Available online at www.sciencedirect.com
ScienceDirect
Journal of Computational Physics 220 (2007) 879–899
JOURNAL OF COMPUTATIONAL PHYSICS
www.elsevier.com/locate/jcp

Efficient computation of compressible and incompressible flows

Cord-Christian Rossow *

DLR, Deutsches Zentrum für Luft- und Raumfahrt, Institut für Aerodynamik und Strömungstechnik, Lilienthalplatz 7, 38108 Braunschweig, Germany

Received 29 December 2005; received in revised form 26 May 2006; accepted 29 May 2006
Available online 17 July 2006

Abstract

The combination of explicit Runge–Kutta time integration with the solution of an implicit system of equations, which in earlier work demonstrated increased efficiency in computing compressible flow on highly stretched meshes, is extended toward conditions where the free stream Mach number approaches zero. Expressing the inviscid flux Jacobians in terms of Mach number, an artificial speed of sound as in low Mach number preconditioning is introduced into the Jacobians, leading to a consistent formulation of the implicit and explicit parts of the discrete equations. Besides extension to low Mach number flows, the augmented Runge–Kutta/Implicit method allowed the admissible Courant–Friedrichs–Lewy number to be increased from O(100) to O(1000). The implicit step introduced into the Runge–Kutta framework acts as a preconditioner which now addresses both, the stiffness in the discrete equations associated with highly stretched meshes, and the stiffness in the analytical equations associated with the disparity in the eigenvalues of the inviscid flux Jacobians. Integrated into a multigrid algorithm, the method is applied to efficiently compute different cases of inviscid flow around airfoils at various Mach numbers, and viscous turbulent airfoil flow with varying Mach and Reynolds number. Compared to well tuned conventional methods, computation times are reduced by half an order of magnitude.

MSC: 65P05; 77A99

Keywords: Computational fluid dynamics; Numerical methods; Multigrid methods; Convergence acceleration; Runge–Kutta/Implicit time integration; Compressible and incompressible flow; Euler and Navier–Stokes equations

1. Introduction

Over the last decades, numerical simulation of fluid flow has evolved from a topic only addressed in basic research toward a tool routinely used for research and design. Today's maturation of computational fluid dynamics (CFD) has enabled the computation of flow around complex aerospace configurations such as complete airplanes, helicopters, and spacecraft [1–4]. Furthermore, in the last years the combination of CFD with other disciplines like structural mechanics and flight mechanics [5–7] has found widespread attention and

* Tel.: +49 531 295 2400; fax: +49 531 295 2320.
E-mail address: cord.rossow@dlr.de.

0021-9991/$ - see front matter © 2006 Elsevier Inc. All rights reserved.
doi:10.1016/j.jcp.2006.05.034

application. This rapid development of numerical flow simulation was driven by both, the successful research and subsequent advancement in efficient solution algorithms, and the continuous increase in available computational power. Having reached today's level of maturity in numerical algorithms, it is tempting to assume that further progress in the applicability of numerical methods may be guaranteed by solely relying on the sustained development of computer technology. However, since relevant problem size will continue to increase as fast as available hardware permits, a number of severe challenges in the development of numerical methods for flow simulation remain. If the requirements of future complex, multidisciplinary applications shall be met, these challenges have to be addressed rigorously, despite the progress in algorithmic development achieved so far. Globally, without going into detail, these challenges may be summarized by the terms efficiency, robustness, and accuracy.

With respect to efficiency, one of the major breakthroughs in numerical methods for flow simulation was the introduction of multigrid [8,9], and for the solution of the inviscid equations, numerical methods may now be considered as fairly effective, without stating that no further improvements may be necessary or possible [10]. In contrast to that, methods for the computation of viscous flow can not be considered as equally mature. This is mainly caused by the inadequacy of today's methods to efficiently take into account the stiffness of the discrete system of equations. Discrete stiffness is provoked by two distinct sources [11]: the first results from the use of a scalar time step which is unable to cope with the disparity in the propagation speeds of convective and acoustic modes. The second source of discrete stiffness is introduced by the highly stretched computational meshes required for economical resolution of boundary layers in high Reynolds number flows. This second source is of far more serious concern than the first, since the corresponding high cell aspect ratios increase discrete stiffness by several orders of magnitude in large portions of the computational domain, resulting in severe convergence problems and very high computation times. One of the first successful attempts to address discrete geometrical stiffness is represented by the work of Martinelli [12], where coefficients of the implicit residual smoothing used in combination with explicit Runge–Kutta time integration were formulated as functions of cell aspect ratio. Subsequently, preconditioning of the discrete equations was used to mitigate the problem of discrete stiffness [11,13,14]. In Ref. [15], the approach which proved to be highly successful to solve the 2D-Euler equations [10] was extended to viscous flows, and favorable convergence rates were obtained for laminar flow with Reynolds numbers as high as 80,000.

Recently, a Runge–Kutta/Implicit method was proposed where the widespread strategy of combining multigrid with Runge–Kutta time integration [9,12,16] was augmented by replacing the implicit residual smoothing with the solution of an implicit system using Symmetric Gauss–Seidel iteration [17]. Here, turbulent flow around airfoils for Reynolds numbers in the order of 6×10^6 was computed, and the approach reduced the number of iterations required for convergence by about a factor of 8, where CPU-time was more than halved. Further work in Ref. [18] demonstrated the ability of the Runge–Kutta/Implicit method to efficiently compute flows with Reynolds numbers up to the order of 10^8 and corresponding cell aspect ratios of about 50,000.

Besides the efficiency in computing flow on meshes with very high cell aspect ratios, another matter of concern in algorithmic research is the robustness of numerical methods with respect to the applicability to both, compressible and incompressible flows. The difficulty here is caused by the disparity in the eigenvalues of the convective flux Jacobians in the compressible equations when approaching the incompressible limit: at low speeds, the largest eigenvalue tends toward the speed of sound, whereas the smallest eigenvalue approaches zero. Thus, the condition number of the system of equations tends to infinity and the stiffness of the system increases. Since this stiffness is not tied to a particular discretization, but solely associated with the analytic equations, it was characterized as analytical stiffness in Ref. [11]. To overcome analytical stiffness when computing low-speed flows with compressible methods, usually the system of compressible equations is 'preconditioned' for easier solution with iterative methods by multiplying the time derivatives with a suitable matrix [19–22]. However, these low-speed preconditioning matrices may become singular at stagnation points or in recirculation regions [23], and robustness of the methods may be impaired [24].

