Eidgenössische Technische Hochschule Zürich

Ecole polytechnique fédérale de Zurich
Politecnico federale di Zurigo
Swiss Federal Institute of Technology Zurich

An efficient sparse finite element solver for the radiative transfer equation

G. Widmer

Research Report No. 2009-11
January 2009

Seminar für Angewandte Mathematik
Eidgenössische Technische Hochschule
CH-8092 Zürich
Switzerland

An efficient sparse finite element solver for the radiative transfer equation

G. Widmer

Abstract

The stationary monochromatic radiative transfer equation (RTE) is posed in five dimensions, with the intensity depending on both a position in a three-dimensional domain as well as a direction. For non-scattering radiative transfer, sparse finite elements [10, 11] have been shown to be an efficient discretization strategy if the intensity function is sufficiently smooth. Compared to the discrete ordinates method, they make it possible to significantly reduce the number of degrees of freedom N in the discretization with almost no loss of accuracy. However, using a direct solver to solve the resulting linear system requires $O(N^3)$ operations. In this paper, an efficient solver based on the conjugate gradient method (CG) with a subspace correction preconditioner is presented. Numerical experiments show that the linear system can be solved at computational costs that are nearly proportional to the number of degrees of freedom N in the discretization.

1 Introduction

It is widely known that radiation is an important mode of heat transfer in participating media at high temperatures. During the last decades, various methods have been developed to solve the RTE (see e.g. [5, 6, 7, 9]). Very popular methods include the P_N-approximation, the finite volume method (FVM), finite element methods (FEM) and in particular the discrete ordinates method (DOM). These methods are all based on a semi-discretization in solid angle with N_S degrees of freedom, which may correspond to spherical harmonics, discrete ordinates or control volumes. Each of the resulting N_S equations is then discretized in physical space with N_D degrees of freedom, leading to $N_S \cdot N_D$ unknowns in total. As both N_S and N_D are the number of degrees of freedom arising from a two- (or three-) dimensional discretization, the computation of the radiative intensity becomes very costly when the meshes are refined.

In [10] and [11], an efficient discretization, the sparse tensor product approximation, was presented. Instead of using $N_S \cdot N_D$ degrees of freedom in the discretization, the number of unknows is proportional to $N_D \log N_S + N_S \log N_D$. This can be achieved with almost no loss of accuracy if the radiation intensity is sufficiently smooth. For functions that do not satisfy the smoothness condition, adaptive sparse finite elements can improve the efficiency of the discretization [11]. In contrast to for example the discrete ordinates method, where the equation is discretized in a first step with respect to solid angle and in a second step with respect to physical space, a scaled least-squares approach similar to [7] is used, where basis and weight functions depend on both direction as well as position in physical space. Another key ingredient is the use of hierarchical basis functions. Instead of standard finite elements, products of hierarchical finite element functions are used to construct the finite element space.

In the papers [10] and [11] mentioned above, however, an essential aspect was missing, namely the discussion of the computational costs that are required to compute the intensity in the new discretization. Clearly, the significant reduction of the number of unknowns by itself reduces the computational costs compared to a standard discretization even if the full stiffness

matrix of the finite element discretization was set up in the most straightforward way. However, if a standard direct solver is used, the computational costs are proportional to up to the third power of the number of unknowns in the discretization and as the matrix is very ill-conditioned, using iterative methods without preconditioning is also not a viable option.

This paper focuses on efficient algorithms that make it possible to solve the linear system arising from the sparse tensor product approximation at computational costs that are almost proportional to the number of degrees of freedom in the discretization. With this method, the RTE can be solved with high accuracy at affordable computational costs.

Until now, this method has only been analysed and tested for the monochromatic RTE in two dimensions without scattering for fully absorbing cold walls as boundary conditions.

$$\begin{aligned} \mathbf{s} \cdot \boldsymbol{\nabla}_{\mathbf{x}} I(\mathbf{x}, \mathbf{s}) + \kappa(\mathbf{x}) I(\mathbf{x}, \mathbf{s}) &= \kappa(\mathbf{x}) I_b(\mathbf{x}),\ (\mathbf{x}, \mathbf{s}) \in D \times S^2 \quad (1) \\ I(\mathbf{x}, \mathbf{s}) &= 0 \text{ for } \mathbf{x} \in \partial D,\ \mathbf{s} \cdot \mathbf{n}(\mathbf{x}) < 0 \quad (2) \end{aligned}$$

Here $\mathbf{x}$ is a position in a two-dimensional domain D, $\mathbf{s}$ a direction in solid angle, $I(\mathbf{x}, \mathbf{s})$ the radiation intensity, $I_b(\mathbf{x})$ the blackbody intensity, $\kappa(\mathbf{x})$ the absorption coefficient, which is assumed to be uniformly bounded from below and above ($0 < \kappa_0 \leq \kappa \leq \kappa_1 < \infty$), and $\mathbf{n}(\mathbf{x})$ the outer unit normal to D. As the RTE is simplified from a three-dimensional to a two-dimensional domain, the intensity is assumed to be constant with respect to the z-coordinate, i.e. $\frac{\partial I}{\partial z} = 0$.

The adaptation of the algorithms to three space dimensions is rather straightforward, while the analysis of the method for problems that take into account scattering effects is open. Both topics will be subjects for future research.

The paper is structured as follows. Section 2 contains a summary of the sparse tensor product discretization as described in [10]. In section 3 the CG-solver with the algorithms for efficient matrix-vector multiplication and preconditioning are presented, while numerical convergence results of the preconditioner are discussed in section 4. Section 5 contains a summary of the results and conclusions.

2 Sparse Tensor Product Discretization

The sparse tensor product discretization is an adaptation of sparse grids [3, 12] for the RTE. The discretization is based on two families of hierarchical basis sets. For the construction of these basis sets, the domains D and S^2 are equipped with diadically refined mesh hierarchies $\mathcal{T}_D^l$ and $\mathcal{T}_S^l$, $l = 0, 2, ..., L$, up to a given level L. Based on these meshes, hierarchical spaces of continuous, piecewise linear functions on $\mathcal{T}_D^L$ and discontinuous, piecewise constant functions on $\mathcal{T}_S^L$ are constructed. In physical space, the hierarchical basis of hat functions [4] is used, while in solid angle, the space is constructed with $L^2(S^2)$-orthogonal Haar wavelets [11].

