Implicit FEM-FCT algorithms and discrete Newton methods for transient convection problems

M. Möller[1,*,†] , D. Kuzmin[1,‡] and D. Kourounis[2]

[1] *Institute of Applied Mathematics (LS III), University of Dortmund*
Vogelpothsweg 87, D-44227, Dortmund, Germany
[2] *Department of Material Sciences and Engineering, University of Ioannina, Greece*

Abstract

A new generalization of the flux-corrected transport (FCT) methodology to implicit finite element discretizations is proposed. The underlying high-order scheme is supposed to be unconditionally stable and produce time-accurate solutions to evolutionary convection problems. Its nonoscillatory low-order counterpart is constructed by means of mass lumping followed by elimination of negative off-diagonal entries from the discrete transport operator. The raw antidiffusive fluxes, which represent the difference between the high- and low-order schemes, are updated and limited within an outer fixed-point iteration. The upper bound for the magnitude of each antidiffusive flux is evaluated using a single sweep of the multidimensional FCT limiter at the first outer iteration. This semi-implicit limiting strategy makes it possible to enforce the positivity constraint in a very robust and efficient manner. Moreover, the computation of an intermediate low-order solution can be avoided. The nonlinear algebraic systems are solved either by a standard defect correction scheme or by means of a discrete Newton approach whereby the approximate Jacobian matrix is assembled edge-by-edge. Numerical examples are presented for two-dimensional benchmark problems discretized by the standard Galerkin FEM combined with the Crank-Nicolson time-stepping.

Key Words: high-resolution schemes; flux-corrected transport; Newton-like solution techniques; sparse Jacobian evaluation; finite elements; implicit time-stepping

1. Introduction

The advent of nonlinear high-resolution schemes for convection-dominated flows traces its origins to the *flux-corrected transport* (FCT) methodology introduced in the early 1970s by Boris and Book [1]. The fully multidimensional generalization proposed by Zalesak [29] has formed a very general framework for the design of FCT algorithms by representing them as a blend of linear high- and low-order approximations. Unlike other limiting techniques, which are

[*] Correspondence to: Matthias Möller, Institute of Applied Mathematics (LS III), University of Dortmund, Vogelpothsweg 87, D-44227, Dortmund, Germany. Tel.: +49 231 755-3071, Fax: +49 231 755-5933
[†] E-mail: matthias.moeller@math.uni-dortmund.de
[‡] E-mail: kuzmin@math.uni-dortmund.de

typically based on geometric design criteria, flux correction of FCT type is readily applicable to finite element discretizations on unstructured meshes [20],[21]. A comprehensive summary of the state of the art can be found in [2],[16],[21],[30].

The design philosophy behind modern front-capturing methods involves a set of physical or mathematical constraints to be imposed on the discrete solution so as to prevent the formation of spurious undershoots and overshoots in the vicinity of steep gradients. To this end, the following algorithmic components are to be specified [16],[30]

- a high-order approximation which may fail to possess the desired properties;
- a low-order approximation which does enjoy these properties but is less accurate;
- a way to decompose the difference between the above into a sum of skew-symmetric internodal fluxes which can be manipulated without violating mass conservation;
- a cost-effective mechanism for adjusting these antidiffusive fluxes in an adaptive fashion so that the imposed constraints are satisfied for a given solution.

Classical FCT algorithms are based on an explicit correction of the low-order solution whose local extrema serve as the upper/lower bounds for the sum of limited antidiffusive fluxes. In the case of an implicit time discretization, which gives rise to a nonlinear algebraic system, the same strategy can be used to secure the positivity of the right-hand side, whereas the left-hand side is required to satisfy the *M-matrix* property [11],[12].

The rationale for the development of implicit FCT algorithms stems from the fact that the underlying linear discretizations must be stable. In particular, the use of an unstable high-order method may give rise to nonlinear instabilities which manifest themselves in significant distortions of the solution profiles as an aftermath of aggressive flux limiting. In the finite element context, a proper amount of streamline diffusion can be used to stabilize an explicit Galerkin scheme. However, the evaluation of extra terms increases the cost of matrix assembly and the time step must satisfy a restrictive 'CFL' condition. On the other hand, unconditionally stable implicit methods can be operated at large time steps (unless iterative solvers fail to converge or the positivity criterion is violated) and there is no need for any extra stabilization. Moreover, the overhead cost is insignificant, since the use of a consistent mass matrix leads to a sequence of linear systems even in the fully explicit case.

The generalized FEM-FCT methodology introduced in [11],[12] and refined in [13],[14] is applicable to implicit time discretizations but the cost of iterative flux correction is rather high if the sum of limited antidiffusive fluxes and the nodal correction factors need to be updated in each outer iteration. In addition, the nonlinear convergence rates leave a lot to be desired in many cases. The use of 'frozen' correction factors computed at the beginning of the time step by the standard Zalesak limiter alleviates the convergence problems but the linearized scheme can no longer be guaranteed to remain positivity-preserving. The semi-implicit limiting strategy proposed in [18] makes it possible to overcome this problem and enforce the positivity constraint at a cost comparable to that of explicit flux correction. The resulting FEM-FCT algorithm is to be recommended for strongly time-dependent problems discretized in time by the Crank-Nicolson scheme. The design of general-purpose flux limiters which are more expensive but do not suffer from a loss of accuracy at large time steps is addressed in [17].

In the present paper, we compare the unpublished semi-implicit FCT scheme [18] to its semi-explicit prototype and focus on the iterative solution of the resulting nonlinear algebraic systems. As an alternative to the straightforward defect correction scheme employed previously [18], a discrete Newton method tailored to the peculiarities of FEM-FCT schemes is developed.

The sparse Jacobian matrix is approximated with second-order accuracy by means of divided differences and assembled edge-by-edge. The semi-implicit nature of the new FCT limiter makes the Jacobian assembly particularly efficient, since the sparsity pattern of the underlying matrices is preserved. A detailed numerical study illustrates the potential of flux-corrected Galerkin schemes combined with discrete Newton methods for the treatment of nonlinearities.

2. Algebraic flux correction

In this paper, we adopt an algebraic approach to the design of high-resolution schemes which consists of imposing certain mathematical constraints on discrete operators, so as to achieve some favorable matrix properties. A very handy algebraic criterion, which represents a multidimensional generalization of Harten's TVD theorem, was introduced by Jameson [7],[8] who proved that a semi-discrete scheme of the form

$$\frac{\mathrm{d}u_i}{\mathrm{d}t} = \sum_{j\neq i} c_{ij}(u_j - u_i), \qquad c_{ij} \geq 0, \quad \forall j \neq i \tag{1}$$

is *local extremum diminishing* (LED). After the discretization in time, such schemes remain positivity-preserving (PP) provided that each solution update $u^n \rightarrow u^{n+1}$ or the converged steady-state solution $u^{n+1} = u^n$ satisfy an algebraic system of the form

$$Au^{n+1} = Bu^n + f, \tag{2}$$

where $A = \{a_{ij}\}$ is an *M-matrix*, whereas $B = \{b_{ij}\}$ and $f = \{f_i\}$ have no negative entries. Under these conditions, the positivity of the old solution carries over to the new one [14],[16]

$$u^n \geq 0 \qquad \Rightarrow \qquad u^{n+1} = A^{-1}[Bu^n + f] \geq 0. \tag{3}$$

If the underlying spatial discretization is LED, then the off-diagonal coefficients of both matrices have the right sign, while the positivity condition $b_{ii} \geq 0$ for the diagonal entries of B yields a readily computable upper bound for admissible time steps [16]

$$1 + \Delta t(1-\theta)\min_i c_{ii}^n \geq 0 \qquad \text{for} \quad 0 \leq \theta < 1. \tag{4}$$

Of course, the above algebraic constraints are not the necessary but merely sufficient conditions for a numerical scheme to be local extremum diminishing and/or positivity preserving. In the linear case, they turn out to be far too restrictive. According to the well-known Godunov theorem, linear schemes satisfying these criteria are doomed to be (at most) first-order accurate. On the other hand, a high-order discretization which fails to satisfy the imposed constraints unconditionally can be adjusted so that it admits an equivalent representation of the form (1) and/or (2), where the matrix entries may depend on the unknown solution. This idea makes it possible to construct a variety of nonlinear high-resolution schemes based on the *algebraic flux correction* paradigm [16],[17].

To keep the presentation self-contained, we will follow the road map displayed in Fig. 1 and explain the meaning of all discrete operators in the next three sections. Roughly speaking, a high-order Galerkin discretization is to be represented in the generic form (2), where the matrices A and B do satisfy the above-mentioned positivity constraint. In order to guarantee

1. Semi-discrete high-order scheme (Galerkin FEM)

$$M_C\frac{\mathrm{d}u}{\mathrm{d}t} = Ku \qquad \text{such that} \quad \exists\, j \neq i : \; k_{ij} < 0$$

2. Semi-discrete low-order scheme $L = K + D$

$$M_L\frac{\mathrm{d}u}{\mathrm{d}t} = Lu \qquad \text{such that} \quad l_{ij} \geq 0, \; \forall j \neq i$$

3. Nonlinear FEM-FCT algorithm $Au^{n+1} = Bu^n + f^*$,

$$\text{where} \quad A = M_L - \theta\Delta t L, \quad B = M_L + (1-\theta)\Delta t L$$

Figure 1. Roadmap of matrix manipulations.

that the vector f poses no hazard to positivity either, it is to be replaced by its limited counterpart f^* such that the right-hand side remains nonnegative for $u^n \geq 0$. This modification is mass-conserving provided that both f and f^* can be decomposed into skew-symmetric internodal fluxes as defined below. A family of implicit FEM-FCT schemes based on this algebraic approach was proposed in [11],[12] and combined with an iterative limiting strategy in [14]. In Section 5.2, we present an alternative generalization of Zalesak's limiter which proves much more robust and efficient. The new approach to flux correction of FCT type is also based on the positivity constraint (2) but enforces it in another way so that the costly computation of nodal correction factors is performed just once per time step. The positivity of the resulting semi-implicit FCT algorithm will be proven in Section 5.3.

A discrete Newton approach to the solution of the nonlinear algebraic equations that need to be solved in each time step is addressed in Section 5.4. Due to the fact that the correction factors are computed once and for all in the first outer iteration, an efficient assembly of the approximate Jacobian matrix is feasible for the semi-implicit FCT algorithm.

3. Semi-discrete high-order scheme

As a standard model problem, consider the time-dependent continuity equation for a scalar quantity u transported by the velocity field $\mathbf{v}$ which is assumed to be known

$$\frac{\partial u}{\partial t} + \nabla \cdot (\mathbf{v}u) = 0. \tag{5}$$

Let the discretization in space be performed by a (Galerkin) finite element method which yields a DAE system for the vector of time-dependent nodal values

$$M_C\frac{\mathrm{d}u}{\mathrm{d}t} = Ku, \tag{6}$$

where $M_C = \{m_{ij}\}$ denotes the consistent mass matrix and $K = \{k_{ij}\}$ is the discrete transport operator. The latter may contain some streamline diffusion used for stabilization purposes and/or to achieve better phase accuracy in the framework of Taylor-Galerkin methods [5]. Its skew-symmetric part $\frac{1}{2}(K - K^T)$ provides a consistent discretization of $\mathbf{v} \cdot \nabla$, whereas the symmetric part $\frac{1}{2}(K + K^T) - \mathrm{diag}\{K\}$ represents a discrete (anti-)diffusion operator [27],[28].

4. Semi-discrete low-order scheme

In the case of linear discretizations, the algebraic constraints (1) and (2) can be readily enforced by means of 'discrete upwinding' as proposed in [11],[12]. For a semi-discrete finite element scheme of the form (6), the required matrix manipulations are as follows

- replace the consistent mass matrix M_C by its lumped counterpart $M_L = \mathrm{diag}\{m_i\}$,
- render the operator K local extremum diminishing by adding an artificial diffusion operator $D = \{d_{ij}\}$ so as to eliminate all negative off-diagonal coefficients.

