A&A 544, A47 (2012)
DOI: 10.1051/0004-6361/201116979

Astronomy & Astrophysics

On multigrid solution of the implicit equations of hydrodynamics

Experiments for the compressible Euler equations in general coordinates

K. Kifonidis and E. Müller

Max-Planck-Institut für Astrophysik, Karl-Schwarzschild-Straße 1, 85741 Garching, Germany
e-mail: kifonidisk@googlemail.com

Received 30 March 2011 / Accepted 14 May 2012

ABSTRACT

Aims. We describe and study a family of new multigrid iterative solvers for the multidimensional, implicitly discretized equations of hydrodynamics. Schemes of this class are free of the Courant-Friedrichs-Lewy condition. They are intended for simulations in which widely differing wave propagation timescales are present. A preferred solver in this class is identified. Applications to some simple stiff test problems that are governed by the compressible Euler equations, are presented to evaluate the convergence behavior, and the stability properties of this solver. Algorithmic areas are determined where further work is required to make the method sufficiently efficient and robust for future application to difficult astrophysical flow problems.
Methods. The basic equations are formulated and discretized on non-orthogonal, structured curvilinear meshes. Roe's approximate Riemann solver and a second-order accurate reconstruction scheme are used for spatial discretization. Implicit Runge-Kutta (ESDIRK) schemes are employed for temporal discretization. The resulting discrete equations are solved with a full-coarsening, nonlinear multigrid method. Smoothing is performed with multistage-implicit smoothers. These are applied here to the time-dependent equations by means of dual time stepping.
Results. For *steady-state* problems, our results show that the efficiency of the present approach is comparable to the best implicit solvers for conservative discretizations of the compressible Euler equations that can be found in the literature. The use of red-black as opposed to symmetric Gauss-Seidel iteration in the multistage-smoother is found to have only a minor impact on multigrid convergence. This should enable scalable parallelization without having to seriously compromise the method's algorithmic efficiency. For *time-dependent* test problems, our results reveal that the multigrid convergence rate degrades with increasing Courant numbers (i.e. time step sizes). Beyond a Courant number of nine thousand, even complete multigrid breakdown is observed. Local Fourier analysis indicates that the degradation of the convergence rate is associated with the coarse-grid correction algorithm. An implicit scheme for the Euler equations that makes use of the present method was, nevertheless, able to outperform a standard explicit scheme on a time-dependent problem with a Courant number of order 1000.
Conclusions. For steady-state problems, the described approach enables the construction of parallelizable, efficient, and robust implicit hydrodynamics solvers. The applicability of the method to time-dependent problems is presently restricted to cases with moderately high Courant numbers. This is due to an insufficient coarse-grid correction of the employed multigrid algorithm for large time steps. Further research will be required to help us to understand and overcome the observed multigrid convergence difficulties for time-dependent problems.

Key words. methods: numerical – hydrodynamics – magnetohydrodynamics (MHD) – relativistic processes – radiative transfer – stars: evolution

1. Introduction

The ability to solve systems of hyperbolic partial differential equations (PDEs) in an efficient manner is crucial to most branches of modern theoretical astrophysics. For instance, it is vital to cosmology, the dynamics of the intergalactic and interstellar medium, the formation and evolution of stars, and the interactions of stars with their environment through stellar winds, novae, and supernova explosions. In all of these fields, solutions to hyperbolic conservation laws are required, such as the equations of classical, special relativistic, or magnetohydrodynamics.

Many astrophysical problems that require the solution of a hyperbolic system of equations involve vastly different physical timescales. This disparity, or "stiffness", can have several origins. It can result from widely varying wave propagation times within the computational domain, from one's wish to follow the evolution of an astrophysical system over *secular* times that are much longer than the wave propagation times, or from additional physics in the form of source terms. Known remedies to each of these difficulties all require some form of *time-implicit* discretization of (parts of) the governing equations. Purely *time-explicit* integration schemes – which calculate the physical state at a given time level using only known information from previous steps – are inefficient for the solution of stiff problems.

If the stiffness is confined to a local source term appearing in the conservation law, a semi-implicit treatment is sufficient, i.e. only the source term has to be treated in a time-implicit fashion in each cell of the grid, while the advection-related part of the equations can still be discretized using an explicit scheme. Hence, only the state variables *within* a cell need to be updated in a tightly coupled manner. A particular difficulty with stiff sources, however, is that owing to their short-time variation these terms typically show appreciable changes over small spatial scales, which may be very hard to resolve. If the employed grid is coarse, one may typically have to tailor the discretization of the advection terms in a way that the scheme at least fulfills certain (problem-dependent) asymptotic properties. In this paper, we do not deal with stiff source terms and their associated discretization-related difficulties, and refer the reader to the significant body of literature that is already available in this context

(e.g. Pember 1993; Caflisch et al. 1997; Miniati & Colella 2007; Dumbser et al. 2008).

We focus here, instead, on stiffness, which is the result of widely varying wave propagation times within the computational domain (and the related problem of following the evolution of a dynamical system over secular timescales). This form of stiffness may be further sub-classified into two types. In the first type, the disparity is due to a large condition number, i.e. a large ratio of the largest to smallest moduli of the wave speeds admitted by the system of conservation laws given the problem, and physical conditions, under study. This may also be referred to as "analytic" stiffness, because it is a property of the analytic (continuous) equations. In the second type, disparity in the wave propagation times arises as a result of the discretization process, as the governing equations often need to be discretized on (highly) non-uniform grids in order to resolve steep gradients in the solution. This may be called "discrete" or geometrical stiffness. Here the stiffness is caused by disparate mesh cell widths. We note that both of these forms of stiffness are typically *non-local*, i.e. the longest and shortest wave propagation times typically occur in different regions of the computational domain. Moreover, both of these forms often occur *simultaneously* in practical problems, so that they greatly exacerbate each other.

An example of analytic stiffness is the multidimensional convection that occurs during quiescent stellar evolution phases. The Mach numbers in stellar convection zones are typically very low (of order 10^{-2} or less), i.e. the propagation time of acoustic waves through the convection zone is much shorter than the timescale of advection. An example where both forms of wave-related stiffness occur, are attempts to follow the convective flows in an entire star (e.g. Eggleton et al. 2002). Here, it is important to resolve the huge gradients of the hydrodynamic variables from the stellar center to the surface. This necessitates a highly non-uniform radial grid, which in turn leads to additional discrete stiffness.

A scheme that discretizes the advection terms of the hydrodynamic equations explicitly, needs to respect the well-known Courant-Friedrichs-Lewy (CFL) stability condition. It thus forces one to follow the solution with a tiny time-step size of order the propagation time of the fastest acoustic waves in the domain, even if one's *actual* interest is to follow some much slower processes, e.g. the convective flows mentioned in the above examples, or even the quasi-steady evolution of the hydrodynamic background state in a stellar evolution calculation. This is a crucial impediment in much of the multidimensional stellar-evolution-related work that has been done to date. As examples, we cite here both two and three-dimensional studies of the core helium flash in low mass stars (Dearborn et al. 2006; Mocák et al. 2008, 2009, 2010), the three-dimensional evolution of shell convective burning in massive stars (Meakin & Arnett 2006), and simulations that try to follow the late-time evolution of core-collapse supernova ejecta into the supernova remnant phase (Gawryszczak et al. 2010). In all these studies – which used explicit schemes – compromises had to be made with respect to the simulated time, as it was impossible, owing to the stiffness of the equations, to follow the system under study over sufficiently long times to obtain satisfactory insight into its long-term physical behavior.

Especially challenging instances of wave-propagation-related stiffness are encountered in the flows of the simmering and ignition phase of classical novae (Kercek et al. 1999; Glasner et al. 2007), X-ray bursts (Lin et al. 2006), and Type Ia supernovae (Höflich & Stein 2002; Zingale et al. 2009). These flows are initially deeply subsonic, but due to energy release by thermonuclear runaway processes, they evolve to become transonic or even supersonic. In these cases, deeply subsonic (nearly incompressible) flow regions can even coexist with (highly compressible) supersonic ones in the same computational domain. Explicit, compressible flow solvers are of very limited use in these situations, as are purely low Mach number flow solvers, such as those developed by Almgren et al. (2006a,b, 2008). To solve flow problems of this kind one needs to develop schemes which can simultaneously handle both the incompressible and compressible regimes.

To be applicable to both of these limits, a numerical scheme must fulfill two essential requirements: it must be free of the CFL condition, and its discretization must account for the different, i.e. elliptic and hyperbolic, character of the equations in the incompressible and compressible regimes, respectively. In other words, the numerical dissipation of the scheme must scale appropriately with the Mach number in both regimes. The latter condition is *not* automatically fulfilled by standard solvers for either compressible or incompressible flow. In compressible flow solvers, this condition has to be enforced by an explicit rescaling of the (dissipative) numerical fluxes at low Mach numbers. A consistent reformulation of Roe's approximate Riemann solver along these lines was presented by Rossow (2007). Alternative, earlier, approaches made use of low Mach number flux-preconditioning matrices for this purpose (see Turkel 1999, for a review). Without these modifications, spurious solutions will result if the Mach number is sufficiently low (Guillard & Viozat 1999).

While problems of stiffness are already inherent to astrophysical systems that are governed by the classical Euler (or Navier-Stokes) equations, these problems are typically exacerbated if, in addition, magnetic fields are present, or if some part of the flow is relativistic. The magnetohydrodynamic (MHD) equations, for instance, admit fast magnetosonic and Alfvén waves as their solutions, which typically lead to larger analytic stiffness than in the pure Euler equations (see also the review of Camenzind 2005). Owing to these difficulties, multidimensional MHD simulations over secular timescales, for instance, are seldom attempted (for an exception, see Glatzmaier & Roberts 1995). Hence, it is unsurprising that our present knowledge of the strength, distribution, and evolution of magnetic fields in stars over these timescales relies entirely on one-dimensional (1D) simulations (Heger et al. 2005), and that parametrizations of these fields are therefore required to study their subsequent role in multidimensional supernova simulations and neutron star formation (Obergaulinger & Janka 2011).

The only way to avoid the stiffness problems associated with the restrictive CFL condition, is to use a globally implicit discretization of the flow equations, i.e. a discretization in which in particular the *space derivative* (advection) terms of the equations are implicitly evaluated. We note that, in terms of computational cost, this is a much more challenging task than dealing with a stiff local source term: the coupling of neighboring zones by discrete implicit advection operators results in very large systems of non-linear algebraic equations, in which all of the state variables across the *entire grid* are coupled with each other. For each time step of a simulation that employs a globally implicit scheme, one or more of these systems need to be solved. As long as they result from one-dimensional problems, conventional (e.g. direct) methods can be used for their solution. Yet, the computational cost of these same solvers applied to the equations resulting from multidimensional, globally implicit discretization is far too high, making these approaches computationally unfeasible.

For multidimensional problems, extraordinarily efficient iterative solvers are urgently required. The computational cost of the latter should, ideally, scale linearly with the number of grid zones. Only if such scaling can be attained will the underlying implicit scheme be of optimal efficiency. Otherwise its cost per time step will (grossly) exceed that of an explicit scheme, possibly by orders of magnitude, thereby offsetting much of the gain achieved through the use of a longer time step. The memory use and the ability to develop a parallel version of the method are additional important factors. Here, too, one should strive for characteristics that are not much more demanding than those of explicit schemes. This is challenging, though, as globally implicit solvers typically need more memory, and are more difficult to parallelize than explicit ones.

Recent approaches to constructing globally implicit flow solvers for astrophysical applications (Lee et al. 2011; Viallet et al. 2011) were unable to fulfill all of the above requirements because they attempted to achieve a maximum of modularity and ease of coding, and to exploit (conventional) linear solver technology. These goals typically lead to algorithms that are based on global linearization and Newton iteration. Lee et al. (2011), for instance, make use of a Newton-Krylov iterative method with a Schwartz preconditioner, which had been formerly employed by Tóth et al. (2006). On the other hand, Viallet et al. (2011) presently employ Newton iteration with an expensive direct linear solver that the authors would like to swap for an iterative one in the future. A common property of all these approaches is that they separate the discretization from the actual solution process, which leads to inefficiency.

The method that we describe in this paper differs markedly from those developed in former works in being a first step in a holistic development process whose long-term goal is to arrive at an optimal implicit solver. The underlying philosophy is to achieve optimality by intertwining the discretization and solution processes in the framework of the multigrid technique, which is the only known approach that can, in principle, lead to algorithms that fulfill all of the above requirements. Optimal multigrid schemes converge with a constant rate, which is of order 0.1 (or smaller). In other words, these algorithms are able to reduce the residual of the discrete implicit equations by (more than) an order of magnitude per iteration cycle, *independent* of the problem size, i.e. the total number of unknowns and, hence, the number of grid zones, N. The arithmetic cost and storage requirements of such methods thus scale like $O(N)$, and are moreover low (Brandt 1984; Trottenberg et al. 2001).

Another very important property of the multigrid technique is its *direct* applicability to non-linear problems. A non-linear multigrid cycling scheme, the so-called "full approximation storage" (FAS) scheme, was devised for this purpose by Brandt (1977, 1984). The FAS scheme has the important advantage that it does not require global linearization of the discretized equations[1], in contrast to the aforementioned Newton-Krylov schemes (see also Knoll & Keyes 2004; and Hujeirat & Rannacher 2001, for these methods). This offers advantages in terms of efficiency (as an outer iteration loop to treat the non-linearity is not required), robustness (as there is no reliance on well-conditioned global Jacobian matrices), and memory use (as the set-up and storage of global Jacobians is completely avoided). In addition, the FAS algorithm fits naturally into the framework of locally and adaptively refined meshes (in this context, it has been referred to as the multilevel adaptive technique, or MLAT, by Brandt 1977, 1984), and it allows for efficient parallelization on distributed memory computers using domain decomposition techniques.

The FAS scheme derives its efficiency from the interplay between two crucial ingredients: a conventional iterative ("relaxation") scheme that acts as a smoother of high frequency components of the solution residual (in Fourier space), and a coarse-grid correction algorithm that annihilates the remaining low frequency modes of the residual, which are responsible for the slow convergence of conventional iterative methods (Brandt 1977, 1984; Wesseling & Oosterlee 2001; Trottenberg et al. 2001). Both the smoother and the coarse-grid correction algorithm depend on the spatial discretization scheme with which one discretizes the equations. Thus, all of these components have to be matched to each other in order to obtain an optimally efficient multigrid solver. This is challenging for the equations of hydrodynamics because the determinant of their (linearized) spatial discretization operator contains hyperbolic as well as elliptic factors. In addition, the solutions of these equations can exhibit anisotropies, and discontinuities such as shocks or contact surfaces. All of these problems complicate the design of effective multigrid schemes (cf. Brandt 1984, 2001; Trottenberg et al. 2001; Thomas et al. 2003). Present experience for steady problems indicates that good multigrid performance can nevertheless be obtained[2], but at the expense of some complexity in the smoother, the spatial discretization scheme, and/or the coarse-grid correction algorithm (see the reviews of Wesseling & Oosterlee 2001; Thomas et al. 2003).

A widely adopted approach for the solution of steady problems in computational fluid dynamics (CFD) is to employ so-called marching schemes as smoothers in FAS multigrid algorithms (Jameson 1983, 1986). The basic idea here is to use the numerical dissipation – which is inherent to any stable spatial discretization scheme of the hydrodynamic equations – to smooth the high frequency content of the solution residual. This is achieved by simply marching the equations forward in a pseudo-time coordinate. The latter approach has the additional advantage that the transient problems, which need to be solved, are always hyperbolic and well-posed. This greatly facilitates convergence to a solution starting from some arbitrary initial state, i.e. it provides the method with a large convergence radius, which is in marked contrast to the small convergence radii of Newton-type schemes.

In addition to smoothing the error, marching schemes also eliminate error modes by advecting them through the boundaries of the computational domain. Hence, numerical stability, and the scheme's error propagation properties are important considerations in hyperbolic multigrid design, and a choice between an explicit and implicit marching method needs to be made (Jameson 1986). This choice, and the anisotropies present in the problem, in turn determine whether a "full-coarsening" strategy (where coarse grids are obtained by doubling the mesh size in every dimension, see Sect. 5.4.2) is sufficient to calculate the coarse-grid correction, or semicoarsening in one (e.g. Pierce & Giles 1997) or even multiple directions needs to be employed (Mulder 1989; Darmofal & Siu 1999; Trottenberg et al. 2001).

To date, multigrid algorithms have been used by the astrophysics community almost exclusively as solvers for *elliptic* equations, particularly the linear Poisson equation. Their use for the solution of non-linear hyperbolic problems is currently

[1] Depending on the employed coarsening strategy and the occurrence of anisotropies in the PDE, some form of semi-global linearization (similar to that used, e.g., in line relaxation) may be helpful, though, in the multigrid smoother, see also Sect. 5.3.3.

[2] Especially for problems with non-closed characteristics.

extremely limited. Among the very few astrophysical studies that have adopted multigrid for hyperbolic problems, are those of Fabiani Bendicho et al. (1997) and Balsara (2001), who have both focused on the stationary radiation-transport equation (which is a single scalar equation). The purpose of our paper is to present the construction of a family of FAS multigrid cycling schemes for a *hyperbolic system*, namely the Euler equations, and to evaluate their convergence and stability characteristics. Our main aim is to judge whether state-of-the-art schemes of this kind, which have been developed predominantly by the CFD and aerospace engineering communities for steady problems of the Euler and Navier-Stokes equations, are already sufficiently efficient, or whether they need to be suitably adapted, to form the computational cores of future globally-implicit, multidimensional, astrophysical flow solvers.

As nearly all astrophysical problems of particular interest are time-dependent, this requires primarily a robust capability of the scheme to handle time-dependent flows. The methods that we study employ the popular dual time-stepping technique (cf. Jameson 1991; Melson et al. 1993) for this purpose. In approaches of this kind, a discretization of the fully time-dependent equations is first cast into the form of a steady-state problem, which is subsequently solved with a FAS multigrid steady-state solver. The smoother is of the aforementioned pseudo-time marching type, i.e. it basically consists of an explicit integrator for ordinary differential equations. The smoothing and stability properties of this basic scheme, however, are significantly enhanced by the use of implicit stages, as proposed by Rossow (2006, 2007) and Swanson et al. (2007).

In this paper, our interest is in the convergence speed and stability properties of these multigrid algorithms. Both convergence speed and stability determine the robustness of a method. Multigrid robustness is known to be sensitive not only to the choice of inappropriate multigrid components, but also to even minor errors in discretization schemes, smoothers, the equations of the base algorithm, or their coding (Brandt 1984). This makes it necessary for us to provide a complete account of the multigrid scheme, and the discretization and smoothing methods that we use, to enable other researchers to reproduce our results.

In this pilot study, we apply the algorithms to several simple 1D and two-dimensional (2D) test problems described by the compressible Euler equations in planar (slab) geometry. These problems exhibit many of the difficulties that one typically also encounters in more complicated situations. When developing multigrid schemes, it is crucial to isolate these difficulties in as simple a problem setup as possible, in order not to cloud their effects by other factors. Only in this way is it possible to arrive at an understanding of the inner workings of a specific multigrid algorithm, which is essential for devising improvements to exploit the full potential of the method (Brandt 1984). An additional consideration is that, for the problems that we have selected, there exists plenty of literature regarding their implicit solution. Hence, the efficiency of our algorithms and our computer implementation can be directly compared to other approaches, and judged in an objective manner. Applications of the schemes to axisymmetric, three-dimensional, and low-Mach number flows, as well as their extension to complex astrophysical problems with general equations of state, and both wave-related stiffness, *and* stiffness due to source terms, will be the subject of future work.

The remainder of this article is organized as follows. We first discuss the governing continuous equations in Sect. 2 in the form as they are discretized in our code, which involves their transformation to generalized curvilinear coordinates that enable us to use non-orthogonal meshes. We then describe our spatial discretization approach in Sect. 3. This is based on the approximate Riemann solver of Roe (1981), and the symmetric limited positive (SLIP) high-resolution scheme of Jameson (1995b) for spatial reconstruction. Temporal discretization with implicit Runge-Kutta (ESDIRK) schemes is treated in Sect. 4. The multigrid solution of the resulting non-linear systems is the subject of Sect. 5. In Sect. 6, we give a brief overview of the local Fourier analysis tool that we employ in this paper to obtain insight into theoretical multigrid convergence rates. The convergence speed, performance, and stability of our algorithms is then evaluated and analyzed in terms of problems of the compressible Euler equations in Sect. 7. On the basis of results for a well-known test problem, we identify in Sect. 7.1 a preferred multigrid solver among the family of methods that we present, which we then use throughout the remainder of the paper for both steady and time-dependent problems. Our conclusions, along with a list of necessary future work, are given in Sect. 8.

2. Basic equations and choice of coordinates

For the sake of keeping the following equations concise and allow the maximum clarity in our exposition, we restrict ourselves here to a discussion of the 2D case. All concepts can be extended naturally to three spatial dimensions, however, and have also been implemented in three dimensions in our code.

2.1. Transforming the equations

In contrast to common astrophysical practice, in which orthogonal meshes tend to be adopted, we use general curvilinear meshes to cover non-rectangular domains in physical space. These meshes may actually be non-orthogonal, and are usually also non-uniform (i.e. non-equidistant). All of these complications can be easily handled if the governing equations formulated in Cartesian coordinates are transformed from physical space into a computational space (spanned by the generalized coordinates ξ, η) in which the mesh is rectangular and uniform, with spacing $\Delta\xi = \Delta\eta = 1$ (see Fig. 1), and the boundaries of the problem coincide with some ξ = const. or η = const. coordinate lines (cf. Thompson et al. 1985).

We define J^{-1} to represent the Jacobian determinant of a (sufficiently smooth) transformation that uniquely maps (ξ, η) into (x, y) space, and J to be the Jacobian determinant of the inverse transformation, i.e.

$$J^{-1} = \left|\frac{\partial(x,y)}{\partial(\xi,\eta)}\right|, \quad J = \left|\frac{\partial(\xi,\eta)}{\partial(x,y)}\right|. \tag{1}$$

Furthermore, we consider a 2D system of first-order hyperbolic conservation laws in physical space given by

$$\frac{\partial \boldsymbol{Q}}{\partial t} + \frac{\partial \boldsymbol{F}(\boldsymbol{Q})}{\partial x} + \frac{\partial \boldsymbol{G}(\boldsymbol{Q})}{\partial y} = \boldsymbol{S}(\boldsymbol{Q}), \tag{2}$$

where $\boldsymbol{Q}$ is the vector of conserved variables, $\boldsymbol{F}$ and $\boldsymbol{G}$ are the Cartesian flux functions, and $\boldsymbol{S}$ is a source term. In general, the source term could be stiff. However since, in this case, the discretization may have to fulfill special (problem-dependent, asymptotic) properties (see the references cited in Sect. 1), and since a discussion of these complications is beyond the scope of this work, we assume $\boldsymbol{S}$ to be non-stiff in the following.