Let

$$\phi_i^{I_i}(\mathbf{x}),\ 0 \leq i \leq L, I_i = 1, \ldots, N_D^i, \quad \text{and} \quad \psi_j^{J_j}(\mathbf{s}),\ 0 \leq j \leq L, J_j = 1, \ldots, N_S^j,$$

denote the set of basis functions in physical space and solid angle, respectively, up to level L. The indices i and j refer to the level of a basis function and the indices I_i and J_j are the level index on a given level i and j, respectively. N_D^i and N_S^j are the number of basis functions on level i in physical space and on level j in solid angle, respectively. As both D and S^2 are two-dimensional domains,

$$N_D^i = O(4^i) \quad \text{and} \quad N_S^j = O(4^j). \quad (3)$$

The intensity function $I(\mathbf{s},\mathbf{x})$ is discretized with a linear combination of those product functions of which the *sum of the levels $i+j$ is smaller or equal to the maximum level L*

$$I(\mathbf{x},\mathbf{s}) \;=\; \sum_{i=0}^{L}\sum_{j=0}^{L-i}\sum_{I_i=1}^{N_D^i}\sum_{J_j=1}^{N_S^j} I_{i,j}^{I_i,J_j}\phi_i^{I_i}(\mathbf{x})\psi_j^{J_j}(\mathbf{s}) \tag{4}$$

This restriction to a *subset* of all product basis functions reduces the number of unknows $I_{i,j}^{I_i,J_j}$ from $N_D \cdot N_S$ to a number $\mathcal{N}_L$ that is proportional to $N_D \log N_S + N_S \log N_D$.

Inserting the ansatz (4) into the weighted-residual (or weighted least-squares) variational formulation for (1)

$$\underbrace{\int_D\int_{S^2} \left(\mathbf{s}\cdot\boldsymbol{\nabla}_{\mathbf{x}} I(\mathbf{s},\mathbf{x}) + \kappa(\mathbf{x})I(\mathbf{s},\mathbf{x})\right) W_{i,j}^{I_i,J_j}(\mathbf{s},\mathbf{x})\; d\mathbf{s}\,d\mathbf{x}}_{=:a(I,W_{i,j}^{I_i,J_j})} = \underbrace{\int_D\int_{S^2} \kappa(\mathbf{x})I_b(\mathbf{x})W_{i,j}^{I_i,J_j}(\mathbf{s},\mathbf{x})\; d\mathbf{s}\,d\mathbf{x}}_{=:f(W_{i,j}^{I_i,J_j})} \tag{5}$$

with weighting functions

$$W_{i,j}^{I_i,J_j}(\mathbf{x},\mathbf{s}) \;=\; \frac{1}{\kappa(\mathbf{x})}\left(\mathbf{s}\cdot\boldsymbol{\nabla}_{\mathbf{x}}\phi_i^{I_i}(\mathbf{x})\psi_j^{J_j}(\mathbf{s}) + \kappa(\mathbf{x})\phi_i^{I_i}(\mathbf{x})\psi_j^{J_j}(\mathbf{s})\right) \tag{6}$$

leads to a symmetric, positive definite linear system in the unknowns

$$I_{i,j}^{I_i,J_j},\; i=0,\dots L,\; j=0,\dots,L-i,\; I_i=1,\dots,N_D^i,\; J_j=1,\dots,N_S^j.$$

Using a level first ordering, these unknows can be stored in the upper left part of a block matrix as displayed for $L=2$

$$\mathbf{I} = \begin{pmatrix} \mathcal{I}_{0,0} & \mathcal{I}_{0,1} & \mathcal{I}_{0,2} \\ \mathcal{I}_{1,0} & \mathcal{I}_{1,1} & 0 \\ \mathcal{I}_{2,0} & 0 & 0 \end{pmatrix} \quad \text{with } \mathcal{I}_{i,j} = \begin{pmatrix} I_{i,j}^{0,0} & \cdots & I_{i,j}^{0,N_D^j} \\ \vdots & \ddots & \vdots \\ I_{i,j}^{N_S^i,0} & \cdots & I_{i,j}^{N_S^i,N_D^j} \end{pmatrix} \tag{7}$$

3 Preconditioned CG-Algorithm

Due to the hierarchical basis functions, the full stiffness matrix from the discretization (4)-(6) is ill-conditioned with the $O(L^2\cdot\mathcal{N}_L)$ matrix entries. Using a standard direct linear solver would lead to computational costs proportional to up to $(\mathcal{N}_L)^3$. An alternative is the use of an iterative method. As the matrix is symmetric and positive definite, the CG-algorithm is the straightforward method of choice. In order to obtain an overall computational effort that scales like the number of unknowns $\mathcal{N}_L$ in the discretization, the number of CG-iterations has to be independent of the refinement level L and the number of operations in the matrix-vector multiplication has to be proportional to $\mathcal{N}_L$.

3.1 Matrix-Vector Multiplication

The matrix-vector multiplication can be carried out with optimal complexity by taking advantage of the fact that the bilinear form of the variational formulation (5) separates in the sense that the integral over the product space can be written as a sum of products of integrals over S^2 and D.

$$a(\phi_i^{I_i}\cdot\psi_j^{J_j},\phi_k^{I_k}\cdot\psi_l^{J_l}) = \sum_{m=1}^{9} b^m(\phi_i^{I_i},\phi_k^{I_k})\cdot c^m(\psi_j^{J_j},\psi_l^{J_l}) \tag{8}$$

For every $m \in \{1,..,9\}$ and $i,j,k,l \in \{1,...,L\}$ the matrices $\mathcal{B}_{ik}^m \in \mathbb{R}^{N_D^i \times N_D^k}$ and $\mathcal{C}_{jl}^m \in \mathbb{R}^{N_S^j \times N_S^l}$ are defined as the stiffness matrices from the bilinear forms $b^m(\phi_i^{I_i}, \phi_k^{I_k})$ and $c^m(\psi_j^{J_j}, \psi_k^{J_k})$ with matrix elements

$$(\mathcal{B}_{ik}^m)_{I_i,I_k} = b^m(\phi_i^{I_i}, \phi_k^{I_k}),\ I_i \le N_D^i,\ I_k \le N_D^k \qquad \left(\mathcal{C}_{jl}^m\right)_{J_j,I_l} = c^m(\psi_j^{J_j}, \psi_l^{J_l}),\ J_j \le N_S^j,\ J_l \le N_S^l \quad (9)$$

