}) \beta(\vec{x}) - 1) I^h(\vec{x}) \frac{\partial J^h(\vec{x})}{\partial z} d\vec{x}. \end{aligned}$$

141 The action of  $\mathbb{A}_2$  on a vector  $\mathbb{X}$  of order  $N_d \cdot n$  may be computed as:  $\forall(k, i) \in \llbracket 1; N_d \rrbracket \times \llbracket 1; n \rrbracket$ ,

$$(\mathbb{A}_2 \mathbb{X})_{k+N_d \cdot (i-1)} = \left( (\vec{s}_{k,1} \mathcal{A}_{2a} + \vec{s}_{k,2} \mathcal{A}_{2b} + \vec{s}_{k,3} \mathcal{A}_{2c}) [x_{k,1} \cdots x_{k,n}]^T \right)_i,$$

142 where  $\vec{s}_{k,j}$  is the  $j$ th coordinate of the discrete direction  $\vec{s}_k$ , for  $j \in \llbracket 1; 3 \rrbracket$ . Another part of the bilinear form is:

$$a_3(\mathbb{I}^h, \mathbb{J}^h) = \int_{\Omega^h} \gamma(\vec{x}) (\mathbb{S} \odot \nabla \mathbb{I}^h(\vec{x})) \otimes (\mathbb{S} \odot \nabla \mathbb{J}^h(\vec{x})) d\vec{x},$$

143 assembled as  $\mathbb{A}_3$ . It is instead possible to assemble the operators  $\mathcal{A}_{3a}$ ,  $\mathcal{A}_{3b}$ ,  $\mathcal{A}_{3c}$ ,  $\mathcal{A}_{3d}$ ,  $\mathcal{A}_{3e}$ , and  $\mathcal{A}_{3f}$ , corresponding  
 144 to the discretization of the following bilinear forms:

$$\begin{aligned} a_{3a}(I^h, J^h) &= \int_{\Omega^h} \gamma(\vec{x}) \frac{\partial I^h(\vec{x})}{\partial x} \frac{\partial J^h(\vec{x})}{\partial x} d\vec{x} \\ a_{3b}(I^h, J^h) &= \int_{\Omega^h} \gamma(\vec{x}) \frac{\partial I^h(\vec{x})}{\partial y} \frac{\partial J^h(\vec{x})}{\partial y} d\vec{x} \\ a_{3c}(I^h, J^h) &= \int_{\Omega^h} \gamma(\vec{x}) \frac{\partial I^h(\vec{x})}{\partial z} \frac{\partial J^h(\vec{x})}{\partial z} d\vec{x} \\ a_{3d}(I^h, J^h) &= \int_{\Omega^h} \gamma(\vec{x}) \left( \frac{\partial I^h(\vec{x})}{\partial x} \frac{\partial J^h(\vec{x})}{\partial y} + \frac{\partial I^h}{\partial y} \frac{\partial J^h}{\partial x} \right) d\vec{x} \\ a_{3e}(I^h, J^h) &= \int_{\Omega^h} \gamma(\vec{x}) \left( \frac{\partial I^h(\vec{x})}{\partial x} \frac{\partial J^h(\vec{x})}{\partial z} + \frac{\partial I^h}{\partial z} \frac{\partial J^h}{\partial x} \right) d\vec{x} \\ a_{3f}(I^h, J^h) &= \int_{\Omega^h} \gamma(\vec{x}) \left( \frac{\partial I^h(\vec{x})}{\partial y} \frac{\partial J^h(\vec{x})}{\partial z} + \frac{\partial I^h}{\partial z} \frac{\partial J^h}{\partial y} \right) d\vec{x}. \end{aligned}$$

145 The action of  $\mathbb{A}_3$  on a vector  $\mathbb{X}$  of order  $N_d \cdot n$  may be computed as:  $\forall(k, i) \in \llbracket 1; N_d \rrbracket \times \llbracket 1; n \rrbracket$ ,

$$(\mathbb{A}_3 \mathbb{X})_{k+N_d \cdot (i-1)} = \left( (\vec{s}_{k,1}^2 \mathcal{A}_{3a} + \vec{s}_{k,2}^2 \mathcal{A}_{3b} + \vec{s}_{k,3}^2 \mathcal{A}_{3c} + \vec{s}_{k,1} \vec{s}_{k,2} \mathcal{A}_{3d} + \vec{s}_{k,1} \vec{s}_{k,3} \mathcal{A}_{3e} + \vec{s}_{k,2} \vec{s}_{k,3} \mathcal{A}_{3f}) [x_{k,1} \cdots x_{k,n}]^T \right)_i.$$

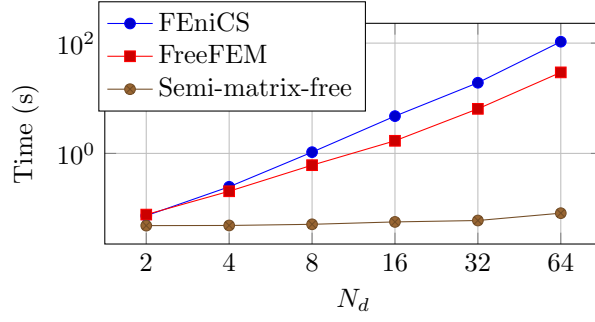


Figure 2: time spent assembling  $\mathbb{A}$  from eq. (7) for a two-dimensional domain with  $\mathbb{P}_1$  finite elements either explicitly or using our semi-matrix-free approach.

146 Eventually, the last part of eq. (6) with a volume integral assembled as  $\mathbb{A}_4$  to deal with is:

$$a_4(\mathbb{I}^h, \mathbb{J}^h) = \int_{\Omega^h} (\sigma(\vec{x})\Phi \diamond \mathbb{I}^h(\vec{x})) \otimes (\mathbb{J}^h(\vec{x}) + \gamma(\vec{x})\mathbb{S} \odot \nabla \mathbb{J}^h(\vec{x})) d\vec{x},$$

147 for which we instead assemble the operators  $\mathcal{A}_{4a}, \mathcal{A}_{4b}, \mathcal{A}_{4c}$ , and  $\mathcal{A}_{4d}$ , corresponding to the discretization of the  
148 following bilinear forms:

$$\begin{aligned} a_{4a}(I^h, J^h) &= \int_{\Omega^h} \sigma(\vec{x}) I^h(\vec{x}) J^h(\vec{x}) d\vec{x} \\ a_{4b}(I^h, J^h) &= \int_{\Omega^h} \sigma(\vec{x}) \gamma(\vec{x}) I^h(\vec{x}) \frac{\partial J^h(\vec{x})}{\partial x} d\vec{x} \\ a_{4c}(I^h, J^h) &= \int_{\Omega^h} \sigma(\vec{x}) \gamma(\vec{x}) I^h(\vec{x}) \frac{\partial J^h(\vec{x})}{\partial y} d\vec{x} \\ a_{4d}(I^h, J^h) &= \int_{\Omega^h} \sigma(\vec{x}) \gamma(\vec{x}) I^h(\vec{x}) \frac{\partial J^h(\vec{x})}{\partial z} d\vec{x}. \end{aligned}$$

149 Now, one can formally view the column vector  $\mathbb{X}$  of dimension  $N_d \cdot n$  as a column-major matrix  $X$  of dimensions  
150  $N_d \times n$ . Let  $Y = \Phi X$  be of dimensions  $N_d \times n$  as well. The action of  $\mathbb{A}_4$  on a vector  $\mathbb{X}$  of order  $N_d \cdot n$  may be  
151 computed as:

$$\forall (k, i) \in \llbracket 1; N_d \rrbracket \times \llbracket 1; n \rrbracket, (\mathbb{A}_4 \mathbb{X})_{k+N_d \cdot (i-1)} = \left( (\vec{s}_{k,1} \mathcal{A}_{4b} + \vec{s}_{k,2} \mathcal{A}_{4c} + \vec{s}_{k,3} \mathcal{A}_{4d} + \mathcal{A}_{4a}) [Y_{k,1} \cdots Y_{k,n}]^T \right)_i.$$

152 Additional work is needed to take into account the surface integral from eq. (6). It cannot be easily decomposed,  
153 neither with FreeFEM nor with FEniCS, in a tensor form, like for the other parts of the variational formulation,  
154 because of the Boolean expression in the definition of  $\Gamma^-$  in eq. (2). Thus, for the last line of eq. (6), the  
155 operators  $\{\mathcal{S}_k\}_{k=1}^{N_d}$  associated with the following bilinear forms have to be evaluated:

$$\forall k \in \llbracket 1; N_d \rrbracket, \zeta_k(I^h, J^h) = \int_{\partial\Omega^h \setminus \Gamma^-} \vec{s}_k \cdot \vec{n} I_k^h(\vec{x}) J_k^h(\vec{x}) d\vec{x}. \quad (10)$$

156 Once again, this would require a number of finite element assemblies that grows linearly with  $N_d$ . Hence, it is  
157 not tractable for large number of directions. To keep a low number of finite element assemblies, we color the  
158  $N_{\text{be}}$  boundary elements of  $\partial\Omega^h \setminus \Gamma^-$  into subsets  $\{\mathcal{F}_k\}_{k=1}^{\chi}$  such that two boundary elements, i.e., faces in 3D or  
159 edges in 2D, of  $\mathcal{F}_k, \forall k \in \llbracket 1; \chi \rrbracket$ , do not share a single vertex. This coloring is computed using a simple greedy  
160 algorithm. The following bilinear forms are then assembled into matrices  $\{\mathcal{S}'_k\}_{k=1}^{\chi}$ :

$$\forall k \in \llbracket 1; \chi \rrbracket, \zeta'_k(I^h, J^h) = \int_{\mathcal{F}_k} I^h(\vec{x}) J^h(\vec{x}) d\vec{x}. \quad (11)$$

161 Furthermore, given a direction  $k \in \llbracket 1; N_d \rrbracket$ , the integrand  $\vec{s}_k \cdot \vec{n}$  is constant per boundary element, so these  
162 scalar products can be precomputed and stored in  $\psi \in \mathbb{R}^{N_d \times N_{\text{be}}}$ . For a spatial degree of freedom  $i \in \llbracket 1; n \rrbracket$ ,  $\pi(i)$   
163 returns the subset of boundary elements to which  $i$  is associated. For a boundary element  $j \in \llbracket 1; N_{\text{be}} \rrbracket$ ,  $\pi'(j)$   
164 returns the color of the given face. The assembled operators  $\{\mathcal{S}_k\}_{k=1}^{N_d}$  which are the discretized counterparts  
165 of eq. (10) are thus computed as follows:

$$\forall (k, i, j) \in \llbracket 1; N_d \rrbracket \times \llbracket 1; n \rrbracket^2, \mathcal{S}_{k,i,j} = \sum_{k=1}^{N_d} \sum_{m \in \pi(i) \cap \pi(j)} \psi_{k,m} \mathcal{S}'_{\pi'(m)}{}_{i,j}. \quad (12)$$

$N_d$	gemm (ms)	CSR (ms)	GFLOP <sub>MF</sub> (/s)	GFLOP <sub>MR</sub>
80	26	115	2 (12)	8
128	48	125	3 (17)	21
320	184	262	10 (22)	129
512	383	395	21 (26)	330
1,280	1,960	1,000	95 (32)	2,060
2,048	4,579	1,574	222 (36)	5,274
5,120	27,281	4,116	1,250 (39)	32,964

Table 1: single-thread performance of the matrix–vector product with an increasing number of directions, for a fixed spatial domain.  $n = 44,204$ ,  $m = 6.29 \cdot 10^5$ , and  $m_S = 89,483$ .

