INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN FLUIDS
Int. J. Numer. Meth. Fluids 2008; **58**:493–526
Published online 25 January 2008 in Wiley InterScience (www.interscience.wiley.com). DOI: 10.1002/fld.1730

Low-cost implicit schemes for all-speed flows on unstructured meshes

T. Kloczko[1, 2, *, †], C. Corre[1] and A. Beccantini[2]

[1]*Laboratoire SINUMEF, ENSAM, 151 Bd de l'Hôpital 75013 Paris, France*
[2]*CEA Saclay, Laboratoire LTMF, 91191 Gif-sur-Yvette Cedex, France*

SUMMARY

Matrix-free implicit treatments are now commonly used for computing compressible flow problems: a reduced cost per iteration and low-memory requirements are their most attractive features. This paper explains how it is possible to preserve these features for all-speed flows, in spite of the use of a low-Mach preconditioning matrix. The proposed approach exploits a particular property of a widely used low-Mach preconditioner proposed by Turkel. Its efficiency is demonstrated on some steady and unsteady applications. Copyright © 2008 John Wiley & Sons, Ltd.

Received 14 June 2007; Revised 20 November 2007; Accepted 20 November 2007

KEY WORDS: finite-volume methods; compressible flow; Navier–Stokes; Euler flows; viscous flows; all-speed flows; low-Mach preconditioning; matrix-free implicit treatment

1. INTRODUCTION

The present paper is devoted to the design of efficient implicit schemes for computing all-speed flows on unstructured grids using a preconditioned density-based flow solver. The computation of steady flows can be considered *efficient* if a steady state is reached for a reduced computational time and also if a low-memory storage is used in this process; for complex industrial applications involving grids with very large numbers of points this latter requirement may become especially critical. The same comment holds of course for unsteady flows, now classically computed by a dual time-stepping approach in which a steady state with respect to the dual time must be reached at each physical time step ideally using a low amount of both computational time and memory storage. Designing an efficient implicit treatment means solving a multi-objective optimization

*Correspondence to: T. Kloczko, INRIA Sophia-Antipolis, SMASH project, 2004 route des Lucioles, 06902 Sophia Antipolis Cedex, France.
†E-mail: thibaud.kloczko@sophia.inria.fr

Copyright © 2008 John Wiley & Sons, Ltd.

problem in which one looks for the simultaneous minimization of the computational time involved in the convergence to steady (physical or dual) state and of the associated storage requirement. The minimization of computational cost can be performed by considering a two-objective problem since the global computational cost spent to reach a steady-state is a combination of *intrinsic efficiency* (i.e. the number of iterations needed to reach this steady-state) and *unit cost* per iteration. Maximization of intrinsic efficiency and minimization of unit cost are usually conflicting objectives so that the multi-objective optimization problem admits a set of optimal trade-off solutions. It is customary to use implicit schemes to achieve fast convergence to steady state: the intrinsic efficiency of an implicit scheme depends both on the implicit stage coupled with the chosen explicit scheme and the solution method applied to the linear system associated with this implicit stage. For instance, directly solved block-implicit schemes [1, 2] minimize the number of iterations needed to reach steady state; hence, provide maximal intrinsic efficiency, at the expense of high unit cost and memory requirement. A way to reduce unit cost and memory requirement is to develop implicit treatments that do not rely on full flux Jacobian matrices usually introduced when linearizing the explicit stage numerical fluxes in order to derive the implicit part of the scheme. The simplifications that can be introduced in the explicit stage linearization are likely to induce a loss of intrinsic efficiency which will have to be balanced through a lower unit cost, so as to preserve the global efficiency. Following this line of idea, Pulliam and Chaussee proposed in 1981 a diagonalized implicit treatment applied to alternate direction factorization method [3], which was next extended to the case of preconditioned schemes within a dual-time framework and successfully applied to the computation of unsteady low-Mach number flows on structured grids [4–7].

So-called matrix-free implicit methods provide even lower unit cost, at the expense of a poorer intrinsic efficiency, but with the advantage of very low-memory requirement. A well-known prototype of such methods is the implicit residual smoothing technique initially introduced in [8] for the Lax-Wendroff scheme and made popular by Jameson and Baker [9]: the basic idea of the residual smoothing technique applied to the Lax-Wendroff implicit scheme is to replace the square of the fluxes Jacobian matrices appearing in the implicit stage with their respective spectral radius (an operation made possible without altering the unconditional linear stability of the scheme thanks to the definite positive character of these dissipation matrices). The linear system associated with this simplified implicit stage is purely scalar so that it can be solved for a reduced unit cost and with very low-memory requirements; naturally, the price to pay for this simplification from block to scalar is a loss of intrinsic efficiency. However, this latter is usually balanced by a low unit cost, which eventually results in a global efficiency as good as that of the original block treatment, with the benefit of a reduced memory storage. Shortly after the introduction of this spectral radius simplification in [8] Jameson and Baker demonstrated in [9] the same matrix-free implicit stage could be successively coupled with the explicit Jameson, Schmidt, Turkel's scheme [10]: despite the lack of consistency between the explicit and implicit stage, the resulting scheme had good stability properties and was much more efficient than a purely explicit scheme for a modest extra-cost due to the very simple form of this matrix-free implicit stage.

Similar ideas made their way in the context of upwind schemes: a first contribution can be found in the works of Jameson and Turkel [11] and Jameson and Yoon [12] where the absolute values of Jacobian matrices appearing in a block implicit stage are replaced with their respective spectral radius, taking advantage here again of the definite positive character of these dissipation matrices. It was next remarked [13] the implicit stage could be further simplified by

Copyright © 2008 John Wiley & Sons, Ltd.

DOI: 10.1002/fld

introducing time increments of the flux vectors instead of products of flux Jacobian matrices and time increments of the conserved variables and relaxing these new unknowns when solving the resulting matrix-free implicit stage. Using simultaneously spectral radius simplification and flux-vector increment relaxation, Löhner *et al.* have successfully developed a matrix-free implicit method for solving the 3D Navier–Stokes equations on unstructured grids [14], which has been further extended to unsteady flows [15] and all-speed flows using a low-Mach preconditioning technique [16, 17].

Since the intrinsic efficiency of matrix-free methods is quite poor, it is absolutely crucial to make the unit cost of these methods as low as possible in order to preserve their global efficiency. When computing compressible flows with no preconditioning on unstructured grids, point-Jacobi or point Gauss–Seidel methods offer such a very low cost per iterations, as well as reduced memory requirements. When a low-Mach preconditioning matrix P_c, such as the Turkel low-Mach preconditioning matrix [18], is introduced in the discretization, it could compromise the matrix-free nature of the implicit stage solution. In fact, as pointed out in [19–21], P_c is such that quantities of the form $(I_d+\alpha P_c)^{-1}$, with I_d the identity matrix and α a scalar coefficient, can be simply expressed as a linear combination of vectors. In the present work, these properties are combined with the matrix-free implicit stage developed in [14] and a solution based on point-relaxation techniques to yield a low-cost matrix-free formulation for computing all-speed flows on unstructured grids.

The paper is organized as follows. Section 2 describes the numerical methods used in the present study to compute compressible flows on unstructured grids, including low-Mach flows; the last part of this section details an inexpensive implicit treatment valid for all-speed flows, derived by taking advantage of the preconditioning matrix P_c being idempotent. Section 3 contains a Fourier analysis of the proposed low-cost treatment and a comparison with conventional block implicit methods in order to get some *a priori* knowledge of the performances to be expected when applying the low-cost implicit scheme to low-Mach flow computations. Section 4 is devoted to the presentation of results obtained for such computations using the low-cost implicit scheme developed in this work in order to demonstrate its practical efficiency.

2. NUMERICAL SCHEMES

2.1. General principles

Let us consider the three-dimensional compressible Navier–Stokes equations:

$$\frac{\partial w}{\partial t}+\nabla\cdot F^E=\nabla\cdot F^V \tag{1}$$

where $w=(\rho,\rho u,\rho v,\rho w,\rho E)^{\mathrm{T}}$ is the vector of the conserved variable with ρ the density, u, v, w the Cartesian components of the velocity vector and E the total energy, t is the physical time, $F^E=(f^E,g^E,h^E)$ and $F^V=(f^V,g^V,h^V)$ are, respectively, the convective and viscous flux vectors. A time-accurate solution of (1) can be efficiently computed for flows at all speed by solving:

$$P_c^{-1}\frac{\partial w}{\partial \tau}+\frac{\partial w}{\partial t}+\nabla\cdot F^E=\nabla\cdot F^V \tag{2}$$

Copyright © 2008 John Wiley & Sons, Ltd.

DOI: 10.1002/fld

where τ is a pseudo or dual time and P_c is a preconditioning matrix which takes, in the present work, the form proposed by Turkel [18], detailed in Section 2.4. These governing equations can also be expressed in the following integral form:

$$P_c^{-1}\frac{\partial}{\partial\tau}\int_\Omega w\,\mathrm{d}\Omega+\frac{\partial}{\partial t}\int_\Omega w\,\mathrm{d}\Omega+\int_\Omega\nabla\cdot(F^E-F^V)\,\mathrm{d}\Omega=0$$

where Ω denotes a control volume. Making use of the Green–Gauss theorem yields the alternate integral form:

$$P_c^{-1}\frac{\partial}{\partial\tau}\int_\Omega w\,\mathrm{d}\Omega+\frac{\partial}{\partial t}\int_\Omega w\,\mathrm{d}\Omega+\int_{\partial\Omega}(F^E-F^V)\cdot\underline{n}=0 \tag{3}$$

where $\partial\Omega$ is the boundary of cell Ω and $\underline{n}$ denotes the outward unit normal vector. Applying (3) on a given cell Ω_i of a general unstructured grid, introducing the average value $\bar{w}$ of w over the cell and decomposing the flux balance as the sum of fluxes through each face Γ_k of cell Ω_i leads to:

$$(P_c^{-1})_i\frac{\partial\bar{w}_i}{\partial\tau}|\Omega_i|+\frac{\partial\bar{w}_i}{\partial t}|\Omega_i|+\sum_k\int_{\Gamma_{i,k}}(F^E-F^V)\cdot\underline{n}=0 \tag{4}$$

where $|\Omega_i|$ is the volume of the ith grid cell Ω_i and $\Gamma_{i,k}$ is the kth interface of this cell. The preconditioned dual-time finite-volume approach considered in the present study drives (4) to steady state with respect to τ using a first-order approximation for the dual-time derivative (which will vanish at steady state anyway), a second-order implicit approximation for the physical time derivative and a numerical flux $\mathscr{H}=\mathscr{H}^E-\mathscr{H}^V$ to approximate the normal physical flux $(F^E-F^V)\cdot\underline{n}$ through a face $\Gamma_{i,k}$; moreover, since the space accuracy is limited to second order in the present work, cell-centered values w_i can be used instead of cell-averaged values without altering the global accuracy of the method. The resulting finite-volume scheme reads:

$$(P_c^{-1})_i^{n,m}\frac{\Delta w_i^{n,m}}{\Delta\tau_i^{n,m}}+\frac{\frac{3}{2}(w_i^{n,m}-w_i^n)-\frac{1}{2}\Delta w_i^{n-1}}{\Delta t}+\frac{1}{|\Omega_i|}\sum_k(\mathscr{H}_{i,k}^E-\mathscr{H}_{i,k}^V)^{n,m+1}S_{i,k}=0 \tag{5}$$

where m is the pseudo-iteration (on dual-time) counter, n is the time step counter, $\Delta w^{n,m}=w^{n,m+1}-w^{n,m}$, $\Delta w^{n-1}=w^n-w^{n-1}$ and index 'i,k' on the numerical fluxes refers to the center of the kth interface of the ith grid cell, the surface of which is denoted $S_{i,k}$. Pseudo-time marching leads to a steady solution $w^{n+1}=w^{n,m+1}=w^{n,m}$ that satisfies:

$$\mathscr{R}_i^{n+1}=\frac{\frac{3}{2}(w_i^{n+1}-w_i^n)-\frac{1}{2}\Delta w_i^{n-1}}{\Delta t}+\frac{1}{|\Omega_i|}\sum_k(\mathscr{H}_{i,k}^E-\mathscr{H}_{i,k}^V)^{n+1}S_{i,k}=0 \tag{6}$$

and approximates therefore at second-order in time the compressible Navier–Stokes equations (1). The space accuracy depends on the choice of the numerical flux formula $\mathscr{H}=\mathscr{H}^E-\mathscr{H}^V$. In the present work, inviscid fluxes are computed using classical inviscid numerical flux formulae for $\mathscr{H}^E$ (e.g. Roe, Rusanov, AUSM+, HLL, etc.), extended to higher order using the primitive variable reconstruction of Barth and Jespersen [22] with a slope limitation based on the technique proposed by Venkatakrishnan [23]. Since the present work deals with the computation of all-speed flows

Copyright © 2008 John Wiley & Sons, Ltd.

DOI: 10.1002/fld

using such density-based solvers, the numerical dissipation associated with each of these schemes is corrected so as to take into account the low Mach number preconditioning appearing in (2). For the Roe scheme first applied to system (1), the inviscid numerical flux in the direction normal to the face Γ_{ik} reads:

$$\mathscr{H}^E_{i,k} = \tfrac{1}{2}(F^E(w_k^{\mathrm{L}})\cdot \underline{n}_{i,k} + F^E(w_k^{\mathrm{R}})\cdot \underline{n}_{i,k}) + \tfrac{1}{2} Q^E_{i,k}(w_k^{\mathrm{L}} - w_k^{\mathrm{R}}) \tag{7}$$

where $w_k^{\mathrm{L/R}}$ are the reconstructed states, respectively, on the left and right side of the interface k. More precisely:

$$w_k^{\mathrm{L}} = w_i + \phi_i \nabla w_i \cdot \underline{r}_{ik}$$

where $\underline{r}_{ik}$ denotes the vector extending from the cell center i to the center of the interface $\Gamma_{i,k}$, the gradient ∇w_i is computed at the cell center i using a least-square formula applied on a prescribed spatial support and ϕ_i is the slope limiter, computed according to Barth and Jespersen's formula as modified by Venkatakrishnan. Similarly:

$$w_k^{\mathrm{R}} = w_{o(i,k)} + \phi_{o(i,k)} \nabla w_{o(i,k)} \cdot \underline{r}_{o(i,k)k}$$

where the index $o(i,k)$ refers to the cell sharing the interface $\Gamma_{i,k}$ with cell i. The numerical dissipation matrix Q^E associated with the Roe scheme is given by

$$Q^E_{i,k} = |(A^E n_x + B^E n_y + C^E n_z)_{i,k}| = |(J^E_\perp)_{i,k}| \tag{8}$$

where $\underline{n}_{i,k} = (n_x, n_y, n_z)_{i,k}$ and the Jacobian matrices $A^E = \mathrm{d}f^E/\mathrm{d}w$, $B^E = \mathrm{d}g^E/\mathrm{d}w$, $C^E = \mathrm{d}h^E/\mathrm{d}w$ are evaluated using the Roe average [24]. If the Roe scheme is applied to the low-Mach system (2), its dissipation matrix must take into account the preconditioning matrix P_c and is thus given by

$$Q^E_{i,k} = (P_c^{-1})_{i,k} |(P_c J^E_\perp)_{i,k}| \tag{9}$$

Similarly, while the scalar Rusanov scheme's dissipation relies on the spectral radius of the normal Jacobian matrix $\rho^E_\perp = \rho(J^E_\perp)$ for compressible flow problems governed by (1), this dissipation is based on $P_c^{-1}\rho(P_c J^E_\perp) = P_c^{-1}\tilde{\rho}^E_\perp$ and hence becomes a matrix dissipation when dealing with low Mach number flow evolutions governed by (2). Low Mach number formulations for AUSM+ and HLL schemes can be, respectively, found in [17, 25]. Viscous fluxes are approximated using a linearly exact extension of the diamond method of Noh [26].

