An efficient and accurate short-characteristics solver for radiative transfer problems

Thomas Hübner, Stefan Turek
Institute for Applied Mathematics, University of Dortmund, Germany

Abstract. In [13], the concept of the *generalized mean intensity* has been proposed as a special numerical approach to the (linear) radiative transfer equations which can result in a significant reduction of the dimension of the discretized system, without eliminating any information for the specific intensities. Moreover, in combination with Krylov-space methods (CG, Bi-CGSTAB, etc.), robust and very efficient solvers as extensions of the classical approximate Λ-iteration have been developed. In this paper, the key tool is the combination of special renumbering techniques together with finite difference-like discretization strategies for the arising transport operators which are based on short-characteristic upwinding techniques of variable order and which can be applied to highly unstructured meshes with locally varying mesh widths. We demonstrate how such special upwinding schemes can be constructed of first order, and particularly of second order accuracy, always leading to lower triangular system matrices. As a consequence, the global matrix assembling can be avoided ('on-the-fly'), so that the storage cost are almost optimal, and the solution of the corresponding convection-reaction subproblems for each direction can be obtained very efficiently. As a further consequence, this approach results in a direct solver in the case of no scattering, while in the case of non-vanishing absorption and scattering coefficients the resulting convergence rates for the full systems depend only on their ratio and the absolute size of these physical quantities, but not on the grid size or mesh topology. We demonstrate these results via prototypical configurations and we examine the resulting accuracy and efficiency for different computational domains, meshes and problem parameters.

1 Introduction

We consider special integro-differential equations which consist of (linear) partial differential operators of transport-reaction type with constant characteristics. Typical examples are Lattice-Boltzmann models (LBM) and, particularly, the radiative transfer equations (RTE). Here, to be precise, we consider the (frequency decoupled) linear radiative transfer problem, formulated for the *specific intensities* $I(x,\omega)$, with x position in space and ω directional unit vector:

$$\mathbf{n}_\omega \cdot \nabla_x I + \kappa(x,\omega) I = \oint_S R(x,\omega,\omega') I \, d\omega' + f(x,\omega)\,, \quad \text{in } \tilde{\Omega} \tag{1}$$

$$I(x,\omega) = g_\omega(x) \quad \text{on } \Gamma_\omega^- \tag{2}$$

$f(x,\omega)$ are given data which may contain the values from previous time steps, or from outer linearizations of nonlinear problems, such that nonstationary approaches may be included, too, and $\tilde{\Omega}$ and Γ_ω^- are defined (with $\Omega \subset \mathbf{R}^n, n = 2,3$ and unit sphere $S \subset \mathbf{R}^d, d = 1,2$) as:

$$\tilde{\Omega} := \Omega \times S \quad , \quad \Gamma_\omega^- := \{x \in \partial\Omega | \mathbf{n}_x \cdot \mathbf{n}_\omega < 0\} \tag{3}$$

$\mathbf{n}_x$ represents the outward normal unit vector at $x \in \partial\Omega$ and $\mathbf{n}_\omega \cdot \nabla_x$ the directional derivative for the direction vector corresponding to ω. Furthermore, we assume the standard product expansion (see [7]) for the scattering phase function R:

$$R = \lambda(x,\omega)P(\omega,\omega') \tag{4}$$

A typical approach, see for instance [12, 8, 6], is based on the *specific intensities* $I(x,\omega)$ which is known to lead to huge storage cost due to the high dimension (2D/3D in space, 1D/2D for the ordinates ω) of the system. As a special case, similar to [13], we additionally assume

$$P(\omega,\omega') = P(\omega') \quad , \quad \oint_S P(\omega')\, d\omega' = 1 \tag{5}$$

which is quite common in astrophysics, e.g. $P = 1/2\pi$ in 2D (see also [4, 7]). Under this additional assumption, we can derive - via purely algebraic transformations - the *generalized mean intensity* approach (GMI) with (at least) the same numerical convergence properties as for the specific intensities [12, 13]. However, this GMI approach requires a much smaller amount of storage which is independent of the number of ordinates in S and which can be formulated in a symmetric fashion (see [13]), too.

2 The Generalized Mean Intensity (GMI) approach

First of all, we discretize the unit sphere S by a set of ordinates/surface points $\omega^m, m = 1,\dots,M$, such that we can replace the integral expression on the right hand side of (1) by an approximate quadrature formula (keep in mind that (4) and (5) are assumed)

$$\oint_S P(\omega')I(x,\omega')\, d\omega' \sim \sum_{m=1}^{M} c^m(P)I(x,\omega^m) \tag{6}$$

with corresponding quadrature weights $c^m(P)$. A standard approach is to introduce spherical coordinates which yields:

$$\oint_S P(\omega')I(x,\omega')\, d\omega' = \int_0^{\pi}\int_0^{2\pi} P(\theta',\varphi')I(x,\theta',\varphi')\, d\theta' \sin\varphi' d\varphi' \tag{7}$$

Approximating separately both one-dimensional integral expressions, e.g. by the trapezoidal rule, leads to

$$\int_0^{\pi}\int_0^{2\pi} P(\theta',\varphi')I(x,\theta',\varphi')\, d\theta' \sin\varphi' d\varphi' \sim \sum_{l=1}^{L}\sum_{k=1}^{K} c^{lk}(P)I(x,\theta^k,\varphi^l) \tag{8}$$

with the quadrature weights

$$c^{lk}(P) = \frac{\pi}{2LK}\, \sin\varphi^l P(\theta^k,\varphi^l). \tag{9}$$

In this case, we obtain the following representation for the direction vectors, namely:

$$\mathbf{n}_{\omega^{lk}} = (\sin\varphi^l \cos\theta^k \,,\, \sin\varphi^l \sin\theta^k \,,\, \cos\varphi^l)^T \tag{10}$$

Applying this semi-discretization in the ordinate space, now we have to solve the following coupled system of partial differential equations (PDEs), with $M := L \cdot K$,

$$\mathbf{n}_{\omega^m} \cdot \nabla_x I^m(x) + \kappa^m(x) I^m(x) = \lambda^m(x) \sum_{n=1}^{M} c^n(P) I^n(x) + f^m(x) \quad , \quad m = 1, \ldots, M, \tag{11}$$

with the following semi-discrete approximations w.r.t. each discrete ordinate ω^m:

$$I^m(x) \approx I(x, \omega^m) \quad , \quad \kappa^m(x) \approx \kappa(x, \omega^m) \quad , \quad \lambda^m(x) \approx \lambda(x, \omega^m) \quad , \quad f^m(x) \approx f(x, \omega^m)$$

Then, we introduce the *generalized mean intensity* $J(x) := \sum_{n=1}^{M} c^n(P) I^n(x)$, so that:

$$\mathbf{n}_{\omega^m} \cdot \nabla_x I^m(x) + \kappa^m(x) I^m(x) = \lambda^m(x)\, J(x) + f^m(x) \quad , \quad m = 1, \ldots, M \tag{12}$$

To continue we need a spatial discretization: the specific details of our applied discretization procedure based on finite difference/finite element methods is described in the following section.