The matrix-vector multiplication for the matrix from the discretization (4)-(6)

$$\tilde{I}_{k,l}^{I_k,J_l} = \sum_{\substack{i+j\le L \\ I_i \le N_D^i, J_j \le N_S^j}} \mathcal{A}_{i,j,k,l}^{I_i,J_j,I_k,I_l} I_{i,j}^{I_i,J_j}, \qquad k+l \le L,\ I_k \le N_D^k,\ I_l \le N_S^l \quad (10)$$

with

$$\mathcal{A}_{i,j,k,l}^{I_i,J_j,I_k,I_l} := a(\phi_i^{I_i} \cdot \psi_j^{J_j}, \phi_k^{I_k} \cdot \psi_l^{J_l}) \quad (11)$$

can then be written as a block matrix multiplication (shown for $L = 2$)

$$\begin{pmatrix} \tilde{\mathcal{I}}_{0,0} & \tilde{\mathcal{I}}_{0,1} & \tilde{\mathcal{I}}_{0,2} \\ \tilde{\mathcal{I}}_{1,0} & \tilde{\mathcal{I}}_{1,1} & 0 \\ \tilde{\mathcal{I}}_{2,0} & 0 & 0 \end{pmatrix} \leftarrow \sum_{m=1}^{9} \underbrace{\begin{pmatrix} \mathcal{B}_{0,0}^m & \mathcal{B}_{0,1}^m & \mathcal{B}_{0,2}^m \\ \mathcal{B}_{1,0}^m & \mathcal{B}_{1,1}^m & \mathcal{B}_{1,2}^m \\ \mathcal{B}_{2,0}^m & \mathcal{B}_{2,1}^m & \mathcal{B}_{2,2}^m \end{pmatrix}^T}_{=:\left(\mathcal{B}_2^m\right)^T} \underbrace{\begin{pmatrix} \mathcal{I}_{0,0} & \mathcal{I}_{0,1} & \mathcal{I}_{0,2} \\ \mathcal{I}_{1,0} & \mathcal{I}_{1,1} & 0 \\ \mathcal{I}_{2,0} & 0 & 0 \end{pmatrix}}_{=:\mathcal{I}_2} \underbrace{\begin{pmatrix} \mathcal{C}_{0,0}^m & \mathcal{C}_{0,1}^m & \mathcal{C}_{0,2}^m \\ \mathcal{C}_{1,0}^m & \mathcal{C}_{1,1}^m & \mathcal{C}_{1,2}^m \\ \mathcal{C}_{2,0}^m & \mathcal{C}_{2,1}^m & \mathcal{C}_{2,2}^m \end{pmatrix}}_{=:\mathcal{C}_2^m} \quad (12)$$

Here, '$\leftarrow$' denotes the partial assignment, where only values that correspond to degrees of freedom in the sparse tensor product space are set.

A straightforward multiplication of the matrices on the right would require additional storage space in the lower right part of the block matrix where the values in the intensity matrix $\mathcal{I}_L$ are zero for the intermediate state. As this would destroy the desired complexity, following the sparse grids multiplication strategy [2], the last matrix is split into an upper part and a lower part.

$$\begin{pmatrix} \mathcal{B}_{0,0}^m & \mathcal{B}_{0,1}^m & \mathcal{B}_{0,2}^m \\ \mathcal{B}_{1,0}^m & \mathcal{B}_{1,1}^m & \mathcal{B}_{1,2}^m \\ \mathcal{B}_{2,0}^m & \mathcal{B}_{2,1}^m & \mathcal{B}_{2,2}^m \end{pmatrix}^T \begin{pmatrix} \mathcal{I}_{0,0} & \mathcal{I}_{0,1} & \mathcal{I}_{0,2} \\ \mathcal{I}_{1,0} & \mathcal{I}_{1,1} & 0 \\ \mathcal{I}_{2,0} & 0 & 0 \end{pmatrix} \begin{pmatrix} \mathcal{C}_{0,0}^m & \mathcal{C}_{0,1}^m & \mathcal{C}_{0,2}^m \\ \mathcal{C}_{1,0}^m & \mathcal{C}_{1,1}^m & \mathcal{C}_{1,2}^m \\ \mathcal{C}_{2,0}^m & \mathcal{C}_{2,1}^m & \mathcal{C}_{2,2}^m \end{pmatrix} \quad (13)$$

$$= \begin{pmatrix} \mathcal{B}_{0,0}^m & \mathcal{B}_{0,1}^m & \mathcal{B}_{0,2}^m \\ \mathcal{B}_{1,0}^m & \mathcal{B}_{1,1}^m & \mathcal{B}_{1,2}^m \\ \mathcal{B}_{2,0}^m & \mathcal{B}_{2,1}^m & \mathcal{B}_{2,2}^m \end{pmatrix}^T \begin{pmatrix} \mathcal{I}_{0,0} & \mathcal{I}_{0,1} & \mathcal{I}_{0,2} \\ \mathcal{I}_{1,0} & \mathcal{I}_{1,1} & 0 \\ \mathcal{I}_{2,0} & 0 & 0 \end{pmatrix} \begin{pmatrix} \mathcal{C}_{0,0}^m & \mathcal{C}_{0,1}^m & \mathcal{C}_{0,2}^m \\ 0 & \mathcal{C}_{1,1}^m & \mathcal{C}_{1,2}^m \\ 0 & 0 & \mathcal{C}_{2,2}^m \end{pmatrix} \quad (14)$$

$$+ \begin{pmatrix} \mathcal{B}_{0,0}^m & \mathcal{B}_{0,1}^m & \mathcal{B}_{0,2}^m \\ \mathcal{B}_{1,0}^m & \mathcal{B}_{1,1}^m & \mathcal{B}_{1,2}^m \\ \mathcal{B}_{2,0}^m & \mathcal{B}_{2,1}^m & \mathcal{B}_{2,2}^m \end{pmatrix}^T \begin{pmatrix} \mathcal{I}_{0,0} & \mathcal{I}_{0,1} & \mathcal{I}_{0,2} \\ \mathcal{I}_{1,0} & \mathcal{I}_{1,1} & 0 \\ \mathcal{I}_{2,0} & 0 & 0 \end{pmatrix} \begin{pmatrix} 0 & 0 & 0 \\ \mathcal{C}_{1,0}^m & 0 & 0 \\ \mathcal{C}_{2,0}^m & \mathcal{C}_{2,1}^m & 0 \end{pmatrix} \quad (15)$$