Expressing the derivatives $\partial/\partial x$ and $\partial/\partial y$, that appear in Eq. (2), in terms of $\partial/\partial\xi$ and $\partial/\partial\eta$ using the chain rule of differentiation, and multiplying the resulting equation with J^{-1}, one

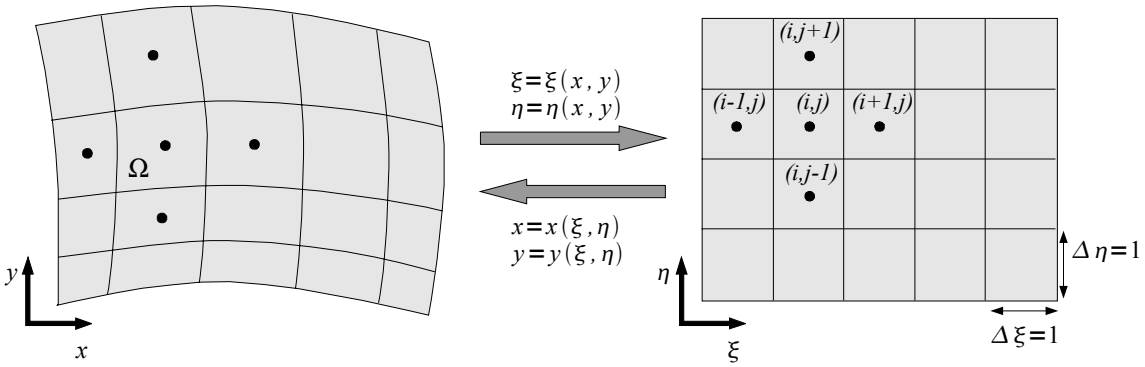

Fig. 1. Illustration of the mappings between the non-equidistant, non-orthogonal mesh in physical space (*left*), and the uniform rectangular mesh in computational space with spacing $\Delta\xi = \Delta\eta = 1$ (*right*).

can show (cf. Vinokur 1974; Anderson et al. 1984) that the transformed system can be written in strong conservation form as

$$\frac{\partial \hat{\boldsymbol{Q}}(\boldsymbol{Q})}{\partial t} + \frac{\partial \hat{\boldsymbol{F}}(\boldsymbol{Q})}{\partial \xi} + \frac{\partial \hat{\boldsymbol{G}}(\boldsymbol{Q})}{\partial \eta} = \hat{\boldsymbol{S}}(\boldsymbol{Q}). \tag{3}$$

The functions $\hat{\boldsymbol{Q}}$, $\hat{\boldsymbol{F}}$, $\hat{\boldsymbol{G}}$, and $\hat{\boldsymbol{S}}$ are given by the equations

$$\hat{\boldsymbol{Q}}(\boldsymbol{Q}) = J^{-1}\boldsymbol{Q}, \tag{4}$$

$$\hat{\boldsymbol{F}}(\boldsymbol{Q}) = J^{-1}\left(\xi_x \boldsymbol{F}(\boldsymbol{Q}) + \xi_y \boldsymbol{G}(\boldsymbol{Q})\right), \tag{5}$$

$$\hat{\boldsymbol{G}}(\boldsymbol{Q}) = J^{-1}\left(\eta_x \boldsymbol{F}(\boldsymbol{Q}) + \eta_y \boldsymbol{G}(\boldsymbol{Q})\right), \tag{6}$$

$$\hat{\boldsymbol{S}}(\boldsymbol{Q}) = J^{-1}\boldsymbol{S}(\boldsymbol{Q}), \tag{7}$$

where the subscripts denote partial differentiation. The coefficients appearing in these equations are the metric derivatives (or metrics) for which the following relations hold (e.g., Anderson et al. 1984)

$$J^{-1}\xi_x = y_\eta, \tag{8}$$

$$J^{-1}\xi_y = -x_\eta, \tag{9}$$

$$J^{-1}\eta_x = -y_\xi, \tag{10}$$

$$J^{-1}\eta_y = x_\xi. \tag{11}$$

For the Jacobian determinant J^{-1}, one has

$$J^{-1} = 1/J = x_\xi y_\eta - x_\eta y_\xi. \tag{12}$$

We note that Eq. (3) is the covariant form of the conservation law Eq. (2), and that Eqs. (8)–(11) simply relate the components of the contravariant base vectors, $(\xi_x, \xi_y)^{\mathrm{T}}$ and $(\eta_x, \eta_y)^{\mathrm{T}}$, of the curvilinear coordinate system to those of the covariant base vectors $(x_\xi, y_\xi)^{\mathrm{T}}$ and $(x_\eta, y_\eta)^{\mathrm{T}}$.

2.2. Euler equations

In the special case of the (homogeneous) Euler equations, the Cartesian vectors are given by

$$\boldsymbol{Q} = \begin{pmatrix} \rho \\ \rho u \\ \rho v \\ \rho E \end{pmatrix}, \quad \boldsymbol{F}(\boldsymbol{Q}) = \begin{pmatrix} \rho u \\ \rho u^2 + p \\ \rho uv \\ \rho uH \end{pmatrix}, \quad \boldsymbol{G}(\boldsymbol{Q}) = \begin{pmatrix} \rho v \\ \rho vu \\ \rho v^2 + p \\ \rho vH \end{pmatrix}, \tag{13}$$

and $\boldsymbol{S}(\boldsymbol{Q}) = 0$. Here ρ is the density of the fluid, p the pressure, u and v the velocities in the x and y direction, respectively,

$$E = e + \frac{u^2 + v^2}{2} \tag{14}$$

is the (specific) total energy, e the internal energy, and

$$H = E + \frac{p}{\rho} \tag{15}$$

the total enthalpy. The system is closed by the perfect gas equation of state

$$p = (\gamma - 1)\rho e, \tag{16}$$

where γ is the ratio of specific heats.

The transformed fluxes for this system can be written in a concise form if we introduce the velocities

$$\bar{U} = \left(J^{-1}\xi_x\right) u + \left(J^{-1}\xi_y\right) v, \tag{17}$$

and

$$\bar{V} = \left(J^{-1}\eta_x\right) u + \left(J^{-1}\eta_y\right) v. \tag{18}$$

From Eqs. (5) and (6), it then follows that

$$\hat{\boldsymbol{F}}(\boldsymbol{Q}) = \begin{pmatrix} \rho\bar{U} \\ \rho\bar{U}u + J^{-1}\xi_x p \\ \rho\bar{U}v + J^{-1}\xi_y p \\ \rho\bar{U}H \end{pmatrix}, \quad \hat{\boldsymbol{G}}(\boldsymbol{Q}) = \begin{pmatrix} \rho\bar{V} \\ \rho\bar{V}u + J^{-1}\eta_x p \\ \rho\bar{V}v + J^{-1}\eta_y p \\ \rho\bar{V}H \end{pmatrix}, \tag{19}$$

which resembles the Cartesian form of the fluxes in Eq. (13).

2.3. Interpretation of the metrics

A simple interpretation of the metrics and the transformed fluxes in Eqs. (4)–(7) can be given if one considers the integral form of Eq. (2) applied to a quadrilateral cell of the computational grid in physical space with volume Ω

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_\Omega \boldsymbol{Q}\,\mathrm{d}\Omega + \int_{\partial\Omega} \boldsymbol{\mathcal{F}} \cdot \mathrm{d}\boldsymbol{\mathcal{A}} = \int_\Omega \boldsymbol{S}\,\mathrm{d}\Omega, \tag{20}$$

where $\boldsymbol{\mathcal{F}} = (\boldsymbol{F}, \boldsymbol{G})$ is the flux-density tensor. By analogy with Eq. (3), one then finds that the vectors

$$J^{-1}\boldsymbol{\nabla}\xi = J^{-1}\begin{pmatrix}\xi_x \\ \xi_y\end{pmatrix}, \quad J^{-1}\boldsymbol{\nabla}\eta = J^{-1}\begin{pmatrix}\eta_x \\ \eta_y\end{pmatrix}, \tag{21}$$

are the directed areas $\mathrm{d}\boldsymbol{\mathcal{A}}$ of cell interfaces belonging to the ξ = const. and η = const. coordinate lines, respectively, and that the transformed fluxes are simply the normal fluxes through the corresponding cell face, multiplied by the area $|\mathrm{d}\boldsymbol{\mathcal{A}}|$ of that face. Moreover, the Jacobian determinant J^{-1} is identical to the cell volume, i.e.

$$\Omega = J^{-1}. \tag{22}$$

2.4. Rationale for the transformation

The above transformation and the strong conservation form of the equations have several particular advantages for use in a multigrid-based solver because they allow for generality, accuracy, and a rapid convergence of the solution algorithm, and at the same time enable a tight integration and simplification of several algorithmic components.

First, the discretized equations are naturally put into conservation form, which is essential to accurately capture shocks and other discontinuities. Second, geometry issues are treated in a generalized, transparent manner. There is thus no need to clutter a computer code with if-statements for different geometries belonging to different problems.

Third, because the equations are solved in the computational space, which conforms to the boundaries, the imposition of accurate boundary conditions is quite straightforward. This is especially important for a multigrid code, since badly conforming boundaries on coarse meshes are avoided by construction. If this were not the case, a decrease in the convergence speed due to inaccurate coarse-grid correction could result, as has been observed with finite volume discretizations in the original non-boundary-fitted (x, y) space (cf. Wesseling & Oosterlee 2001, and the references cited therein).

Fourth, the mapping makes it possible to *always* employ a *uniform* mesh in computational space, even if the grid in physical space is highly non-uniform. Spatial discretization is then performed in the computational space, hence enormously simplified. Some popular schemes that have been formulated for uniform grids (as, e.g., the Weighted Essentially Non-Oscillatory, or WENO schemes, see Shu 1997) can be used, whose extension to non-uniform meshes would otherwise be rather complicated. Similarly, all multigrid transfer operators need to be coded for uniform rectangular grids only.

And fifth, the capability to use non-orthogonal meshes is vital for an extension of the solution algorithm to block-structured, composite grids. These are required for problems that are difficult to solve accurately on a single grid due to either a complex geometry, or the presence of coordinate singularities in popular standard meshes, such as those based on spherical coordinates. The polar singularities of the latter pose significant difficulties for the solution of some important astrophysical flow problems (see e.g. Kifonidis et al. 2006; Gawryszczak et al. 2010), and the use of "cubed sphere" composite grids (Ronchi et al. 1996, and the references therein) is expected to improve the solution accuracy substantially in these cases.

Finally, there are also two disadvantages of using the strong conservation form of the governing equations, which are both connected to the appearance of the metric derivatives in the transformed fluxes $\hat{\boldsymbol{F}}$ and $\hat{\boldsymbol{G}}$ (cf. Eqs. (5) and (6)). The first, and more severe, of these drawbacks is that the evaluation of the numerical versions of these fluxes becomes non-trivial, as particular care has to be taken in order not to introduce spurious sources into the discretization owing to the metrics. We deal with this problem in Sect. 3.2. The second (rather minor) drawback is that, to allow for a rapid flux evaluation, some memory needs to be reserved in order to precompute and store the metrics on the grid.

3. Spatial discretization

To disentangle spatial from temporal discretization issues, we discretize the governing PDEs with the method of lines (e.g., Toro 1997). This allows for substantial flexibility in the choice of the spatial and temporal discretization schemes (see also Sect. 4), and enables us to employ the multigrid techniques discussed in Sect. 5, that lead to a significantly faster calculation of the implicit solution than would otherwise be possible.

3.1. Cell-centered, finite volume, semi-discretization

After discrete evaluation of the integral terms, Eq. (20) assumes the semi-discrete form

$$\frac{\mathrm{d}\hat{\boldsymbol{Q}}_{i,j}}{\mathrm{d}t} + \left(\hat{\boldsymbol{F}}_{i+\frac{1}{2},j} - \hat{\boldsymbol{F}}_{i-\frac{1}{2},j} + \hat{\boldsymbol{G}}_{i,j+\frac{1}{2}} - \hat{\boldsymbol{G}}_{i,j-\frac{1}{2}}\right) = \hat{\boldsymbol{S}}_{i,j}. \tag{23}$$

Here $\hat{\boldsymbol{Q}}_{i,j}$ and $\hat{\boldsymbol{S}}_{i,j}$ are to be understood as the products of the cell volume, $\Omega_{i,j}$, and the cell averages $\boldsymbol{Q}_{i,j}$ and $\boldsymbol{S}_{i,j}(\boldsymbol{Q}_{i,j})$, respectively, in zone (i, j). The transformed fluxes are evaluated in the midst of zone interfaces, hence denoted by a half-integer index in one and an integer index in the other dimension. In writing out Eq. (23), we have assumed that by construction

$$\Delta\xi = \xi_{i+\frac{1}{2},j} - \xi_{i-\frac{1}{2},j} = 1, \tag{24}$$

$$\Delta\eta = \eta_{i,j+\frac{1}{2}} - \eta_{i,j-\frac{1}{2}} = 1. \tag{25}$$

In what follows, we assume, in addition, that the mesh does not change with time. Equation (23) can then be rewritten as

$$\frac{\mathrm{d}\boldsymbol{Q}_{i,j}}{\mathrm{d}t} + \boldsymbol{R}(\boldsymbol{Q}_{i,j}) = 0, \tag{26}$$

with the spatial residual of all physical terms defined as

$$\boldsymbol{R}(\boldsymbol{Q}_{i,j}) = \frac{\hat{\boldsymbol{F}}_{i+\frac{1}{2},j} - \hat{\boldsymbol{F}}_{i-\frac{1}{2},j} + \hat{\boldsymbol{G}}_{i,j+\frac{1}{2}} - \hat{\boldsymbol{G}}_{i,j-\frac{1}{2}}}{\Omega_{i,j}} - \boldsymbol{S}_{i,j}(\boldsymbol{Q}_{i,j}). \tag{27}$$

We note that the *temporal integration* in Eqs. (26), (27) is performed in an unsplit, i.e. multidimensional, manner, even though the single fluxes may be actually computed in a 1D fashion. This is crucial in allowing for the use of large time steps, and for an accurate calculation and exact preservation of steady states. We can thus maintain up to machine precision the multidimensional balance of the fluxes (and the source term), in case the time step tends to infinity, and the temporal derivative vanishes. We also note that the transformed state, $\hat{\boldsymbol{Q}}_{i,j}$, no longer appears in the equations. It is therefore the Cartesian state vector, $\boldsymbol{Q}_{i,j}$, that is typically stored in a computer code implementation.

3.2. Discretization of the metrics

As we noted in Sect. 2.4, the appearance of the metric derivatives in Eqs. (5) and (6) leads to complications in calculating the numerical fluxes that enter Eq. (27). It should first be noted that ξ_x, ξ_y, η_x, and η_y are not needed themselves since they are preceded by a factor J^{-1} in Eqs. (5) and (6). It is actually the components of the directed areas $J^{-1}\boldsymbol{\nabla}\xi$ and $J^{-1}\boldsymbol{\nabla}\eta$ (or, equivalently, the components x_ξ, y_ξ, x_η, and y_η of the covariant base vectors, see Eqs. (8)–(11)) that need to be evaluated to calculate $\hat{\boldsymbol{F}}$ and $\hat{\boldsymbol{G}}$.

In general, the coordinate transformation from (ξ, η) to (x, y) space will not be known other than through the given (x, y) coordinates of the vertices of some grid[3] on which we would like to solve the equations. The covariant metrics thus need to be evaluated numerically. However, even in cases where the transformation is known analytically, these metrics should *always* be calculated by *appropriate* finite differences. Both analytic derivatives,

[3] Which might have been generated externally.

and inappropriate finite differencing of the metrics, are known to introduce spurious source terms into the (semi) discrete Eq. (26). This was emphasized by Thompson & Warsi (1982, pp. 84–88), and Thompson et al. (1985, pp. 158–166), who illustrated the problem in great detail by considering the case of a uniform flow.

Following the line of their discussion, it can be shown that if these spurious sources are to be avoided, the metrics (which are evaluated in the midst of zone faces) need to fulfill the constraints

$$\left[(y_\eta)_{i+\frac{1}{2},j} - (y_\eta)_{i-\frac{1}{2},j}\right] - \left[(y_\xi)_{i,j+\frac{1}{2}} - (y_\xi)_{i,j-\frac{1}{2}}\right] = 0 \tag{28}$$

$$\left[(x_\xi)_{i,j+\frac{1}{2}} - (x_\xi)_{i,j-\frac{1}{2}}\right] - \left[(x_\eta)_{i+\frac{1}{2},j} - (x_\eta)_{i-\frac{1}{2},j}\right] = 0. \tag{29}$$

Equations (28) and (29), which are also called the *metric identities*, are the numerical equivalent of the analytic relations $y_{\eta\xi} = y_{\xi\eta}$ and $x_{\xi\eta} = x_{\eta\xi}$. These constraints need to be enforced by appropriate construction of the discrete scheme. In this respect, they are akin to the well-known $\boldsymbol{\nabla} \cdot \boldsymbol{B} = 0$ constraint of MHD.

To satisfy the metric identities, we use the second-order accurate finite differences

$$(y_\eta)_{i+\frac{1}{2},j} = y_{i+\frac{1}{2},j+\frac{1}{2}} - y_{i+\frac{1}{2},j-\frac{1}{2}} \tag{30}$$

$$(y_\eta)_{i-\frac{1}{2},j} = y_{i-\frac{1}{2},j+\frac{1}{2}} - y_{i-\frac{1}{2},j-\frac{1}{2}} \tag{31}$$

$$(y_\xi)_{i,j+\frac{1}{2}} = y_{i+\frac{1}{2},j+\frac{1}{2}} - y_{i-\frac{1}{2},j+\frac{1}{2}} \tag{32}$$

$$(y_\xi)_{i,j-\frac{1}{2}} = y_{i+\frac{1}{2},j-\frac{1}{2}} - y_{i-\frac{1}{2},j-\frac{1}{2}}, \tag{33}$$

where half-integer indices in both dimensions denote coordinates of zone vertices, and equivalent equations hold for the derivatives of x.

Avoiding spurious sources due to the metrics becomes significantly more involved in three spatial dimensions, and for varying (e.g. moving) meshes. Appropriate discretization formulae for the three-dimensional case were given, e.g., by Tweedt et al. (1997).

3.3. Roe upwind flux

When the metrics are known, the numerical fluxes occurring in Eq. (27) are computed "dimension by dimension" by successive 1D sweeps over the grid. To obtain a high-resolution scheme that is able to accurately capture shocks and other flow discontinuities, suitable left and right states, $\boldsymbol{Q}^{\mathrm{L}}$ and $\boldsymbol{Q}^{\mathrm{R}}$, are first interpolated at the cell faces within each sweep. This defines 1D Riemann problems, which are subsequently solved with the approximate solver of Roe (1981).

For the moment, we assume that $\boldsymbol{Q}^{\mathrm{L}}$ and $\boldsymbol{Q}^{\mathrm{R}}$ are known (their actual calculation out of the zone averages $\boldsymbol{Q}_{i,j}$ is described in more detail in Sect. 3.4 and Appendix A). A closed-form expression for the Roe flux is then given at, e.g., cell face $i + \frac{1}{2}$ by

$$\hat{\boldsymbol{F}}_{i+\frac{1}{2}} = \frac{1}{2}\left(\hat{\boldsymbol{F}}\left(\boldsymbol{Q}^{\mathrm{L}}\right) + \hat{\boldsymbol{F}}\left(\boldsymbol{Q}^{\mathrm{R}}\right)\right)_{i+\frac{1}{2}} - \frac{1}{2}\left(\left|\hat{\boldsymbol{A}}\right|\left(\boldsymbol{Q}^{\mathrm{R}} - \boldsymbol{Q}^{\mathrm{L}}\right)\right)_{i+\frac{1}{2}}, \tag{34}$$

where the first term is a face-centered arithmetic average of transformed fluxes, to which a stabilizing matrix dissipation term is added (note that the second spatial index has been dropped in this equation since we consider only the 1D sweep in the ξ-direction).

The matrix $\hat{\boldsymbol{A}}$ is the Jacobian corresponding to flux $\hat{\boldsymbol{F}}$

$$\hat{\boldsymbol{A}} = \left.\frac{\partial \hat{\boldsymbol{F}}(\boldsymbol{Q})}{\partial \boldsymbol{Q}}\right|_{\boldsymbol{Q}=\boldsymbol{Q}^{\mathrm{Roe}}} \tag{35}$$

evaluated at the Roe-averaged state

$$\boldsymbol{Q}^{\mathrm{Roe}} = \boldsymbol{Q}^{\mathrm{Roe}}\left(\boldsymbol{Q}^{\mathrm{L}}, \boldsymbol{Q}^{\mathrm{R}}\right), \tag{36}$$

such that the relation

$$\hat{\boldsymbol{F}}\left(\boldsymbol{Q}^{\mathrm{R}}\right) - \hat{\boldsymbol{F}}\left(\boldsymbol{Q}^{\mathrm{L}}\right) = \hat{\boldsymbol{A}}\left(\boldsymbol{Q}^{\mathrm{R}} - \boldsymbol{Q}^{\mathrm{L}}\right) \tag{37}$$

is satisfied exactly (see Roe 1981). $\hat{\boldsymbol{A}}$ is diagonalized by the transformation

$$\hat{\boldsymbol{A}} = \boldsymbol{T}\boldsymbol{\Lambda}\boldsymbol{T}^{-1}, \tag{38}$$

where $\boldsymbol{T}$ ($\boldsymbol{T}^{-1}$) is the matrix whose columns (rows) are the right (left) eigenvectors of $\hat{\boldsymbol{A}}$, and $\boldsymbol{\Lambda}$ is a diagonal matrix whose entries are the eigenvalues of $\hat{\boldsymbol{A}}$. The dissipation matrix $\left|\hat{\boldsymbol{A}}\right|$ appearing in Eq. (34) is obtained by replacing $\boldsymbol{\Lambda}$ in Eq. (38) by the diagonal matrix $|\boldsymbol{\Lambda}|$ that is made up of the absolute eigenvalues of $\hat{\boldsymbol{A}}$

$$\left|\hat{\boldsymbol{A}}\right| = \boldsymbol{T}\,|\boldsymbol{\Lambda}|\,\boldsymbol{T}^{-1} = \boldsymbol{T}\,\mathrm{diag}(|\lambda_1|, |\lambda_2|, \ldots)\,\boldsymbol{T}^{-1}. \tag{39}$$

The calculation of the flux-averaged and dissipation terms in Eq. (34) (and in the equivalent equation for the sweep in the η-direction) is one of the few spots in the algorithm where the specific form of the hyperbolic system to be solved actually enters. It requires the complete characteristic decomposition of the hyperbolic system, i.e. the eigenvalues as well as the left and right eigenvectors of the Jacobians $\hat{\boldsymbol{A}}$ and $\hat{\boldsymbol{B}}$, corresponding to the fluxes $\hat{\boldsymbol{F}}$ and $\hat{\boldsymbol{G}}$, have to be known (we note already here that flux Jacobians also appear in the implicit stages, whereas their spectral radii are required for the local pseudo-time stepping; we discuss both cases in Sect. 5).

In the case of the Euler equations, the flux-averaged terms are easily evaluated using Eqs. (17)–(19) once the metrics have been calculated by Eqs. (30)–(33) and (8)–(11). To efficiently evaluate the dissipation terms, we use the special forms of $\left|\hat{\boldsymbol{A}}\right|$ and $\left|\hat{\boldsymbol{B}}\right|$ for the Euler equations that are given in Swanson & Turkel (1992, 1997). Their formulae are not repeated here since at least the former reference should be widely accessible.

3.4. Interpolation of interface values

The present method requires, in general, the calculation of fluxes (more precisely, flux Jacobians) with both second- *and* first-order spatial accuracy (cf. Sect. 5.3.3). First-order accuracy is easily achieved by choosing the cell face values $\boldsymbol{Q}^{\mathrm{L}}$ and $\boldsymbol{Q}^{\mathrm{R}}$ as

$$\boldsymbol{Q}^{\mathrm{L}}_{i+\frac{1}{2}} = \boldsymbol{Q}_i, \quad \boldsymbol{Q}^{\mathrm{R}}_{i+\frac{1}{2}} = \boldsymbol{Q}_{i+1}. \tag{40}$$

Second-order spatial accuracy is obtained by interpolating $\boldsymbol{Q}^{\mathrm{L}}$ and $\boldsymbol{Q}^{\mathrm{R}}$ from the zone averages in neighboring zones using the SLIP scheme proposed by Jameson (1995b). Explicit formulae for the calculation of $\boldsymbol{Q}^{\mathrm{L}}$ and $\boldsymbol{Q}^{\mathrm{R}}$ with the SLIP scheme are given in Appendix A.

The SLIP interpolation gives a robust, monotonic, slope-limited scheme that captures discontinuities that may develop in the solution without (Gibbs-) oscillations, overshoots, or undershoots. In multidimensional applications, SLIP-based discretization schemes provide an accuracy similar to that of the finite volume WENO3 scheme of Shu (1997), or the second-order total variation diminishing (TVD) MUSCL/KAPPA schemes of Van Leer (1985). However, compared to these methods, the SLIP scheme has two important advantages, which lead to faster and more robust convergence in a time-implicit code that is based on

a multigrid framework. First, its particular discretization stencil leads to exceptionally low amplification factors for the high-frequency error modes in Fourier space, especially if the scheme is used together with the multistage implicit smoothers that we describe in Sect. 5.3.3. We verified that this is the case by performing a local Fourier analysis (cf. Sect. 6), which gave us distinctly better smoothing factors for SLIP-based discretizations compared to the other two schemes. And second, the slope limiter that is employed by the SLIP scheme interferes only weakly with the implicit convergence process. Convergence problems due to TVD slope limiters are common with implicit codes (see Jameson 1995a), and in fact restrict the class of suitable spatial discretization schemes quite severely.

Rather extensive numerical experiments, in which we used the multigrid methods described in this paper together with SLIP, MUSCL/KAPPA, and WENO3 spatial discretization, confirmed this behavior. The SLIP-based multigrid variants always converged the fastest, clearly showed the most robust convergence behavior, and were insensitive to becoming trapped in limit cycles, in contrast, especially, to their MUSCL-based relatives.