Let $T_h^m I_h^m$ be a discrete version of the left hand side in (12)

$$T_h^m I_h^m \approx \mathbf{n}_{\omega^m} \cdot \nabla_x I^m(x) + \kappa^m(x) I^m(x), \tag{13}$$

and analogously define $L_h^m J_h$ as discrete version of $\lambda^m(x) J(x)$, resp., f_h^m of $f^m(x)$. Then, I_h^m, resp., J_h, is the coefficient vector corresponding to $I^m(x)$, resp., $J(x)$; T_h^m the discretization matrix due to the differential operator $\mathbf{n}_{\omega^m} \cdot \nabla_x + \kappa^m(x)$, and L_h^m denotes the (mass) matrix corresponding to the coefficient function $\lambda^m(x)$ ('scattering') which is a diagonal matrix in the finite difference context or if an appropriate mass lumping is performed for FEM approaches [12]; the same can be done for the (partial) matrix in T_h^m due to the absorption coefficient $\kappa^m(x)$. Additionally, the special upwinding procedure ('short characteristics', see the following section) for treating the convective parts due to $\mathbf{n}_{\omega^m} \cdot \nabla_x$ guarantees stiffness matrices T_h^m which are always lower triangular, and this process works independently for arbitrary geometries and meshes. These aspects are very essential for the high efficiency of the following solution methods.

Following the steps performed so far, the resulting discrete form of the system (12) reads

$$T_h^m I_h^m = L_h^m J_h + f_h^m \quad , \quad m = 1, \ldots, M \tag{14}$$

respectively,

$$I_h^m = (T_h^m)^{-1}(L_h^m J_h + f_h^m) \quad , \quad m = 1, \ldots, M. \tag{15}$$

Multiplying with $c^m(P)$ and summing up for all m leads to

$$\sum_{m=1}^{M} c^m(P) I_h^m = \sum_{m=1}^{M} c^m(P)(T_h^m)^{-1} L_h^m J_h + \sum_{m=1}^{M} c^m(P)(T_h^m)^{-1} f_h^m \tag{16}$$

and finally,

$$J_h = T_h J_h + F_h \tag{17}$$

with

$$T_h = \sum_{m=1}^{M} c^m(P)(T_h^m)^{-1} L_h^m \quad \text{and} \quad F_h = \sum_{m=1}^{M} c^m(P)(T_h^m)^{-1} f_h^m. \tag{18}$$

Hence, we end up with the (well known) discrete *mean intensity* formulation (see [2])

$$A_h J_h := (I - T_h) J_h = F_h \tag{19}$$

where I denotes the corresponding identity matrix. The size of this discrete system depends on the spatial discretization only, independent of the number of discrete ordinates. However, the numerical effort for the solution process which mainly consists of matrix-vector multiplications with the (implicitly given) matrix T_h, is about the same as in the case of directly working with the specific intensities (see [12]) via classical Λ-Iteration: The same number of transport problems with stiffness matrices T_h^m has to be solved, namely for each discrete ordinate, but in a sequential way such that the partial results may be successively added - this is one of the key features of the GMI approach. Then, having calculated the mean intensity J_h, also the specific intensities I_h^m are determined and can be computed in a single postprocessing step by solving equation (14) for all desired ordinates m.

In the following, we briefly discuss appropriate solution procedures for the resulting linear system $A_h J_h = F_h$ with $A_h = I - T_h$. Since T_h is the weighted sum of many inverse transport-reaction matrices, A_h cannot be given explicitly as sparse matrix and iterative methods have to be used. Therefore, the simple (damped) Richardson iteration

$$J_h^{(n+1)} = J_h^{(n)} - \sigma_h (A_h J_h^{(n)} - F_h) \quad , \quad \sigma_h > 0 \tag{20}$$

is a quite general candidate (see also [12]). It is obvious that the special case $\sigma_h = 1$ corresponds to the classical *approximate* Λ-*iteration* (see for instance [4],[10],[15]). This well-known, but usually very slow defect correction method, especially in the case of high absorption κ and scattering values λ, can be accelerated by additional preconditioning

$$J_h^{(n+1)} = J_h^{(n)} - \sigma_h C_h^{-1} (A_h J_h^{(n)} - F_h) \tag{21}$$

where C_h should be easy to invert. Here, the problem is that A_h is only implicitly given. However, a simple but nevertheless successful method is to take the diagonal of A_h which can be calculated if all partial matrices T_h^m are strictly lower diagonal and if L_h^m is diagonal:

$$C_h = \mathrm{diag}(A_h) = I - \sum_{m=1}^{M} c^m(P) \mathrm{diag}((T_h^m)^{-1} L_h^m) = I - \sum_{m=1}^{M} c^m(P) \mathrm{diag}(T_h^m)^{-1} L_h^m \tag{22}$$

This special preconditioner requires as additional numerical effort only the scaling by a vector and is well known in astrophysics ([4, 10, 15]) as *purely local approximate* Λ-*operator*. Further extensions, also known as *non-local* or *nearest neighbor approximate* Λ-*operator*, may consist

of calculating the first subdiagonal of each $(T_h^m)^{-1}$ and to sum it up for all m. However, the numerical amount will be significantly larger, and the numerical properties of this preconditioner are not clear yet.

In [12, 13] we had shown that in the case of large absorption ($\kappa \gg 1$) and simultaneously large scattering values ($\lambda/\kappa \approx 1, \lambda < \kappa$), this approximate Λ-iteration collapses and the convergence rates behave like the ratio λ/κ, which cannot be essentially improved by the local preconditioner. A remedy proposed in [12, 13] was to use Krylov-space methods, especially CG ('conjugate gradient') and Bi-CGSTAB (see [14]), which can be interpreted as improved Richardson schemes in which case the relaxation parameters σ_h are determined adaptively in each iteration step. Additionally, the analysis of the eigenvalues of the operator A_h shows a strong clustering of certain eigenvalues which is very favorable for (preconditioned) Krylov-space solvers. Applying such Krylov-space schemes requires only matrix-vector multiplications with matrix $A_h = I - T_h$ and additional preconditioning with C_h^{-1}, which mainly corresponds to the basic iteration in (21) such that the additional numerical amount to accelerate the simple Richardson scheme is almost negligible.