A key point in making the dual-time approach efficient is to treat (5) in an implicit way; this means the steady solution (6) is obtained after a reduced number of pseudo-iterations by solving:

$$(P_c^{-1})_i^{n,m} \frac{\Delta w_i^{n,m}}{\Delta \tau_i^{n,m}} + \frac{3}{2}\frac{\Delta w_i^{n,m}}{\Delta t} + \frac{1}{|\Omega_i|}\sum_k (\Delta \mathscr{H}^{(i)}_{i,k})^{n,m} S_{i,k} = -\mathscr{R}_i^{n,m} \tag{10}$$

where $\Delta(\mathscr{H}^{(i)})^{n,m} = (\mathscr{H}^{(i)})^{n,m+1} - (\mathscr{H}^{(i)})^{n,m}$ and $\mathscr{H}^{(i)} = \mathscr{H}^{E(i)} - \mathscr{H}^{V(i)}$ denotes the numerical flux formula retained in the implicit stage, which is not necessarily the same than the one used in the explicit stage $\mathscr{R}$. Indeed, since the numerical approach used to compute the left-hand side (LHS) of Equation (10) does not affect the (space and time) accuracy of the method (provided that the convergence on m is reached), the numerical schemes used to compute the LHS (10) are not necessarily consistent with the ones used to compute the right-hand side (RHS), which determine the space accuracy of the discrete solution. The basic motivation for a choice of a numerical flux

Copyright © 2008 John Wiley & Sons, Ltd.

DOI: 10.1002/fld

$\mathscr{H}^{(i)}$ simpler than its explicit counterpart $\mathscr{H}$ lies in an expected reduction of the computational cost spent in solving at each physical time step the linear algebraic system associated with the implicit stage. This unit cost reduction is achieved at the expense of the intrinsic efficiency which is clearly optimal when the implicit stage remains consistent with the explicit one (fully implicit scheme). For instance, it is now very common practice to retain in the implicit stage a first-order expression of the numerical flux formula (Roe, Rusanov, AUSM+, HLL, etc.) used at second or third order in the explicit stage. When performed on structured grids, such a choice simplifies the algebraic systems associated with the implicit stage from pentadiagonal to tridiagonal (see [27] for instance) thus offering a significant reduction of the unit computational cost. At the same time the implicit first-order/explicit third-order combination provides a poorer damping of the error than a fully implicit third-order scheme [28]. However, the unit cost reduction offered by the implicit stage simplification is such that it largely balances the loss of efficient intrinsic efficiency and such a non-consistent implicit strategy is now widely adopted (see, for instance, codes such as OVERFLOW or CFL3D). Further simplifications of the implicit stage can be performed leading to so-called matrix-free implicit schemes, as developed in particular by Löhner and co-workers for computing compressible and low-Mach number flows on unstructured grids [14, 15, 17] on the basis of previous implicit stage simplifications proposed in [12, 13]. The next section is devoted to the derivation of such a simplified low-cost implicit stage for the low-Mach Navier–Stokes equations, and to the description of solution methods preserving its reduced computational cost.

2.2. *Unstructured grid low-cost implicit formulation*

2.2.1. Simplified inviscid implicit stage. The early proposal of Jameson and Yoon [12] can be reinterpreted as choosing the first-order Rusanov numerical flux for the inviscid numerical flux $\mathscr{H}^{E(i)}$ appearing in the implicit stage, whatever the higher-order inviscid numerical flux formula used in the explicit stage; namely, (10) is applied for all-speed flow computations with:

$$\mathscr{H}^{E(i)}_{i,k} = \tfrac{1}{2}(F^E_i \cdot \underline{n}_{i,k} + F^E_{o(i,k)} \cdot \underline{n}_{i,k}) + \tfrac{1}{2}(P_c^{-1}\rho(P_c J^E_\perp))_{i,k}(w_i - w_{o(i,k)}) \tag{11}$$

where $F^E_i = (f^E(w_i), g^E(w_i), h^E(w_i))$ and similarly for $F^E_{o(i,k)}$ computed using the state at the center of cell $o(i,k)$; meanwhile, the higher-order numerical flux used in the explicit stage would be for instance the Roe-MUSCL formula defined by (7). The linearization of the time increment of this implicit numerical flux leads to the following approximation:

$$(\Delta\mathscr{H}^{E(i)})^{n,m}_{i,k} = \tfrac{1}{2}((\Delta F^E_i)^{n,m} \cdot \underline{n}_{i,k} + (\Delta F^E_{o(i,k)})^{n,m} \cdot \underline{n}_{i,k}) + \tfrac{1}{2}(P_c^{-1}\tilde{\rho}^E_\perp)^{n,m}_{i,k}(\Delta w^{n,m}_i - \Delta w^{n,m}_{o(i,k)})$$

with the preconditioned inviscid spectral radius defined as $\tilde{\rho}^E_\perp = \rho(P_c J^E_\perp)$. The resulting contribution of the inviscid numerical flux balance to the LHS of (10) reads:

$$\sum_k (\Delta\mathscr{H}^{E(i)}_{i,k})^{n,m} S_{i,k} = \frac{1}{2}\left[\sum_k (\Delta F^E_{o(i,k)})^{n,m} \cdot \underline{n}_{i,k} + \sum_k (P_c^{-1}\tilde{\rho}^E_\perp)^{n,m}_{i,k}(\Delta w^{n,m}_i - \Delta w^{n,m}_{o(i,k)})\right] S_{i,k} \tag{12}$$

since $\sum_k (\Delta F^E_i)^{n,m} \cdot \underline{n}_{i,k} S_{i,k} = (\Delta F^E_i)^{n,m} \cdot (\sum_k \underline{n}_{i,k} S_{i,k}) = 0$.

2.2.2. Simplified viscous implicit stage. For the sake of presentation simplicity, the derivation of the viscous implicit stage is performed in the 2D case. The time increment of the physical viscous

Copyright © 2008 John Wiley & Sons, Ltd.

DOI: 10.1002/fld

flux normal to a face can be linearized as follows:

$$\begin{aligned}\Delta(F^V)^{n,m}\cdot\underline{n} = {} & \Delta(f^V(w,\nabla w))^{n,m}n_x+\Delta(g^V(w,\nabla w))^{n,m}n_y\\ & +[(A_0^V)^{n,m}\Delta w^{n,m}+(A_1^V)^{n,m}\Delta w_x^{n,m}+(A_2^V)^{n,m}\Delta w_y^{n,m}]n_x\\ & +[(B_0^V)^{n,m}\Delta w^{n,m}+(B_1^V)^{n,m}\Delta w_x^{n,m}+(B_2^V)^{n,m}\Delta w_y^{n,m}]n_y\end{aligned}$$

where the viscous Jacobian matrices have been introduced: $A_0^V=\partial f^V/\partial w$, $A_1^V=\partial f^V/\partial w_x$, $A_2^V=\partial f^V/\partial w_y$ and similarly $B_0^V=\partial g^V/\partial w$, $B_1^V=\partial g^V/\partial w_x$, $B_2^V=\partial g^V/\partial w_y$. The partial derivatives w_x, w_y with respect to the x and y space directions are related to the partial derivatives $w_\perp$, $w_\parallel$ with respect to the (local) normal and tangential directions relative to a face by the equalities $w_x=w_\perp n_x-w_\parallel n_y$ and $w_y=w_\perp n_y+w_\parallel n_x$. Inserting these relationships into the above flux linearization yields:

$$\Delta(F^V)^{n,m}\cdot\underline{n}=(J_0^V)^{n,m}\cdot\Delta w^{n,m}+(J_\perp^V)^{n,m}\cdot\Delta w_\perp^{n,m}+(J_\parallel^V)^{n,m}\cdot\Delta w_\parallel^{n,m}$$

where $J_0^V=[A_0^V n_x+B_0^V n_y]$, $J_\perp^V=[A_1^V n_x^2+B_2^V n_y^2+(A_2^V+B_1^V)n_x n_y]$ and $J_\parallel=[A_2^V n_x^2-B_1^V n_y^2+(B_2^V-A_1^V)n_x n_y]$. This highly expensive linearization is drastically simplified by retaining only the contributions involving a positive-definite matrix coefficient, namely:

$$\Delta(F^V)^{n,m}\cdot\underline{n}\approx(J_\perp^{V(i)})^{n,m}\cdot\Delta w_\perp^{n,m}$$

with $J_\perp^{V(i)}=[A_1^V n_x^2+B_2^V n_y^2]$ in the $2D$ case and $J_\perp^{V(i)}=[A_1^V n_x^2+B_2^V n_y^2+C_3^V n_z^2]$ in the 3D case, where $C_3^V=\partial h^V/\partial w_z$. Furthermore, since $J_\perp^{V(i)}$ is a positive-definite matrix it can be replaced by its spectral radius $\rho_\perp^{V(i)}$ without compromising the linear stability of the implicit stage. The normal derivative of w with respect to a face Γ_{ik} is computed with a simple 2-point formula using the difference between the variable in the i-cell and its kth neighbor, divided by the sum of the distances between the involved cell centers and the interface $[d(i,(i,k))+d((i,k),o(i,k))]$:

$$\Delta(w_\perp^{n,m})_{i,k}=\frac{\Delta w_{o(i,k)}-\Delta w_i}{d(i,(i,k))+d((i,k),o(i,k))}$$

The resulting contribution of the viscous numerical flux balance to the LHS of (10) reads:

$$\sum_k(\Delta\mathscr{H}_{i,k}^{V(i)})^{n,m}S_{i,k}=-\sum_k(\tilde{\rho}_\perp^{V(i)})_{i,k}^{n,m}(\Delta w_i^{n,m}-\Delta w_{o(i,k)}^{n,m})S_{i,k} \tag{13}$$

with $(\tilde{\rho}_\perp^{V(i)})_{i,k}=\rho_\perp^{V(i)}/(d(i,(i,k))+d((i,k),o(i,k)))$.

2.2.3. Simplified full implicit stage. Inserting (12) and (13) into the LHS of (10) leads to a simplified implicit stage of the form:

$$D_i^{n,m}\Delta w_i^{n,m}+\frac{1}{2|\Omega_i|}\sum_k(\Delta F_{o(i,k)}^E)^{n,m}\cdot\underline{n}_{i,k}S_{i,k}-\sum_k C_{i,k}^{n,m}\Delta w_{o(i,k)}^{n,m}=-\mathscr{R}_i^{n,m} \tag{14}$$

Copyright © 2008 John Wiley & Sons, Ltd.

DOI: 10.1002/fld

with

$$C_{i,k}^{n,m} = \frac{1}{|\Omega_i|}\left(Q^V + \frac{1}{2}Q^E\right)_{i,k}^{n,m} S_{i,k}$$
$$D_i^{n,m} = \left(\frac{P_c^{-1}}{\Delta\tau}\right)_i^{n,m} + \frac{3}{2\Delta t} + \sum_k C_{i,k}^{n,m} \tag{15}$$

and the coefficients Q^V and Q^E are given by

$$Q^E = P_c^{-1}\rho(P_c J_\perp^E) = P_c^{-1}\tilde{\rho}_\perp^E, \quad Q^V = \tilde{\rho}_\perp^{V(i)} Id = \frac{\rho(J_\perp^{V(i)})}{d(i,(i,k)) + d((i,k),o(i,k))} Id \tag{16}$$

2.3. Solution methods for the implicit stage

2.3.1. Direct solver. The non-linear implicit stage (14) contains dual-time increments for both the vector of conserved variables $\Delta w^{n,m}$ and the inviscid physical fluxes. Introducing the inviscid Jacobians allows to express these latter in terms of $\Delta w^{n,m}$ and to turn (14) into the following linear system:

$$D_i^{n,m}\Delta w_i^{n,m} - \sum_k (C_a)_{i,k}^{n,m}\Delta w_{o(i,k)}^{n,m} = -\mathscr{R}_i^{n,m} \tag{17}$$

with

$$(C_a)_{i,k}^{n,m} = \frac{S_{i,k}}{|\Omega_i|}\left[\left(Q^V + \frac{1}{2}Q^E\right)_{i,k}^{n,m} - \frac{1}{2}(J_\perp^E)_{i,k}^{n,m}\right] = C_{i,k}^{n,m} - \frac{1}{2}\frac{S_{i,k}}{|\Omega_i|}(J_\perp^E)_{i,k}^{n,m}$$

and the coefficient $D_i^{n,m}$ of $\Delta w_i^{n,m}$ is unchanged with respect to (15)–(16). The linear system (17) can then be solved exactly, using for instance a GMRES approach, in which case the implicit scheme (17) will be referred to as a direct solver. Alternatively, an approximate solution of (17) can be iteratively obtained thanks to some kind of point-relaxation techniques; other standard approaches such as approximation factorizations or line-relaxation methods are not considered in this work since not directly adapted to the case of unstructured grid computations.

2.3.2. Relaxation methods. Point Jacobi: Applying a Point-Jacobi-relaxation technique to solve (17) yields:

$$\Delta w_i^{(0)} = 0$$
$$l = 0, p-1$$
$$\Delta w_i^{(l+1)} = (D_i^{n,m})^{-1}\left(-\mathscr{R}_i^{n,m} + \sum_k (C_a)_{i,k}^{n,m}\Delta w_{o(i,k)}^{(l)}\right)$$
$$w_i^{n+1} = w_i^n + \Delta w_i^{(p)}$$

Copyright © 2008 John Wiley & Sons, Ltd.

