INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN FLUIDS
Int. J. Numer. Meth. Fluids 2007; **55**:611–635
Published online 20 March 2007 in Wiley InterScience (www.interscience.wiley.com). DOI: 10.1002/fld.1470

Efficient solution techniques for implicit finite element schemes with flux limiters

M. Möller*, †

Institute of Applied Mathematics (*LS III*), *University of Dortmund*, *Vogelpothsweg 87*, *Dortmund D-44227*, *Germany*

SUMMARY

The *algebraic flux correction* (AFC) paradigm is equipped with efficient solution strategies for implicit time-stepping schemes. It is shown that Newton-like techniques can be applied to the nonlinear systems of equations resulting from the application of high-resolution flux limiting schemes. To this end, the Jacobian matrix is approximated by means of first- or second-order finite differences. The edge-based formulation of AFC schemes can be exploited to devise an efficient assembly procedure for the Jacobian. Each matrix entry is constructed from a differential and an average contribution edge by edge. The perturbation of solution values affects the nodal correction factors at neighbouring vertices so that the stencil for each individual node needs to be extended. Two alternative strategies for constructing the corresponding sparsity pattern of the resulting Jacobian are proposed. For nonlinear governing equations, the contribution to the Newton matrix which is associated with the discrete transport operator is approximated by means of divided differences and assembled edge by edge. Numerical examples for both linear and nonlinear benchmark problems are presented to illustrate the superiority of Newton methods as compared to the standard defect correction approach. Copyright © 2007 John Wiley & Sons, Ltd.

Received 29 August 2006; Revised 31 January 2007; Accepted 1 February 2007

KEY WORDS: Newton-like solution techniques; sparse Jacobian evaluation; high-resolution schemes; flux-corrected transport; finite elements

1. INTRODUCTION

For decades, the development of reliable discretization techniques for convection-dominated flows has been one of the primary interests in computational fluid dynamics. A variety of stabilization techniques and high-resolution schemes based on flux/slope limiting have been presented in the literature to combat the formation of non-physical oscillations which would be generated otherwise.

*Correspondence to: M. Möller, Institute of Applied Mathematics (LS III), University of Dortmund, Vogelpothsweg 87, Dortmund D-44227, Germany.
†E-mail: matthias.moeller@math.uni-dortmund.de

As a matter of fact, a serious disadvantage of many existing numerical schemes is their lack of generality. The foundations of modern high-resolution schemes were developed in the finite difference framework using essentially one-dimensional concepts and, typically, geometric criteria. As a consequence, most algorithms popular today are only suitable for Cartesian meshes and/or explicit time stepping.

The origins of modern high-resolution schemes can be traced back to the renowned SHASTA scheme by Boris and Book [1] who set up the *flux-corrected transport* (FCT) methodology in the realm of finite differences. The fully multidimensional generalization proposed by Zalesak [2] has put FCT algorithms into a general framework: the solution is computed by blending linear approximations of high and low order so as to prevent the formation of wiggles. This reformulation has paved the way for the generalization of FCT concepts to explicit Galerkin schemes based on (bi-)linear finite element discretizations on unstructured meshes [3, 4]. As an alternative, *total variation diminishing* (TVD) schemes have been introduced in the context of finite differences [5, 6] and extended to explicit finite element/volume schemes [7, 8].

Classical FCT schemes are based on an explicit correction of the auxiliary low-order solution whose local extrema serve as upper/lower bounds for the sum of limited antidiffusive fluxes. Due to the explicit nature, the time step must satisfy a restrictive 'CFL' condition which may increase the computational costs especially for steady-state problems. Furthermore, the use of stable discretizations, e.g. Taylor–Galerkin, is mandatory for the overall success of the flux limiter. In particular, the use of an unstable high-order method may give rise to nonlinear instabilities which manifest themselves in distortions of the 'corrected' solution profiles.

These restrictions have led to the development of a generalized FEM-FCT methodology introduced by Kuzmin and Turek [9] and refined by Kuzmin *et al.* [10–12]. Flux correction of FCT type is readily applicable to Galerkin schemes with a consistent mass matrix. The use of a second-order Crank–Nicolson time discretization suggests itself for the simulation of strongly time-dependent problems. In comparison to their explicit counterparts, semi-implicit schemes considerably relax the 'CFL' condition and allow for employing moderate time steps. Moreover, unconditionally stable implicit methods can be operated at arbitrarily large time steps (unless iterative solvers fail to converge or the positivity criterion is violated) which makes them a favourable tool for the efficient treatment of steady-state problems.

For flux correction of FCT type, the amount of admissible antidiffusion is inversely proportional to the time step, which compromises the advantages of unconditionally stable implicit schemes. On the other hand, flux limiting schemes of TVD type are independent of the time step, and hence, they represent a good candidate for the treatment of stationary problems. Standard TVD limiters can be integrated into unstructured grid codes and applied edge by edge [7, 8] or node by node [13, 14], so as to control the slope ratio for a local one-dimensional stencil or the net antidiffusion, respectively. Recently, a subtle consolidation of FCT and TVD paradigms has been proposed by Kuzmin [15], who presented a *general purpose* limiter that can be applied to transient and steady-state problems alike. In essence, the new algorithm represents the successful marriage of a symmetric flux limiter for the contribution of the consistent mass matrix, which has to be sacrificed in classical TVD schemes, and an upwind-biased one for monitoring the antidiffusive contribution of convective fluxes.

In any case, the price to be paid for the great flexibility provided by (semi-)implicit high-resolution flux correction schemes is a nonlinear algebraic system that has to be solved in each (pseudo-)time step even if the problem at hand is linear. Due to the fact that the nonlinearity originates from the discretization, it has no continuous counterpart that could be differentiated

DOI: 10.1002/fld

analytically and supplied to Newton-type methods. Since flux limiters typically make use of non-differentiable functions, the 'Jacobian' is approximated by means of divided differences. In the present paper, an efficient assembly algorithm for the Newton matrix is devised by exploiting the edge-based formulation of algebraic flux correction schemes. In general, the sparsity pattern of the global stiffness matrix needs to be extended by one 'connectivity layer', that is, the kth column of the Newton matrix possesses non-zero entries for all degrees of freedom which are adjacent to node k and the neighbours thereof. Moreover, the approximate Jacobian for a nonlinear transport operator can be decomposed into edge contributions so as to allow for an efficient edge-by-edge assembly. Numerical examples are presented for linear and nonlinear two-dimensional stationary benchmark problems to demonstrate the benefits of Newton-type methods as compared to standard defect correction approaches. For the treatment of transient convection problems by means of implicit FEM-FCT algorithms equipped with an algebraic Newton strategy, the interested reader is referred to the recent publication [16].

2. ALGEBRAIC FLUX CORRECTION

In this paper, we adopt an algebraic approach to the design of high-resolution schemes which consists of imposing mathematical constraints on discrete operators so as to achieve certain matrix properties. A detailed description of this so-called *algebraic flux correction* (AFC) paradigm can be found in [9–15, 17]. As a model problem, consider a stationary conservation law for a scalar quantity u whereby $\mathbf{f}$ is a generic and possibly nonlinear flux function

$$\nabla \cdot \mathbf{f}(u) = 0 \quad \text{in } \Omega \tag{1}$$

For the time being, let us assume that $\mathbf{f}$ is composed by convective and diffusive fluxes, i.e. $\mathbf{f}(u) = \mathbf{v}u - d\nabla u$, where $\mathbf{v} = \mathbf{v}(\mathbf{x}, t)$ denotes a non-uniform velocity field and d is the physical diffusion coefficient. The above problem statement is completed by the prescription of concomitant boundary conditions of Dirichlet and/or Neumann type. Let the equation at hand be discretized by a high-order finite element (Galerkin) method and apply algebraic flux correction to turn it into a high-resolution approximation. Even in case the conservation law (1) is linear, this yields a *nonlinear* algebraic equation system for the vector of nodal values

$$K^*(u)u = 0 \tag{2}$$

where the modified transport operator exhibits the following structure [13]:

$$K^*(u) = L + F(u) = K + D + F(u) \tag{3}$$

Here, $K = \{k_{ij}\}$ denotes the original transport operator resulting from the Galerkin finite element approximation of convective terms. The artificial diffusion operator $D = \{d_{ij}\}$ is designed to eliminate all negative off-diagonal coefficients from the high-order operator in order to turn K into its *local extremum diminishing* (LED) counterpart $L = K + D$. The error induced by this so-called 'discrete upwinding' technique [9] is compensated by applying an antidiffusive correction term $F(u)$ which will be addressed in more detail below.

Due to the fact that D is a discrete diffusion operator defined as a symmetric matrix with zero row and column sums, the term Du can be decomposed into a sum of skew-symmetric internodal

DOI: 10.1002/fld

fluxes which are associated with the *edges* of the sparsity graph [13]

$$(Du)_i := -\sum_{j\neq i} f_{ij}, \quad f_{ij} = d_{ij}(u_i - u_j) = -f_{ji} \tag{4}$$

Here and below, the stencil $\mathscr{S}_i$ of node i is defined as the set of indices j for which the finite element basis functions φ_i and φ_j have overlapping supports. As a consequence, there exists an edge **ij** in the global stiffness matrix iff $j \in \mathscr{S}_i$ and $j \neq i$. For piecewise linear finite elements their number equals the number of physical mesh edges, whereas multilinear and high-order FEM approximations allow for interactions of all nodes sharing the same element.

In any case, a natural choice for the artificial diffusion coefficient for the edge **ij** is [9]

$$d_{ij} = \max\{-k_{ij}, 0, -k_{ji}\} = d_{ji} \tag{5}$$

As a result, the off-diagonal coefficients of the low-order operator $l_{ij} := k_{ij} + d_{ij} \geqslant 0$ are non-negative which is a prerequisite for our scheme to possess the LED property [18, 19]. Due to the fact that the operator D exhibits zero row sums, the diagonal entries of L are given by

$$l_{ii} := k_{ii} - \sum_{j\neq i} d_{ij} \tag{6}$$

For our purpose, it is expedient to introduce the following convention: without loss of generality, let the edge **ij** be oriented so that $l_{ji} \geqslant l_{ij} = \max\{0, k_{ij}\}$, which implies that node i is located 'upwind' and corresponds to the row number of the eliminated negative coefficient [13].