3 The 'short-characteristic' discretization procedure

3.1 Finite Difference upwinding discretization

To solve the radiative transfer equation in (1), we have seen that the key tool is to solve a set of transport problems on unstructured meshes with (different) constant characteristics $\mathbf{n}_\beta$:

$$\mathbf{n}_\beta \cdot \nabla_x u(x) = g(x) \quad \text{, resp., } \quad \mathbf{n}_\beta \cdot \nabla_x u(x) + a(x)u(x) = g(x) \tag{23}$$

In view of repeatedly solving such problems in an iterative scheme due to the special form of the resulting matrix A_h in (19), we have to find an efficient way to treat such special transport problems. Adjusting the direction of view to the constant characteristic $\mathbf{n}_\beta$ yields a problem in 1D, so that the partial differential equation (23) transforms into an ODE at each point v_0

$$\mathbf{n}_\beta \cdot \nabla_x u(v_0) = u'(v_0) = g(v_0) \tag{24}$$

which can be solved using a simple backward difference quotient. Assuming that the solution u is already known in all 'backward' nodes, the use of an upwind difference scheme leads to a system of linear equations that can be solved directly (without computing any inverse matrix) if an appropriate numbering of the unknowns is used. This results, for each characteristic direction, in a corresponding matrix of lower triangular form which therefore can be inverted directly in $O(n)$ operations, n denoting the total number of unknowns.

The basic upwind difference scheme of first order reads:

$$u'(v_0) = \frac{u(v_0) - u(v_1)}{h} + O(h) \tag{25}$$

When v_1 is not a node of the grid, the value of $u(v_1)$ has to be determined by linear interpolation of the values in the adjacent nodes p^1 and p^2, using weights $\alpha \in [0,1]$ (see Fig. 2) which can be expressed as:

$$u(v_1) \sim (1-\alpha)u(p^1) + \alpha u(p^2) \tag{26}$$

To achieve a discretization of higher order, an upwind difference quotient of (at least) second order accuracy can be used, for example in the nodes v_0, v_1, v_2. Then, the second order difference scheme (with $h_1 + h_2 = r \cdot h_1,\ r > 1$) reads after transforming into an 1D problem:

$$\mathbf{n}_\beta \cdot \nabla_x u(v_0) = u'(v_0) = \frac{(r^2-1))u(v_0) - r^2 u(v_1) + u(v_2)}{h_1(r^2 - r)} + O(h_1, h_2)^2 \tag{27}$$

The equidistant case ($h_1 = h_2 \Leftrightarrow r = 2$) yields the well-known scheme:

$$u'(v_0) \sim \frac{3u(v_0) - 4u(v_1) + u(v_2)}{2h_1} \tag{28}$$

In this case, the values in both 'virtual' nodes v_1, v_2 have to be interpolated (see Fig. 2).
To sustain the higher order of this difference scheme, it is not sufficient to use only the nodal points on each edge for the (linear) interpolation: We need higher order interpolation rules, or equivalently information from additional nodes which can be reached by two possibilities. We could take nodes from neighboring elements, but this leads to additional implementation and computational efforts as the strict data locality of the scheme would be disturbed, particularly on unstructured meshes. Another possible way is to apply finite element (FEM) techniques for piecewise quadratic polynomial spaces. Consequently, we think in terms of a (piecewise) quadratic approximation on each triangle which results in the use of the nodal and midpoint values (on the edges) as degrees of freedom. Then, it is possible to apply quadratic interpolation on the edges, using the values $u(p_1^1), u(p_1^2)$ and $u(m_1)$ to approximate the value $u(v_1)$ (see Fig. 2)

$$u(v_1) = \lambda_1 u(p_1^1) + \lambda_2 u(p_1^2) + \lambda_m u(m_1) \tag{29}$$

with special weights depending on α_1, resp., α_2 to approximate $u(v_2)$:

$$\lambda_1 = (1-\alpha_1)(1-2\alpha_1)\,,\ \lambda_2 = \alpha_1(2\alpha_1 - 1)\,,\ \lambda_m = 4\alpha_1(1-\alpha_1) \tag{30}$$

3.2 Special numbering techniques

What is the correct numbering of the unknowns yielding a lower triangular form for the transport matrices? Figure 4 shows a geometrical approach, sending a 'numbering front' orthogonal to direction β with the characteristic vector $\mathbf{n}_\beta$, beginning at the inflow boundary. Unfortunately, there occurs a deficiency using this 'geometrical' technique even for a cartesian grid: Node p_1^1

has a higher numbering index than node v_0, but it is already used for the quadratic interpolation in v_1 (by the three values on the edge related to midpoint m_1). Hence v_0, using the solution in v_1 for determining $u(v_0)$, leads to loss of the lower triangular property.

A better approach is based on 'topological' sorting known from graph theory. The structure of the grid can be regarded as a directed graph, with an adjacency matrix corresponding to the transport matrix. 'Unknowns' from the grid are represented by nodes in the graph, and there exists a directed edge from node v to node w in the graph if $u(w)$ is used to compute $u(v)$ in the upwinding scheme. Deo proposes in [1] that a numbering of the nodes of a directed graph yielding a triangular form exists iff the graph is acyclic. In our work, we assume that the transport problems, and hence the upwind difference equations on any triangular mesh, are acyclic when regarded as graphs. That means: If the solution $u(v)$ uses $u(w)$ in the difference scheme, then $u(v)$ is not used by $u(w)$ vice versa, neither directly nor indirectly via any other unknown, which would lead to a circle in the graph. Numerical tests on highly unstructured meshes have confirmed the supposition (see [3]), and the algorithm described below always resulted in a numbering with the desired property.

The corresponding algorithm to sort a total number of n nodes is based on topological sorting used in the proof of the proposition in [1]. The graph's adjacency matrix as well as its transposed are stored using sparse matrix techniques. An array is initialized which stores the number of row entries of the matrix during the algorithm (corresponding to the out-degree of every node). All inflow boundary nodes have out-degree 0: As the solution is given there, they do not use other nodes in any difference scheme, and they are written in a queue. Then, we remove successively the queue's nodes from the graph, and for the vanishing edges we decrement the array of the out-degrees using the information from the transposed matrix. If an out-degree falls to zero, the node is written into the queue, too. The order in which the nodes have been written to the queue depicts the new numbering of the nodes. If the queue ended before we had extracted exactly n nodes, this would mean that the graph contained a circle. So, the algorithm yields the desired numbering requiring $O(n)$ operations, which is of the same order as the runtime to solve the transport problem using this numbering. This will be confirmed by the following test calculations.

4 Numerical tests

First, the 'flow around cylinder' configuration from a classical CFD benchmark [9] has been chosen to solve the transport equation $\mathbf{n}_\beta \cdot \nabla_x u(x) + au(x) = g(x)$ for various directions $\mathbf{n}_\beta = (\beta_1, \beta_2)$ with the exact (smooth) solution $u(x) = u(x_1, x_2) = x_1(2.5 - x_1)x_2(0.41 - x_2)$. Table 1 shows the 'init-time' (renumbering procedure for all directions) and 'solve-time' (solving the transport problems) for increasing problem size n. The linear increase in time confirms the suggested efficiency of the algorithm. Furthermore, we calculated the L_2-error using the upwind difference schemes of first and second order for different grid-refinement levels. Table 2 shows the results for varying a and two different directions β. It confirms the expected behavior of

$O(h)$, respectively $O(h^2)$, even on unstructured grids.