As an alternative to preconditioning the compressible equations for low Mach number problems, codes primarily designed for incompressible flows are extended toward the compressible flow regime [25–27]. When

computing incompressible flows, such methods respect the constraint of a divergence-free velocity field by solving a Poisson equation for pressure, which is not directly accounted for in the preconditioning approach [28]. Using this pressure based approach, in general the strong conservation form of the governing equations is not strictly respected [25–27], and up to now such methods did not gain widespread acceptance in the aerospace community.

In the present work, the discrete stiffness associated with high aspect ratio cells and the analytic stiffness at low Mach numbers shall be addressed by a unified approach. The Runge–Kutta/Implicit scheme introduced in Ref. [17], where the solution of an implicit system of equations was embedded into the framework of Runge–Kutta time integration, will serve as the basis for coping with discrete geometrical stiffness. For extending compressible codes to the incompressible flow regime, in Refs. [28,29] a suitable equation for pressure was derived by exploiting principles of pressure based methods, and the role of an artificial speed of sound to appropriately scale terms of the numerical dissipation was discussed. To address analytical stiffness in the Runge–Kutta/Implicit method, this artificial speed of sound will be introduced into the implicit system similarly to the formulation of the pressure equation in Refs. [28,29] to allow efficient computation of low-speed flows. The performance of the proposed method will be assessed by computing steady inviscid flow at various Mach numbers, and steady viscous flows at different Reynolds and Mach numbers.

2. Governing equations

We consider the two-dimensional Navier–Stokes equations for compressible flow. For a control volume fixed in time and space, the system of partial differential equations in integral form is given by

$$\int\int_{Vol} \frac{\partial \boldsymbol{W}}{\partial t}\,\mathrm{d}V + \int_S \boldsymbol{F}\cdot\boldsymbol{n}\,\mathrm{d}S = 0, \tag{1}$$

where $\boldsymbol{W}$ represents the vector of conservative variables, $\boldsymbol{F}$ is the flux-density tensor, and Vol, S, and $\boldsymbol{n}$ denote volume, surface, and outward facing normal of the control volume. The flux density tensor $\boldsymbol{F}$ may be split into an inviscid, convective part $\boldsymbol{F}_\mathrm{c}$ and a viscous part $\boldsymbol{F}_\mathrm{v}$:

$$\boldsymbol{F} = \boldsymbol{F}_\mathrm{c} + \boldsymbol{F}_\mathrm{v}, \tag{2}$$

where $\boldsymbol{F}_\mathrm{c}$ and $\boldsymbol{F}_\mathrm{v}$ are given by

$$\boldsymbol{F}_\mathrm{c} = \begin{bmatrix} \rho\boldsymbol{q} \\ \rho u\boldsymbol{q} + p\boldsymbol{i}_x \\ \rho v\boldsymbol{q} + p\boldsymbol{i}_y \\ \rho H\boldsymbol{q} \end{bmatrix}, \quad \boldsymbol{F}_\mathrm{v} = \begin{bmatrix} 0 \\ \tau_{xx}\boldsymbol{i}_x + \tau_{xy}\boldsymbol{i}_y \\ \tau_{xy}\boldsymbol{i}_x + \tau_{yy}\boldsymbol{i}_y \\ \left(u\tau_{xx} + v\tau_{xy} + k\frac{\partial T}{\partial x}\right)\boldsymbol{i}_x + \left(u\tau_{xy} + v\tau_{yy} + k\frac{\partial T}{\partial y}\right)\boldsymbol{i}_y \end{bmatrix}, \tag{3}$$

where $\boldsymbol{q}$ is the velocity vector with Cartesian components u, v, and $\boldsymbol{i}_x$, $\boldsymbol{i}_y$ denote the unit vectors in direction of the Cartesian coordinates x and y. The variables ρ, p, H, T represent density, pressure, total specific enthalpy, and temperature, k is the coefficient of thermal heat conductivity, and τ_{xx}, τ_{yy}, τ_{xy} are the viscous stress tensor components.

In order to close the system given by Eq. (1), the equation of state

$$\frac{p}{\rho} = R\cdot T \tag{4}$$

is used with R as specific gas constant.

3. Basic solution scheme

The basic solution scheme employs a cell centered, finite volume space discretization on structured meshes [30], and time integration is achieved by combining an explicit Runge–Kutta scheme with the solution of an implicit system of equations [17]. Using the finite volume technique for space discretization, a semi-discrete form of Eq. (1) may be written as

$$Vol\frac{\partial \boldsymbol{W}}{\partial t}+\sum_{\text{all faces}}\boldsymbol{F}\cdot\boldsymbol{n}S=0, \tag{5}$$

where Vol now represents the volume of a computational cell and S is the area of a cell face. Eq. (5) can be rearranged to

$$\frac{\partial \boldsymbol{W}}{\partial t}+\frac{1}{Vol}\sum_{\text{all faces}}\boldsymbol{F}_n S=0, \tag{6}$$

where $\boldsymbol{F}_n$ is the flux density vector corresponding to the direction normal to the cell face. For evaluation of the flux density vector on a cell face, the flux difference splitting (FDS) technique [31] is used. The convective part of the flux density vector $\boldsymbol{F}_c$ normal to a cell interface may then be written as:

$$\boldsymbol{F}_c=\frac{1}{2}\left(\boldsymbol{F}^L+\boldsymbol{F}^R\right)-\frac{1}{2}|\boldsymbol{A}_n|\cdot\Delta\boldsymbol{W}, \tag{7}$$

where $\boldsymbol{F}^L$ and $\boldsymbol{F}^R$ are the left and right states of the inviscid flux density vector normal to the cell interface, $\boldsymbol{A}_n$ is the corresponding flux Jacobian, and $|\boldsymbol{A}_n|$ represents the Jacobian with the modulus of all eigenvalues. The term $\Delta\boldsymbol{W}$ denotes differences in conservative variables on the left side L and right side R of a cell interface, giving $\Delta\boldsymbol{W}=\boldsymbol{W}^R-\boldsymbol{W}^L$, where the normal vector $\boldsymbol{n}$ at a cell interface is pointing from left to right when the cell boundary is traversed in a mathematically positive sense. Following Ref. [30], $|\boldsymbol{A}_n|\cdot\Delta\boldsymbol{W}$ is expressed in terms of the cell interface Mach number M_0, with M_0 defined as:

$$M_o=\min(|M|,1)\cdot\operatorname{sign}(M). \tag{8}$$

The resulting expressions of $|\boldsymbol{A}_n|\cdot\Delta\boldsymbol{W}$ are summarized in Table 1 as dissipative flux differences $\Delta\boldsymbol{F}$ for the continuity, momentum, and energy equations. In Table 1, q_n denotes the velocity normal to the cell interface defined as $q_n=\boldsymbol{q}\cdot\boldsymbol{n}$, where $\boldsymbol{n}=(n_x,n_y)^T$, and the operator Δ indicates differences of variables between right and left states of the cell interface. For second order accuracy, the variables ρ, ρu, ρv, ρH are reconstructed, and the Symmetric LImited Positive (SLIP) limiter [32] is used according to the implementation outlined in Ref. [33]. Discretization of the viscous terms $\boldsymbol{F}_v$ is performed by central differences, and implementation details may be found in Refs. [16,30].

In the basic code, time integration of Eq. (6) is achieved by a Runge–Kutta/Implicit method, which combines an explicit 5-stage Runge–Kutta scheme with the solution of an implicit system of equations. The Runge–Kutta/Implicit time integration method was first formulated in Ref. [17] in order to alleviate the Courant–Friedrichs–Lewy (CFL) restriction of the explicit Runge–Kutta scheme and to address discrete stiffness. As outlined in Refs. [34,35], an implicit step also substantially increases the damping properties of the basic Runge–Kutta time integration, provided the implicit system is solved without factorization error. Thus, when using a properly constructed Runge–Kutta/Implicit method as relaxation scheme in a multigrid algorithm, considerable improvements in convergence rates are achievable [17]. For completeness, the construction of the Runge–Kutta/Implicit method will be summarized following the outline in Ref. [17].

Using a purely explicit Runge–Kutta scheme, conservative variables $\boldsymbol{W}$ are updated by

$$\boldsymbol{W}_{i,j}^{(m)}=\boldsymbol{W}_{i,j}^{(0)}+\alpha^{(m)}\cdot\boldsymbol{R}_{i,j}^{(m-1)}, \tag{9}$$

where the superscripts (m) denote the stage count with m running from 1 to the maximum number of stages, the subscripts i,j correspond to the location of control volumes in the flow field, and $\alpha^{(m)}$ is the stage coefficient of the (m)-stage. The vector $\boldsymbol{R}_{i,j}^{(m-1)}$ represents the conservative residuals of continuity, momentum and energy equations evaluated using the variables of the previous $(m-1)$-stage:

Table 1
Flux difference splitting dissipation expanded in terms of Mach number

ΔF_ρ =	$\frac{1}{c}(1-\lvert M_0\rvert)\Delta p$		$+\rho M_0\Delta q_n$	$+\lvert q_n\rvert\Delta\rho$
$\Delta F_{\rho u}=n_xM_0\Delta p$	$+\frac{1}{c}u(1-\lvert M_0\rvert)\Delta p$	$+n_x\rho c(1-\lvert M_0\rvert)\Delta q_n$	$+\rho uM_0\Delta q_n$	$+\lvert q_n\rvert\Delta\rho u$
$\Delta F_{\rho v}=n_yM_0\Delta p$	$+\frac{1}{c}v(1-\lvert M_0\rvert)\Delta p$	$+n_y\rho c(1-\lvert M_0\rvert)\Delta q_n$	$+\rho vM_0\Delta q_n$	$+\lvert q_n\rvert\Delta\rho v$
$\Delta F_{\rho E}$ =	$\frac{1}{c}H(1-\lvert M_0\rvert)\Delta p+q_nM_0\Delta p$	$+q_n\rho c(1-\lvert M_0\rvert)\Delta q_n$	$+\rho HM_0\Delta q_n$	$+\lvert q_n\rvert\Delta\rho E$

$M_0=\min(|M|,1)\cdot\operatorname{sign}(M)$.

$$\boldsymbol{R}_{i,j}^{(m-1)} = -\frac{\delta t_{i,j}}{Vol_{i,j}} \sum_{\text{all faces}} \boldsymbol{F}_n(\boldsymbol{W}^{m-1})S, \tag{10}$$

with $\delta t_{i,j}$ as the local time step of cell i, j.

To accelerate convergence toward steady state, at each stage the explicit Runge–Kutta time stepping of Eq. (9) is augmented by an implicit step [17], which is derived from an implicit formulation of Eq. (6):

$$\frac{\partial \boldsymbol{W}}{\partial t} + \frac{1}{Vol} \sum_{\text{all faces}} \boldsymbol{F}_n^{(n+1)} S = 0, \tag{11}$$

where $\boldsymbol{F}_n^{(n+1)}$ is evaluated at the new time level $(n+1)$. Linearizing $\boldsymbol{F}_n^{(n+1)}$ about the current time level (n), one obtains

$$\boldsymbol{F}_n^{(n+1)} = \boldsymbol{F}_n^{(n)} + \left(\frac{\partial \boldsymbol{F}_n}{\partial \boldsymbol{W}}\right)\delta \boldsymbol{W} = \boldsymbol{F}_n^{(n)} + \boldsymbol{A}_n \delta \boldsymbol{W}, \tag{12}$$

with $\delta \boldsymbol{W}$ defining the time difference of conservative variables, and $\boldsymbol{A}_n$ is as denoted previously the flux Jacobian in the direction normal to a cell face S:

$$\delta \boldsymbol{W} = \boldsymbol{W}^{(n+1)} - \boldsymbol{W}^{(n)}, \quad \boldsymbol{A}_n = \frac{\partial \boldsymbol{F}_n}{\partial \boldsymbol{W}}. \tag{13}$$