3.5. Calculation of the zone volume

To evaluate the spatial residual (cf. Eq. (27)), we still need to calculate the zone volume, $\Omega_{i,j}$, or, equivalently, the local value of the Jacobian determinant J^{-1}. We could compute J^{-1} using Eq. (12) and the finite differences given in Sect. 3.2, but a more accurate representation is obtained by calculating the volume of the quadrilateral cell directly, which in the 2D planar case considered for most of the test calculations in this paper leads to (cf. Toro 1997)

$$\Omega_{i,j} = J^{-1}_{i,j} = \frac{1}{2}\left|A_1 - A_2\right|, \tag{41}$$

with

$$A_1 = \left(x_{i-\frac{1}{2},j+\frac{1}{2}} - x_{i+\frac{1}{2},j-\frac{1}{2}}\right) \times \left(y_{i-\frac{1}{2},j-\frac{1}{2}} - y_{i+\frac{1}{2},j+\frac{1}{2}}\right)$$
$$A_2 = \left(y_{i-\frac{1}{2},j+\frac{1}{2}} - y_{i+\frac{1}{2},j-\frac{1}{2}}\right) \times \left(x_{i-\frac{1}{2},j-\frac{1}{2}} - x_{i+\frac{1}{2},j+\frac{1}{2}}\right).$$

4. Temporal discretization

According to Eq. (26), spatial discretization has resulted in a large system of stiff ordinary differential equations of the form

$$\frac{\mathrm{d}\boldsymbol{Q}}{\mathrm{d}t} + \boldsymbol{R}(\boldsymbol{Q}) = 0, \tag{42}$$

where we have dropped the zone indices since $\boldsymbol{Q}$ and $\boldsymbol{R}$ are now to be understood as state and spatial residual vectors across the entire grid. Equation (42) can be solved by a suitable integrator for stiff ordinary differential equations (ODEs). This entails discretizing it in time with an implicit scheme, and solving the resulting non-linear algebraic systems.

There are in principle two classes of ODE solvers from which a suitable scheme can be selected: multistep/single-stage, and single-step/multistage solvers. Among the former are the popular backward difference formulae (BDFe). Among the latter are the less-known implicit Runge-Kutta (RK) schemes. Good discussions of the advantages and disadvantages of BDF versus implicit RK schemes for the solution of fluid dynamics problems are given in Bijl et al. (2002) and Jothiprasad et al. (2003).

A crucial consideration in the choice of a temporal integrator, is the extent of its domain of stability. If the scheme is to be applied to problems of considerable stiffness, it should be unconditionally stable. In this case, the time step size will be determined solely by accuracy considerations (making an estimate of the scheme's truncation error necessary, see Sect. 4.2). The L-stable (cf. Hairer & Wanner 1991) ESDIRK schemes that we describe below are unconditionally stable for linear (or quasi-linear, e.g. smooth flow) problems. However, since these schemes are of higher than first-order accuracy, and make use of linear coefficients, they are subject to Godunov's theorem, i.e. they can lead to non-monotone solutions, even if the employed spatial discretization fulfills a TVD property. This is typically the case when the schemes are used with very large time steps. If the problem to be solved then happens to be highly non-linear, and to contain, e.g., shocks, the lack of an unconditional TVD property will typically lead to a non-linear instability. A non-linear stability constraint will then have to be imposed on the time step (cf. Sect. 7.4). It must also be noted that these stability conditions are not yet sufficient for the *complete* implicit scheme to be numerically stable. For this, the actual solver – which in our case is multigrid – must be stable too. We return to this last point in Sect. 5.

4.1. Singly diagonally implicit Runge-Kutta schemes with an explicit first stage (ESDIRK)

We consider first the class of singly diagonally implicit Runge-Kutta (SDIRK) schemes (see Hairer & Wanner 1991, for an overview) for which the so-called Butcher coefficient matrix a_{kl} is lower triangular, and the diagonal coefficients, $a_{kk} = \beta$, are all the same. A complete time step Δt in the case of such a scheme, which takes the state $\boldsymbol{Q}$ from time level t^n to t^{n+1} consists of s RK stages. The kth stage applied to Eq. (42) can be written as

$$\boldsymbol{Q}^{(k)} = \boldsymbol{Q}^n - \Delta t \sum_{l=1}^{k} a_{kl}\boldsymbol{R}\left(\boldsymbol{Q}^{(l)}\right), \quad k = 1, \ldots, s \tag{43}$$

where state updates belonging to a specific RK stage have been given superscripts in brackets. We note the appearance of $\boldsymbol{Q}^{(k)}$ in both the left-hand side, and in the sum of residuals on the right-hand side of Eq. (43). This renders the scheme implicit, requiring the solution of a non-linear system *per stage* in order to calculate the staged updates $\boldsymbol{Q}^{(k)}$. After all stages have been completed, the final value of $\boldsymbol{Q}$ at time t^{n+1} is obtained as

$$\boldsymbol{Q}^{n+1} = \boldsymbol{Q}^n - \Delta t \sum_{l=1}^{s} b_l\boldsymbol{R}\left(\boldsymbol{Q}^{(l)}\right), \tag{44}$$

with some known weights b_l.

The ESDIRK schemes on which we focus in the following are *almost* SDIRK methods. They differ only in that they require the coefficient a_{11} to vanish, and the stiffly accurate conditions

$$b_l = a_{sl}, \quad \text{for } l = 1, \ldots, s \tag{45}$$

to be fulfilled, which allow the schemes to attain a stage-order of two, i.e. to make the single stages second-order accurate. An ESDIRK scheme for Eq. (42) can therefore be written as

$$\boldsymbol{Q}^{(1)} = \boldsymbol{Q}^n \tag{46}$$

$$\boldsymbol{Q}^{(k)} = \boldsymbol{Q}^n - \Delta t \sum_{l=1}^{k} a_{kl}\boldsymbol{R}\left(\boldsymbol{Q}^{(l)}\right), \quad k = 2, \ldots, s \tag{47}$$

$$\boldsymbol{Q}^{n+1} = \boldsymbol{Q}^{(s)}. \tag{48}$$

The first stage is explicit and moreover trivial, the only things to be done are to initialize $\boldsymbol{Q}^{(1)}$ with the old solution value, $\boldsymbol{Q}^n$, and to calculate the spatial residual $\boldsymbol{R}\left(\boldsymbol{Q}^{(1)}\right)$. Starting only with the second stage a non-linear system needs to be solved per stage (see also Algorithm 1 in Appendix D). Hence, an s-stage ESDIRK scheme requires the solution of only $s-1$ (as opposed to s) non-linear systems. Finally, having completed the calculations for the last stage, one has automatically found $\boldsymbol{Q}^{n+1}$, the solution at the new time level. A separate evaluation of Eq. (44) is thus not required.

The scheme that we henceforth denote ESDIRK23 (where the first digit gives the scheme's order of accuracy, while the second digit gives the number of stages) was identified as an ESDIRK scheme by Hosea & Shampine (1996). It is also known as the TR-BDF2 method because it can be interpreted as a one-step method that is a combination of the trapezoidal rule and the second-order backward difference formula. We list the Butcher coefficients of this scheme in Appendix B.

Higher-order ESDIRK34, and ESDIRK46 schemes were given by Kennedy & Carpenter (2001). These methods were evaluated for the solution of the Navier-Stokes equations by Bijl et al. (2002) and Jothiprasad et al. (2003). For the numerical experiments presented in this paper, we make use of only the ESDIRK23 and ESDIRK46 schemes.

4.2. Time-step control for the ESDIRK schemes

A significant advantage of ESDIRK schemes (and actually most Runge-Kutta methods) is that they include an embedded formula for the calculation of an approximation $\bar{\boldsymbol{Q}}^{n+1}$ with a different order of accuracy than the scheme's main solution approximation $\boldsymbol{Q}^{n+1}$. This is achieved at almost no additional cost using information that has already been computed in the course of the scheme. In analogy to Eq. (44), the embedded formula can be written as

$$\bar{\boldsymbol{Q}}^{n+1} = \boldsymbol{Q}^n - \Delta t \sum_{l=1}^{s} \bar{b}_l \boldsymbol{R}\left(\boldsymbol{Q}^{(l)}\right), \tag{49}$$

with another known set of coefficients $\bar{b}_l$ (see Appendix B). The difference

$$\boldsymbol{e} = \boldsymbol{Q}^{n+1} - \bar{\boldsymbol{Q}}^{n+1} \tag{50}$$

then provides an estimate of the truncation error for the lower order approximation, which can be used, in turn, to estimate the step size. With both the ESDIRK23 and ESDIRK46 schemes that we use in this paper, the embedded formula for the calculation of $\bar{\boldsymbol{Q}}^{n+1}$ is of third-order accuracy. The particular time step controller that we presently employ, and to which $\boldsymbol{e}$ is input, is the controller given by Press et al. (1992) for their Kaps-Rentrop routine. The reader is referred to this original work for details on how a viable step size is determined given both some predefined accuracy requirement on the solution, and the error estimate of Eq. (50), and for the criteria that we employ for accepting a step, or for rejecting and repeating it with a smaller step size.

We note here that the design of optimal step-size controllers for implicit solvers is a highly non-trivial issue that is beyond the scope of this paper, and that further work is required to arrive at an optimal algorithm. For instance, the use of the standard error estimator (Eq. (50)) for stiff problems was criticized by Hosea & Shampine (1996). However, the alternative that these authors suggested requires the calculation and storage of the Jacobian matrix for the complete high-order discretization scheme, which is what we wish to avoid in the first place by employing the FAS algorithm.

5. Multigrid solution of the discrete equations

5.1. Reformulation of the non-linear systems

The non-linear algebraic systems expressed by Eq. (47) are not yet in the form in which they can be conveniently solved by a multigrid method. To obtain this form and simplify the notation, we rewrite Eq. (47) as

$$\mathcal{N}(\boldsymbol{U}) = \boldsymbol{f}. \tag{51}$$

Here, and in all that follows, $\boldsymbol{U}$ is the vector of unknowns to be solved for, while the right hand-side, the so-called "forcing function", $\boldsymbol{f}$, comprises the terms in Eq. (47) that depend only upon known information, but not on the solution, $\boldsymbol{U}$, itself. For the non-linear system to be solved during the kth ESDIRK stage (with $k \geq 2$), we have

$$\boldsymbol{U} = \boldsymbol{Q}^{(k)} \tag{52}$$

and

$$\boldsymbol{f} = C\frac{\boldsymbol{Q}^n}{\Delta t} - C\sum_{l=1}^{k-1} a_{kl}\boldsymbol{R}\left(\boldsymbol{Q}^{(l)}\right), \tag{53}$$

where

$$C = \frac{1}{\beta} = \frac{1}{a_{kk}} \tag{54}$$

is a constant. To evaluate the expression for $\boldsymbol{f}$, the spatial residuals $\boldsymbol{R}\left(\boldsymbol{Q}^{(l)}\right)$ of all previous stages ($l = 1, \ldots, k-1$) need to be stored in memory.

The left-hand side of Eq. (51) consists of the total discretization operator

$$\mathcal{N}(\boldsymbol{U}) = C\frac{\boldsymbol{U}}{\Delta t} + \boldsymbol{R}(\boldsymbol{U}). \tag{55}$$

A crucial quantity for the multigrid solution of Eq. (51) is the total residual function, or defect,

$$\boldsymbol{D}(\boldsymbol{U}) = \mathcal{N}(\boldsymbol{U}) - \boldsymbol{f}, \tag{56}$$

in terms of which Eq. (51) can finally be expressed as

$$\boldsymbol{D}(\boldsymbol{U}) = 0. \tag{57}$$

5.2. Dual time framework

Equation (57) is solved within the non-linear multigrid time-stepping framework proposed by Jameson (1991), in which an artificial pseudo-time derivative, $\mathrm{d}\boldsymbol{U}/\mathrm{d}\tau$, is added to it, to obtain an ordinary differential equation of the form

$$\frac{\mathrm{d}\boldsymbol{U}}{\mathrm{d}\tau} + \boldsymbol{D}(\boldsymbol{U}) = 0. \tag{58}$$

Equation (57) is then regarded as the (pseudo-time) steady-state problem of Eq. (58), that is, the solution of Eq. (57) is computed by simply marching Eq. (58) forward in pseudo-time until the term $\mathrm{d}\boldsymbol{U}/\mathrm{d}\tau$ vanishes, i.e. until a steady state is reached.

In selecting the marching procedure, we again face the problem of choosing between an explicit and implicit scheme. As for the original Eq. (42), the pseudo-time Eq. (58) is stiff itself.

Its introduction may hence not appear to have resulted in any advantage. Yet, there is a crucial difference in solving these two equations: the pseudo-time integration is *not* required to be time-accurate, i.e. the actual path for reaching the pseudo-time steady state is immaterial. The pseudo-time thus has no physical significance. It only serves as a continuous "counter" variable in an iteration process toward the solution of Eq. (57). As long as this process converges, the actual pseudo-time marching scheme can even be inconsistent (cf. Jameson 2003)! This is easily seen by writing out Eq. (58) as

$$\frac{\mathrm{d}\boldsymbol{U}}{\mathrm{d}\tau} + C\frac{\boldsymbol{U}}{\Delta t} + \boldsymbol{R}(\boldsymbol{U}) - \boldsymbol{f}(\Delta t) = 0. \tag{59}$$

We note that this equation contains two time coordinates, the pseudo-time, τ, and the real (or physical) time, t, expressed by the appearance of the ESDIRK discretization terms in physical time, i.e. by the second and fourth term on the left-hand side. Once the first term in this equation vanishes by pseudo-time marching, the solution will be fully consistent with both the ESDIRK temporal discretization, and the employed spatial discretization, where the latter is represented by the third term on the left-hand side. Owing to its use of two time coordinates, the entire solution procedure is also referred to as the dual-time method. We note that within this procedure, Δt is simply a given constant.

The very fact that the pseudo-time iteration process does not require a consistent scheme, allows one to speed up its convergence, by designing special explicit marching schemes (i.e. ODE integrators) which sacrifice time-accuracy for a maximized stability domain, and an optimized damping of high frequency components of the iteration error. It also allows one to employ local pseudo-time steps (see Sect. 5.3.2), which can be viewed as a simple local (and thus parallelizable) preconditioning technique to enhance the performance of the iteration. While it is possible to use such a marching scheme as a standalone iterative solver for Eq. (57), its convergence speed will still be limited by low frequency modes of the iteration error, especially if the physical time step Δt that is used with the ESDIRK discretizations is large.

Given that the marching schemes are easily optimized to yield good damping for high-frequency error modes, it makes more sense to use them as smoothers within a (pseudo) time stepping, FAS multigrid context (see Sect. 5.4.1). In such a setting, the marching algorithm is responsible for annihilating only the high frequency content of the solution's defect. This is achieved by performing a few pseudo-time iterations on an iterand of the solution vector. The effect of these iterations is simply that the defect is smoothed by the numerical dissipation (or artificial viscosity) inherent to the employed spatial discretization scheme. The remaining low-frequency content of the defect is subsequently dealt with by the employment of coarser grids.

As we pointed out in the introduction, the actual efficiency of a multigrid approach depends sensitively on both the quality of the smoother and the employed coarse-grid correction scheme. With the full-coarsening strategy implemented in our code the coarse-grid correction can eliminate only error modes that have long wavelengths, i.e. low frequencies, in *both* coordinate directions (the so-called "low-low modes"). If good multigrid efficiency is to be obtained, all other error components on a given grid level (i.e. high-high, low-high, and high-low modes, see Pierce & Giles 1997, for more details) *must* then be removed by the smoother. For hyperbolic problems, even a (small) specific part of the low-low spectrum, the so-called characteristic error components, needs to be removed by the smoother (Brandt 1984; Thomas et al. 2003).

The damping of low frequency modes is, unfortunately, a weak point of explicit schemes. It can be improved by the introduction of a judicious amount of implicitness into the smoother that serves to ensure some non-local coupling between zones. At the same time, this will improve the error propagation properties of the scheme. We address these points in the framework of multistage-implicit smoothers in Sect. 5.3.3.

5.3. The smoothing algorithm

5.3.1. Explicit multistage smoothing scheme

The basic scheme, with which the defect in Eqs. (57), (58) is smoothed, is an explicit multistage (Runge-Kutta) scheme consisting of K stages. A complete step performed with this scheme is of the form

$$\boldsymbol{U}^{\{0\}} = \boldsymbol{U}^{m} \tag{60}$$

$$\boldsymbol{U}^{\{k\}} = \boldsymbol{U}^{\{0\}} - \alpha_k \Delta\tau\, \boldsymbol{D}\left(\boldsymbol{U}^{\{k-1\}}\right), \quad k = 1, \ldots, K \tag{61}$$

$$\boldsymbol{U}^{m+1} = \boldsymbol{U}^{\{K\}}. \tag{62}$$

Here $\Delta\tau = \tau^{m+1} - \tau^{m}$ is the pseudo-time step, and superscripts m and $m+1$ indicate the old and new pseudo-time level, respectively, while the superscripts in braces denote the stage.

The number of stages, K, and the stage coefficients, α_k, are chosen such that the scheme's stability domain and its high-frequency damping characteristics are maximized. We implemented the five and three-stage schemes with

$$[\alpha_1, \ldots, \alpha_5] = [0.066, 0.16, 0.307, 0.576, 1.0], \tag{63}$$

$$[\alpha_1, \ldots, \alpha_3] = [0.15, 0.4, 1.0], \tag{64}$$

whose coefficient sets were obtained from local Fourier analysis by Dick & Riemslagh (1995) and Van Leer et al. (1989), respectively. In addition, we also implemented the two-stage scheme with

$$[\alpha_1, \alpha_2] = [0.32, 1.0], \tag{65}$$

whose only variable coefficient α_1 was optimized by numerical experiments. If implicit stages, as described in Sect. 5.3.3, are not used, then the five-stage scheme is clearly superior to the three and two-stage schemes in terms of smoothing. If, on the other hand, implicit stages are employed, the smoothing performance of the simpler schemes is on par with the five-stage scheme (see Swanson et al. 2007), which then becomes unattractive due to its larger cost.

As they stand, the multistage schemes given by Eqs. (60)–(62) do not yet provide a robust means of marching the dual-time Eq. (59) forward in pseudo-time. They work well if the ratio of the pseudo-time step to real time step, $\Delta\tau/\Delta t$, is small, but become unstable if this ratio becomes large. The reason for this is that, as $\Delta\tau/\Delta t$ increases from zero, the term $C\boldsymbol{U}/\Delta t$ in Eq. (59) leads to a shift of an explicit scheme's (limited) stability domain in Fourier space relative to the Fourier footprint of the spatial discretization operator. For sufficiently large $\Delta\tau/\Delta t$, some of the eigenvalues of this operator will therefore find themselves lying outside the stability domain, and no longer be damped (see Melson et al. 1993).

To avoid this, Melson et al. treated the $C\boldsymbol{U}/\Delta t$ term implicitly by setting $C\boldsymbol{U}/\Delta t = C\boldsymbol{U}^{\{k\}}/\Delta t$ in the kth stage of the pseudo-time integration and moving this term to the left-hand side of Eq. (61).

While these modifications are recommended if the multistage scheme is to be used as a smoother on its own, they are superfluous if the scheme is to be augmented by implicit smoothing stages, as described below. The implicit stages already render the multistage smoother unconditionally stable, as they include an implicit treatment of all relevant terms.

5.3.2. Local stepping in pseudo-time

A simple idea that improves the smoothing properties of the multistage iteration, is to update every computational cell with its own pseudo-time step. This can be viewed as a simple local (scalar) preconditioner, that improves the damping of the high-frequency error spectrum. Notice that the pseudo-time step of an explicit multistage scheme is restricted by a CFL-limit to the pseudo-time Courant number σ_τ. The use of local pseudo-time steps, allows one to fix σ_τ to be the same in every zone, so that it can be set to its optimal value with respect to smoothing (where the latter depends on the coefficients of the multistage scheme being used). This enhances the smoothing properties of the multistage iteration by clustering the Fourier footprint of the spatial discretization operator inside the stability domain of the multistage scheme in the complex plane (see Van Leer et al. 1989). Equation (61) is therefore actually implemented as

$$\boldsymbol{U}_{i,j}^{\{k\}} = \boldsymbol{U}_{i,j}^{\{0\}} - \alpha_k \Delta\tau_{i,j}\, \boldsymbol{D}_{i,j}\left(\boldsymbol{U}^{\{k-1\}}\right), \tag{66}$$

where the local pseudo-time step is determined from

$$\Delta\tau_{i,j} = \sigma_\tau \frac{\Omega_{i,j}}{\lambda^{\xi}_{i,j} + \lambda^{\eta}_{i,j}}. \tag{67}$$

In this last expression, the velocities $\lambda^{\xi}_{i,j}$ and $\lambda^{\eta}_{i,j}$ are the spectral radii (i.e. the maxima of the moduli of the eigenvalues) of the flux Jacobians $\partial\hat{\boldsymbol{F}}(\boldsymbol{U})/\partial\boldsymbol{U}$ and $\partial\hat{\boldsymbol{G}}(\boldsymbol{U})/\partial\boldsymbol{U}$, respectively, in cell (i, j). Thus, $\Delta\tau_{i,j}$ is the timescale corresponding to the maximum speed with which information can propagate through the mesh cell. For the special case of the Euler equations, we have

$$\lambda^{\xi} = \left|\bar{U}\right| + c\sqrt{(J^{-1}\xi_x)^2 + (J^{-1}\xi_y)^2} \tag{68}$$

and

$$\lambda^{\eta} = \left|\bar{V}\right| + c\sqrt{(J^{-1}\eta_x)^2 + (J^{-1}\eta_y)^2}, \tag{69}$$

where $\bar{U}$ and $\bar{V}$ are given by Eqs. (17) and (18), and c is the local sound speed. We note that for use in Eq. (67), λ^{ξ} and λ^{η} need to be evaluated at zone centers, where discrete values of the metrics are unavailable. To obtain these zone-centered metrics, arithmetic averages of the face-centered metric derivatives calculated in Sect. 3.2 can be used.

The optimal values of the parameter σ_τ can either be taken from Van Leer et al. (1989) or be determined by numerical experiments. Values around $\sigma_\tau \approx 1.5$ are optimal for the explicit five-stage scheme, while somewhat smaller values should be used with the three- and two-stage schemes, owing to their smaller domains of stability. With the implicit schemes described in the next section, there is no stability limit to σ_τ.

5.3.3. Implicit smoothing stages

The smoothing and the stability properties of multistage smoothers are further improved, dramatically, if a specific amount of implicitness is incorporated into them. This needs to be done carefully in order not to sacrifice their simplicity and excellent parallelizability. The approach that we propose here is based on an extension of a method introduced by Rossow (2006, 2007), whose basic idea can be understood as follows.

We consider the update to the kth stage of a multistage scheme (Eq. (61)) in the form

$$\boldsymbol{U}^{\{k\}} = \boldsymbol{U}^{\{0\}} + \alpha_k\, \delta\boldsymbol{U}, \tag{70}$$

where the correction $\delta\boldsymbol{U}$ is defined as

$$\delta\boldsymbol{U} = -\Delta\tau\, \boldsymbol{D}^{\{k-1\}} = -\Delta\tau\, \boldsymbol{D}\left(\boldsymbol{U}^{\{k-1\}}\right). \tag{71}$$

To obtain a good smoother for full-coarsening multigrid, we would like to replace the standard correction term $\delta\boldsymbol{U}$ of the explicit multistage scheme by a modified correction term

$$\overline{\delta\boldsymbol{U}} = \boldsymbol{P}\, \delta\boldsymbol{U}, \tag{72}$$

where $\boldsymbol{P}$ is a matrix that introduces some non-local coupling of zones, so that not only the high-high, but also the high-low, and low-high modes of the defect are efficiently damped.

The matrix $\boldsymbol{P}$ itself is not a priori known, but reasonable choices can be made for its inverse, $\boldsymbol{P}^{-1}$. Low-order, implicit discretizations of the governing equations have proven to be particularly useful in this respect, as they lead to relations of the form

$$\boldsymbol{P}^{-1}\, \overline{\delta\boldsymbol{U}} = \delta\boldsymbol{U}, \tag{73}$$

in which the involved matrices are rather sparse. The resulting multistage-implicit schemes were, moreover, found to work very well for fluid dynamics problems. Rossow (2007) has, for instance, obtained remarkable convergence rates in calculations of stationary flows over airfoils using a first-order accurate implicit discretization of the linearized Euler equations for Eq. (73). Within each stage of the multistage smoother, he first calculated the modified corrections $\overline{\delta\boldsymbol{U}}$ by inverting Eq. (73) (very approximately) with a few symmetric point Gauss-Seidel iterations. The modified corrections were then substituted for the $\delta\boldsymbol{U}$ in Eq. (70) to update $\boldsymbol{U}$ within that stage.