The skew-symmetric raw antidiffusive flux f_{ij} from node j into its upwind neighbour i which offsets the error induced by our discrete upwinding is defined in (4). In order to prevent the formation of non-physical local extrema, it is multiplied by a suitable correction factor $0 \leqslant \alpha_{ij} \leqslant 1$ which is determined by means of a multidimensional flux limiter (see below). As a result, the net antidiffusion which is applied to the upwind node i can be expressed as follows:

$$(Fu)_i = \sum_{j\neq i} f_{ij}^*, \quad f_{ij}^* := \alpha_{ij} f_{ij} \tag{7}$$

By definition, the downwind node j receives the flux $f_{ji}^* := -f_{ij}^*$ which is of the same magnitude but exhibits the opposite sign so that there is no net loss or gain of mass.

Putting it all together, the contribution of the modified transport operator $K^*(u)$ applied to the vector of nodal unknowns u can be expressed for each node i as follows:

$$(K^*u)_i = \sum_{j\neq i} k_{ij}^*(u_j - u_i) + u_i \sum_j k_{ij}^* \tag{8}$$

In the above equation, the solution-dependent matrix coefficients are given by

$$k_{ij}^* = k_{ij} + (1 - \alpha_{ij})d_{ij}, \quad k_{ii}^* = k_{ii} - \sum_{j\neq i} (1 - \alpha_{ij})d_{ij} \tag{9}$$

whereby the reactive term in Equation (8) represents a discrete counterpart of $-u\nabla\cdot\mathbf{v}$. It vanishes for divergence-free velocity fields and is responsible for a physical growth of local extrema otherwise. Due to the zero row sum property of the operator D, it is not affected by discrete upwinding and algebraic flux correction.

At the end of the day, the modified transport operator $K^*(u)$ defined in (3) represents a nonlinear combination of the low-order scheme ($\alpha_{ij} \equiv 0$) and the original high-order one ($\alpha_{ij} \equiv 1$). The task

DOI: 10.1002/fld

of the flux limiter is to determine an optimal correction factor α_{ij} so as to remove as much artificial diffusion as possible without generating spurious oscillations.

The idea of node-based flux limiting can be traced back to the multidimensional FCT limiter proposed by Zalesak [2] and has been adopted in the AFC framework [9–15, 17]. In short, antidiffusive fluxes $f_{ij} = p_{ij}(u_j - u_i)$ which are proportional to solution differences multiplied by coefficients $p_{ij} \leqslant 0$ violate the LED criterion introduced in [18, 19], and hence, need to be limited. On the other hand, edge contributions with non-negative coefficients resemble diffusive fluxes and are harmless. Some portion of antidiffusion, say, from node j into node i is only admissible if it can be interpreted as a diffusive flux from another node, that is, if there exists a solution-dependent coefficient $q_{ik} \geqslant 0$ such that $f_{ij} = q_{ik}(u_k - u_i)$.

A general framework for flux correction in multidimensions is presented in a recent publication by Kuzmin [15]. Here, we will only address *upwind-biased* flux limiters which are appropriate for the treatment of stationary problems. For the design of symmetric flux correction schemes which are more convenient for the treatment of time-dependent flows, the interested reader is referred to [10, 15] and the references therein.

To begin with, let us 'prelimit' the raw antidiffusive flux (4) so as to obtain

$$f'_{ij} = \min\{d_{ij}, l_{ji}\}(u_i - u_j) \tag{10}$$

It is worth mentioning that the above flux reduces to f_{ij}, unless both off-diagonal entries of the high-order operator K are negative (a rather unusual situation).

For each node, the net antidiffusion may consist of both positive and negative edge contributions, but in the worst case, all fluxes have the same sign. Hence, it is worth treating the positive and negative ones separately, as proposed by Zalesak [2]. The total amount of raw antidiffusion received by node i from its downwind neighbours is given by

$$P_i^{\pm} = \sum_{j \in \mathscr{J}_i} \begin{matrix}\max \\ \min\end{matrix} \{0, f'_{ij}\} \quad \text{where } \mathscr{J}_i = \{j \neq i : l_{ji} > l_{ij} = 0\} \tag{11}$$

The upper/lower bounds to be imposed by the flux limiter can be computed from the off-diagonal coefficients of the low-order operator L which are non-negative by construction

$$Q_i^{\pm} = \sum_{j \neq i} l_{ij} \begin{matrix}\max \\ \min\end{matrix} \{0, u_j - u_i\}, \quad l_{ij} \geqslant 0 \ \ \forall j \neq i \tag{12}$$

For each node, the admissible antidiffusion is given by the nodal correction factors

$$R_i^+ = \min\{1, Q_i^+ / P_i^+\}, \quad R_i^- = \min\{1, Q_i^- / P_i^-\} \tag{13}$$

They are designed so as to enforce the LED constraint $|R_i^{\pm} P_i^{\pm}| \leqslant |Q_i^{\pm}|$ for the upwind node. Consequently, the final correction factor α_{ij} is taken as the nodal multiplier of node i

$$\alpha_{ij} = \begin{cases} R_i^+ & \text{if } f'_{ij} > 0 \\ R_i^- & \text{otherwise} \end{cases} \tag{14}$$

so that the (prelimited) antidiffusive flux (10) can be corrected according to $f_{ij}^* = \alpha_{ij} f'_{ij}$. For a detailed description of flux limiting in multidimensions, including rigorous positivity proofs, the interested reader is referred to the publications [10, 13, 15] and the references therein.

DOI: 10.1002/fld

3. NONLINEAR SOLUTION STRATEGIES

A common practice in the computation of steady-state solutions of partial differential equations is to march the pseudo-transient counterpart of conservation law (1) to the stationary limit

$$\frac{\partial u}{\partial \tau} + \nabla \cdot \mathbf{f}(u) = 0 \tag{15}$$

where τ denotes the artificial time variable. Let us adopt the algebraic flux correction approach to approximate the spatial derivatives and use the fully implicit, unconditionally stable backward-Euler scheme for the discretization in (pseudo-)time

$$M_{\mathrm{L}} \frac{u^{n+1} - u^n}{\Delta \tau} = K^*(u^{n+1}) u^{n+1} \tag{16}$$

Here, $M_{\mathrm{L}} = \{m_i\}$ denotes the lumped mass matrix counterpart of the consistent mass matrix which comes from the Galerkin discretization of the time derivative. Due to the fact that the backward-Euler method is unconditionally positivity-preserving [9], it can be operated at arbitrarily large steps $\Delta\tau$. Interestingly enough, algebraic flux correction methods of TVD type [13, 14] are derived on the semi-discrete level, and hence, they are independent of the employed (pseudo-)time step size. As a consequence, the converged steady-state solution to problem (16) can be computed very efficiently if $\Delta\tau \to \infty$ for $n \to \infty$ without loss of accuracy.

It is noteworthy that the implicit Euler method (16) leads to algebraic equations which are very similar to those resulting from the use of under-relaxation applied to steady-state flow problems [20, pp. 148–149]. If the same time step is adopted for all equations, this corresponds to taking a variable under-relaxation factor for each nodal equation. Conversely, the use of a constant under-relaxation factor is equivalent to adopting a different time step for the computation of each nodal solution value u_i^{n+1} by means of pseudo-time-stepping.

3.1. Fixed-point iteration

As a matter of fact, Equation (16) exhibits some nonlinearity due to flux correction which calls for an iterative solution strategy even if the governing equation is linear. To make the presentation self-contained, let us recapitulate nonlinear solution techniques in a more general framework and apply them to the residual form of the algebraic system (16) afterwards

$$g^{n+1} := [M_{\mathrm{L}} - \Delta\tau K^*(u^{n+1})]u^{n+1} - M_{\mathrm{L}} u^n = 0 \tag{17}$$

Given the global vector of unknowns u which may be either the solution from the last pseudo-time step ($u = u^n$) or an initial guess ($u = u_0$), the end-of-step solution u^{n+1} at 'time' $\tau^{n+1} = \tau^n + \Delta\tau$ can be computed by the (possibly relaxed) fixed-point iteration

$$u^{(m+1)} = u^{(m)} - \omega^{(m)} [C^{(m)}]^{-1} g^{(m)}, \quad u^{(0)} = u, \quad m = 0, 1, 2, \ldots \tag{18}$$

Here, $\omega^{(m)}$ denotes the damping parameter of the mth cycle and $C^{(m)}$ is a suitable 'preconditioner' to be defined below. The iteration process is terminated based on a required drop in the norm of the residual and/or a sufficiently small solution increment

$$\|g^{(m+1)}\| \leqslant \varepsilon_1 \|g^{(0)}\|, \quad \|\Delta u^{(m+1)}\| \leqslant \varepsilon_2 \|u^{(m)}\|$$

DOI: 10.1002/fld

Here, ε_1 and ε_2 are user-defined parameters, $\|\cdot\|$ denotes an arbitrary norm and the term $\Delta u^{(m+1)} = u^{(m+1)} - u^{(m)}$ refers to the solution increment to be computed.

Note that, monitoring only the relative changes may be insufficient since small values of the relaxed solution increment may also be caused by strong under-relaxation $|\omega^{(m+1)}| \ll 1$ which leads to extremely slow convergence. Moreover, theoretically motivated stopping criteria can be designed as discussed in [21, 22] making use of the fact that the iteration procedure results from a finite element approximation of a partial differential equation.

In a practical implementation, the 'inversion' of the preconditioner matrix $C^{(m)}$ is also performed by a suitable iteration procedure for solving the *linear* subproblem

$$C^{(m)} \Delta u^{(m+1)} = -g^{(m)}, \quad m = 0, 1, 2, \ldots \tag{19}$$

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

$$u^{(m+1)} = u^{(m)} + \omega^{(m+1)} \Delta u^{(m+1)}, \quad u^{(0)} = u \tag{20}$$

It remains to specify a suitable preconditioner. A multivariate Taylor expansion of the residual term $g^{(m+1)}$ about the current state $u^{(m)}$ yields the following approximation:

$$g^{(m+1)} \simeq g^{(m)} + J^{(m)}(u^{(m+1)} - u^{(m)}) \tag{21}$$

which requires the evaluation of the Jacobian matrix at the last iterate $u^{(m)}$, that is

$$J^{(m)} = \{J_{ij}^{(m)}\} \quad \text{where } J_{ij}^{(m)} = \left.\frac{\partial g_i(u)}{\partial u_j}\right|_{u = u^{(m)}} \tag{22}$$

Neglecting terms of higher-order curvature in the linearized model (21) and recalling the postulated relation $g^{n+1} = 0$, one ends up with the well-known *Newton method*

$$u^{(m+1)} = u^{(m)} - [J^{(m)}]^{-1} g^{(m)} \tag{23}$$

It can be readily derived from the iteration scheme (18) by setting $C^{(m)} = J^{(m)}$.