This straightforward 'postprocessing' transforms (6) into its linear LED counterpart

$$M_L \frac{\mathrm{d}u}{\mathrm{d}t} = Lu, \qquad L = K + D, \tag{7}$$

where D is supposed to be a symmetric matrix with zero row and column sums. For each pair of nonzero off-diagonal coefficients k_{ij} and k_{ji} of the high-order operator K, the optimal choice of the artificial diffusion coefficient d_{ij} reads [12],[16]

$$d_{ij} = \max\{-k_{ij}, 0, -k_{ji}\} = d_{ji}. \tag{8}$$

Alternatively, one can apply discrete upwinding to the skew-symmetric part $\frac{1}{2}(K - K^T)$ of the original transport operator K, which corresponds to [17]

$$d_{ij} = \frac{|k_{ij} - k_{ji}|}{2} - \frac{k_{ij} + k_{ji}}{2} = d_{ji}. \tag{9}$$

In either case, the off-diagonal coefficients of the low-order operator $l_{ij} := k_{ij} + d_{ij}$ are nonnegative, as required by the LED criterion (1). Due to the zero row sum property of the artificial diffusion operator D, the diagonal coefficients of L are given by

$$l_{ii} := k_{ii} - \sum_{j \neq i} d_{ij}. \tag{10}$$

The semi-discretized equation for the nodal value $u_i(t)$ can be represented as

$$m_i \frac{\mathrm{d}u_i}{\mathrm{d}t} = \sum_{j \neq i} l_{ij}(u_j - u_i) + u_i \sum_j l_{ij}, \tag{11}$$

where $m_i = \sum_j m_{ij} > 0$ and $l_{ij} \geq 0,\ \forall i \neq j$. The last term in the above expression represents a discrete counterpart of $-u\nabla \cdot \mathbf{v}$ which is responsible for a physical growth of local extrema [16]. Recall, that the operator D has zero row sums so $u_i \sum_j l_{ij} = u_i \sum_j k_{ij}$ in equation (11). In the semi-discrete case, it is harmless since (cf. [9])

$$u_i(t) = 0, \quad u_j(t) \geq 0, \quad \forall j \neq i \qquad \Rightarrow \qquad \frac{\mathrm{d}u_i}{\mathrm{d}t} \geq 0, \tag{12}$$

which proves that the low-order scheme (7) is positivity-preserving. For the fully discrete system to inherit this property, the time step should be chosen in accordance with the CFL-like condition (4) unless the backward Euler time-stepping ($\theta = 1$) is employed.

5. Nonlinear FEM-FCT algorithm

The high-order system (6) discretized in time by a standard two-level θ-scheme

$$[M_C - \theta \Delta t K]u^{n+1} = [M_C + (1-\theta)\Delta t K]u^n \tag{13}$$

admits an equivalent representation in the form (2) amenable to flux correction

$$[M_L - \theta \Delta t L]u^{n+1} = [M_L + (1-\theta)\Delta t L]u^n + f(u^{n+1}, u^n). \tag{14}$$

The last term in the right-hand side is assembled from skew-symmetric internodal fluxes f_{ij} which can be associated with the edges of the sparsity graph [16]

$$f_i = \sum_{j \neq i} f_{ij}, \qquad \text{where} \quad f_{ji} = -f_{ij}. \tag{15}$$

Specifically, these *raw antidiffusive fluxes*, which offset the discretization error induced by mass lumping and discrete upwinding, are given by the formula [14],[16]

$$f_{ij} = [m_{ij} + \theta \Delta t d_{ij}^{n+1}]\ (u_i^{n+1} - u_j^{n+1}) - [m_{ij} - (1-\theta)\Delta t d_{ij}^n]\ (u_i^n - u_j^n). \tag{16}$$

Interestingly enough, the contribution of the consistent mass matrix consists of a truly antidiffusive implicit part and a diffusive explicit part which has a strong damping effect. In fact, explicit mass diffusion of the form $(M_C - M_L)u^n$ has been used to construct the 'monotone' low-order method in the framework of explicit FEM-FCT algorithms [20].

In the case of an implicit time discretization ($0 < \theta \leq 1$), the nonlinearities inherent to the governing equation and/or to the employed high-resolution scheme call for the use of an iterative solution strategy. Let successive approximations to the solution u^{n+1} at the new time level $t^{n+1} = t^n + \Delta t$ be computed step-by-step in the framework of a fixed-point iteration

$$u^{(m+1)} = u^{(m)} + [C^{(m)}]^{-1} r^{(m)}, \qquad m = 0, 1, 2, \ldots \tag{17}$$

where $C^{(m)}$ denotes a suitable 'preconditioner' (to be defined below) that should be easy to invert. The corresponding residual vector of the m-th outer iteration is given by

$$r^{(m)} = b^{(m)} - A u^{(m)}. \tag{18}$$

Here, A represents the 'monotone' evolution operator for the underlying low-order scheme

$$A = M_L - \theta \Delta t L, \qquad L = K + D \tag{19}$$

which enjoys the M-matrix property, since the off-diagonal entries of L are nonnegative by construction. The right-hand side $b^{(m)}$, which needs to be updated in each outer iteration, consists of a low-order part augmented by limited antidiffusion [16]

$$b^{(m)} = B u^n + f^*(u^{(m)}, u^n), \qquad B = M_L + (1-\theta)\Delta t L. \tag{20}$$

In order to prevent the formation of nonphysical undershoots and overshoots, the raw antidiffusive fluxes f_{ij} should be multiplied by suitable correction factors so that

$$f_i^* = \sum_{j\neq i} \alpha_{ij} f_{ij}, \qquad \text{where} \qquad 0 \leq \alpha_{ij} \leq 1. \tag{21}$$

This adjustment transforms (14) into a nonlinear combination of the low-order scheme ($\alpha_{ij} \equiv 0$) and the original high-order one ($\alpha_{ij} \equiv 1$). The task of the flux limiter is to determine an optimal value of each correction factor α_{ij} individually so as to remove as much artificial diffusion as possible without violating the positivity constraint.

In a practical implementation, the 'inversion' of the operator $C^{(m)}$ is also performed by a suitable iteration procedure for solving the sequence of linear subproblems

$$C^{(m)} \Delta u^{(m+1)} = r^{(m)}, \qquad m = 0, 1, 2, \dots \tag{22}$$

After a certain number of inner iterations, the increment $\Delta u^{(m+1)}$ is applied to the last iterate, whereby the solution from the previous time step provides a reasonable initial guess

$$u^{(m+1)} = u^{(m)} + \Delta u^{(m+1)}, \qquad u^{(0)} = u^n. \tag{23}$$

A natural choice for the preconditioner $C^{(m)}$ is the monotone low-order operator (19) so that the iteration procedure (17) yields the standard fixed-point defect correction scheme [25]

$$A u^{(m+1)} = b^{(m)}, \qquad m = 0, 1, 2, \dots \tag{24}$$

As a (potentially) more efficient alternative, let $C^{(m)}$ be the discrete Jacobian matrix

$$J^{(m)} = - \left. \frac{\partial r(u)}{\partial u} \right|_{u=u^{(m)}} = \left. \frac{\partial [Au - f^*(u, u^n)]}{\partial u} \right|_{u=u^{(m)}} \tag{25}$$

evaluated at the last iterate $u^{(m)}$ so as to recover *Newton's method* from (17). It is well-known that its convergence behavior is quite sensitive to the initial guess $u^{(0)}$ and the quality of the solution increment $\Delta u^{(m+1)}$. Due to the fact that the linear subproblems (22) are solved by an iterative technique, the resulting algorithm is categorized as an inexact Newton method [4]. A simple inexact scheme is based on the following convergence criteria on each linear iteration

$$\| J^{(m)} \Delta u^{(m+1)} - r^{(m)} \| \leq \eta \| r^{(m)} \|, \tag{26}$$

whereby the so-called forcing term $\eta \in [0, 1)$ can be chosen adaptively [6]. Furthermore, some globalization strategy may be required to compensate the lack of convergence robustness of Newton's method. For a detailed description of such techniques which are mainly designed to guarantee a sufficient decrease of the nonlinear residual (18), the interested reader is referred to the literature, e.g., [10]. As we are about to see, globalization is less critical due to the fact that the proposed FEM-FCT algorithm is tailored to the treatment of strongly time-dependent problems so that the solution from the last time step may serve as a good initial guess.

Let us briefly address some strategies for solving the large sparse, non-symmetric systems of linear equations (22). In our experience, Krylov subspace methods such as BiCGSTAB and GMRES, combined with preconditioning of ILU type will do. Interestingly enough, the incomplete LU factorization of the evolution operator (19) unconditionally exists and is unique due to the M-matrix property [22]. Hence, it is advisable to use A as preconditioner for the Krylov solver even if the Jacobian matrix (25) is adopted in the outer iteration procedure.

5.1. Semi-explicit FCT limiter

The first implicit FCT algorithm for finite element discretizations on unstructured meshes [11],[12] was based on the following limiting strategy which was eventually superseded by further extensions proposed in a series of subsequent publications [13],[14]

1. Compute the high-order solution to (14) in an iterative way by solving (17) using the total amount of raw antidiffusion ($\alpha_{ij} \equiv 1$) to assemble the term f^*.
2. Evaluate the contribution of the consistent mass matrix to the raw antidiffusive fluxes (16) using the converged high-order solution as a substitute for u^{n+1}.
3. Solve the explicit subproblem $M_L \tilde{u} = Bu^n$ for the positivity-preserving intermediate solution $\tilde{u}$ which represents an explicit low-order approximation to $u(t^{n+1-\theta})$.
4. Invoke Zalesak's multidimensional FCT limiter to determine the correction factors α_{ij} so as to secure the positivity of the right-hand side as explained below.
5. Compute the final solution by solving the linear system $Au^{n+1} = b$, where

$$b_i = m_i \tilde{u}_i + \sum_{j \neq i} f_{ij}^*, \qquad f_{ij}^* = \alpha_{ij} f_{ij}. \tag{27}$$

In the fully explicit case ($\theta = 0$), we have $A = M_L$ so that $u^{n+1} = M_L^{-1} b$ can be computed explicitly from (24), and the classical FEM-FCT algorithm of Löhner *et al.* [20],[21] is recovered. The crux of the above generalization lies in the special choice of the operator A which guarantees that the positivity of the right-hand side is preserved, whence

$$\tilde{u} \geq 0 \quad \Rightarrow \quad b \geq 0 \quad \Rightarrow \quad u^{n+1} = A^{-1} b \geq 0. \tag{28}$$

The flux correction process starts with an optional 'prelimiting' of the raw antidiffusive fluxes f_{ij}. It consists of cancelling the 'wrong' ones which tend to flatten the intermediate solution and create numerical artifacts. The required adjustment is given by [17]

$$f'_{ij} := \max\{0, p_{ij}\}(\tilde{u}_i - \tilde{u}_j), \qquad p_{ij} = f_{ij}/(\tilde{u}_i - \tilde{u}_j). \tag{29}$$

The remaining fluxes are truly antidiffusive and need to be limited. The upper and lower bounds to be imposed on the net antidiffusive flux depend on the local extrema

$$\tilde{u}_i^{\max} = \max_{j \in S_i} \tilde{u}_j, \qquad \tilde{u}_i^{\min} = \min_{j \in S_i} \tilde{u}_j, \tag{30}$$

where $S_i = \{j \,|\, (\varphi_i, \varphi_j) \neq 0\}$ denotes the set of nodes which share an element with node i.

In the worst case, all antidiffusive fluxes into node i have the same sign. Hence, it is worthwhile to treat the positive and negative ones separately, as proposed by Zalesak [29]

1. Evaluate the sums of all positive and negative antidiffusive fluxes into node i

$$P_i^+ = \sum_{j \neq i} \max\{0, f'_{ij}\}, \qquad P_i^- = \sum_{j \neq i} \min\{0, f'_{ij}\}. \tag{31}$$

2. Compute the distance to a local maximum/minimum of the low-order solution

$$Q_i^+ = \tilde{u}_i^{\max} - \tilde{u}_i, \qquad Q_i^- = \tilde{u}_i^{\min} - \tilde{u}_i. \tag{32}$$

3. Calculate the nodal correction factors which prevent overshoots/undershoots

$$R_i^+ = \min\{1, m_i Q_i^+/P_i^+\}, \qquad R_i^- = \min\{1, m_i Q_i^-/P_i^-\}. \tag{33}$$

4. Check the sign of f'_{ij} and apply $R_i^{\pm}$ or $R_j^{\mp}$, whichever is smaller, so that

$$\alpha_{ij} = \begin{cases} \min\{R_i^+, R_j^-\}, & \text{if } f'_{ij} > 0, \\ \min\{R_i^-, R_j^+\}, & \text{otherwise.} \end{cases} \tag{34}$$

This symmetric limiting strategy guarantees that the corrected right-hand side (27) satisfies the constraint $\tilde{u}_i^{\min} \leq b_i/m_i \leq \tilde{u}_i^{\max}$. Due to the fact that the low-order operator A was designed to be an M-matrix, the resulting scheme proves positivity-preserving [12],[16].