We applied this approach, which Rossow (2006, 2007) and Swanson et al. (2007) formulated for stationary problems, to the full dual-time Eq. (58). The form of Eq. (73) that we employ in this case reads

$$\left(\left(1 + \epsilon\frac{C\Delta\tau_{i,j}}{\Delta t}\right)\boldsymbol{I} + \epsilon\frac{\Delta\tau_{i,j}}{\Omega_{i,j}}\boldsymbol{M}_{i,j} - \epsilon\Delta\tau_{i,j}\left(\frac{\partial\boldsymbol{S}}{\partial\boldsymbol{U}}\right)_{i,j}\right)\overline{\delta\boldsymbol{U}}_{i,j} = - \Delta\tau_{i,j}\boldsymbol{D}_{i,j}^{\{k-1\}} - \epsilon\frac{\Delta\tau_{i,j}}{\Omega_{i,j}}\boldsymbol{L}_{i,j}. \tag{74}$$

Here $\boldsymbol{I}$ is an (appropriately dimensioned) identity matrix, and the matrices

$$\boldsymbol{M}_{i,j} = \hat{\boldsymbol{A}}^{+}_{i-\frac{1}{2},j} - \hat{\boldsymbol{A}}^{-}_{i+\frac{1}{2},j} + \hat{\boldsymbol{B}}^{+}_{i,j-\frac{1}{2}} - \hat{\boldsymbol{B}}^{-}_{i,j+\frac{1}{2}} \tag{75}$$

and

$$\boldsymbol{L}_{i,j} = \hat{\boldsymbol{A}}^{-}_{i+\frac{1}{2},j}\overline{\delta\boldsymbol{U}}_{i+1,j} - \hat{\boldsymbol{A}}^{+}_{i-\frac{1}{2},j}\overline{\delta\boldsymbol{U}}_{i-1,j} + \hat{\boldsymbol{B}}^{-}_{i,j+\frac{1}{2}}\overline{\delta\boldsymbol{U}}_{i,j+1} - \hat{\boldsymbol{B}}^{+}_{i,j-\frac{1}{2}}\overline{\delta\boldsymbol{U}}_{i,j-1} \tag{76}$$

are comprised of (interface-centered) local flux Jacobians. They reflect the five-point stencil of the first-order spatial operator that was used to derive Eq. (74) (cf. Rossow 2007, for details). The

factor ϵ was introduced by Swanson et al. (2007) as a weighting factor for the degree of implicitness of the pseudo-time discretization. It serves to optimize the smoothing properties of the scheme, and its optimal value is typically determined by a local Fourier analysis.

The flux Jacobians that occur in the equations given above are defined as

$$\hat{\boldsymbol{A}}^{+} = \frac{1}{2}\left(\hat{\boldsymbol{A}} + \left|\hat{\boldsymbol{A}}\right|\right), \qquad \hat{\boldsymbol{A}}^{-} = \frac{1}{2}\left(\hat{\boldsymbol{A}} - \left|\hat{\boldsymbol{A}}\right|\right), \tag{77}$$

$$\hat{\boldsymbol{B}}^{+} = \frac{1}{2}\left(\hat{\boldsymbol{B}} + \left|\hat{\boldsymbol{B}}\right|\right), \qquad \hat{\boldsymbol{B}}^{-} = \frac{1}{2}\left(\hat{\boldsymbol{B}} - \left|\hat{\boldsymbol{B}}\right|\right), \tag{78}$$

where superscripts + and − indicate flux Jacobians associated with positive and negative eigenvalues (i.e. wave speeds), respectively. All of these Jacobians, which make up the actual implicit operator, are evaluated using the formulae of Swanson & Turkel (1992) and employing Roe-averaging of left and right states, $\boldsymbol{Q}^{\mathrm{L}}$ and $\boldsymbol{Q}^{\mathrm{R}}$, that correspond to *first-order* spatial accuracy. The defect, $\boldsymbol{D}$, on the other hand, is calculated with full second-order accuracy in space. This double discretization approach allows one to use high-order accuracy in the actual discretization without having to deal with the stability problems encountered in also using the high-order spatial operator for smoothing (Brandt 2001).

We note that the terms containing $\boldsymbol{M}$ and $\boldsymbol{L}$ already appeared in Rossow (2007) and Swanson et al. (2007) but in a different notation that should not be confused with ours. The terms with the factors $C/\Delta t$ and $\partial \boldsymbol{S}/\partial \boldsymbol{U}$ are, however, new. They describe the fully implicit treatment of the dual-time term $\boldsymbol{C}\boldsymbol{U}/\Delta t$, and the physical source term $\boldsymbol{S}$. The term $\partial \boldsymbol{S}/\partial \boldsymbol{U}$ is the Jacobian resulting from the linearization of $\boldsymbol{S}$ in zone (i, j).

To approximately solve Eq. (74), the coefficients of the $\overline{\delta \boldsymbol{U}}_{i,j}$ on the left-hand side (which are 4×4 matrices in the case of the 2D Euler equations) are first lower-upper (LU) decomposed. Subsequently, a point Gauss-Seidel iteration is performed on the $\overline{\delta \boldsymbol{U}}_{i,j}$. Within this procedure iterands of the $\overline{\delta \boldsymbol{U}}_{i,j}$ are calculated by back-substituting with the right-hand side of Eq. (74), while sweeping through the grid points in a Gauss-Seidel manner and keeping the LU-factorized coefficient matrices "frozen". The iteration is carried out with the initial and boundary conditions $\overline{\delta \boldsymbol{U}} = 0$.

The ordering of the grid points in the Gauss-Seidel scheme can be any, e.g. lexicographic, symmetric, or red-black. A symmetric Gauss-Seidel (SGS) ordering, as proposed by Rossow (2007) and Swanson et al. (2007), is the most efficient for serial machines. Two SGS iterations are typically sufficient to result in an efficient smoother. In very good agreement with the results obtained by C. Swanson (priv. comm.), we find that local Fourier analysis gives smoothing factors $\mu \sim 0.3$ for the three-stage implicit scheme with the coefficients of Eq. (64), if the latter scheme is applied with an $\epsilon = 0.7$ to steady-state problems (where $C = 0$).

We note that the smoothing factor (which is defined as the maximum amplification factor of the high-frequency error modes in Fourier space), is a predictor for the ideal convergence rate of a multigrid method (Brandt 1984). Assuming an ideal coarse grid correction, the asymptotic convergence rate of multigrid can be calculated from the smoothing factor as

$$\varrho_{\mathrm{MG}} = \mu^{(n_{\mathrm{pre}}+n_{\mathrm{post}})}, \tag{79}$$

where n_{pre} and n_{post} are the number of pre- and post-smoothing steps, respectively (see Sect. 5.4.1 and Appendix D).

It is a distinctive feature and particular advantage of the present smoother that its smoothing factor is quite insensitive as to whether SGS, or red-black Gauss-Seidel (RBGS) iteration is employed in the implicit stages (see Roberts & Swanson 2005, and our results in Sect. 7.1). This is crucial for an efficient parallel implementation of the method on distributed memory machines. With SGS, the algorithm could only be vectorized along diagonals. With RBGS, on the other hand, the method is[4] both efficiently vectorizable, and parallelizable using domain decomposition techniques.

5.4. The multigrid scheme

5.4.1. Basic concepts

The basic idea of the FAS (pseudo) time-stepping multigrid scheme, that we use here, is to remove high frequency modes of the defect by sweeping a few times over a fine grid with our nonlinear smoothing algorithm (this is known as pre-smoothing, see also Hackbusch 1985; Wesseling 1990, and our flow chart of the FAS algorithm presented in Appendix D). Low frequency modes of the defect, however, are hardly affected (hence cannot be removed) by smoothing on fine grids. These low frequency modes are transferred (restricted) to coarser grids (along with the full approximation of the solution, cf. Appendix D) in what constitutes a downward leg of either a V, W, or F-multigrid-cycle (see Fig. 2). On the coarser grids, the formerly low-frequency modes become part of the high frequency spectrum, and can thus be damped by the smoother. Equally importantly, however, the coarse grids also speed up the propagation of low-frequency error modes off the computational domain. On sufficiently coarse grids, the CFL stability condition for explicit marching schemes is not restrictive, hence information can propagate over the entire domain in only a few pseudo time steps.

To correct the solution on the fine grids, correction terms obtained from the solutions on coarser grids are interpolated back (prolongated) to the finer grids in the upward leg(s) of a cycle. However, this coarse-grid correction typically does not completely annihilate the fine grid error within a single multigrid cycle, because we do not solve the coarse grid problems exactly, and because the transfer operators inevitably introduce both some high and low frequency errors into the fine grid solution. The high frequency content of the defect introduced by interpolation can, again, be diminished by a few post-smoothing passes with the smoothing algorithm. However, the possible presence of remaining low frequency errors requires, in general, that multiple multigrid cycles be performed in an iterative fashion, in order to meet some predefined accuracy requirement of the solution.

5.4.2. Coarse grid generation

The coarse grids are generated using "full" or "standard" coarsening. This means that, starting from the finest resolution level, every second coordinate line is deleted in *both* mesh dimensions. Coarse cells are then obtained from the unions of four cells of the overlying finer mesh. A coarse cell's center therefore does not coincide with any cell center on the finer mesh (see, e.g., Mohr & Wienands 2004). These cell-centered multigrid approaches are natural, and easy to implement for finite volume discretizations. Owing to their use of a cell-centered location of all variables they are also straightforward to incorporate into all adaptive mesh refinement codes presently in use by the astrophysics community.

[4] For the employed five-point stencil.

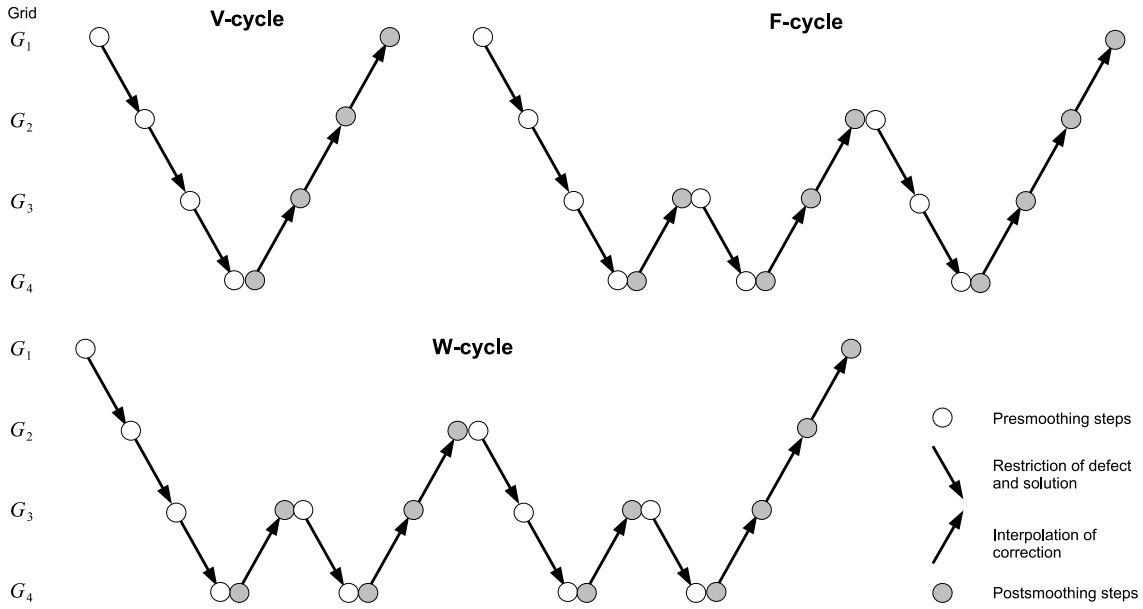

Fig. 2. Different multigrid cycle types, shown here on a grid hierarchy with four levels of mesh resolution. G_1 is the finest grid, while G_4 is the coarsest. Restriction and prolongation are indicated by downward/upward facing arrows, while pre- and post-smoothing are represented as white/gray circles, respectively.

The coarsening process is terminated before the coarsest level grid ends up containing fewer interior zones, in any coordinate direction, than the combined widths of the boundary layers used to prescribe the boundary conditions in that direction (cf. Sect. 5.4.3). For instance, with a two-zone wide boundary layer at each edge, the coarsening is stopped once any of the coordinate directions is left with four interior zones.

The coarsening is done in both physical and computational space. Hence, and similar to the finest resolution level, a one-to-one mapping between the physical mesh, and the coordinates of a uniform rectangular computational mesh with zone width $\Delta\xi = \Delta\eta = 1$, exists on every coarsened level. In an actual code, it is only necessary to calculate the metric coefficients of these mappings and to store them along with the zone volumes on all levels. These are essentially the only grid-related quantities required to define the problems that need to be solved on the coarse grids.

5.4.3. Coarse grid discretization and treatment of boundaries

The FAS algorithm requires the evaluation of the spatial residual operator, $\boldsymbol{R}(\boldsymbol{U})$, not only on the fine but also the coarse grids (cf. Appendix D). These coarse grid operators are obtained by rediscretizing the governing PDEs on the coarse meshes in a manner that is completely analogous to the discretization on the finest mesh. Numerical experiments indicate that the best convergence rates are obtained if the spatial discretization is also monotonic on the coarse levels, and if it is of at least the same order of accuracy as that on the finest grid. Hence, we use the full SLIP scheme on all levels. This is in accordance with results obtained by Rossow (priv. comm.), who found that employment of the second-order scheme on all levels is important in order to achieve mesh-independent convergence rates.

The boundary conditions on coarse levels are also set up using the same method as on the finest grid. This is done by prescribing the physical boundary conditions of the problem in ghost cells, which are situated exterior to the computational domain, and updated at every stage of the multistage smoother. Since the 1D SLIP interpolation scheme has a five-point stencil, a two-zone wide boundary layer is required at each boundary.

5.4.4. Transfer operators

Our description of the multigrid scheme is concluded with the specification of the restriction and prolongation operators. For hyperbolic problems, these operators should satisfy the condition

$$m_{\mathcal{P}} + m_{\mathcal{R}} \geq m_{\mathrm{PDE}} + p \tag{80}$$

(Yavneh 1998). Here $m_{\mathcal{P}}$ is the order of accuracy of the prolongation operator, $m_{\mathcal{R}}$ is the order of the operator with which the defect is restricted to coarse grids, m_{PDE} is the order of the highest derivative in the governing PDEs, and p is the order of accuracy of the (coarse-grid) discretization scheme. The Euler equations are a first-order hyperbolic system, hence $m_{\mathrm{PDE}} = 1$, and since for the SLIP scheme $p = 2$, we obtain

$$m_{\mathcal{P}} + m_{\mathcal{R}} \geq 3. \tag{81}$$

Among the several possibilities that we tested for satisfying Eq. (81), the following was found to be a reasonably efficient choice. The coarse grid correction is transferred from coarse to fine grids using a simple constant interpolation, so that every fine grid cell contained within an underlying coarse cell, receives the same correction from that coarse cell. The corresponding prolongation operator is therefore given in stencil notation (cf. Wesseling 1990; Trottenberg et al. 2001) by

$$\mathcal{P} = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}. \tag{82}$$

It is only first-order accurate ($m_{\mathcal{P}} = 1$) but has a very compact stencil that does not need any adjustments near boundaries. This avoids the necessity for extrapolation, which would amplify the defect near boundaries.

The solution and the defect are both transferred from fine to coarse grids using the same restriction operator. This operator

is the adjoint of the linear interpolation in triangles described in Wesseling (1990). As such, it is second-order accurate ($m_{\mathcal{R}} = 2$), and has a rather wide stencil of the form

$$\tilde{\mathcal{R}} = \mathcal{R} = \frac{1}{16}\begin{bmatrix} nw & n & 0 & 0 \\ w & 4-nw & 4-n-e & 0 \\ 0 & 4-w-s & 4-se & e \\ 0 & 0 & s & se \end{bmatrix}. \tag{83}$$

Here the single coefficients represent the weights with which the fine grid zones contribute to the coarse zone solution or defect. The weights n, w, e, and, s are equal to zero, when the corresponding fine grid node lies outside the computational domain, and one otherwise. The restriction operator given in Eq. (83) was proposed by Khalil & Wesseling (1992) for the case of a diffusion-like problem with Neumann boundaries. It also appears to work reasonably well within cell-centered multigrid for other types of boundary conditions. For the hyperbolic PDEs considered here, the aforementioned transfer operators are adequate as long as the flow Mach numbers do not largely exceed unity. The algorithms that we describe in this paper are unsuited to compute highly supersonic flows (with Mach numbers greater than about three). For these flows upwind prolongation and restriction are expected to be required (see Koren & Hemker 1991).

As a final word of caution on restriction, we note that some authors prefer to define the (local) defect as

$$\boldsymbol{D}_{i,j} = \Omega_{i,j}\left(C\frac{\boldsymbol{U}_{i,j}}{\Delta t} + \boldsymbol{R}_{i,j} - \boldsymbol{f}_{i,j}\right), \tag{84}$$

which differs from our Eq. (56) by a factor $\Omega_{i,j}$. In this case the restriction operators for the solution and the defect, $\tilde{\mathcal{R}}$ and $\mathcal{R}$, differ from each other, as the scaling factor for $\mathcal{R}$ needs to be changed from $1/16$ to $1/4$ in Eq. (83). One would not otherwise obtain correctly scaled defects on the coarser grid.

5.5. Memory requirements

We finally give some details of the memory requirements of the multigrid schemes that we have presented in the preceding sections. For this purpose, we focus on schemes that are based on the ESDIRK46 temporal integrator, as these are the most demanding in terms of memory use within the family of methods that we presented.

As a standard for comparison, we use a corresponding fully explicit scheme that is based on the popular third-order TVD Runge-Kutta temporal integrator of Shu & Osher (1988). We refer to this here as the TVD-RK33 scheme. As a first representative of our multigrid-based ESDIRK46 implicit schemes, we consider a scheme with a five-stage *explicit* smoother (see Sect. 5.3.1), which is called the ESDIRK46-MS5E scheme in the following. The last scheme that we consider is ESDIRK46-MS3I/SGS, an implicit scheme with multigrid and three-stage *implicit* SGS smoothing (cf. Sect. 5.3.3).

Table 1 shows the memory requirements of these schemes for the 2D problem setup described in Sect. 7.5 on a grid of 256×128 zones, as measured on an Intel system running Linux. We note that the tabulated values are intended to give only a very rough indication of the memory requirements, as the absolute amount of the memory used strongly depends on the details of the particular implementation, and the trade-offs made therein with respect to memory versus CPU-time optimization, or ease of coding.

Table 1. Memory requirements of different schemes for the 2D problem-setup described in Sect. 7.5 on a grid of 256×128 zones.

Method	Memory [Mbytes]	Memory factor
TVD-RK33	114	1
ESDIRK46-MS5E	194	1.7
ESDIRK46-MS3I/SGS	680	6.0

Notes. See the main text for details.

As expected, TVD-RK33 is the most memory-efficient scheme of all, but since it is explicit it is unsuitable for stiff problems. The ESDIRK46-MS5E scheme that is suitable for moderately stiff problems, requires about twice the memory of TVD-RK33. This increase in memory usage is necessary because of the need to store both the solution and several residual vectors on the fine grid, and to also store part of this information on the coarse grids. The ESDIRK46-MS3I/SGS scheme – which among the three considered integrators is the preferred solver for stiff problems – needs about three times the memory of the ESDIRK46-MS5E scheme, hence about six times the memory of the purely explicit TVD-RK33 scheme. This is mainly due to the implicit smoothing stages. Our present implementation is designed to save computer time at the expense of some memory when calculating these stages. This is achieved by precomputing and storing, and by subsequently reusing the frozen and LU-decomposed local flux Jacobians in each of the Gauss-Seidel iterations expressed by Eq. (74). While this requires that 2×16 variables per cell face are stored for the 2D Euler equations, it also allows for a general implementation, in which, for instance, SGS can be easily exchanged by RBGS, and where essentially the same implicit smoothing code can also be used for other systems of conservation laws.

With the introduction of some additional coding complexity more memory-efficient implementations are conceivable, which, moreover, would not significantly increase the CPU-time requirements of the method. By storing the flux Jacobians in their symmetrized form (Jameson 2003), the ESDIRK46-MS3I/SGS scheme could be made to require only about three times the memory of the TVD-RK33 scheme. If one wishes to confine oneself to applications governed by the Euler equations, the most memory-efficient implementation of the implicit SGS-based smoothing scheme is probably the one described by Rossow (2007), which stores only the Mach number and two more scalar quantities on all cell faces, and reconstructs the flux Jacobians (in the space of the primitive variables) from this information. Its memory requirements are only about twice as high as those of the TVD-RK33 scheme. This implementation, however, is not applicable to more general systems of conservation laws. It can only be used for the Euler equations, and some related systems where the Mach number is well-defined.

The scaling behavior of the memory requirements, however, is actually even more important than the absolute memory use for a given problem size. For all the algorithms that we present in this paper, the memory requirements increase only linearly (hence scale optimally) with the number of grid zones.

6. Local Fourier analysis

When developing multigrid methods, it is important to have theoretical estimates of the convergence rate of the employed multigrid algorithm. These are very helpful in debugging and improving a multigrid code. The usual way to obtain these estimates is to perform local Fourier smoothing- and two-grid-analyses

(see Brandt 1984; and Trottenberg et al. 2001, for overviews on these techniques). Both methods assume an infinite grid (which corresponds to the use of periodic boundaries) and consider linearized versions of the discretization operators with constant coefficients. Whilst smoothing analysis considers only the performance of the smoother on the finest grid, in order to yield an estimate of the smoothing factor, the two-grid analysis attempts to also take into account the effects of the coarse-grid correction. This is achieved by approximating the full multigrid operator with a two-grid operator, and by assuming that the equations on the coarse grid are solved exactly.

The local Fourier analysis code that we use in this work is a modified version of the MATLAB program developed by Miczek (2008), where more details of the code are given. This tool allows for the calculation of both the one-grid smoothing factor, μ, as well as the two-grid asymptotic convergence factor, $\varrho_{2-\mathrm{grid}}$, of the particular multigrid algorithm that we use here. The calculation of the necessary Fourier symbols of the linearized discretization and smoothing operators follow mostly Swanson et al. (2007), while the Fourier symbols of the cell-centered transfer operators required for the two-grid analysis stem from Mohr & Wienands (2004). We use this tool to provide us with estimates of the convergence rates of our multigrid algorithm in order to obtain theoretical insight into the performance of our method for steady and unsteady problems, namely the numerical setups described in Sects. 7.1 and 7.5.

7. Numerical tests

The test calculations that we present in the following were selected to allow us to test the convergence and stability properties of our multigrid-based solvers under various conditions and difficulties. These include sub, transonic, and moderately supersonic flows with shocks, flow alignment with the grid, the use of non-orthogonal meshes, and the presence of simple (constant) source terms (cf. Sect. 7.2).

Since the efficient calculation of steady states is an important application area of implicit schemes, our first two tests (Sects. 7.1, and 7.2) are pure steady-state problems. These problems are run by setting $C = 0$ in all of the above equations, such that the dual-time related terms (the second and fourth term in Eq. (59), and the $C/\Delta t$ term in Eq. (74)) vanish. Eliminating the dual-time stepping in this way, enables us to march to the steady state by solving just a single non-linear system with the pseudo-time iteration. This approach is the most efficient for steady-state calculations and allows, moreover, for an accurate comparison with convergence rates previously published in the literature.

The last three tests are all run with the full dual-time scheme. They contain one steady state (Sect. 7.4), and two truly time-dependent problems (Sects. 7.3 and 7.5). Our objective here is to test the stability and convergence behavior of the dual-time algorithm starting from small, and progressing to ever larger time steps, or, equivalently, Courant numbers. This is of particular importance for an algorithm intended for astrophysical use, as the Courant numbers in astrophysical problems can vary widely, posing serious challenges to any numerical method. These tests have exposed an unexpected robustness difficulty in the present algorithm for time-dependent flows with very large Courant numbers, and we attempt to investigate the origin of this issue in Sect. 7.5.

Unless stated otherwise, all problems are run with our standard scheme, which uses the SLIP discretization with the Van Leer-like limiter ($\nu = 2$ in Eq. (A.2)), and the Roe flux without an entropy fix. Smoothing is performed with the three-stage scheme of Eqs. (61) and (64), which makes use of the implicit stages described in Sect. 5.3.3. The linear systems arising in these stages are approximately inverted with two SGS iterations. An implicit parameter $\epsilon = 0.7$ is used. Although smaller values of ϵ were found to improve the convergence rates for subsonic problems, they led to reduced robustness when shocks were present in the solution.