The upper part (14) is evaluated by first partially multiplying the last two matrices

$$\begin{pmatrix} \widehat{\mathcal{I}}_{0,0} & \widehat{\mathcal{I}}_{0,1} & \widehat{\mathcal{I}}_{0,2} \\ \widehat{\mathcal{I}}_{1,0} & \widehat{\mathcal{I}}_{1,1} & 0 \\ \tilde{\mathcal{I}}_{2,0} & 0 & 0 \end{pmatrix} \leftarrow \begin{pmatrix} \mathcal{I}_{0,0} & \mathcal{I}_{0,1} & \mathcal{I}_{0,2} \\ \mathcal{I}_{1,0} & \mathcal{I}_{1,1} & 0 \\ \mathcal{I}_{2,0} & 0 & 0 \end{pmatrix} \begin{pmatrix} \mathcal{C}_{0,0}^m & \mathcal{C}_{0,1}^m & \mathcal{C}_{0,2}^m \\ 0 & \mathcal{C}_{1,1}^m & \mathcal{C}_{1,2}^m \\ 0 & 0 & \mathcal{C}_{2,2}^m \end{pmatrix} \quad (16)$$

before partially carrying out the first multiplication

$$\begin{pmatrix} \tilde{\mathcal{I}}_{0,0} & \tilde{\mathcal{I}}_{0,1} & \tilde{\mathcal{I}}_{0,2} \\ \tilde{\mathcal{I}}_{1,0} & \tilde{\mathcal{I}}_{1,1} & 0 \\ \tilde{\mathcal{I}}_{2,0} & 0 & 0 \end{pmatrix} \leftarrow \begin{pmatrix} \mathcal{B}_{0,0}^m & \mathcal{B}_{0,1}^m & \mathcal{B}_{0,2}^m \\ \mathcal{B}_{1,0}^m & \mathcal{B}_{1,1}^m & \mathcal{B}_{1,2}^m \\ \mathcal{B}_{2,0}^m & \mathcal{B}_{2,1}^m & \mathcal{B}_{2,2}^m \end{pmatrix}^T \begin{pmatrix} \widehat{\mathcal{I}}_{0,0} & \widehat{\mathcal{I}}_{0,1} & \widehat{\mathcal{I}}_{0,2} \\ \widehat{\mathcal{I}}_{1,0} & \widehat{\mathcal{I}}_{1,1} & 0 \\ \tilde{\mathcal{I}}_{2,0} & 0 & 0 \end{pmatrix} \quad (17)$$

For the lower part (15), the first two matrices are partially multiplied before evaluating the second matrix multiplication.

As hierarchical bases are used, the block matrices $\mathcal{B}_L^m$ and $\mathcal{C}_L^m$ of level L have $O(L\cdot 4^L)$ matrix entries and the partial matrix-vector multiplication requires $O(L^2 4^L) = O(L\mathcal{N}_L)$ operations. In order to obtain computational costs that are proportional to $\mathcal{N}_L$, basis transformations between hierarchical basis functions and standard basis functions are used.

$$\phi_i^{I_i}(\mathbf{x}) = \sum_{i'=0}^{N_D^0+\ldots+N_D^{l_D}} \left(\mathcal{R}^{i,l_D}\right)_{I_i,i'} \varphi_{i'}^{l_D}(\mathbf{x}), \quad \psi_j^{J_j}(\mathbf{s}) = \sum_{j'=0}^{N_S^0+\ldots+N_S^{l_S}} \left(\mathcal{S}^{j,l_S}\right)_{J_j,j'} \chi_{j'}^{l_S}(\mathbf{s}) \tag{18}$$

Here, $\varphi_{i'}^{l_D}(\mathbf{x}),\ i' \in \{1,...,N_D^0+...+N_D^{l_D}\}$ is the basis of standard finite element hat functions of level l_D on $\mathcal{T}_D^{l_D}$, $\chi_{j'}^{l_S}(\mathbf{s}), j' \in \{1,...,N_S^0+...+N_S^{l_S}\}$ the basis of piecewise constant functions of level l_S on $\mathcal{T}_S^{l_S}$, $\mathcal{R}^{i,l_D}$ is the matrix of basis transform from the standard basis of level l_D to the hierarchical basis on level i in physical space and $\mathcal{S}^{j,l_S}$ the matrix of basis transform from the standard basis of level l_S to the Haar basis on level j in solid angle. The submatrices $\mathcal{B}_{ik}^m$ and $\mathcal{C}_{jl}^m$ can then be written as

$$\mathcal{B}_{ik}^m = \mathcal{R}^{i,l_D}\mathbf{B}^{m,l_D}\left(\mathcal{R}^{k,l_D}\right)^T,\ i,k \le l_D, \quad \text{and} \quad \mathcal{C}_{jl}^m = \mathcal{S}^{j,l_S}\mathbf{C}^{m,l_S}\left(\mathcal{S}^{l,l_S}\right)^T,\ j,l \le l_S, \tag{19}$$

where $\mathbf{B}^{m,l_D}$ and $\mathbf{B}^{m,l_S}$ are the standard finite element matrices with matrix entries

$$\left(\mathbf{B}^{m,l_D}\right)_{i',k'} = b^m(\phi_{i'}^{l_D}, \phi_{k'}^{l_D}), \qquad \left(\mathbf{C}^{m,l_S}\right)_{j',l'} = c^m(\chi_{j'}^{l_S}, \chi_{l'}^{l_S}) \tag{20}$$

and the block matrices up to level l_D and l_S, respectively, can be written as

$$\mathcal{B}_{l_D}^m = \mathcal{R}^{l_D}\mathbf{B}^{m,l_D}\left(\mathcal{R}^{l_D}\right)^T \quad \text{and} \quad \mathcal{C}_{l_S}^m = \mathcal{S}^{l_S}\mathbf{C}^{m,l_S}\left(\mathcal{S}^{l_S}\right)^T \tag{21}$$

with

$$\mathcal{R}^{l_D} := \begin{pmatrix} \mathcal{R}^{0,l_D} \\ \vdots \\ \mathcal{R}^{l_D,l_D} \end{pmatrix} \quad \text{and} \quad \mathcal{S}^{l_S} := \begin{pmatrix} \mathcal{S}^{0,l_S} \\ \vdots \\ \mathcal{S}^{l_S,l_S} \end{pmatrix} \tag{22}$$