DOI: 10.1002/fld

As pointed out by Sharov and Nakahashi [13], a better starting point for deriving the Point-Jacobi implicit stage is (14), which leads to:

$$\Delta w_i^{(0)} = 0$$

$$l = 0, p-1$$

$$\Delta w_i^{(l+1)} = (D_i^{n,m})^{-1} \left(-\mathscr{R}_i^{n,m} - \frac{1}{2|\Omega_i|} \sum_k (\Delta F^E_{o(i,k)})^{(l)} \cdot \underline{n}_{i,k} S_{i,k} + \sum_k C_{i,k}^{n,m} \Delta w_{o(i,k)}^{(l)} \right) \tag{18}$$

$$w_i^{n+1} = w_i^n + \Delta w_i^{(p)}$$

When dealing with compressible flow problems ($P_c = Id$), coefficients $D_i^{n,m}$ and $C_i^{n,m}$ are purely scalar so that the implicit treatment (18) involves strictly no matrix–vector product: such a matrix-free implicit stage offers a very low-unit cost, which generally makes up for the loss of intrinsic efficiency induced by the spectral radius simplifications. It will be explained in Section 2.4 how such a low cost can be preserved when the low-Mach preconditioning is switched on, i.e. with P_c a full matrix.

Symmetric Gauss–Seidel: The Point-Jacobi algorithm can be improved using a Gauss–Seidel-type procedure in which the unknown increments Δw_i evaluated during a sweep are immediately used to compute the next ones; moreover, a reverse sweep can be performed leading to the symmetric Gauss–Seidel (SGS) relaxation procedure. The unstructured grid SGS methodology relies on two (forward and backward) sweeps using mesh ordering and thus requires the definition of sets of interfaces $L(i)$ and $U(i)$ so as to determine the terms to be relaxed at each step of the sweeps. Let k be an interface of a mesh cell i, the sets of interfaces $L(i)$ and $U(i)$ can be defined as follows:

$$k \in L(i) \Leftrightarrow o(i,k) < i, \quad k \in U(i) \Leftrightarrow o(i,k) > i$$

where it is recalled that the index $o(i,k)$ refers to the cell sharing the interface k with cell i. Using this definition, the SGS algorithm reads:

$$\Delta w_i^{(0)} = 0$$

$$l = 0, p-1$$

Forward sweep:

$$\Delta w_i^{(*)} = (D_i^{n,m})^{-1} \left(-\mathscr{R}_i^{n,m} - \frac{1}{2|\Omega_i|} \sum_{k \in L(i)} (\Delta F^E_{o(i,k)})^{(*)} \cdot \underline{n}_{i,k} S_{i,k} + \sum_{k \in L(i)} C_{i,k}^{n,m} \Delta w_{o(i,k)}^{(*)} \right.$$

$$\left. - \frac{1}{2|\Omega_i|} \sum_{k \in U(i)} \left(\Delta F^E_{o(i,k)} \right)^{(l)} \cdot \underline{n}_{i,k} S_{i,k} + \sum_{k \in U(i)} C_{i,k}^{n,m} \Delta w_{o(i,k)}^{(l)} \right)$$

Copyright © 2008 John Wiley & Sons, Ltd.

DOI: 10.1002/fld

Backward sweep:

$$\Delta w_i^{(l+1)} = (D_i^{n,m})^{-1}\left(-\mathscr{R}_i^{n,m} - \frac{1}{2|\Omega_i|}\sum_{k\in L(i)}(\Delta F^E_{o(i,k)})^{(*)}\cdot\underline{n}_{i,k}S_{i,k} + \sum_{k\in L(i)} C_{i,k}^{n,m}\Delta w_{o(i,k)}^{(*)}\right.$$

$$\left. -\frac{1}{2|\Omega_i|}\sum_{k\in U(i)}(\Delta F^E_{o(i,k)})^{(l+1)}\cdot\underline{n}_{i,k}S_{i,k} + \sum_{k\in U(i)} C_{i,k}^{n,m}\Delta w_{o(i,k)}^{(l+1)}\right)$$

$$w_i^{n+1} = w_i^n + \Delta w_i^{(p)} \tag{19}$$

where the exponent $*$ denotes provisional values stored after the forward step. It is clear that the above algorithm does not add complexity to the method since, in the manner of the PJ procedure, it only requires the inversion of the diagonal coefficient.

2.4. Low-cost treatment for all-speed flows

Let us first turn our attention to the Point-Jacobi procedure (18) and explain how each sub-iteration can be evaluated for a reduced cost even when the low-Mach number preconditioning matrix P_c appears in the formulation of the implicit stage—the technique detailed hereafter can be similarly applied to the SGS procedure and only the final formulation will be provided in that case. Each sub-iteration of the PJ procedure requires to compute:

$$\Delta w_i^{(l+1)} = (D_i^{n,m})^{-1}\left(-\mathscr{R}_i^{n,m} - \frac{1}{2|\Omega_i|}\sum_k(\Delta F^E_{o(i,k)})^{(l)}\cdot\underline{n}_{i,k}S_{i,k} + \sum_k C_{i,k}^{n,m}\Delta w_{o(i,k)}^{(l)}\right)$$

with blocks $D_i^{n,m}$ and $C_{i,k}^{n,m}$ defined by (15)–(16). Under the assumption of a systematic evaluation of the preconditioning matrix P_c^{-1} at the cell center i, the above relationship can also be recast under the form:

$$\Delta w_i^{(l+1)} = (D_i^{n,m})^{-1}\mathrm{RHS}_c + (D_i^{n,m})^{-1}(P_c^{-1})_i^{n,m}\mathrm{RHS}_d \tag{20}$$

where the so-called consistent and dissipative parts of the RHS read:

$$\begin{aligned}\mathrm{RHS}_c &= -\mathscr{R}_i^{n,m} - \frac{1}{2|\Omega_i|}\sum_k(\Delta F^E_{o(i,k)})^{(l)}\cdot\underline{n}_{i,k}S_{i,k} + \frac{1}{|\Omega_i|}\sum_k\tilde{\rho}_\perp^{V(i)}S_{i,k}\Delta w_{o(i,k)}^{(l)}\\ \mathrm{RHS}_d &= \frac{1}{2|\Omega_i|}\sum_k\tilde{\rho}_\perp^{E}S_{i,k}\Delta w_{o(i,k)}^{(l)}\end{aligned} \tag{21}$$

while coefficient $D_i^{n,m}$ is now expressed as

$$D_i^{n,m} = a_i^{n,m}(P_c^{-1})_i^{n,m} + b_i^{n,m}I_d \tag{22}$$

Copyright © 2008 John Wiley & Sons, Ltd.

DOI: 10.1002/fld

with the scalar coefficients $a_i^{n,m}$, $b_i^{n,m}$ defined by

$$a_i^{n,m}=\frac{1}{\Delta\tau_i^{n,m}}+\sum_k \frac{S_{i,k}}{2|\Omega_i|}(\tilde{\rho}_\perp^E)_i^{n,m},\quad b_i^{n,m}=\frac{3}{2\Delta t}+\sum_k \frac{S_{i,k}}{|\Omega_i|}(\tilde{\rho}_\perp^{V(i)})_i^{n,m}$$

In the case of the preconditioned steady Euler equations, $D_i^{n,m}$ reduces to $a_i^{n,m}(P_c^{-1})_i^{n,m}$ so that (20) simplifies into:

$$\Delta w_i^{(l+1)}=\frac{1}{a_i^{n,m}}[(P_c)_i^{n,m}\mathrm{RHS}_c+\mathrm{RHS}_d]$$

The remaining extra-cost with respect to the compressible case lies in the computation of the matrix–vector product $(P_c)_i^{n,m}\mathrm{RHS}_c$, which can be simplified by taking advantage of some specific properties of the Turkel low-Mach preconditioning matrix [18] considered in the present work, which have already been pointed out in [20, 21]. The Turkel low-Mach preconditioning has been derived from the Euler equations expressed in entropic variables $V^{\mathrm{T}}=(p,u,v,S)$ (pressure, velocity components and entropy) and, when applied to this specific form of the Euler equations, takes the form $P_e=I_d+(\beta^2-1)Q_e$ where $Q_e=\mathrm{Diag}(1,0,0,0)$; the preconditioning parameter β is a function of the local Mach number that will be detailed later on. It must be noted that matrix Q_e is idempotent, i.e. such that $Q_e^2=Q_e$. The preconditioning matrix P_c appearing in (2) is related to P_e through the relationship:

$$P_c=\left(\frac{\partial w}{\partial V}\right)\cdot P_e\cdot\left(\frac{\partial V}{\partial w}\right)$$

so that

$$P_c=I_d+(\beta^2-1)Q_c \tag{23}$$

with $Q_c=(\partial w/\partial V)\cdot Q_e\cdot(\partial V/\partial w)$ an idempotent matrix. An explicit expression for Q_c in the case where (2) is complemented with a perfect gas equation of state is provided in [19, 29]:

$$Q_c=\frac{\gamma-1}{c^2}\begin{pmatrix} q^2=\frac{1}{2}(u^2+v^2) & -u & -v & 1\\ uq^2 & -u^2 & -uv & u\\ vq^2 & -uv & -v^2 & v\\ Hq^2 & -uH & -vH & H\end{pmatrix}=\frac{\gamma-1}{c^2}\begin{pmatrix}1\\u\\v\\H\end{pmatrix}(q^2,-u,-v,1) \tag{24}$$

where c is the speed of sound, H the total enthalpy and γ the ratio of specific heats.

As pointed out by Turkel, the matrix–vector product involving P_c or Q_c can be performed by using definition (24), namely,

$$Q_c\cdot X=\frac{\gamma-1}{c^2}(q^2X^{(1)}-uX^{(2)}-vX^{(3)}+X^{(4)})\cdot(1,u,v,H)^{\mathrm{T}}$$

with $X^{(i)}$ the ith component of vector X. Therefore a sub-iteration of the Point-Jacobi process in the case of the steady Euler equations with low-Mach preconditioning can be computed at a

Copyright © 2008 John Wiley & Sons, Ltd.

DOI: 10.1002/fld

reduced cost using the following expression:

$$\Delta w_i^{(l+1)} = \frac{1}{a_i^{n,m}}[\mathrm{RHS}_c + \mathrm{RHS}_d] + \left[\frac{\gamma-1}{ac^2}\right]_i^{n,m} ((q^2)_i^{n,m}\mathrm{RHS}_c^{(1)} - u_i^{n,m}\mathrm{RHS}_c^{(2)} - v_i^{n,m}\mathrm{RHS}_c^{(3)} + \mathrm{RHS}_c^{(4)}) \cdot \begin{pmatrix} 1 \\ u \\ v \\ H \end{pmatrix}_i^{n,m}$$

As soon as $b_i^{n,m}$ is different from zero in the definition of $D_i^{n,m}$ (unsteady problems or/and viscous flows), the evaluation of the matrix coefficients $(D_i^{n,m})^{-1}$ and $(D_i^{n,m})^{-1}(P_c^{-1})_i^{n,m}$ in expression (20) seems to become more involved. In fact, taking advantage of the property $Q_c^2 = Q_c$ it is possible to compute in an explicit way D^{-1} as well as $D^{-1}P_c^{-1}$. Indeed, if $D_i^{n,m}$ is computed using the approximate formula (22), it is easy to show its inverse is given by

$$(D_i^{n,m})^{-1} = \left[\frac{1}{a+b}\left(I_d + \frac{a(\beta^2-1)}{a+b\beta^2}Q_c\right)\right]_i^{n,m} \tag{25}$$

Now, since $P_c^{-1} = Id + (1/\beta^2 - 1)Q_c$, an immediate calculation yields:

$$(D_i^{n,m})^{-1}(P_c^{-1})_i^{n,m} = \left[\frac{1}{a+b}\left(I_d - \frac{b(\beta^2-1)}{a+b\beta^2}Q_c\right)\right]_i^{n,m} \tag{26}$$

Inserting (25) and (26) into (20) leads to the following simplified formulation of a Point-Jacobi sub-iteration in the case of all-speed flows:

$$\Delta w_i^{(l+1)} = \frac{1}{(a+b)_i^{n,m}}[\mathrm{RHS}_c + \mathrm{RHS}_d] + \left[\frac{(\beta^2-1)Q_c}{(a+b)(a+b\beta^2)}\right]_i^{n,m} (a_i^{n,m}\mathrm{RHS}_c - b_i^{n,m}\mathrm{RHS}_d)$$

Here again, the remaining matrix–vector product involves the Turkel preconditioning matrix and can be easily computed using (24) to yield the following simple, hence low-cost, formulation of the preconditioned Point-Jacobi sub-iteration:

$$\Delta w_i^{(l+1)} = \frac{1}{(a+b)_i^{n,m}}[\mathrm{RHS}_c + \mathrm{RHS}_d] + \Delta\phi_i^{(l)} \begin{pmatrix} 1 \\ u \\ v \\ H \end{pmatrix}_i^{n,m} \tag{27}$$

Copyright © 2008 John Wiley & Sons, Ltd.

DOI: 10.1002/fld

with

$$\Delta\phi_i^{(l)} = \left[\frac{(\beta^2-1)}{(a+b)(a+b\beta^2)}\frac{\gamma-1}{c^2}\right]_i^{n,m} \times[(q^2)_i^{n,m}\widetilde{\mathrm{RHS}}^{(1)} - u_i^{n,m}\widetilde{\mathrm{RHS}}^{(2)} - v_i^{n,m}\widetilde{\mathrm{RHS}}^{(3)} + \widetilde{\mathrm{RHS}}^{(4)}] \tag{28}$$

$$\widetilde{\mathrm{RHS}} = a_i^{n,m}\mathrm{RHS}_c - b_i^{n,m}\mathrm{RHS}_d$$

Similar simplifications can be performed when using the SGS point-relaxation technique so as to yield for the forward sweep:

$$\Delta w_i^{(*)} = \frac{1}{(a+b)_i^{n,m}}[\mathrm{RHS}_c^F + \mathrm{RHS}_d^F] + \Delta\phi_i^{(*)}\begin{pmatrix}1\\ u\\ v\\ H\end{pmatrix}_i^{n,m} \tag{29}$$

with

$$\mathrm{RHS}_c^F = -\mathscr{R}_i^{n,m} - \frac{1}{2|\Omega_i|}\sum_{k\in L(i)}(\Delta F_{o(i,k)}^E)^{(*)}\cdot\underline{n}_{i,k}S_{i,k} + \frac{1}{|\Omega_i|}\sum_{k\in L(i)}\tilde{\rho}_\perp^{V(i)}S_{i,k}\Delta w_{o(i,k)}^{(*)}$$

$$-\frac{1}{2|\Omega_i|}\sum_{k\in U(i)}(\Delta F_{o(i,k)}^E)^{(l)}\cdot\underline{n}_{i,k}S_{i,k} + \frac{1}{|\Omega_i|}\sum_{k\in U(i)}\tilde{\rho}_\perp^{V(i)}S_{i,k}\Delta w_{o(i,k)}^{(l)}$$

$$\mathrm{RHS}_d^F = \frac{1}{2|\Omega_i|}\sum_{k\in L(i)}\tilde{\rho}_\perp^E S_{i,k}\Delta w_{o(i,k)}^{(*)} + \frac{1}{2|\Omega_i|}\sum_{k\in U(i)}\tilde{\rho}_\perp^E S_{i,k}\Delta w_{o(i,k)}^{(l)}$$

and

$$\Delta\phi_i^{(*)} = \left[\frac{(\beta^2-1)}{(a+b)(a+b\beta^2)}\frac{\gamma-1}{c^2}\right]_i^{n,m}[(q^2)_i^{n,m}\widetilde{\mathrm{RHS}}_F^{(1)} - u_i^{n,m}\widetilde{\mathrm{RHS}}_F^{(2)} - v_i^{n,m}\widetilde{\mathrm{RHS}}_F^{(3)} + \widetilde{\mathrm{RHS}}_F^{(4)}]$$

$$\widetilde{\mathrm{RHS}}_F = a_i^{n,m}\mathrm{RHS}_c^F - b_i^{n,m}\mathrm{RHS}_d^F$$

A similar inexpensive expression can be derived for the backward sweep. It must be emphasized at this stage that the simplified expressions (27) and (29) are strictly equivalent to the original formulations (18) and (19) when applied to linear or linearized problems. The simplified formulations differ from the original ones for non-linear problems because they rely on a single evaluation of P_c at the cell center i instead of several evaluations both at the cell center and at the faces of the control volume for the original formulations. The global efficiency of these preconditioned matrix-free formulations (27) and (29) will be practically assessed in Section 4 by computing low-Mach number flows with these methods. Before proceeding to this application of the proposed low-cost implicit treatments, a von Neumann analysis is performed in the next section to obtain an *a priori* estimate of their efficiency.