The standard scheme makes use of F(1, 0) multigrid cycles, i.e. F-cycles with one pre-smoothing and no post-smoothing step (see Appendix D), and employs the prolongation and restriction operators given by Eqs. (82) and (83). One-dimensional problems, for which the 2D operator of Eq. (83) is ill-defined, are run with simple constant restriction. In choosing the pseudo-time Courant number, we follow Rossow (2007) and set $\sigma_\tau = 40$ for the first four multigrid cycles, while for all subsequent cycles we increase σ_τ to 1000. Our numerical experiments did not require that σ_τ be limited for use in the multistage-implicit schemes, values as large as 10^{12} having indeed been tested in some instances. However, there was also usually no benefit in increasing σ_τ beyond a certain point.

7.1. Steady-state flow in a channel with a bump

Implicit hydrodynamics simulations of this standard test problem were published as early as in the GAMM workshop (Rizzi & Viviand 1981). More recent results were given by, e.g., Mulder (1989), Darmofal & Siu (1999), and Nishikawa & van Leer (2003). The problem domain consists of a channel five units long and two units wide, with a solid bump at the lower wall. The type of bump that we consider here is smooth and sinusoidal.

The employed mesh is non-orthogonal with compression toward the lower wall. The grid-generating transformation is given by

$$x(\tilde{\xi}, \tilde{\eta}) = 5\tilde{\xi} - 2 \tag{85}$$

and

$$y(\tilde{\xi}, \tilde{\eta}) = \begin{cases} z, & \text{for } |x| \geq 0.5 \\ z + 0.042\cos^2(\pi x)(2 - z), & \text{for } |x| < 0.5 \end{cases} \tag{86}$$

with

$$z = 0.313\,(\exp(2\tilde{\eta}) - 1). \tag{87}$$

Here $\tilde{\xi}$ and $\tilde{\eta}$ are assumed to be appropriately scaled generalized coordinates with values between zero and unity. For instance, if ξ and η are assumed to be non-negative, the scaled generalized coordinates are simply given by $\tilde{\xi} = \xi/\max_i(\xi)$ and $\tilde{\eta} = \eta/\max_j(\eta)$. A grid with 32×16 zones, generated by this transformation, is shown in Fig. 3.

Steady solutions are considered in which a flow, with prespecified Mach number at infinity, M_∞, enters the channel at the left and leaves it from the right boundary. Several values of M_∞, several finest mesh sizes, and several multigrid schemes have been tested. In all cases, the iteration was started by initializing the flow variables in the interior of the domain with their values at infinity. Fully converged solutions on a 256×128 zone grid (obtained with our standard scheme) are displayed in Fig. 4b for $M_\infty = 0.5$, and in Fig. 4c for $M_\infty = 1.4$. The convergence histories for these cases are shown in Fig. 5, along with the convergence histories of the corresponding runs at resolutions of 128×64 and 64×32 zones.

A more comprehensive overview of all simulations performed is given in Table 2, in which we list, for four different

Table 2. Number of F(1, 0) multigrid cycles, computational work, and CPU times (in seconds) on a single processor of an Intel Xeon system required to reduce the defect by six orders of magnitude for the bump problem.

		MS5E			MS3I/SGS			MS3I/RBGS(4)			MS3I/RBGS(8)		
M_∞	Mesh	N_{cyc}	Work	CPU	N_{cyc}	Work	CPU	N_{cyc}	Work	CPU	N_{cyc}	Work	CPU
0.3	32×16	383	3404	2.6	18	96	0.29	18	96	0.32	14	75	0.34
	64×32	588	5227	16	22	117	1.5	25	133	1.8	17	91	1.7
	128×64	1151	10 231	123	31	165	10	38	203	13	24	128	11
	256×128	2069	18 391	892	48	256	66	64	341	101	36	192	75
0.5	32×16	269	2391	1.8	13	69	0.21	14	75	0.26	12	64	0.27
	64×32	406	3609	11	14	75	0.94	18	96	1.3	13	69	1.3
	128×64	752	6684	79	20	107	6.4	26	139	8.7	18	96	8.0
	256×128	1330	11 822	575	30	160	41	42	224	66	26	139	56
0.8	32×16	298	2649	2.2	21	112	0.33	24	128	0.40	23	123	0.55
	64×32	556	4942	16	58	309	4.2	57	304	4.1	54	288	5.4
	128×64	720	6400	83	43	229	14	40	213	13	36	192	16
	256×128	DNC	–	–	81	432	112	67	357	107	107	571	217
1.4	32×16	319	2836	2.3	29	155	0.45	27	144	0.48	47	251	1.1
	64×32	445	3956	13	25	133	1.7	27	144	1.9	25	133	2.5
	128×64	DNC	–	–	25	133	8.2	34	181	11	30	160	15
	256×128	DNC	–	–	29	155	40	62	331	99	41	219	83

Notes. DNC: did not converge, MS5E: multigrid solver based on explicit five-stage smoothing scheme, MS3I/SGS, MS3I/RBGS: multigrid solver based on a three-stage implicit smoothing scheme with either symmetric or red-black Gauss-Seidel smoothing stages. Values in parentheses denote the number of Gauss-Seidel sweeps per implicit stage.

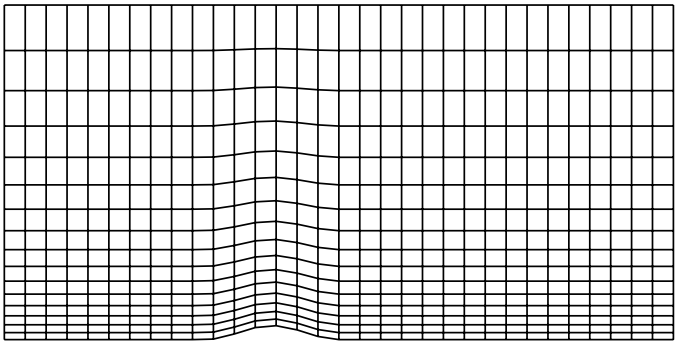

Fig. 3. A 32×16 zone, non-orthogonal grid for the channel flow problem generated by the transformation given by Eqs. (85)–(87).

multigrid schemes, the number of F(1, 0) cycles necessary to reduce the defect by six orders of magnitude, the total computational work required to evaluate the non-linear defects, and the actual CPU time measured on a single processor of an Intel Xeon system.

The computational work is estimated as

$$W = (n_{pre} + n_{post}) \times K \times N_{cyc} \times \Gamma_{cyc}, \tag{88}$$

where n_{pre} and n_{post} are the number of pre- and post-smoothing steps, respectively, K is the number of stages performed within the employed multistage smoothing scheme, N_{cyc} is the total number of multigrid cycles, and the factor Γ_{cyc} accounts for the work expended on the coarse grids, and depends upon the multigrid cycle-type used. For F-cycles, it has the value 16/9. We note that by definition, one work unit corresponds to one evaluation of the defect on the finest grid. In the present, source-term free problem, the work in evaluating the defect is dominated by the solution of the Riemann problem. Hence, Eq. (88) essentially gives the work in terms of Riemann solutions on the finest grid.

To obtain the convergence results listed in Table 2, it was essential to use 1D characteristic variables, i.e. the appropriate Riemann invariants and entropy, in formulating the far-field boundary conditions. For an overview of how to implement these in non-orthogonal coordinates, as well as the reflecting (inviscid wall) boundary conditions that are used at the top and bottom of the channel, we refer for instance to Swanson & Turkel (1997).

The "bump problem" is quite difficult to solve efficiently because the flow is aligned with one of the coordinate directions of the mesh throughout the entire computational domain. Under such conditions, standard multigrid schemes – which are based on upwind discretization, and full coarsening – experience convergence difficulties caused by the lack of cross-flow damping (cf. Wesseling 1990). This is especially pronounced if explicit smoothers are used, as can be clearly seen from the data given in the third, fourth, and fifth columns of Table 2. These data were obtained by running our full-coarsening pseudo-time multigrid solver with the explicit five-stage smoothing scheme, whose coefficients are given by Eq. (63). In what follows, we refer to this multigrid variant as the MS5E solver. The subsonic runs with this scheme were done with the second-order SLIP reconstruction, but with the limiting switched off, as described in Appendix A. Otherwise, convergence by six orders of magnitude could not be obtained[5]. Moreover, a Courant number $\sigma_\tau = 1.5$ was used in the pseudo-time stepping. The number of work units required for convergence by six orders of magnitude with the MS5E scheme is very high. It ranges from a few thousand on the coarsest meshes to more than ten thousand on the 256×128 zone mesh. Moreover, the number of cycles required by the MS5E scheme for convergence shows a pronounced dependence on the Mach number. The $M_\infty = 0.3$ case converges slower than the $M_\infty = 0.5$ case, while the transonic and supersonic cases on the finer grids do not converge at all with this scheme.

To demonstrate the tremendous gains in robustness and efficiency that can be achieved by adding implicit stages to the smoother, we list the results obtained with our standard pseudo-time multigrid scheme, referred to as the MS3I/SGS scheme in what follows, in the sixth to eighth columns of Table 2. It is obvious that convergence was achieved for all setups that were

[5] Note that this is in contrast to the rest of the results presented in this paper, which were obtained with the limiter being enabled.

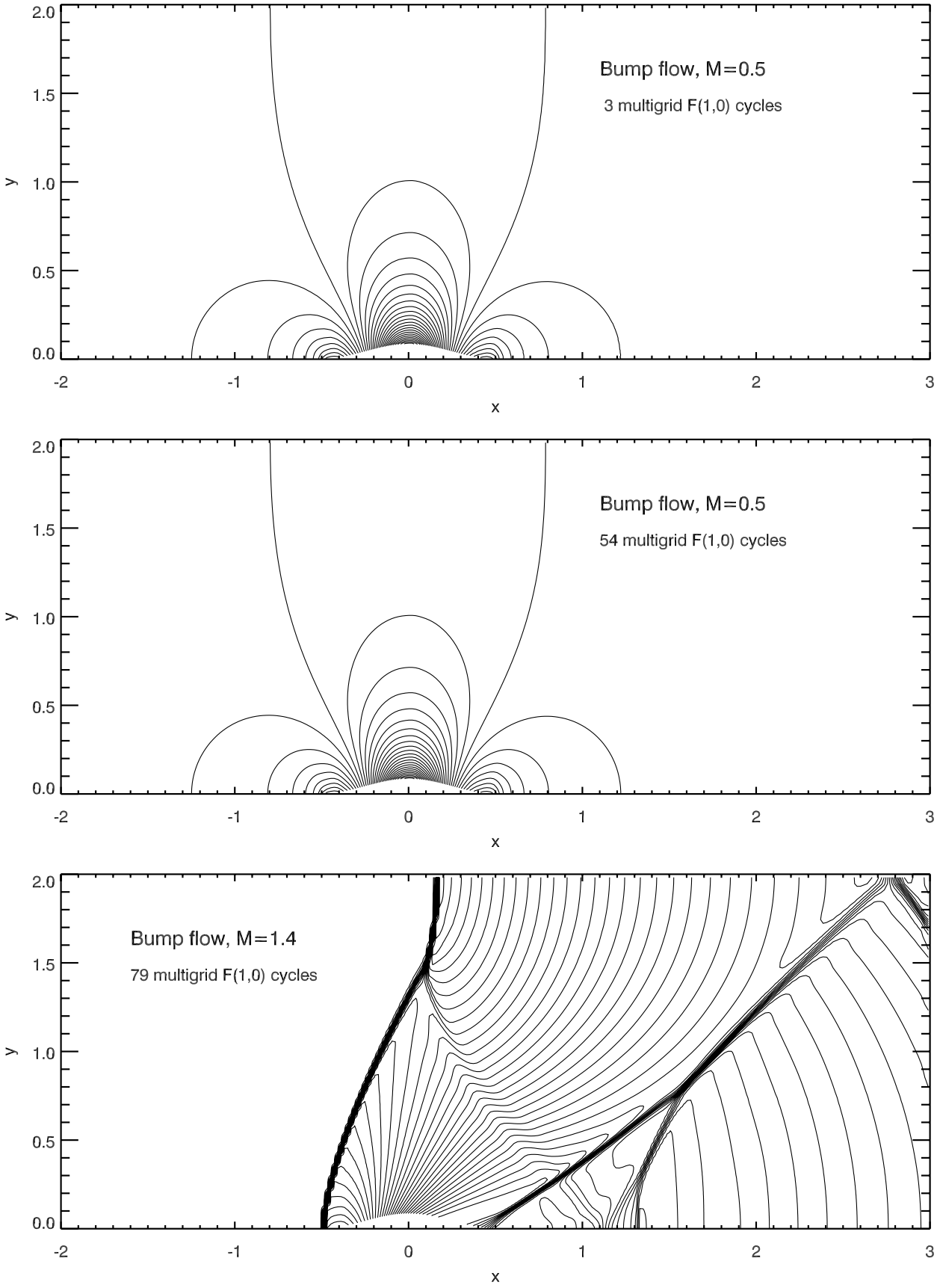

Fig. 4. Pressure distributions of bump flow solutions on the 256 × 128 zone mesh using the standard scheme, MS3I/SGS. Fifty equidistantly spaced contours are shown between the pressure minimum and maximum. From top to bottom: **a)** $M_\infty = 0.5$ case after only three multigrid F(1, 0) cycles, **b)** fully converged $M_\infty = 0.5$ case (after 54 F(1, 0) cycles that reduced the defect by 12 orders of magnitude), and **c)** fully converged $M_\infty = 1.4$ case.

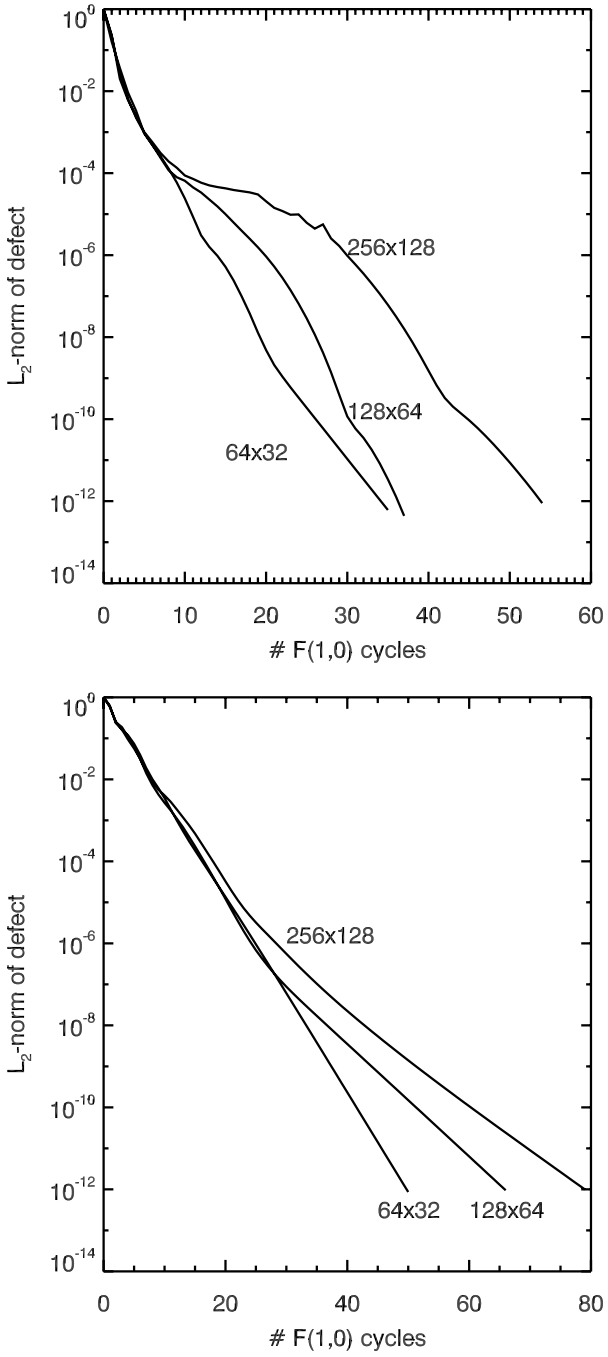

Fig. 5. Convergence of the standard scheme, MS3I/SGS, for the bump flow problem. The L_2-norm of the defect for the density field (normalized with respect to its initial value) is shown as a function of the number of F(1, 0) multigrid cycles performed for spatial resolutions of 64 × 32, 128 × 64, and 256 × 128 zones. From top to bottom: **a)** $M_\infty = 0.5$ case, **b)** $M_\infty = 1.4$ case.

tested, despite the limiter having been switched on again for these and all subsequent runs. The sensitivities of the convergence rate to both the mesh size and the Mach number are also strongly reduced, though not completely eliminated (see also the discussion below). Moreover, both the number of iteration cycles and the work units spent in evaluating the defects show drastic improvements compared to the explicit smoothing case. In terms of the amount of work completed, these improvements range up to a factor of ∼70, as only a few hundred work units are now required.

However, this does not translate directly into a corresponding improvement in speed relative to the MS5E scheme, since we have so far neglected the work required to carry out the SGS iteration for the implicit smoothing. In particular, the matrix-vector products required by Eq. (76) are a particular bottleneck in our present code version. As we have striven for an implementation that is sufficiently general to be applied to different systems of conservation laws, we coded these products in a rather straightforward fashion, not exploiting any special properties of the underlying system of equations to optimize these operations, as done, e.g., in the algorithm for the Euler equations proposed by Rossow (2007). Our measured run times therefore show the MS3I/SGS scheme to be faster than the MS5E scheme by "only" a factor of up to ∼15. This is more than enough to develop schemes that use additional implicit smoothing stages, which are our methods of choice among the class of solvers that we have presented, and to forgo, in what follows, multigrid solvers with purely explicit smoothers (such as the MS5E scheme). Our preferred solver for the remaining test problems in this paper is therefore our standard, MS3I/SGS, scheme.

It is interesting to compare the present work estimates for the MS3I/SGS scheme with the multiple-semicoarsening multigrid results given in the work of Darmofal & Siu (1999). These authors applied their codes to the same test problem, using the same mesh sizes, and the same convergence criterion, so that a reasonable comparison can actually be made. From their Table III, one can infer that their (unpreconditioned) multiple-semicoarsening multigrid scheme typically required between 500 and 1400 work units to converge to the same residual level as our MS3I/SGS scheme. The latter typically needs 100–300 work units, but, as explained above, this estimate does not include the work of the Gauss-Seidel iteration and needs to be multiplied by a correction factor. For our present code version, this factor is approximately five. Hence, both approaches appear to yield similar efficiency in two dimensions. In three dimensions, we expect the MS3I/SGS-based, full coarsening multigrid scheme to be more efficient because of the significant increase in computational complexity associated with the multiple-semicoarsening method. This approach applies semicoarsening to each coordinate direction, thus generates a correspondingly large number of grids on the coarse levels (Mulder 1989; Darmofal & Siu 1999; Trottenberg et al. 2001).

To indicate the potential for parallelization of multigrid schemes that are based on multistage-implicit smoothing, the last six columns of Table 2 finally display convergence data that were obtained by replacing the SGS iteration in the MS3I/SGS scheme by RBGS iteration. We note that for all runs performed with the MS3I/SGS scheme in this paper, we make use of two SGS iterations within each implicit stage. This amounts to two forward and two backward Gauss-Seidel sweeps through the grid, i.e. to four sweeps in total, per implicit stage. Using this same number of Gauss-Seidel sweeps with red-black Gauss-Seidel iteration gives the MS3I/RBGS(4) results listed in Table 2. On the other hand, doubling the number of Gauss-Seidel sweeps in the latter case to eight, leads to the data denoted MS3I/RBGS(8) in Table 2.

An inspection of the CPU-times obtained for all MS3I-type multigrid schemes shows that the RBGS-variants are only slightly less efficient than the SGS-based scheme. For most of the cases listed in Table 2, the overall computer time required by the RBGS-based schemes grew only by about 20−30%, relative to the SGS-variant. Although there is some notable dependence

on M_∞ and the number of red-black Gauss-Seidel sweeps employed, in only a few cases is the CPU time increased by about a factor of two relative to the MS3I/SGS scheme. Given the large speed-ups offered by parallel processors, the small loss in arithmetic efficiency of the method resulting from replacement of SGS by RBGS iteration should be meaningless. The MS3I/RBGS(4) variant involves half the amount of both communication and floating point operations required by the MS3I/RBGS(8) scheme per implicit smoothing stage. However, since the ratio of computation to communication is about equal, and since in certain cases it can be beneficial to use a larger number of Gauss-Seidel sweeps (as indicated by Table 2) both variants could be used in a scalable parallel implementation.

We point out that all the results shown in Table 2 were obtained by reducing the defect by six orders of magnitude. This is actually not required in practice, as discretization accuracy is reached much earlier. For steady problems, only about five iterations, or less, are typically required. To illustrate this, we display in Fig. 4a the flow field for the $M_\infty = 0.5$ case after only three iterations with the MS3I/SGS scheme. A comparison to the fully converged solution shown in Fig. 4b demonstrates that convergence has essentially been achieved within three multigrid F(1, 0) cycles, and that a further reduction of the defect only yields imperceptible improvements to the solution.

It is, moreover, noteworthy, that with the present MS3I/SGS multigrid scheme, about nine multigrid cycles are required in the $M_\infty = 0.5$ case to reduce the defect by the first four orders of magnitude, regardless of the mesh size that is employed (Fig. 5a). This corresponds to an initial convergence rate of 0.36. The smoothing factor predicted by the local Fourier analysis (LFA) is $\mu = 0.35$ (provided that we use the three-stage scheme with the coefficients of Eq. (64), more than two SGS sweeps, and the parameters $\epsilon = 0.7$, and $\sigma_\tau = 1000$). This is in reasonable agreement with the actual measured value of the initial convergence rate.

From Fig. 5, it is also apparent, however, that – in contrast to the predictions of the LFA – the *asymptotic* convergence rate of our current multigrid implementation *does* depend on the mesh size. This is the case for both the subsonic and supersonic setups. There are several possible reasons for this. Shocks, for instance, are known to lead to a degradation of the asymptotic multigrid convergence rate (Brandt 1984; Mulder 1989; Darmofal & Siu 1999). The same is true for non-trivial boundary conditions. With all but periodic boundary conditions, the residual in multigrid codes is often seen to be dominated by high-frequency error components near boundaries. Such errors are removed much slower by global smoothing than high frequency errors in the interior. Once a few multigrid cycles have been performed, the near-boundary residuals may determine the asymptotic convergence rate, which in turn badly affects the mesh independent convergence of the entire method (Brandt 1984; Brandt & Dym 1996).

We have, indeed, observed large high-frequency, near-boundary residuals in most fluid dynamics problems that we have solved so far. The simplest way to restore the original (interior) convergence rate is then to use additional local smoothing sweeps near the boundaries, as recommended by Brandt (1984). However, these local sweeps are truly appropriate only if the resulting code is developed specifically for serial machines. On parallel systems, this remedy would lead to a severe load imbalance, as the local sweeps cannot proceed concurrently with the regular smoothing pass over the entire domain. For this reason, we have so far refrained from implementing local smoothing sweeps in our code.

A final factor that may contribute to the degradation of our asymptotic convergence rates is probably that (for reasons of simplicity and to maintain consistency with the LFA code) we did not use an entropy fix with the Roe flux in the present simulations. The larger numerical viscosity obtained by the use of an entropy fix might mitigate somewhat the convergence difficulties owing to the problem of the vanishing cross-flow numerical viscosity, which we mentioned earlier, potentially at the cost of losing some solution accuracy.

7.2. Relaxation of a uniform density gas in a homogeneous gravitational field

This 1D problem was considered previously by Fryxell et al. (1986). It involves the relaxation of a uniform density gas to a steady state configuration driven by a source term. The initial state (slightly modified from that of Fryxell et al.) is here defined to be

$$(\rho, p, u) = (1, 100, 1). \tag{89}$$

The source term is a constant gravitational acceleration $g = 0.1$ to the left. As in Fryxell et al., reflecting boundaries are used on both sides. The initial state, and the solution as computed with our standard scheme on a uniform mesh of 64 zones after only two, and five F(1, 0) multigrid cycles, are shown in Fig. 6. Both of the latter solutions are identical and demonstrate that convergence has been achieved within only two multigrid cycles. The dips in the density profile at $x = 0$ and $x = 100$ can also be seen in the plots of Fryxell et al. (1986), and are a consequence of using reflecting boundary conditions for the present problem. These dips are a true feature of the solution, and are unrelated to the slow removal of near-boundary residuals that was discussed in Sect. 7.1.