Substituting Eq. (12) into Eq. (11) and replacing the time derivative in Eq. (11) with a finite difference approximation one obtains

$$\frac{\delta \boldsymbol{W}}{\delta t} + \frac{1}{Vol} \sum_{\text{all faces}} \boldsymbol{A}_n \delta \boldsymbol{W} S = -\frac{1}{Vol} \sum_{\text{all faces}} \boldsymbol{F}_n^{(n)} S. \tag{14}$$

Rearranging leads to

$$\left(\boldsymbol{I} + \frac{\delta t}{Vol} \sum_{\text{all faces}} \boldsymbol{A}_n S\right)\delta \boldsymbol{W} = -\frac{\delta t}{Vol} \sum_{\text{all faces}} \boldsymbol{F}_n^{(n)} S = \boldsymbol{R}^{(n)}, \tag{15}$$

where $\boldsymbol{R}^{(n)}$ has been introduced to denote the residual at current time level (n), and $\boldsymbol{I}$ represents the Identity matrix. A consistent linearization of the second order accurate space discretization of the explicit residual $\boldsymbol{R}^{(n)}$ would require a second order discretization for the implicit operator on the left hand side, resulting in Newton's method for infinitely large time steps [36]. However, such methods incur very large memory requirements due to the necessary storage of neighbor-of-neighbor information, and robustness problems occur as long as the method is not sufficiently close to its zone of attraction, leading to substantial difficulties in the start-up process of a computation. Therefore, in Ref. [17] only a first order representation for the implicit operator is employed, as is commonly practiced when constructing implicit iterative methods.

The matrix $\boldsymbol{A}_n$ in Eq. (15) has real eigenvalues and may be split into two matrices $\boldsymbol{A}_n^+$ and $\boldsymbol{A}_n^-$,

$$\boldsymbol{A}_n^+ = 0.5(\boldsymbol{A}_n + |\boldsymbol{A}_n|), \quad \boldsymbol{A}_n^- = 0.5(\boldsymbol{A}_n - |\boldsymbol{A}_n|), \tag{16}$$

and with definition of Eq. (16), the implicit system of Eq. (15) for $\delta \boldsymbol{W}$ may be rewritten as:

$$\left(\boldsymbol{I} + \frac{\delta t_{i,j}}{Vol_{i,j}} \sum_{\text{all faces}} \boldsymbol{A}_n^+ S\right)\delta \boldsymbol{W}_{i,j} = \boldsymbol{R}_{i,j}^{(n)} - \frac{\delta t_{i,j}}{Vol_{i,j}} \sum_{\text{all faces}} \boldsymbol{A}_n^- \delta \boldsymbol{W}_{\text{NB}} S, \tag{17}$$

where indices i, j denote the current cell, and NB are all direct neighbors of cell i, j. In Eq. (17) $\boldsymbol{A}_n^- \delta \boldsymbol{W}$ represents the flux density change associated with waves having negative wave speed, i.e. waves that enter cell i, j from outside. Only neighbor cells NB can contribute to these changes in flux density. Similarly, $\boldsymbol{A}_n^+ \delta \boldsymbol{W}$ represents flux density changes associated with positive wave speeds, i.e. waves which leave cell i, j. These flux density changes are determined only by information from within cell i, j.

Eq. (17) is combined with the explicit Runge–Kutta scheme of Eq. (9) by interpreting the changes of conservative variables $\delta \boldsymbol{W}$ in Eq. (17) as new residuals $\tilde{\boldsymbol{R}}$

$$\left(\boldsymbol{I}+\frac{\delta t_{i,j}}{Vol_{i,j}}\sum_{\text{all faces}}\boldsymbol{A}_n^+S\right)\tilde{\boldsymbol{R}}_{i,j}=\boldsymbol{R}_{i,j}^{(m-1)}-\frac{\delta t_{i,j}}{Vol_{i,j}}\sum_{\text{all faces}}\boldsymbol{A}_n^-\tilde{\boldsymbol{R}}_{\text{NB}}S. \tag{18}$$

Note that the residual $\boldsymbol{R}^{(n)}$ of current time level (n) in Eq. (17) has been replaced by the residual of the previous Runge–Kutta stage $\boldsymbol{R}^{(m-1)}$. The new residuals $\tilde{\boldsymbol{R}}$ obtained from the solution of Eq. (18) are then used in the Runge–Kutta framework given by Eq. (9) for updating conservative variables:

$$\boldsymbol{W}_{i,j}^{(m)}=\boldsymbol{W}_{i,j}^{(0)}+\alpha^{(m)}\cdot\tilde{\boldsymbol{R}}_{i,j}. \tag{19}$$

Despite the use of only a first order approximation on the left hand side, storing the split flux Jacobians matrices $\boldsymbol{A}_n^+$ and $\boldsymbol{A}_n^-$ in Eq. (17) or (18) still requires a substantial amount of computational memory. For solution of a 2D (3D) problem, these are 4×4 (5×5) matrices, requiring storage of 32 (50) variables per cell face. Additionally, the necessary matrix times vector operations are associated with a high operation count, and as a consequence implicit methods based on Eq. (17) are generally not applied to solve for practical problems. Instead, LU schemes with simplified Jacobians [37,38] or matrix free methods [39] are frequently employed. However, these simplifications lead to deterioration of damping properties, resulting in degraded convergence especially when solving the Navier–Stokes equations, even though sometimes high CFL numbers can be used. In contrast to that, in Ref. [17] storage requirements are reduced without any further approximations by formulating the implicit system in primitive variables $\boldsymbol{U}=[\rho,p,u,v]^{\text{T}}$ and expressing the flux Jacobians in terms of Mach number [30]. The implicit system of Eq. (17) formulated in primitive variables reads:

$$\left(\boldsymbol{I}+\frac{\delta t_{i,j}}{Vol_{i,j}}\sum_{\text{all faces}}\boldsymbol{P}_n^+S\right)\delta\boldsymbol{U}_{i,j}=\boldsymbol{Q}_{i,j}^{(n)}-\frac{\delta t_{i,j}}{Vol_{i,j}}\sum_{\text{all faces}}\boldsymbol{P}_n^-\delta\boldsymbol{U}_{\text{NB}}S, \tag{20}$$

where $\delta\boldsymbol{U}$ denotes the temporal changes of primitive variables, and $\boldsymbol{Q}^{(n)}$ represents the explicit residuals in primitive variables computed from the conservative explicit residuals $\boldsymbol{R}^{(n)}$ by using the associated Jacobians to transform from conservative to primitive variables:

$$\boldsymbol{Q}_{i,j}^{(n)}=\frac{\partial\boldsymbol{U}}{\partial\boldsymbol{W}}\boldsymbol{R}_{i,j}^{(n)}. \tag{21}$$