Keep in mind that the matrices  $\{\mathcal{S}'_k\}_{k=1}^X$  are very sparse since they correspond to the discretization of a surface variational formulation. As a consequence, the matrices  $\{\mathcal{S}_k\}_{k=1}^{N_d}$  are also sparse, even though eq. (12) uses dense notations for the row and column indices.

For realistic geometries, it is expected that the number of unknowns associated with degrees of freedom on the surface  $\partial\Omega^h \setminus \Gamma^-$  will be lower than those associated with degrees of freedom in the volume  $\Omega^h$ . The cost of evaluating these surface integrals is thus expected to be much lower than the cost of evaluating the fixed number of reference matrices with volume integrals. Using the same test case as for matrix-ready assemblies in section 3.2, FreeFEM is now used to assemble the reference matrices  $\mathcal{A}_1, \mathcal{A}_{2a,2b}, \mathcal{A}_{3a,3b,3c,3d}$ , and  $\mathcal{A}_{4a,4b,4c}$ , as well as  $\{\mathcal{S}'_k\}_{k=1}^X$ . Note that for two-dimensional test cases, the matrices  $\mathcal{A}_{2c}, \mathcal{A}_{3c,3e,3f}$ , and  $\mathcal{A}_{4d}$  do not exist (only nine reference matrices). As displayed in fig. 2, the semi-matrix-free approach is clearly faster than matrix-ready assemblies by multiple orders of magnitude. Moreover, the cost of this part of the solver is now only loosely dependent on  $N_d$  because of the surface integrals. If the coefficient matrix of order  $N_d \cdot n$  from eq. (7) is not needed explicitly, we have derived a way to compute implicitly its action on a vector by assembling only reference matrices of order  $n$ .

#### 4. Iterative scheme/preconditioning

In the previous paragraph, it was shown how one may assemble auxiliary matrices of a much lower dimension than the original coefficient matrix from eq. (7), to discretize the RTE using the DOM coupled with the FEM. The resulting system, however, still needs to be solved. This is the focus of this section.

##### 4.1. Matrix–vector products

Because the coefficient matrix from eq. (7) is not known explicitly, it seems natural to use for the solution phase methods that only require the action of the matrix on vectors. Krylov methods [63] which are widely used in conjunction with the FEM seem like good candidates. Because they iterate by computing matrix–vector products, it is of paramount importance to design an efficient routine for our semi-matrix-free approach. To do so, a key observation is that, thanks to the compact support of finite element basis functions, the sparsity pattern of all fourteen reference matrices is the same. Thus, no matter the format used to store these sparse matrices, their pattern may be traversed only once. In the case of FreeFEM (and FEniCS), sparse matrices may be stored using the compressed sparse row (CSR) format, which is also widely used in common linear algebra backends such as PETSc [64]. Hence, if all reference matrices have  $m$  nonzero coefficients, we will assume that two arrays  $\mathcal{I}$  (of size  $n + 1$ ) and  $\mathcal{J}$  (of size  $m$ ) are used to traverse the matrices. We also denote by  $\mathcal{A}_\star(i)$  the  $i$ th nonzero coefficient of the reference matrix  $\star$ . The same strategy and notations are used for the matrices associated with eq. (10) ( $m_S, \mathcal{I}_S, \mathcal{J}_S$ , and  $\mathcal{S}_\star(i)$ ). All the previous operations from section 3.3 may now be implemented as in fig. 3. If  $\mathbb{A}$  from eq. (7) would have been explicitly assembled, its action on a vector would require  $2 \cdot m \cdot N_d^2$  double-precision floating-point operations. Our semi-matrix-free matrix–vector product performs the following number of FLOP:

$$\underbrace{N_d \cdot n \cdot (N_d - 1)}_{\text{line 2 of fig. 3}} + \underbrace{28 \cdot m \cdot N_d}_{\text{lines 4–11}} + \underbrace{2 \cdot m_S \cdot N_d}_{\text{lines 12–16}}. \quad (13)$$

Its efficiency is assessed in table 1. This time, a cube meshed with 235 thousand tetrahedra is used. The corresponding finite element space has 44 thousand degrees of freedom. It is assumed that the matrices  $\mathcal{A}_\star$  and  $\mathcal{S}_\star$  have already been assembled. The following results have been averaged over five consecutive runs, performed after cache warming. The time spent in the dense matrix–matrix product (line 2) as well as in the CSR traversal (lines 3–17) are reported in the second and third column respectively, with increasing number of directions  $N_d$ , reported in the first column. The number of FLOP performed in our product is reported in the fourth column. The numbers in parentheses are the actual throughputs in GFLOP/s of the matrix–vector product. In the fifth



```

1: function  $y = \text{MATMULTADD}(x, \Phi, \mathcal{I}, \mathcal{J}, \mathcal{A}_*, \mathcal{I}_S, \mathcal{J}_S, \mathcal{S}_*, \vec{s})$ 
2:  $\text{gemm}(\text{"N"}, \text{"N"}, N_d, n, N_d, 1, \Phi, N_d, x, N_d, 0, Y, N_d)$ 
3: do  $i = 0, n$  ▷ row indices
4:   do  $j = \mathcal{I}(i), \mathcal{I}(i + 1)$  ▷ column indices of the volume reference matrix
5:     do  $k = 0, N_d$  ▷ direction indices
6:        $b_1 = (\mathcal{A}_{2a}(j) + \vec{s}_{k,1}\mathcal{A}_{3a}(j) + \vec{s}_{k,2}\mathcal{A}_{3d}(j) + \vec{s}_{k,3}\mathcal{A}_{3e}(j)) \vec{s}_{k,1}$ 
7:        $b_2 = (\mathcal{A}_{2b}(j) + \vec{s}_{k,2}\mathcal{A}_{3b}(j) + \vec{s}_{k,3}\mathcal{A}_{3f}(j)) \vec{s}_{k,2}$ 
8:        $b_3 = (\mathcal{A}_{2c}(j) + \vec{s}_{k,3}\mathcal{A}_{3c}(j)) \vec{s}_{k,3}$ 
9:        $y(i \cdot N_d + k) = y(i \cdot N_d + k) + (\mathcal{A}_1(j) + b_1 + b_2 + b_3)x(\mathcal{J}(j) \cdot N_d + k) + (\vec{s}_{k,1}\mathcal{A}_{4b}(k) + \vec{s}_{k,2}\mathcal{A}_{4c}(k) + \vec{s}_{k,3}\mathcal{A}_{4d}(k) + \mathcal{A}_{4a}(k))Y(\mathcal{J}(j) \cdot N_d + k)$ 
10:     end do
11:   end do
12:   do  $j = \mathcal{I}_S(i), \mathcal{I}_S(i + 1)$  ▷ column indices of the surface reference matrix
13:     do  $k = 0, N_d$  ▷ direction indices
14:        $y(i \cdot N_d + k) = y(i \cdot N_d + k) + \mathcal{S}_k(j)x(\mathcal{J}_S(j) \cdot N_d + k)$ 
15:     end do
16:   end do
17: end do
18: end function ▷ return  $y = y + \mathbb{A}x$ 

```

Figure 3: matrix–vector product using the semi-matrix-free discretization.

column, knowing that the number of nonzero entries  $m$  in each reference matrix is  $6.29 \cdot 10^5$ , the number of double-precision floating-point operations that would have required a matrix-ready matrix–vector product is reported. For small number of directions ( $N_d \leq 320$ ), the performance of our algorithm is constrained by the single traversal of the sparsity pattern of the reference matrices, lines 3–17. With the CSR format, this is known to be an inefficient operation for discretizations on unstructured meshes [65]. When the number of directions becomes greater, the performance is driven by the one of the matrix–matrix product, line 2. Moreover, the vectorization of the inner loops lines 5 and 13 is most efficient for large values of  $N_d$ . In general, one may notice that our approach is one order of magnitude more efficient, needing fewer operations, for a wide range of directions. Note that for the case with  $N_d = 5,120$ , if  $\mathbb{A}$  were to be assembled explicitly using double-precision floating-point numbers, it would require  $6.29 \cdot 10^5 \cdot 5,120^2 \cdot 8 \approx 132$  TB of memory (not accounting for integers used to represent the sparsity pattern of  $\mathbb{A}$ ). With the semi-matrix-free approach, this operator is assembled on an ordinary desktop computer. Though the Intel MKL (with AVX-512) is used to perform the matrix–matrix product, we are focused here once again on the single-thread performance of our algorithm on an Intel Xeon Platinum 8168.

#### 4.2. Semi-matrix-free preconditioning

Krylov methods usually require a preconditioner so that they converge in a timely manner to an accurate solution. Because the coefficient matrix from eq. (7) is not assembled explicitly, standard preconditioners such as the restricted additive Schwarz method [66], or algebraic multigrid methods [67], are not usable. Matrix-free solvers typically rely on either: a) basic preconditioners such as the Jacobi method, or b) more sophisticated tools like geometric multigrid methods [68] or element-wise preconditioning [69]. In the case of the RTE, the former certainly makes a Krylov method such as the GMRES [70] converge, but with an insufficient convergence rate. When using different high-level languages, the latter is not easily interoperable since it requires a tight link with the language data structures, e.g., hierarchical meshes. In the next paragraph, an auxiliary operator will be explicitly assembled so that it can be used to precondition eq. (7) using standard techniques.

The diffusion synthetic acceleration (DSA) is a common tool in the radiative transfer community [71, 72] to improve the convergence of iterative schemes when dealing with scattering media with large optical depths. It is based on a simplification of eq. (1) using the diffusion approximation. To build an efficient preconditioner, we use **another** approximation by assuming that the phase function matrix  $\phi$  in eq. (3) is now diagonal:

$$\forall k \in \llbracket 1; N_d \rrbracket, (\vec{s}_k \cdot \nabla + \beta(\vec{x}))I_k(\vec{x}) = \sigma(\vec{x})\omega_k I_k(\vec{x})\phi_{k,k}.$$

The following physical phenomena are encompassed in these pruned PDEs: transport  $\vec{s}_k \cdot \nabla I_k(\vec{x})$ , absorption  $\kappa(\vec{x})I_k(\vec{x})$ , out-scattering  $\sigma(\vec{x})I_k(\vec{x})$ , and unidirectional in-scattering  $\sigma(\vec{x})\omega_k I_k(\vec{x})\phi_{k,k}$ . This construction makes the preconditioner able to handle a wide range of radiative transfer problems with absorbing, scattering, transparent, or semi-transparent media, as displayed in the following numerical experiments.