Another attractive algorithm for the solution of nonlinear equations that turns (18) into a fixed-point defect correction procedure [23] is based on the 'monotone' low-order operator

$$C^{(m)} = M_{\mathrm{L}} - \Delta\tau L^{(m)} \tag{24}$$

which was designed to be an *M-matrix* and hence exhibits favourable matrix properties [12, 14].

It is worth mentioning that intermediate solutions $u^{(m)}$ are not required to be positive so that convergence of algorithm (18) is a prerequisite for positivity. The use of an iterative solver applicable to large sparse, non-symmetric systems of linear equations is mandatory. 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 low-order operator (24) unconditionally exists and is unique due to the M-matrix property [24]. Hence, it is advisable to use it as preconditioner for the Krylov subspace method even if the Jacobian matrix (22) is adopted in the outer iteration procedure (18).

It is well known that the performance of Newton's method strongly depends on the quality of the initial guess u. For the solution of the steady Euler equations, Hemker and Koren [25] suggest a two-step defect correction approach. A provisional first-order solution for the stationary problem is

DOI: 10.1002/fld

computed directly, that is, without resorting to pseudo-time stepping. Next, the low-order profile is used as initial guess for a second-order accurate defect correction iteration, whereby the first-order operator serves as a preconditioner.

Within a Newton iteration approach, this idea may be adopted as follows. As before, let u denote the initial solution for the current iteration step, i.e. $u = u^n$ for time-dependent problems or $u = u_0$ in the stationary case. In order to obtain a usable initial guess for the Newton iteration, perform a small number of 'presmoothing' steps. To this end, the low-order operator (24) can be applied either *per se* (without resorting to algebraic flux correction) or as a preconditioner within a high-resolution flux/defect correction scheme. After a few iterations, the full Jacobian (22) is used so that the iteration procedure (18) yields Newton's algorithm.

3.2. Globalization

Since the linear system (19) is solved iteratively, and hence, the computation of the 'exact' solution of the Newton equation is quite costly, the resulting algorithm is categorized as an *inexact Newton method* [26]. Obviously, the accuracy of the (inner) linear solver greatly affects the convergence behaviour of the (outer) nonlinear Newton algorithm. If the linear subproblems are not solved accurately enough more Newton steps are required, and hence, the nonlinear convergence rate deteriorates. Conversely, a very small tolerance for the linear solver results in a drastic increase of inner iterations which does not pay off. Moreover, if $u^{(m)}$ is not sufficiently close to the desired root then the linearization by means of the Taylor series expansion (21) may not reflect the behaviour of the nonlinear residual (17) very well. As a consequence, solving the linear system (19) for the increment $\Delta u^{(m+1)}$ with high accuracy one may obtain a poor Newton update which deteriorates the nonlinear convergence behaviour [27, 28]. This phenomenon is typically known as *oversolving* [29].

A common practice is to choose the so-called forcing term $\eta^{(m)} \in [0, 1)$ *a priori* and require the linear solver to compute the increment $\Delta u^{(m+1)}$ from the Newton equation (19) to a certain accuracy so that the following convergence criterion holds:

$$\|g^{(m)} + J^{(m)} \Delta u^{(m+1)}\| \leqslant \eta^{(m)} \|g^{(m)}\| \tag{25}$$

Recall that the left-hand side of the above inequality is both the residual of the linear subproblem and the linearized model of $g^{(m+1)}$ given by the first-order terms of the Taylor series expansion (21). Several strategies for choosing the forcing term in an adaptive fashion are proposed by Eisenstat and Walker [29, 30]. A viable choice is to employ

$$\eta^{(m+1)} = \gamma(\|g^{(m+1)}\|/\|g^{(m)}\|)^{\alpha}, \quad \eta^{(0)} \in [0, 1) \tag{26}$$

where the auxiliary coefficients $\gamma \in [0, 1]$ and $\alpha \in (1, 2]$. In our simulations, we adopted $\gamma = 0.5$ and $\alpha = (1 + \sqrt{5})/2$ as proposed in [30]. A *local* convergence theory of inexact Newton methods and a detailed discussion about the impact of forcing terms is given in [26].

It is well known that this technique is prone to diverge for crude starting values. Due to this lack of convergence robustness, the use of some 'globalization' technique is mandatory. To this end, the computed solution increment needs to be relaxed by the factor $\omega^{(m+1)}$ so that the *sufficient decrease condition* holds on each Newton step [30]

$$\|g(u^{(m)} + \omega^{(m+1)} \Delta u^{(m+1)})\| \leqslant [1 - \xi(1 - \eta^{(m)})] \|g^{(m)}\| \tag{27}$$

DOI: 10.1002/fld

where $\xi \in (0, 1)$ represents the prescribed reduction tolerance. In practice, line search and trust region methods are frequently employed to compute an acceptable increment [31, 32]. In our implementation, we make use of a simple backtracking strategy [27] which computes $\omega^{(m+1)}$ as the minimizer of $\frac{1}{2}\|g(u^{(m)} + \omega^{(m+1)}\Delta u^{(m+1)})\|^2$ by means of quadratic interpolation.

4. CALCULATION OF JACOBIANS

Now that we have discussed solution strategies for nonlinear problems in a general framework, let us consider the calculation of the Jacobian matrix so as to build up the preconditioner matrix $C^{(m)} = J^{(m)}$ for the fixed-point iteration (18). For simplicity, let us omit the superscript m in what follows. The formal definition (22) requires the 'differentiation' of the modified transport operator $K^*(u)$ which consists of diffusive and antidiffusive contributions (9). Recall the fact that flux limiters frequently make use of functions which lack the global differentiability [10, 13, 15] so that no analytical expression for the Jacobian is available.

The use of an iterative method, such as BiCGSTAB or GMRES, for the solution of the linear system (19) only requires the computation of Jacobian-vector products which may be approximated by means of first-order (forward/backward) divided differences

$$Jv \simeq \pm\frac{g(u \pm \sigma v) - g(u)}{\sigma} \tag{28}$$

Alternatively, the product Jv can be approximated by the second-order central difference

$$Jv \simeq \frac{g(u + \sigma v) - g(u - \sigma v)}{2\sigma} \tag{29}$$

which requires a double evaluation of the nonlinear residual. In practice, the performance of Newton's method is quite sensitive [33] to the size of the perturbation parameter σ which should be sufficiently small to obtain a good approximation to the derivative. Following a strategy proposed by Nielsen *et al.* [34], the step size can be determined using the expression

$$\sigma\|v\| = \sqrt{\varepsilon} \tag{30}$$

where ε denotes the machine precision. Some alternative choices are given in a survey paper on Jacobian-free Newton–Krylov methods by Knoll and Keyes [35].

What makes such *matrix-free* approaches most attractive at first glance is their Newton-like nonlinear convergence behaviour without the costs of computing and storing the Jacobian explicitly. However, these advantages are not as overwhelming as one might immediately think. For finite element problems, the Jacobian matrix is typically very sparse and hence the savings in terms of memory usage are insignificant. The crucial point is that there are only few preconditioners which can be applied without knowing the system matrix explicitly [36]. As a consequence, the iteration process may converge poorly or even fail to converge at all. If approximate preconditioners are employed, the use of sophisticated flexible GMRES [37] or GMRES-R Krylov subspace methods [38] is mandatory which call for some extra dense vector storage. Finally, the costs of performing flux correction following the algorithmic steps (10)–(14) in each iteration of the linear solver rapidly grow to an impractical amount.

In light of the above, the explicit formation of Jacobians gains more attraction, provided a sufficiently accurate approximation can be computed at reasonable costs. For our purpose,

DOI: 10.1002/fld

it makes sense to introduce the divided difference operator $\mathscr{D}_k$ for a generic function $f(u)$ so that each entry of the Jacobian can be approximated with second-order accuracy

$$\mathscr{D}_k[f] := \frac{f(u+\sigma e_k) - f(u-\sigma e_k)}{2\sigma} \quad \Rightarrow \quad J_{ik} = \mathscr{D}_k[g_i] \tag{31}$$

whereby e_k denotes the kth unit vector. The approximation error is proportional to σ^2 which, adopting definition (30) for the perturbation parameter σ, is the same for all columns.

4.1. Discrete transport operators

Algebraic flux correction schemes [9–15, 17] are designed as 'black-box' post-processing tools which extract all required information from the matrix and make use of the solution values to modify the right-hand side and the residual, respectively. With this observation in mind, definition (31) may be used to devise a straightforward algorithm for assembling the Jacobian. In essence, each column of the matrix J can be constructed by taking the difference between the residuals evaluated at the 'perturbed' solutions $u \pm \sigma e_k$ and scaling the result by 2σ. However, this approach is prohibitively expensive since it does not exploit the sparsity pattern of the stiffness matrix. A common practice in finite element methods is to assemble the Jacobian matrix element by element [39]. As we are about to see, the edge-based formulation of our algebraic flux correction techniques can be utilized to construct J edge by edge.