The matrices $\boldsymbol{P}_n^+$ and $\boldsymbol{P}_n^-$ in Eq. (20) are the flux Jacobians in primitive variables, which correspond to the definition in conservative variables of Eq. (16). The product of $\boldsymbol{P}_n^+$ and $\boldsymbol{P}_n^-$ with the changes $\delta\boldsymbol{U}$ can be expressed in an efficient, storage reducing way [17]:

$$\begin{aligned}
\boldsymbol{P}_n^+\cdot\delta\boldsymbol{U}&=\begin{bmatrix}(q_n+|q_n|)\delta\rho+ & \frac{1}{c}(1-|M_0|)\delta p & + & \rho(1+M_0)\delta q_n\\ (q_n+|q_n|)\delta p+ & (\gamma-1)\frac{h}{c}(1-|M_0|)\delta p & + & \gamma p(1+M_0)\delta q_n\\ (q_n+|q_n|)\delta u+ & n_x\frac{1}{\rho}(1+M_0)\delta p & + & n_x c(1-|M_0|)\delta q_n\\ (q_n+|q_n|)\delta v+ & n_y\frac{1}{\rho}(1+M_0)\delta p & + & n_y c(1-|M_0|)\delta q_n\end{bmatrix},\\
\boldsymbol{P}_n^-\cdot\delta\boldsymbol{U}&=\begin{bmatrix}(q_n-|q_n|)\delta\rho- & \frac{1}{c}(1-|M_0|)\delta p & + & \rho(1-M_0)\delta q_n\\ (q_n-|q_n|)\delta p- & (\gamma-1)\frac{h}{c}(1-|M_0|)\delta p & + & \gamma p(1-M_0)\delta q_n\\ (q_n-|q_n|)\delta u+ & n_x\frac{1}{\rho}(1-M_0)\delta p & - & n_x c(1-|M_0|)\delta q_n\\ (q_n-|q_n|)\delta v+ & n_y\frac{1}{\rho}(1-M_0)\delta p & - & n_y c(1-|M_0|)\delta q_n\end{bmatrix},
\end{aligned} \tag{22}$$

where the temporal change of normal velocity δq_n is defined as:

$$\delta q_n=n_x\delta u+n_y\delta v, \tag{23}$$

and γ, h, c denote the ratio of specific heats, specific enthalpy, and speed of sound, respectively.

The definitions for $\boldsymbol{P}^+\delta\boldsymbol{U}$ and $\boldsymbol{P}^-\delta\boldsymbol{U}$ by Eq. (22) allow an implementation of the implicit scheme of Eq. (20) with a comparatively low computational effort: only the expressions $|q_n|$, M_0, $(1-|M_0|)$ are pre-computed and stored for each cell face, all other components can efficiently be recomputed whenever necessary. The contri-

butions of the viscous flux Jacobians are incorporated with the definitions outlined in Ref. [40] for primitive variables.

The implicit system in primitive variables is combined with the basic Runge–Kutta scheme analogously to Eq. (18) by interpreting the changes $\delta \boldsymbol{U}$ as new primitive residuals $\tilde{\boldsymbol{Q}}$:

$$\left(\boldsymbol{I} + \frac{\delta t_{i,j}}{Vol_{i,j}} \sum_{\text{all faces}} \boldsymbol{P}_n^+ S\right) \tilde{\boldsymbol{Q}}_{i,j} = \boldsymbol{Q}_{i,j}^{(m-1)} - \frac{\delta t_{i,j}}{Vol_{i,j}} \sum_{\text{all faces}} \boldsymbol{P}_n^- \tilde{\boldsymbol{Q}}_{\text{NB}} S. \tag{24}$$

Solution of Eq. (24) is achieved with symmetric lexicographic Gauss–Seidel (SGS) sweeps, where the explicit residuals $\boldsymbol{Q}^{(m-1)}$ are used as initial conditions for the unknown residuals $\tilde{\boldsymbol{Q}}$ [17]. After solution of Eq. (24), the new residuals $\tilde{\boldsymbol{Q}}$ are transformed to conservative residuals $\tilde{\boldsymbol{R}}$ by

$$\tilde{\boldsymbol{R}}_{i,j} = \frac{\partial \boldsymbol{W}}{\partial \boldsymbol{U}} \tilde{\boldsymbol{Q}}_{i,j}, \tag{25}$$

to update conservative variables in the Runge–Kutta framework according to Eq. (19).

This Runge–Kutta/Implicit time integration, comprised of the Runge–Kutta scheme of Eq. (19) combined with the solution of the fully implicit system of Eq. (24), allowed CFL numbers of O(100) [17,18]. Time integration is further enhanced by employing a multigrid algorithm following the ideas of Jameson [9]. The influence of turbulence is modeled according to Baldwin and Lomax [41].

In Ref. [17], this basic method was applied to compute two-dimensional transonic, turbulent flow around airfoils for Reynolds numbers in the range of $Re_\infty = 6.5 \times 10^6$. Using meshes of C-type topology, convergence rates of about 0.85 were obtained for residual reduction by more than 12 orders of magnitude, and computation times were more than halved compared to other methods commonly in use [17,18].

4. Extension of the basic approach to low Mach number flow

To use compressible methods for the computation of nearly incompressible flows where $M \to 0$, it is well known that the artificial dissipation of the spatial discretization needs to be scaled appropriately [19], since the discrete equations support pressure disturbances of O(M), in contrast to the analytical equations which only support O(M^2) disturbances [42,43]. For the FDS-discretization of Eq. (7) this scaling can easily be achieved by replacing the speed of sound c in Table 1 by an artificial speed of sound c' [30], which is of the order of the mean flow speed [19,42,43]. This strongly amplifies the dominance of the pressure differences multiplied by $\frac{1}{c'}(1 - |M_0|)$ given in Table 1 [28–30].