lev	n	INIT-Time	SOLVER-Time	INIT-Time	SOLVER-Time
2	12992	$5.00 \cdot 10^{-2}$	$1.00 \cdot 10^{-2}$	$8.00 \cdot 10^{-2}$	$1.00 \cdot 10^{-2}$
3	50560	$2.50 \cdot 10^{-1}$	$3.00 \cdot 10^{-2}$	$2.70 \cdot 10^{-1}$	$4.00 \cdot 10^{-2}$
4	199424	$1.10 \cdot 10^{+0}$	$1.30 \cdot 10^{-1}$	$1.31 \cdot 10^{+0}$	$2.50 \cdot 10^{-1}$
5	792064	$5.53 \cdot 10^{+0}$	$8.00 \cdot 10^{-1}$	$6.59 \cdot 10^{+0}$	$1.24 \cdot 10^{+0}$
6	3156992	$31.16 \cdot 10^{+0}$	$4.83 \cdot 10^{+0}$	$36.81 \cdot 10^{+0}$	$6.59 \cdot 10^{+0}$
7	12605440	$174.40 \cdot 10^{+0}$	$26.15 \cdot 10^{+0}$	$208.51 \cdot 10^{+0}$	$37.99 \cdot 10^{+0}$

Table 1: CPU times (in sec) for renumbering phase (INIT) and solution procedure (SOLVE) on different mesh levels, for 8 angles/directions, on a UltraSPARC-III workstation with 900MHz: upwind 1st order (left) vs. 2nd order (right) (Remark: Increasing the problem size by a factor of 4 seems to increase the CPU time by a factor of 5-6 which is mainly due to more cache misses since mainly indirect adressing is employed, see [11])

UPW=1	Level 3	Level 4	Level 5	Level 6	Level 7
$0°$ $(a = 10^{-2})$	$8.40 \cdot 10^{-2}$	$4.34 \cdot 10^{-2}$	$2.21 \cdot 10^{-2}$	$1.12 \cdot 10^{-2}$	$5.62 \cdot 10^{-3}$
$0°$ $(a = 10^{+0})$	$4.30 \cdot 10^{-2}$	$2.22 \cdot 10^{-2}$	$1.13 \cdot 10^{-2}$	$5.68 \cdot 10^{-3}$	$2.85 \cdot 10^{-3}$
$0°$ $(a = 10^{+2})$	$9.90 \cdot 10^{-4}$	$5.02 \cdot 10^{-4}$	$2.54 \cdot 10^{-4}$	$1.28 \cdot 10^{-4}$	$6.40 \cdot 10^{-5}$
$135°$ $(a = 10^{-2})$	$1.87 \cdot 10^{-1}$	$9.36 \cdot 10^{-2}$	$4.68 \cdot 10^{-2}$	$2.34 \cdot 10^{-2}$	$1.17 \cdot 10^{-2}$
$135°$ $(a = 10^{+0})$	$1.52 \cdot 10^{-1}$	$7.63 \cdot 10^{-2}$	$3.83 \cdot 10^{-2}$	$1.92 \cdot 10^{-2}$	$9.60 \cdot 10^{-3}$
$135°$ $(a = 10^{+2})$	$5.80 \cdot 10^{-3}$	$2.93 \cdot 10^{-3}$	$1.47 \cdot 10^{-3}$	$7.39 \cdot 10^{-4}$	$3.70 \cdot 10^{-4}$

Table 2: L_2-error for 1st order upwinding

UPW=2	Level 3	Level 4	Level 5	Level 6	Level 7
$0°$ $(a = 10^{-2})$	$9.00 \cdot 10^{-4}$	$2.18 \cdot 10^{-4}$	$5.29 \cdot 10^{-5}$	$1.26 \cdot 10^{-5}$	$2.88 \cdot 10^{-6}$
$0°$ $(a = 10^{+0})$	$8.16 \cdot 10^{-4}$	$1.98 \cdot 10^{-4}$	$4.83 \cdot 10^{-5}$	$1.16 \cdot 10^{-5}$	$2.67 \cdot 10^{-6}$
$0°$ $(a = 10^{+2})$	$8.23 \cdot 10^{-5}$	$2.17 \cdot 10^{-5}$	$5.69 \cdot 10^{-6}$	$1.48 \cdot 10^{-6}$	$3.78 \cdot 10^{-7}$
$135°$ $(a = 10^{-2})$	$9.96 \cdot 10^{-3}$	$2.33 \cdot 10^{-3}$	$5.59 \cdot 10^{-4}$	$1.36 \cdot 10^{-4}$	$3.37 \cdot 10^{-5}$
$135°$ $(a = 10^{+0})$	$8.25 \cdot 10^{-3}$	$1.90 \cdot 10^{-3}$	$4.48 \cdot 10^{-4}$	$1.08 \cdot 10^{-4}$	$2.65 \cdot 10^{-5}$
$135°$ $(a = 10^{+2})$	$1.03 \cdot 10^{-3}$	$3.04 \cdot 10^{-4}$	$8.30 \cdot 10^{-5}$	$2.09 \cdot 10^{-5}$	$4.93 \cdot 10^{-6}$

Table 3: L_2-error for 2nd order upwinding

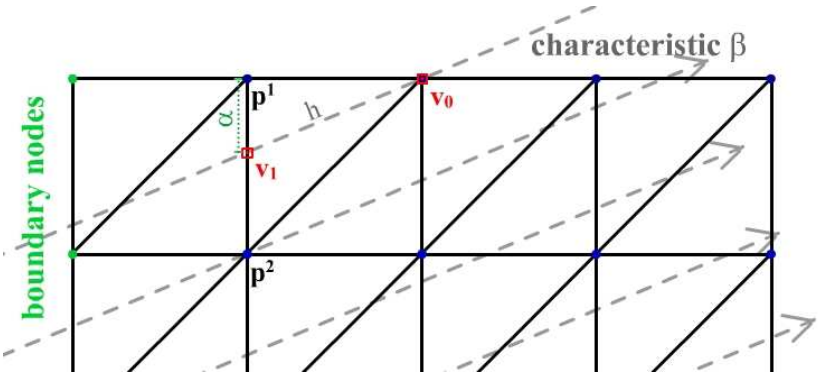

Figure 1: Discretization of first order of the transport problem $\mathbf{n}_\beta \cdot \nabla_x u(v_0) = u'(v_0) = g(v_0)$