For fixed level j and index J_j in solid angle, with $\mathbf{I}_j^{J_j}$ denoting the corresponding column vector up to level $L-j$ of the coefficient matrix $\mathcal{I}_L$, the operation $\tilde{\mathbf{I}}_j^{J_j} = \left(\mathcal{B}_{L-j}^m\right)^T \mathbf{I}_j^{J_j}$ can be written as

$$\tilde{\mathbf{I}}_j^{J_j} = \left(\mathcal{B}_{L-j}^m\right)^T \mathbf{I}_j^{J_j} = \mathcal{R}^{l_{L-j}}\left(\mathbf{B}^{m,L-j}\right)^T\left(\mathcal{R}^{l_{L-j}}\right)^T \mathbf{I}_j^{J_j} \tag{23}$$

The matrix-vector multiplication $\left(\mathcal{R}^{l_{L-j}}\right)^T \mathbf{I}_j^{J_j}$ (and analogously for $\mathcal{R}^{l_{L-j}}$) can be carried out at computational costs proportional to the length of the vector $\mathbf{I}_j^{J_j}$, using standard hierarchical reconstruction techniques [4]. As the finite element matrices with respect to standard basis are sparse, the matrix-vector multiplication (23) requires a computational effort proportional to the number of elements in $\mathbf{I}_j^{J_j}$ and the partial matrix-vector multiplication as in (17) requires $O(\mathcal{N}_L)$ operations. With analogous arguments for the solid angle matrices it is easy to see that also the partial matrix multiplication with respect to solid angle as in (16) can be computed with the same complexity which makes it possible to compute the matrix-vector multiplication for the linear system (10) with the optimal complexity of $O(\mathcal{N}_L)$ operations.

3.2 Preconditioning

In order to be able to compute the intensity at computational costs that are proportional to the number of degrees of freedom in the discretization $\mathcal{N}_L$, the number of iteration steps in the CG-solver must be independent of the refinement level. Without preconditioning, numerical experiments show that the number of iteration steps to solve the linear system grows rapidly when the discretization is refined (Figs. 4 and 5). Therefore, an efficient preconditioner is crucial in order to obtain the desired complexity of the solver.

A preconditioner for a linear system

$$\mathbf{Ax} = \mathbf{b} \tag{24}$$

is a regular linear operator $\mathbf{M}$ that is used to transform (24) into an equivalent linear system

$$\mathbf{MAx} = \mathbf{Mb} \tag{25}$$

For a preconditioner to be efficient it has to satisfy two criteria:

1. For any vector $\mathbf{y}$ the matrix-vector multiplication $\mathbf{My}$ has to be easy to compute, optimally at computational costs proportional to the dimension of $\mathbf{y}$.

2. $\mathbf{MA}$ should be close the identity in the sense that the condition number of $\mathbf{MA}$ is independent of the refinement level and close to the identity matrix as this guarantees the number of CG-iterations to be small and bounded by a constant that is independent of the refinement level.

Numerical experiments show that using a multiplicative subspace preconditioner is a successful strategy for the linear system from (4)-(6). Similarly to sparse grids preconditioners for elliptic problems [1], the sparse tensor product space is divided into (L+1) *overlapping* subspaces (see Fig. 1).

$$\widehat{V}_L = \sum_{l_D+l_S=L} V_{l_D,l_S}, \tag{26}$$

where the space V_{l_D,l_S} contains all product basis functions up to level l_D in physical space and up to level l_S in solid angle. However, as the RTE is a hyperbolic problem, some adaptations of the preconditioner are required.

The preconditioner is based on the following idea: Each of these subspaces V_{l_D,l_S} corresponds to a discrete ordinates discretization of a level l_D in physical space and a level $l_S = L - l_D$ in solid angle. If the standard discretization was used with hat functions of level l_D in physical space and characteristic functions of level l_S in solid angle, the RTE could be solved separately for each spherical triangle using an algebraic multigrid solver (AMG) [8]. As the relative complexity of the AMG algorithms is nearly independent of the number of degrees of freedom in the discretization, the computational costs for solving the RTE on such a subsystem with standard discretizations is nearly proportional to the number of degrees of freedom in the subspace.

The linear operator that is used as a preconditioner $\mathbf{M}$ for the linear system from the sparse tensor product discretization is a multiplicative subspace correction preconditioner that iteratively computes corrections on the subspaces $V_{l_D,L-l_D},\ l_D = 0, ..., L$, and adds them to the current approximation (algorithm 1).