It is worth mentioning that the constituents of the sums $P_i^{\pm}$ vary with Δt, while the corresponding upper/lower bounds $Q_i^{\pm}$ are fixed. Consequently, the correction factors α_{ij} produced by Zalesak's limiter depend on the time step. This dependence, which is typical of FCT methods, turns out to be a blessing and a curse at the same time. On the one hand, a larger portion of the raw antidiffusive flux f_{ij} may be retained as the time step is refined. On the other hand, the accuracy of FCT algorithms deteriorates as Δt increases, since the positivity constraint (2) becomes too restrictive. The iterative limiting strategy proposed in [14] alleviates this problem to some extent by adjusting the correction factors α_{ij} in each outer iteration so as to recycle the rejected antidiffusion step-by-step. However, the cost of iterative flux correction is rather high and severe convergence problems may occur. Therefore, other limiting techniques such as the general-purpose (GP) flux limiter introduced in [17] are to be preferred for marching the solution to a steady state.

5.2. Semi-implicit FCT limiter

For truly time-dependent problems, the use of moderately small time steps is dictated by accuracy considerations so that flux limiting of FCT type is appropriate. In this case, the underlying time-stepping method should provide (unconditional) stability and be at least second-order accurate in order to capture the evolutionary details. For this reason, we favor an implicit time discretization of Crank-Nicolson type ($\theta = 1/2$) and mention the strongly A-stable fractional-step $\theta-$scheme [25] as a promising alternative.

The semi-explicit limiting strategy presented in the previous section can be classified as an algorithm of predictor-corrector type since the implicit part of the raw antidiffusive flux (16) is evaluated using the converged high-order solution in place of u^{n+1}. This handy linearization, which can be traced back to the classical FEM-FCT procedure [20], makes it possible to perform flux correction in a very efficient way, since Zalesak's limiter is invoked just once per time step. However, a lot of CPU time needs to be invested in the iterative solution of the ill-conditioned high-order system and the convergence may even fail if the time step is too large. Moreover, the final solution fails to satisfy the nonlinear algebraic system (17) upon substitution. On the other hand, an update of the auxiliary quantities $P_i^{\pm}$, $Q_i^{\pm}$, and $R_i^{\pm}$ in each outer iteration would trigger the cost of flux limiting and compromise the benefits of implicit time-stepping. In order to circumvent this problem, let us introduce a semi-implicit FCT algorithm [18] which can be implemented as follows:

- At the first outer iteration ($m = 1$), compute a set of antidiffusive fluxes $\tilde{f}_{ij}$ which provide an explicit estimate for the admissible magnitude of $f_{ij}^* = \alpha_{ij} f_{ij}$

    1. Initialize all auxiliary arrays by zeros: $P_i^{\pm} \equiv 0,\ Q_i^{\pm} \equiv 0,\ R_i^{\pm} \equiv 0.$

2. Compute the positivity-preserving intermediate solution of low order

$$\tilde{u} = u^n + (1-\theta)\Delta t M_L^{-1} L u^n. \tag{35}$$

3. For each pair of neighboring nodes i and j, evaluate the raw antidiffusive flux

$$f_{ij}^n = \Delta t d_{ij}^n (u_i^n - u_j^n) \tag{36}$$

and add its contribution to the sums of positive/negative edge contributions

$$P_i^{\pm} := P_i^{\pm} + \begin{matrix}\max\\ \min\end{matrix} \{0, f_{ij}^n\}, \qquad P_j^{\pm} := P_j^{\pm} + \begin{matrix}\max\\ \min\end{matrix} \{0, -f_{ij}^n\}. \tag{37}$$

4. Update the maximum/minimum admissible increments for both nodes

$$Q_i^{\pm} := \begin{matrix}\max\\ \min\end{matrix} \left\{Q_i^{\pm}, \tilde{u}_j - \tilde{u}_i\right\}, \qquad Q_j^{\pm} := \begin{matrix}\max\\ \min\end{matrix} \left\{Q_j^{\pm}, \tilde{u}_i - \tilde{u}_j\right\}. \tag{38}$$

5. Relax the constraint $R_i^{\pm} \leq 1$ for the nodal correction factors and compute

$$R_i^{\pm} := m_i Q_i^{\pm} / P_i^{\pm}. \tag{39}$$

6. Multiply the raw antidiffusive fluxes f_{ij}^n by the minimum of $R_i^{\pm}$ and $R_j^{\mp}$

$$\tilde{f}_{ij} = \begin{cases} \min\{R_i^+, R_j^-\} f_{ij}^n, & \text{if } f_{ij}^n > 0, \\ \min\{R_i^-, R_j^+\} f_{ij}^n, & \text{otherwise.} \end{cases} \tag{40}$$

- At each outer iteration ($m = 1, 2, \dots$), assemble f^* and plug it into (20)

1. Update the target flux (16) using the solution from the previous iteration

$$\begin{aligned} f_{ij} &= [m_{ij} + \theta \Delta t d_{ij}^{(m)}](u_i^{(m)} - u_j^{(m)}) \\ &- [m_{ij} - (1-\theta)\Delta t d_{ij}^n]\ (u_i^n - u_j^n). \end{aligned} \tag{41}$$

2. Constrain each flux f_{ij} so that its magnitude is bounded by that of $\tilde{f}_{ij}$

$$f_{ij}^* = \begin{cases} \min\{f_{ij}, \max\{0, \tilde{f}_{ij}\}\}, & \text{if } f_{ij} > 0, \\ \max\{f_{ij}, \min\{0, \tilde{f}_{ij}\}\}, & \text{otherwise.} \end{cases} \tag{42}$$

3. Insert the limited antidiffusive fluxes f_{ij}^* into the right-hand side (20)

$$b_i^{(m)} := b_i^{(m)} + f_{ij}^*, \qquad b_j^{(m)} := b_j^{(m)} - f_{ij}^*. \tag{43}$$

Due to the fact that f_{ij}^n is not the real target flux but merely an explicit predictor used to estimate the maximum amount of admissible antidiffusion, the multipliers $R_i^{\pm}$ are redefined so that the ratio $\tilde{f}_{ij}/f_{ij}^n$ may exceed unity. However, the effective correction factors $\alpha_{ij} := f_{ij}^*/f_{ij}$ are bounded by 0 and 1, as required for consistency.

Instead of computing the optimal upper/lower bounds (32) for a given time step, it is also possible to use some reasonable fixed bounds and adjust the time step if this is necessary to satisfy a CFL-like condition (as in the case of TVD methods). For instance, the auxiliary quantities $Q_i^{\pm}$ can be computed using u^n instead of $\tilde{u}$

$$Q_i^+ = \max_{j \in S_i} u_j^n - u_i^n, \qquad Q_i^- = \min_{j \in S_i} u_j^n - u_i^n. \tag{44}$$

The corresponding nodal correction factors $R_i^\pm$ should be redefined as [17]

$$R_i^\pm = (m_i - m_{ii})Q_i^\pm / P_i^\pm, \tag{45}$$

where $m_i - m_{ii} = \sum_{j\neq i} m_{ij}$ is the difference between the diagonal entries of the consistent and lumped mass matrices. This modification eliminates the need for evaluation of the intermediate solution $\tilde{u}$ in (35) and leads to a single-step FCT algorithm.

For a given time step, the multipliers (45) will typically be smaller than those defined by (39). However, in either case the denominator $P_i^\pm$ is proportional to Δt. Therefore, the difference between the effective correction factors α_{ij} will shrink and eventually vanish as the time step is refined. As long as Δt is sufficiently small, the accuracy of both FCT techniques depends solely on the choice of the underlying high-order scheme.

5.3. Positivity proof

The positivity proof for the semi-implicit FCT algorithm (35)–(43) follows that for the classical Zalesak limiter, see [12],[16]. In the nontrivial case $f_i^* \neq 0$, the i–th component of the right-hand side (20) admits the following representation

$$b_i^* = m_i\tilde{u}_i + f_i^* = (m_i - \alpha_i)\tilde{u}_i + \alpha_i\tilde{u}_k, \tag{46}$$

where the coefficient $\alpha_i = f_i^*/(\tilde{u}_k - \tilde{u}_i)$ is defined in terms of the local extremum

$$\tilde{u}_k = \begin{cases} \tilde{u}_i^{\max}, & \text{if } f_i^* > 0, \\ \tilde{u}_i^{\min}, & \text{if } f_i^* < 0. \end{cases} \tag{47}$$

This definition implies that $f_i^* = \alpha_i Q_i^\pm$, where $\alpha_i > 0$. By virtue of (46), the sign of the intermediate solution $\tilde{u}$ is preserved if the inequality $m_i - \alpha_i \geq 0$ holds.

In the case $f_i^* < 0$, the antidiffusive correction to node i is bounded from below by

$$m_iQ_i^- \leq R_i^-P_i^- \leq \sum_{j\neq i} \min\{0, \tilde{f}_{ij}\} \leq f_i^* = \alpha_i Q_i^-. \tag{48}$$

Likewise, a strictly positive antidiffusive correction $f_i^* > 0$ is bounded from above by

$$\alpha_i Q_i^+ = f_i^* \leq \sum_{j\neq i} \max\{0, \tilde{f}_{ij}\} \leq R_i^+P_i^+ \leq m_iQ_i^+. \tag{49}$$

It follows that $0 \leq \alpha_i \leq m_i$, which proves that $b_i^* \geq 0$ provided that $\tilde{u}_i \geq 0$ and $\tilde{u}_k \geq 0$.

In light of the above, the semi-implicit FCT limiter is positivity-preserving as long as the diagonal coefficients of the matrix B as defined in (20) are nonnegative. The corresponding CFL-like condition (4) for the maximum admissible time step reads

$$(1-\theta)\Delta t \leq \min_i |m_i/l_{ii}|. \tag{50}$$

The positivity of the single-step algorithm based on the slack bounds (44)–(45) can be proven in a similar way using the following representation of the right-hand side

$$b_i^* = (m_i - \alpha_i)u_i^n + \alpha_i u_k^n + (1-\theta)\Delta t \sum_j l_{ij}u_j^n. \tag{51}$$

In this case, the limited antidiffusive correction to node i can be estimated as follows

$$(m_i - m_{ii})Q_i^- \leq f_i^* \leq (m_i - m_{ii})Q_i^+ \tag{52}$$

so that $m_i - \alpha_i \geq m_{ii}$. Thus, the right-hand side given by (51) preserves the sign of u^n if the time step satisfies the positivity constraint for all diagonal coefficients

$$(1-\theta)\Delta t \leq \min_i |m_{ii}/l_{ii}|. \tag{53}$$

Under the above conditions, the M-matrix property of the low-order operator (19) is sufficient to guarantee that *each* solution update is positivity-preserving if the fixed-point iteration (17) is preconditioned by $C^{(m)} = A$, $\forall m$. On the other hand, only the fully converged solution is certain to remain positive if Newton's method ($C^{(m)} = J^{(m)}$, $\forall m$) is employed.

5.4. Calculation of Jacobians

So far, the evaluation of the Jacobian matrix necessary for Newton's method has not been addressed. The formal definition (25) requires the 'differentiation' of the antidiffusive contribution (21) which is constructed at the fully discrete level, see (27)–(34) and/or (35)–(43). Due to the lack of a continuous counterpart that could be differentiated 'by hand' no analytical expression for the Jacobian matrix is available. Furthermore, the derivative of the low-order operator $A(u)u$ needs to be considered if the governing equation is nonlinear.