Assembling explicitly the pruned coefficient matrix  $B$  from the approximated RTE becomes much cheaper than assembling  $\mathbb{A}$ , since the dense  $N_d \times N_d$  blocks now become diagonal  $N_d \times N_d$  blocks. Indeed, the various

```

1: function  $M = \text{PRUNEDPC}(x, \Phi, \mathcal{I}, \mathcal{J}, \mathcal{A}_*, \vec{s})$ 
2: allocate  $\mathcal{I}_B$  to store  $N_d \cdot n + 1$  integers
3: allocate  $\mathcal{J}_B$  to store  $N_d \cdot m$  integers
4: allocate  $\mathcal{B}$  to store  $N_d \cdot m$  scalars
5: do  $i = 0, n$  ▷ row indices
6:    $m_{\text{row}} = \mathcal{I}(i + 1) - \mathcal{I}(i); j_{\text{shifted}} = N_d \cdot \mathcal{I}(i)$ 
7:   do  $k = 0, N_d$  ▷ direction indices
8:      $\mathcal{I}_B(N_d \cdot i + k) = j_{\text{shifted}} + k \cdot m_{\text{row}}$  ▷ number of nonzeros of this row
9:     do  $j = \mathcal{I}(i), \mathcal{I}(i + 1)$  ▷ column indices of the volume reference matrix
10:       $\text{idx} = j_{\text{shifted}} + k \cdot m_{\text{row}} + (j - \mathcal{I}(i))$ 
11:       $\mathcal{J}_B(\text{idx}) = \mathcal{J}(j) \cdot N_d + k$  ▷ column index of the pruned preconditioner
12:       $b_1 = (\mathcal{A}_{2a}(j) + \vec{s}_{k,1}\mathcal{A}_{3a}(j) + \vec{s}_{k,2}\mathcal{A}_{3d}(j) + \vec{s}_{k,3}\mathcal{A}_{3e}(j)) \vec{s}_{k,1}$ 
13:       $b_2 = (\mathcal{A}_{2b}(j) + \vec{s}_{k,2}\mathcal{A}_{3b}(j) + \vec{s}_{k,3}\mathcal{A}_{3f}(j)) \vec{s}_{k,2}$ 
14:       $b_3 = (\mathcal{A}_{2c}(j) + \vec{s}_{k,3}\mathcal{A}_{3c}(j)) \vec{s}_{k,3}$ 
15:       $\mathcal{B}(\text{idx}) = \mathcal{B}(\text{idx}) + \mathcal{A}_1(j) + b_1 + b_2 + b_3 + \left( \vec{s}_{k,1}\mathcal{A}_{4b}(k) + \vec{s}_{k,2}\mathcal{A}_{4c}(k) + \vec{s}_{k,3}\mathcal{A}_{3d}(k) + \mathcal{A}_{4a}(k) \right) \Phi_{k,k}$ 
16:    end do
17:  end do
18: end do
19:  $\mathcal{I}_B(N_d \cdot n) = N_d \cdot m$  ▷ total number of nonzeros
20: end function ▷ return  $B = \text{CSR}(\mathcal{I}_B, \mathcal{J}_B, \mathcal{B})$ 

```

Figure 4: explicit assembly of the pruned preconditioner.

$N_d$	Assembly (s)	$n_B$	$m_B$	Setup (s)
80	1.9	$4 \cdot 10^6$	$5 \cdot 10^7$	9.1
128	2.9	$6 \cdot 10^6$	$8 \cdot 10^7$	15.4
320	7.5	$1 \cdot 10^7$	$2 \cdot 10^8$	42.5
512	12.1	$2 \cdot 10^7$	$3 \cdot 10^8$	72.7
1,280	31.0	$6 \cdot 10^7$	$8 \cdot 10^8$	198.2
2,048	53.0	$9 \cdot 10^7$	$1 \cdot 10^9$	317.5
5,120	102.1	$2 \cdot 10^8$	$3 \cdot 10^9$	1,181.7

Table 2: time spent assembling the pruned preconditioner.  $n_B$  and  $m_B$  represent the order and the number of nonzero entries respectively in  $B$ .

242 components of  $\mathbb{I}^h$  in eq. (6) are only coupled through  $\Phi$ . The assembly cost is thus divided by  $N_d$ . Zero off-  
243 diagonal coefficients from the dense blocks of  $B$  are obviously not stored. Thus, there is no block structure in  
244  $B$  and it is assumed that it is stored in a standard sparse matrix using the CSR format. Its arrays are noted  
245  $\mathcal{I}_B, \mathcal{J}_B$ , and  $\mathcal{B}$ . In fig. 4, the procedure used to allocate and assemble the pruned preconditioner is displayed.  
246 For the same test case as in table 1 and on the same hardware, the time spent in this routine is shown in the  
247 second column of table 2, averaged over two runs. In the third (resp. fourth) column, the order of the system  
248 (resp. number of nonzero entries) is displayed. Choosing an efficient preconditioner to approximate the action  
249 of  $B^{-1}$  is not trivial. In our experiments, we used *hypre* algebraic multigrid solver BoomerAMG [73] with the  
250 following parameters: ext+i interpolation, HMIS coarsening, two sweeps of  $l_1$  Gauss–Seidel smoothing per level,  
251 and aggressive coarsening on all levels. Similar parameters are used in the neutron transport community [74].  
252 In the fifth column of table 2, the time spent in BoomerAMG for constructing the preconditioner is reported.  
253 Obviously, with a fixed spatial mesh but a refined angular mesh, a steady increase in setup time is observed.  
254 In the following section, a strategy to mitigate this issue will be presented, based on a hybrid multigrid in  
255 space–domain decomposition in angles preconditioner.

## 256 5. Large-scale numerical results

257 Both the matrix–vector product and the preconditioner have been defined in the previous paragraphs. In  
258 this section, the performance of the complete solver will be assessed for large-scale simulations. One application  
259 of the solver is its use to improve the design of heterogeneous participating media for material science. Thus,  
260 for a given geometry, multiple problems have to be solved, for example because of changing physical properties.  
261 In [75], this solver is used to study the response of open-cell foams made of SiC or  $\text{Al}_2\text{O}_3$  to radiations, which  
262 represent two different classes of materials (opaque or semi-transparent). We will thus focus on the strong  
263 scaling of the solver. However, since the transport community is heavily focused on weak scaling, there is **also**

264 such an analysis in section 6.4. Before presenting any scaling tests of the solver, it is first validated in the next  
 265 paragraph.

### 266 5.1. Validation

267 Problems 3i and 3ii from Kobayashi benchmarks [76] are used to validate the solver. These test cases  
 268 correspond to monochromatic emitting source radiation problems within heterogeneous materials with purely  
 269 reflecting walls. They have already been used to validate radiation transport solvers in the literature [77, 78, 79].

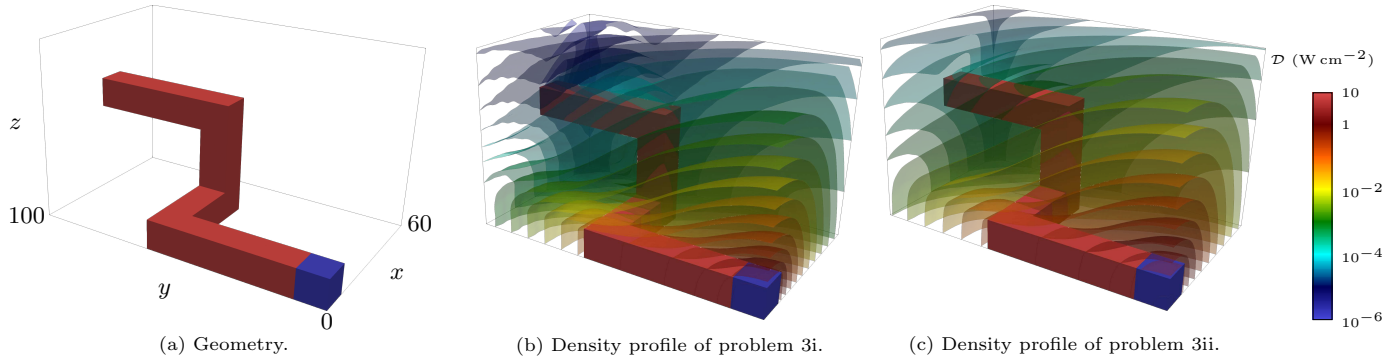


Figure 5: Kobayashi benchmark.

270 The material geometry for both problems 3i and 3ii is a  $60 \times 100 \times 60 \text{ cm}^3$  rectangular cuboid which con-  
 271 sists of three regions: source, void, and shield. They are respectively highlighted in blue, red, and transparent  
 272 in fig. 5a. The radiative exchange in both of these problems is driven by the source region, a  $10 \times 10 \times 10 \text{ cm}^3$   
 273 cube located at the origin of the coordinate system. It emits a black-body radiation of unit strength. The  
 274  $xz$ -plane for  $y = 0$ ,  $xy$ -plane for  $z = 0$ , and  $yz$ -plane for  $x = 0$ , i.e., front, bottom, and left planes, are specularly  
 275 reflecting, while the three other planes are vacuum walls. Problem 3i involves absorption, emission, and reflection  
 276 but neglects scattering. The coefficients ( $\kappa, \sigma$ ) are constant per region:  $(0.1 \text{ cm}^{-1}, 0 \text{ cm}^{-1})$  in the source,  
 277  $(10^{-4} \text{ cm}^{-1}, 0 \text{ cm}^{-1})$  in the void, and  $(0.1 \text{ cm}^{-1}, 0 \text{ cm}^{-1})$  in the shield region. Problem 3ii involves all modes  
 278 of radiation: absorption, emission, scattering, and reflection. The coefficients are now:  $(0.05 \text{ cm}^{-1}, 0.05 \text{ cm}^{-1})$ ,  
 279  $(5 \cdot 10^{-5} \text{ cm}^{-1}, 5 \cdot 10^{-5} \text{ cm}^{-1})$ , and  $(0.05 \text{ cm}^{-1}, 0.05 \text{ cm}^{-1})$ , respectively.

280 Problems 3i and 3ii are prone to ray effects and not trivial to solve with the DOM [78]. As such, these  
 281 benchmarks require a fine spatial and angular meshes, cf. fig. 1. This mitigates errors due to ray effects. Two  
 282 sets of meshes were used with our solver to further quantify ray effects and validate our results. A coarse set  
 283 made of a spatial mesh with  $1.41 \cdot 10^5$  nodes and  $8.15 \cdot 10^5$  elements and an angular mesh with 320 directions.  
 284 And a fine set made of a spatial mesh with  $7.88 \cdot 10^5$  nodes and  $4.71 \cdot 10^6$  elements and an angular mesh with  
 285 5,120 directions. As suggested in Kobayashi benchmark document [76], densities are extracted at certain point  
 286 locations within the materials. Table 3 provides these specific densities obtained with four different approaches:

- 287 1. numerical evaluation of an exact expression only available for problem 3i [76], column “reference”,
- 288 2. Monte Carlo solver [80], column “MC”,
- 289 3. spherical harmonics  $P_{21}$  solver [77], column “ $P_{21}$ ”,
- 290 4. our solver with both sets of meshes, column “DOM<sub>c</sub>” and “DOM<sub>f</sub>” for the coarse and the fine set,  
 291 respectively.

292 The data provided in the table suggests that the density computed by our solver when using the fine set of  
 293 meshes are in close agreement with the ones provided by other approaches. Moreover for both problems 3i and  
 294 3ii, the accuracy of the results improves significantly when switching from the coarse to the fine set of meshes.  
 295 Contours of the density in logscale are presented in fig. 5b (resp. fig. 5c) for problem 3i (resp. 3ii).

### 296 5.2. Parallelism

297 *Workload distribution.* The finite element method is most often parallelized on distributed-memory systems  
 298 using a coarse-grained approach such as domain decomposition. One of the many linear algebra backends that  
 299 does support this workload distribution is PETSc, that will be used in all the following tests. Considering  
 300 that an input mesh is split into  $P$  domains using a mesh partitioner such as Metis [81], FreeFEM is in charge  
 301 of generating a global unknown numbering of  $\mathcal{V}^h$ . This numbering is such that all  $P$  processes will own a  
 302 contiguous chunk of rows of all reference matrices  $\mathcal{A}_\star$  (one-dimensional row-wise distribution).

Table 3: density (W cm<sup>-2</sup>) of Kobayashi problems 3i and 3ii computed with: an exact expression [76], GMVP [80], Ardra [77] or our solver, for  $x = z = 5$  cm.