Let us split the Jacobian matrix into its convective part $T^* = \{t^*_{ij}\}$ and the contribution resulting from the pseudo-time discretization so as to obtain

$$J = M_{\mathrm{L}} - \Delta\tau T^* + \mathscr{O}(\sigma^2), \quad t^*_{ij} = \frac{\partial (K^*(u)u)_i}{\partial u_j} \tag{32}$$

Recall that the modified evolution operator $K^*(u)$ given by Equation (3) represents a nonlinear combination of the original high-order operator K and its low-order counterpart $L = K + D$ which can be recovered by varying α_{ij} between zero and unity. Therefore, it suffices to devise an efficient algorithm for evaluating the convective operator T^* whose entries can be approximated by divided differences $t^*_{ik} = \mathscr{D}_k[K^*(u)u]_i$ with second-order accuracy. Let us replace both matrix–vector products by the decomposition (8) so as to obtain

$$t^*_{ik} = \sum_{j\neq i} \frac{k^*_{ij}(u+\sigma e_k)}{2\sigma}(u_j + \sigma\delta_{jk} - u_i - \sigma\delta_{ik}) + (u_i + \sigma\delta_{ik})\sum_j \frac{k^*_{ij}(u+\sigma e_k)}{2\sigma} \tag{33}$$

$$-\sum_{j\neq i} \frac{k^*_{ij}(u-\sigma e_k)}{2\sigma}(u_j - \sigma\delta_{jk} - u_i + \sigma\delta_{ik}) - (u_i - \sigma\delta_{ik})\sum_j \frac{k^*_{ij}(u-\sigma e_k)}{2\sigma} \tag{34}$$

where δ_{ab} denotes the standard Kronecker delta symbol. In addition to the divided difference operator $\mathscr{D}_k$, let us define the standard average $\mathscr{A}_k$ for a generic function f as follows:

$$\mathscr{A}_k[f] := \frac{f(u+\sigma e_k) + f(u-\sigma e_k)}{2} \tag{35}$$

DOI: 10.1002/fld

As a consequence, the above expression for the coefficient t_{ik}^* can be cast into the form

$$t_{ik}^* = \mathscr{A}_k[k_{ik}^*] + \sum_{j\neq i} \mathscr{D}_k[k_{ij}^*](u_j - u_i) + u_i \sum_j \mathscr{D}_k[k_{ij}^*] \tag{36}$$

According to definition (9), the contributions of the modified transport operator are given by

$$\begin{aligned} \mathscr{A}_k[k_{ij}^*] &= \mathscr{A}_k[k_{ij}] + \mathscr{A}_k[(1-\alpha_{ij})d_{ij}], \\ \mathscr{D}_k[k_{ij}^*] &= \mathscr{D}_k[k_{ij}] + \mathscr{D}_k[(1-\alpha_{ij})d_{ij}], \end{aligned} \quad j \neq i \tag{37}$$

Due to the zero row sum property of discrete diffusion operators, the diffusive contribution cancels out in the last term of (36) so that only the Galerkin part $\mathscr{D}_k[k_{ij}]$ is present in the second sum of t_{ik}^*. Moreover, the average term on the diagonal of T^* is given by

$$\mathscr{A}_k[k_{ii}^*] = \mathscr{A}_k[k_{ii}] - \sum_{j\neq i} \mathscr{A}_k[(1-\alpha_{ij})d_{ij}] \tag{38}$$

Let us consider the special case that flux limiting is deactivated ($\alpha_{ij} \equiv 1$) so that Equation (36) yields the approximate Jacobian contribution $T^H = \{t_{ij}^H\}$ of the original high-order scheme

$$t_{ik}^H = \mathscr{A}_k[k_{ik}] + \sum_j \mathscr{D}_k[k_{ij}]u_j \tag{39}$$

Interestingly enough, this expression represents the divided difference approximation of $K + K'u$, whereby the average term can also be replaced by k_{ik}. Likewise, the approximate derivative of the operator L is recovered from (36) if all coefficients α_{ij} are equal to zero

$$t_{ik}^L = \mathscr{A}_k[l_{ik}] + \sum_j \mathscr{D}_k[l_{ij}]u_j \tag{40}$$

We would like to emphasize the fact that the decomposition into individual contributions (39) and also (40) provides an efficient alternative to the traditional element-by-element procedure [39] which is commonly used to assemble Jacobians arising from finite element discretizations. It is noteworthy that for typical evolution operators $K(u)$ or $L(u)$, most contributions to the off-diagonal entries are likely to vanish in a practical implementation.

So far, the conservation law (1) is supposed to be nonlinear so that the operator $K^*(u)$ depends on the unknown solution vector u due to both a physical/natural and a numerical nonlinearity. The latter one results from the application of algebraic flux correction and is still present in the case of a linear governing equation. In this situation, the discrete derivative of the high-order transport operator K vanishes, i.e. $\mathscr{D}_k[k_{ij}] = 0$ in relation (37), so that

$$t_{ik}^* = \mathscr{A}_k[k_{ik}^*] + \sum_{j\neq i} \mathscr{D}_k[1-\alpha_{ij}]d_{ij}(u_j - u_i) \tag{41}$$

Note that the artificial diffusion coefficient d_{ij} no longer depends on the solution vector u, and hence, it is not affected by the divided difference approximation. Moreover, the last term in the more general definition (36) vanishes due to the zero row sum property of discrete diffusion operators. With these observations in mind, it is easy to verify that the differentiation only affects the *net correction factors* $(1-\alpha_{ij})$ which are still nonlinear, whereas the derivative of the linear transport operator K and that of the discrete diffusion term D are no longer present in the Jacobian matrix (36). As before, the average term can be replaced by the operator $K^*(u)$.

DOI: 10.1002/fld

4.2. Sparsity pattern

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 K^*. This is due to the fact that perturbation of the nodal solution value u_i affects the diffusion coefficients d_{ij} as well as the corresponding correction factors α_{ij}.

To begin with, consider an unstructured mesh consisting of conforming triangles and/or quadrilaterals as shown in Figure 1 (left). Recall that the stencil of node k is defined as the set of vertices l for which the finite element basis functions have overlapping support. For the time being, the orientation convention introduced in Section 2 is neglected. The sparsity pattern of the global system matrix can be constructed by considering all sets $\mathcal{S}_k$. In Figure 1 (left), the dashed lines point to the column numbers of non-zero entries in the kth row of the finite element matrix. Likewise, for each of these neighbours the corresponding rows exhibit a non-zero entry in the kth column due to the symmetry of the undirected connectivity graph.

From the definition of the stencil $\mathcal{S}_k$, it follows that the perturbation of the solution vector u at vertex k only affects the quantities $P_l^{\pm}$ and $Q_l^{\pm}$ defined in (11)/(12) if $l \in \mathcal{S}_k$. As a consequence, the corresponding multipliers $R_l^{\pm}$ need to be recomputed from formula (13). For all other nodes i which are not comprised in the set $\mathcal{S}_k$, the nodal quantities $P_i^{\pm}$, $Q_i^{\pm}$ and $R_i^{\pm}$ coincide with their unperturbed counterparts which are already known.

Recall that the correction factor α_{ij} is taken as $R_i^{\pm}$ depending on the sign of the antidiffusive flux (14). Obviously, local perturbations of the nodal solution value u_k are quite likely to affect the magnitude of α_{ij} if the upwind node i belongs to $\mathcal{S}_k$, i.e. the set of neighbours directly connected to the perturbed vertex k. In other words, the impact of 'joggling' the solution value u_k may propagate along paths of length two until some node $j \notin \mathcal{S}_k$ is reached for which there exists an edge **ij** such that $i \in \mathcal{S}_k$. This observation suggests the definition of an extended list of neighbouring nodes

$$\tilde{\mathcal{S}}_k = \bigcup_{l \in \mathcal{S}_k} \mathcal{S}_l \tag{42}$$

so as to reflect the new connectivity pattern of the resulting matrix. The structure of non-zero entries in the kth column of the Jacobian is illustrated in Figure 1 (right). Those edges **ij** for which both the diffusion coefficient d_{ij} and the correction factor α_{ij} may exhibit a change in magnitude due to the perturbation of the solution at node k are marked by dashed edges. Moreover, dots are

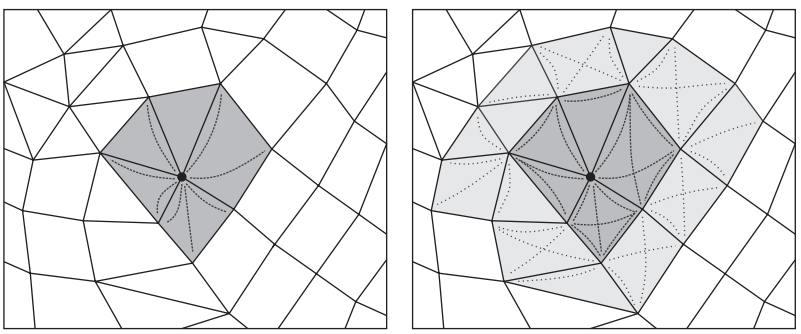

Figure 1. Connectivity graph: mass matrix *vs* Jacobian operator.

DOI: 10.1002/fld

used to indicate those edges which indirectly depend on the perturbed solution value u_k. Even though the corresponding diffusion coefficients d_{ij} coincide with their unperturbed counterparts, the net antidiffusive contributions $\mathscr{D}_k[1-\alpha_{ij}]d_{ij}$ may persist due to different magnitudes of the multipliers α_{ij} which depend on $u \pm \sigma e_k$.

Interestingly enough, the extended sparsity pattern of the approximate Jacobian resembles that of the edge-oriented FEM stabilization technique [40, 41]. A detailed description of the underlying data structure for non-conforming bilinear finite elements is given in [42]. The presented storage algorithm can be carried over to conforming (bi-)linear finite elements but care must be taken to avoid duplicate entries in the rows of the finite element matrix.

Let us briefly review some ideas from classical graph theory and devise an alternative storage algorithm which is directly tailored to conforming (bi-)linear finite elements. In what follows, $A=\{a_{ij}\}$ denotes the adjacency matrix which represents the undirected connectivity graph of the stiffness matrix. The corresponding coefficients are given by

$$a_{ij} \in \{0, 1\} \quad : \quad a_{ii}=1, \quad a_{ij}=1 \;\Leftrightarrow\; \exists \mathbf{ij} \tag{43}$$

that is, a_{ij} does not vanish if nodes i and j share the same element. In other words, there exists a path of length one connecting vertices i and j. Furthermore, let $Z=A^2=A\cdot A$. Then $z_{ij}>0$ if and only if there exists a path of length not longer than two connecting nodes i and j. This can be easily verified by recalling that

$$z_{ij}=\sum_k a_{ik}\,a_{kj}>0 \;\Leftrightarrow\; \exists k : a_{ik}=1 \wedge a_{kj}=1 \tag{44}$$

that is, vertex j can be reached from i and *vice versa via* node k passing two edges. As a result, a standard algorithm [43] for sparse matrix multiplications can be employed to assemble Z which can be used to construct the sparsity pattern of the Jacobian matrix.