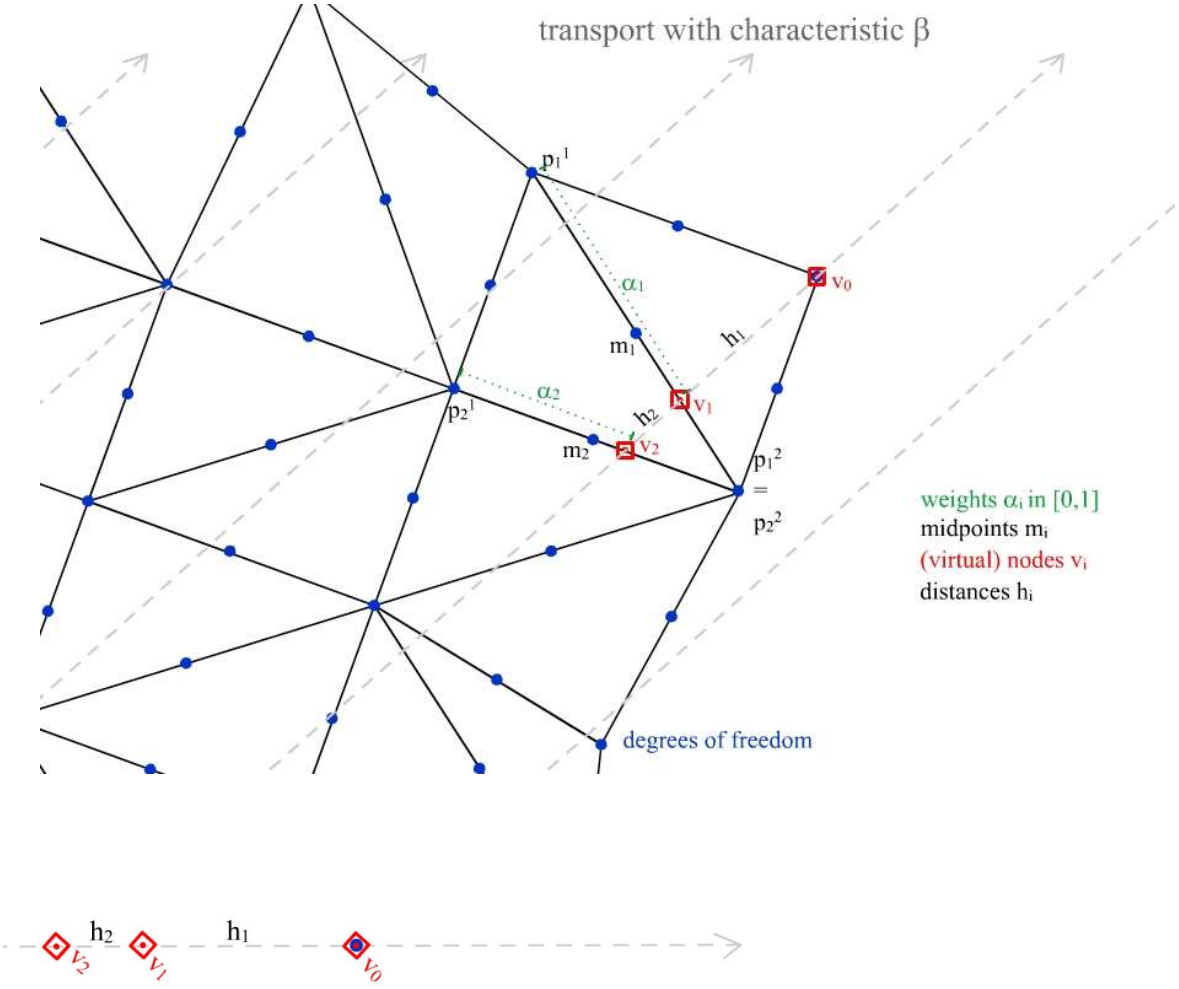

Figure 2: Second order discretization and 1D view of the transport problem

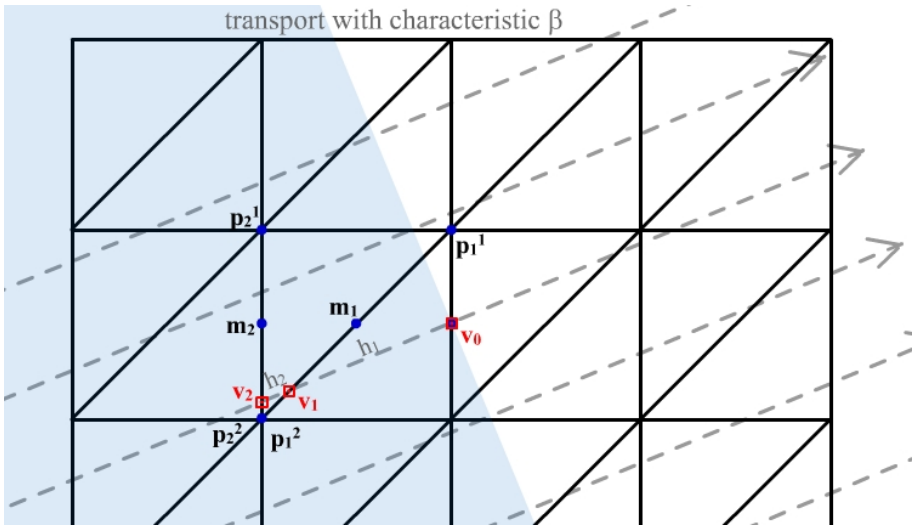

Figure 3: Lexicographical sorting technique

ORDER (QUEUE[*], IN-NODES[*][*], OUT-NODES[*][*], NVT)

0. **INIT:**
i.) QUEUE[*]=0, OUTDEG[*] = 0 , k = 1
ii.) **FOR EACH ENTRY IN** OUT-NODES[i][*] **DO** OUTDEG[i]++
iii.) **FOR EACH** i **WITH** OUTDEG[i]=0 **DO** i → QUEUE

1. **DO WHILE** k < NVT
a.) v=QUEUE[k]
b.) **IF** v=0 **THEN OUTPUT** 'Graph is cyclic!', **STOP!**
c.) **FOR EACH** j **IN** IN-NODES[v][*] **DO:**
d.) OUTDEG[j]- -
e.) **IF** OUTDEG[j]=0 **THEN** j → QUEUE
f.) **END FOR**
g.) k = k + 1

Figure 4: Topological sorting for arbitrary grids

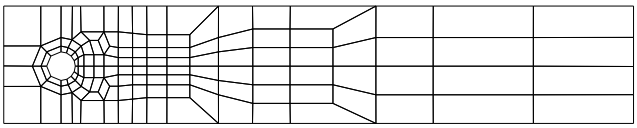

Figure 5: Coarse grid for 'flow around cylinder'; all further refinements are obtained via connecting opposite midpoints

To analyze a nonsmooth solution, we set $g = 0$ and $u = 0$ on the outer inflow boundary in the example from above. By setting $u = 1$ on the inner inflow boundary ('circle'), a beam of intensity 1 is transported into the domain according to the characteristic vector $\mathbf{n}_\beta$ (see Fig. 6).

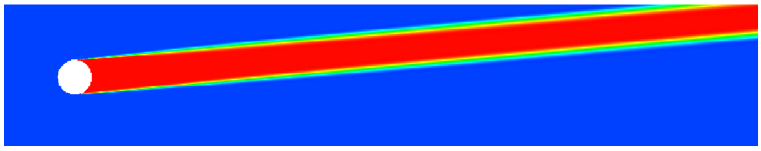

Figure 6: Typical solution behavior for 'non-smooth' test configuration

As can be seen from Fig. 7, the 2nd order upwinding still shows much higher accuracy of the approximated solution, but tends to the expected over- and undershoots, too, while the 1st order upwinding is much less accurate and shows significant numerical diffusion for the solution. Nevertheless, it is monotone and exhibits no spurious oscillations. A remedy which should combine the accuracy of the one and monotony of the other scheme, could be to combine both into an adaptive scheme. As the transport matrices of the two schemes are both lower triangular, a weighted combination of both still remains lower triangular, and therefore sustains the overall computational efficiency when inverting the transport steps. Figure 7 shows the corresponding cutlines through the beam at the (right) 'outflow' part of the computational domain on different mesh levels, indicating the future potential of such special adaptive discretization schemes.