$y$	Problem 3i					Problem 3ii				
	Reference	MC	$P_{21}$	DOM <sub>c</sub>	DOM <sub>f</sub>	MC	$P_{21}$	DOM <sub>c</sub>	DOM <sub>f</sub>	
5	$5.96 \cdot 10^0$	$5.94 \cdot 10^0$	$5.96 \cdot 10^0$	$6.1 \cdot 10^0$	$5.96 \cdot 10^0$	$8.62 \cdot 10^0$	$8.61 \cdot 10^0$	$8.8 \cdot 10^0$	$8.63 \cdot 10^0$	
15	$1.37 \cdot 10^0$	$1.37 \cdot 10^0$	$1.34 \cdot 10^0$	$1.4 \cdot 10^0$	$1.37 \cdot 10^0$	$2.16 \cdot 10^0$	$2.13 \cdot 10^0$	$2.2 \cdot 10^0$	$2.16 \cdot 10^0$	
25	$5.01 \cdot 10^{-1}$	$5.01 \cdot 10^{-1}$	$4.95 \cdot 10^{-1}$	$5.1 \cdot 10^{-1}$	$5.00 \cdot 10^{-1}$	$8.94 \cdot 10^{-1}$	$8.84 \cdot 10^{-1}$	$9.0 \cdot 10^{-1}$	$8.93 \cdot 10^{-1}$	
35	$2.52 \cdot 10^{-1}$	$2.53 \cdot 10^{-1}$	$2.49 \cdot 10^{-1}$	$2.6 \cdot 10^{-1}$	$2.50 \cdot 10^{-1}$	$4.78 \cdot 10^{-1}$	$4.72 \cdot 10^{-1}$	$4.8 \cdot 10^{-1}$	$4.74 \cdot 10^{-1}$	
45	$1.50 \cdot 10^{-1}$	$1.50 \cdot 10^{-1}$	$1.56 \cdot 10^{-1}$	$1.4 \cdot 10^{-1}$	$1.48 \cdot 10^{-1}$	$2.89 \cdot 10^{-1}$	$2.99 \cdot 10^{-1}$	$2.8 \cdot 10^{-1}$	$2.87 \cdot 10^{-1}$	
55	$9.92 \cdot 10^{-2}$	$9.92 \cdot 10^{-2}$	$1.20 \cdot 10^{-1}$	$8.3 \cdot 10^{-2}$	$9.84 \cdot 10^{-2}$	$1.93 \cdot 10^{-1}$	$2.24 \cdot 10^{-1}$	$1.7 \cdot 10^{-1}$	$1.91 \cdot 10^{-1}$	
65	$4.23 \cdot 10^{-2}$	$4.23 \cdot 10^{-2}$	$5.10 \cdot 10^{-2}$	$3.9 \cdot 10^{-2}$	$4.20 \cdot 10^{-2}$	$1.05 \cdot 10^{-1}$	$1.19 \cdot 10^{-1}$	$9.9 \cdot 10^{-2}$	$1.04 \cdot 10^{-1}$	
75	$1.15 \cdot 10^{-2}$	$1.15 \cdot 10^{-2}$	$8.91 \cdot 10^{-3}$	$1.2 \cdot 10^{-2}$	$1.14 \cdot 10^{-2}$	$3.38 \cdot 10^{-2}$	$3.02 \cdot 10^{-2}$	$3.4 \cdot 10^{-2}$	$3.35 \cdot 10^{-2}$	
85	$3.25 \cdot 10^{-3}$	$3.25 \cdot 10^{-3}$	$1.95 \cdot 10^{-3}$	$3.4 \cdot 10^{-3}$	$3.33 \cdot 10^{-3}$	$1.08 \cdot 10^{-2}$	$8.54 \cdot 10^{-3}$	$1.1 \cdot 10^{-2}$	$1.08 \cdot 10^{-2}$	
95	$9.48 \cdot 10^{-4}$	$9.49 \cdot 10^{-4}$	$6.82 \cdot 10^{-4}$	$7.7 \cdot 10^{-4}$	$9.69 \cdot 10^{-4}$	$3.40 \cdot 10^{-3}$	$2.83 \cdot 10^{-3}$	$3.0 \cdot 10^{-3}$	$3.40 \cdot 10^{-3}$	

303 *Matrix–vector product.* As it is the case with standard matrix types from PETSc (and other backends like  
304 *hypr* [82]), it is furthermore assumed that the local representation of each matrix  $\mathcal{A}_*$  is split into two matrices  
305  $\{D_i\}_{i=1}^P$  and  $\{O_i\}_{i=1}^P$ , where the unknowns associated with the  $\{D_i\}_{i=1}^P$  are purely local to each domain, whereas  
306 those associated with the  $\{O_i\}_{i=1}^P$  involve interprocess communication for computing matrix–vector products,  
307 cf. [83]. By using a matrix of type “shell”, one can wrap the matrix–vector product from fig. 3 so that PETSc  
308 will use it in its Krylov solvers. If the reference matrices are split using the previous decomposition, it is trivial to  
309 extend the algorithm so that it overlaps communication due to interprocess unknowns shared with neighboring  
310 domains with local computation.

311 *Pruned preconditioner.* In the previous paragraphs, matrix-free system eq. (7) is solved by using an assembled  
312 preconditioner  $B$  defined in section 4.2. The strategy used to assemble  $B$  was to prune all off-diagonal nonzeros  
313 in the  $N_d \times N_d$  dense blocks of  $\mathbb{A}$ . The pruned operator  $B$  of dimension  $N_d \cdot n$  thus still remains structured,  
314 with blocks of dimension  $N_d$ , but they are now diagonal instead of dense. As displayed in the previous section  
315 in table 2, setting up BoomerAMG on the complete pruned operator  $B$  becomes costly for large numbers  
316 of directions. To decrease the setup cost, we propose to modify the preconditioner by first using a domain  
317 decomposition-like approach in angles, and then using multigrid in space. Instead of using a single instance of  
318 BoomerAMG with the single operator  $B$ , each structured diagonal block of dimension  $N_d$  will be decomposed  
319 into  $q$  subblocks, where  $q$  is a common divisor of the number of directions  $N_d$  and the number of processing  
320 elements  $P$ . This may be interpreted as nonoverlapping domain decomposition according to the angles. Then,  
321  $q$  instances of BoomerAMG are created for each subsystem of dimension  $\frac{N_d}{q} \cdot n$ . This is standard multigrid in  
322 space. Suitable restriction and interpolation operators will be defined later on to go from the full space to a  
323 single subsystem and vice versa. This is somehow similar to the anglesets strategy from [40]. From an algebraic  
324 point of view, the modified pruned preconditioner is now written as:

$$B^{-1} = \sum_{k=1}^q R_k^T (R_k B R_k^T)^{-1} R_k,$$

325 where  $q$  is a common divisor of the number of directions  $N_d$  and the number of processing elements  $P$ , and  $R_k$   
326 is the restriction on the  $k$ th angleset. Formally, for each spatial unknown,  $R_k$  selects from all  $N_d$  directions only  
327 the  $M_q = N_d/q$  ones from the appropriate angleset. In matrix form,  $R_k$  is a full-rank diagonal Boolean matrix  
328 written as:

$$\forall k \in \llbracket 1; q \rrbracket, R_k = \begin{bmatrix} J_k & & \\ & \ddots & \\ & & J_k \end{bmatrix} \in \{0, 1\}^{n \cdot M_q \times n \cdot N_d},$$

329 with  $J_k = [\mathbb{0}_{M_q \times (k-1) \cdot M_q} \quad \mathbb{1}_{M_q} \quad \mathbb{0}_{M_q \times (q-k) \cdot M_q}] \in \{0, 1\}^{M_q \times N_d}$ . Within the PETSc framework, this construc-  
330 tion of  $B^{-1}$  could be performed using an additive fieldsplit preconditioner (with as many fields as anglesets),  
331 but this would have two main bottlenecks:

- 332 • the operator  $B$  would have to be assembled explicitly first,
- 333 •  $q$  would need to be low for the preconditioner assembly to be efficient.

334 Instead, as it is standard when doing domain decomposition in space, the respective  $\{R_k B R_k^T\}_{k=1}^q$  are directly  
335 assembled on  $q$  different MPI communicators of size  $P/q$ . This can be done by looping on the appropriate  
336 directions from a given angleset line 7 from fig. 4, instead of looping on all  $N_d$  directions. For better memory  
337 efficiency, we use PETSc matrices of type “hypr” to store these operators, so that there is no copy or conversion

338 from the default PETSc type “aij” to “hypre”. Once the problem has been decomposed in angles,  $q$  instances of  
 339 BoomerAMG are called concurrently to approximate the action of each  $\{(R_k B R_k^T)^{-1}\}_{k=1}^q$  (multigrid in space),  
 340 using the same parameters as in section 3.3.

### 341 5.3. Test case and scaling

342 *Hardware and software settings.* Results were obtained on a partition of Irène composed of 1,656 nodes with  
 343 two 24-core Intel Skylake clocked at 2.7 GHz. The interconnect is an InfiniBand EDR full fat tree and the MPI  
 344 implementation exploited was IntelMPI version 2019.0.5.281. All binaries and shared libraries were compiled  
 345 with Intel compilers and Math Kernel Library support (for dense linear algebra computations) version 19.0.5.281.  
 346 PETSc version 3.12.4 with *hypre* version 2.18.2 were used.

347 *Geometry and physics.* For the spatial domain, a three-dimensional centimetric rectangular cuboid of dimension  
 348  $[0; 1] \times [0; 1] \times [0; 3]$  is used. In the rectangular cuboid lies a collection of multiple small spheres with varying  
 349 physical parameters, cf. fig. 6. The resulting mesh generated by Gmsh [84] is made of 1.9 million tetrahedra,  
 350 which are uniformly refined once, for a total of  $8 \times 1.9 = 15$  million tetrahedra. Before the refinement step,  
 351 the unstructured mesh is partitioned in parallel using ParMETIS [85]. The background coefficients used in the  
 352 rectangular cuboid are:  $\kappa = 10^{-6} \text{ cm}^{-1}$  and  $\sigma = 10^{-6} \text{ cm}^{-1}$ , i.e., a nearly transparent medium. In the spheres,  
 353 the values of this pair of coefficients vary between:  $\{10^{-6}, 10\}, \{10, 10^{-6}\}, \{0.1, 0.1\}, \{10, 10\}$ . These values  
 354 represent a mix of highly scattering, highly absorbing, and semi-transparent materials. They are considered  
 355 homogenized with  $\phi$  modeled using the Henyey–Greenstein phase function [86], with the anisotropy factor  
 356  $g = 0.5$ . As inflow boundary conditions, the rectangular cuboid is impinged with a collimated external radiative  
 357 source  $\mathbb{I}_{\text{inflow}}$  on its left face:

$$\mathbb{I}_{\text{inflow}} = I_0 \mathbb{1}_{(\vec{x}, \vec{s}) \in \{0\} \times [0.25; 0.75] \times [0.25; 0.75] \times \{[1, 0, 0]^T\}},$$

358 where  $I_0 = 1 \text{ W cm}^{-2} \text{ sr}^{-1}$ .  $\mathbb{1}$  represents the vectorial characteristic function of a set. Since emission is neglected  
 359 and the condition on angles reads  $\vec{s} \in \{[1, 0, 0]^T\}$ , the right-hand side  $\mathbb{B}$  of eq. (7) is nonzero for a single  
 360 component  $I^h$  associated with this direction. In total,  $N_d = 5,120$  directions are used to discretize the unit  
 361 sphere, cf. fig. 1b. This is an extremely fine angular mesh. Such a resolution will be necessary in a following test  
 362 when dealing with specular reflections on spheres in order to solve the physics accurately [32]. These directions  
 363 are grouped in  $q = 32$  anglesets of dimension  $N_d/q = 160$ . To the best of our knowledge, radiative transfer  
 364 equation solvers using Krylov methods have never been able to deal with such an angular resolution, mostly due  
 365 to memory limitations. After a simple post-processing step, it is easy to compute the spatial density profile  $\mathcal{D}$ ,  
 366 i.e., at the continuous level, the integral over the complete solid angle of the radiative intensity, by computing  
 367 for each spatial degree of freedom:  $\forall j \in [1; n], \mathcal{D}_j \approx \sum_{i=1}^{N_d} \omega_i x_{i,j}$ , assuming  $x$  is the solution vector from eq. (7).  
 368 For this test, the density is shown in fig. 7. Notice the jumps in the density due to the presence of spheres with  
 369 nonzero extinction coefficients along the collimated radiation.