In the sparse tensor product discretization the subspaces $V_{l_D,L-l_D}$ are discretized with hierarchical bases instead of standard bases as required for the AMG-solver. Therefore, after restricting the residual $\mathbf{r} = \mathbf{b} - \mathbf{Ax}$ to the subspace $V_{l_D,L-l_D}$, it has to be transformed into

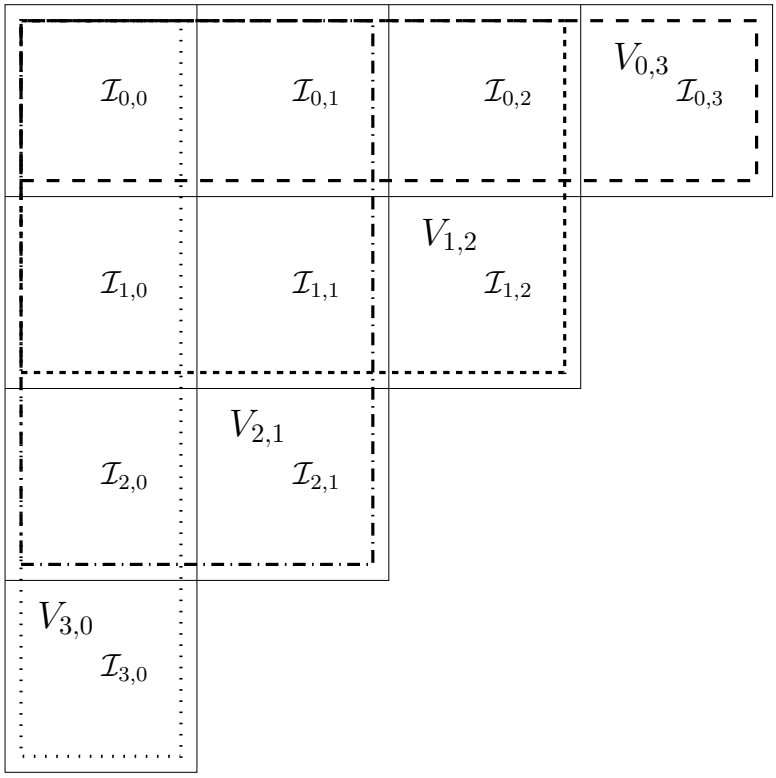

Figure 1: The subspace V_{l_D,l_S} contains all degrees of freedom up to level l_D in physical space and up to level l_S in solid angle. The figure shows the subspaces V_{l_D,l_S} for $L = 3$.

Algorithm 1 Subspace Correction Preconditioner $\mathbf{x} \leftarrow \mathbf{My}$

$\mathbf{x} \leftarrow \mathbf{0}$
for $l_S = 0$ to L {Loop over levels in solid angle} **do**
  $l_D = L - l_D$ {Level in physical space}
  $\mathbf{r} \leftarrow \mathbf{y} - \mathbf{Ax}$
  $\mathbf{c}_{l_D,l_S} \leftarrow \mathit{Subspace\_Solve}(l_D, l_S, \mathbf{r})$ { Subspace correction on V_{l_D,l_S}}
  $\mathbf{x} \leftarrow \mathbf{x} + \mathbf{c}_{l_D,l_S}$
end for
for $l_D = 0$ to L {Loop over levels in physical space} **do**
  $l_S = L - l_D$ {Level in solid angle}
  $\mathbf{r} \leftarrow \mathbf{y} - \mathbf{Ax}$
  $\mathbf{c}_{l_D,l_S} \leftarrow \mathit{Subspace\_Solve}(l_D, l_S, \mathbf{r})${ Subspace correction on V_{l_D,l_S}, algorithm 2}
  $\mathbf{x} \leftarrow \mathbf{x} + \mathbf{c}_{l_D,l_S}$
end for
return $\mathbf{x}$

Algorithm 2 Subspace_Solve $\mathbf{c}_{l_D,l_S} \leftarrow \mathit{Subspace\_Solve}(l_D, l_S, \mathbf{r})$

$\mathbf{r}_{l_D,l_S} \leftarrow \mathbf{r}$ {Restriction of residual to levels (l_D, l_S)}
$\tilde{\mathbf{r}} \leftarrow \mathbf{r}_{l_D,l_S} \left(\mathcal{S}^{l_S}\right)^{-T}$ {Transformation to standard representation in solid angle}
for $i = 0$ to $\sum_{j=0}^{l_S} \mathcal{N}_i$ {Loop over triangles in solid angle} **do**
  $\mathbf{r}_i^S \leftarrow \left(\mathcal{R}^{l_D}\right)^{-1} \tilde{\mathbf{r}}_i$ {Transformation to standard representation in physical space}
  $\mathbf{c}_i^S \leftarrow \left(\sum_{m=1}^{9} \left(B_{l_D}^m\right)^T \left(C_{l_S}^m\right)_{i,i}\right)^{-1} \mathbf{r}_i^S$ {Solving the linear system using AMG}
  $\tilde{\mathbf{c}}_i \leftarrow \left(\mathcal{R}^{l_D}\right)^{-T} \mathbf{c}_i^S$ {Transformation to hierarchical representation in physical space}
end for
$\mathbf{c}_{l_D,l_S} \leftarrow \tilde{\mathbf{c}} \left(\mathcal{S}^{l_S}\right)^{-1}$ {Transformation to hierarchical representation in solid angle}
return $\mathbf{c}_{l_D,l_S}$

standard bases with respect to space as well as solid angle. After solving the linear subsystem with respect to the standard bases, the correction has to be transformed back into hierarchical representation $\mathbf{c}_{l_D,l_S}$. In terms of matrices, the linear system is transformed into a linear system with respect to the standard bases using basis transformation matrices (22)

$$\sum_{m=1}^{9} \left(\mathcal{B}_{l_D}^m\right)^T \mathbf{c}_{l_D,l_S} \mathcal{C}_{l_S}^m = \mathbf{r}_{l_D,l_S} \tag{27}$$

$$\mathcal{R}^{l_D} \sum_{m=1}^{9} \left(\left(\mathbf{B}_{l_D}^m\right)^T \left(\mathcal{R}^{l_D}\right)^T \mathbf{c}_{l_D,l_S} S^{l_S} \mathbf{C}^{m,l_S}\right) \left(S^{l_S}\right)^T = \mathbf{r}_{l_D,l_S} \tag{28}$$

$$\sum_{m=1}^{9} \left(\left(\mathbf{B}_{l_D}^m\right)^T \underbrace{\left(\mathcal{R}^{l_D}\right)^T \mathbf{c}_{l_D,l_S} S^{l_S}}_{=:\mathbf{c}_{l_D,l_S}^S} \mathbf{C}^{m,l_S} \right) = \underbrace{\left(\mathcal{R}^{l_D}\right)^{-1} \mathbf{r}_{l_D,l_S} \left(S^{l_S}\right)^{-T}}_{\mathbf{r}_{l_D,l_S}^S} \tag{29}$$

$$\sum_{m=1}^{9} \left(\left(\mathbf{B}_{l_D}^m\right)^T \mathbf{c}_{l_D,l_S}^S \mathbf{C}^{m,l_S}\right) = \mathbf{r}_{l_D,l_S}^S \tag{30}$$

Here, $\mathbf{r}_{l_D,l_S}^S$ and $\mathbf{c}_{l_D,l_S}^S$ denote the residual and the correction matrix with respect to the standard bases.