Using the appropriately scaled dissipative terms in the explicit residual, one may now expect an implicit scheme as represented by Eq. (15) to enable the computation of incompressible flow, since the disparity in propagation speeds should be taken into account by solving the implicit system. However, when combining the implicit step with the Runge–Kutta scheme, the implicit system is not solved until convergence, but only approximately with three SGS-sweeps [17,18]. Consequently, any stiffness caused by a disparity in propagation speeds may reduce the efficiency of the limited number of SGS-sweeps and will result in lower CFL-bounds. Since in this work only steady flows are computed and local time stepping is used as an acceleration technique, time accuracy is destroyed. Therefore, a manipulation of the propagation speeds may be considered as further means to alleviate the stiffness of the implicit system. As indicated above, the implicit operator on the left hand side of Eq. (15) should match the explicit right hand side as closely as possible. Consequently, when for low Mach number flows the artificial speed of sound c' is introduced into the discretization of the explicit residual [30], this should be matched accordingly in the implicit operator on the left hand side in order to obtain similar efficiency to that in Ref. [17]. Note that for the pressure equation of the implicit system (24), this scaling leads to a similar formulation as already employed in Refs. [28,29], where, however, only the pressure equation was regarded and not the whole system of equations.

The expressions $\boldsymbol{P}_{\boldsymbol{n}}^{+}\cdot\delta\boldsymbol{U}$ and $\boldsymbol{P}_{\boldsymbol{n}}^{-}\cdot\delta\boldsymbol{U}$ in Eq. (22) then become

$$\boldsymbol{P}_{n}^{+}\cdot\delta\boldsymbol{U}=\begin{bmatrix}(q_n+|q_n|)\delta\rho+ & \frac{1}{c'}(1-|M_0|)\delta p & + & \rho(1+M_0)\delta q_n\\ (q_n+|q_n|)\delta p+ & (\gamma-1)\frac{h}{c'}(1-|M_0|)\delta p & + & \gamma p(1+M_0)\delta q_n\\ (q_n+|q_n|)\delta u+ & n_x\frac{1}{\rho}(1+M_0)\delta p & + & n_x c'(1-|M_0|)\delta q_n\\ (q_n+|q_n|)\delta v+ & n_y\frac{1}{\rho}(1+M_0)\delta p & + & n_y c'(1-|M_0|)\delta q_n\end{bmatrix},$$

$$\boldsymbol{P}_{n}^{-}\cdot\delta\boldsymbol{U}=\begin{bmatrix}(q_n-|q_n|)\delta\rho- & \frac{1}{c'}(1-|M_0|)\delta p & + & \rho(1-M_0)\delta q_n\\ (q_n-|q_n|)\delta p- & (\gamma-1)\frac{h}{c'}(1-|M_0|)\delta p & + & \gamma p(1-M_0)\delta q_n\\ (q_n-|q_n|)\delta u+ & n_x\frac{1}{\rho}(1-M_0)\delta p & - & n_x c'(1-|M_0|)\delta q_n\\ (q_n-|q_n|)\delta v+ & n_y\frac{1}{\rho}(1-M_0)\delta p & - & n_y c'(1-|M_0|)\delta q_n\end{bmatrix}. \tag{26}$$

The artificial speed of sound c' in Eq. (26) is evaluated as [28,30]:

$$\begin{aligned}c'&=\sqrt{\alpha^2q^2+M_r^2c^2},\\ \alpha&=\frac{1}{2}(1-M_r^2),\end{aligned} \tag{27}$$

where q represents the flow speed, and M_r is a reference Mach number defined as

$$M_r^2=\min\left[\max\left(\frac{q^2}{c^2},k\frac{q_\infty^2}{c_\infty^2}\right),1\right], \tag{28}$$

with k set to unity.

When using the artificial speed of sound in Eq. [26], for low speed flows the terms including pressure fluctuations on the left hand side of the discrete equations are strongly amplified, thus matching the behavior of the corresponding artificial dissipative terms on the right hand side. Note, however, that in the scaling term $\frac{h}{c'}(1-|M_0|)$ of the pressure equation, the specific enthalpy h has to be evaluated with the physical speed of sound to achieve this amplification [29]. The artificial speed of sound c' is also used to compute the admissible local time step, which leads to a substantial increase in δt when computing low Mach number flows [19]. Furthermore, for computation of low Mach number flows, the initial values of the unknown residuals $\tilde{\boldsymbol{Q}}$ in Eq. (24) are set to zero when solving the implicit system by SGS sweeps, thus deviating from the practice of Refs. [17,18], where the explicit residuals $\boldsymbol{Q}^{(m-1)}$ were used for initialization.

The Runge–Kutta/Implicit scheme, where the implicit step of Eq. (18) is combined with the explicit Runge–Kutta scheme of Eq. (9), may be viewed as using a preconditioner $\boldsymbol{P}$ to the Runge–Kutta scheme:

$$\boldsymbol{W}_{i,j}^{(m)}=\boldsymbol{W}_{i,j}^{(0)}+\alpha^{(m)}\cdot\boldsymbol{P}\boldsymbol{R}_{i,j}^{(m-1)}, \tag{29}$$

where the inverse of preconditioner $\boldsymbol{P}$ is given by the left hand side of Eq. (15)

$$\boldsymbol{P}^{-1}=\left(\boldsymbol{I}+\frac{\delta t}{Vol}\sum_{\text{all faces}}\boldsymbol{A}_nS\right). \tag{30}$$

In the actual implementation, $\boldsymbol{P}^{-1}$ is defined by Eq. (24) and Eqs. (21), (25) for transformation to and from primitive variables. When employing Eq. (26) to include the artificial speed of sound c' for the low Mach number extension, the preconditioner $\boldsymbol{P}$ defined by Eq. (30) then addresses both, discrete stiffness due to high cell aspect ratios, and analytical stiffness due to the disparity in the eigenvalues of the flux Jacobians.

5. Computational results

For all computations presented in this study, a 4-level W-cycle is employed in the multigrid algorithm, and the corresponding coarse meshes are created by successively omitting every second grid line. On the finest mesh, a second order space discretization is employed, which on coarse meshes is reduced to first order. For time integration generally a 5 stage Runge–Kutta scheme with coefficients of Ref. [44] for second order spatial accuracy is employed, combined with solution of the implicit system (24) at each stage. Unless noted

otherwise, three Symmetric Gauss–Seidel sweeps are used for iterative solution of the implicit system given by Eq. (24) with initializing the unknowns by zero. All computations presented are performed on a SGI Octane workstation using a single 360 MHz processor.