5. NUMERICAL EXAMPLES

In order to demonstrate the ideas presented in this paper, we apply our algebraic Newton strategy to a number of two-dimensional benchmark problems and compare the nonlinear convergence behaviour to that of the classical defect correction procedure.

All tests were performed on an Intel Core2 Duo E6600 (2.4 GHz, FSB 1066 MHz) processor with 2048 MB (667 MHz) of system memory. The code was compiled with the Intel Fortran 9.1 Computer for Linux making use of inter-procedural optimization (`-ipo`) for the target platform (`-O3 -xT`). Moreover, optimized BLAS routine where employed to increase performance.

5.1. Stationary convection–diffusion

To begin with, consider the model problem (1) and let the flux function $\mathbf{f}(u):=\mathbf{v}u-d\nabla u$ so as to recover the stationary convection–diffusion equation

$$TP1: \quad \mathbf{v}\cdot\nabla u - d\Delta u = 0 \quad \text{in } \Omega=(0,1)\times(0,1)$$

DOI: 10.1002/fld

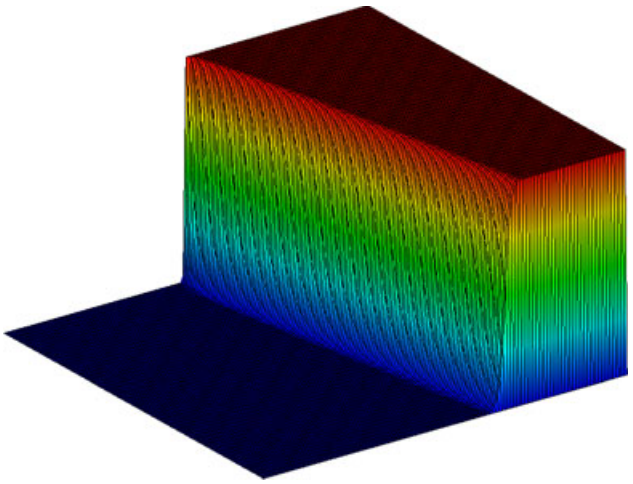

Figure 2. TP1: FEM-TVD, $128 \times 128 Q_1$-elements.

where $\mathbf{v} = (\cos 10°, \sin 10°)$ is the constant velocity and $d = 10^{-6}$ denotes the diffusion coefficient. The concomitant boundary conditions read

$$u(x, 0) = 0, \quad \frac{\partial u}{\partial y}(x, 1) = 0, \quad u(0, y) = \begin{cases} 1 & \text{if } y \geqslant 0.5 \\ 0 & \text{otherwise} \end{cases} \qquad u(1, y) = 0, \tag{45}$$

The solution to this singularly perturbed elliptic problem is characterized by the presence of a sharp front next to the line $x = 1$. The boundary layer develops because the solution of the reduced problem ($d = 0$) does not satisfy the homogeneous Dirichlet boundary conditions.

A reasonable initial guess for the desired stationary solution is given by

$$u_0(x, y) = \begin{cases} 1 - x & \text{if } y \geqslant 0.5 \\ 0 & \text{otherwise} \end{cases} \tag{46}$$

The numerical solution depicted in Figure 2 was computed by the backward-Euler FEM-TVD method on a uniform mesh of 128×128 bilinear elements. The resolution of the thin boundary layer is very crisp and moreover the solution is completely free of spurious oscillations.

Recall that the problem at hand is linear so that the nonlinearity stems from the high-resolution discretization scheme. To analyse the performance of the different nonlinear solution strategies, we applied them to the pseudo-transient equation (15) on four successively refined quadrilateral meshes. Since only the converged steady-state solution was of interest for this benchmark, it was sufficient to perform exactly one nonlinear iteration per time step. One should be aware of the fact that this technique may not retain the full stability of the implicit Euler method so that the size of the largest admissible time step may be bounded from above [20]. Nevertheless, this approach combined with some under-relaxation of the fixed-point iteration (18) allowed for an efficient computation of the final solution.

In general, stagnation of steady-state convergence is an unpleasant byproduct of limiting schemes which frequently make use of non-differentiable functions such as the *max* or the *min* function. Most

DOI: 10.1002/fld

Table I. TP1: FEM-TVD, defect correction.

NLEV	NVT	CPU	NN	NL	NL/NN
$\Delta\tau = 0.1$					
5	1089	<1	227	452	1.99
6	4225	1	346	744	2.15
7	16 641	6	457	553	1.21
8	66 049	62	907	1138	1.25
$\Delta\tau = 1.0$					
5	1089	<1	208	1276	6.13
6	4225	3	353	3107	8.80
7	16 641	20	530	4665	8.80
8	66 049	161	1099	6020	5.48
$\Delta\tau = 10.0$					
5	1089	<1	229	2001	8.73
6	4225	5	398	5990	15.05
7	16 641	63	613	17 069	27.85
8	66 049	1343	1268	68 192	53.78
$\Delta\tau = 100.0$					
5	1089	<1	254	2629	10.35
6	4225	7	455	8645	19.00
7	16 641	96	702	26 873	38.28
8	66 049	2371	1509	131 453	87.11

limiters are designed to avoid oscillations and produce highly accurate solution profiles. However, this concept may lead to severe convergence problems. As a remedy, differentiable limiters [44] can be employed which lead to more diffusive numerical results. For our simulations, we utilized the upwind-biased flux limiting approach presented in Section 2 and relaxed the fixed-point iteration (18) by the under-relaxation factor $\omega = 0.8$.

The simulation was stopped once the nonlinear residual (17) satisfied the following criterion:

$$\|g\| = \sqrt{g^{\mathrm{T}} g} \leqslant 10^{-9} \tag{47}$$

A detailed comparison between the defect correction method and the algebraic Newton approach is presented in Tables I and II. The first three columns display the refinement level NLEV, the number of vertices NVT and the total CPU time required to compute the steady-state solution. In the next three columns, the total number of nonlinear iterations (NN) which equals the number of pseudo-time steps, the total number of linear iterations (NL) and the number of linear iterations per nonlinear iteration (NL/NN) are displayed in successive order.

The results for the standard defect correction approach which corresponds to adopting the monotone low-order operator $C = M_{\mathrm{L}} - \Delta\tau L$ as preconditioner for the fixed-point iteration (18) are presented in Table I. Obviously, the nonlinear convergence behaviour deteriorates significantly if the computational mesh is refined. Taking larger pseudo-time steps goes along with a slight increase in the number of nonlinear iterations. In addition, the task of the linear BiCGSTAB solver which is preconditioned by the ILU decomposition of C becomes much more challenging if $\Delta\tau$

DOI: 10.1002/fld

Table II. TP1: FEM-TVD, algebraic Newton.

NLEV	NVT	CPU	NN	NL	NL/NN
$\Delta\tau = 0.1$					
5	1089	<1	82	82	1.00
6	4225	2	66	250	3.78
7	16641	9	60	386	6.43
8	66049	56	56	1014	18.11
$\Delta\tau = 1.0$					
5	1089	<1	30	154	5.13
6	4225	1	35	271	7.74
7	16641	7	36	463	12.86
8	66049	68	40	1671	41.77
$\Delta\tau = 10.0$					
5	1089	<1	28	182	6.50
6	4225	1	30	393	13.10
7	16641	7	38	520	13.68
8	66049	117	75	2832	37.76
$\Delta\tau = 100.0$					
5	1089	<1	33	207	6.27
6	4225	1	36	402	11.17
7	16641	9	45	589	13.09
8	66049	111	84	2488	29.62

is increased from 0.1 to 100. Speaking in terms of total CPU time, the standard defect correction scheme is not competitive for realistically fine grids.

In contrast, the algebraic Newton approach ($\sigma = \sqrt{\varepsilon}$, η from (26)) being applied to the same benchmark scenario performs quite well (cf. Table II). A moderate number of nonlinear iteration steps suffices to compute the steady-state solution for all configurations. The total CPU time is considerably smaller than that of the defect correction scheme when the computation is performed on realistically fine grids. As before, the linear convergence rates depend on the size of the pseudo-time step which is due to the fact that the condition number of the Jacobian matrix J gets worse for larger $\Delta\tau$. We would like to point out that our algebraic Newton algorithm achieves only superlinear convergence which is the best one can hope for. This may be attributed to the fact that the 'artificial' nonlinearity is engendered by the flux limiter which builds on non-differentiable functions (see algorithm (10)–(14)). Hence, it is advisable to operate the nonlinear solver with conservative settings instead of tuning the perturbation parameter σ and/or the forcing term η for each individual simulation by hand.

The pseudo-transient behaviour of both solution algorithms is displayed in Figure 3 for the 'relaxation parameters' $\Delta\tau = 1$ and 10, respectively. For the defect correction scheme, cf. Figure 3 (left), the number of pseudo-time steps which equal the number of nonlinear iterations (see above) required to obtain the converged solution depends entirely on the mesh width and, to some extent, on the size of the underlying time step. As depicted in the right diagrams of Figure 3, Newton's method requires much less iterations to satisfy the stopping criterion (47) but convergence may be wiggly or even fail if the quantity $M_{\mathrm{L}}^{-1}\Delta\tau$ becomes too large. This can be rectified by allowing