Copyright © 2008 John Wiley & Sons, Ltd.

DOI: 10.1002/fld

3. IMPLICIT TREATMENT ANALYSIS

The stability and intrinsic efficiency properties of the simplified implicit scheme (17), approximately solved using a PJ or SGS procedure, are now analyzed through the study of its associated amplification factor. In order to perform such a von Neumann analysis, schemes (17)–(19) are first expressed in the case of the 2D linearized preconditioned Euler equations solved on a uniform Cartesian grid. General formulae are then derived for the amplification factor and comparisons are performed between the low-unit cost implicit solver and an expensive but intrinsically efficient block implicit scheme.

3.1. Cartesian grid implicit formulations

Let us consider the 2D linearized preconditioned Euler equations:

$$P_c^{-1} w_t + A^E(w_0) w_x + B^E(w_0) w_y = 0 \tag{30}$$

where the inviscid Jacobian matrices are computed for a given physical state w_0. System (30) is discretized on a uniform Cartesian grid ($x_i = i\delta x$, $y_j = j\delta y$) with constant steps δx and δy. For the sake of clarity, the analysis is performed in the specific case of a Roe-type explicit stage, extended to third-order using variable reconstruction. It was checked that using a AUSM+, Rusanov or HLL explicit stage leads to similar conclusions. The 2D finite difference equivalent of (14) reads:

$$\mathscr{H} \cdot \Delta w_{i,j}^n = -\mathscr{K} \cdot w_{i,j}^n \tag{31}$$

with $\mathscr{H}$ and $\mathscr{K}$, respectively, the implicit and explicit difference operator. In the specific case of a third-order Roe-type explicit stage, the operator $\mathscr{K}$ takes the form:

$$\mathscr{K} = \dot{A}^E (I - \tfrac{1}{6})\delta_1 \mu_1 + \dot{B}^E (I - \tfrac{1}{6})\delta_2 \mu_2 + \tfrac{1}{12} P_c^{-1} |P_c \dot{A}^E| \delta_1^4 + \tfrac{1}{12} P_c^{-1} |P_c \dot{B}^E| \delta_2^4 \tag{32}$$

where difference and average operators acting in each grid direction have been introduced:

$$(\delta_1 \phi)_{i+1/2,j} = \phi_{i+1,j} - \phi_{i,j}, \quad (\delta_2 \phi)_{i,j+1/2} = \phi_{i,j+1} - \phi_{i,j}$$

$$(\mu_1 \phi)_{i+1/2,j} = \tfrac{1}{2}(\phi_{i+1,j} + \phi_{i,j}), \quad (\mu_2 \phi)_{i,j+1/2} = \tfrac{1}{2}(\phi_{i,j+1} + \phi_{i,j})$$

and with $\dot{A}^E = \Delta t / \delta x \cdot A^E$, $\dot{B}^E = \Delta t / \delta y \cdot B^E$. The implicit operator $\mathscr{H}$ takes the general form:

$$\mathscr{H} = [P_c^{-1} + \dot{A}^E \mu_1 \delta_1 + \dot{B}^E \mu_2 \delta_2 - \tfrac{1}{2} \dot{Q}_1^E \delta_1^2 - \tfrac{1}{2} \dot{Q}_2^E \delta_2^2]^n \tag{33}$$

where $\dot{Q}_1^E = \Delta t / \delta x \cdot Q_1^E$, $\dot{Q}_2^E = \Delta t / \delta y \cdot Q_2^E$ and the dissipation matrices Q_1^E, Q_2^E can be evaluated using expression (9) or (16) leading either to a classical implicit block scheme (see, for instance, [7, 27, 28]) or to a matrix-free scheme. The implicit operator can also be expressed as:

$$\mathscr{H} = D - \dot{A}^+ \mathscr{E}_1^- - \dot{B}^+ \mathscr{E}_2^- + \dot{A}^- \mathscr{E}_1^+ + \dot{B}^- \mathscr{E}_2^+ \tag{34}$$

with

$$D = P_c^{-1} + \dot{Q}_1^E + \dot{Q}_2^E, \quad \dot{A}^\pm = \tfrac{1}{2}(\dot{A}^E \pm \dot{Q}_1^E), \quad \dot{B}^\pm = \tfrac{1}{2}(\dot{B}^E \pm \dot{Q}_2^E) \tag{35}$$

Copyright © 2008 John Wiley & Sons, Ltd.

DOI: 10.1002/fld

and the shift operators defined by $\mathscr{E}_1^{\pm}\cdot\phi_{i,j}=\phi_{i\pm1,j}$, $\mathscr{E}_2^{\pm}\cdot\phi_{i,j}=\phi_{i,j\pm1}$. The linear system associated with (31) can be solved directly or using an iterative-relaxation procedure (Point Jacobi or SGS described in the previous section). A general iterative scheme reads:

$$\begin{aligned}\Delta w_{i,j}^{(0)}&=0\\ l&=0,p-1\\ \alpha&=1,N\\ \mathscr{H}_\alpha\cdot\Delta w_{i,j}^{(lN+\alpha)}&=-\mathscr{K}\cdot w_{i,j}^{n}-(\mathscr{H}-\mathscr{H}_\alpha)\cdot\Delta w_{i,j}^{(lN-\alpha-1)}\\ \Delta w_{i,j}^{n}&=\Delta w_{i,j}^{(pN)}\end{aligned}\tag{36}$$

where $\mathscr{H}_\alpha$ is a difference operator that depends on the specific choice of the iterative method. The Point-Jacobi-relaxation procedure described by (18) in the general case corresponds in this linear analysis on a Cartesian grid to a single difference operator:

$$\mathscr{H}_1^{\mathrm{PJ}}=D$$

while the SGS method described by (19) in the general case is simply expressed using the following operators ($N=2$):

$$\mathscr{H}_1^{\mathrm{SGS}}=D-\dot{A}^{+}\mathscr{E}_1^{-}-\dot{B}^{+}\mathscr{E}_2^{-},\quad \mathscr{H}_2^{\mathrm{SGS}}=D+\dot{A}^{-}\mathscr{E}_1^{+}+\dot{B}^{-}\mathscr{E}_2^{+}$$

3.2. Von Neumann analysis

In the framework of a von Neumann analysis, the stability and efficiency properties of the direct solver (31) or iterative solver (36) are analyzed by computing the spectral radius of the amplification matrix associated with each implicit scheme. This amplification matrix is obtained from a Fourier transform of (31) or (36). For the direct solver, a Fourier transform of (31) yields:

$$H\cdot\Delta\widehat{w}_{i,j}^{n}=-K\cdot\widehat{w}_{i,j}^{n}\tag{37}$$

where $\widehat{w}$ is the transformed function of w^n and H, K denote, respectively, the transformed functions of $\mathscr{H}$, $\mathscr{K}$ which are easily computed from (32) and (33). Introducing the reduced wave numbers ξ and η associated, respectively, with the x and y directions and denoting $s_1=\sin(\xi)$, $s_2=\sin(\eta)$, $z_1=\frac{1}{2}(1-\cos(\xi))$, $z_2=\frac{1}{2}(1-\cos(\eta))$, symbols H and K can be cast in the compact form:

$$K=\tfrac{4}{3}P_c^{-1}|P_c\dot{A}^{E}|z_1^2+\tfrac{4}{3}P_c^{-1}|P_c\dot{B}^{E}|z_1^2+\underline{\mathrm{i}}[(I+\tfrac{2}{3}z_1)s_1\dot{A}^{E}+(I+\tfrac{2}{3}z_2)s_2\dot{B}^{E}]\tag{38}$$

$$H=P_c^{-1}+\dot{Q}_1^{E}z_1+\dot{Q}_2^{E}z_2+\underline{\mathrm{i}}(\dot{A}^{E}s_1+\dot{B}^{E}s_2)\tag{39}$$

Since the amplification matrix is such that $\widehat{w}^{n+1}(\xi,\eta)=G(\xi,\eta)\cdot\widehat{w}^{n}(\xi,\eta)$, the amplification matrix G_* associated with the direct solver (31) reads:

$$G_*=I_d-H^{-1}\cdot K\tag{40}$$

with H, K given by (38), (39).

Copyright © 2008 John Wiley & Sons, Ltd.

DOI: 10.1002/fld

For the iterative solver, a Fourier transform of (36) yields the following form for the amplification matrix G_p:

$$G_p = G_* + V^p(I_d - G_*)$$
$$V = \prod_{\alpha=1}^{N} (I_d - H_\alpha^{-1} \cdot H) \tag{41}$$

with H_α the Fourier transform of the operator $\mathscr{H}_\alpha$. For the PJ method, $N=1$ and $H_1 = D$ while $N=2$ and $H_1 = D - \dot{A}^+ e^{-\mathrm{i}\xi} - \dot{B}^+ e^{-\mathrm{i}\eta}$, $H_2 = D + \dot{A}^- e^{\mathrm{i}\xi} + \dot{B}^- e^{\mathrm{i}\eta}$ for the SGS approach.

The amplification matrices G_* and G_p are functions of the reduced wave numbers ξ, η as well as of a limited number of numerical and physical parameters. The key numerical parameters are the Cournant–Friedrichs–Lewy (CFL) number and the preconditioning parameter β, whereas the Mach number M, speed of sound c and velocity components ratio $\delta = v/u$ form the key physical parameters of the study. The CFL number is the ratio of the numerical time step Δt to the characteristic time step associated with the physical problem and can be interpreted as the acceleration factor allowed by the implicit time integration; for the linearized Euler equations, this characteristic time step is defined by $\Delta t^E = \min(\Delta x, \Delta y)/(c(M+1))$. The preconditioning parameter is prescribed as $\beta = 1$ when no preconditioning is applied and $\beta = M$ when looking for a steady solution of the preconditioned linearized Euler equations; other definitions, adapted to viscous or/and unsteady flows will be given and used in the section devoted to the applications.

Scheme (31) will be stable if and only if the modulus of $\rho(G_*)$, the spectral radius of the amplification matrix G_*, known also as the amplification factor, is lower than unity for all wave numbers; similarly, (36) is stable iff $|\rho(G_p)| \leqslant 1 \ \forall(\xi, \eta)$. Since it was observed that the amplification factor depends mainly on the wave numbers (ξ, η), Mach number M (through the definition of β) and CFL, two types of analysis will be considered hereafter:

(i) local analysis: for a given set of parameters (CFL, M), $\rho(G)$ is computed with a sweep on the wave numbers space; if there exists at least a couple (ξ, η) such that $\rho(G) > 1$ then the scheme is unstable for the considered parameters. Besides, achieving the smallest possible value of $\rho(G)$ for all couple (ξ, η) ensures a fast damping of the error hence a convergence to steady state for a limited number of time iterations, a desirable property also referred to as high intrinsic efficiency.

(ii) global analysis: the maximum value of the amplification factor is defined as $\rho^{\max}(M, \mathrm{CFL}) = \max_{(\xi,\eta)\in[0,\pi]^2}[\rho(G(\xi, \eta, M, \mathrm{CFL}))]$. It is computed for each combination of Mach and CFL numbers, which enables to draw the stability map of the scheme in the (CFL, M) plane and to detect for which conditions $\rho(G) > 1$. The mean value of the amplification factor is defined as $\rho^{\mathrm{ave}}(M, \mathrm{CFL}) = (1/\pi^2) \int_0^\pi \int_0^\pi \rho(\xi, \eta; M, \mathrm{CFL})\, \mathrm{d}\xi\, \mathrm{d}\eta$. When the scheme is stable, this mean value provides a quantitative information on the error damping provided by the scheme: the lower the value of $\rho^{\mathrm{ave}}(M, \mathrm{CFL})$, the better the intrinsic efficiency of the numerical treatment.

The mean value $\rho^{\mathrm{ave}}(M, \mathrm{CFL})$ for the direct solver (31) with $\beta = 1$ is plotted in Figure 1: the choice (9) for $\dot{Q}_1^E$, $\dot{Q}_2^E$ corresponds to the block scheme while choosing (16) yields the matrix-free scheme. Note the maximum value $\rho^{\max}(M, \mathrm{CFL})$ is not plotted here since always equal to unity, which means that the block and matrix-free schemes are unconditionally stable when directly

Copyright © 2008 John Wiley & Sons, Ltd.

DOI: 10.1002/fld

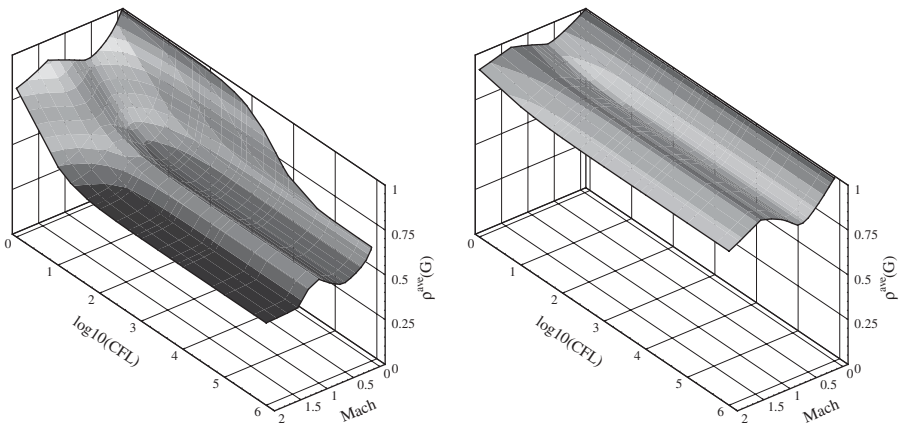

Figure 1. Mean value $\rho^{\text{ave}}(M, \text{CFL})$ for the unpreconditioned schemes solved directly. Left: block scheme. Right: matrix-free scheme.