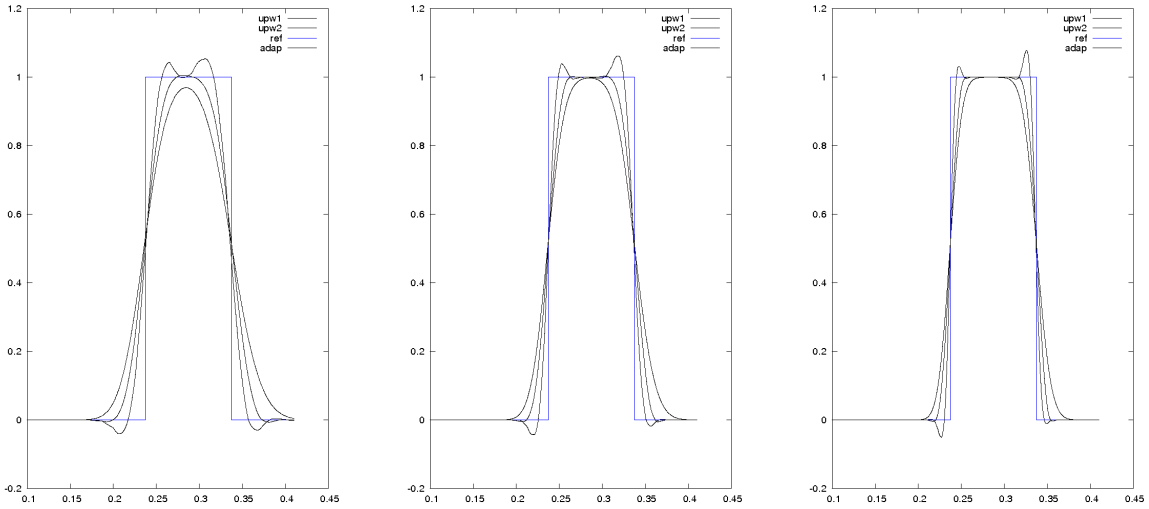

Figure 7: Vertical cutlines through the approximated solutions on 3 levels of grid refinement (from left to right), using 1st order upwind (smearing), 2nd order upwind (oscillating) and adaptive upwind; see [3] for more details

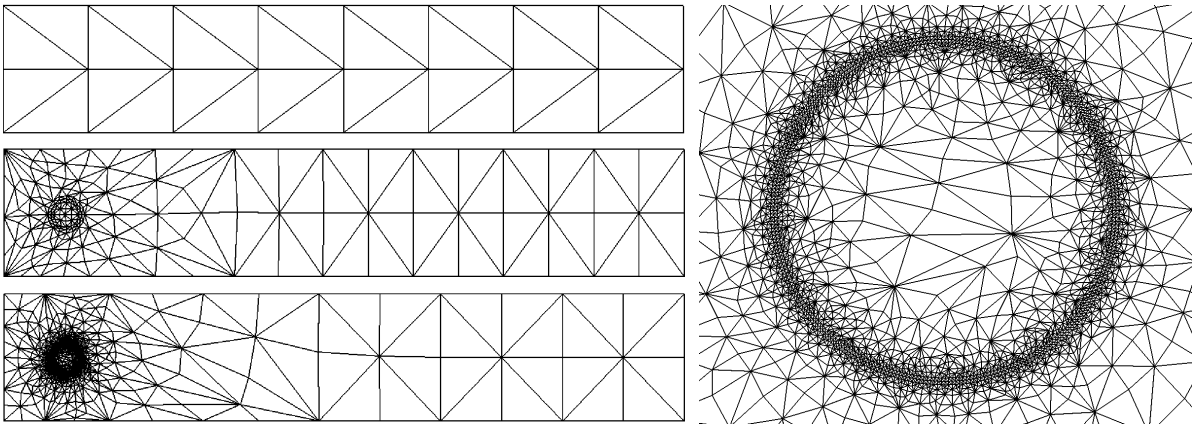

Figure 8: Uniform up to highly adapted grids and zoom for the refined region

The next test configurations indicate the advantageous behavior on locally refined grids. The analytical solution is a peak, being C^1-differentiable, which increases from a zero value to a plateau ('circle') of 1 in a layer of width ε. By varying the parameter ε, one can set the gradient of the solution to be arbitrarily steep. We carried out an error analysis on a uniform mesh, a moderately refined and a significantly locally adapted grid which were refined in the region of the large gradient (see Figure 8).

ε	Level	UPW1	UPW2	UPW1	UPW2	UPW1	UPW2
$\varepsilon = 0.05$	1. level	$8,64 \cdot 10^{-2}$	$2,06 \cdot 10^{-2}$	$8,77 \cdot 10^{-2}$	$1,12 \cdot 10^{-2}$	$1,20 \cdot 10^{-1}$	$1,92 \cdot 10^{-2}$
	2. level	$4,80 \cdot 10^{-2}$	$5,73 \cdot 10^{-3}$	$4,65 \cdot 10^{-2}$	$3,37 \cdot 10^{-3}$	$6,21 \cdot 10^{-2}$	$4,05 \cdot 10^{-3}$
	3. level	$2,58 \cdot 10^{-2}$	$1,49 \cdot 10^{-3}$	$2,43 \cdot 10^{-2}$	$9,70 \cdot 10^{-4}$	$3,15 \cdot 10^{-2}$	$1,06 \cdot 10^{-3}$
$\varepsilon = 0.01$	1. level	$1,78 \cdot 10^{-1}$	$2,20 \cdot 10^{-1}$	$3,52 \cdot 10^{-2}$	$3,44 \cdot 10^{-2}$	$8,74 \cdot 10^{-2}$	$2,23 \cdot 10^{-2}$
	2. level	$1,11 \cdot 10^{-1}$	$6,68 \cdot 10^{-2}$	$1,86 \cdot 10^{-2}$	$2,45 \cdot 10^{-3}$	$4,46 \cdot 10^{-2}$	$2,45 \cdot 10^{-3}$
	3. level	$6,82 \cdot 10^{-2}$	$1,87 \cdot 10^{-2}$	$9,81 \cdot 10^{-3}$	$4,87 \cdot 10^{-4}$	$2,28 \cdot 10^{-2}$	$6,37 \cdot 10^{-4}$
$\varepsilon = 0.005$	1. level	$2,64 \cdot 10^{-1}$	$9,36 \cdot 10^{-1}$	$4,33 \cdot 10^{-2}$	$7,64 \cdot 10^{-2}$	$2,14 \cdot 10^{-2}$	$1,99 \cdot 10^{-2}$
	2. level	$1,37 \cdot 10^{-1}$	$2,47 \cdot 10^{-1}$	$2,32 \cdot 10^{-2}$	$5,53 \cdot 10^{-3}$	$1,17 \cdot 10^{-2}$	$2,54 \cdot 10^{-3}$
	3. level	$8,44 \cdot 10^{-2}$	$5,47 \cdot 10^{-2}$	$1,24 \cdot 10^{-2}$	$1,21 \cdot 10^{-3}$	$6,48 \cdot 10^{-3}$	$6,58 \cdot 10^{-4}$
$\varepsilon = 0.001$	1. level	$9,70 \cdot 10^{-1}$	$2,51 \cdot 10^{0}$	$8,34 \cdot 10^{-2}$	$4,26 \cdot 10^{-1}$	$6,51 \cdot 10^{-2}$	$3,80 \cdot 10^{-1}$
	2. level	$6,50 \cdot 10^{-1}$	$1,79 \cdot 10^{0}$	$4,38 \cdot 10^{-2}$	$1,11 \cdot 10^{-1}$	$2,06 \cdot 10^{-2}$	$3,90 \cdot 10^{-2}$
	3. level	$2,14 \cdot 10^{-1}$	$1,18 \cdot 10^{0}$	$2,45 \cdot 10^{-2}$	$1,50 \cdot 10^{-2}$	$8,40 \cdot 10^{-3}$	$9,79 \cdot 10^{-3}$