To this end, let us split the Jacobian operator J into its low-order part $T = \{t_{ij}\}$ and the contribution of the nonlinear flux limiter $T^* = \{t_{ij}^*\}$, which yields

$$J = T + T^* + \mathcal{O}(\sigma^2), \qquad t_{ij} \simeq \frac{\partial (A(u)u)_i}{\partial u_j}, \qquad t_{ij}^* \simeq -\frac{\partial f_i^*(u, u^n)}{\partial u_j}. \tag{54}$$

Here, σ is a small scalar to be specified below. For our purposes, it is worthwhile to introduce the central difference operator $\mathscr{D}_k[\,\cdot\,]$ for a generic function $f : \mathbb{R}^n \to \mathbb{R}$ according to

$$\mathscr{D}_k[f(u)] := \frac{f(u + \sigma e_k) - f(u - \sigma e_k)}{2\sigma}, \tag{55}$$

where e_k denotes the k-th unit vector. As a result, each entry of the differentiated evolution operator $A(u)+A'(u)u$ can be approximated with second-order accuracy by $t_{ik} = \mathscr{D}_k[(A(u)u)_i]$. Recall that the right-hand side of equation (11) represents the convective contribution to node i which results from the application of the modified transport operator L to the vector of nodal values u. Substitution of the decomposed matrix-vector product $(L(u)u)_i$ into the approximate derivative (54) and some tedious algebraic manipulations lead to [23]

$$t_{ik} = \delta_{ik} m_i - \theta \Delta t \mu_{ik} - \theta \Delta t \sum_j \mathscr{D}_k[l_{ij}(u)] u_j, \tag{56}$$

where $\delta_{ik} \in \{0, 1\}$ denotes the standard Kronecker delta symbol and the auxiliary quantity μ_{ik} stands for the average of the perturbed evolution coefficients resulting from discrete upwinding

$$\mu_{ik} = \frac{l_{ik}(u + \sigma e_k) + l_{ik}(u - \sigma e_k)}{2}, \qquad \forall\ k \neq i. \tag{57}$$

Moreover, it follows from (10) that the average term on the diagonal of T is given by

$$\mu_{ii} = \frac{k_{ii}(u + \sigma e_i) + k_{ii}(u - \sigma e_i)}{2} - \sum_{j \neq i} \frac{d_{ij}(u + \sigma e_i) + d_{ij}(u - \sigma e_i)}{2}. \tag{58}$$

Interestingly enough, expression (56) is very similar to that obtained by applying the central difference approximation (55) directly to the differentiated evolution operator $A(u)+A'(u)u$. This can be readily seen by considering the following sequence of equations

$$\begin{aligned} \frac{\partial (A(u)u)_i}{\partial u_k} &= a_{ik} + \left(\left[\frac{\partial A}{\partial u_k}\right] u\right)_i & (59)\\ &= \delta_{ik} m_i - \theta\Delta t l_{ik} - \theta \Delta t \sum_j \left[\frac{\partial L}{\partial u_k}\right]_{ij} u_j. & (60) \end{aligned}$$

The main difference between equations (60) and (56) consists in the treatment of the second term on the right-hand side. Note that it is also possible to abstain from averaging of the perturbed low-order coefficients in expression (56) and employ $\mu_{ij} = l_{ij}$. In any case, it is noteworthy that the above definition of the approximate Jacobian matrix for the low-order evolution operator A automatically reduces to $T = A$ if the governing equation is linear.

It remains to consider the contribution of the antidiffusive fluxes. In essence, the k-th column of $T^* = \{t^*_{ij}\}$ could be readily constructed by taking the difference between the corrected fluxes (21) evaluated at $u + \sigma e_k$ and $u - \sigma e_k$ and scale the result by 2σ. However, this approach is prohibitively expensive since it does not exploit the sparsity of the Jacobian matrix which is known *a priori* [23]. As a rule, node-oriented flux limiters give rise to some fill-in of the Jacobian, i.e., the stencil of T^* is wider than that of A. In particular, this turns out to be the case for fully implicit algebraic flux correction schemes of FCT and TVD type [15],[16],[17] but the semi-implicit limiting strategy (35)–(43) is free of this drawback (see below). The semi-explicit algorithm based on (27)–(34) is of predictor-corrector type and yields a non-converged solution, so that Newton's method is not applicable at the flux correction step.

Let $X = \mathcal{G}(A) \in \{0,1\}^{n\times n}$ denote the adjacency graph of the global stiffness matrix A. Consequently, $x_{ij} = 1$ iff the finite element basis functions φ_i and φ_j have overlapping supports, that is, if there exists an edge ij. Despite the fact that A is a non-symmetric matrix, its sparsity pattern X is symmetric so that $x_{ij} = x_{ji}$. To construct the approximate Jacobian J, let us perturb the solution vector u at some node, say k, and evaluate the corrected fluxes (21) for $u + \sigma e_k$ following the semi-explicit limiting strategy presented in section 5.1. By construction, the nodal correction factors $R_i^\pm$ defined in (33) may be affected by the perturbation for all $i \in \mathcal{S}_k$. As a result, the final correction factors α_{ij} may differ from their unperturbed counterparts if at least one of the nodes i and j belongs to $\mathcal{S}_k$. With this observation in mind, the enlarged sparsity pattern of J satisfies the following relation

$$\mathcal{G}(J) = \mathcal{G}(Y), \qquad \text{where} \qquad Y = X^2 \in \{0,1,2\}^{n\times n}. \tag{61}$$

It is easy to verify that $y_{ij} > 0$ if and only if there exists a path of edges with length not greater than two such that node i is reachable from j and vice versa

$$y_{ij} = \sum_k x_{ik} x_{kj} > 0 \qquad \Leftrightarrow \qquad \exists k\, :\, x_{ik} = 1 \wedge x_{kj} = 1. \tag{62}$$

Let us remark, that the impact of 'joggling' the solution value at some node k usually propagates along two edges by virtue of the correction factors which need to be recalculated in each iteration. As we are about to see, this is not the case for the semi-implicit FEM-FCT algorithm introduced in Section 5.2 which will turn out to be the method of choice within a discrete Newton approach. Recall that the antidiffusive fluxes $\tilde{f}_{ij}$ are computed once and for all at the beginning of each time step, see (35)–(40). In fact, they are used to constrain the target fluxes (16) instead of adopting some multiplier α_{ij} which depends on the solution $u^{(m)}$. As a consequence, the sparsity pattern of the global evolution operator A is inherited by the Jacobian matrix J. This leads to substantial savings in terms of memory requirement as compared to the extended sparsity pattern (61). Even more important is the increase of total efficiency which follows from the fact, that both the assembly of J and its application within a matrix-vector multiplication requires less CPU time. Let us present an efficient algorithm for assembling the operator T^* for the semi-implicit FCT limiter in an edge-based fashion:

- At each outer iteration ($m = 1, 2, \dots$), initialize $T^* \equiv 0$ and rebuild it in a loop over edges ij. In each step, let $k = i, j$ one after the other:

  1. Evaluate the explicit antidiffusive contribution which does not depend on $u^{(m)}$
  $$f_{ij}^n = [m_{ij} - (1-\theta)\Delta t d_{ij}^n](u_i^n - u_j^n) \tag{63}$$
  2. Compute the auxiliary coefficients using the solution from the previous iteration
  $$a_{ij}^k = m_{ij} + \theta\Delta t d_{ij}(u^{(m)} + \sigma e_k), \qquad b_{ij}^k = m_{ij} + \theta\Delta t d_{ij}(u^{(m)} - \sigma e_k) \tag{64}$$
  3. Update the perturbed target fluxes (16) depending on the index k
  $$g_{ij}^k = \begin{cases} a_{ij}^k(u_i^{(m)} - u_j^{(m)} + \sigma) + f_{ij}^n & \text{if } k = i, \\ a_{ij}^k(u_i^{(m)} - u_j^{(m)} - \sigma) + f_{ij}^n & \text{otherwise,} \end{cases} \tag{65}$$
  $$h_{ij}^k = \begin{cases} b_{ij}^k(u_i^{(m)} - u_j^{(m)} - \sigma) + f_{ij}^n & \text{if } k = i, \\ b_{ij}^k(u_i^{(m)} - u_j^{(m)} + \sigma) + f_{ij}^n & \text{otherwise.} \end{cases} \tag{66}$$
  4. Constrain each flux g_{ij}^k and h_{ij}^k so that its magnitude is bounded by that of $\tilde{f}_{ij}$
  $$g_{ij}^* = \begin{cases} \min\{g_{ij}^k, \max\{0, \tilde{f}_{ij}\}\}, & \text{if } g_{ij}^k > 0, \\ \max\{g_{ij}^k, \min\{0, \tilde{f}_{ij}\}\}, & \text{otherwise,} \end{cases} \tag{67}$$
  $$h_{ij}^* = \begin{cases} \min\{h_{ij}^k, \max\{0, \tilde{f}_{ij}\}\}, & \text{if } h_{ij}^k > 0, \\ \max\{h_{ij}^k, \min\{0, \tilde{f}_{ij}\}\}, & \text{otherwise.} \end{cases} \tag{68}$$
  5. Compute the divided difference and insert it into the Jacobian matrix T^*
  $$f_{ij}^* = \frac{g_{ij}^* - h_{ij}^*}{2\sigma} \to \begin{cases} t_{ii}^* := t_{ii}^* - f_{ij}^*, \quad t_{ji}^* := t_{ji}^* + f_{ij}^* & \text{if } k = i, \\ t_{ij}^* := t_{ij}^* - f_{ij}^*, \quad t_{jj}^* := t_{jj}^* + f_{ij}^* & \text{otherwise.} \end{cases} \tag{69}$$

The last three steps call for further explanation. Following expression (41), the target fluxes (65)–(66) are evaluated with respect to the perturbed solution difference $(u_i \pm \sigma e_k) - (u_j \pm \sigma e_k)$ whereby k equals i and j one after the other. Obviously, this yields four combinations of $u_i - u_j \pm \sigma$ multiplied by a_{ij}^k and b_{ij}^k, respectively. Furthermore, the magnitude of the unilaterally perturbed raw antidiffusive fluxes is bounded by that of the threshold $\tilde{f}_{ij}$. Finally, the limited central difference approximation is built into the Jacobian matrix rather than applying it to the right-hand side (20). Note that (69) corresponds to inserting f_{ij}^* into the i-th and j-th row of the k-th column of T^* following the algorithmic step (43) but with opposite sign.

The above algorithm is applicable to linear and nonlinear governing equations alike. It is worth mentioning that in the linear case the auxiliary quantities a_{ij} and b_{ij} are the same for both $k = i$ and $k = j$, whereas the perturbed fluxes satisfy the following relation

$$g^i_{ij} = f_{ij} + \sigma[m_{ij} + \theta\Delta t d_{ij}] = h^j_{ij}, \tag{70}$$

$$g^j_{ij} = f_{ij} - \sigma[m_{ij} + \theta\Delta t d_{ij}] = h^i_{ij}. \tag{71}$$

Here, the unperturbed target flux f_{ij} is defined as in (41). Hence, it suffices to compute the divided difference (69) only for one index, say i, and update the Jacobian matrix as follows

$$\begin{array}{ll} t^*_{ii} := t^*_{ii} - f^*_{ij} & \qquad t^*_{ji} := t^*_{ji} + f^*_{ij}, \\ t^*_{jj} := t^*_{jj} - f^*_{ij} & \qquad t^*_{ij} := t^*_{ij} + f^*_{ij}. \end{array} \tag{72}$$

Roughly speaking, the calculation of the matrix T^* can be considered to be approximately twice as expensive as augmenting the right-hand side (20) by the antidiffusive fluxes, see (41)–(43). As we are about to see, this extra cost clearly pays off in terms of total efficiency when it comes to time accurate simulation of transient flows. Remarkably, this improvement is already observed if the evolution operator $A = M_L - \theta\Delta t L$ is constant and can be assembled once and for all at the beginning of the simulation so that the standard defect correction approach (24) does not require further matrix evaluations. The benefits of Newton's method become even more significant if the preconditioner (19) needs to be updated in each outer iteration due to a nonlinear governing equation or a linear but time-dependent velocity field $\mathbf{v} = \mathbf{v}(\mathbf{x}, t)$ so that the costs for assembling the operators T and T^* may be neglected.