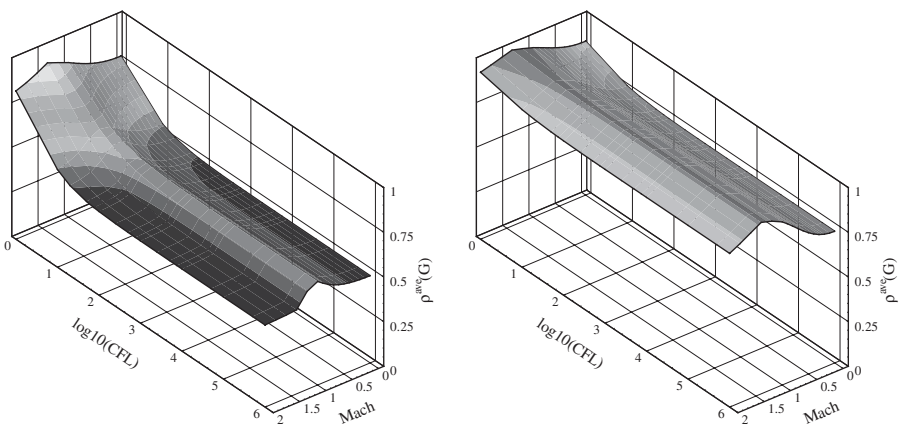

Figure 2. Mean value $\rho^{\text{ave}}(M, \text{CFL})$ for the preconditioned schemes solved directly. Left: block scheme. Right: matrix-free scheme.

solved. Clearly, both schemes yield poor efficiency properties when M goes to zero, which was expected since no preconditioning is applied. Moreover, the scalar implicit dissipation term retained for the matrix-free scheme severely impacts the efficiency of this scheme for all the flow regimes: an asymptotic value of the average error damping is rapidly reached when increasing the CFL number and this value remains well above that provided by the block scheme which offers a much better intrinsic efficiency—at the expense however of a larger unit cost. The lack of efficiency observed in the low-Mach region can be circumvented using the low-Mach preconditioning as shown in Figure 2; note the preconditioned block and matrix-free schemes remain unconditionally

Copyright © 2008 John Wiley & Sons, Ltd.

DOI: 10.1002/fld

stable when directly solved. The low-Mach preconditioning ($\beta = M$) drastically lowers the mean value $\rho^{\text{ave}}(M, \text{CFL})$ for both schemes in the low-Mach region with respect to the previous non-preconditioned case ($\beta = 1$). In the case of the matrix-free scheme, the positive impact of this preconditioning on intrinsic efficiency is especially strong in the quasi-incompressible regime because choosing $\beta = M$ makes the condition number of system (30) close to unity: the difference between all the eigenvalues is then smaller than for compressible flows and, consequently, the spectral radius simplification becomes less penalizing.

Since the intrinsic efficiency offered by the directly solved matrix-free scheme remains significantly lower than the one provided by the directly solved block scheme, global efficiency for the matrix-free scheme can only be achieved by reducing as much as possible the unit cost of the approach. In practice, the matrix-free scheme is solved using a PJ or SGS iterative technique and the treatment previously described in Section 2.4 aims precisely at achieving such a low cost. The present Von Neumann analysis is used to determine *a priori* the optimal number of sweeps required by the PJ and SGS matrix-free schemes summarized by formulae (27) and (29). Since the amplification factor of these implicit schemes tends to the ideal amplification factor associated with the direct solver (31) when the number of inner iterations p goes to infinity, the idea is to find the minimal value of p that ensures both the stability of the iterative solvers and an efficiency close enough to the ideal one. In Figure 3, the mean and maximum value of the amplification factor for both techniques are compared with the direct solver at $M = 10^{-3}$ and $\text{CFL} = 10^6$. As far as efficiency is concerned, 20 inner iterations are sufficient for the SGS scheme to recover the performances of the direct solver, whereas the PJ method requires at least 50 inner iterations to approach the same efficiency. Moreover, the SGS scheme is unconditionally stable, whereas the PJ approach suffers from instability for $p < 100$. A local analysis, not presented here for the sake of conciseness, reveals that the PJ instability arises for very low-frequency modes that can be captured on very fine grids only. In practice, the amount of sub-iterations used for the PJ scheme depends on the mesh size: it will vary between 30 and 120 in the numerical experiments of the next section. For these same applications, the SGS matrix-free scheme will be systematically used with at most 50 sub-iterations as suggested by the present Fourier analysis.

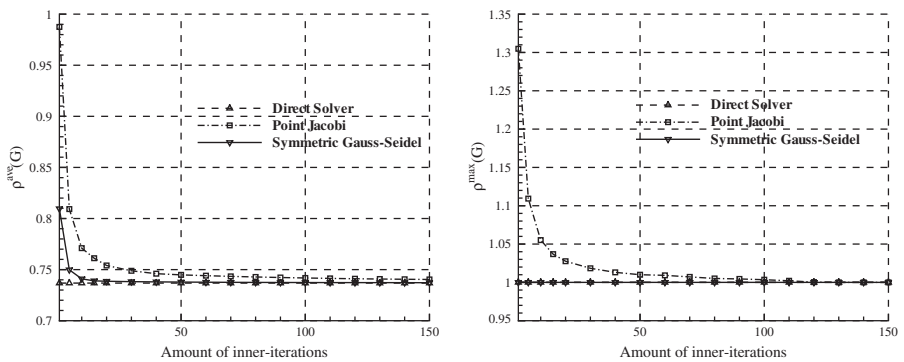

Figure 3. Impact of the amount of inner-iterations on the efficiency and the stability of PJ and SGS algorithms coupled with the matrix-free implicit stage at $M = 10^{-3}$ and $\text{CFL} = 10^6$. Left: mean value ρ^{ave}. Right: maximum value $\rho^{\max}$.

Copyright © 2008 John Wiley & Sons, Ltd.

DOI: 10.1002/fld

4. APPLICATIONS

The Fourier analysis presented in the previous section demonstrates that a steady solution can be obtained in a stable way for all-speed flows using a matrix-free preconditioned scheme solved with a point-relaxation technique. The proposed matrix-free approach will be globally more efficient than a conventional block technique provided its cost per iteration is low enough to make up for its lack of intrinsic efficiency, made obvious in Figure 2. Another measure of efficiency, not emphasized until now, is the memory requirement associated with the numerical scheme. The present section aims at assessing the cost per iteration and the related global cost as well as the storage cost for the block and matrix-free preconditioned schemes by applying these schemes to a series of steady and unsteady low-Mach number flows.

4.1. Inviscid steady flow in a sine-bump channel

4.1.1. Description of the 2D case. The inviscid flow of a perfect gas ($\gamma=1.4$) in a channel is computed for flow regimes ranging from supersonic down to nearly incompressible. The channel is enclosed in a rectangular domain $(x, y)\in[0; 4]\times[0; 1]$, with a straight upper wall and a curved lower wall, shaped as a sine bump described by

$$\begin{aligned} &x<1, \quad y=0\\ 1\leqslant x\leqslant 3, \quad &y=0.1\ (1-\cos[(x-1)\pi])\\ &x>3, \quad y=0 \end{aligned}$$

The flow is computed using a series of three unstructured grids, containing, respectively 3468, 7898 and 14 104 triangular elements for the coarse, medium and fine one. The inlet Mach number successively takes the values $M_\infty=2$, $M_\infty=0.6$, $M_\infty=0.5$, $M_\infty=0.1$ and $M_\infty=10^{-4}$. For $M_\infty=2$, supersonic boundary conditions are considered: density, velocity and pressure are imposed at the inlet while, at the outlet, the same quantities are extrapolated from the interior domain. For all the other inlet Mach numbers, subsonic conditions are considered: total enthalpy, entropy and flow angle are imposed at the inlet while pressure is extrapolated from the interior domain; static pressure is imposed at the outlet while density and velocity components are extrapolated from the interior domain. No-slip conditions are imposed on the upper and lower wall of the domain. The preconditioned Euler equations, with $\beta=\min(1, M_\infty)$ (see also the Appendix for a summary of the choice of the preconditioning parameter), are solved using:

(i) the Point-Jacobi matrix-free scheme (27) with α inner iterations, MF-PJ(α);
(ii) the SGS matrix-free scheme (29) with α inner iterations, MF-SGS(α);
(iii) the block scheme defined by (17) with a solution of the block implicit stage based on a Newton–Krylov algorithm (GMRES with ILUT preconditioner, Krylov space dimension equal to 50). The Newton–Krylov iterative algorithm denoted NK(α) is run till the residual is lower than $10^{-\alpha}$.

While a Roe-type explicit stage was considered up to now in the description and analysis of the numerical treatments under study (for the sake of simplicity in their description), the AUSM+(P) scheme [25] with second-order limited reconstruction on the primitive variables will be used throughout this section. Consequently, the dissipation matrices Q_1^E, Q_2^E appearing in the block implicit stage solved by NK(α) are no longer given by (9) but have been recomputed from the

Copyright © 2008 John Wiley & Sons, Ltd.

DOI: 10.1002/fld

expression of the AUSM+(P) numerical flux formula. Meanwhile, the implicit stages associated with MF-PJ(α) or MF-SGS(α) remain unaffected by this change of explicit stage: only $\mathscr{R}_i^{n,m}$ is modified in (20), leaving (27) and (29) formally unchanged. Block and matrix-free implicit treatments will be systematically compared, with varying flow regime and grid refinement, from the viewpoint of both global efficiency and memory requirements, that is minimal CPU time and storage should be consumed to reach steady state. The steady solution will not be analyzed since it depends only on the standard AUSM+(P) explicit stage and is therefore the same whatever the implicit stage solution.

4.1.2. Numerical results for the 2*D case.* The elements of comparison between block and matrix-free schemes are summarized in Tables I–V, organized as follows: each table corresponds to a given Mach number; for each Mach number, the coarse-, medium- and fine-grid computations are referenced, respectively, as mesh 1, 2 and 3. *Iterations* denote the number of iterations needed to reach steady state with machine accuracy, which describes intrinsic efficiency. CPU denotes the CPU time consumed to reach this steady state, which corresponds to global efficiency. RCPI represents the relative cost per iteration for each method; it is computed for a given Mach number

Table I. 2D sine-bump channel. Supersonic case: $M=2$.

Mesh	Method	Iterations	CPU	RCPI
1	MF-PJ(30)	730	65	0.8
	MF-SGS(10)	670	76	1
	NK(3)	100	180	16
2	MF-PJ(40)	680	272	1
	MF-SGS(20)	690	275	1
	NK(3)	215	1180	14
3	MF-PJ(60)	820	590	1
	MF-SGS(20)	860	617	1
	NK(3)	185	2290	17

Table II. 2D sine-bump channel. Transonic case: $M=0.6$.

Mesh	Method	Iterations	CPU	RCPI
1	MF-PJ(30)	1210	130	0.9
	MF-SGS(10)	940	115	1
	NK(3)	75	160	17
2	MF-PJ(40)	1650	480	1.1
	MF-SGS(10)	1160	300	1
	NK(3)	190	1280	26
3	MF-PJ(60)	1100	785	1.5
	MF-SGS(10)	1110	522	1
	NK(3)	190	2815	31

Copyright © 2008 John Wiley & Sons, Ltd.

DOI: 10.1002/fld

Table III. 2D sine-bump channel. Subsonic case: $M = 0.5$.

Mesh	Method	Iterations	CPU	RCPI
1	MF-PJ(80)	410	184	1.4
	MF-SGS(15)	420	136	1
	NK(3)	65	158	7.5
2	MF-PJ(100)	480	604	0.97
	MF-SGS(40)	480	624	1
	NK(3)	90	714	6.1
3	MF-PJ(100)	650	1455	0.95
	MF-SGS(40)	610	1432	1
	NK(3)	75	1443	8.2

Table IV. 2D sine-bump channel. Low-Mach number case: $M = 0.1$.

Mesh	Method	Iterations	CPU	RCPI
1	MF-PJ(80)	290	127	1.4
	MF-SGS(20)	300	92	1
	NK(3)	45	120	8.7
2	MF-PJ(100)	380	474	0.98
	MF-SGS(40)	370	472	1
	NK(3)	90	682	5.9
3	MF-PJ(120)	430	1120	1.15
	MF-SGS(40)	430	970	1
	NK(3)	65	1707	11.6

Table V. 2D sine-bump channel. Very low-Mach number case: $M = 10^{-4}$.

Mesh	Method	Iterations	CPU	RCPI
1	MF-PJ(80)	150	69	1.4
	MF-SGS(20)	160	52	1
	NK(3)	22	146	20
2	MF-PJ(100)	180	228	0.98
	MF-SGS(40)	175	227	1
	NK(3)	22	590	20
3	MF-PJ(120)	225	585	1.15
	MF-SGS(40)	220	514	1
	NK(3)	EMR*	—	—

on a given grid as the ratio of CPU to *Iterations*, normalized by the cost per iteration of the MF-SGS method, retained as reference. All the computations were performed using an Intel Xeon 3 GHz processor with 2 GB RAM Memory.

Copyright © 2008 John Wiley & Sons, Ltd.

DOI: 10.1002/fld

This systematic comparison yields the following main conclusions:

- the Newton–Krylov approach is the fastest method in terms of iterations; in other words, it provides the best intrinsic efficiency. However, with a cost per iteration much larger than PJ-MF and SGS-MF (from 5.9 to 31 times larger than the cost per iteration for SGS-MF depending on the grid refinement and inlet Mach number), the resulting global computational cost of this approach is always higher than the global cost of the matrix-free schemes.
- the matrix-free schemes require a large amount of iterations to achieve convergence to steady state and this amount increases as the grid becomes finer, a behavior that was forecast by a local von Neumann analysis not presented here for the sake of conciseness. In order tomaintain a good damping of the low-frequency error modes, it is necessary to increase the number of inner iterations, especially for the Point-Jacobi algorithm. However, as the cost per iteration of the matrix-free schemes are especially small, PJ-MF and SGS-MF remain globally more efficient than their Newton–Krylov block counterpart.
- in the case of very low-Mach number flows, the NK approach displayed some lack of robustness which made it necessary to increase the size of the ILUT preconditioner up to a point where the NK solver could no longer be used on the available computer (hence the EMR mention in Table V which stands for *Exceeded Memory Requirements*). Meanwhile, the simplicity of the matrix-free methods allowed to perform fast simulations for all configurations including the very low-Mach number one.