7.3. Sod shock tube problem

The first problem that we run with the full dual-time scheme is the well-known shock tube problem of Sod (1978). This is an unusual problem for an implicit scheme, as a solution to this problem does not require a non-uniform grid, and the features of interest are non-linear waves and contact discontinuities, whose propagation speeds are of similar size. Hence, there is neither discrete nor analytic stiffness in this problem. To follow the wave propagation accurately, one needs to choose the time step such that the local Courant number associated with the fastest waves, $CFL_{i,j}^{|u|+c}$, does not exceed unity anywhere on the grid. This local Courant number is defined as

$$CFL_{i,j}^{|u|+c} = \frac{\Delta t \left(\lambda_{i,j}^{\xi} + \lambda_{i,j}^{\eta}\right)}{\Omega_{i,j}}, \tag{90}$$

where the $\lambda_{i,j}$ are given by Eqs. (68)–(69). The above condition is, of course, fulfilled if we require the maximum (or global) Courant number over the domain

$$CFL^{|u|+c} = \max_{i,j}\left(CFL_{i,j}^{|u|+c}\right) \tag{91}$$

to be of order unity or smaller

$$CFL^{|u|+c} \leq 1. \tag{92}$$

Equation (92) simply restates that the problem is not stiff. If the accuracy requirements of the solution result in the necessity to fulfill Eq. (92) over the largest part of a simulation, implicit

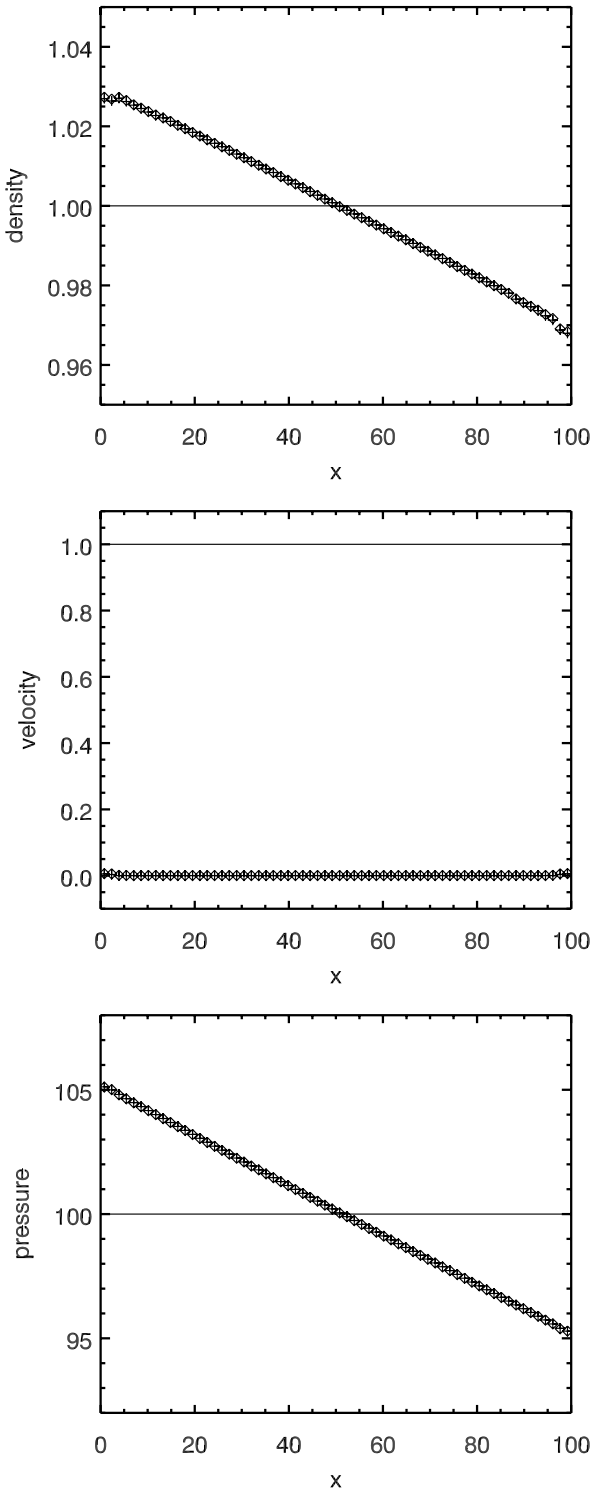

Fig. 6. Relaxation of a constant density gas in a gravitational field. Solid lines show the constant initial state. Diamonds give solution values obtained with the standard scheme (MS3I/SGS) on a mesh of 64 zones after two iteration cycles, while crosses mark the solution values obtained after five iterations.

schemes should *not* be used, because they are not competitive and are greatly outperformed by explicit ones.

Nevertheless, in a real-world calculation one may temporarily enter this regime, and the stability of the scheme for small time steps then becomes important. The dual-time schemes driven by the multistage-implicit smoothers can calculate stable solutions under these circumstances, despite the difficulties that this involves, as discussed in Sect. 5.3.1.

We demonstrate this here by showing the results of the Sod problem as computed with the ESDIRK23-MS3I/SGS scheme, and the step size controller of Press et al. (1992). The latter was used with the requirement that the relative error in all conserved variables remain smaller than 1% per step. We have run this problem on a uniform 256 zone mesh with an initial time step size $\Delta t = 10^{-4}$, corresponding to a $CFL^{|u|+c} \approx 0.1$, which was almost immediately raised by the step-size controller to values corresponding to $CFL^{|u|+c} \approx 0.9$. To obtain an accurate time-evolution, it was sufficient to reduce the L_2-norm of the density defect by three orders of magnitude in each implicit ESDIRK stage.

A snapshot from this run (at time $t = 5.98 \times 10^{-2}$) is shown in Fig. 7a, together with a reference solution (Fig. 7b) that was obtained by replacing the implicit integrator by the third-order explicit TVD Runge-Kutta scheme of Shu & Osher (1988) in order to evolve the semi-discrete system expressed by Eq. (42). The solutions obtained in both runs are indistinguishable, and demonstrate that the ESDIRK23-MS3I/SGS scheme combined with the time-step controller of Press et al. (1992) did a good job. The small glitch in the solution at the location of the original discontinuity, $x = 0.5$, is due to a start-up error (see LeVeque 1998), which is not smoothed out with time because we did not use an entropy fix in our Roe solver. This glitch could be reduced by slightly smearing the initial, sharply discontinuous profile. Overall, the accuracy of the solution is very satisfactory for an implicit scheme. No oscillations are present, and the shock is captured within four zones. Owing to the lack of a contact steepener in the SLIP spatial discretization, the contact discontinuity is smeared out to about eight zones.

The convergence behavior of the multigrid iterations for this run is shown in Fig. 8. Convergence is very rapid, requiring only about five F(1, 0) multigrid cycles to reduce the defect by three orders of magnitude for every implicit ESDIRK stage. This is due to the dominance of the dual-time source term $\boldsymbol{C}\boldsymbol{U}/\Delta t$ in Eq. (59), which is caused by the appearance of the (small) time step size, Δt, in its denominator. This term renders the discretized equations diagonally dominant, so that they can be efficiently solved using nearly any iterative scheme.

7.4. Steady, regular shock reflection off a wall

The second problem that we run with the full dual-time scheme is the well-known regular reflection of a Mach 2.9 shock from a wall (for details on the setup see, e.g., Colella 1990, and the references therein). This problem is run on a uniform Cartesian mesh of extent $[0, 4] \times [0, 1]$. The evolution is started from a uniform initial state, with the flow variables in the entire domain set equal to their values at the left boundary.

This test is a steady-state problem. As such, it should allow us to probe the robustness properties and the convergence behavior of the multigrid solver – as used within the dual-time framework – for a Courant number $CFL^{|u|+c}$ that is orders of magnitude larger than in the Sod problem. A difficulty unrelated to the multigrid algorithm, is encountered in the strive for such large Courant numbers: for sufficiently large time steps, the underlying ESDIRK temporal integrators, which are only *conditionally* TVD, will no longer preserve the monotonicity property of the spatial discretization. Since the present test problem involves two rather strong shocks, such a loss of monotonicity will lead to a non-linear instability (cf. Sect. 4). Hence, one cannot

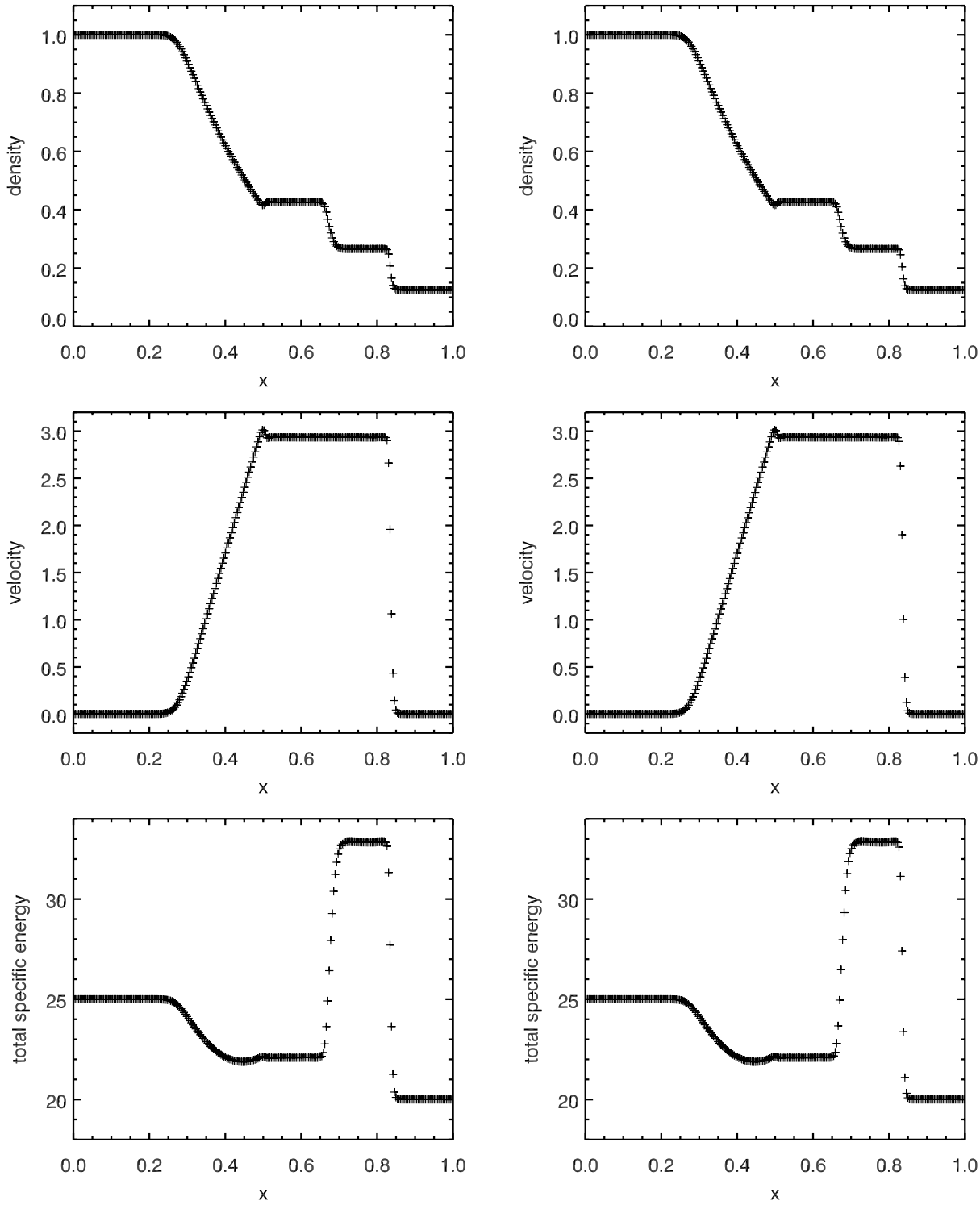

Fig. 7. Sod problem on a 256 zone mesh. From left to right **a)** implicit ESDIRK23 solution (obtained by reducing the defect by three orders of magnitude for each implicit ESDIRK23 stage), and **b)** explicit reference solution calculated with third order TVD RK-scheme.

increase Δt (or $CFL^{|u|+c}$) unconditionally. One instead needs to impose a constraint on $CFL^{|u|+c}$ to render the solution monotone, and the calculation non-linearly stable. Employing the ESDIRK23 scheme and the non-compressive minmod-limiter, we had to limit $CFL^{|u|+c}$ to a value of about 400. We note that these constraints were *not* needed in calculating steady solutions with the pure pseudo-time algorithm in Sects. 7.1 and 7.2. If the steady-state solution is all that is required, the pure pseudo-time scheme, which is based on the unconditionally TVD, backward Euler method, should always be preferred over the full dual-time scheme.

Nevertheless, the full dual-time algorithm can converge quite rapidly to steady states. Figure 9 demonstrates this for a simulation performed with the ESDIRK23-MS3I/SGS scheme on a grid of 128×64 zones, and a constant time step size $\Delta t = 2$, corresponding to the aforementioned limit of $CFL^{|u|+c} \approx 400$. The

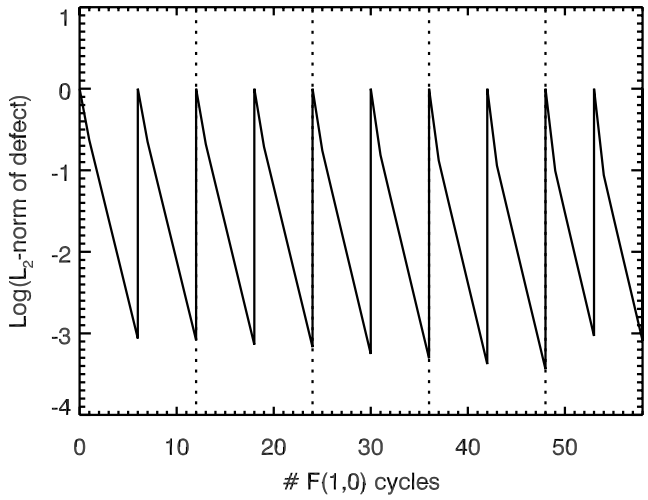

Fig. 8. Convergence of the multigrid scheme for the Sod shock tube problem as computed with the ESDIRK23 discretization. The logarithm of the L_2-norm of the defect for the density field (normalized with respect to its initial value at the beginning of a multigrid iteration) is depicted as a function of the total number of F(1, 0) multigrid cycles performed. The first five (physical) time steps are shown, separated by dotted lines. Each ESDIRK23 time step contains two implicit stages. For each of these stages, an implicit system given by Eq. (47) has to be solved by the multigrid solver. These stages can be discerned by vertical jumps in the defect, which signify the start of the multigrid iteration for the next implicit stage.

defect was again reduced by three orders of magnitude in each implicit ESDIRK stage.

It is evident from Fig. 9 that after one time step, both shocks are clearly present on the grid. Their shape has approached the steady-state solution after only four time steps, and the solution no longer changes after eight steps.

The convergence of the multigrid iterations is quite rapid and displayed in Fig. 10. We note that, as an ESDIRK23-step contains two implicit stages, *two* non-linear systems need to be solved per time step. For each of these systems, about ten F(1, 0) multigrid cycles are required to reduce the defect by three orders of magnitude. This is comparable to the convergence rates observed for the high Mach-number setup in Sect. 7.1. An important difference from the convergence histories obtained with the pure pseudo-time scheme in Sect. 7.1, however, is that beginning with the third pseudo-time iteration the defect *increases* within the first multigrid cycle by a factor of a few and requires about three cycles to be reduced back to a value of unity. As we demonstrate in the following section, by applying the method to a problem with even larger Courant numbers, this behavior is a first indication of a multigrid robustness issue for large time steps, whose deeper origin is not yet understood.

7.5. Vortex shedding behind a cylinder

Our last test probes the robustness behavior of the dual-time multigrid solver in the range $10^3 < CFL^{|u|+c} < 10^4$, i.e. for Courant numbers that are yet another order of magnitude larger than in the shock reflection problem. The limit of large Courant numbers is actually the most important for implicit solvers, as it is in this regime that these methods will be primarily used. To reach these high Courant numbers, we consider, in the following, a shock-free laminar flow problem. Moreover, we replace the ESDIRK23 temporal discretization by the higher-order ESDIRK46 scheme given by Kennedy & Carpenter (2001). The coefficients of the latter scheme can also be found in Bijl et al. (2002). The high Courant numbers, that we consider here, preclude the use of the ESDIRK23 scheme. At the resulting large time steps, the accuracy of the ESDIRK23 scheme is no longer sufficient to fulfill the standard requirements on temporal accuracy that we have used so far in this paper.

The multigrid solver that we employ differs from our standard scheme only in that we make use of simple V-cycles. For reasons connected to the performance of the coarse-grid correction (which become clearer in the following) simple V-Cycles, which spend much less computational effort on the coarse levels than F-cycles, gave almost the same convergence rate as the latter. The choice of V-cycles thus helped to reduce the (rather large) computational cost of an implicit ESDIRK46 step, which involves the solution of five non-linear systems. Each of these systems was iteratively solved until the defect had dropped by three orders of magnitude.

The problem that we now study is vortex shedding in the flow behind a 2D cylinder, with a Mach number of 0.3 at infinity. This is a standard problem for Navier-Stokes codes, for which accurate comparisons to experimental results have been performed (see, e.g. Melson et al. 1993; Bijl et al. 2002; Jothiprasad et al. 2003, and the references therein). We note that it is not our aim here to provide a physically satisfactory simulation of this problem. This is indeed impossible by solving the inviscid (Euler) equations. In a simulation based on these equations, the boundary layer from which the vortices separate is purely due to *numerical* viscosity, hence one will only obtain a *qualitatively* correct picture of the flow. This is sufficient for us here, as our actual goal is to study the stability and efficiency of different numerical methods. In this vein, we use the present setup as a toy problem to illustrate a widespread computational difficulty that also appears in complex astrophysical contexts, namely that stiffness can arise because the time step in explicit flow simulations may be limited by both regions and waves that differ from those that one is actually interested in.

The mesh on which we solve this problem is an orthogonal polar mesh, consisting of 256×128 zones, with equidistant spacing in angle and strong radial compression toward the cylinder's wall. The distance between the wall and the next mesh line is 0.1% of the cylinder's diameter. Owing to the strong mesh compression, the local Courant numbers associated with both acoustic and vorticity waves, $CFL_{i,j}^{|u|+c}$, and $CFL_{i,j}^{|u|}$, respectively, have global maxima on the cylinder's surface, and decrease sharply with growing distance from the cylinder. Hence, the maximum Courant number in the domain, $CFL^{|u|+c}$, occurs on the cylinder's surface. However, in determining the time step for following the actual features of interest in this problem, i.e. the vortices, it is the local $CFL_{i,j}^{|u|}$ in the wake region that should be kept below some threshold value. In this region, $CFL_{i,j}^{|u|}$ has dropped by three orders of magnitude compared to $CFL^{|u|+c}$. This gives rise to a (moderately) stiff problem, where the stiffness is of the discrete type.

A viscous wall boundary condition is used at the cylinder's surface, in which all velocity components are set to vanish at the wall, while far-field conditions based on characteristic variables are used at the outer boundary, which is located at a distance of 20 cylinder diameters from the center. The flow is initially homogeneous and symmetric. A small random perturbation thus has to be added to the flow field to break this initial symmetry. A consequence of the initial conditions is that the evolution is initially dominated by an acoustic wave, which forms at the cylinder's

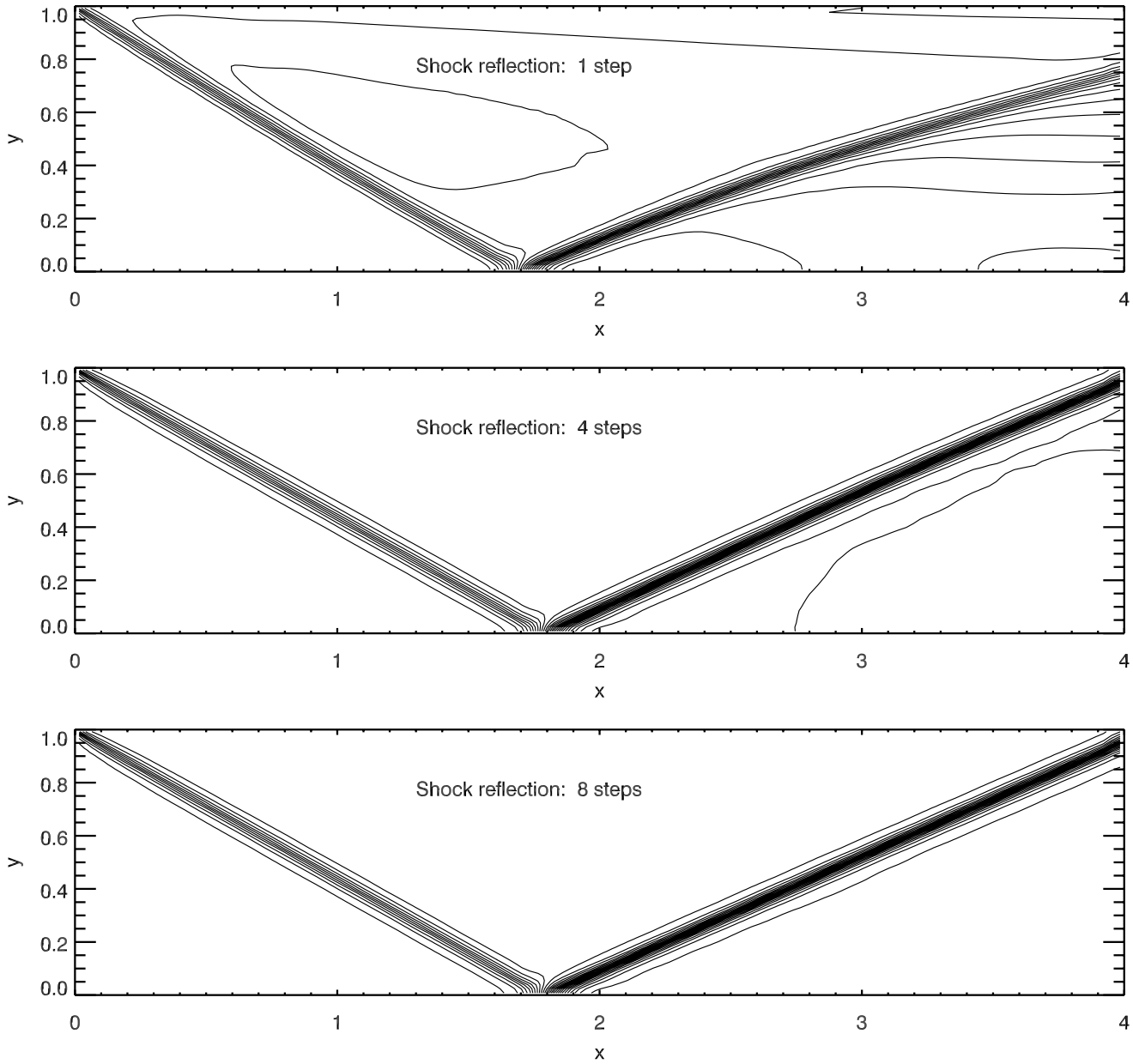

Fig. 9. Evolution of the pressure distribution in the shock reflection problem on a grid of 128×64 zones. From top to bottom **a)** after one time step, **b)** after four time steps, and **c)** after eight time steps with $CFL^{|u|+c} \approx 400$. The full dual-time scheme (with ESDIRK23 temporal, and SLIP spatial discretization with minmod limiter) was used. For each state, 30 equidistantly spaced contours are shown between the pressure minimum and maximum.

surface and starts to propagate outwards before any vortex shedding (which occurs on fluid as opposed to acoustic timescales) has set in. We therefore decided to evolve the flow first with the explicit Shu & Osher (1988) integrator, up to a point where the wave had left the computational domain. The solution was then dumped to a restart file.

Restarting from this state, we first ran the dual-time scheme for a single time-step corresponding to a $CFL^{|u|+c} = 2000$. This was then repeated with $CFL^{|u|+c} = 4000$ and $CFL^{|u|+c} = 8000$, which are all Courant numbers in the vicinity of the $CFL^{|u|+c}$ yielded by our step size controller. To obtain a reference for comparison of the implicit scheme, we also did additional runs using the explicit Shu & Osher integrator. These were started from the same restart file and were run up to the same evolutionary times as the implicit simulations. Table 3 gives an overview. Pressure distributions as obtained from the implicit run requiring one step at $CFL^{|u|+c} = 4000$, and the corresponding explicit run, that required more than 6600 steps at a $CFL^{|u|+c} = 0.6$ to cover the same time, are displayed in Fig. 11. One can see that around this early phase in the evolution the vortices separating from the cylinder's surface are still symmetric with respect to the $y = 0$ line of the grid. Moreover, the implicit and explicit solutions show only minor differences, which demonstrates that the implicit scheme also works well in two dimensions.