5.5. Convergence behavior

A remark is in order regarding the convergence behavior of the fixed-point iteration (17). The converged solution u^{n+1} is supposed to satisfy a nonlinear algebraic system of the form

$$A^* u^{n+1} = Bu^n, \tag{73}$$

where A^* is the nonlinear FCT operator which includes some built-in antidiffusion

$$A^* u^{n+1} := Au^{n+1} - f^*. \tag{74}$$

Clearly, the rate of convergence will depend on $||A^* - C||$, that is, the approximation property of the preconditioner C. On the one hand, the operator A as defined in (19) is linear and easy to 'invert' because it is an M-matrix. On the other hand, it represents a rather poor approximation to the original Galerkin operator $M_C - \theta\Delta t K$ which is recovered in the limit $\alpha_{ij} \to 1$. As a result, the convergence of a highly accurate FCT algorithm based on the standard defect correction approach is likely to slow down as the high-order solution is approached.

In light of the above, the lumped-mass version, which is obtained by setting $m_{ij} = 0$ in the definition of the raw antidiffusive flux, converges much faster than the one based on the consistent target flux (16). However, mass lumping may have a devastating effect on the accuracy of a time-dependent solution, as demonstrated by the numerical study performed in the next section. At the same time, the high phase accuracy provided by the consistent mass matrix comes at the cost of slower convergence, due to the fact that the 'monotone' preconditioner A is based on M_L rather than M_C. The original high-order system (13) which corresponds to $\alpha_{ij} \equiv 1$ is particularly difficult to solve, even though it is linear (see below). Moreover, the number of outer iterations tends to increase as the mesh is refined.

In general, there is a trade-off between the accuracy of the numerical solution and convergence of the fixed-point iteration (24). Any modification of the flux limiter which makes it possible to accept more antidiffusion has an adverse effect on the nonlinear convergence rates. Conversely, more diffusive schemes converge much better but their accuracy leaves a lot to be desired. To overcome this shortcoming, the use of the discrete Newton method is advisable. The number of outer iterations required to drive the residual to some prescribed tolerance is drastically reduced and becomes largely independent of the grid refinement level. However, one should keep in mind that the Jacobian matrix (25) does not feature the M-matrix property so that intermediate solutions are not necessarily positivity-preserving.

6. Numerical examples

In order to evaluate the performance of the new algorithm, we apply it to several time-dependent benchmark problems discretized using the standard Galerkin method and the second-order accurate Crank-Nicolson time-stepping. After flux limiting, the order of approximation (in space and time) may vary depending on the local smoothness of the solution. The goal of this numerical study is to examine the accuracy of the resulting high-resolution scheme as well as the convergence behavior of the fixed-point iteration (17) and the implications of mass lumping. To this end, the semi-implicit FCT method (35)–(43) is compared to its semi-explicit prototype (29)–(34) and to the standard Galerkin discretization. Moreover, the standard defect correction scheme (24) and the discrete Newton approach are compared with respect to their nonlinear convergence rates as well as computational efficiency, that is, the total CPU time required to solve the nonlinear algebraic system (14) up to a prescribed tolerance.

6.1. Convection skew to the mesh

In order to study the convergence behavior of the semi-implicit and semi-explicit FEM-FCT algorithms as compared to that of the underlying Galerkin scheme, let us solve equation (5) with $\mathbf{v} = (1, 1)$ so that the initial profile is translated along the diagonal of the computational domain $\Omega = (0, 1) \times (0, 1)$. The numerical study is to be performed for two different initial configurations centered at the reference point $(x_0, y_0) = (0.3, 0.3)$

TP1 The first test problem corresponds to the discontinuous initial condition

$$u(x, y, 0) = \begin{cases} 1 & \text{if } \max\{|x - x_0|, |y - y_0|\} \leq 0.1, \\ 0 & \text{otherwise.} \end{cases} \tag{75}$$

TP2 The second test problem deals with translation of a smooth function defined as

$$u(x, y, 0) = \frac{1}{4}[1 + \cos(10\pi(x - x_0))][1 + \cos(10\pi(y - y_0))] \tag{76}$$

within the circle $\sqrt{(x - x_0)^2 + (y - y_0)^2} \leq 0.1$ and equal to zero elsewhere.

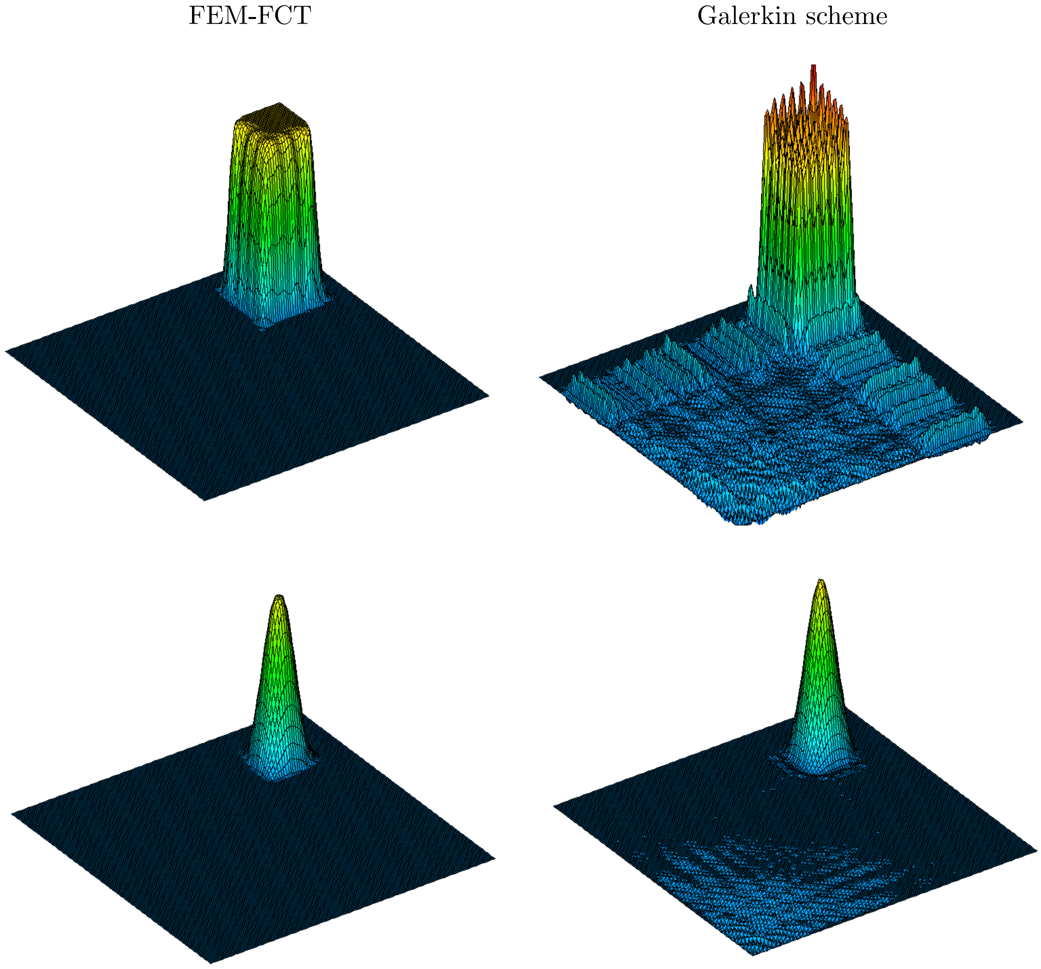
Figure 2 displays the approximate solutions at $t = 0.5$ computed using $\Delta t = 10^{-3}$ on a quadrilateral mesh consisting of 128×128 bilinear elements. The left diagrams were produced by the consistent-mass semi-implicit FCT algorithm which yields nonoscillatory solutions bounded by 0 and 1. The underlying high-order scheme remains stable but gives rise to nonphysical undershoots and overshoots, as seen in the right diagrams.

In either case, the numerical solution was computed in an iterative way using the fixed-point defect correction scheme (17) preconditioned by the low-order operator (19). The stopping criterion was based on the Euclidean norm of the residual vector

$$r = Au^{n+1} - Bu^n - f^*, \qquad ||r|| = \sqrt{r^T r} \tag{77}$$

which was required to satisfy the inequality $||r|| \leq 10^{-4}$. The difference between the exact solution u and its finite element approximation u_h was measured in the L_1-norm

$$||u - u_h||_1 = \int_\Omega |u - u_h| \, \mathrm{d}x \approx \sum_i m_i |u(x_i, y_i) - u_i| \tag{78}$$



FEM-FCT

Galerkin scheme

Figure 2. Convection skew to the mesh: 128 × 128 Q_1−elements, $t = 0.5$.

standard defect correction

NLEV	NVT	NDC	$\|\|u-u_h\|\|_1$	$\|\|u-u_h\|\|_2$	$u_{\min}$	$u_{\max}$

semi-implicit FCT / consistent mass matrix

6	4,225	2,500	1.1737e-2	6.2176e-2	0.0	1.0
7	16,641	2,461	7.3688e-3	4.8577e-2	0.0	1.0
8	66,049	2,489	4.7039e-3	3.8715e-2	0.0	1.0

semi-implicit FCT / lumped mass matrix

6	4,225	751	1.9356e-2	8.4294e-2	0.0	0.9988
7	16,641	1,000	1.2402e-2	6.5356e-2	0.0	1.0000
8	66,049	1,014	7.8511e-3	5.1182e-2	0.0	1.0000

Galerkin scheme / consistent mass matrix

6	4,225	4,666	3.6283e-2	7.4952e-2	-0.2557	1.4505
7	16,641	7,379	2.7340e-2	5.8124e-2	-0.2743	1.3797
8	66,049	13,852	2.3000e-2	5.2536e-2	-0.4437	1.4080

Galerkin scheme / lumped mass matrix

6	4,225	1,000	6.5181e-2	1.3073e-1	-0.4022	1.5608
7	16,641	1,423	4.7055e-2	9.8663e-2	-0.4340	1.5732
8	66,049	1,500	3.5126e-2	8.0298e-2	-0.3713	1.5628

semi-explicit FCT / consistent mass matrix

6	4,225	3,190	9.3328e-3	5.4115e-2	0.0	1.0
7	16,641	3,220	5.4794e-3	4.1218e-2	0.0	1.0
8	66,049	3,590	3.3680e-3	3.2369e-2	0.0	1.0

semi-explicit FCT / lumped mass matrix

6	4,225	1,500	1.9098e-2	8.3498e-2	0.0	0.9989
7	16,641	1,501	1.2422e-2	6.5348e-2	0.0	1.0000
8	66,049	1,540	7.8662e-3	5.1167e-2	0.0	1.0000

Table I. Convection skew to the mesh: discontinuous initial data.

as well as in the L_2-norm defined by the following formula

$$||u-u_h||_2^2 = \int_\Omega |u-u_h|^2 \,\mathrm{d}x \approx \sum_i m_i |u(x_i,y_i)-u_i|^2, \tag{79}$$

where $m_i = \int_\Omega \varphi_i \,\mathrm{d}x$ are the diagonal coefficients of the lumped mass matrix. Furthermore, the global minimum $u_{\min} = \min_i u_i$ and maximum $u_{\max} = \max_i u_i$ of the discrete solution u_h were compared to their analytical values 0 and 1.

Tables I and II illustrate the convergence behavior of the iterative flux/defect correction scheme as applied to the test problems TP1 and TP2 on three successively refined meshes. The first three columns in each table display the refinement level NLEV, the number of vertices/nodes NVT and the total number of outer iterations NDC required to compute the numerical solution at $t = 0.5$. The different performance of the six algorithms under consideration supports the arguments presented in Section 5.5. In particular, it can readily be seen that the use of the consistent mass matrix results in a much better accuracy but the convergence slows down, whereas the lumped-mass version is less accurate but much more efficient. If the difference $||u^{n+1} - u^n||$ is large, mass antidiffusion affects the convergence rates even stronger than the convective part of the antidiffusive flux. Since the latter is proportional to Δt, the mass lumping error plays a dominant role at small time steps such that $A \approx M_L$. On the other hand, the linear convergence rates improve since the condition number of A decreases and its diagonal dominance is enhanced as the time step is refined.