The behavior of the same three methods is now investigated on the 3D extension of the sine-bump channel flow where the memory requirements will become as crucial an issue as the convergence speed.

4.1.3. Description of the 3D case. This test-case is in fact a nominally 2D configuration—the one treated in the previous paragraph—computed on three-dimensional grids in order to emphasize the efficiency and robustness of the proposed matrix-free implicit treatment. The previous 2D channel is now a xz section of a 3D channel extending from $y=0$ to 0.5. The initial and boundary conditions remain the same with respect to the previous test case and need only to be completed with symmetry boundary conditions along the plane $y=0$ and 0.5. Three grids made of six-face prisms are used for computing the inviscid flow with $M_\infty=0.1$: they contain, respectively, 34 780 elements and 91 428 faces for the coarse grid 118 470 elements and 306 323 faces for the medium grid and 282 080 elements and 723 304 faces for the fine grid.

4.1.4. Numerical results for the 3D case. The convergence characteristics of PJ-MF(α), PJ-SGS(α) and NK(α) when applied to this 3D low-Mach problem are summarized in Table VI. The most striking feature is that the Newton–Krylov scheme is quickly limited because of its very demanding memory requirements: in fact, the memory resources of regular computers such as the one used in the study amount to 2 GB of RAM memory, which is not sufficient to deal with more than 10 000 elements in 3D. On the other hand, using matrix-free schemes on the same computer, it is possible to compute flows in grids involving more than 592 350 elements. For this 3D test problem, the very simple MF-PJ scheme turns out to be globally more efficient than MF-SGS: the intrinsic efficiency of this latter approach is sensitive to the ordering of the grid cells; moreover, applying SGS requires some knowledge of cell connectivity, which increases the cost per iteration. Note the

Copyright © 2008 John Wiley & Sons, Ltd.

DOI: 10.1002/fld

Table VI. 3D sine-bump channel. Low-Mach number case: $M = 0.1$

Mesh	Method	Iterations	CPU	RCPI
1	MF-PJ(80)	310	2570	0.88
	MF-SGS(40)	340	3180	1
	NK(3)	EMR	—	—
2	MF-PJ(100)	410	15000	0.88
	MF-SGS(50)	450	18800	1
	NK(3)	EMR	—	—
3	MF-PJ(120)	580	62400	1.08
	MF-SGS(50)	630	62500	1
	NK(3)	EMR	—	—

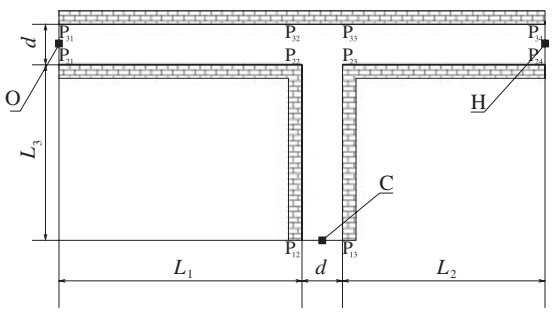

Figure 4. Tee junction. H represents the inlet of the hot fluid, C of the cold one and O is the output.

simplicity of the MF-PJ method can also be exploited to implement the approach within a parallel framework [30].

4.2. Numerical investigation of a tee junction

4.2.1. Introduction. The efficiency of the free-matrix algorithms is now assessed by computing the steady and unsteady laminar air flow in a 2D tee junction, air being considered as a calorically perfect gas. The mixing of a hot and cold fluid in a tee junction (see Figure 4 for a schematic view of the configuration) has been the object of several investigations in the nuclear energy industry (see, for instance, [31, 32]) for the following reasons.

In the region where hot and cold flow streams join, random fluctuations of the coolant temperature can occur and cause cyclical thermal stresses (known as thermal stripping) with subsequent fatigue cracking of the piping. Accidents linked to thermal stripping already occurred at some nuclear power plants (Oskarshamn/Ringhals1/Barsabeck2 in Sweden, Tsuruga in Japan, Civaux in France), which increased the interest for this phenomenon. As mentioned in [33], such accidents are very complex to analyze since they involved several fields (thermal-hydraulics, heat transmission, mechanics and material science) and the understanding of thermal stripping is beyond the purpose of this article. From the pure viewpoint of numerical fluid mechanics, such low-speed flows

Copyright © 2008 John Wiley & Sons, Ltd.

DOI: 10.1002/fld

are usually computed using asymptotic pressure-based methods specifically developed for this purpose. Therefore, the flow configuration described in Figure 4 offers a significant test case to assess the efficiency of the all-speed flow low-cost matrix-free approach developed in this work. In what follows, the numerical results obtained using the newly proposed MF-SGS(α) approach will be compared with reference results provided by an asymptotic pressure-based solver valid for low Mach number flows only and described in [34], with the objective to demonstrate that the all-purpose preconditioned matrix-free method represents an attractive alternative to an existing technique limited to a specific flow regime. Both approaches will be, respectively, referred to as *asymptotic compressible* and *fully compressible* and compared from the viewpoint of efficiency and accuracy—the solution of the asymptotic compressible or pressure-based solver being considered as reference.

4.2.2. Description of the steady problem. The flow in the tee junction admits a steady state for the configuration described in Figure 4: the gas flows through the H(ot) inlet with a temperature T_H and a momentum m_H and through the C(old) inlet with a temperature T_C and a momentum m_C. The inlet temperatures are constant in space and time, while the inlet momenta follow a parabolic distribution:

$$m_H(s)=\frac{6\bar{m}_H}{d^2}\left(\frac{d^2}{4}-s^2\right), \quad m_C(s)=\frac{6\bar{m}_C}{d^2}\left(\frac{d^2}{4}-s^2\right)$$

where s is the distance with respect to the tube axis, $\bar{m}_H$ and $\bar{m}_C$ are the average momenta such that $\bar{m}_t=\bar{m}_H+\bar{m}_C$. At the outlet 0, the pressure is constant (in space and time) and equal to P_0 (plus the hydrostatic component). The velocity is set to zero at the walls, which are impermeable and adiabatic everywhere. The dynamic viscosity μ_0 and thermal diffusivity k_0 are assumed constant for the sake of simplicity.

It can be shown that the non-dimensional solution of this problem depends on the following non-dimensional parameters: L_1/d, L_2/d and L_3/d (with a limited influence in the mixing region provided their values are large enough), the specific heat ratio γ, the Reynolds number $Re=\bar{m}_t d/\mu_0$, the Prandtl number $Pr=(\mu_0/\lambda_0)\gamma/(\gamma-1)R$ with R being the gas constant, the Mach number $M=((\bar{m}_t{}^2/\gamma P_0^2)R\bar{T})^{1/2}$ with $\bar{T}=\frac{1}{2}(T_C+T_H)$, the Froude number $Fr=(\bar{m}_t R\bar{T}/P_0)^2 1/gd$ where g is the gravity acceleration, the ratios $\alpha_H=\bar{m_H}/\bar{m}_t$ and $\varepsilon=(T_H-T_C)/(T_H+T_C)$. The steady problem under study is defined by the following set of non-dimensional parameters:

L_1/d	L_2/d	L_3/d	γ	ε	α_H	Pr	Re	M	Fr
12	9	7	1.4	0.2	0.8	0.7	100	$\frac{1}{300}$	$\frac{1}{9}$

This set of non-dimensional parameters can be completed with the following choice of numerical values:

$$d=1\,\mathrm{m}, \quad \bar{\rho}=\frac{P_0}{R\bar{T}}=1\,\mathrm{kg\,m^{-3}}, \quad \bar{m}_t=1\,\mathrm{kg\,m^{-2}\,s^{-1}}, \quad R=288\,\mathrm{J\,kg^{-1}\,K^{-1}}$$

Copyright © 2008 John Wiley & Sons, Ltd.

DOI: 10.1002/fld

which allows to compute all the dimensional quantities involved in the problem (SI units):

L_1	L_2	L_3	P_0	$\bar{T}$	T_H	T_C	$\bar{m}_H$	$\bar{m}_C$	μ_0	λ_0
12	9	7	64 290	223.2	267.8	178.6	0.8	0.2	0.01	14.4

4.2.3. Numerical results for the steady case. In order to achieve grid convergence, three grids of increasing refinement are used, containing, respectively, 12 630, 22 400 and 35 000 elements. The computations were run on a Pentium 4, 2 GHz with 2 Gbyte of RAM memory and a convergence of 10 orders on the numerical residual has been achieved in each case. The steady solution associated with the data set defined in the previous paragraph and computed using the matrix-free solver on the finest grid is presented in Figure 5. It can be deduced from these plots of streamlines and temperature contours that the hot and the cold fluid do not mix: the heat transmission is achieved through thermal diffusion only. Comparisons of the solutions provided by the asymptotic solver and the preconditioned matrix-free method are presented in Figure 6 where the temperature and velocity distributions along sections $P_{22}P_{23}$ and $P_{22}P_{32}$ of the tee junction are plotted (refer Figure 4 for the location of these sections in the tee geometry). The fully compressible solver makes use of a preconditioning parameter β designed for steady viscous flow (see Appendix for more details). The solution of the preconditioned matrix-free solver is in excellent agreement with the reference solution obtained using the asymptotic pressure-based solver. The main results regarding the efficiency of both techniques are summarized in Table VII. The proposed fully compressible approach based on a preconditioned matrix-free implicit scheme offers a very low cost per iteration, more than 15 times lower than the unit cost of the asymptotic solver on the finest grid. As a consequence, the resulting global cost of the preconditioned matrix-free method is almost half that of the reference asymptotic solver on the fine grid.

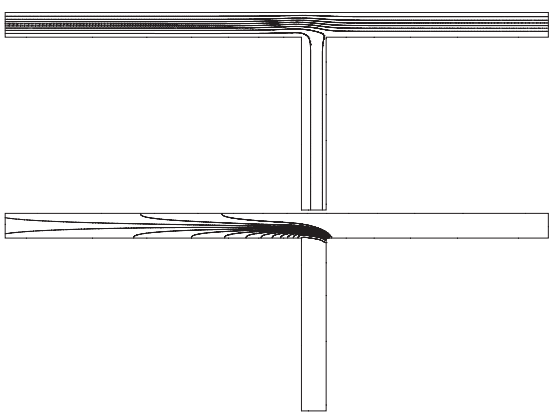

Figure 5. Tee junction. Steady solution. Streamlines (top) and temperature isolines (bottom). There are 14 temperature isolines with a lowest value (cold leg, beginning of the mixing region) at $T = 182\,\mathrm{K}$ and a highest value (hot leg, beginning of the mixing region) at $T = 260\,\mathrm{K}$.

Copyright © 2008 John Wiley & Sons, Ltd.

DOI: 10.1002/fld

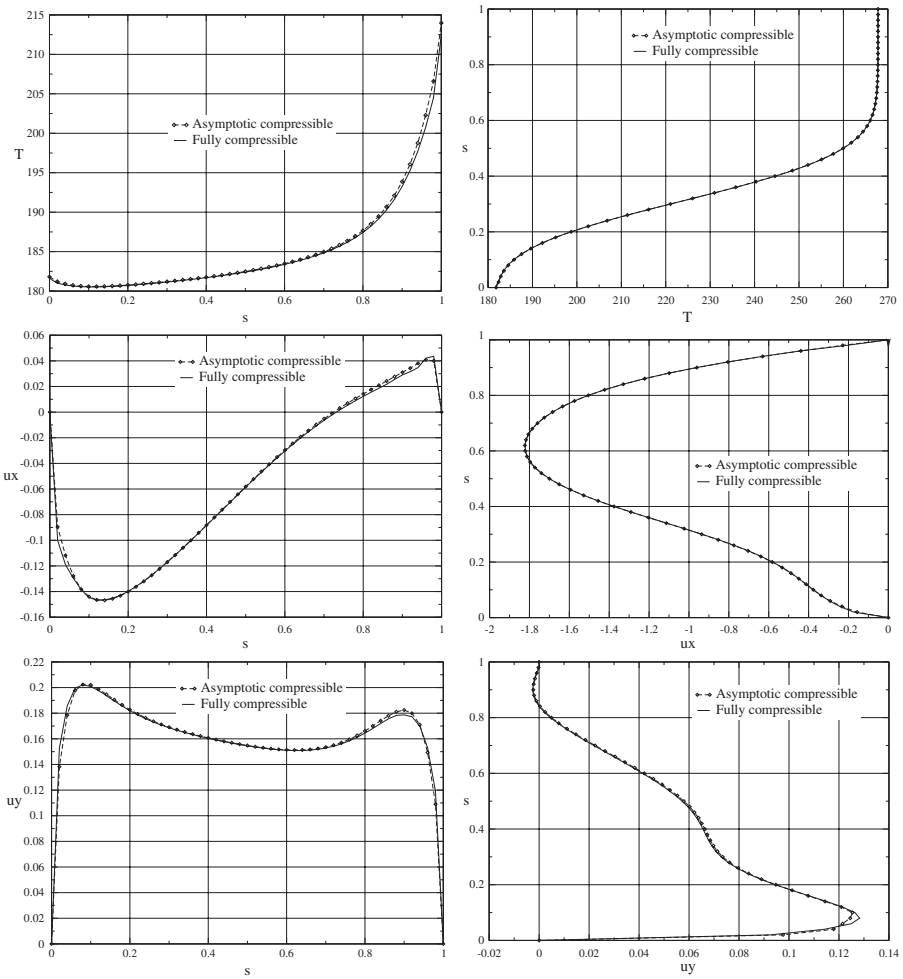

Figure 6. Steady solution for the tee junction problem. From top to bottom: distribution of temperature and velocity components (u_x, u_y) along sections $P_{22}P_{23}$ (left) and $P_{22}P_{32}$ (right),with s the curvilinear abscissa along each section. All quantities are expressed in SI units.

4.2.4. Description of the unsteady case. Starting from the previously computed steady solution, the boundary condition is modified for the inlet H. The momentum $\tilde{m}_H$ is left unchanged but the temperature $\tilde{T}_H$ is now made time dependent:

$$\tilde{T}_H(t) = \tilde{T}_H(\tilde{t}=0)(1 + A\sin(\omega\tilde{t}))$$

where A and ω denote, respectively, the amplitude and pulsation of the temperature fluctuation. These parameters are chosen so as to yield significant temperature variations in the flow field during one period: $A = 0.1$ and $\omega = 0.1\,\mathrm{s}^{-1}$. The corresponding evolution of the inlet temperature

Copyright © 2008 John Wiley & Sons, Ltd.

DOI: 10.1002/fld

Table VII. Steady flow in the tee junction. Efficiency comparison for the asymptotic pressure-based solver and the preconditioned matrix-free method.