DOI: 10.1002/fld

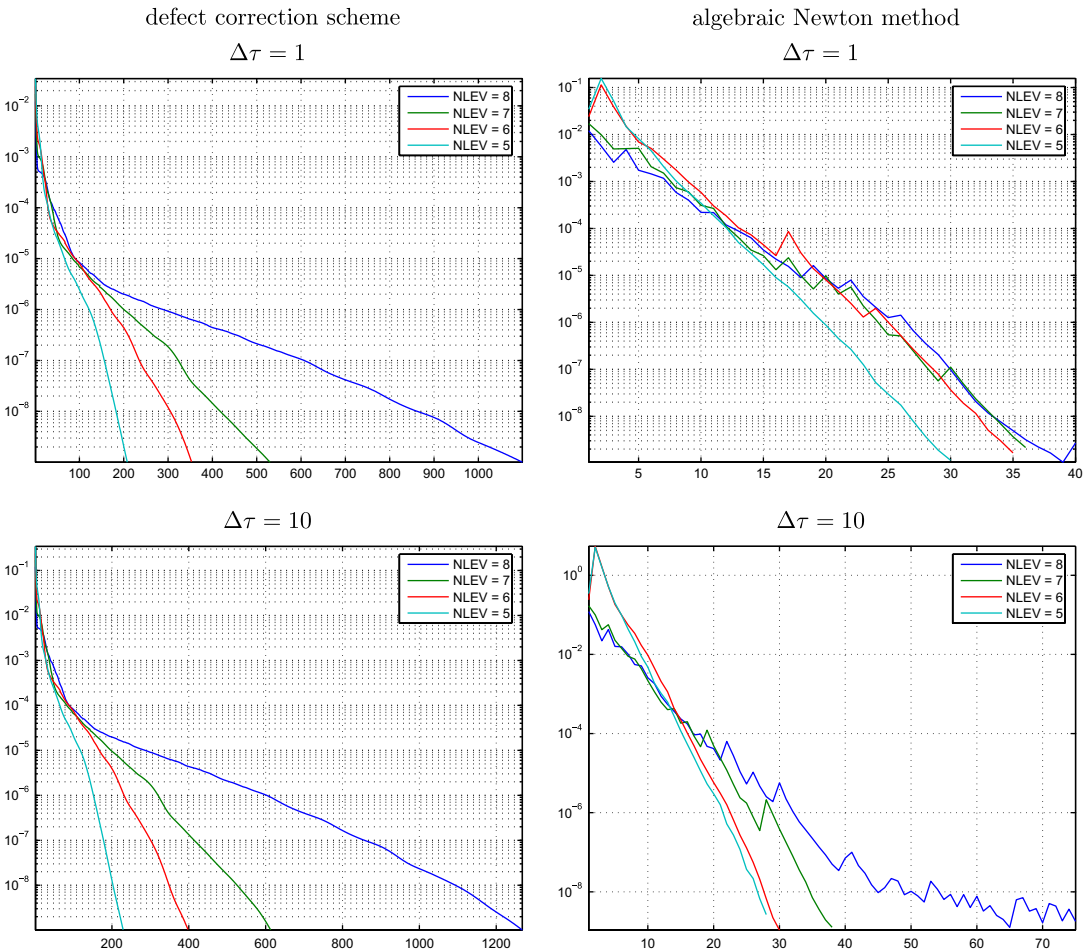

Figure 3. TP1: pseudo-transient convergence, FEM-TVD.

multiple nonlinear iterations per time step or reducing the size of $\Delta\tau$ adaptively if the residual norm starts to oscillate. Anyway, finding the optimal ratio between the inner defect correction/Newton method and the outer pseudo-time stepping loop is still a delicate task which may strongly depend on the benchmark problem.

5.2. *Convection in space–time*

The second test case deals with pure convection in the space–time domain $\Omega = I \times (0, T)$. To this end, let the flux function of our model problem (1) be defined according to

$$\nabla \cdot \mathbf{f}(u) := \frac{\partial f(u)}{\partial x} + \frac{\partial u}{\partial t} \tag{48}$$

DOI: 10.1002/fld

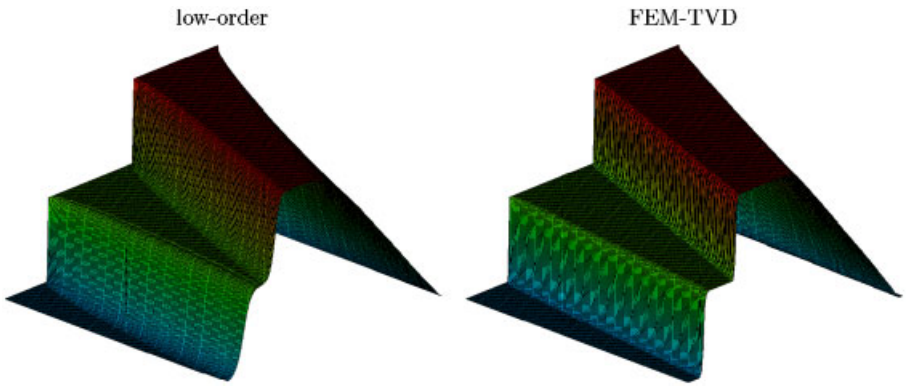

Figure 4. TP2: low-order *vs* FEM-TVD.

where $f(u)$ denotes a nonlinear function for the scalar variable u and $t \in (0, T)$ stands for the physical time. Interestingly enough, the above equation can be interpreted as a one-dimensional time-dependent conservation law defined on the spatial interval I for which the final solution is obtained at time T. The reformulation as a two-dimensional steady-state problem corresponds to computing the solution values for all time levels simultaneously instead of doing it step by step. The stationary conservation law (48) is complemented by suitable boundary conditions which need to be prescribed at the 'inlet' of the space–time domain Ω. Moreover, the initial data can be chosen arbitrarily since they do not affect the converged steady-state solution.

For the choice $f(u) = \frac{1}{2}u^2$, the well-known inviscid Burgers' equation is recovered from (48)

$$TP2: \quad u\frac{\partial u}{\partial x} + \frac{\partial u}{\partial t} = 0$$

For this nonlinear conservation law, let $T = 0.5$ be the right endpoint of the time domain and prescribe the following boundary conditions at the 'inlet' of $\Omega = (0, 1) \times (0, 0.5)$:

$$u(x,t) = \begin{cases} 1 & \text{if } 0 \leqslant x < 0.4 \wedge t = 0 \\ 0.5 & \text{if } 0.4 \leqslant x \leqslant 0.8 \wedge t = 0 \\ 0 & \text{if } 0.8 < x \leqslant 1 \wedge t = 0 \\ & \text{or } x = 0 \wedge 0 \leqslant t \leqslant 0.5 \end{cases} \tag{49}$$

The numerical solutions computed by the low-order upwind method and the FEM-TVD scheme on a uniform grid of 32 768 triangles are presented in Figure 4. Both profiles are free of spurious oscillations but only the use of algebraic flux correction yields an accurate resolution of the two shock waves and of the smooth transition along the rarefaction fan. In contrast, the discontinuities along the shocks are smeared by the overly diffusive upwind scheme.

The nonlinear convergence behaviour for the monotone low-order method and the FEM-TVD scheme is presented in Tables III–VI. For this test problem, multiple (<100) nonlinear iterations were accepted and the defect correction/Newton method was supposed to gain

DOI: 10.1002/fld

Table III. TP2: discrete upwind, defect correction.

NLEV	NVT	CPU	NN	NN/$\Delta\tau$	NL	NL/NN	$\|u-u_h\|_1$	$\|u-u_h\|_2$
$\Delta\tau=0.01$								
5	1089	<1	228	1.98	228	1.00	8.3934e−2	1.1657e−2
6	4225	2	190	1.98	190	1.00	5.2835e−2	8.3776e−2
7	16 641	5	208	2.36	208	1.00	3.1362e−2	5.9108e−2
8	66 049	29	303	3.61	303	1.00	1.7928e−2	4.1497e−2
$\Delta\tau=0.1$								
5	1089	<1	57	2.48	148	2.59	8.3934e−2	1.1657e−1
6	4225	<1	63	3.00	233	3.70	5.2835e−2	8.3776e−2
7	16 641	3	74	3.70	367	4.96	3.1362e−2	5.9108e−2
8	66 049	22	103	5.15	690	6.70	1.7928e−2	4.1497e−2
$\Delta\tau=1.0$								
5	1089	<1	33	3.30	218	6.60	8.3934e−2	1.1657e−1
6	4225	<1	36	4.00	392	10.89	5.2835e−2	8.3776e−2
7	16 641	5	48	5.33	973	20.27	3.1362e−2	5.9108e−2
8	66 049	45	67	7.44	2085	31.12	1.7928e−2	4.1497e−2

Table IV. TP2: discrete upwind, algebraic Newton.

NLEV	NVT	CPU	NN	NN/$\Delta\tau$	NL	NL/NN	$\|u-u_h\|_1$	$\|u-u_h\|_2$
$\Delta\tau=0.01$								
5	1089	<1	115	1.00	115	1.00	8.3934e−2	1.1657e−1
6	4225	1	160	1.66	160	1.00	5.2835e−2	8.3776e−2
7	16 641	6	192	2.18	192	1.00	3.1362e−2	5.9108e−2
8	66 049	40	299	3.56	299	1.00	1.7928e−2	4.1497e−2
$\Delta\tau=0.1$								
5	1089	<1	60	2.61	60	1.00	8.3934e−2	1.1657e−1
6	4225	1	77	3.66	94	1.22	5.2835e−2	8.3776e−2
7	16 641	3	76	3.80	180	2.37	3.1362e−2	5.9108e−2
8	66 049	18	84	4.20	349	4.15	1.7928e−2	4.1497e−2
$\Delta\tau=1.0$								
5	1089	<1	37	3.70	49	1.32	8.3934e−2	1.1657e−1
6	4225	<1	34	3.78	92	2.70	5.2835e−2	8.3776e−2
7	16 641	2	37	4.11	184	4.97	3.1362e−2	5.9108e−2
8	66 049	15	45	5.00	396	8.80	1.7928e−2	4.1497e−2

two digits per pseudo-time step. In our experience, the use of a more restrictive criterion does not pay off since the efficiency of the outer time-stepping loop could not be improved significantly. For the low-order method, no under-relaxation was adopted but the use of $\omega=0.8$ in Equation (18) was mandatory for the FEM-TVD solution to reach the stationary limit.

DOI: 10.1002/fld

Table V. TP2: FEM-TVD, defect correction.