This transformation requires that the inverse of the operator $\mathcal{R}^{l_D}$ from the left and the inverse of $\left(\mathcal{S}^{l_S}\right)^T$ from the right are applied to the residual matrix in order to obtain the residual in standard bases and, after the linear system has been solved in standard bases, analogous transformations have to be carried out to obtain the correction matrix in hierarchical representation.

As the hierarchical basis functions in solid angle are $L^2(S^2)$- orthogonal, the linear operator S^{l_S} satisfies

$$\left(\mathcal{S}^{l_S}\right)^T \mathcal{S}^{l_S} = D \quad \Rightarrow \quad \left(\mathcal{S}^{l_S}\right)^{-1} = \left(\mathcal{S}^{l_S}\right)^T D, \tag{31}$$

where D is a diagonal matrix with positive matrix entries. Therefore, the inverse operators $\left(\mathcal{S}^{l_S}\right)^{-1}$ and $\left(\mathcal{S}^{l_S}\right)^{-T}$ can be applied at computational costs proportional to the number of degrees of freedom in V_{l_D,l_S}.

In physical space, the operators $\left(\mathcal{R}^{l_D}\right)^{-1}$ and $\left(\mathcal{R}^{l_D}\right)^{-T}$ can be applied to the subspace coefficients by using the standard decomposition strategy [4] for the hierarchical basis at computational costs proportional to the number of degrees of freedom in V_{l_D,l_S}.

As the total sum of the degrees of freedom in the $L+1$ subspaces V_{l_D,l_S} is bounded by $2\mathcal{N}_L$,

$$\sum_{l=0}^{L} V_{l,L-l} = \sum_{l=0}^{L} \mathcal{N}_l \leq 2\mathcal{N}_L, \tag{32}$$

the relative complexity of the preconditioner is nearly independent of the refinement level L.

4 Numerical Results and Discussion

The computational effort in each CG-iteration step is nearly proportional to the number of degrees of freedom $\mathcal{N}_L$ in the sparse tensor product discretization. The efficiency of the sparse tensor product solver therefore depends on the quality of the preconditioner. An optimal preconditioner reduces the number of iteration steps to a small number that is independent of the refinement level. In practice, however, a preconditioner where the number of iterations grows very slowly when the refinement level is increased is already very satisfactory.

The preconditioned CG-method for sparse finite elements is tested for the two-dimensional test problems on the unit circle presented in [10]. Example 1 is of rotational symmetry. The absorption coefficient is equal to 5 and the blackbody intensity is given by

$$I_b(\mathbf{x}) = \begin{cases} 10 \cdot (0.8 - |\mathbf{x}|)/0.8, & |\mathbf{x}| \leq 0.8 \\ 0, & |\mathbf{x}| > 0.8. \end{cases} \tag{33}$$

In example 2, there are three emitting sources with maximum intensity 10 that decay exponentially (see Fig. 3). The absorption coefficient is equal to the blackbody intensity, increased by 0.5. In physical space, the coarsest mesh $\mathcal{T}_D^0$ is show in Fig. 2, while the vertices of $\mathcal{T}_S^0$ are the

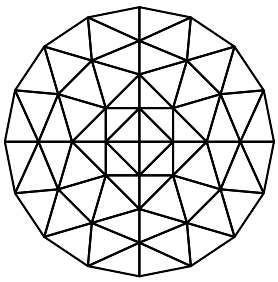

Figure 2: The coarsest mesh $\mathcal{T}_D^0$ in physical space.

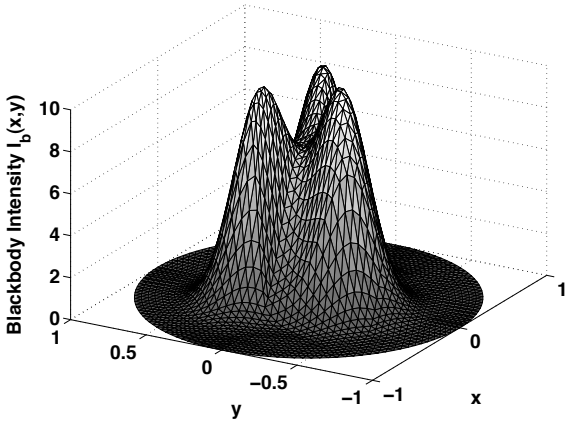

Figure 3: The blackbody intensity $I_b(\mathbf{x})$ of example 2.

corners of the octahedron.

The performance of the preconditioner is measured by comparing the number of CG-iteration steps with and without subspace correction preconditioner. Figure 4 shows the convergence of the CG-method to reduce the relative residual $\frac{\|r\|_A}{\|f\|_A}$ for test example 1 with and without preconditioning . Here $\|.\|_A$ denotes the energy norm corresponding to the bilinear form in (5). Although the CG-method converges without preconditioning to the required tolerance of 10^{-10} for all levels, the number of iterations grows faster than linearly with respect to the refinement level. Applying the multiplicative subspace preconditioner significantly reduces the number of iterations to a very small numbers of 5, 6 and 9 iterations on levels 1, 2 and

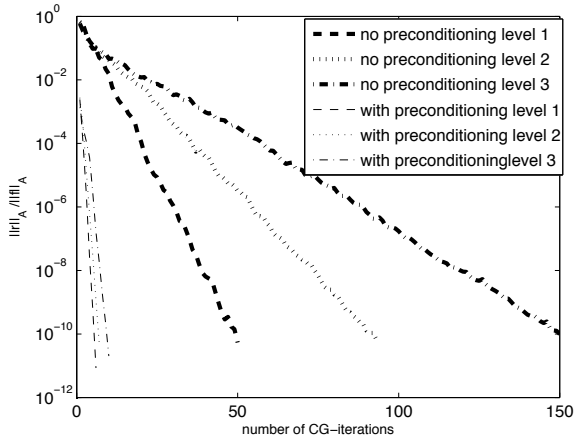

Figure 4: Number of CG iterations with and without preconditioning for example 1.