5.1. Addressing analytical stiffness for low mach number flows

First, the proposed method is applied to compute inviscid flow around the NACA 0012 airfoil at various Mach numbers. The computational mesh is an O-mesh with 160 cells around the airfoil and 32 cells in normal direction. Fig. 1 shows pressure distributions obtained for free stream conditions $M_\infty = 0.001$, $\alpha = 2.0°$; $M_\infty = 0.63$, $\alpha = 2.0°$; and $M_\infty = 0.8$, $\alpha = 1.25°$.

In Fig. 2, convergence histories are displayed for a residual reduction by 12 orders of magnitude. As is common practice with implicit methods, the CFL number is increased to its final value after a start-up phase [36]. For all inviscid cases, during the first 4 multigrid cycles the CFL number is set to CFL = 40, and then for all

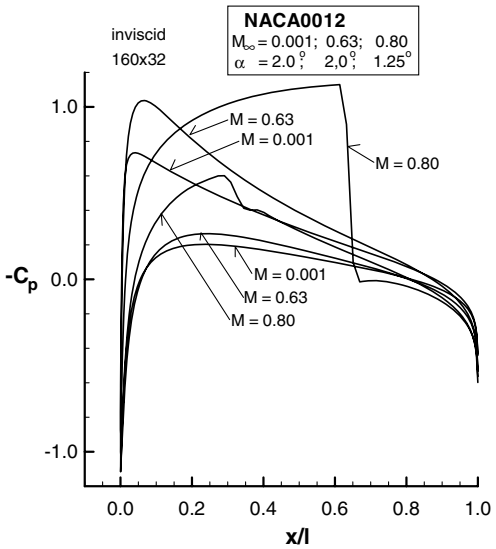

Fig. 1. Surface pressure distributions for inviscid flow around NACA0012 airfoil at different free stream Mach numbers.

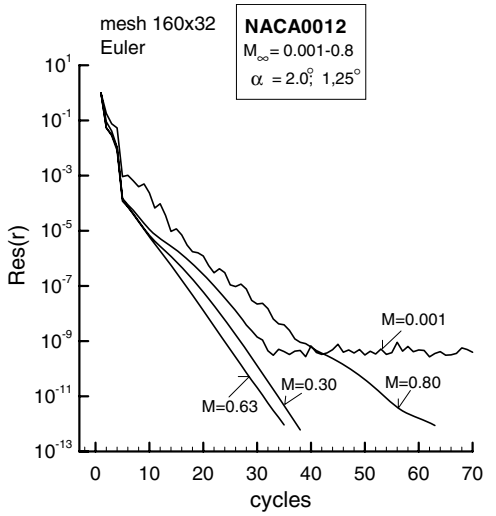

Fig. 2. Convergence histories for computation of inviscid flow around NACA0012 airfoil at different free stream Mach numbers.

subsequent iterations it is raised to CFL = 1000. The corresponding residuals are normalized with the residual of the first iteration for each case. For most of the purely subsonic cases, convergence rates between 0.45 and 0.5 are obtained, and under transonic conditions the observed rate is 0.65. The deterioration in convergence for the transonic case is suspected to be caused by the limiter function, expressed by the peaks in the corresponding convergence history. For the nearly incompressible case at $M_\infty = 0.001$, convergence stalls after nine orders of magnitude in residual reduction, caused by appearance of round-off errors when reducing the free stream Mach number [45]. This is further illustrated in Fig. 3, where convergence histories are shown for free stream Mach numbers $M_\infty = 0.001; 0.01; 0.1; 0.3$. As can be seen in Fig. 3, the level where the residual stalls is reduced with increasing Mach number: for $M_\infty \geqslant 0.1$ the residual is decreased by twelve orders of magnitude without showing round-off errors. For Mach numbers of $M_\infty \leqslant 0.1$, the convergence rate becomes independent of the Mach number until reaching machine zero. Removal of round-off errors at low Mach numbers may be achieved by introducing a gauge pressure [45].

Second, a Mach number variation is performed for viscous, turbulent flow employing the present method. Here, the same structured C-mesh around the RAE2822 airfoil as in Ref. [17] is used with 64 cells in normal direction, 320 cells in circumferential direction, and 256 cells are located on the airfoil surface. As in the inviscid cases, the 5 stage Runge–Kutta scheme is used with a final CFL number of CFL = 1000, but start-up is achieved with 8 multigrid cycles and a CFL number of CFL = 16, corresponding to the practice also employed in Ref. [17]. No attempts were made to optimize the start-up procedure for different cases, but the same procedure is used for all viscous computations considered in this study. Note that in Ref. [17] after the first 8 multigrid cycles the CFL number could only be raised to CFL = 160. Mach number is successively reduced from $M_\infty = 0.75$ to $M_\infty = 0.001$, with the Reynolds number set to $Re_\infty = 6.5 \times 10^6$ and the angle of attack held fixed at $\alpha = 2.8°$. In Fig. 4, the computed pressure distributions are displayed showing a qualitative variation with Mach number as in the inviscid investigations. Corresponding convergence histories are displayed in Fig. 5, with the residuals normalized as for the inviscid cases. Reducing residuals by eight orders of magnitude, convergence rates vary between 0.67 and 0.75. Similarly to the inviscid cases, for $M_\infty \leqslant 0.1$ convergence becomes independent of the free stream Mach number, confirming the successful extension of the method to compute nearly incompressible flows.

5.2. Addressing discrete stiffness for Reynolds number variation

The computations above demonstrate the ability of the present method to alleviate analytical stiffness in low Mach number flows. Next, discrete stiffness on meshes with high aspect ratio cells will be addressed.