Mesh	Elements	Method	Iterations	CPU	RCPI
1	12 540	Asymptotic	180	4185	20.5
	12 630	MF-SGS(15)	1050	1190	1
2	22 320	Asymptotic	220	7775	17.6
	22 400	MF-SGS(15)	1380	2770	1
3	34 900	Asymptotic	260	13 000	15.8
	35 000	MF-SGS(15)	2200	6950	1

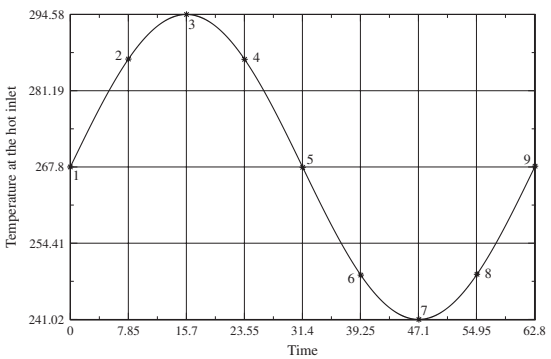

Figure 7. Unsteady solution for the tee junction. Evolution of the temperature T_H as a function of time. All quantities are expressed in SI units.

T_H over one period is plotted in Figure 7 where labels 1–9 associated with some particular points of the cycle are also indicated for later reference.

4.2.5. Analysis of the unsteady solution. The temperatures isolines in the hot tube are displayed in Figure 8 at different times ranging from $\tilde{t}=7.85\,\mathrm{s}$ to $\tilde{t}=31.4\,\mathrm{s}$ (respectively labels 2–5 on the cycle presented in Figure 7): the fluctuations of temperature at the hot inlet are clearly observed on these solutions obtained using the preconditioned matrix-free solver on the finest grid. The effects of these variations are further investigated by focusing on the temperature and velocity components distributions along section $P_{23}P_{33}$ plotted in Figure 9 for moments 1–5 of the inlet cycle. At $\tilde{t}=7.85\,\mathrm{s}$ (label 2), a large variation of the velocity $\tilde{u}_x$ is observed, whereas the fluid elements coming from the hot inlet have not yet reached the section (cf. Figure 8 top-left frame). In fact, as the inlet momentum $\tilde{m}_H$ remains constant in time, the inlet velocity varies with temperature as

$$\tilde{u}_{x,H}=\frac{\tilde{m}_H}{\tilde{P}}R\tilde{T}_H$$

Copyright © 2008 John Wiley & Sons, Ltd.

DOI: 10.1002/fld

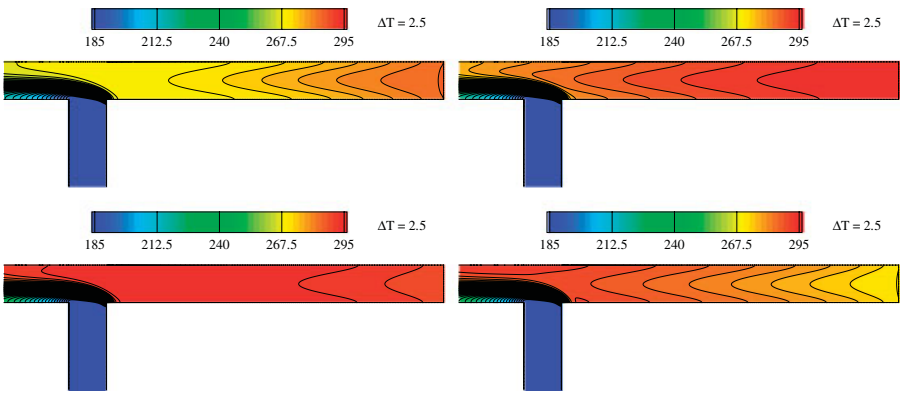

Figure 8. Unsteady solution for the tee junction computed using the preconditioned matrix-free solver. Temperature isolines (K) in the hot tube at $\tilde{t}=7.85\,\mathrm{s}$ (top-left), $\tilde{t}=15.7\,\mathrm{s}$ (top-right), $\tilde{t}=23.55\,\mathrm{s}$ (bottom-left) and $\tilde{t}=31.4\,\mathrm{s}$ (bottom-right).

Besides, according to asymptotic analysis, the velocity divergence is subjected to the following condition:

$$\tilde{\nabla}\cdot\tilde{\mathbf{u}}=\frac{\gamma-1}{\gamma\tilde{P}}\tilde{\nabla}\cdot(\tilde{\lambda}\tilde{\nabla}\tilde{T})$$

In the hot tube, that is between sections $P_{23}P_{33}$ and $P_{24}P_{34}$, the RHS of the above expression remains small so that $\tilde{\nabla}\cdot\tilde{\mathbf{u}}\approx 0$; this elliptic constraint is responsible for the fact that the variation of the inlet velocity $\tilde{u}_{x,H}$ immediately affects the speed $\tilde{u}_x$ in section $P_{23}P_{33}$. Meanwhile, the temperature does not exhibit a significant variation because, as mentioned earlier, the fluid elements arriving from the hot inlet have not yet reached the section. Since the temperature remains almost constant while the local velocity $\tilde{u}_x$ increases, the local momentum in the section increases; thus, reducing the impact of the cold fluid and diminishing the $\tilde{u}_y$ velocity component. Between $\tilde{t}=7.85\,\mathrm{s}$ (label 2) and $\tilde{t}=15.7\,\mathrm{s}$ (label 3) the fluid elements coming from the hot inlet have reached section $P_{23}P_{33}$ (cf. Figure 8 top right frame) so that the temperature variation becomes large. The temperature increase is higher in the middle of the tube where the speed is the largest, i.e. where the convective heat exchanges are the most important. The x and y velocity components have increased, with the increase of $\tilde{u}_y$ related to the decrease of the momentum $\tilde{m}_x$ with respect to its value at $\tilde{t}=7.85\,\mathrm{s}$ and also to the increase of the buoyancy force caused by the temperature increase. From $\tilde{t}=15.7$ to $23.6\,\mathrm{s}$ (label 3 and 4 in Figure 9), the velocity $\tilde{u}_x$ decreases with time as expected. Nevertheless, because of the time delay linked to the fluid transport, the temperature in the section is still increasing. From $\tilde{t}=23.6$ to $31.4\,\mathrm{s}$, the temperature decrease reaches its peak at the tube centerline where the speed—hence the convective heat exchange—is maximum. The flow behavior at other instants is qualitatively the same, namely, the $\tilde{u}_x$ velocity component immediately follows the temperature variation at the hot inlet; the temperature itself is perturbed with a delay due to convection, while the $\tilde{u}_y$ velocity component follows both temperature and momentum variations.

4.2.6. Numerical results for the unsteady case. The flow under study is usually computed using the pressure-based asymptotic solver with the non-linear system derived from the finite-element space discretization and the second-order implicit time discretization solved using a Picard-type

Copyright © 2008 John Wiley & Sons, Ltd.

DOI: 10.1002/fld

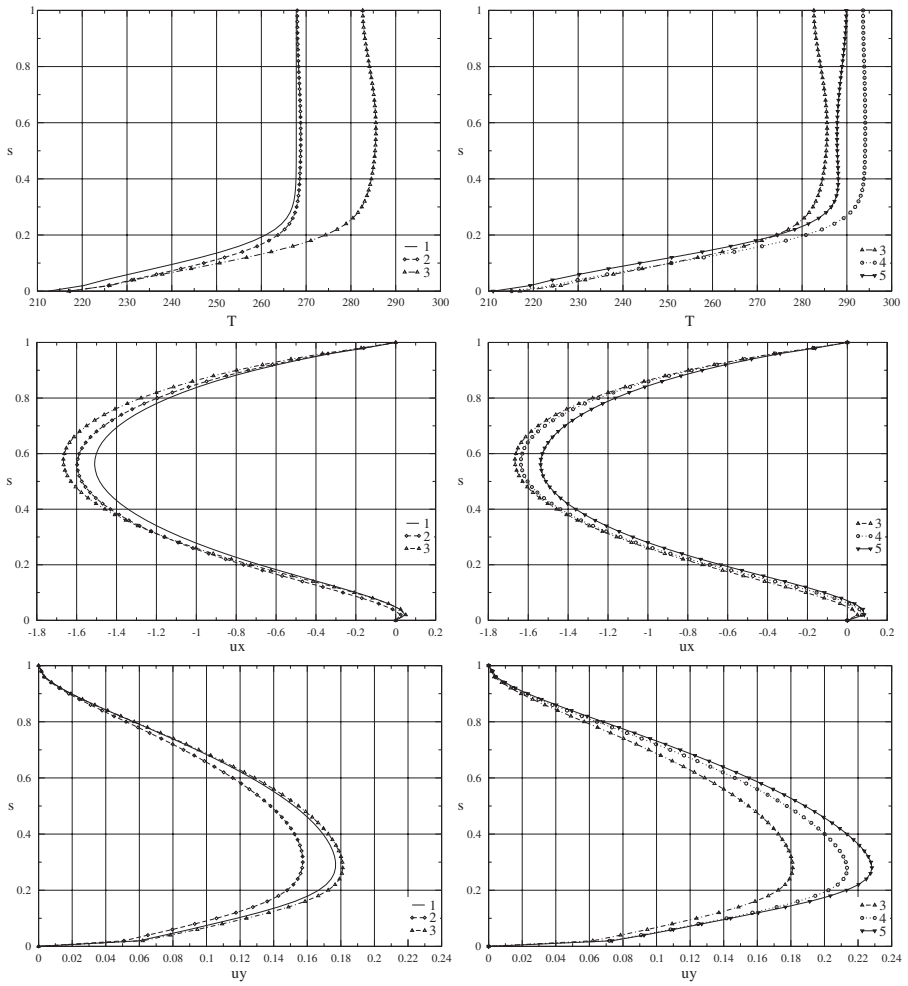

Figure 9. Unsteady solution for the tee junction. Evolution of temperature and velocity along section $P_{23}P_{33}$; time labels 1–5 refer to the cycle depicted in Figure 7. All quantities are expressed in SI units.

fixed-point algorithm. The convergence criterion between two physical time levels is based on the reduction of the temperature residual by five orders of magnitude during the iterative process. In the present study, the flow is also computed using the proposed low-cost solver for all-speed flows in a dual-time-stepping framework, with a second-order-accurate discretization of the physical time derivative and the MF-SGS scheme retained for solving the implicit stage. For unsteady flows governed by the particle wave speed, the CFL number based on that specific wave speed, that is $\mathrm{CFL}_u = u\Delta t/\delta x$, should be chosen of order unity (or less) to ensure an accurate description of the flow physics. However, larger values of this parameter are often retained in practice to achieve higher efficiency of the numerical approach. The optimal time steps, that is the values offering the best trade-off between accuracy and efficiency, have been determined for both schemes by successive trials and were taken equal to $\Delta t = (\pi/\omega)/N$ with $N = 80$ for the asymptotic

Copyright © 2008 John Wiley & Sons, Ltd.

DOI: 10.1002/fld

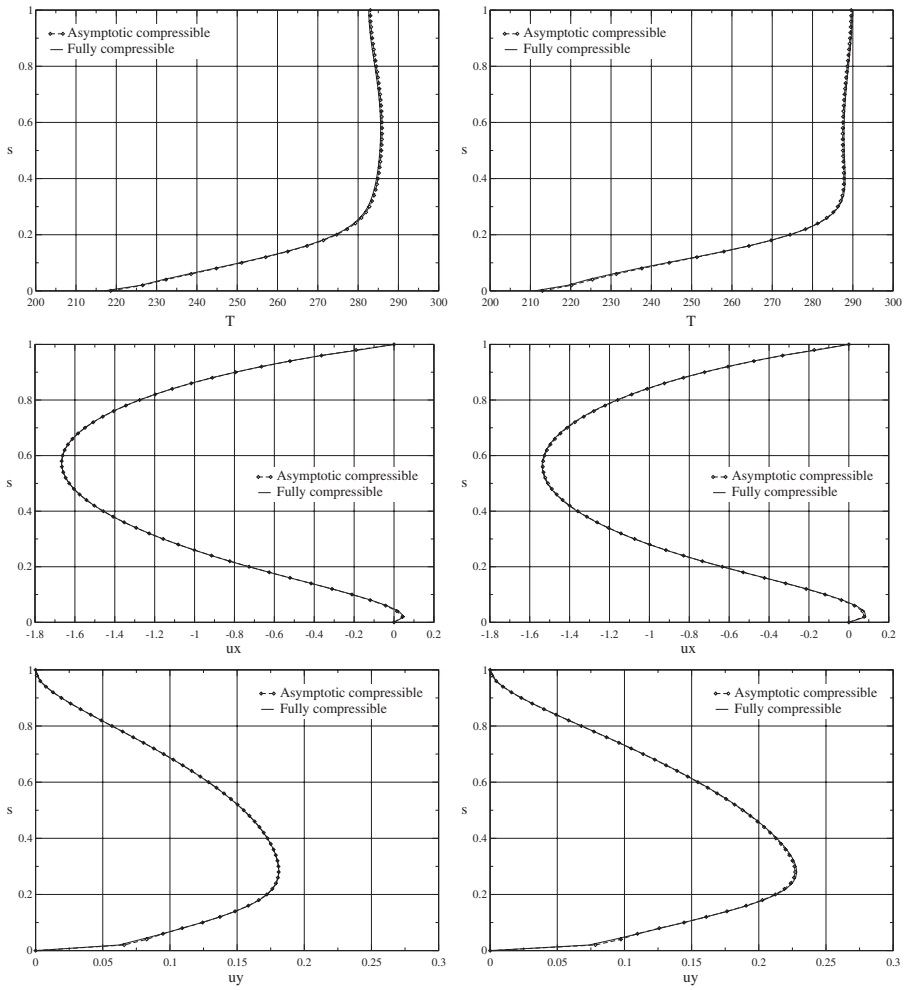

Figure 10. Tee junction. Non-stationary solution. Comparison between the pressure-based and the density-based solvers. Temperature (top frames), $\tilde{u}_x$ velocity (middle frames) and $\tilde{u}_y$ velocity (bottom frames) are represented on section $P_{23}P_{33}$ and for the following time levels: $\tilde{t} = 15.7$ and 31.4 s. All quantities are expressed in SI units.

pressure-based solver and $N = 160$ for the preconditioned scheme. Furthermore, concerning this latter approach only, a preconditioning parameter β designed for unsteady viscous flows has been used (see Appendix for its description).

The evolutions of the temperature and the velocity along the section $P_{23}P_{33}$ displayed in Figures 10 and 11 for both approaches demonstrate that the all-speed compressible flow solver provides a solution as accurate as the one obtained using the asymptotic pressure-based solver specifically devoted to such unsteady low-Mach number flows. Regarding efficiency, the main features of the fine grid computations performed on an Intel Xeon 3 GHz with 2 GM Ram memory are summarized in Table VIII: a 35% increase of the total CPU cost is observed when using the

Copyright © 2008 John Wiley & Sons, Ltd.