Before one can attempt a fair comparison of the efficiency of the implicit and explicit schemes for this problem, a close look into the convergence characteristics of the present implicit method is required. In Fig. 12, we show, for the very first implicit stage performed with the ESDIRK46 scheme after restarting, the dependence of the multigrid convergence history on the Courant

Table 3. Courant numbers, number of time steps, work spent in evaluating the non-linear defects, and CPU times in seconds for the different runs of the vortex-shedding problem described in the text.

Method	$CFL^{\lvert u\rvert+c}$	Steps	Work	CPU time [s]	Speed-up	Potential speed-up
ESDIRK46-MS3I/SGS	2000	1	432	136	3.0	11
TVD-RK33 explicit	0.6	3334	10 002	402	–	–
ESDIRK46-MS3I/SGS	4000	1	624	195	4.1	19
TVD-RK33 explicit	0.6	6667	20 001	802	–	–
ESDIRK46-MS3I/SGS	8000	1	872	273	5.9	36
TVD-RK33 explicit	0.6	13 334	40 002	1610	–	–

Notes. The listed speed-ups are calculated with the explicit runs taken as the reference. The last column gives the estimated speed-up of an ideal ESDIRK46-MS3I/SGS scheme, whose multigrid solver would converge at a rate that equals the smoothing factors listed in Table 4.

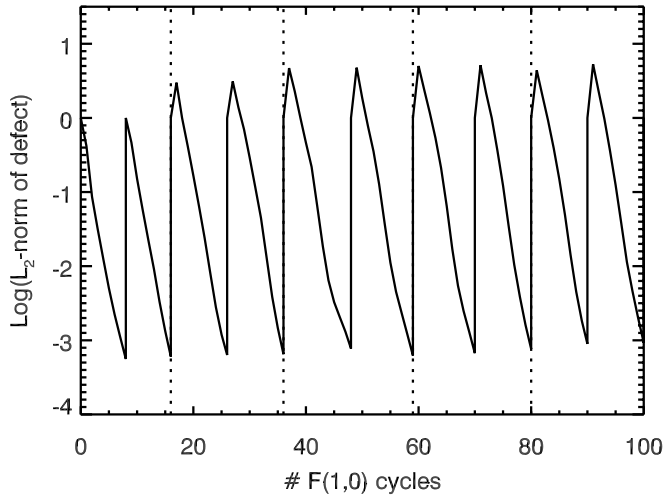

Fig. 10. Same as Fig. 8 but for the first five time steps of the shock reflection problem.

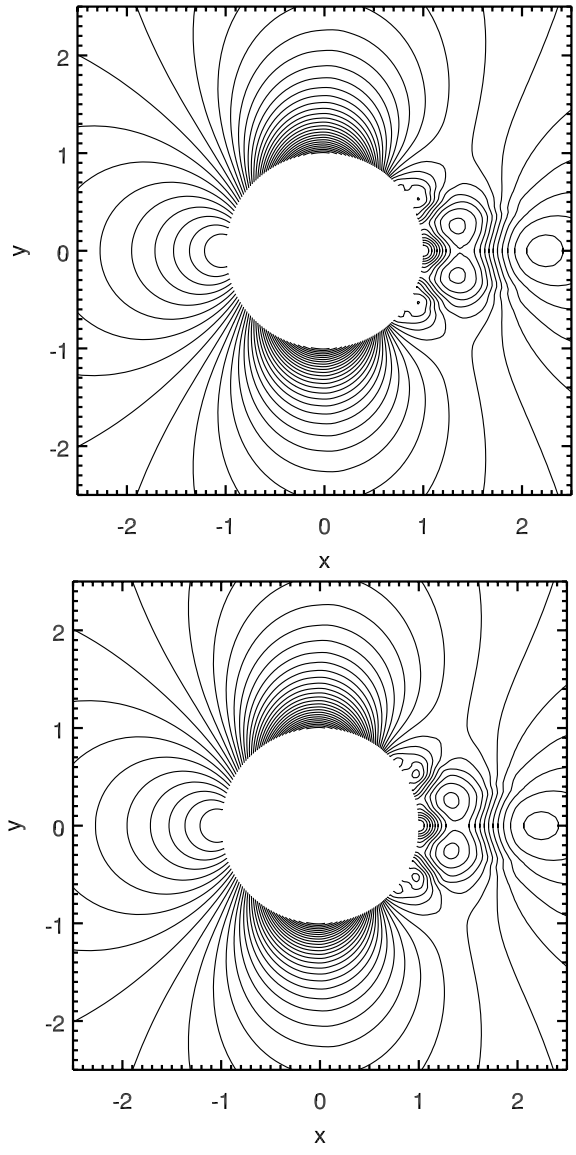

Fig. 11. Pressure distributions for the vortex shedding problem after restarting the calculations as described in the text. From top to bottom **a)** at $t = 70.9$ after performing one step with the implicit scheme at $CFL^{\lvert u\rvert+c} = 4000$, **b)** at the same evolutionary time, but calculated with the Shu & Osher TVD RK-scheme that required more than 6600 explicit steps.

number. Two things are noticeable. First, there is a quick rise of the residual within the very first multigrid cycle, a phenomenon that was previously observed in the shock reflection problem (cf. Fig. 10). Second, the asymptotic convergence rate of the multigrid solver degrades as the chosen $CFL^{\lvert u\rvert+c}$ (and therefore the time step Δt) increases. Beyond a critical $CFL^{\lvert u\rvert+c}$ of about 9000, we encountered an instability, i.e. the employed multigrid solver no longer converged.

The reason for this unexpected breakdown in the multigrid solution is unclear. We were, however, able to narrow down the possible origins of the problem by obtaining insight into the theoretical convergence behavior of the method using local Fourier smoothing, and two-grid analysis (cf. Sect. 6). To perform the local Fourier analysis (LFA), the non-linear equations were linearized around a reference state with a Mach number of 0.3, and a flow inclination angle of 45° relative to a uniform Cartesian mesh with 64×64 zones. The latter was used because LFA only admits meshes with a spatially constant aspect ratio. Parameters that are inherent to the numerical scheme were chosen exactly as in our actual simulation code. For the present ESDIRK46-MS3I/SGS scheme, we used $\epsilon = 0.7$, $C = 4$, and $\sigma_\tau = 1000$, while two SGS iterations were assumed for calculating the Fourier symbol of the implicit smoothing operator.

The convergence rates determined from the LFA as a function of $CFL^{\lvert u\rvert+c}$ are given in Table 4, where they are also compared to the actual asymptotic multigrid convergence rates that we determined from Fig. 12 (by excluding the first six multigrid cycles). The most important point to note in Table 4 is that the smoothing factor, μ, determined for an implicit stage of the ESDIRK46-MS3I/SGS scheme shows *no* significant dependence on $CFL^{\lvert u\rvert+c}$. Moreover, with values close to 0.3 it is of the same size as the smoothing factors for steady problems that

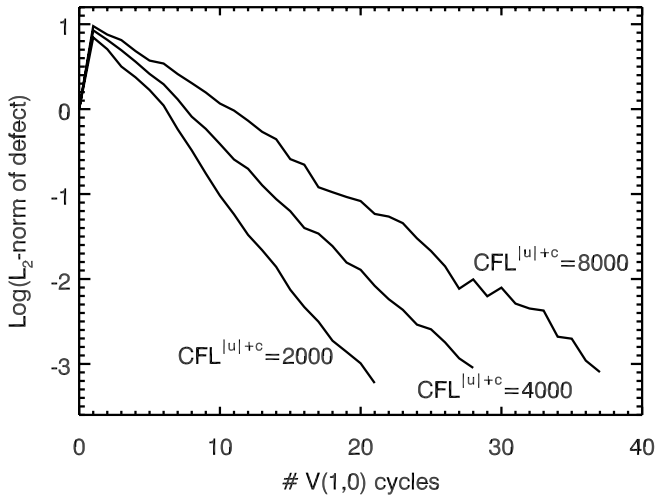

Fig. 12. Multigrid convergence for the first implicit stage of the ESDIRK46-based dual-time scheme in the vortex shedding problem after restarting, using Courant numbers, $CFL^{|u|+c}$, of 2000, 4000, and 8000.

Table 4. Smoothing factor, μ, and two-grid asymptotic convergence factor, $\varrho_{2-\mathrm{grid}}$, as obtained by local Fourier analysis for an implicit stage of the ESDIRK46-MS3I/SGS scheme in the vortex shedding problem setup, compared to the observed asymptotic multigrid convergence rate determined from Fig. 12, as a function of the Courant number.

$CFL^{\lvert u\rvert+c}$	μ	$\varrho_{2-\mathrm{grid}}$	$\varrho_{\mathrm{MG,observed}}$
2000	0.3078	0.7224	0.63
4000	0.3081	0.7632	0.71
8000	0.3082	0.7948	0.77
10000	0.3083	0.8025	–

we determined in Sects. 5.3.3 and 7.1. In terms of smoothing, there is thus no difference between the pseudo-time scheme for steady problems – in which the dual-time terms vanish because $C = 0$ – and a stage of the full dual-time scheme with $C \neq 0$. The implicit smoothers that we have described in this paper should work equally well for *both* cases. This is an important result, because it implies that it should be possible to construct a multigrid scheme based on these smoothers, whose convergence rate should remain constant at ∼0.3, even for time-dependent problems, and regardless of the Courant number.

However, this is not the behavior that we observe with the present variant of the algorithm. According to the results of the two-grid analysis for the dual-time scheme, this discrepancy is caused by an ineffective coarse-grid correction, if the Courant numbers are of order 10^3 or larger. This follows from the unfavorable two-grid convergence factors $\varrho_{2-\mathrm{grid}}$ shown in Table 4, which are far from the ideal convergence rate predicted by the smoothing factor μ. We note, moreover, that up to $CFL^{|u|+c} \sim 8000$, the two-grid factors progressively degrade with an increase in the Courant number. They display (at least qualitatively) the same behavior as the actual multigrid convergence rates measured with the simulation code. This provides an additional indication that it is indeed the coarse-grid correction that is responsible for the unsatisfactory multigrid performance, although $\varrho_{2-\mathrm{grid}}$ ultimately asymptotes at a value of ∼0.8, and does not display evidence of the occurrence of the instability that is observed in practice.

Unfortunately, attempts to use different prolongation, restriction, and discretization operators with the aim of finding a combination that still satisfies Eq. (80), but has a better aggregate coarse grid correction for the dual-time multigrid scheme, have so far been unsuccessful. Despite the evident robustness and efficiency issues for large Courant numbers, we still attempt to judge the extent to which both the current implicit method and possible future improvements are useful for the solution of the present stiff time-dependent problem. For this purpose, we compare the cost of our implicit and explicit runs in what follows.

The computational work required to evaluate the non-linear defects in one implicit stage of the ESDIRK46-based scheme, can be estimated from Eq. (88) using $\Gamma_{\mathrm{cyc}} = 4/3$, as we employ V-type multigrid cycles in the present simulations. Since we have to perform a total of five implicit ESDIRK stages (for which slightly varying numbers of multigrid cycles were required in practice) in order to perform a single ESDIRK46 step, we obtain the work numbers that are given in the fourth column of Table 3. Depending on the Courant number, between 400 and 900 work units were required for a single implicit step. As always, one work unit corresponds to one evaluation of the defect on the finest grid. Since the present test problem does not include a source term, $\boldsymbol{S}$, the work required to compute the defect is dominated by the solution of the Riemann problem (cf. Eqs. (27) and (55), (56)). Hence, one work unit essentially corresponds to one solution of the Riemann problem on the finest grid. The explicit Shu & Osher scheme only needs three defect evaluations (hence three work units) per step, but the number of explicit steps required to cover the same time is up to about four orders of magnitude larger than with the implicit scheme. This results in 23–46 times more defect evaluations for the explicit than the implicit scheme. However, and as discussed in Sect. 7.1, this does not translate into a corresponding speed-up for the implicit scheme, as we have so far neglected the work required to carry out the SGS iteration in the implicit smoother. The measured run times thus show the implicit scheme to be faster than the explicit one by factors of between three and six.

We need to emphasize here, though, that these speed-ups are still very tentative, as they are based on the present sub-optimal coarse-grid correction algorithm for the time-dependent implicit scheme. If the coarse-grid correction algorithm can be improved to the point that the multigrid convergence rate equals the ideal rate predicted by the smoothing factor, μ, and if the (low) computational cost of the coarse-grid correction is insignificantly increased by these improvements, then the speed-up will be given by the last column of Table 3. We conclude that the estimates shown there, provide motivation to justify additional development effort.

To conclude our discussion of the present test problem, we finally show in Fig. 13 the density distribution behind the cylinder at a time $t = 3.8 \times 10^2$, as calculated with the implicit solver after restarting from the aforementioned restart file, and performing 80 implicit time steps using the step size controller with the same parameters as employed in Sect. 7.3. Despite the coarse resolution of our mesh at distances from the center that are larger than about one cylinder diameter, several vortices are visible in the cylinder's wake, which have separated successively from alternate sides of the cylinder's surface.

8. Conclusions

We have described a family of full-coarsening, non-linear multigrid schemes for the solution of the implicitly discretized equations of multidimensional compressible hydrodynamics. We have presented a first step toward an application of these techniques to astrophysical flow problems, in which stiffness due to disparate wave propagation times is present, evaluating the

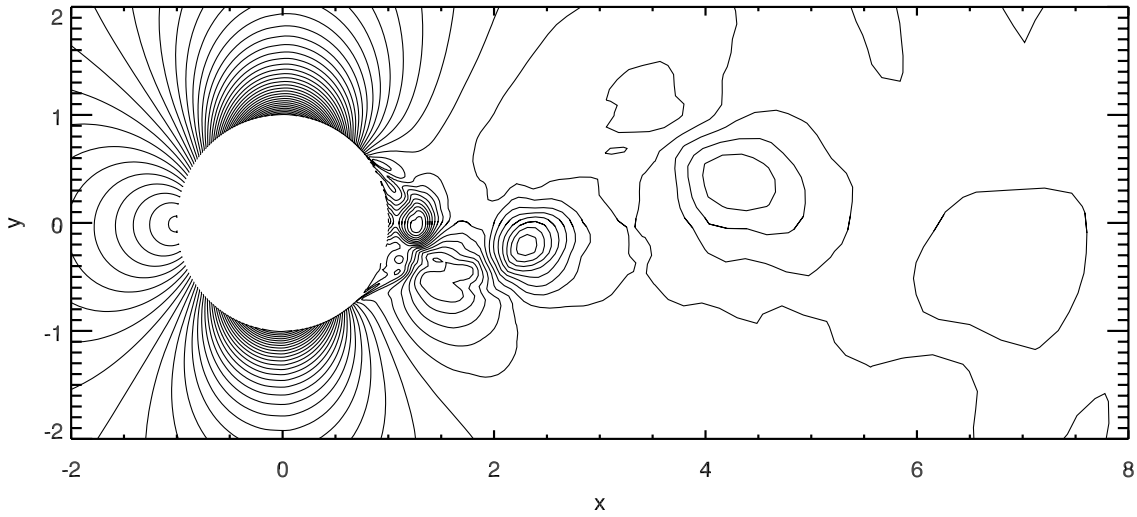

Fig. 13. Density distribution in the vortex shedding problem at time $t = 3.8 \times 10^2$ in the innermost region of the grid, close to the cylinder, computed by restarting and performing 80 time steps with the implicit scheme as described in the text. The outer part of the grid, which extends to 20 cylinder diameters, is not shown. Fifty equidistant contours are used between the density minimum and maximum.

convergence and stability properties of the methods on a number of simple test problems, and identifying algorithmic areas where improvements are required.

Our preferred scheme among this family of multigrid methods is based on a smoother that employs multistage pseudo-time marching with implicit smoothing stages (Rossow 2006, 2007; Swanson et al. 2007). This scheme is remarkably robust for the solution of steady-state problems governed by the Euler equations. Furthermore, for a 2D steady problem, we have found that its efficiency is comparable to the best implicit solvers for conservative discretizations of the compressible Euler equations present in the literature, such as the multiple-semicoarsening multigrid scheme of Darmofal & Siu (1999).

The method has several particular advantages. As the scheme of Darmofal & Siu, our algorithm achieves convergence rates for steady problems that approach the convergence rates of "textbook efficient" multigrid methods. However, since our method does not rely on any form of semicoarsening, it is easier to code and expected to have a lower operation count in three spatial dimensions than the scheme of these authors. The latter requires semicoarsening in each coordinate direction, thus results in a large number of grids on the coarse levels (see Mulder 1989; Darmofal & Siu 1999; Trottenberg et al. 2001). Moreover, our method has significant potential to also be efficient on parallel computers, as the use of point red-black Gauss-Seidel iteration in the multistage implicit smoother (as opposed to the originally suggested symmetric Gauss-Seidel scheme, cf. Rossow 2007) was found not to degrade the multigrid convergence rate appreciably. This should enable scalable parallelization without having to seriously compromise the method's algorithmic efficiency. Furthermore, it should allow one to apply the method to multiblock grids, something that is not easily possible with multigrid approaches based either on semicoarsening or line smoothers.

For our implementation of the method, we found that the initial convergence rates for steady problems are independent of the mesh size and match the theoretical convergence results of local Fourier analysis. We also noticed that our asymptotic multigrid convergence rates are affected by near-boundary residuals. Some additional (probably modest) work will thus be required to make our convergence rates mesh-independent for steady problems.

However, the efficient and robust application of the method to the time-dependent Euler equations will require substantial additional research effort. Somewhat surprisingly, we could not confirm the expectation that the fully time-dependent implicit Euler equations should be easier to solve with multigrid than the steady Euler equations (Brandt 2001). While this is actually true for small (physical) time-step sizes, where the discrete equations are diagonally dominant (cf. Sect. 7.3), we found a dual-time multigrid algorithm to exhibit a degrading convergence rate, and to even diverge in the limit of large time steps (i.e. large Courant numbers). This is unfortunate, since fast and robust convergence is required most urgently in exactly this limit. To our knowledge, these convergence difficulties of dual-time multigrid algorithms for the time-dependent Euler equations have never been previously reported, even in the vast literature that stems from the CFD and engineering communities. For a successful use of the described techniques in future globally-implicit astrophysical codes, the described algorithm must be improved to handle time-dependent problems more robustly and efficiently. To achieve this, it will be imperative to make its convergence rate *independent* of the time step size.

It is presently unclear whether this convergence difficulty is a consequence of the hyperbolic character of the time-dependent Euler equations. After all, the scheme works well for steady problems of these equations, even in the transonic and (moderately) supersonic regimes, where the steady equations contain hyperbolic partitions or are fully hyperbolic, respectively. Local Fourier two-grid analysis predicts a progressive degradation in the efficiency of the coarse-grid correction with increasing Courant numbers. This is in accordance with the observed convergence rates in our numerical experiments. However, the convergence rates obtained from the two-grid analysis ultimately asymptote at values of ~0.8, while, in practice, the algorithm exhibits instability for sufficiently large Courant numbers.

The deeper cause of these problems is not yet understood, and substantial further analysis work, that will have to also include the effects of the boundary conditions (which can be crucial in the hyperbolic case, cf. Brandt 1984), is required to improve both efficiency and robustness of the scheme for time-dependent applications. We believe, however, that this effort will be worth spent, as the potential of the method could lead to a boost of progress in several branches of astrophysical fluid dynamics. This is indicated by both the tentative speed-ups by about half an order of magnitude, that we measured with the

present sub-optimal dual-time multigrid scheme relative to an explicit scheme, and by the much more promising estimated speed-ups that an ideal version of our multigrid algorithm would yield on the same time-dependent, moderately stiff test problem.

One particular branch, where multigrid-based flow solvers could become indispensable in the future, is multidimensional stellar evolution. Here an urgent need exists for numerical methods that are free of the CFL condition, and are capable of treating simultaneously both the deeply subsonic (nearly incompressible), and the transonic and supersonic (compressible) flow regimes. To help develop these methods, the numerical dissipation of our spatial discretization scheme must be made to scale appropriately with the Mach number, as this number approaches zero. This can be done, e.g., by adopting the reformulation of Roe's scheme presented by Rossow (2007). An alternative approach that replaces the dissipation matrix in Roe's original Riemann solver with a preconditioned matrix that yields the right dissipation in the low-Mach number regime (Turkel 1999), is described in Miczek (2008). Miczek's flux-preconditioned extension of our dual-time multigrid code to the low-Mach number regime, however, shows similar convergence difficulties for time-dependent problems as we have discussed here in Sect. 7.5.

Acknowledgements. K. Kifonidis is particularly grateful to Dr. Charles Swanson (NASA Langley Research Center, Hampton), and to Prof. Cord Rossow (DLR, Braunschweig) for many helpful hints and discussions on multigrid techniques, to Prof. Norbert Kroll for the hospitality of DLR's CASE institute, and to Prof. Tomasz Plewa (Florida State University) for his continuous encouragement and support. We would also like to thank the reviewer for helpful comments on the manuscript, F. Miczek, X. Bian, and W. Högele (all MPA, Garching) for their assistance in code testing and development, and F. Miczek and Prof. C.-W. Shu (Division of Applied Mathematics, Brown University) for providing us with versions of their Fourier analysis, and finite difference WENO codes, respectively. This research was supported by the Transregional Collaborative Research Center SFB/TR27 of the Deutsche Forschungsgemeinschaft.

Appendix A: SLIP spatial discretization

The SLIP (symmetric limited positive) scheme makes use of the slope limiter

$$L(a,b) = \frac{1}{2}R(a,b)\,(a+b), \tag{A.1}$$

where the function $R(a,b)$ is given by

$$R(a,b) = 1 - \left|\frac{a-b}{|a|+|b|}\right|^{\nu}, \tag{A.2}$$

and a and b are state differences defined below. The exponent ν is an integer that controls the dissipation of the scheme. For $\nu = 1$, one obtains the well-known minmod limiter, while $\nu = 2$ leads to a less dissipative limiter equivalent to Van Leer's limiter (see Jameson 1995a).

The states $\boldsymbol{Q}^{\mathrm{L}}$ and $\boldsymbol{Q}^{\mathrm{R}}$ are obtained by separate application of the limited interpolation scheme to each component, q, of the conserved variable vector $\boldsymbol{Q}$, leading to

$$q^{\mathrm{L}}_{i+\frac{1}{2}} = q_i + \frac{1}{2}L\left(\Delta q_{i+\frac{3}{2}}, \Delta q_{i-\frac{1}{2}}\right) \tag{A.3}$$

$$q^{\mathrm{R}}_{i+\frac{1}{2}} = q_{i+1} - \frac{1}{2}L\left(\Delta q_{i+\frac{3}{2}}, \Delta q_{i-\frac{1}{2}}\right), \tag{A.4}$$

where

$$\Delta q_{i+\frac{3}{2}} = q_{i+2} - q_{i+1} \tag{A.5}$$

$$\Delta q_{i-\frac{1}{2}} = q_i - q_{i-1}. \tag{A.6}$$

In case one may wish to use the scheme with second-order accuracy but the limiting switched off, one may set $R(a,b) = 1$, so that the left and right states are computed using arithmetic averaging of the slopes.

Appendix B: Butcher coefficients for the ESDIRK23 temporal discretization

The Butcher tableau for a three-stage ESDIRK scheme has the form

$$\begin{array}{c|ccc} 0 & 0 & 0 & 0 \\ a_{21} & a_{22} & 0 \\ b_1 & b_2 & b_3 \\ \hline & b_1 & b_2 & b_3 \\ \hline & b_1 & b_2 & b_3 \end{array} \quad . \tag{B.1}$$

For the second-order accurate ESDIRK23 scheme that we use for temporal discretization of most of the test problems in this paper, the coefficients are (cf. Hosea & Shampine 1996)

$$\begin{array}{c|ccc} 0 & 0 & 0 \\ \theta/2 & \theta/2 & 0 \\ \omega & \omega & \theta/2 \\ \hline & \omega & \omega & \theta/2 \\ \hline & (1-\omega)/3 & (3\omega+1)/3 & \theta/6 \end{array} \quad , \tag{B.2}$$

where $\omega = \sqrt{2}/4$, and $\theta = 2 - \sqrt{2}$.