Note that the consistent-mass Galerkin scheme faces severe convergence problems and the error may even increase in the course of mesh refinement (see Table II). By contrast, the results computed by the semi-implicit FCT algorithm exhibit a monotone grid convergence as well as some improvement of the convergence rates. Even the consistent-mass version converges slowly but surely to a nonoscillatory time-accurate solution. For large time steps, the single-step implementation based on (44)–(45) would be more diffusive and converge faster. However, for time steps as small as the one employed in this section, it would be just as accurate and converge at the same rate as the algorithm (35)–(43). The values of $u_{\max}$ in Table II reveal that flux correction may lead to undesirable 'peak clipping', which is a well-known phenomenon discussed, e.g., in [16],[29]. On the other hand, the associated high-order solution is corrupted by undershoots and overshoots which are particularly large for discontinuous initial data (Table I) and less pronounced for the smooth cosine hill (Table II). These nonphysical oscillations result in a dramatic loss of accuracy and slow/no convergence, so that the results are inferior to those produced by the semi-implicit FCT algorithm using the same settings.

It is not unusual that semi-explicit flux correction (29)–(34) as applied at the end of each time step to the converged high-order predictor requires less outer iterations than the underlying Galerkin scheme (see Tables I and II). However, the residual of the flux-corrected solution can no longer be controlled and the total number of defect correction steps is considerably greater than that for the semi-implicit FCT limiter, whereas the accuracy of the resulting solutions is comparable. Of course, the linear system (13) could be solved in one step (without resorting to defect correction) but this straightforward approach would inevitably lead to a severe deterioration of the linear convergence rates. Indeed, the high-order operator $M_C - \theta\Delta t K$ is much harder to 'invert' than the preconditioner A which enjoys the M-matrix property. In many cases, the high-order solution may prove prohibitively expensive or even impossible to compute in such a brute-force way, unless a direct solver is employed. Hence, even linear high-order systems of the form (14) call for the use of iterative defect correction [12].

In order to obtain a better insight into the error reduction rate, Figure 3 displays the L_1-errors (top) and L_2-errors (bottom) of all six methods for both benchmark configurations. For each discretization, the solid line denotes the consistent mass matrix whereas the 'lumped' version is indicated by dashed lines. Obviously, the rate of convergence is the same for the implicit (circular markers) and explicit (square markers) FCT algorithm whereby the norm of the error is slightly smaller for the latter one if the consistent mass matrix is adopted. Interestingly enough, both FCT algorithms produce nearly the same results if mass lumping is performed. On the other hand, the solution produced by the high-order Galerkin scheme denotes by triangular markers is less accurate which manifests itself in greater error norms. Moreover, it suffers from severe convergence problems if the consistent mass matrix is adopted and fails completely for the second test problem if the mesh is successively refined.

The marginally better accuracy of the semi-explicit FEM-FCT scheme as compared to its semi-implicit counterpart can be attributed to the better phase characteristics of the high-order Galerkin scheme employed at the predictor step. On the other hand, the involved splitting error may become pronounced in other settings, especially as the time step is increased. Moreover, the linear and/or nonlinear convergence rates leave a lot to be desired so that the semi-implicit approach combined with the discrete Newton method is preferable in many cases.

standard defect correction

NLEV	NVT	NDC	$\|\|u - u_h\|\|_1$	$\|\|u - u_h\|\|_2$	$u_{\min}$	$u_{\max}$
semi-implicit FCT / consistent mass matrix						
6	4,225	2,486	1.4799e-3	9.2813e-3	0.0	0.8562
7	16,641	1,833	4.3436e-4	2.7820e-3	0.0	0.9418
8	66,049	2,867	1.7887e-4	1.2032e-3	0.0	0.9740
semi-implicit FCT / lumped mass matrix						
6	4,225	1,000	4.2704e-3	2.7827e-2	0.0	0.7308
7	16,641	1,000	1.7834e-3	1.1294e-2	0.0	0.9218
8	66,049	736	7.6982e-4	4.6142e-3	0.0	0.9612
Galerkin scheme / consistent mass matrix						
6	4,225	2,500	1.3961e-3	2.6234e-3	-0.0158	0.9890
7	16,641	6,437	1.8892e-3	3.9001e-3	-0.0480	0.9925
8	66,049	13,700	2.3237e-3	8.1553e-3	-0.1363	1.0012
Galerkin scheme / lumped mass matrix						
6	4,225	1,000	1.0904e-2	4.2409e-2	-0.1911	0.8809
7	16,641	1,000	3.4837e-3	1.4234e-2	-0.0811	1.0098
8	66,049	1,000	1.3092e-3	4.3179e-3	-0.0322	1.0046
semi-explicit FCT / consistent mass matrix						
6	4,225	2,651	1.0770e-3	7.6799e-3	0.0	0.8555
7	16,641	2,328	2.8414e-4	2.1692e-3	0.0	0.9471
8	66,049	3,434	1.3188e-4	9.8597e-4	0.0	0.9775
semi-explicit FCT / lumped mass matrix						
6	4,225	1,500	4.2671e-3	2.7760e-2	0.0	0.7296
7	16,641	1,500	1.7751e-3	1.1237e-2	0.0	0.9211
8	66,049	1,500	6.4767e-4	3.8591e-3	0.0	0.9653

Table II. Convection skew to the mesh: smooth initial data.

6.2. Swirling flow

Let us proceed to another two-dimensional benchmark problem proposed by LeVeque [19]. It deals with a swirling deformation of initial data by the incompressible velocity field given by

$$v_x = \sin^2(\pi x)\sin(2\pi y)g(t), \quad v_y = -\sin^2(\pi y)\sin(2\pi x)g(t).$$

The initial condition to be prescribed is a discontinuous function of the spatial coordinates which equals unity within a circular sector of $\pi/2$ radians and zero elsewhere:

$$u(x,y,0) = \begin{cases} 1 & \text{if } (x-1)^2 + (y-1)^2 < 0.8, \\ 0 & \text{otherwise.} \end{cases}$$

TP3 For the first test problem, let us employ a constant velocity profile which corresponds to

$$g(t) \equiv 1.$$

TP4 For the second test problem, we adopt a more ‘agile’ velocity field and let

$$g(t) = \cos(\pi t/T), \qquad 0 \le t \le T.$$

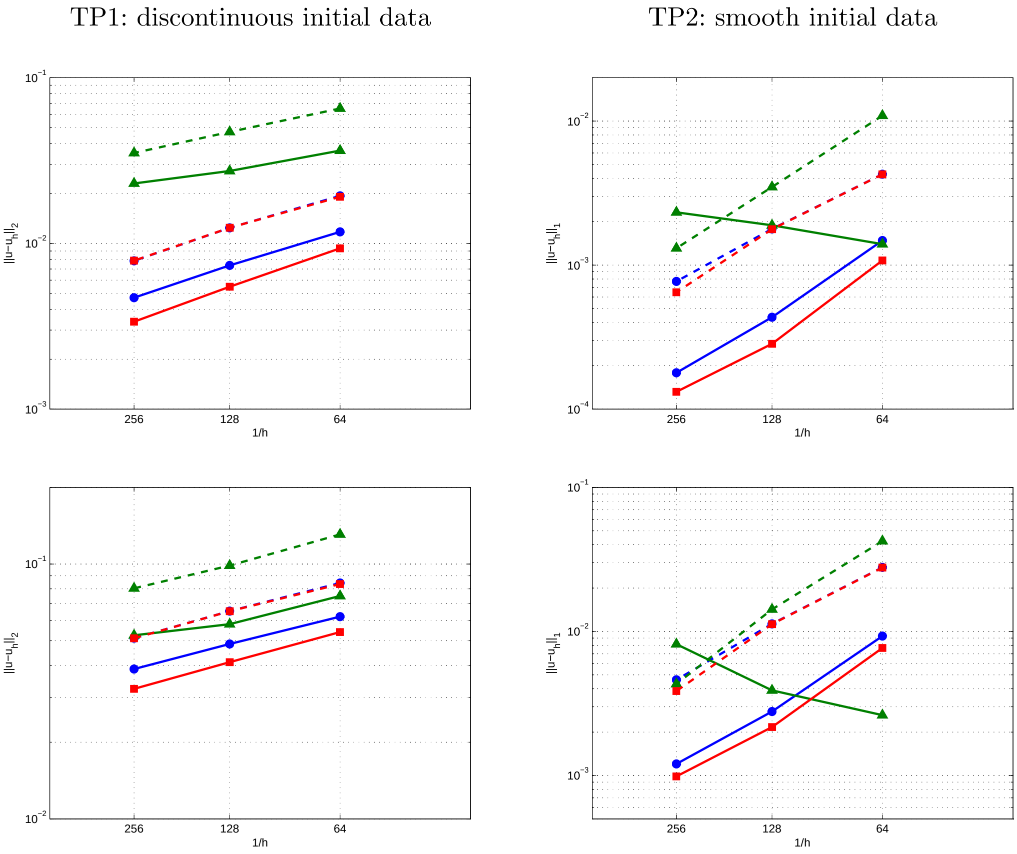
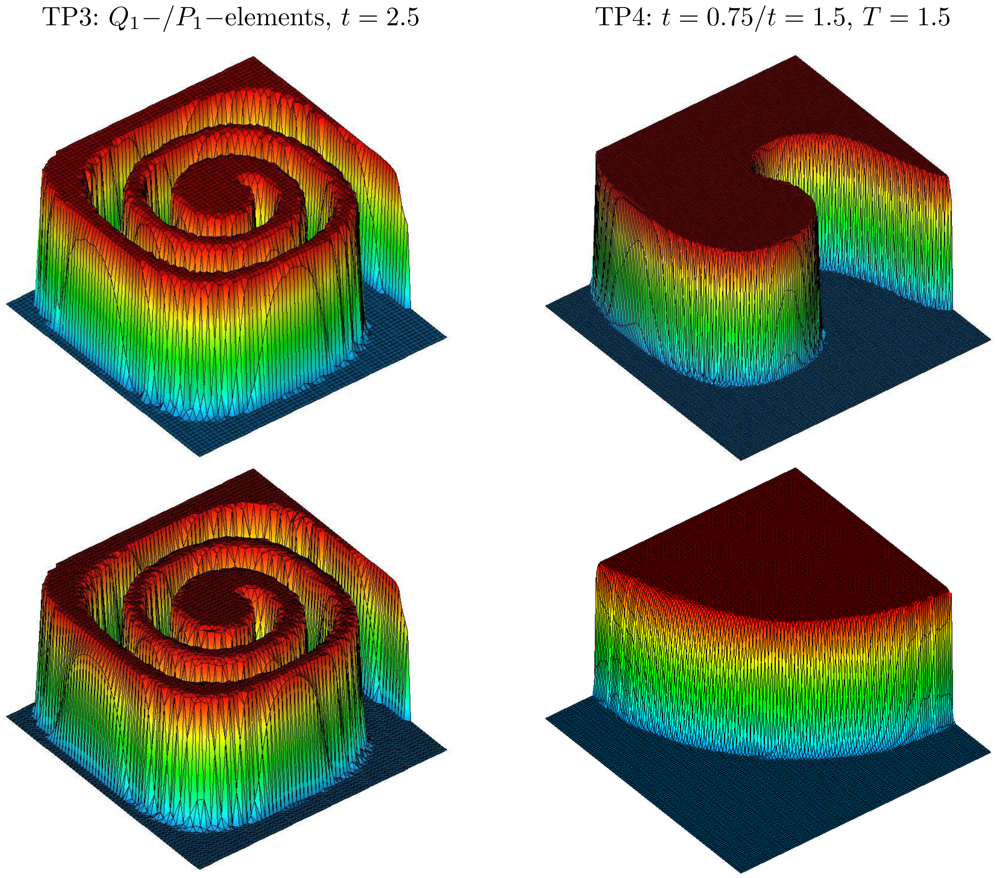


Figure 3. Convection skew to the mesh: error reduction.