NLEV	NVT	CPU	NN	NN/$\Delta\tau$	NL	NL/NN	$\|u-u_h\|_1$	$\|u-u_h\|_2$
$\Delta\tau=0.01$								
5	1089	5	4316	7.54	4316	1.00	4.2399e−2	7.8401e−2
6	4225	26	6329	11.72	6329	1.00	2.2989e−2	5.3425e−2
7	16 641	176	8822	20.00	8822	1.00	1.1813e−2	3.4460e−2
8	66 049	1075	11 362	40.43	11 362	1.00	5.7655e−3	1.9679e−2
$\Delta\tau=0.1$								
5	1089	5	3276	39.47	8462	2.58	4.2399e−2	7.8401e−2
6	4225	24	4443	54.18	11 067	2.49	2.2989e−2	5.3425e−2
7	16 641	n.a.	n.a.	n.a.	n.a.	n.a.	n.a.	n.a.
8	66 049	n.a.	n.a.	n.a.	n.a.	n.a.	n.a.	n.a.
$\Delta\tau=1.0$								
5	1089	3	1237	82.47	8316	6.72	4.2399e−2	7.8401e−2
6	4225	23	2709	30.78	16 037	5.92	2.2989e−2	5.3425e−2
7	16 641	n.a.	n.a.	n.a.	n.a.	n.a.	n.a.	n.a.
8	66 049	n.a.	n.a.	n.a.	n.a.	n.a.	n.a.	n.a.

Table VI. TP2: FEM-TVD, algebraic Newton.

NLEV	NVT	CPU	NN	NN/$\Delta\tau$	NL	NL/NN	$\|u-u_h\|_1$	$\|u-u_h\|_2$
$\Delta\tau=0.01$								
5	1089	8	2225	4.03	2225	1.00	4.23990e−2	7.84011e−2
6	4225	39	2661	4.68	2661	1.00	2.2989e−2	5.3425e−2
7	16 641	176	2371	5.75	3675	1.55	1.1813e−2	3.4461e−2
8	66 049	750	2090	6.85	5886	2.81	5.7655e−3	1.9679e−2
$\Delta\tau=0.1$								
5	1089	2	481	6.17	1336	2.78	4.2399e−2	7.8401e−2
6	4225	10	557	7.23	2867	5.15	2.2989e−2	5.3425e−2
7	16 641	99	1174	19.57	5630	4.79	1.1813e−2	3.4461e−2
8	66 049	732	1814	35.57	9837	5.42	5.7655e−3	1.9679e−2
$\Delta\tau=1.0$								
5	1089	1	126	9.69	745	5.91	4.2399e−2	7.8401e−2
6	4225	6	236	18.15	2041	8.65	2.2989e−2	5.3425e−2
7	16 641	33	287	26.09	3540	12.33	1.1813e−2	3.4461e−2
8	66 049	252	390	35.45	7494	19.22	5.7655e−3	1.9679e−2

The exact solution to this hyperbolic conservation law can be constructed from the shock speeds $s_1=0.75$ and $s_2=0.25$, and the following relation for the rarefaction wave:

$$u(x,t)=\begin{cases} x/0.5 & 0\leqslant x\leqslant 0.5 \\ 1 & x>0.5 \end{cases} \tag{50}$$

DOI: 10.1002/fld

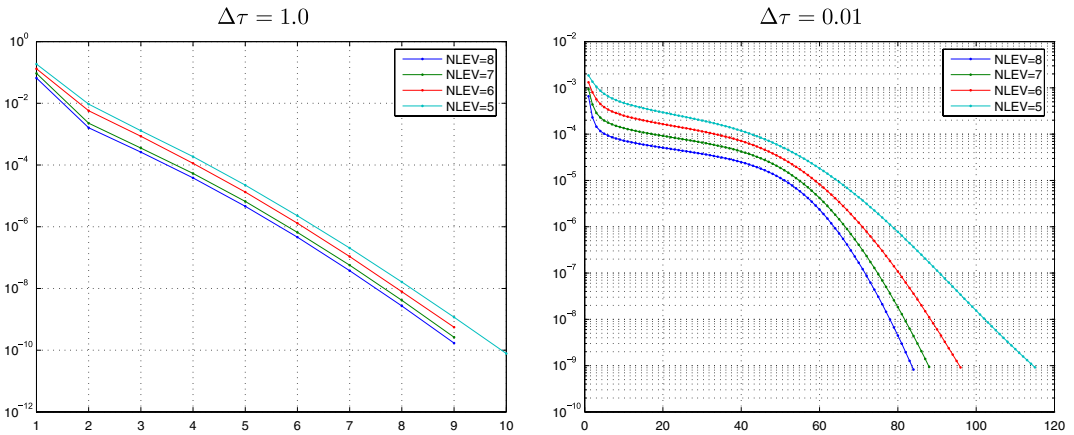

Figure 5. TP2: pseudo-transient convergence, discrete upwind.

The difference between u and the approximate solution u_h was measured in the L_1-norm

$$\|u - u_h\|_1 = \int_\Omega |u - u_h| \, \mathrm{d}x \approx \sum_i m_i |u(x_i, y_i) - u_i| \tag{51}$$

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

$$\|u - u_h\|_2^2 = \int_\Omega |u - u_h|^2 \, \mathrm{d}x \approx \sum_i m_i |u(x_i, y_i) - u_i|^2 \tag{52}$$

where $m_i = \int_\Omega \varphi_i \, \mathrm{d}x$ are the diagonal coefficients of the lumped mass matrix.

The first three columns of each of Tables III and IV display the refinement level (NLEV), the number of vertices (NVT) and the overall CPU time. In the next four columns, the total number of nonlinear cycles (NN), the average number of nonlinear iterations per pseudo-time step (NN/$\Delta\tau$), the total number of linear substeps (NL) and the average number of BiCGSTAB iterations required within each nonlinear step (NL/NN) are presented. All simulations were performed on four globally refined meshes with three different choices for the parameter $\Delta\tau \in \{0.01, 0.1, 1.0\}$. Recall that the Jacobian operator contains more non-zero matrix entries so that both the assembly and its application are more costly. If the time step is chosen too small, e.g. $\Delta\tau = 0.01$, then Newton's method ($\sigma = \sqrt{\varepsilon}$, forcing term from (26)) cannot demonstrate its full potential and falls behind the defect correction scheme speaking in terms of CPU time. On the other hand, the performance of the latter one deteriorates if larger time steps are employed whereas the moderate extra costs for our algebraic Newton approach pay off.

Figure 5 illustrates the pseudo-transient convergence behaviour which is quite similar for both nonlinear solution strategies. Only a moderate number of 9–10 steps is required to attain the steady-state solution if large time steps, e.g. $\Delta\tau = 1.0$, are adopted. This is where the benefit of the unconditionally stable backward-Euler method comes into play. In contrast, about 10 times more outer iterations have to be performed, if the step size is reduced to $\Delta\tau = 0.01$ which is inappropriate for the simulation of steady-state flow problems. It is worth mentioning that the use

DOI: 10.1002/fld

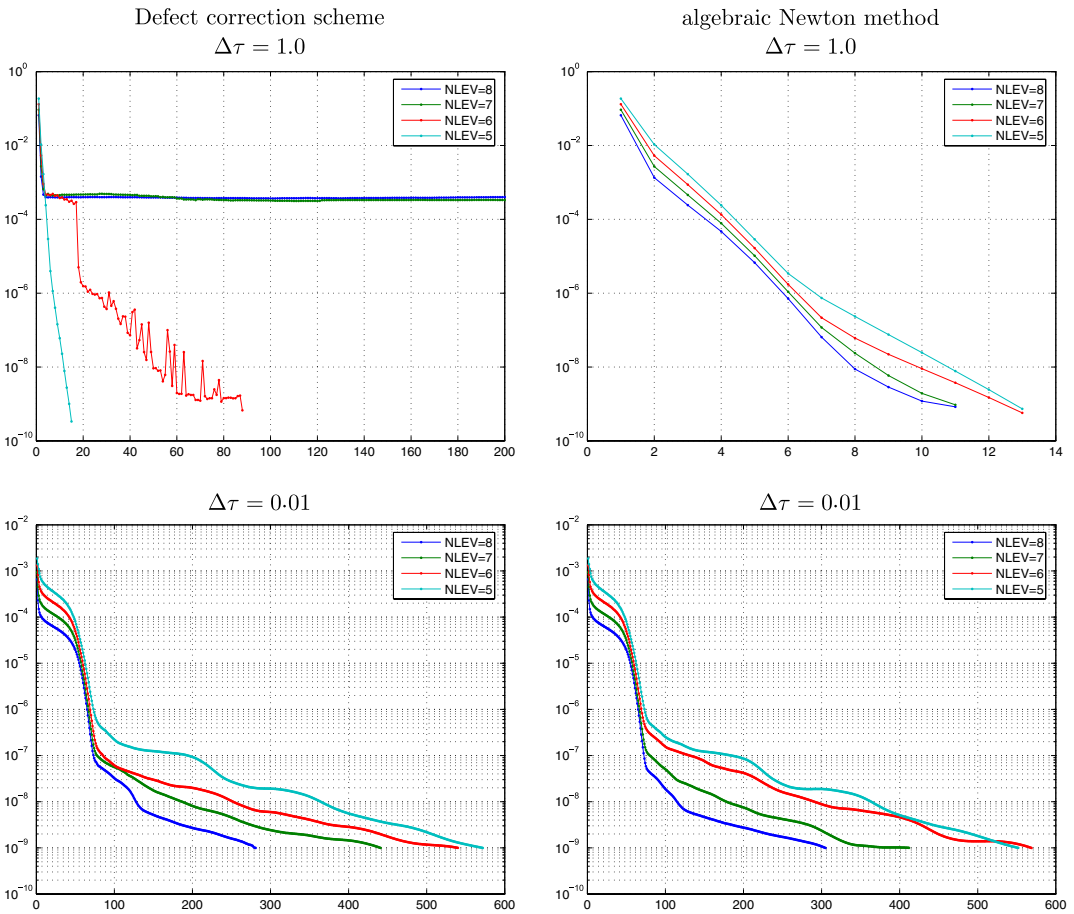

Figure 6. TP2: pseudo-transient convergence, FEM-TVD.

of an explicit time-stepping scheme would require even smaller values to ensure stability so that the use of implicit discretizations gains more attraction.

In order to reach steady-state convergence for the high-resolution FEM-TVD scheme, under-relaxation by the factor $\omega = 0.8$ was applied to the fixed-point iteration (18). Figure 6 illustrates the pseudo-transient convergence behaviour of both nonlinear strategies for two different time step sizes. Obviously, the standard defect correction scheme suffers from severe convergence problems for $\Delta\tau = 1.0$ if the mesh is refined. This is not the case for the algebraic Newton approach which allows for an efficient computation of the converged solution in 11–13 outer iterations. The performance of Newton's method deteriorates significantly and resembles that of the defect correction scheme if smaller time steps, e.g. $\Delta\tau = 0.01$, are employed.