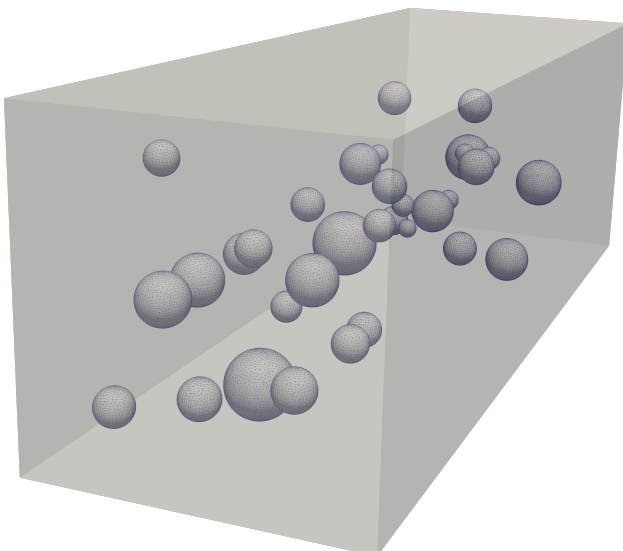


Figure 6: Gmsh geometry for the large-scale experiments.

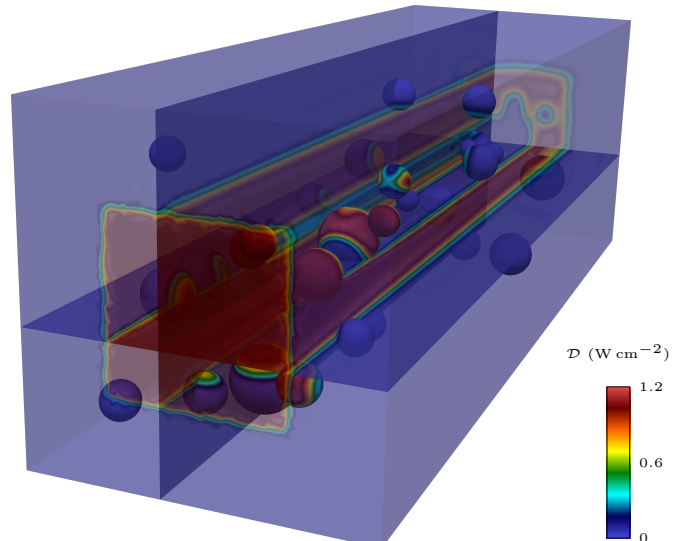
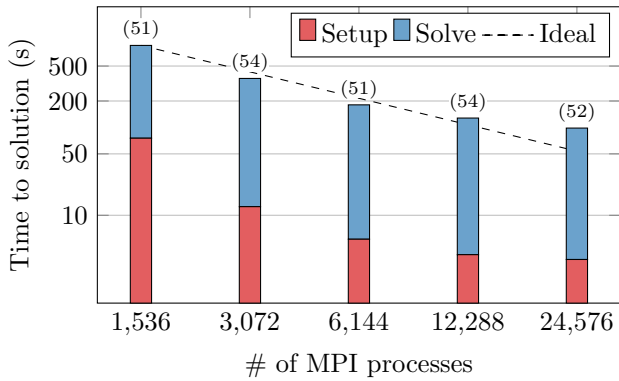


Figure 7: density profile  $\mathcal{D}$ . The rectangular cuboid is impinged on the front face and the radiation travels up to the back face, while being partially absorbed in the spheres.



(a) Time to solution. Each color represents the fraction of the total time spent in the respective tasks. In parenthesis, number of iterations.

$P$	Setup (s)	Solve (s)	# of iterations	Speedup
1,536	549.6	309.1	51	–
3,072	155.1	206.2	54	2.4
6,144	58.3	122.5	51	4.7
12,288	33.5	94.4	54	6.7
24,576	24.5	73.8	52	8.7

(b) Timings of the setup and the solution phases.

Figure 8: strong scaling analysis of the complete solver for a system of thirteen billion unknowns (without reflection).

371 *Performance.* In the following numerical experiments and as standard in the radiative transfer community [49,  
372 57, 87], convergence is assumed to be reached when the relative residual norm is lower than  $10^{-6}$ . Such a stopping  
373 criterion provided accurate final residuals<sup>1</sup>. Since the global operator is ill-conditioned, the long-recurrence  
374 Krylov method GMRES(30)<sup>2</sup> using double-precision floating-point numbers is preferred over short-recurrence  
375 methods such as BiCGSTAB, cf. [17] for a comparison with radiative transfer operators.

376 The performance of the complete solver is assessed in the strong scaling regime on 1,536 up to 24,576 MPI  
377 processes in fig. 8. The following steps of our method are timed, and averaged over two consecutive runs (same  
378 node distribution): assembly of the reference matrices  $\mathcal{A}_*$  and of the surface integrals (section 3.3), of the  
379 preconditioner  $B$  (section 4.2), and preconditioner setup (all three steps are summed and referenced to as the  
380 “setup” phase), and solution phase. In fig. 8a, each color represents the fraction of the total time spent in  
381 these two phases. For example, with 1,536 MPI processes, the height of the red bar is  $\frac{549.6}{549.6+309.1} = 64\%$   
382 of the height of the complete bar (figures taken from table 8b). These two colors are used to differentiate at a  
383 glance both phases, in order to estimate the setup time to solve time ratio. This may be useful to infer how to  
384 amortize setup costs when doing successive linear solves. For the exact timings, interested readers are referred  
385 to the right-hand side table. Overall, even with a rather high number of anglesets and aggressive multigrid  
386 parameters, the preconditioner remains very robust in terms of iterations on a wide range of MPI process  
387 counts. The test case was chosen so that it saturates the node memory when running on 1,536 MPI processes.  
388 For moderate number of MPI processes, computations (and thus cache effects) prevail over communications,  
389 which explains ideal speedups. At scale, the total time starts to be dominated by the solution phase. Indeed,  
390 on 24,576 processes, the average number of tetrahedra per process is a mere 610, so that the communication  
391 cost in the matrix–vector product cannot be fully overlapped and is no more negligible compared to the highly  
392 efficient computation, cf. table 1. For this largest configuration, the average grid and operator complexities of  
393  $\{(R_k B R_k^T)^{-1}\}_{k=1}^{32}$ , as returned by *hypr*, are 1.09 and 1.17 respectively. These low numbers imply that most  
394 of the communications take place in the matrix–vector products. Note that their volume of communication is  
395 proportional to  $N_d$ , while their computational cost is proportional to  $N_d^2$ , cf. eq. (13).

## 396 6. Specular reflections

397 In the previous sections, a solver for the radiative transfer equation was derived using a mixed matrix-  
398 ready/matrix-free approach. The goal of the following paragraphs is to show how this solver may be extended  
399 to deal with specular reflections. Such phenomena appear when radiations reach surfaces with different reflective  
400 indices. The radiative intensity is thus modified since a portion of the incident radiations is reflected back into  
401 the domain, according to Fresnel equations.

### 402 6.1. Assembly

403 To take into account these reflections, the bilinear form from eq. (6) has to be modified. Indeed, it is now  
404 assumed that the inflow boundary condition for all  $\{I_k^h\}_{k=1}^{N_d}$  is given, for all  $(\vec{x}, \vec{s}_k) \in \Gamma^-$ , by:

$$I_k^h = I_{k,\text{inflow}} + \sum_{\substack{j \in [1; N_d] \\ \vec{s}_j \cdot \vec{n} > 0}} \rho_{\vec{n}, k, j} I_j^h. \quad (14)$$

<sup>1</sup>checked with the PETSc option `-ksp_view_final_residual`

<sup>2</sup>the default restart value of PETSc



405 Describing precisely how the coefficients  $\rho$  are computed is beyond the scope of this paper. There are various  
 406 ways to do so, for example using the 1-DP (used in this paper), 3-DP, or the partition methods [88]. Their  
 407 values depend on the reflective index of the surface material, the direction of the normal  $\vec{n}$  at the point of  
 408 reflection, as well as the incident and reflected directions  $\vec{s}_k$  and  $\vec{s}_j$ . It is assumed that the reflective surface  $\Gamma^-$   
 409 may be decomposed into a union of surfaces with  $p$  different normals, i.e.,  $\Gamma^- = \cup_{r=1}^p \Gamma_r^-$ . For example, if  $\Omega^h$  is  
 410 a hexahedron,  $p$  is equal to six. With the new inflow boundary condition, for all components of  $\mathbb{I}^h$ , the bilinear  
 411 form  $a$  of eq. (6) has to be augmented by the following surface integral, for each component:

$$a(\mathbb{I}^h, \mathbb{J}^h) += \sum_{r=1}^p \int_{\Gamma_r^-} (\mathcal{P}_r \diamond \mathbb{I}^h(\vec{x})) \otimes \mathbb{J}^h(\vec{x}) d\vec{x}, \quad (15)$$

412 with the matrices  $\{\mathcal{P}_r\}_{r=1}^p$  being defined as:

$$\forall r \in \llbracket 1; p \rrbracket, \mathcal{P}_r = \begin{bmatrix} \rho_{\vec{n}, 1, 1} & \cdots & \rho_{\vec{n}, 1, N_d} \\ \vdots & \ddots & \vdots \\ \rho_{\vec{n}, N_d, 1} & \cdots & \rho_{\vec{n}, N_d, N_d} \end{bmatrix}.$$

413 For all  $(\vec{x}, \vec{s}_k) \in \Gamma^-$ ,  $\vec{s}_k \cdot \vec{n} < 0$ , cf. the definition of  $\Gamma^-$  given eq. (2). To take into account the conditional sum  
 414 from eq. (14),  $\vec{s}_j \cdot \vec{n} > 0$ , the diagonals of  $\{\mathcal{P}_r\}_{r=1}^p$  are set to zero. Moreover, since a large number of directions  
 415 are used, these matrices are relatively sparse. In practice, they are stored using the coordinate list format. As  
 416 in section 3.3, instead of assembling the vectorial bilinear form (15), the following variational formulations are  
 417 assembled:

$$\forall r \in \llbracket 1; p \rrbracket, q_r(I^h, J^h) = \int_{\Gamma_r^-} I^h(\vec{x}) J^h(\vec{x}) d\vec{x}.$$

418 Let  $n_{\Gamma^-}$  (resp.  $\{n_{\Gamma_r^-}\}_{r=1}^p$ ) be the number of unknowns from  $\mathcal{V}^h$  associated with elements of  $\Gamma^-$  (resp.  $\{\Gamma_r^-\}_{r=1}^p$ ).  
 419 The assembly cost of eq. (15) is proportional to  $N_d \cdot n_{\Gamma^-}$ , while the total assembly cost of all reference matrices  
 420 from the previous equation is proportional to  $\sum_{r=1}^p n_{\Gamma_r^-}$ .