TP3: Q_1-/P_1-elements, $t = 2.5$

TP4: $t = 0.75/t = 1.5$, $T = 1.5$

Figure 4. Swirling deformation: semi-implicit FEM-FCT, 128×128 finite elements.

For both benchmark configurations, the mass distribution assumes a complex spiral shape in the course of deformation. Figure 4 displays the numerical solutions calculated by the semi-implicit FCT algorithm (35)–(43) with consistent mass matrix using the time step $\Delta t = 10^{-3}$.

Recall that for TP3, the low-order evolution operator A remains constant and can be assembled once and for all at the beginning of the simulation. The numerical results at time $t = 2.5$ are computed on a uniform mesh of 128×128 bilinear finite elements and depicted in Figure 4 (top-left). The use of a piecewise-linear finite element approximation on a triangular mesh with the same number of nodes yields virtually the same solution, see Figure 4 (bottom-left). For the difference between the underlying triangulations to be visible, both profiles were output on a coarser mesh consisting of 4,225 vertices. In either case, the resolution of discontinuities is seen to be remarkably crisp. These results compare well to those presented in [16] using algebraic flux correction schemes of TVD type.

On the other hand, the velocity vector is strongly time-dependent for benchmark TP4. Right after the startup, the flow slows down and reverses at $t = T/2$ such that the initial profile is recovered as exact solution at the final time $t = T$, that is, $u(x, y, T) = u(x, y, 0)$. The value

NLEV	NVT	CPU	NN	NN/Δt	NL	NL/NN	$u_{\min}$	$u_{\max}$
$\|r\| \leq 10^{-4}$								
5	1,089	6	2,500	1.0	2,500	1.0	0.0	1.0
6	4,225	23	2,500	1.0	5,000	2.0	0.0	1.0
7	16,641	95	2,500	1.0	5,000	2.0	0.0	1.0
8	66,049	422	2,500	1.0	5,000	2.0	0.0	1.0
$\|r\| \leq 10^{-8}$								
5	1,089	52	46,809	18.72	46,820	1.0	0.0	1.0
6	4,225	200	47,046	18.82	49,546	1.05	0.0	1.0
7	16,641	896	44,827	17.93	51,858	1.17	0.0	1.0
8	66,049	4,212	43,405	17.36	63,425	1.46	0.0	1.0
$\|r\| \leq 10^{-12}$								
5	1,089	148	142,586	57.03	142,597	1.0	0.0	1.0
6	4,225	676	160,254	64.10	162,754	1.01	0.0	1.0
7	16,641	3,311	177,602	71.04	184,550	1.04	0.0	1.0
8	66,049	15,875	192,895	77.16	212,495	1.10	0.0	1.0

Table III. TP3: semi-implicit FCT with consistent mass matrix, defect correction.

$T = 1.5$ is used which corresponds to performing $1,500$ time steps of size $\Delta t = 10^{-3}$. The numerical solutions at $t = 0.75$ and $t = 1.5$, which are displayed in Figure 4 (right), were calculated on a mesh of 128×128 bilinear finite elements by the semi-implicit FCT algorithm with the consistent mass matrix. The solution profiles resulting from the application of the lumped mass matrix are slightly more diffusive but 'look' quite similar.

For these two benchmark configurations, we performed an in-depth convergence study on four successively refined quadrilateral meshes. A detailed comparison between the standard defect correction method and the discrete Newton approach is presented in Tables III–V.

As before, the first two columns display the refinement level NLEV and the number of vertices/nodes NVT. All tests were performed on an Intel Core Duo T2400 (1.83 GHz, FSB 667 MHz) processor with 1024 MB (553 MHz) of system memory. The code was compiled with the Intel Fortran 9.1 Compiler for Linux making use of the `-fast` switch which yields the best results for this setup. The total CPU time (in seconds) required to reduce the norm of the nonlinear residual to the prescribed tolerance in each time step is given in the third column. In the next four columns, the total number of nonlinear iterations (NN), the number of nonlinear iterations per time step (NL/Δt), the total number of linear iterations (NL) and the number of linear iterations per nonlinear iteration (NL/NN) are displayed in successive order. Due to the lack of an exact solution for this benchmark configuration only the global minimum and maximum of the discrete solution u_h are compared to their analytical values 0 and 1.

It can be seen from Table III that the convergence behavior of the standard defect correction scheme deteriorates significantly if the tolerance for the residual norm is reduced from 10^{-8} to 10^{-12}. Moreover, for the latter one, the number of outer iterations increases if the mesh is successively refined. On the other hand, the minimal and maximal solution values perfectly

NLEV	NVT	CPU	NN	NN/Δt	NL	NL/NN	$u_{\min}$	$u_{\max}$
				$\|r\| \leq 10^{-4}$				
5	1,089	7	2,500	1.0	2,500	1.0	-1.789e-02	1.026
6	4,225	24	2,500	1.0	2,500	1.0	-9.153e-03	1.054
7	16,641	102	2,500	1.0	2,500	1.0	-3.906e-02	1.121
8	66,049	442	2,500	1.0	2,500	1.0	-5.087e-02	1.202
				$\|r\| \leq 10^{-8}$				
5	1,089	32	9,891	3.96	44,547	4.50	-1.319e-11	1.0
6	4,225	138	9,917	3.97	48,379	4.88	-1.430e-09	1.0
7	16,641	611	9,294	3.72	49,464	5.32	-5.427e-12	1.0
8	66,049	2,736	8,974	3.59	50,991	5.68	-8.505e-09	1.0
				$\|r\| \leq 10^{-12}$				
5	1,089	84	25,640	10.26	123,412	4.81	0.0	1.0
6	4,225	369	26,538	10.62	141,645	5.34	0.0	1.0
7	16,641	1,674	25,061	10.02	146,212	5.83	0.0	1.0
8	66,049	7,113	22,287	8.91	139,868	6.28	0.0	1.0

Table IV. TP3: semi-implicit FCT with consistent mass matrix, Newton's method, $\eta = 10^{-4}$.

match their analytical bounds 0 and 1 due to the M-matrix property of A.

The convergence behavior of the discrete Newton approach making use of a constant forcing term $\eta = 10^{-4}$ as suggested in [3] is displayed in Table IV. This choice is quite restrictive and requires uniformly close approximations of Newton steps in each nonlinear iteration. It reportedly yields local q-linear convergence in some special norm [6]. As compared to the defect correction approach, the number of outer iterations is drastically reduced for all prescribed tolerances and, in addition, it does not increase for finer grids. Based on the moderate number of linear sub-iterations we believe that the ILU-decomposition of the monotone evolution operator A constitutes an appropriate preconditioner for the employed BiCGSTAB algorithm. We would like to emphasize that convergence of the fixed-point iteration is a prerequisite for the Newton method to remain positivity-preserving. This is best illustrated by the unsatisfactory minimal and maximal solution values for the loose residual tolerance 10^{-4}.

Let us briefly address the phenomenon of *oversolving* [6] the linear subproblems. To this end, we relax the forcing term $\eta = 10^{-1}$ and leave all other parameters unchanged. The results computed by the discrete Newton method are shown in Table V. The nonlinear convergence behavior is quite similar to that observed for the more restrictive choice $\eta = 10^{-4}$. However, the number of inner iterations is reduced by a factor of $2.5 - 3$, which results in a significant reduction of the overall CPU time. Our experiments with different strategies for choosing the forcing term η adaptively [6] and even solving the linear subproblems directly [26] revealed the fact, that the simplest choice $\eta = 10^{-1}$ yields the most competitive results in terms of overall performance for this class of time-dependent flows. On the one hand, the time step $\Delta t = 10^{-3}$ was chosen moderately small to resolve the temporal evolution with high precision. On the other hand, the amount of antidiffusion accepted by the FCT flux limiter is inversely

NLEV	NVT	CPU	NN	NN/Δt	NL	NL/NN	$u_{\min}$	$u_{\max}$
$\|r\| \leq 10^{-4}$								
5	1,089	7	2,500	1.0	2,500	1.0	-1.789e-02	1.026
6	4,225	24	2,500	1.0	2,500	1.0	-9.153e-03	1.054
7	16,641	104	2,500	1.0	2,500	1.0	-3.906e-02	1.121
8	66,049	443	2,500	1.0	2,500	1.0	-5.085e-02	1.202
$\|r\| \leq 10^{-8}$								
5	1,089	22	10,008	4.0	17,436	1.74	-4.087e-10	1.0
6	4,225	85	9,997	4.0	17,574	1.76	-1.165e-09	1.0
7	16,641	370	9,938	3.98	17,944	1.81	-9.854e-09	1.0
8	66,049	1,597	9,472	3.79	18,005	1.90	-1.797e-07	1.0
$\|r\| \leq 10^{-12}$								
5	1,089	56	26,448	10.75	45,770	1.70	0.0	1.0
6	4,225	223	27,697	11.08	48,512	1.75	0.0	1.0
7	16,641	939	25,922	10.37	49,030	1.89	0.0	1.0
8	66,049	3,787	22,540	9.02	47,634	2.11	0.0	1.0

Table V. TP3: semi-implicit FCT with consistent mass matrix, Newton's method, $\eta = 10^{-1}$.

proportional to Δt so that the computed solution profiles become more diffusive if larger time steps are employed. Consequently, the convergence rates of the defect correction method and of the discrete Newton algorithm improve but the solution is smeared by numerical diffusion.

It is well known that choosing an appropriate perturbation parameter σ is a delicate task. In our simulations we employed $\sigma = [(1 + \|u\|)\epsilon]^{1/3}$, where ϵ denotes the machine precision, as proposed by Pernice *et al.* and successfully used in the NITSOL package [24]. In order to investigate the influence of this 'free' parameter we repeated the simulation on mesh level 7 for fixed parameter values $\sigma = \epsilon$ and $\sigma = 0.01$, respectively. Figure 5 (left) displays the nonlinear convergence behavior for the different solution strategies. The creeping defect correction method is marked by stars whereas circles stand for the most rapidly converging Newton method ($\eta = 10^{-1}$, $\sigma = \epsilon$). Using machine precision as perturbation parameter works for this test case, but it is likely to diverge in other situations due to round-off errors and, hence, not to be recommended in general. The strategy proposed by Pernice *et al.* (square markers) requires slightly more nonlinear steps but turns out to be more robust. Furthermore, the devastating effect of choosing the perturbation parameter too large, e.g., $\sigma = 0.01$, is illustrated by the fourth curve (triangles). If σ is increased even further, the convergence of Newton's method slows down until it resembles that of the defect correction approach.

Another quantity of interest is the computing time per time step spent for each vertex, which is illustrated in Figure 5 (right). Here, the circles correspond to the standard defect correction approach whereas triangles and squares stand for Newton's method making use of the forcing term $\eta = 10^{-4}$ and $\eta = 10^{-1}$, respectively. The three curves plotted for each method denote the different tolerances for the nonlinear residual. It is worth mentioning that for the least restrictive choice $\eta = 10^{-1}$, the nodal CPU time remains nearly constant if the number of

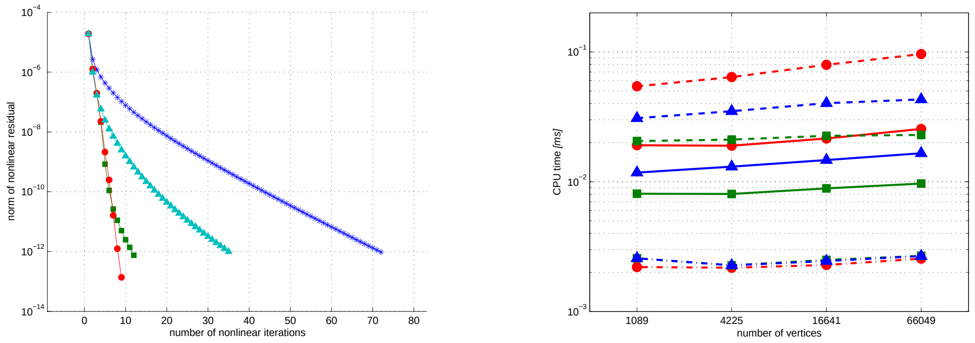


Figure 5. TP3: influence of σ, $t \in [1.0, 1.0 + \Delta t]$ / nodal CPU time.

vertices is increased, whereas a systematic growth is observed for both other methods.