Appendix C: Verification of the accuracy-order of the discretization schemes

To demonstrate that the discretization schemes, that we use in this work, achieve their formal order of accuracy, we consider the 2D problem of an isentropic vortex in a freestream flow according to Yee et al. (2000). The freestream flow field is characterized by the velocity components $u_\infty = 1$ and $v_\infty = 0$, the pressure $p_\infty = 1$, and the density $\rho_\infty = 1$. An isentropic vortex ($\delta S = 0$) that is expressed in terms of the perturbations

$$(\delta u, \delta v) = \frac{\hat{\beta}}{2\pi}\mathrm{e}^{\frac{1-r^2}{2}}\,(-\bar{y}, \bar{x}), \tag{C.1}$$

$$\delta T = \frac{(\gamma-1)\hat{\beta}^2}{8\gamma\pi^2}\mathrm{e}^{1-r^2}, \tag{C.2}$$

is added to the freestream flow. Here, $T = p/\rho$, and the parameters γ and $\hat{\beta}$ are the ratio of specific heats and the vortex strength, respectively. The coordinates $r^2 = \bar{x}^2 + \bar{y}^2$, $\bar{x} = x - x_0$, and $\bar{y} = y - y_0$, depend on the initial coordinates of the center of the vortex, x_0, and y_0. The initial conditions are given by

$$\rho = (T_\infty + \delta T)^{\frac{1}{\gamma-1}} \tag{C.3}$$

$$\rho u = \rho\,(u_\infty + \delta u) \tag{C.4}$$

$$\rho v = \rho\,(v_\infty + \delta v) \tag{C.5}$$

$$p = \rho^\gamma \tag{C.6}$$

$$\rho E = \frac{p}{\gamma-1} + \frac{1}{2}\rho\left(u^2+v^2\right). \tag{C.7}$$

Since the entire flow field is required to be isentropic, one has, for a perfect gas, $p/(\rho^\gamma) = 1$, hence $T_\infty = 1$.

The solution of this problem is a simple advection of the vortex with the freestream velocity (u_∞, v_∞). The conserved quantities at time t are thus given by Eqs. (C.1)–(C.7), provided that one sets

$$\bar{x} = x - (x_0 + u_\infty t), \tag{C.8}$$

$$\bar{y} = y - (y_0 + v_\infty t). \tag{C.9}$$

To obtain the following results, we employed the parameters, $\hat{\beta} = 0.75$, and $\gamma = 1.4$. Our computational domain is

Table C.1. Error norms of the density field, and derived orders of accuracy for the combinations of dicretization schemes applied to the isentropic vortex problem as described in the text.

		SLIP unlimited scheme			SLIP limited scheme ($\nu = 2$)		
Mesh	Δt	L_2-norm of error	Error ratio	Observed order	L_2-norm of error	Error ratio	Observed order
ESDIRK23 stepping							
16×16	4.00E-01	8.083686E-04	–	–	1.195121E-03	–	–
32×32	2.00E-01	1.829503E-04	4.419	2.14	5.355714E-04	2.231	1.16
64×64	1.00E-01	3.918942E-05	4.668	2.22	1.771569E-04	3.023	1.60
128×128	5.00E-02	8.995759E-06	4.356	2.12	3.806272E-05	4.654	2.22
256×256	2.50E-02	2.165725E-06	4.154	2.05	7.076837E-06	5.378	2.43
512×512	1.25E-02	5.327543E-07	4.065	2.02	1.433251E-06	4.938	2.30
ESDIRK46 stepping							
16×16	4.00E-01	8.305988E-04	–	–	1.225907E-03	–	–
64×64	2.00E-01	3.909528E-05	21.246	4.41	1.826213E-04	6.713	2.75
256×256	1.00E-01	2.151586E-06	18.170	4.18	7.185873E-06	25.414	4.67
1024×1024	5.00E-02	1.315482E-07	16.356	4.03	2.940004E-07	24.442	4.61

Notes. In case of the ESDIRK46-based schemes, the listed order refers solely to the temporal discretization.

the rectangle $[0, 8] \times [-4, 4]$. The vortex is initially centered at $(x_0, y_0) = (4, 0)$. All our simulations are run up to a time $t = 0.4$, in which the vortex is advected 0.4 units to the right.

Simulations were performed on a sequence of uniform Cartesian grids with refinement of the resolution in both space and time. Combinations of either the ESDIRK23 or ESDIRK46 implicit temporal scheme, and the SLIP spatial discretization schemes with and without slope limiting were considered. The non-linear systems that arose in the simulations were iteratively solved by dual-time multigrid. To ensure that the (total) error is dominated by the discretization (and not the iteration) error, the defect was reduced by four orders of magnitude. The L_2 norm of the error in the density field at the final time, $t = 0.4$, was computed for each run, and the ratio of the errors for successive resolutions, and the observed order of accuracy of the discretization schemes were derived. The results for the SLIP/ESDIRK23 combinations are displayed in the upper half of Table C.1, while the lower half of that table shows results obtained for the corresponding SLIP/ESDIRK46 schemes.

The combined SLIP/ESDIRK23 discretization is formally second-order accurate in both space and time, as long as no slope limiter is used in the SLIP scheme. A refinement of the grid and the time step by a factor of two should thus lead to a decrease in the error by (at least) a factor of four. The results for the unlimited scheme in Table C.1 confirm this. Using the Van-Leer-like slope limiting (i.e. $\nu = 2$, in Eq. (A.2)), we observe that the order of accuracy of the SLIP/ESDIRK23 scheme decreases somewhat on very coarse grids. This behavior is actually expected, if the grids are too coarse to resolve even the smooth gradients that characterize the present flow problem. For the higher grid resolutions of practical interest, the scheme, however, regains, and even slightly exceeds, second-order accuracy.

Having shown that the SLIP/ESDIRK23 combination, and thus also the pure SLIP spatial discretization are second-order accurate, we still need to demonstrate that the SLIP/ESDIRK46 scheme achieves fourth-order accuracy in time. For this, we successively refine the time step as before by a factor of two. However, in order not to be affected by the lower (second-order) accuracy of the spatial discretization, we need to refine each spatial dimension by a factor of four. The results in Table C.1 confirm that fourth-order accuracy in time is observed on all considered grids if the slope limiting is switched off. With the Van-Leer-like slope limiter enabled, the results show, again, the expected order reduction caused by the spatial discretization on very coarse grids, and fourth-order accuracy in time, once all spatial gradients are adequately resolved.

Appendix D: Algorithm flow charts

We now summarize, in several flow charts, an entire implicit time step with the standard dual-time scheme. As the flow charts are self-explanatory, we add only some general remarks here.

The computational domain is covered by a sequence of grids G_L (with $L = 1, \ldots, N_L$) that represent different levels of resolution, where G_1 is the finest grid level and G_{N_L} is the coarsest, as shown in Fig. 2. A time step on the finest level is carried out by function Esdirk (cf. Algorithm 1): first, the explicit stage required by an ESDIRK discretization is performed. Then the implicit ESDIRK stages are carried out, and the implicit Riemann problems contained therein are solved. For this, initial values for the solution and right-hand side vectors of Eq. (51) are set up on the finest level, $L = 1$, (see Sect. 5.1), and subsequently the FAS multigrid solver, Mg-fas, is called to provide the iterands of the solution.

The multigrid solver is described by Algorithm 2. Its main input are the current grid level, L, and the current approximation of the solution, $\boldsymbol{U}_L$, and the forcing function, $\boldsymbol{f}_L$, on that level. Additional input values are the number of pre-smoothing steps, n_{pre}, the number of post-smoothing steps, n_{post}, and the parameter n_γ, which determines the cycle type or scheduling, i.e. the order in which the grids are visited. The choices $n_\gamma = 1$ and $n_\gamma = 2$ give the familiar V and W-cycles, respectively[6] (see Fig. 2). The function Smooth is the multistage smoothing algorithm (for which full details are provided in Sect. 5.3), whereas $\tilde{\mathcal{R}}$ and $\mathcal{R}$ are the restriction operators for solutions and defects, respectively (cf. Eq. (83)), and $\mathcal{P}$ is the prolongation operator for solutions, Eq. (82).

We note the recursive nature of Mg-fas, which calls itself to operate on a coarser grid level every time one executes line 11. Lines 6–12 constitute the coarse grid correction, which is skipped, of course, on the coarsest grid $L = N_L$. There, the solution is approximated by performing $n_{\text{pre}} + n_{\text{post}}$ steps with

[6] Another cycle type that we use extensively is the F-cycle, which is a hybrid of the V and W-cycles (cf. Trottenberg et al. 2001). Although it does not correspond to some unique n_γ, the F-cycle has a recursive definition that deviates only slightly from the algorithm given here. It can be implemented as easily as the former two cycle types according to algorithm MG2 of Wesseling (1990, p. 174).

the smoothing algorithm. We also note that the solution of a Riemann problem is required (cf. Eq. (27)) every time the spatial residual, $\boldsymbol{R}(\boldsymbol{U})$, or the defect, $\boldsymbol{D}(\boldsymbol{U})$, need to be evaluated for some given input state $\boldsymbol{U}$. The main place in the algorithm where multiple Riemann solutions are performed is the multistage smoother Smooth.

Algorithm 1 Performs a single implicit ESDIRK time step.

function Esdirk($\Delta t, \boldsymbol{Q}^n, n_{\rm pre}, n_{\rm post}, n_\gamma, \varepsilon$)
  $\boldsymbol{Q}^{(1)} := \boldsymbol{Q}^n$ ▷ Explicit 1st stage, Eq. (46)
  $\boldsymbol{R}^{(1)} := \boldsymbol{R}(\boldsymbol{Q}^{(1)})$ ▷ Eq. (27) (solve Riemann problem)
  for $k := 2, \ldots, s$ **do** ▷ Implicit stages in Eq. (47)
    $L := 1$ ▷ Finest level index
    $\boldsymbol{U}_L := \boldsymbol{Q}^{(k-1)}$ ▷ Solution guess
    $\boldsymbol{f}_L := C\boldsymbol{Q}^{(1)}/\Delta t - C\sum_{l=1}^{k-1} a_{kl}\boldsymbol{R}^{(l)}$ ▷ Eq. (53)
    repeat ▷ Solve implicit systems
      $\boldsymbol{U}_L := $ Mg-fas$(L, \boldsymbol{U}_L, \boldsymbol{f}_L, n_{\rm pre}, n_{\rm post}, n_\gamma)$
      $\boldsymbol{D}_L := $ Defect$(\boldsymbol{U}_L, \boldsymbol{f}_L)$
    until $|\boldsymbol{D}_L| < \varepsilon$ ▷ Iteration tolerance
    $\boldsymbol{R}^{(k)} := \boldsymbol{D}_L - C\boldsymbol{U}_L\Delta t + \boldsymbol{f}_L$ ▷ Eqs. (55), (56)
    $\boldsymbol{Q}^{(k)} := \boldsymbol{U}_L$ ▷ Solution known now
  end for
  $\boldsymbol{Q}^{n+1} := \boldsymbol{Q}^{(s)}$ ▷ Eq. (48)
  return $\boldsymbol{Q}^{n+1}$ ▷ Evolved state
end function

Algorithm 2 Dual-time FAS multigrid solver.

function Mg-fas$(L, \boldsymbol{U}_L, \boldsymbol{f}_L, n_{\rm pre}, n_{\rm post}, n_\gamma)$
  if $(L = N_L)$ **then** ▷ Coarsest level
    $\boldsymbol{U}_L := $ Smooth$^{n_{\rm pre}+n_{\rm post}}(\boldsymbol{U}_L, \boldsymbol{f}_L)$
  else
    $\boldsymbol{U}_L := $ Smooth$^{n_{\rm pre}}(\boldsymbol{U}_L, \boldsymbol{f}_L)$ ▷ Pre-smoothing
    $\boldsymbol{D}_L := $ Defect$(\boldsymbol{U}_L, \boldsymbol{f}_L)$
    $\boldsymbol{D}_{L+1} := \mathcal{R}\boldsymbol{D}_L$ ▷ Restrict defect
    $\boldsymbol{U}_{L+1} := \tilde{\mathcal{R}}\boldsymbol{U}_L$ ▷ Restrict solution
    $\boldsymbol{U}^\star_{L+1} := \boldsymbol{U}_{L+1}$ ▷ Make a copy
    $\boldsymbol{f}_{L+1} := $ Rhs$(\boldsymbol{U}_{L+1}, \boldsymbol{D}_{L+1})$ ▷ Compute rhs
    $\boldsymbol{U}_{L+1} := $ Mg-fas$^{n_\gamma}(L+1, \boldsymbol{U}_{L+1}, \boldsymbol{f}_{L+1}, n_{\rm pre}, n_{\rm post}, n_\gamma)$
    $\boldsymbol{U}_L := \boldsymbol{U}_L + \mathcal{P}\left(\boldsymbol{U}_{L+1} - \boldsymbol{U}^\star_{L+1}\right)$ ▷ Add correction
    $\boldsymbol{U}_L := $ Smooth$^{n_{\rm post}}(\boldsymbol{U}_L, \boldsymbol{f}_L)$ ▷ Post-smoothing
  end if
  return $\boldsymbol{U}_L$
end function

function Defect$(\boldsymbol{U}, \boldsymbol{f})$
  $\boldsymbol{R} := \boldsymbol{R}(\boldsymbol{U})$ ▷ Eq. (27) (solve Riemann problem)
  $\boldsymbol{D} := C\boldsymbol{U}/\Delta t + \boldsymbol{R} - \boldsymbol{f}$ ▷ Eqs. (55), (56)
  return $\boldsymbol{D}$
end function

function Rhs$(\boldsymbol{U}, \boldsymbol{D})$
  $\boldsymbol{R} := \boldsymbol{R}(\boldsymbol{U})$ ▷ Eq. (27) (solve Riemann problem)
  $\boldsymbol{f} := C\boldsymbol{U}/\Delta t + \boldsymbol{R} - \boldsymbol{D}$ ▷ Eqs. (55), (56)
  return $\boldsymbol{f}$
end function

function Smooth$(\boldsymbol{U}, \boldsymbol{f})$ ▷ Multistage smoother for Mg-fas
  $\boldsymbol{U}^{\{0\}} := \boldsymbol{U}$ ▷ Eq. (60)
  for $k := 1, \ldots, K$ **do** ▷ Loop over stages in Eq. (61)
    $\boldsymbol{D}^{\{k-1\}} := $ Defect$\left(\boldsymbol{U}^{\{k-1\}}, \boldsymbol{f}\right)$
    $\delta\boldsymbol{U} := -\Delta\tau\boldsymbol{D}^{\{k-1\}}$ ▷ Eq. (71)
    $\overline{\delta\boldsymbol{U}} := $ Gauss-Seidel$(\boldsymbol{P}^{-1}, \delta\boldsymbol{U})$ ▷ Solve Eq. (73)/(74)
    $\boldsymbol{U}^{\{k\}} := \boldsymbol{U}^{\{0\}} + \alpha_k\overline{\delta\boldsymbol{U}}$ ▷ Eq. (70)
  end for
  return $\boldsymbol{U}^{\{K\}}$ ▷ Eq. (62)
end function

References

Almgren, A. S., Bell, J. B., Rendleman, C. A., & Zingale, M. 2006a, ApJ, 637, 922
Almgren, A. S., Bell, J. B., Rendleman, C. A., & Zingale, M. 2006b, ApJ, 649, 927
Almgren, A. S., Bell, J. B., Nonaka, A., & Zingale, M. 2008, ApJ, 684, 449
Anderson, D. A., Tannehill, J. C., & Pletcher, R. H. 1984, Computational Fluid Mechanics and Heat Transfer (Washington: Hemisphere Publ.)
Balsara, D. 2001, JQSRT, 69, 671
Bijl, H., Carpenter, M. H., Vatsa, V. N., & Kennedy, C. A. 2002, J. Comput. Phys., 179, 313
Brandt, A. 1977, Math. Comput., 31, 333
Brandt, A. 1984, Multigrid techniques: 1984 Guide with applications to fluid dynamics (Bonn: Gesellschaft für Mathematik und Datenverarbeitung)
Brandt, A. 2001, in Multigrid, eds. U. Trottenberg, C. W. Oosterlee, & A. Schüller (London: Academic Press)
Brandt, A., & Dym, J. 1996, in Proc. Seventh Copper Mountain Conference on Multigrid Methods, eds. N. Duane Melson, T. A. Manteuffel, S. F. McCormick, & C. C. Douglas, NASA Conf. Pub., 3339 (Langley: National Aeronautics and Space Administration), 97
Caflisch, R. E., Jin, S., & Russo, G. 1997, SIAM J. Numer. Anal., 34, 246
Camenzind, M. 2005, in Cosmic Magnetic Fields, eds. R. Wielebinski, & R. Beck, Lect. Notes Phys. (Berlin: Springer Verlag), 664, 255
Colella, P. 1990, J. Comput. Phys., 87, 171
Darmofal, D. L., & Siu, K. 1999, J. Comput. Phys., 151, 728
Dearborn, D. S. P., Lattanzio, J. C., & Eggleton, P. P. 2006, ApJ, 639, 405
Dick, E., & Riemslagh, K. 1995, J. Comput. Appl. Math., 59, 339
Dumbser, M., Enaux, C., & Toro, E. 2008, J. Comput. Phys., 227, 3971
Eggleton, P. P., Bazan, G., Cavallo, R. M., et al. 2002, in BAAS, 35, 144.09
Fabiani Bendicho, P., Trujillo Bueno, J., & Auer, L. 1997, A&A, 324, 161
Fryxell, B. A., Woodward, P. R., Colella, P., & Winkler, K.-H. 1986, J. Comput. Phys., 63, 283
Gawryszczak, A., Guzman, J., Plewa, T., & Kifonidis, K. 2010, A&A, 521, A38
Glasner, S. A., Livne, E., & Truran, J. W. 2007, ApJ, 665, 1321
Glatzmaier, G. A., & Roberts, P. H. 1995, Nature, 377, 203
Guillard, H., & Viozat, C. 1999, Comput. Fluids, 28, 63
Hackbusch, W. 1985, Multi-Grid Methods and Applications, Springer Series in Computational Mathematics (Berlin: Springer), 4
Hairer, E., & Wanner, G. 1991, Solving Ordinary Differential Equations II, Springer Series in Computational Mathematics (Berlin: Springer)
Heger, A., Woosley, S. E., & Spruit, H. C. 2005, ApJ, 626, 350
Höflich, P., & Stein, J. 2002, ApJ, 568, 779
Hosea, M. E., & Shampine, L. F. 1996, Appl. Num. Math., 20, 21
Hujeirat, A., & Rannacher, R. 2001, New Astron. Rev., 45, 425
Jameson, A. 1983, Appl. Math. Comput., 13, 327
Jameson, A. 1986, in Proc. Second European Conference on Multigrid Methods, eds. W. Hackbusch, & U. Trottenberg (Berlin: Springer), Lect. Notes Math., 1228, 166
Jameson, A. 1991, AIAA Paper, 91
Jameson, A. 1995a, Int. J. Comput. Fluid D, 4, 171
Jameson, A. 1995b, Int. J. Comput. Fluid D, 5, 1
Jameson, A. 2003, in 2003 AFOSR Work Shop on Advances and Challenges in Time-Integration of PDEs
Jothiprasad, G., Mavriplis, D. J., & Caughey, D. A. 2003, J. Comput. Phys., 191, 542
Kennedy, C. A., & Carpenter, M. H. 2001, NASA Technical Memorandum TM-2001-211038
Kercek, A., Hillebrandt, W., & Truran, J. W. 1999, A&A, 345, 831
Khalil, M., & Wesseling, P. 1992, J. Comput. Phys., 98, 1
Kifonidis, K., Plewa, T., Scheck, L., Janka, H.-T., & Müller, E. 2006, A&A, 453, 661
Knoll, D. A., & Keyes, D. E. 2004, J. Comput. Phys., 193, 357
Koren, B., & Hemker, P. W. 1991, Appl. Num. Math., 7, 309
Lee, D., Xia, G., Daley, C., et al. 2011, Ap&SS, 336, 157
LeVeque, R. J. 1998, in Computational Methods for Astrophysical Fluid Flow, eds. O. Steiner, & A. Gautschy (Berlin: Springer)
Lin, D. J., Bayliss, A., & Taam, R. E. 2006, ApJ, 653, 545
Meakin, C. A., & Arnett, D. 2006, ApJ, 637, L53
Melson, N. D., Sanetrik, M. D., & Atkins, H. L. 1993, in Sixth Copper Mountain Conference on Multigrid Methods, NASA Conf. Publ., 3224, Part 2, eds. N. D. Melson, T. A. Manteuffel, & S. F. McCormick, 423
Miczek, F. 2008, Diploma thesis, Technische Universität München
Miniati, F., & Colella, P. 2007, J. Comput. Phys., 224, 519
Mocák, M., Müller, E., Weiss, A., & Kifonidis, K. 2008, A&A, 490, 265
Mocák, M., Müller, E., Weiss, A., & Kifonidis, K. 2009, A&A, 501, 659
Mocák, M., Campbell, S. W., Müller, E., & Kifonidis, K. 2010, A&A, 520, A114

Mohr, M., & Wienands, R. 2004, Comput. Visual. Sci., 7, 129
Mulder, W. A. 1989, J. Comput. Phys., 83, 303
Nishikawa, H., & van Leer, B. 2003, J. Comput. Phys., 190, 52
Obergaulinger, M., & Janka, H. 2011, A&A, submitted [arXiv:1101.1198]
Pember, R. 1993, SIAM J. Sci. Comput., 14, 824
Pierce, N. A., & Giles, M. B. 1997, J. Comput. Phys., 136, 425
Press, W. H., Teukolsky, S. A., Vetterling, W. T., & Flannery, B. P. 1992, Numerical Recipes in FORTRAN, The Art of Scientific Computing, second edition (Cambridge: Cambridge University Press)
Rizzi, A., & Viviand, H. 1981, Numerical methods for the computation of inviscid transonic flows with shock waves, A GAMM workshop (Braunschweig: Vieweg)
Roberts, T. W., & Swanson, R. C. 2005, AIAA Paper, 5229
Roe, P. L. 1981, J. Comput. Phys., 43, 357
Ronchi, C., Iacono, R., & Paolucci, P. S. 1996, J. Comput. Phys., 124, 93
Rossow, C.-C. 2006, AIAA Journal, 44, 345
Rossow, C.-C. 2007, J. Comput. Phys., 220, 879
Shu, C.-W. 1997, ICASE Report No. 97-65, NASA/CR-97-206253
Shu, C.-W., & Osher, S. 1988, J. Comput. Phys., 77, 439
Sod, G. A. 1978, J. Comput. Phys., 27, 1
Swanson, R. C., & Turkel, E. 1992, J. Comput. Phys., 101, 292
Swanson, R. C., & Turkel, E. 1997, NASA Technical Paper, 3631
Swanson, R. C., Turkel, E., & Rossow, C.-C. 2007, J. Comput. Phys., 224, 365
Thomas, J. L., Diskin, B., & Brandt, A. 2003, Ann. Rev. Fluid Mech., 35, 317
Thompson, J. F., & Warsi, Z. U. A. 1982, J. Comput. Phys., 47, 1
Thompson, J. F., Warsi, Z. U. A., & Wayne Mastin, C. 1985, Numerical Grid Generation (New York: North-Holland)
Toro, E. F. 1997, Riemann solvers and numerical methods for fluid dynamics – A practical introduction (Berlin: Springer)
Tóth, G., de Zeeuw, D. L., Gombosi, T. I., & Powell, K. G. 2006, J. Comput. Phys., 217, 722
Trottenberg, U., Oosterlee, C. W., & Schüller, A. 2001, Multigrid (London: Academic Press)
Turkel, E. 1999, Ann. Rev. Fluid Mech., 31, 385
Tweedt, D. L., Chima, R. V., & Turkel, E. 1997, NASA Technical Memorandum 113120
Van Leer, B. 1985, in Large Scale Computations in Fluid Mechanics, eds. B. Enquist, S. Osher, & R. Somerville, Lectures in Applied Mathematics II (Providence, Rhode Island: American Mathematical Society), 327
Van Leer, B., Tai, C. H., & Powell, K. G. 1989, AIAA Paper, 89
Viallet, M., Baraffe, I., & Walder, R. 2011, A&A, 531, A86
Vinokur, M. 1974, J. Comput. Phys., 14, 105
Wesseling, P. 1990, An introduction to multigrid methods (Chichester: Wiley)
Wesseling, P., & Oosterlee, C. W. 2001, J. Comp. Appl. Math., 128, 311
Yavneh, I. 1998, SIAM J. Sci. Comput., 19, 1682
Yee, H. C., Vinokur, M., & Djomehri, M. J. 2000, J. Comput. Phys., 162, 33
Zingale, M., Almgren, A. S., Bell, J. B., Nonaka, A., & Woosley, S. E. 2009, ApJ, 704, 196