## 421 6.2. Matrix–vector products

422 The initial algorithm from fig. 3 has to be extended to take into account the reference matrices from  
 423 the previous paragraph. It is assumed that the bilinear forms  $\{q_r\}_{r=1}^p$  are assembled using the CSR format  
 424  $\{\mathcal{I}_{q_r}, \mathcal{J}_{q_r}, \mathcal{Q}_{q_r}\}_{r=1}^p$ . This time, the sparsity pattern of the reference matrices corresponding to the surface integral  
 425 may be different. For a surface  $r \in \llbracket 1; p \rrbracket$ , and for all unknown indices  $i \in \llbracket 1; n_{\Gamma_r^-} \rrbracket$ , we assume that  $n_{\Gamma_r^-}(i)$   
 426 returns the unknown index in the finite element space  $\mathcal{V}^h$ . The extended matrix–vector algorithm is presented  
 427 in fig. 9. For each reflective surface, it involves first gathering the relevant values from the input vector of  
 428 size  $N_d \cdot n$ , then computing a sparse matrix–dense matrix product, and eventually, scattering back the result  
 429 into the output vector. When dealing with meshes with a large amount of different reflective surfaces (large  $p$ ),  
 430 this algorithm may not scale well. However, it is important to keep in mind that for realistic three-dimensional  
 431 geometries, the number of unknowns associated with surface elements  $n_{\Gamma^-}$  is expected to be much lower than  
 432 the global number of unknowns  $n$ . Thus, the most time-consuming part of the matrix–vector procedure should  
 433 still be the algorithm from fig. 3.

434  
 435 Because of the definition of  $\{\mathcal{P}_r\}_{r=1}^p$ , eq. (15) would not introduce any diagonal values in the dense  $N_d \times N_d$   
 436 blocks of  $\mathbb{A}$  from eq. (7) if it were to be assembled explicitly. Thus, the definition of the preconditioning matrix  
 437  $B$  from section 4.2 will remain the same for the numerical experiments.

## 438 6.3. Test case and scaling

439 The same geometry with the same background coefficients from section 5.3 will be used. The main difference  
 440 now is that all the spheres inside the rectangular cuboid model reflecting materials with no transmissivity. As  
 441 such, their insides are not meshed anymore, and their reflective indices are picked from the following set:  $\{1.8,$   
 442  $2.0, 2.2, 2.5\}$ . The number of directions is still set to  $N_d = 5,120$ , and with a mesh of 14.5 million tetrahedra,  
 443 the discrete system is of order  $14 \times 10^9$ . The outer surface of the rectangular cuboid is nonreflective (reflective  
 444 index set to one). The same preconditioner as before is used: hybrid multigrid in space–domain decomposition  
 445 in angles with the same BoomerAMG parameters and  $q = 32$  anglesets. In the density now displayed in fig. 10,  
 446 one can see that at the surface of the spheres in the rectangular cuboid, part of the radiative intensity is  
 447 reflected back into the spatial domain. The impact of reflections that cause more variations in the density  
 448 profile compared to fig. 7 may also be observed. Notice that for this test case, the total number of reflective  
 449 surfaces  $p$  is equal to 407. Depending on how the global domain is decomposed, each process has a different  
 450 number of local reflective surfaces  $\{p_i\}_{i=1}^P$ . Accordingly, they only load the corresponding matrices  $\{\mathcal{P}_r\}_{r=1}^{p_i}$  that  
 451 are generated in a pre-processing step. As these numbers decrease, so does the local cost of the matrix–vector  
 452 product (outer loop line 2 from fig. 9). For this configuration, the performance of the solver, cf. fig. 11, delivers  
 453 similar speedups as in the previous section (without reflective surface).

```

1: function  $y = \text{MATMULTADD}(x, \{\mathcal{P}_r, \mathcal{I}_{q_r}, \mathcal{J}_{q_r}, \mathcal{Q}_r\}_{r=1}^p)$ 
2:   do  $r = 1, p$ 
3:     allocate  $Y$  to store  $N_d \cdot n_{\Gamma_r^-}$  scalars
4:     allocate  $\text{tmp}$  to store  $N_d \cdot n_{\Gamma_r^-}$  scalars
5:     do  $i = 0, n_{\Gamma_r^-}$ 
6:       do  $k = 0, N_d$ 
7:          $Y(N_d \cdot i + k) = x(N_d \cdot n_{\Gamma_r^-}(i) + k)$ 
8:       end do
9:     end do
10:     $\text{tmp} = \mathcal{P}_r Y$ 
11:    do  $i = 0, n_{\Gamma_r^-}$ 
12:      do  $j = \mathcal{I}_{q_r}(i), \mathcal{I}_{q_r}(i+1)$ 
13:        do  $k = 0, N_d$ 
14:           $\text{idx} = N_d \cdot n_{\Gamma_r^-}(i) + k$ 
15:           $y(\text{idx}) = y(\text{idx}) + \text{tmp}(N_d \cdot i + k) \mathcal{Q}_r(j)$ 
16:        end do
17:      end do
18:    end do
19:  end do
20: end function

```

Figure 9: matrix–vector product for taking reflections into account.

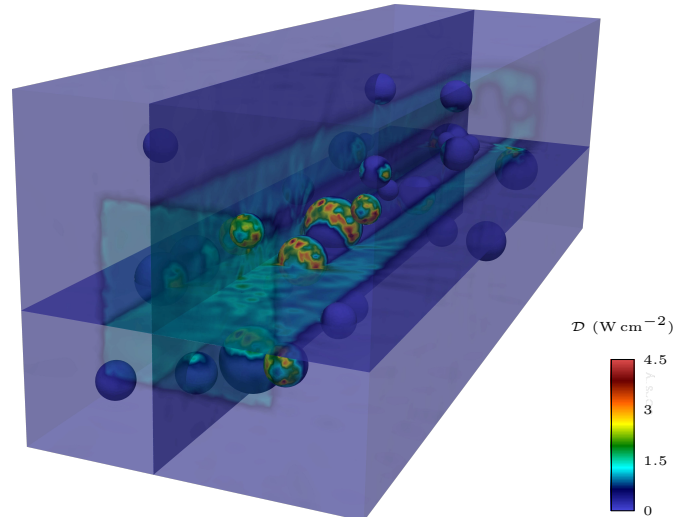
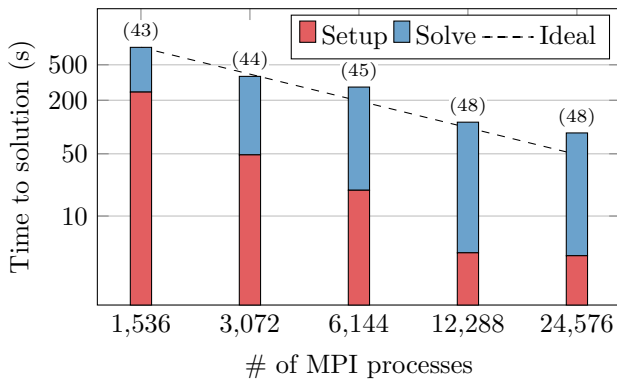


Figure 10: density profile  $\mathcal{D}$  when dealing with reflective surfaces. The rectangular cuboid is impinged on the front face and the radiation travels up to the back face, while being reflected on the surface of the spheres.



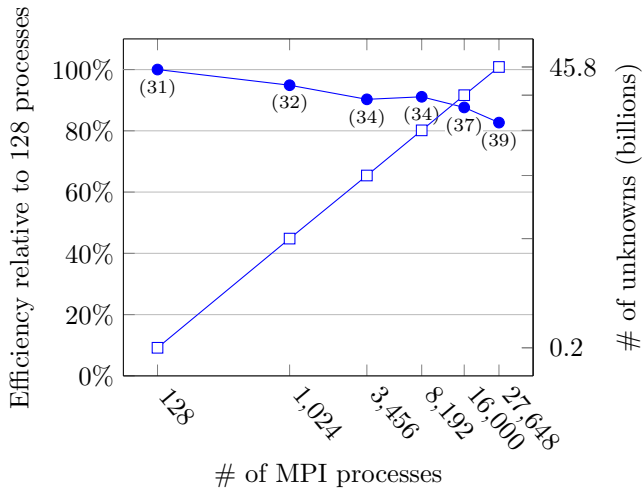
(a) Time to solution. Each color represents the fraction of the total time spent in the respective tasks. In parenthesis, number of iterations.

$P$	Setup (s)	Solve (s)	# of iterations	$\max\{p_i\}_{i=1}^P$
1,536	649.5	136.4	43	156
3,072	243.6	127.3	44	89
6,144	148.1	133.1	45	79
12,288	32.4	81.0	48	53
24,576	24.6	61.2	48	37

(b) Timings of the setup and the solution phases. The last column provides the maximum number of local reflective surfaces (which depends on the domain decomposition).

Figure 11: strong scaling analysis of the complete solver for a system of fourteen billion unknowns (with reflections).





(a) Efficiency and number of unknowns with an increasing number of uniform mesh refinements. In parenthesis, number of iterations.

$P$	Setup + solve (s)	# of iterations	$q$
128	133.0	31	2
1,024	140.4	32	8
3,456	145.5	34	16
8,192	143.8	34	32
16,000	149.6	37	64
27,648	157.1	39	128

(b) Timings of the solver. The last column provides the number of anglesets, equal to the number of *hypre* instances working concurrently.

Figure 12: weak scaling analysis of the complete solver for a system of 1.7 million unknowns per MPI process (with six reflective surfaces).

#### 6.4. Spatial mesh refinement and weak scaling

Results were obtained on a partition of Irène composed of 2,292 nodes with two 64-core AMD Rome clocked at 2.6 GHz. The interconnect is an InfiniBand HDR full fat tree. A simple three-dimensional unstructured cube being impinged with a collimated radiative beam is now considered. It is again generated by Gmsh, with approximately 750k tetrahedra. The reflective index on the six faces of the cube is 1.5. The number of directions is now set to  $N_d = 1,280$ . This saturates the memory available on one node of Irène, so it will be used next to perform a weak scaling analysis. Inside the cube, the following coefficients are used:  $(\kappa, \sigma) = (0.085, 20.10)$ , as suggested in the literature [89]. After using ParMETIS to decompose this initial mesh in  $128 \times i^3$  for  $i \in \llbracket 1, 6 \rrbracket$ , and assigning one subdomain per MPI rank, uniform spatial mesh refinement is performed concurrently on all processes. Each tetrahedron is split into  $i^3$  elements, by splitting all edges of the initial elements in  $i$ . No refinement is performed for  $i = 1$ . The smallest problem generated for  $i = 1$  has a total of 216 million unknowns, while the largest problem, running on 216 compute nodes for  $i = 6$ , has 45.7 billion unknowns. While the number of iterations to solve the global problem slightly increases, from 31 to 39, see third column in fig. 12b, one can notice that the overall efficiency remains above 80%, see fig. 12a. The same steps as in the previous strong scaling experiments are timed: assembly of the reference matrices  $\mathcal{A}_*$  and of the surface integrals (section 3.3), of the preconditioner  $B$  (section 4.2), preconditioner setup, and solution phase. To keep the preconditioner setup tractable, the number of anglesets  $q$  is increased with the global number of processes as shown in the rightmost column of fig. 12b. On this range of process numbers, the global preconditioner remains numerically efficient, and it converges in fewer iterates than in our previous strong scaling experiments with spherical inclusions and internal reflections, see figs. 8b and 11b for a comparison.

## 7. Conclusion

The radiative transfer equation is a high-dimensional integro-differential equation. In this paper, a discretization in both space and angles was used to solve it numerically. Our solver is able to discretize the RTE efficiently by using a high-level finite element language, FreeFEM. By using such a language, most of the burden inherent of the finite element method, e.g., mesh data structure, is hidden from the user. Unfortunately, these languages do not exploit the structure of the RTE and cannot be used out of the box to directly discretize it.