DOI: 10.1002/fld

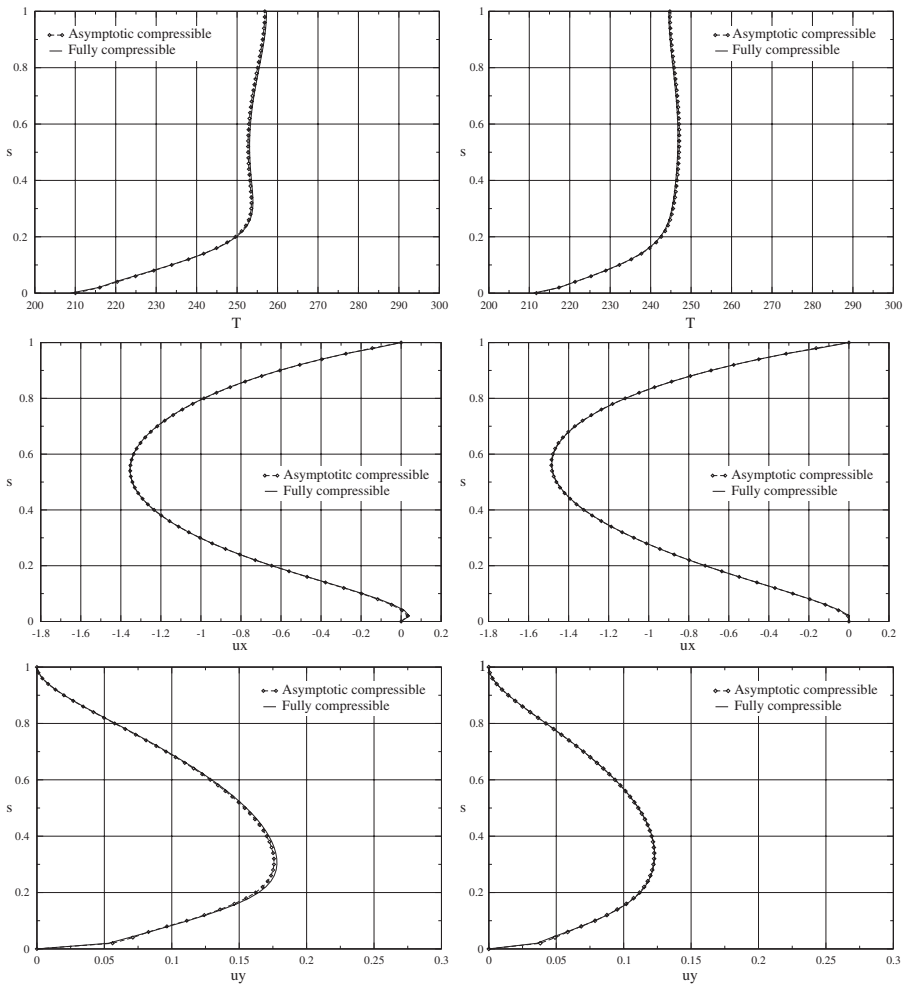

Figure 11. Tee junction. Non-stationary solution. Comparison between the pressure-based and the density-based solvers. Temperature (top frames), $\tilde{u}_x$ velocity (middle frames) and $\tilde{u}_y$ velocity (bottom frames) are represented on section $P_{23}P_{33}$ and for the following time levels: $\tilde{t}=47.1$ and 62.8 s. All quantities are expressed in SI units.

Table VIII. Unsteady flow in the tee junction. Global cost of the pressure-based asymptotic solver and the preconditioned compressible solver.

Mesh	Elements	Method	CPU
3	34 900	Asymptotic	27 000
	35 000	MF-SGS(30)	36 500

Copyright © 2008 John Wiley & Sons, Ltd.

DOI: 10.1002/fld

general-purpose preconditioned solver. This loss of performance in the unsteady case is related to the difficulty of setting an appropriate value for the preconditioning parameter when computing low-Mach unsteady flows, as already pointed out in [7].

5. CONCLUSION AND PERSPECTIVES

The development of low-Mach preconditioning techniques has made possible the application of fully compressible or density-based solvers to the computation of flows at all speeds, from the supersonic down to the nearly incompressible regime. Such general-purpose tools are particularly useful when computing flows of interest for the nuclear industry: for instance, hydrogen safety studies require the simulation of hydrogen combustion which gives rise to very weakly compressible flows when a slow deflagration takes place or for a case of flame diffusion but also to highly compressible phenomena when a denotation occurs. The method developed in the present study allows to combine the flexibility offered by low-Mach preconditioning with the very low-cost of a matrix-free implicit scheme. Taking advantage of some specific properties of the commonly used Turkel low-Mach preconditioning matrix, it was possible to derive a low-cost implicit scheme for all-speed flows. The efficiency of the scheme with respect to an existing block implicit preconditioned solver, with a better intrinsic efficiency but also a higher cost per iteration, was demonstrated on the standard test case of a steady inviscid flow in a 2D and 3D channel; the reduced memory requirements of the approach were also pointed out. The all-speed scheme was eventually compared with an existing pressure-based solver specifically devoted to the analysis of low-Mach flows of interest for the nuclear industry. When computing the steady laminar flow in a tee-junction, the new approach was found competitive with respect to the standard tool since it offered the same solution accuracy for a reduced computational time, thanks to its very low unit cost. However, in the case of unsteady flow, the all-speed treatment provided an accurate solution but for a cost higher than the existing pressure-based method. The key reason of this behavior is found in the difficulty to set a preconditioning parameter that simultaneously ensures fast convergence to steady state for the dual-time-stepping process and a proper scaling of the numerical dissipation that yields an accurate solution. The next step of this work will be therefore to develop a separate strategy for the preconditioning formulation used for the time derivative, responsible for convergence efficiency, and that used for the artificial dissipation terms, responsible for accuracy; such a formulation will be implemented within a multiple pseudo-time framework as recently discussed in [35].

APPENDIX A: CHOICE OF PRECONDITIONING PARAMETER

The preconditioning parameter β is chosen such that the eigenvalues of the preconditioned system keep the same order of magnitude when the Mach number becomes small. The following definitions have been used for the computations presented in the paper.

A.1. Inviscid steady flow

The original definition proposed by Turkel [18] has been used:

$$\beta = \min(\max(M, M_\infty), 1) \tag{A1}$$

Copyright © 2008 John Wiley & Sons, Ltd.

DOI: 10.1002/fld

where M is the local Mach number and M_∞ denotes the reference Mach number (inflow Mach number for external flows or maximum Mach number within the field for internal flows).

A.2. Viscous steady flow

The previous definition fails to ensure a correct efficiency for viscous steady flows. In that case, it is necessary to redefine the preconditioning parameter; the efficiency-preserving choice proposed by Venkateswaran and Merkle [36] is adopted:

$$\beta = \min(\max(M, \beta_v), 1) \quad \text{with } \beta_v = \frac{\mu \cdot \delta x}{\rho \cdot c} \tag{A2}$$

where μ is the local dynamic viscosity, ρ is the density and c is the local speed of sound.

A.2.1. Viscous unsteady flow. The previous definition does not provide fast convergence to a pseudo-time steady state when viscous unsteady flows are computed. An alternative expression for the preconditioning parameter was thus given by Venkateswaran and Merkle [36]:

$$\beta = \min(\max(M, \beta_v, \beta_u), 1) \quad \text{with } \beta_u = \frac{l_{\text{ref}}}{\pi \cdot \Delta t \cdot c} \tag{A3}$$

where the length l_{ref} is usually taken as the length of the computational domain. However, as pointed out in [7], such a definition may prevent a proper scaling of the numerical dissipation and lead to inaccurate solutions. The following strategy was thus adopted for the unsteady computations presented in this paper: a large cut-off value was first set at the beginning of the dual-time-stepping process using (A3), then this value was progressively reduced throughout the dual-time convergence process until reaching a value of the order of the Mach number, corresponding to (A2). Such a choice allowed to recover an accurate solution but at the expense of a poorer efficiency.

REFERENCES

1. Riggins D, Walters R, Pelletier D. The use of direct solvers for compressible flow computations. *AIAA Paper* 1988; 88-0229.
2. Khalfallah K, Lacombe G, Lerat A. Analysis of implicit treatments for a centred Euler solver. *Computers and Fluids* 1993; **22**:381–406.
3. Pulliam T, Chaussee D. A diagonal form of an implicit approximate-factorization algorithm. *Journal of Computational Physics* 1981; **39**:347–363.
4. Buelow P, Schwer D, Feng J, Merkle C. A preconditioned dual-time, diagonalized ADI scheme for unsteady computations. *AIAA Paper* 1997; 97-2101.
5. Buelow P, Venkateswaran S, Merkle C. Stability and convergence analysis of implicit upwind schemes. *Computers and Fluids* 2001; **30**:961–988.
6. Caughey D. Implicit multigrid computation of unsteady flows past cylinders of square cross-section. *Computers and Fluids* 2001; **30**:939–960.
7. Pandya S, Venkateswaran S, Pulliam T. Implementation of preconditioned dual-time procedures in OVERFLOW. *AIAA Paper*, 2003; 2003-0072.
8. Lerat A, Sides J, Daru V. *An Implicit Finite-volume Method for Solving the Euler Equations*. Lecture Notes in Physics, vol. 17. Springer: Berlin, 1982; 343–349.
9. Jameson A, Baker T. Solution of the Euler equations for complex configurations. *AIAA Paper*, 1983; 83-1929.
10. Jameson A, Schmidt W, Turkel E. Numerical solutions of the Euler equations by finite volume methods using Runge–Kutta time-stepping schemes. *AIAA Paper*, 1981; 81-1259.
11. Jameson A, Turkel E. Implicit schemes and LU decompositions. *Mathematics of Computations* 1981; **37**:385–397.

Copyright © 2008 John Wiley & Sons, Ltd.

DOI: 10.1002/fld

12. Jameson A, Yoon S. Lower-upper implicit schemes with multiple grids for the Euler equations. *AIAA Journal* 1987; **25**:929–935.
13. Sharov D, Nakahashi K. Reordering of 3-D hybrid unstructured grids for vectorized LU-SGS Navier–Stokes computations. *AIAA Paper*, 1997; 97-0617.
14. Luo H, Baum J, Löhner R. A fast, matrix-free implicit method for compressible flows on unstructured grids. *Journal of Computational Physics* 1998; **146**:664–690.
15. Luo H, Baum J, Löhner R. An accurate, fast, matrix-free implicit method for computing unsteady flows on unstructured grids. *Computers and Fluids* 2001; **30**:137–159.
16. Luo H, Baum J, Löhner R. A fast, matrix-free implicit method for computing low-Mach number flows on unstructured grids. *International Journal of Computational Fluid Dynamics* 2001; **14**:133–157.
17. Luo H, Baum J, Löhner R. Extension of Harten–Lax-van Leer scheme for flows at all speeds. *AIAA Journal* 2005; **43**:1160–1166.
18. Turkel E. Preconditioned methods for solving the incompressible and low Mach compressible equations. *Journal of Computational Physics* 1987; **72**:277–298.
19. Turkel E. Preconditioning techniques in CFD. *Annual Review of Fluid Mechanics* 1999; **31**:385–416.
20. Beccantini A, Corre C, Kloczko T. A matrix-free implicit method for flows at all speeds. In *Computational Fluid Dynamics 2004—Proceedings of ICCFD3*, Toronto, 12–16 July 2004, Groth C, Zingg D (eds), 2006; 129–134.
21. Turkel E, Vatsa V. Local preconditioners for steady and unsteady flow applications. *ENSAIM*: *Mathematical Modelling and Numerical Analysis* 2005; **39**:515–536.
22. Barth T, Jespersen D. The design and application of upwind schemes on unstructured meshes. *AIAA Paper*, 1989; 89-0366.
23. Venkatakrishnan V. Convergence to steady-state solutions of the Euler equations on unstructured grids with limiters. *Journal of Computational Physics* 1995; **118**:120.
24. Roe PL. Approximate Riemann solvers, parameters vectors and difference schemes. *Journal of Computational Physics* 1981; **43**:357–372.
25. Edwards J, Liou M. Low-diffusion flux-splitting methods for flows at all speeds. *AIAA Journal* 1998; **36**: 1610–1617.
26. Noh W. CEL: a time-dependent two-space dimensional coupled Eulerian–Lagrangian code. *Methods in Computational Physics* 1964; **3**:117–179.
27. Pulliam T, MacCormack R, Venkateswaran S. Convergence characteristics of several approximate factorization methods. *16th International Conference on Numerical Methods in Fluid Dynamics*, Lecture Notes in Physics, vol. 515. Arcachon (France), 1998.
28. Corre C, Lerat A. Stability and efficiency of implicit residual-based compact schemes. In *Computing the Future IV—Frontiers of CFD—2004*, Hafez M, Caughey D (eds). 2005.
29. Guillard H, Viozat C. On the behaviour of upwind schemes in the low Mach number limit. *Computers and Fluids* 1999; **28**:63–86.
30. Shende N, Arora K, Balakrishnan N. Convergence acceleration using implicit relaxation procedures for cell centre and cell vertex finite volume schemes. *Fluid Mechanics Report 2001FM03*, Indian Institute of Science, Bangalore, India, 2001.
31. Hu LW, Nagasawa K, Hejzlar P, Kazimi M. Thermal stripping in LWR piping systems. *Technical Report MIT-NSP-TR-007*, Center for Advanced Nuclear Energy Systems (CANES), MA, U.S.A., 2001.
32. Hu LW, Lee J, Saha P, Kazimi M. Thermal stripping in LWR piping system. *Technical Report MIT-NSP-TR-017*, Center for Advanced Nuclear Energy Systems (CANES), MA, U.S.A., 2003.
33. Chapuliot S, Gourdin C, Payen T, Magnaud J, Monavon A. Hydro-thermal-mechanical analysis of thermal fatigue in a mixing tee. *Nuclear Engineering and Design* 2005; **39**:617.
34. Paillère H, Quéré PL, Weisman C, Vierendeels J, Dick E, Braack M, Dabbene F, Beccantini A, Studer E, Kloczko T, Corre C, Heuveline V, Darbandi M, Hosseinizadeh S. Modelling of natural convection flows with large temperature differences: a benchmark problem for low Mach number solvers. Part 2: contributions to the June 2004 Conference. *ENSAIM*: *Mathematical Modelling and Numerical Analysis* 2005; **39**:617–621.
35. Venkateswaran S, Merkle C, Zeng X, Li D. Influence of large scale pressure changes on preconditioned solutions at low speeds. *AIAA Journal* 2004; **42**:2490–2498.
36. Venkateswaran S, Merkle C. Analysis of preconditioning methods for Euler and Navier–Stokes equation. *Technical Report 1999-03*, VKI Lecture Series, 1999.

Copyright © 2008 John Wiley & Sons, Ltd.

DOI: 10.1002/fld