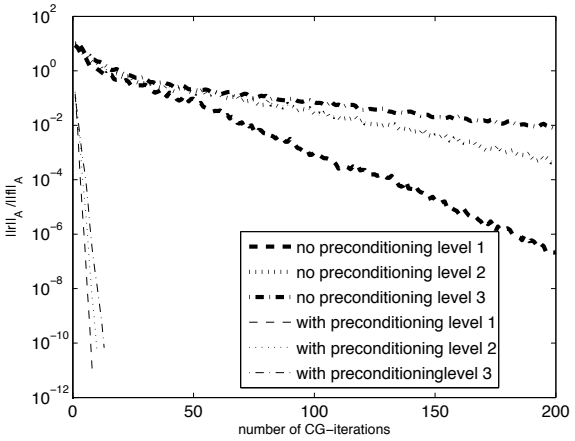

Figure 5: Number of CG iterations with and without preconditioning for example 2.

3. Although the number of iterations grows slowly when the space is refined, the increase is substantionally smaller than linear with respect to the number of levels and the total number of iterations is very small.

The convergence of the CG-method in test example 2, where the absorption coefficient is inhomogenious, is shown in Fig. 5. The difference between the number of iterations with and without preconditioning is even more evident than for example 1. Without preconditioner, the convergence rate of the CG-method deteriorates drastically when the refinement level L is increased. The preconditioner reduces the number of iterations even more significantly than in example 1 to 7, 9 and 12 iterations.

5 Summary and Conclusions

An efficient solver for computing the radiation intensity for the non-scattering RTE with the sparse tensor product discretization has been presented. The algorithm uses the preconditionened CG-method with an efficient matrix-vector multiplication strategy and a multiplicative subspace preconditioner. The matrix-vector multiplication can be carried out at computational costs proportional to the number of degrees of freedom in the discretization $\mathcal{N}_L$. The number of operations to apply the preconditioner, with is based on an algebraic multigrid solver, is nearly proportional to $\mathcal{N}_L$. The preconditioner significantly reduces the number of CG-iterations to a number that grows very slowly when the discretization level L is increased. This makes it possible to solve the RTE with high accuracy at affordable computational costs.

References

[1] S. Achatz, *Adaptive finite Dünngitter-Elemente höherer Ordnung für elliptische partielle Differentialgleichungen mit variablen Koeffizienten.*, Ph.D. thesis, Technische Universität München, 2003.

[2] H.-J. Bungartz, *A multigrid algorithm for higher order finite elements on sparse grids*, ETNA **6** (1997), 63–77.

[3] H.-J. Bungartz and M. Griebel, *Sparse grids.*, Acta Numerica **13** (2004), 147–123.

[4] A. Cohen, *Numerical analysis of wavelet methods.*, Elsevier Amsterdam, Amsterdam, 2003.

[5] J. R. Howell, *Thermal radiation in participating media: The past, the present and possible futures.*, J. Heat Transf. **110** (1988), 1220–1229.

[6] M. F. Modest, *Radiative heat transfer*, 2nd ed., Academic Press, Amsterdam, 2003.

[7] L. M. Ruan, W. An, Tan H. P., and Qi H., *Least-squares finite-element method of multidimensional radiative heat transfer in absorbing and scattering media.*, Numerical Heat Transfer, Part A. **51** (2007), 657–677.

[8] H. De Sterck, Thomas A. Manteuffel, Stephen F. McCormick, and Luke Olson, *Least-squares finite element methods and algebraic multigrid solvers for linear hyperbolic pdes*, SIAM Journal on Scientific Computing **26** (2004), no. 1, 31–54.

[9] R. Viskanta and M. P. Mengüc, *Radiating heat transfer in combustion systems.*, Prog. Energ. Combust. Sci. **13** (1987), no. 2, 97–160.

[10] G. Widmer and R. Hiptmair, *Sparse finite elements for non-scattering radiative transfer in diffuse regimes*, 5th International Symposium of Radiative Transfer, Bodrum, Turkey, ICHMT, 2007.

[11] G. Widmer, R. Hiptmair, and C. Schwab, *Sparse adaptive finite elements for radiative transfer*, J. Comp. Phys. **227** (2008), no. 12, 6071–6105.

[12] C. Zenger, *Sparse grids*, Parallel algorithms for partial differential equations: Proceedings of the Sixth GAMM-Seminar, Kiel, Vieweg-Verlag, 1990, pp. 241–251.

Research Reports

No.	Authors/Title
09-11	*G. Widmer* An efficient sparse finite element solver for the radiative transfer equation
09-10	*P. Benner, D. Kressner, V. Sima, A. Varga* Die SLICOT-Toolboxen für Matlab
09-09	*H. Heumann, R. Hiptmair* A semi-Lagrangian method for convection of differential forms
09-08	*M. Bieri* A sparse composite collocation finite element method for elliptic sPDEs
09-07	*M. Bieri, R. Andreev, C. Schwab* Sparse tensor discretization of elliptic sPDEs
09-06	*A. Moiola* Approximation properties of plane wave spaces and application to the analysis of the plane wave discontinuous Galerkin method
09-05	*D. Kressner* A block Newton method for nonlinear eigenvalue problems
09-04	*R. Hiptmair, J. Li, J. Zou* Convergence analysis of Finite Element Methods for H(curl;Ω)-elliptic interface problems
09-03	*A. Chernov, T. von Petersdorff, C. Schwab* Exponential convergence of *hp* quadrature for integral operators with Gevrey kernels
09-02	*A. Cohen, R. DeVore, C. Schwab* Convergence rates of best N-term Galerkin approximations for a class of elliptic sPDEs
09-01	*B. Adhikari, R. Alam, D. Kressner* Structured eigenvalue condition numbers and linearizations for matrix polynomials
08-32	*R. Sperb* Optimal bounds in reaction diffusion problems with variable diffusion coefficient
08-31	*R. Hiptmair* Discrete compactness for p-version of tetrahedral edge elements
08-30	*H. Heumann, R. Hiptmair* Extrusion contraction upwind schemes for convection-diffusion problems