The mixed matrix-ready/matrix-free approach introduced in this paper has the two following important properties. First, it relies on a low number of reference matrices of a much lower dimension than the original system. These may be assembled fast using almost any domain specific language at a relatively low memory and computational cost. Second, the matrix-vector product uses a dense matrix-matrix product, which may be highly optimized, and it is possible to assemble explicitly a preconditioner to reduce the number of iterations needed by Krylov methods to converge.

By using a hybrid multigrid in space-domain decomposition in angles preconditioner, we are able to discretize and solve a 45.7 billion unknown system in less than three minutes on 27,648 MPI processes. Moreover, we show how our solver may be extended to deal with physics involving specular reflections on both exterior or interior surfaces with no orthogonality condition. Thanks to the flexibility of modern domain specific languages and

490 of PETSc, it would be easy to couple our solver with other fields, e.g., through a “fieldsplit” preconditioner<sup>3</sup>.  
491 Overall, we exhibit large-scale deterministic results on fine unstructured and arbitrarily-decomposed spatial  
492 meshes and large numbers of directions, with a mix of heterogeneous scattering, absorbing, and reflecting  
493 materials.

## 494 Acknowledgments

495 This work was granted access to the HPC resources of TGCC@CEA under the allocations A0050607519,  
496 A0070607519, and A0090607519 made by GENCI.

## 497 References

- 498 [1] P. J. Coelho, Advances in the discrete ordinates and finite volume methods for the solution of radiative heat  
499 transfer problems in participating media, *Journal of Quantitative Spectroscopy and Radiative Transfer* 145  
500 (2014) 121–146.
- 501 [2] M. Ghattassi, J.-R. Roche, F. Asllanaj, M. Boutayeb, Galerkin method for solving combined radiative and  
502 conductive heat transfer, *International Journal of Thermal Sciences* 102 (2016) 122–136.
- 503 [3] P. Lesaint, P.-A. Raviart, On a finite element method for solving the neutron transport equation, *Publi-  
504 cations Mathématiques et Informatique de Rennes* (1974) 1–40.
- 505 [4] E. E. Lewis, W. F. Miller, *Computational methods of neutron transport*, John Wiley and Sons, Inc., 1984.
- 506 [5] S. Richling, E. Meinköhn, N. Kryzhevoi, G. Kanschat, Radiative transfer with finite elements - I. Basic  
507 method and tests, *Astronomy & Astrophysics* 380 (2001) 776–788.
- 508 [6] E. Meinköhn, S. Richling, Radiative transfer with finite elements - II. Ly $\alpha$  line transfer in moving media,  
509 *Astronomy & Astrophysics* 392 (2002) 827–839.
- 510 [7] G. S. Abdoulaev, A. H. Hielscher, Three-dimensional optical tomography with the equation of radiative  
511 transfer, *Journal of Electronic Imaging* 12 (2003) 594–602.
- 512 [8] T. Tarvainen, M. Vauhkonen, S. R. Arridge, Gauss–Newton reconstruction method for optical tomography  
513 using the finite element solution of the radiative transfer equation, *Journal of Quantitative Spectroscopy  
514 and Radiative Transfer* 109 (2008) 2767–2778.
- 515 [9] G. Kanschat, J. C. Ragusa, A Robust Multigrid Preconditioner for  $S_N$  DG Approximation of Monochro-  
516 matic, Isotropic Radiation Transport Problems, *SIAM Journal on Scientific Computing* 36 (2014) A2326–  
517 A2345.
- 518 [10] D. Ruh, S. Subramanian, M. Theodor, H. Zappe, A. Seifert, Radiative transport in large arteries, *Biomed-  
519 ical Optics Express* 5 (2014) 54–68.
- 520 [11] A. T. Till, J. S. Warsa, J. E. Morel, Application of linear multifrequency-grey acceleration to preconditioned  
521 Krylov iterations for thermal radiation transport, *Journal of Computational Physics* 372 (2018) 931–955.
- 522 [12] T. S. Haut, C. D. Ahrens, A. Jonko, R. B. Lowrie, A. T. Till, A new multigroup method for cross-sections  
523 that vary rapidly in energy, *Journal of Quantitative Spectroscopy and Radiative Transfer* 187 (2017)  
524 461–471.
- 525 [13] A. G. Clements, R. Porter, A. Pranzitelli, M. Pourkashanian, Evaluation of FSK models for radiative heat  
526 transfer under oxyfuel conditions, *Journal of Quantitative Spectroscopy and Radiative Transfer* 151 (2015)  
527 67–75.
- 528 [14] V. Kez, F. Liu, J.-L. Consalvi, J. Ströhle, B. Epple, A comprehensive evaluation of different radiation  
529 models in a gas turbine combustor under conditions of oxy-fuel combustion with dry recycle, *Journal of  
530 Quantitative Spectroscopy and Radiative Transfer* 172 (2016) 121–133.
- 531 [15] R. Viskanta, P. M. Mengüç, Radiation heat transfer in combustion systems, *Progress in Energy and  
532 Combustion Science* 13 (1987) 97–160.

---

<sup>3</sup><http://www.mcs.anl.gov/petsc/petsc-current/docs/manualpages/PC/PCFIELDSPLIT.html>

- 533 [16] D. Poitou, M. El Hafi, B. Cuenot, Analysis of radiation modeling for turbulent combustion: development  
534 of a methodology to couple turbulent combustion and radiative heat transfer in LES, *Journal of Heat*  
535 *Transfer* 133 (2011).
- 536 [17] G. Kanschat, Solution of radiative transfer problems with finite elements, in: *Numerical Methods in*  
537 *Multidimensional Radiative Transfer*, Springer, 2009, pp. 49–98.
- 538 [18] T. Camminady, M. Frank, J. Kusch, Highly uniform quadrature sets for the discrete ordinates method,  
539 in: *The International Conference on Mathematics and Computational Methods Applied to Nuclear Science*  
540 *and Engineering*, 2019, pp. 25–29.
- 541 [19] C. P. Thurgood, A. Pollard, H. A. Becker, The  $T_N$  quadrature set for the discrete ordinates method,  
542 *Journal of Heat Transfer* 117 (1995) 1068–1070.
- 543 [20] M. A. Badri, P. Jolivet, B. Rousseau, S. Le Corre, H. Dignonnet, Y. Favennec, Vectorial finite elements for  
544 solving the radiative transfer equation, *Journal of Quantitative Spectroscopy and Radiative Transfer* 212  
545 (2018) 59–74.
- 546 [21] A. N. Brooks, T. J. Hughes, Streamline upwind/Petrov–Galerkin formulations for convection dominated  
547 flows with particular emphasis on the incompressible Navier–Stokes equations, *Computer Methods in*  
548 *Applied Mechanics and Engineering* 32 (1982) 199–259.
- 549 [22] M. Avila, R. Codina, J. Principe, Spatial approximation of the radiation transport equation using a  
550 subgrid-scale finite element method, *Computer Methods in Applied Mechanics and Engineering* 200 (2011)  
551 425–438.
- 552 [23] J. R. Howell, M. Perlmutter, Monte Carlo solution of thermal transfer through radiant media between gray  
553 walls, *Journal of Heat Transfer* 86 (1964) 116–122.
- 554 [24] L. Liu, Discrete curved ray-tracing method for radiative transfer in an absorbing–emitting semitransparent  
555 slab with variable spatial refractive index, *Journal of Quantitative Spectroscopy and Radiative Transfer*  
556 83 (2004) 223–228.
- 557 [25] A. Badal, F. Zafar, H. Dong, A. Badano, A real-time radiation dose monitoring system for patients and  
558 staff during interventional fluoroscopy using a GPU-accelerated Monte Carlo simulator and an automatic  
559 3D localization system based on a depth camera, in: *Physics of Medical Imaging*, International Society for  
560 Optics and Photonics, 2013.
- 561 [26] M. Min, C. P. Dullemond, C. Dominik, A. de Koter, J. W. Hovenier, Radiative transfer in very optically  
562 thick circumstellar disks, *Astronomy & Astrophysics* 497 (2009) 155–166.
- 563 [27] R. Fournier, S. Blanco, V. Eymet, M. El Hafi, C. Spiesser, Radiative, conductive and convective heat-  
564 transfers in a single Monte Carlo algorithm, in: *Journal of Physics: Conference Series*, volume 676, IOP  
565 Publishing, 2016.
- 566 [28] S. Chandrasekhar, *Radiative transfer*, Oxford University Press (1950).
- 567 [29] H. Fujiwara, An Accurate Quadrature Rule on the Sphere for Fast Computation of the Radiative Transport  
568 Equation, arXiv:1409.0111 [math.NA] (2014).
- 569 [30] Y. Lu, A. F. Chatzioannou, A parallel adaptive finite element method for the simulation of photon migra-  
570 tion with the radiative transfer-based model, *International Journal for Numerical Methods in Biomedical*  
571 *Engineering* 25 (2009) 751–770.
- 572 [31] G. Widmer, An efficient sparse finite element solver for the radiative transfer equation, *Journal of Heat*  
573 *Transfer* 132 (2010).
- 574 [32] D. Le Hardy, M. A. Badri, B. Rousseau, S. Chupin, D. Rochais, Y. Favennec, 3D numerical modelling  
575 of the propagation of radiative intensity through a X-ray tomographed ligament, *Journal of Quantitative*  
576 *Spectroscopy and Radiative Transfer* 194 (2017) 86–97.
- 577 [33] S. H. Kang, T.-H. Song, Finite element formulation of the first- and second-order discrete ordinates  
578 equations for radiative heat transfer calculation in three-dimensional participating media, *Journal of*  
579 *Quantitative Spectroscopy and Radiative Transfer* 109 (2008) 2094–2107.
- 580 [34] W. A. Fiveland, Discrete ordinates solutions of the radiative transport equation for rectangular enclosures,  
581 *Journal of Heat Transfer* 106 (1984) 699–706.