Table 4: L_2-error for uniform grid (left), moderately refined (middle) and highly adapted grid (right) for several refinement levels with comparable total number of unknowns

Table 4 presents the L_2-errors on several refinement levels (leading to a comparable number of unknowns for each of the 3 grids: n for one direction for uniform, moderately refined and highly adapted grid, each on 1. level, is 66.177, 38.993, 41.885, respectively): The calculations on the uniform mesh show very poor results and a loss of order for both upwinding schemes for small ε. The order can be recovered on the locally refined grids up to the value $\varepsilon = 0.005$. On such

special grids, the accuracy is increased by two orders of magnitude, even for the configuration $\varepsilon = 0.001$ which however could not be fully resolved by the given meshes.

Finally, we provide numerical convergence results regarding the discussed Richardson, resp. Bi-CGSTAB solver from the previous section. Following our theoretical considerations, we expect numerical convergence rates which are (more or less) independent of the mesh size h, and also the shape of the underlying computational mesh. However, the results will significantly depend on the absolute size of the parameters κ and λ and also their ratio, that means $\frac{\lambda(x)}{\kappa(x)}$ of *scattering* vs. *absorption* .

For the computational tests, we performed the same boundary and right hand side settings as before, however using 8 directions, and the utilized computational meshes are those from Fig. 8. We show the corresponding results for both upwinding schemes ('u1' first order, 'u2' second order, 'ad' adaptive scheme), applying both solvers without as well as with the proposed diagonal preconditioner.

	$\kappa = 1$			$\kappa = 1$			$\kappa = 1$			$\kappa = 10$			$\kappa = 100$			$\kappa = 1000$			$\kappa = 10000$		
	$\lambda = 0.5$			$\lambda = 0.9$			$\lambda = 0.99$			$\lambda = 9.9$			$\lambda = 99$			$\lambda = 990$			$\lambda = 9900$		
n	u1	u2	ad	u1	u2	ad	u1	u2	ad	u1	u2	ad	u1	u2	ad	u1	u2	ad	u1	u2	ad
66.177	13	13	13	18	18	18	19	19	19	118	125	125	1202	1422	1375	2010	2060	2015	2060	2072	2039
263.425	13	13	12	18	18	17	19	19	19	122	125	122	1306	1437	1360	2028	2051	1893	2067	2071	1955
1.051.137	13	13	13	18	18	18	19	19	19	123	125	125	1368	1439	1430	2044	2053	2036	2068	2065	2035
66.177	12	13	13	17	17	17	18	19	19	103	110	110	522	655	623	161	185	170	29	42	37
263.425	13	13	12	18	18	17	19	19	18	113	118	115	784	898	848	288	323	287	46	72	64
1.051.137	13	13	13	18	18	18	19	19	19	119	121	121	1023	1104	1092	492	538	519	78	105	94

Table 5: Quasi-uniform mesh: plain (top) vs. preconditioned Richardson solver (bottom)

	$\kappa = 1$			$\kappa = 1$			$\kappa = 1$			$\kappa = 10$			$\kappa = 100$			$\kappa = 1000$			$\kappa = 10000$			$\kappa = 100000$		
	$\lambda = 0.5$			$\lambda = 0.9$			$\lambda = 0.99$			$\lambda = 9.9$			$\lambda = 99$			$\lambda = 990$			$\lambda = 9900$			$\lambda = 99000$		
n	u1	u2	ad	u1	u2	ad	u1	u2	ad	u1	u2	ad	u1	u2	ad	u1	u2	ad	u1	u2	ad	u1	u2	ad
66.177	5	5	5	6	6	6	6	6	6	17	18	18	58	59	60	66	73	65	25	56	46	8	13	13
263.425	5	5	5	6	6	6	6	6	6	18	19	19	64	66	60	69	72	74	28	51	47	15	26	26
1.051.137	5	5	5	6	6	6	6	6	6	19	19	19	69	71	75	69	79	74	32	49	54	15	14	14
66.177	5	5	5	6	6	6	6	6	6	17	17	17	55	61	60	35	37	34	11	13	12	5	6	6
263.425	5	5	5	6	6	6	6	6	6	18	18	19	65	63	62	47	46	45	15	18	19	7	7	7
1.051.137	5	5	5	6	6	6	6	6	6	18	19	19	71	62	62	56	62	62	22	24	24	7	9	9

Table 6: Quasi-uniform mesh: plain (top) vs. preconditioned Bi-CGSTAB solver (bottom)

The results clearly show the expected advantageous behavior of the additional diagonal preconditioner, particularly for very large values for κ and λ, and of the Bi-CGSTAB solver. Keep in mind that 1 iteration with the Bi-CGSTAB scheme has approximately the same numerical

effort as 2 steps with the Richardson solver.

Finally, we perform the same tests on the moderately, resp., highly locally refined meshes (see Fig. 8) which show an only weak dependence by the convergence behavior of the underlying computational meshes due to the exact solution of the discrete transport problems as leading differential operator in the RTE equations.