A more detailed comparison of both methods is given in Tables V and VI. Due to the fact that the FEM-TVD algorithm was derived on the semi-discrete level, the solution error does not depend on the size of the underlying time step. However, the fixed-point defect correction scheme can only be operated at moderately small time steps in order to reach steady-state convergence.

DOI: 10.1002/fld

Newton's method is free of this drawback and performs best at $\Delta\tau = 1.0$. Speaking in terms of overall efficiency, the solution to the inviscid Burgers' equation on the finest grid level was obtained in 23% of the CPU time required by the defect correction method.

6. CONCLUSIONS

In this paper, we demonstrated that implicit high-resolution discretization schemes and efficient solution strategies can be successfully combined. The algebraic flux correction paradigm [10, 13–15] was revisited. The application of limiters eventually led to nonlinear algebraic systems which called for strong iterative solution strategies. Due to the lack of a continuous counterpart of the problem at hand, Newton-type schemes were derived on the fully discrete level. The Jacobian matrix was approximated by means of second-order divided differences. Each entry of the Newton operator was rewritten in terms of edge contributions so that an efficient assembly of the Jacobian could be performed edge by edge. This idea turned out to be an interesting alternative to the elementwise evaluation of the Jacobian which is traditionally employed in the finite element context [39]. We identified the individual sources of nonlinearities, i.e. the physical one present in the conservation law and the numerical one engendered by the flux limiter, and analysed their contributions to the global Jacobian. The indirect coupling of solution values at non-neighbouring vertices *via* the nodal correction factors made it necessary to extend the sparsity pattern of the underlying finite element matrix. Due to the similarities to edge-oriented stabilization techniques, existing storage algorithms could be adopted with slight modifications. An alternative strategy for generating the connectivity pattern by means of matrix multiplication was derived based on classical graph theory.

High-resolution schemes based on *algebraic* flux correction techniques can be equipped with efficient nonlinear solution strategies of algebraic Newton-type. This concept can also be extended [16] to flux limiters which are designed for the treatment of transient flows, e.g. the semi-implicit FEM-FCT limiter proposed in [10]. Even in case the time step needs to be moderately small due to physical reasons, the use of (semi-)implicit time-stepping schemes may be favourable due to less severe stability restrictions. The benefits of implicit discretization schemes become even more obvious if local grid refinement is employed which would require impractically small time steps for explicit methods. A promising direction for further research is the application of the discrete Newton method to the Euler and Navier–Stokes equations for which algebraic flux correction can be performed as explained in [17].

ACKNOWLEDGEMENTS

The author is grateful to Prof. Dmitri Kuzmin and to Prof. Stefan Turek for many valuable remarks.

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. Zalesak ST. Fully multidimensional flux-corrected transport algorithms for fluids. *Journal of Computational Physics* 1979; **31**:335–362.
3. 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.

DOI: 10.1002/fld

4. Löhner R, Morgan K, Vahdati M, Boris JP, Book DL. FEM-FCT: combining unstructured grids with high resolution. *Communications in Applied Numerical Methods* 1988; **4**:717–729.
5. Harten A. High resolution schemes for hyperbolic conservation laws. *Journal of Computational Physics* 1983; **49**:357–393.
6. Harten A. On a class of high resolution total-variation-stable finite-difference-schemes. *SIAM Journal on Numerical Analysis* 1984; **21**:1–23.
7. Arminjon P, Dervieux A. Construction of TVD-like artificial viscosities on two-dimensional arbitrary FEM grids. *Journal of Computational Physics* 1993; **106**(1):176–198.
8. Lyra PRM. Unstructured grid adaptive algorithms for fluid dynamics and heat conduction. *Ph.D. Thesis*, University of Wales, Swansea, 1994.
9. Kuzmin D, Turek S. Flux correction tools for finite elements. *Journal of Computational Physics* 2002; **175**(2): 525–558.
10. Kuzmin D, Kourounis D. A semi-implicit FEM-FCT algorithm for efficient treatment of time-dependent problems. *Technical Report 302*, University of Dortmund, 2005.
11. Kuzmin D, Möller M, Turek S. Multidimensional FEM-FCT schemes for arbitrary time-stepping. *International Journal for Numerical Methods in Fluids* 2003; **42**(3):265–295.
12. Kuzmin D, Möller M, Turek S. High-resolution FEM-FCT schemes for multidimensional conservation laws. *Computer Methods in Applied Mechanics and Engineering* 2004; **193**(45–47):4915–4946.
13. 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: Germany, 2005; 155–206.
14. Kuzmin D, Turek S. High-resolution FEM-TVD schemes based on a fully multidimensional flux limiter. *Journal of Computational Physics* 2004; **198**(1):131–158.
15. Kuzmin D. On the design of general-purpose flux limiters for finite element schemes. I. Scalar convection. *Journal of Computational Physics* 2006; **219**(2):513–531.
16. Möller M, Kuzmin D, Kourounis D. Implicit FEM-FCT algorithms and discrete Newton methods for transient convection problems. *Technical Report 340*, University of Dortmund, Dortmund, 2007.
17. Kuzmin D, Möller M. Algebraic flux correction II. Compressible Euler equations. In *Flux-Corrected Transport*, *Principles*, *Algorithms*, *and Applications*, Kuzmin D, Löhner R, Turek S (eds). Springer: Germany, 2005; 207–250.
18. Jameson A. Computational algorithms for aerodynamic analysis and design. *Applied Numerical Mathematics* 1993; **13**(5):383–422.
19. Jameson A. Analysis and design of numerical schemes for gas dynamics 1. Artificial diffusion, upwind biasing, limiters and their effect on accuracy and multigrid convergence. *International Journal of Computational Fluid Dynamics* 1995; **4**:171–218.
20. Ferziger JH, Perić M. *Computational Methods for Fluid Dynamics*. Springer: Berlin, 1996.
21. Arioli M, Loghin D, Wathen AJ. Stopping criteria for iterations in finite element methods. *Numerische Mathematik* 2005; **99**(3):381–410.
22. Arioli M, Noulard E, Russo A. Vector stopping criteria for iterative methods: applications to PDEs. *Calcolo* 2001; **38**:97–112.
23. Turek S. *Efficient Solvers for Incompressible Flow Problems*: *An Algorithmic and Computational Approach*. Lecture Notes in Computational Science and Engineering, vol. 6. Springer: Berlin, 1999.
24. Meijerink JA, van der Vorst HA. Iterative solution method for linear systems of which the coefficient matrix is a symmetric m-matrix. *Mathematics of Computation* 1977; **31**:148–162.
25. Hemker PW, Koren B. Defect correction and nonlinear multigrid for steady Euler equations. *Technical Report*, Centre for Mathematics and Computer Science, Amsterdam, Netherlands, 1988.
26. Dembo RS, Eisenstat SC, Steihaug T. Inexact Newton methods. *SIAM Journal on Numerical Analysis* 1982; **19**(2):400–408.
27. Shadid JN, Tuminaro RS, Walker HF. An inexact Newton method for fully coupled solution of the Navier–Stokes equations with heat and mass transport. *Journal of Computational Physics* 1997; **137**:155–185.
28. Tuminaro RS, Walker HF, Shadid JN. On backtracking failure in Newton-GMRES methods with a demonstration for the Navier–Stokes equations. *Journal of Computational Physics* 2002; **180**:549–558.
29. Eisenstat SC, Walker HF. Choosing the forcing term in an inexact Newton method. *SIAM Journal on Scientific Computing* 1996; **17**:16–32.
30. Eisenstat SC, Walker HF. Globally convergent inexact Newton methods. *SIAM Journal on Optimization* 1994; **4**(2):393–422.

DOI: 10.1002/fld

31. Moré JJ, Thuente DJ. Line search algorithms with guaranteed sufficient decrease. *ACM Transactions on Mathematical Software* 1984; **20**:286–307.
32. Pawlowski RP, Shadid JN, Simonis JP, Walker HF. Globalization techniques for Newton–Krylov methods and applications to the fully-coupled solution of the Navier–Stokes equations. *Report SAND2004-1777*, Sandia National Laboratories, 2004.
33. Hron J, Ouazzi A, Turek S. A computational comparison of two FEM solvers for nonlinear incompressible flow. *Technical Report 228*, University of Dortmund, 2003.
34. Nielsen EJ, Anderson WK, Walters RW, Keyes DE. Application of Newton–Krylov methodology to a three-dimensional unstructured Euler code. *AIAA 95-0221*, 1995.
35. Knoll DA, Keyes DE. Jacobian-free Newton–Krylov methods: a survey of approaches and applications. *Journal of Computational Physics* 2004; **193**:357–397.
36. Choquet R. A matrix-free preconditioner applied to CFD. *Technical Report 2605*, INRIA, 1995.
37. Saad Y. A flexible inner-outer preconditioned GMRES algorithm. *SIAM Journal on Scientific Computing* 1993; **14**:461–469.
38. van der Vorst HA, Vuik C. A comparison of some GMRES-like methods. *Linear Algebra and its Applications* 1992; **160**:131–162.
39. Capon PJ, Jimack PK. An inexact Newton method for systems arising from the finite element method. *Applied Mathematics Letters* 1997; **10**(3):9–12.
40. Burman E, Ern A. Stabilized Galerkin approximation of convection–diffusion–reaction equations: discrete maximum principle and convergence. *Mathematics of Computation* 2005; **74**(252):1637–1652.
41. Burman E, Hansbo P. Edge stabilization for Galerkin approximations of convection–diffusion–reaction problems. *Computer Methods in Applied Mechanics and Engineering* 2005; **193**(15–16):1437–1453.
42. Ouazzi A, Turek S. Unified edge-oriented stabilization of nonconforming finite element methods for incompressible flow problems. *Technical Report 284*, University of Dortmund, Dortmund, 2005.
43. Bank EB, Douglas CC. SMMP: sparse matrix multiplication package. *Advances in Computational Mathematics* 1993; **1**:127–137.
44. Venkatakrishnan V. Convergence to steady state solutions of the Euler equations on unstructured grids with limiters. *Journal of Computational Physics* 1995; **118**:120–130.

DOI: 10.1002/fld