- 582 [35] M. L. Adams, E. W. Larsen, Fast iterative methods for discrete ordinates particle transport calculations,  
583 Progress in Nuclear Energy 40 (2002) 3–159.
- 584 [36] D. Mihalas, Stellar atmospheres, W. H. Freeman and Company, 1978.
- 585 [37] S. Kumar, A. Humphrey, W. Usher, S. Petruzza, B. Peterson, J. A. Schmidt, D. Harris, B. Isaac,  
586 J. Thornock, T. Harman, et al., Scalable data management of the Uintah simulation framework for  
587 next-generation engineering problems with radiation, in: Asian Conference on Supercomputing Frontiers,  
588 Springer, 2018, pp. 219–240.
- 589 [38] T. M. Evans, A. S. Stafford, R. N. Slaybaugh, K. T. Clarno, Denovo: a new three-dimensional parallel  
590 discrete ordinates code in SCALE, Nuclear Technology 171 (2010) 171–200.
- 591 [39] U. Hannebutte, P. Brown, Ardra: Scalable parallel code system to perform neutron-and radiation-transport  
592 calculations, Technical Report, Lawrence Livermore National Laboratory, 1999.
- 593 [40] M. P. Adams, M. L. Adams, W. D. Hawkins, T. Smith, L. Rauchwerger, N. M. Amato, T. S. Bailey, R. D.  
594 Falgout, Provably optimal parallel transport sweeps on regular grids, in: International Conference on  
595 Mathematics and Computational Methods Applied to Nuclear Science and Engineering, 2013.
- 596 [41] S. D. Pautz, T. S. Bailey, Parallel Deterministic Transport Sweeps of Structured and Unstructured Meshes  
597 with Overloaded Mesh Decompositions, Nuclear Science and Engineering 185 (2017) 70–77.
- 598 [42] M. P. Adams, M. L. Adams, W. D. Hawkins, T. Smith, L. Rauchwerger, N. M. Amato, T. S. Bailey, R. D.  
599 Falgout, A. Kunen, P. Brown, **Provably optimal parallel transport sweeps on semi-structured grids**, Journal  
600 of Computational Physics 407 (2020) 109234.
- 601 [43] S. J. Plimpton, B. Hendrickson, S. P. Burns, W. McLendon III, L. Rauchwerger, Parallel  $S_N$  sweeps on  
602 unstructured grids: algorithms for prioritization, grid partitioning, and cycle detection, Nuclear Science  
603 and Engineering 150 (2005) 267–283.
- 604 [44] F. Asllanaj, V. Feldheim, P. Lybaert, Solution of radiative heat transfer in 2D geometries by a modified  
605 finite volume method based on a cell vertex scheme using unstructured triangular meshes, Numerical Heat  
606 Transfer, Part B: Fundamentals 51 (2007) 97–119.
- 607 [45] Y. Wang, J. C. Ragusa, Diffusion synthetic acceleration for high-order discontinuous finite element  $S_N$   
608 transport schemes and application to locally refined unstructured meshes, Nuclear Science and Engineering  
609 166 (2010) 145–166.
- 610 [46] T. S. Haut, P. G. Maginot, V. Z. Tomov, B. S. Southworth, T. A. Brunner, T. S. Bailey, An Efficient  
611 Sweep-Based Solver for the  $S_N$  Equations on High-Order Meshes, Nuclear Science and Engineering 193  
612 (2019) 746–759.
- 613 [47] J. I. Vermaak, J. C. Ragusa, M. L. Adams, J. E. Morel, Massively parallel transport sweeps on meshes  
614 with cyclic dependencies, Journal of Computational Physics (2020) 109892.
- 615 [48] M. Seaid, M. Frank, A. Klar, R. Pinnau, G. Thömmes, Efficient numerical methods for radiation in gas  
616 turbines, Journal of Computational and Applied Mathematics 170 (2004) 217–239.
- 617 [49] W. F. Godoy, P. E. DesJardin, On the use of flux limiters in the discrete ordinates method for 3D radiation  
618 calculations in absorbing and scattering media, Journal of Computational Physics 229 (2010) 3189–3213.
- 619 [50] B. W. Patton, J. P. Holloway, Application of preconditioned GMRES to the numerical solution of the  
620 neutron transport equation, Annals of Nuclear Energy 29 (2002) 109–136.
- 621 [51] C. K. Krishnaprakas, K. Badari Narayana, P. Dutta, Use of GCG methods for the efficient solution of  
622 matrix problems arising from the FVM formulation of radiative transfer, Numerical Heat Transfer. Part  
623 B: Fundamentals 40 (2001) 515–533.
- 624 [52] J. P. L. Lorca, G. Kanschat, Multilevel Schwarz Preconditioners for Singularly Perturbed Symmetric  
625 Reaction–Diffusion Systems, arXiv:1811.03839 [math.NA] (2018).
- 626 [53] M. R. Charest, C. P. Groth, Ö. L. Gülder, Solution of the equation of radiative transfer using a Newton–  
627 Krylov approach and adaptive mesh refinement, Journal of Computational Physics 231 (2012) 3023–3040.
- 628 [54] B. J. Adigun, A. G. Buchan, A. Adam, S. Dargaville, M. A. Goffin, C. C. Pain, A Haar wavelet method for  
629 angularly discretising the Boltzmann transport equation, Progress in Nuclear Energy 108 (2018) 295–309.

- 630 [55] L. Soucasse, A. G. Buchan, S. Dargaville, C. C. Pain, An angular reduced order model for radiative transfer  
631 in non grey media, *Journal of Quantitative Spectroscopy and Radiative Transfer* 229 (2019) 23–32.
- 632 [56] G. Widmer, R. Hiptmair, C. Schwab, Sparse adaptive finite elements for radiative transfer, *Journal of*  
633 *Computational Physics* 227 (2008) 6071–6105.
- 634 [57] K. Grella, C. Schwab, Sparse tensor spherical harmonics approximation in radiative transfer, *Journal of*  
635 *Computational Physics* 230 (2011) 8452–8473.
- 636 [58] D. A. Knoll, D. E. Keyes, Jacobian-free Newton–Krylov methods: a survey of approaches and applications,  
637 *Journal of Computational Physics* 193 (2004) 357–397.
- 638 [59] W. F. Godoy, X. Liu, Parallel Jacobian-free Newton–Krylov solution of the discrete ordinates method with  
639 flux limiters for 3D radiative transfer, *Journal of Computational Physics* 231 (2012) 4257–4278.
- 640 [60] W. Bangerth, R. Hartmann, G. Kanschat, deal.II: a general-purpose object-oriented finite element library,  
641 *ACM Transactions on Mathematical Software* 33 (2007).
- 642 [61] A. Logg, K.-A. Mardal, G. Wells, Automated solution of differential equations by the finite element method:  
643 the FEniCS book, volume 84, Springer Science & Business Media, 2012.
- 644 [62] F. Hecht, New development in FreeFem++, *Journal of Numerical Mathematics* 20 (2012) 251–266.
- 645 [63] H. A. Van der Vorst, Iterative Krylov methods for large linear systems, volume 13, Cambridge University  
646 Press, 2003.
- 647 [64] S. Balay, S. Abhyankar, M. F. Adams, J. Brown, P. Brune, K. Buschelman, L. Dalcin, V. Eijkhout, W. D.  
648 Gropp, D. Kaushik, M. G. Knepley, L. C. McInnes, K. Rupp, B. F. Smith, S. Zampini, H. Zhang, PETSc  
649 web page, 2019. URL: <http://www.mcs.anl.gov/petsc>.
- 650 [65] G. Goumas, K. Kourtis, N. Anastopoulos, V. Karakasis, N. Koziris, Performance evaluation of the sparse  
651 matrix–vector multiplication on modern architectures, *The Journal of Supercomputing* 50 (2009) 36–77.
- 652 [66] X.-C. Cai, M. Sarkis, A restricted additive Schwarz preconditioner for general sparse linear systems, *SIAM*  
653 *Journal on Scientific Computing* 21 (1999) 792–797.
- 654 [67] K. Stüben, A review of algebraic multigrid, in: *Partial Differential Equations*, Elsevier, 2001, pp. 281–309.
- 655 [68] D. A. May, J. Brown, L. Le Pourhiet, A scalable, matrix-free multigrid preconditioner for finite element  
656 discretizations of heterogeneous Stokes flow, *Computer Methods in Applied Mechanics and Engineering*  
657 290 (2015) 496–523.
- 658 [69] O. Axelsson, R. Blaheta, M. Neytcheva, Preconditioning of boundary value problems using elementwise  
659 Schur complements, *SIAM Journal on Matrix Analysis and Applications* 31 (2009) 767–789.
- 660 [70] Y. Saad, M. H. Schultz, GMRES: a generalized minimal residual algorithm for solving nonsymmetric linear  
661 systems, *SIAM Journal on Scientific and Statistical Computing* 7 (1986) 856–869.
- 662 [71] E. W. Larsen, Diffusion-synthetic acceleration methods for discrete-ordinates problems, *Transport Theory*  
663 *and Statistical Physics* 13 (1984) 107–126.
- 664 [72] T. S. Haut, B. S. Southworth, P. G. Maginot, V. Z. Tomov, Diffusion synthetic acceleration preconditioning  
665 for discontinuous galerkin discretizations of  $s_n$  transport on high-order curved meshes, *Journal on Scientific*  
666 *Computing* 42 (2020) B1271–B1301.
- 667 [73] V. E. Henson, U. M. Yang, *BoomerAMG*: A parallel algebraic multigrid solver and preconditioner, *Applied*  
668 *Numerical Mathematics* 41 (2002) 155–177.
- 669 [74] M. Hanuš, J. C. Ragusa, M. Hackemack, A study of various thermal upscattering acceleration schemes  
670 for massively parallel transport sweeps, in: *International Conference on Mathematics & Computational*  
671 *Methods Applied to Nuclear Science & Engineering*, 2017.
- 672 [75] M. A. Badri, Efficient finite element strategies for solving the radiative transfer equation, Ph.D. thesis,  
673 Université de Nantes, 2018.
- 674 [76] K. Kobayashi, N. Sugimura, Y. Nagaya, 3D radiation transport benchmark problems and results for simple  
675 geometries with void region, *Progress in Nuclear Energy* 39 (2001) 119–144.

- 676 [77] P. N. Brown, B. Chang, U. R. Hanebutte, Spherical harmonic solutions to the 3D Kobayashi benchmark  
677 suite, Technical Report, Lawrence Livermore National Laboratory, 1999.
- 678 [78] I. Zmijarevic, R. Sanchez, Deterministic solutions for 3D Kobayashi benchmarks, Progress in Nuclear  
679 Energy 39 (2001) 207–221.
- 680 [79] Y. Y. Azmy, F. X. Gallmeier, D. A. Lillie, TORT solutions for the 3D radiation transport benchmarks for  
681 simple geometries with void region, Progress in Nuclear Energy 39 (2001) 155–166.
- 682 [80] Y. Nagaya, K. Okumura, T. Mori, M. Nakagawa, MVP/GMVP 2: general purpose Monte Carlo codes for  
683 neutron and photon transport calculations based on continuous energy and multigroup methods, Technical  
684 Report, Japan Atomic Energy Research Institute, 2005.
- 685 [81] G. Karypis, V. Kumar, A fast and high quality multilevel scheme for partitioning irregular graphs, SIAM  
686 Journal on Scientific Computing 20 (1998) 359–392.
- 687 [82] R. Falgout, U. M. Yang, *hypre*: a library of high performance preconditioners, Computational Science—  
688 ICCS 2002 (2002) 632–641.
- 689 [83] A. Bienz, J. Calhoun, L. Olson, M. Snir, W. D. Gropp, Analyzing the performance of a sparse matrix vector  
690 multiply for extreme scale computers, in: The International Conference for High Performance Computing  
691 Networking, Storage, and Analysis, 2015.
- 692 [84] C. Geuzaine, J.-F. Remacle, Gmsh: a 3-D finite element mesh generator with built-in pre- and post-  
693 processing facilities, International Journal for Numerical Methods in Engineering 79 (2009) 1309–1331.
- 694 [85] G. Karypis, V. Kumar, Multilevel  $k$ -way partitioning scheme for irregular graphs, Journal of Parallel and  
695 Distributed computing 48 (1998) 96–129.
- 696 [86] L. G. Henyey, J. L. Greenstein, Diffuse radiation in the galaxy, The Astrophysical Journal 93 (1941) 70–83.
- 697 [87] G. Krishnamoorthy, R. Rawat, P. J. Smith, Parallel computations of radiative heat transfer using the  
698 discrete ordinates method, Numerical Heat Transfer 47 (2004) 19–38.
- 699 [88] D. Le Hardy, Y. Favennec, B. Rousseau, F. Hecht, Specular reflection treatment for the 3D radiative  
700 transfer equation solved with the discrete ordinates method, Journal of Computational Physics 334 (2017)  
701 541–572.
- 702 [89] P. Yang, Q. Cheng, Z. Zhang, Radiative Properties of Ceramic  $\text{Al}_2\text{O}_3$ ,  $\text{AlN}$  and  $\text{Si}_3\text{N}_4$ —II: Modeling,  
703 International Journal of Thermophysics 38 (2017) 1–18.