Table VI illustrates the convergence behavior of the different solution methods and the errors of the finite element approximation u_h at time $t = 1.5$ for our benchmark configuration TP4. For all computations, a moderate stopping criterion $\|r\| \leq 10^{-8}$ was used and the approved forcing strategy $\eta = 10^{-1}$ was adopted for Newton's method. Moreover, the perturbation parameter σ was computed as proposed by Pernice *et al.* [24] and utilized for the divided difference approximation. All other parameter settings, e.g., the configuration of the linear solver, remain unchanged. It can be readily seen that the discrete Newton approach outperforms the standard defect correction scheme in all situations.

Figure 6 (left), illustrates the CPU time spent per node in each time step which remains nearly constant for all mesh levels. As before, the circular markers correspond to the standard defect correction method, whereas squares are used for the discrete Newton approach. Here, the dashed lines represent the lumped-mass versions of the two algorithms. The significant overhead costs of the slowly converging defect correction method are clearly visible. The solution errors, which are virtually the same for both nonlinear solution strategies, exhibit a monotone reduction on sufficiently fine meshes as illustrated in Figure 6 (right).

7. Conclusions

The semi-implicit approach to flux correction of FCT type leads to a robust and efficient special-purpose algorithm for time-dependent problems discretized in space by the finite element method. The accuracy of the resulting scheme improves as the time step is refined and the consistent mass matrix can be included in a positivity-preserving fashion. The new limiting strategy makes it possible to avoid a repeated computation of the nodal correction factors at each outer iteration. Therefore, the use of an implicit time-stepping method pays off in spite of the CFL-like condition to be satisfied by the time step in the case $\theta < 1$. For sufficiently small time steps, the new algorithm is more accurate and/or efficient than the algebraic flux correction schemes proposed in [12],[14]. On the other hand, it is not to be recommended for steady-state computations which call for the use of large time steps. In this case, both the

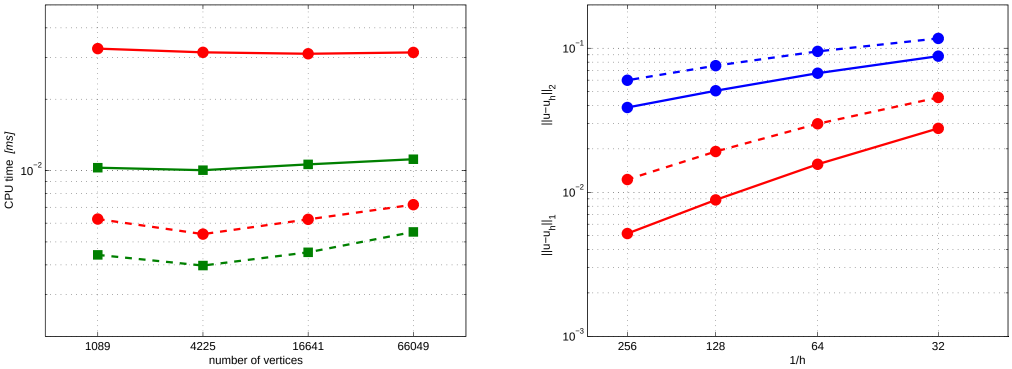


Figure 6. TP4: nodal CPU time / error reduction.

NLEV	NVT	CPU	NN	NN/Δt	NL	NL/NN	$\|u-u_h\|_1$	$\|u-u_h\|_2$
defect correction / consistent mass matrix								
5	1,089	89	24,136	16.09	24,136	1.0	2.7748e-2	8.8019e-2
6	4,225	333	22,694	15.13	23,488	1.03	1.5630e-2	6.7038e-2
7	16,641	1,293	19,883	13.26	21,954	1.10	8.8456e-3	5.0641e-2
8	66,049	5,203	17,959	11.97	22,812	1.27	5.1680e-3	3.8747e-2
defect correction / lumped mass matrix								
5	1,089	17	2,720	1.81	2,720	1.0	4.5446e-2	1.1689e-1
6	4,225	57	2,738	1.83	3,481	1.27	2.9877e-2	9.4992e-2
7	16,641	259	2,804	1.87	3,818	1.36	1.9192e-2	7.5658e-2
8	66,049	1,186	2,953	1.97	4,777	1.62	1.2250e-2	5.9934e-2
Newton's method / consistent mass matrix								
5	1,089	28	5,506	3.67	9,501	1.73	2.7743e-2	8.8007e-2
6	4,225	106	5,241	3.94	9,074	1.73	1.5624e-2	6.7021e-2
7	16,641	442	4,831	3.22	8,342	1.73	8.8374e-3	5.0609e-2
8	66,049	1,844	4,506	3.0	7,802	1.73	5.1604e-3	3.8719e-2
Newton's method / lumped mass matrix								
5	1,089	12	1,510	1.01	1,510	1.0	4.5441e-2	1.16882e-1
6	4,225	42	1,513	1.01	1,513	1.0	2.9868e-2	9.4975e-2
7	16,641	188	1,572	1.05	1,572	1.0	1.9175e-2	7.5623e-2
8	66,049	910	1,803	1.20	1,803	1.0	1.2231e-2	5.9887e-2

Table VI. TP4: semi-implicit FCT, $\|r\| \leq 10^{-8}$, $\eta = 10^{-1}$.

limiting strategy and the underlying constraints need to be redefined as explained in [17].

In order to solve the nonlinear algebraic systems, a discrete Newton approach was devised making use of the fact that the underlying sparsity pattern is known *a priori*. The Jacobian matrix was assembled edge-by-edge using numerical differentiation as applied to the low-order operator and to the vector of limited antidiffusive fluxes. The use of a new semi-implicit limiting strategy makes it possible to assemble the Jacobian matrix in a particularly efficient way, which results in a significant reduction (by a factor of $2.5 - 3.5$) of the total CPU time as compared to standard defect correction. The semi-explicit FCT algorithm was found to provide a slightly better accuracy for the test cases considered in the present paper. However, the high-order system to be solved at the predictor step is extremely ill-conditioned, which calls for the use of a slowly converging defect correction scheme preconditioned by the low-order operator.

REFERENCES

1. Boris JP, Book DL. Flux-corrected transport. I. SHASTA, A fluid transport algorithm that works. *Journal of Computational Physics* 1973; **11**: 38–69.
2. Book DL. The conception, gestation, birth, and infancy of FCT. In *Flux-Corrected Transport: Principles, Algorithms, and Applications*, Kuzmin D, Löhner R, Turek S (eds). Springer, 2005; 5-28.
3. Cai XC, Gropp WD, Keyes DE, Tidriri MD. Newton-Krylov-Schwarz methods in CFD. In *Proceedings of the International Workshop on the Navier-Stokes Equations, Notes in Numerical Fluid Mechanics*, Rannacher R (eds). Vieweg, 1994.
4. Dembo RS, Eisenstat SC, Steihaug T. Inexact Newton methods. *SIAM J. Numer. Anal.* 1982; **19**:400-408.
5. Donea J, Quartapelle L, Selmin V. An analysis of time discretization in the finite element solution of hyperbolic problems. *Journal of Computational Physics* 1987; **70**:463–499.
6. Eisenstat SC, Walker HF. Choosing the forcing term in an inexact Newton method. *SIAM J. Sci. Comput.* 1996; **17**:16-32.
7. Jameson A. Computational algorithms for aerodynamic analysis and design. *Applied Numerical Mathematics* 1993; **13**:383-422.
8. Jameson A. Positive schemes and shock modelling for compressible flows. *International Journal for Numerical Methods in Fluids* 1995; **20**:743–776.
9. Jongen T, Marx YP. Design of an unconditionally stable, positive scheme for the $K-\varepsilon$ and two-layer turbulence models. *Computers and Fluids* 1997; **26**(5):469-487.
10. Kelley CT. Iterative Methods for Linear and Nonlinear Equations. SIAM, Philadelphia, 1995.
11. Kuzmin D. Positive finite element schemes based on the flux-corrected transport procedure. In: *Computational Fluid and Solid Mechanics*, Bathe KJ (ed). Elsevier, 2001; 887-888.
12. Kuzmin D, Turek S. Flux correction tools for finite elements. *Journal of Computational Physics* 2002; **175**:525-558.
13. Kuzmin D, Möller M, Turek S. Multidimensional FEM-FCT schemes for arbitrary time-stepping. *International Journal for Numerical Methods in Fluids* 2003; **42**:265-295.
14. Kuzmin D, Möller M, Turek S. High-resolution FEM-FCT schemes for multidimensional conservation laws. *Computer Methods in Applied Mechanics and Engineering* 2004; **193**:4915-4946.
15. Kuzmin D, Turek S. High-resolution FEM-TVD schemes based on a fully multidimensional flux limiter. *Journal of Computational Physics* 2004; **198**:131-158.
16. Kuzmin D, Möller M. Algebraic flux correction I. Scalar conservation laws. In *Flux-Corrected Transport: Principles, Algorithms, and Applications*, Kuzmin D, Löhner R, Turek S (eds). Springer, 2005; 155-206.
17. Kuzmin D. On the design of general-purpose flux limiters for implicit FEM with a consistent mass matrix. *J. Comput. Phys.* 2006; **219**:513-531.
18. Kuzmin D., Kourounis, D. A semi-implicit FEM-FCT algorithm for efficient treatment of time-dependent problems. Technical report **302**, Institute of Applied Mathematics, University of Dortmund, 2005. Available at `http://www.mathematik.uni-dortmund.de/lsiii/static/schriften_ger.html`
19. LeVeque RJ. High-resolution conservative algorithms for advection in incompressible flow. *Siam Journal of Numerical Analysis* 1996; **33**:627–665.
20. Löhner R, Morgan K, Peraire J, Vahdati M. Finite element flux-corrected transport (FEM-FCT) for the Euler and Navier-Stokes equations. *International Journal for Numerical Methods in Fluids* (1987 **7**:1093–1109.

21. Löhner R, Baum JD. 30 Years of FCT: Status and Directions. In *Flux-Corrected Transport: Principles, Algorithms, and Applications*, Kuzmin D, Löhner R, Turek S (eds). Springer, 2005; 251-296.
22. Meijerink JA, van der Vorst HA. Iterative solution method for linear systems of which the coefficient matrix is a symmetric M-matrix. *Math. Comput.* 1977; **31**:148-162.
23. Möller M. Efficient solution techniques for implicit finite element schemes with flux limiters. *Technical report* **328**. University of Dortmund, 2006. Submitted to *International Journal for Numerical Methods in Fluids* Available at `http://www.mathematik.uni-dortmund.de/lsiii/static/schriften_ger.html`
24. Pernice M, Walker HF. NITSOL: A Newton iterative solver for nonlinear systems. *SIAM J. Sci. Comput.* 1998; **19**:302–318
25. Turek S. *Efficient Solvers for Incompressible Flow Problems: An Algorithmic and Computational Approach.* Springer, 1999.
26. Davis TA. Algorithm 832: UMFPACK V4.3—an unsymmetric-pattern multifrontal method. *ACM Trans. Math. Softw.* 2004; **30**:196–199.
27. Verstappen RWCP, Veldman AEP. Preserving symmetry in convection-diffusion schemes. In: *Turbulent flow computation*, Drikakis D, Geurts J(eds). *Fluid Mechanics and its Applications* **66**. Kluwer:Dordrecht, 2002; 75-100.
28. Verstappen RWCP, Veldman AEP. Symmetry-preserving discretization of turbulent flow. *Journal of Computational Physics* 2003; **187**:343-368.
29. Zalesak ST. Fully multidimensional flux-corrected transport algorithms for fluids. *Journal of Computational Physics* 1979; **31**:335–362.
30. Zalesak ST. The design of Flux-Corrected Transport (FCT) algorithms for structured grids. In *Flux-Corrected Transport: Principles, Algorithms, and Applications*, Kuzmin D, Löhner R, Turek S (eds). Springer, 2005; 29-78.