	$\kappa=1$			$\kappa=1$			$\kappa=1$			$\kappa=10$			$\kappa=100$			$\kappa=1000$			$\kappa=10000$			$\kappa=100000$		
	$\lambda=0.5$			$\lambda=0.9$			$\lambda=0.99$			$\lambda=9.9$			$\lambda=99$			$\lambda=990$			$\lambda=9900$			$\lambda=99000$		
n	u1	u2	ad	u1	u2	ad	u1	u2	ad	u1	u2	ad	u1	u2	ad	u1	u2	ad	u1	u2	ad	u1	u2	ad
38.993	5	5	5	6	6	6	6	6	6	18	19	19	62	69	66	80	80	79	59	78	69	21	36	35
155.297	5	5	5	6	6	6	6	6	6	18	19	18	66	69	72	79	88	86	68	76	79	32	40	50
619.841	5	5	5	6	6	6	6	6	6	19	19	19	73	77	69	79	92	90	77	83	78	35	52	48
38.993	5	5	5	6	6	6	6	6	6	16	18	17	46	60	59	51	77	75	32	41	38	10	15	14
155.297	5	5	5	6	6	6	6	6	6	18	18	17	52	64	70	70	118	95	38	66	51	14	20	20
619.841	5	5	5	6	6	6	6	6	6	18	19	19	58	63	64	69	124	109	50	78	75	19	30	26

Table 7: Moderately refined mesh: plain (top) vs. preconditioned Bi-CGSTAB (bottom)

	$\kappa=1$			$\kappa=1$			$\kappa=1$			$\kappa=10$			$\kappa=100$			$\kappa=1000$			$\kappa=10000$			$\kappa=100000$		
	$\lambda=0.5$			$\lambda=0.9$			$\lambda=0.99$			$\lambda=9.9$			$\lambda=99$			$\lambda=990$			$\lambda=9900$			$\lambda=99000$		
n	u1	u2	ad	u1	u2	ad	u1	u2	ad	u1	u2	ad	u1	u2	ad	u1	u2	ad	u1	u2	ad	u1	u2	ad
41.885	5	5	5	6	6	6	6	6	6	17	18	18	58	66	61	79	81	82	66	77	68	25	61	50
167.385	5	5	5	6	6	6	6	6	6	18	19	19	60	70	70	91	84	84	70	91	76	47	47	56
669.233	5	5	5	6	6	6	6	6	6	19	20	20	66	70	76	89	89	87	76	87	89	55	74	69
41.885	5	5	5	6	6	6	6	6	6	12	16	15	33	51	45	44	86	69	32	66	54	16	27	21
167.385	5	5	5	6	6	6	6	6	6	15	17	17	43	64	59	57	96	80	46	79	64	22	36	34
669.233	5	5	5	6	6	6	6	6	6	17	18	18	47	73	68	66	99	91	62	101	93	32	49	44

Table 8: Highly refined mesh: plain (top) vs. preconditioned Bi-CGSTAB (bottom)

5 Conclusion and outlook

An improved combination of discretization and solution techniques has been presented for the numerical treatment of (linear) radiative transfer problems as prototypical examples for more general integro-differential equations with special transport operators including constant characteristics. The key technique is a short-characteristic upwinding discretization on unstructured meshes which in combination with corresponding renumbering techniques, based on topological sorting methods, allows strictly lower triangular transport operators for each direction. Together with appropriate interpolation routines, based on finite element techniques for higher order polynomials, special finite difference-like discretization techniques of variable order are constructed;

in this paper, we demonstrated the corresponding numerical behavior for first and second order methods.

The big advantage of this approach is that the assembling of global stiffness matrices is avoided ('on-the-fly') and the partial transport problems for each direction can be directly solved, independent of the underlying computational mesh. Then, together with preconditioned Krylov-space methods, iterative solvers for the (linear) radiative transfer equations can be constructed with a numerical convergence behavior which does not depend on the grid size and mesh topology, but only on the (absolute) size of absorption and scattering data. The corresponding numerical tests for prototypical geometries, computational meshes and parameter constellations demonstrate the high efficiency of this approach, and show that this methodology is an excellent candidate for more complex integro-differential equations, particularly for Lattice-Boltzmann methods which are subject of our current research activities. Related to such extensions, further studies have to address limiter techniques of TVD type for stabilizing the discrete solutions for steep gradients, and the embedding of the linear problems into the fully nonlinear RTE, resp., Lattice-Boltzmann context to treat efficiently the additional nonlinearities. Finally, more advanced preconditioners have to be developed and analyzed to handle parameter configurations with variable and large coefficients for the absorption and scattering terms.

References

[1] Deo, N.: *Graph Theory with Applications to Engineering and Computer Science*. Prentice Hall International, (1974), 222–333

[2] Johnson, C., Pitkäranta, J.: *Convergence of a fully discrete scheme for two–dimensional neutron transport*, SIAM J. Numer. Anal., 20, 951–966 (1983)

[3] Hübner, Th.: *Spezielle Diskretisierungs- und Lösungsmethoden für Integro-Differentialgleichungen am Beispiel der Strahlungstransportgleichung*, Diploma Thesis, Dortmund, 2005

[4] Kalkofen, W.: *Numerical Radiative Transfer*, Cambridge University Press, 1987

[5] Klar, A., Seaid, M.: *Efficient Preconditioning of Linear Systems Arising from the Discretization of Radiative Transfer Equation*, Challenges in Scientific Computing, Berlin 2002, Lecture Notes in Computational Science and Engineering, 2003

[6] Meinköhn, E.: *Modellierung dreidimensionaler Strahlungsfelder im frühen Universum*, PhD Thesis, 2002, Heidelberg

[7] Mihalas, D., Weibel–Mihalas, B.: *Foundations of Radiation Hydrodynamics*, Oxford University Press, 1984

[8] Richling, S., Meinköhn, E., Kryzhevoi, N., Kanschat, G.: *Radiative Transfer with Finite Elements*, Astronomy and Astrophysics **380**, 776-788 (2001)

[9] Schäfer, M., Turek, S. (with support by F. Durst, E. Krause, R. Rannacher): *Benchmark computations of laminar flow around cylinder*, in E.H. Hirschel (editor), *Flow Simulation with High-Performance Computers II*, Volume 52 of *Notes on Numerical Fluid Mechanics*, 547–566, Vieweg, 1996

[10] Steiner, O.: *A rapidly converging temperature correction procedure using operator perturbation*, Astronomy and Astrophysics, Springer, 231, 278–288 (1990)

[11] Turek, S., Runge, A., Becker, Ch.: *The FEAST Indices - Realistic evaluation of modern software components and processor technologies*, Computer and Mathematics with Applications, 41, 1431 - 1464, 2001

[12] Turek, S.: *An efficient solution technique for the radiative transfer equation*, Impact of computing in science and engineering, 5, 201–214, 1993

[13] Turek, S.: *A Generalized Mean Intensity Approach for the Numerical Solution of the Radiative Transfer Equation*, Preprint, 1994

[14] Van der Vorst, H.: *BI–CGSTAB: A fast and smoothly converging variant of BI–CG for the solution of nonsymmetric linear systems*, SIAM J. Sci. Stat. Comput., 13, 631–644 (1992)

[15] Väth, H.M.: *Three-dimensional radiative transfer on a massively parallel computer*, Astronomy and Astrophysics, Springer, 284, 319, 1994