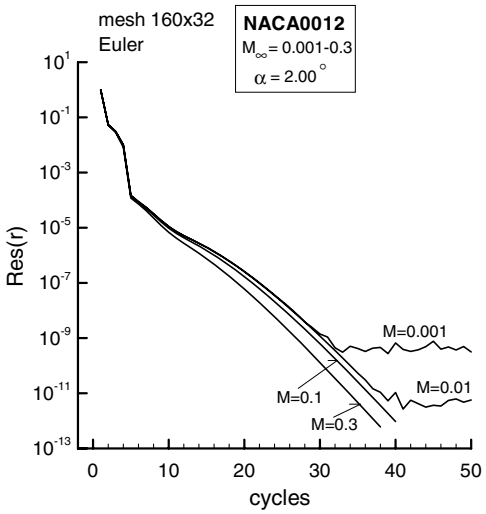

Fig. 3. Influence of round-off error for computations at low Mach number free stream conditions.

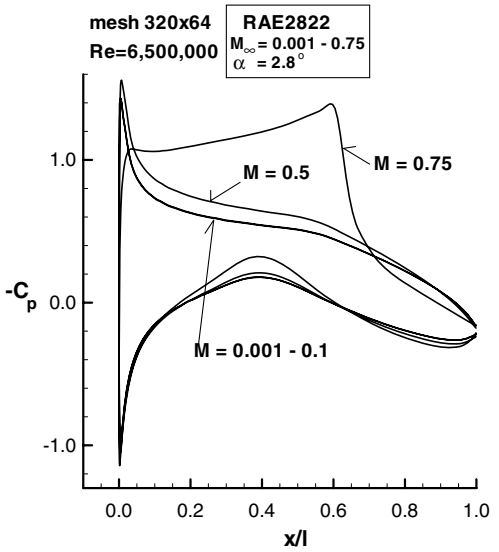

Fig. 4. Surface pressure distributions for viscous turbulent flow around RAE2822 airfoil at different free stream Mach numbers.

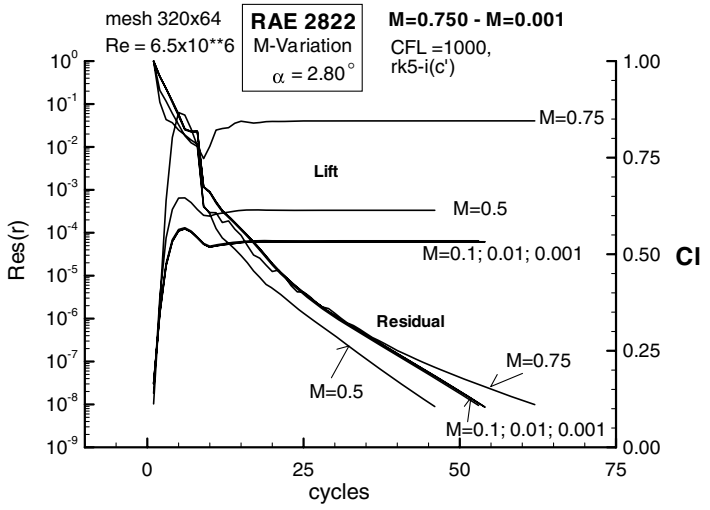

Fig. 5. Convergence histories of residual and lift coefficient for Mach number variation of viscous flow around RAE2822 airfoil.

A detailed study with respect to discrete stiffness was conducted in Ref. [18], where the performance of the basic method was investigated by computing flows at various Reynolds numbers. In that study, starting from the subsonic conditions of Case 1 in the investigations of the RAE2822 airfoil by Cook et al. [46], at fixed Mach number and angle of attack, Reynolds number was varied by more than an order of magnitude from $Re = 5.7 \times 10^6$ to 100×10^6. For the computational meshes a C-topology with 312 cells along the airfoil, 56 cells in the wake region, and 88 cells in normal direction was used. Keeping topology and number of cells identical, the meshes were adapted to the corresponding Reynolds numbers, leading to cell aspect ratios varying from about 3000 to over 50,000 [47]. These meshes are also used here to investigate the influence of introducing an artificial speed of sound according to Eq. (26). Following Ref. [18], a pure second order reconstruction of variables without limiting is employed to prevent distortion of convergence by numerical noise in the limiting procedure on different meshes. Fig. 6 shows a comparison of results from Ref. [18] and the present method. Similar to previous cases, for the present method the admissible CFL number is increased from

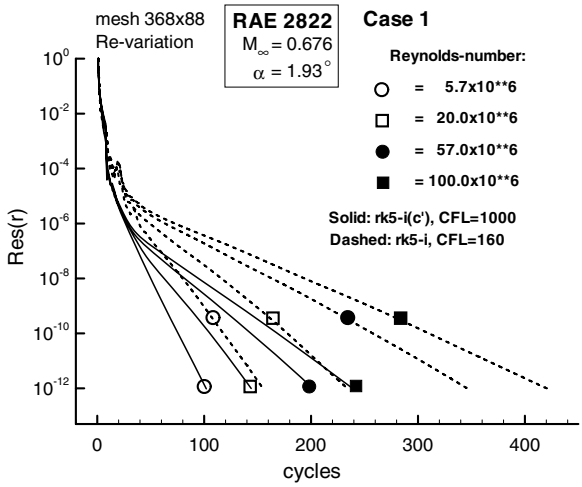

Fig. 6. Convergence histories for Reynolds number variation, starting from Case 1 (rk5-i(c'): present method with artificial speed of sound, CFL = 1000; rk5-i: basic method without artificial speed of sound, CFL = 160).

CFL = 160 to CFL = 1000. For the lowest Reynolds number of $Re = 5.7 \times 10^6$, the present method reduces the number of multigrid cycles required for convergence by about 25% compared to the basic method, and for the highest Reynolds number of $Re = 100 \times 10^6$ the reduction amounts to almost 50%. This is a substantial improvement with respect to the basic method, which already showed superior performance in computing flows on highly stretched meshes [18].

To address both, analytical and discrete stiffness, the study of Reynolds number variation is extended to the computation of nearly incompressible flow. Using the same meshes and Reynolds numbers as before, the Mach number is reduced to $M_\infty = 0.001$. Results are displayed in Fig. 7, and it can be seen that equivalent efficiency as in the compressible cases is achieved.

5.3. Efficiency of the Runge–Kutta/Implicit scheme

To investigate the performance of the present method compared to the basic scheme, the same compressible flow cases as in Ref. [17] are computed. These cases comprise Case 1, Case 9, and Case 10 from the investigations of Cook et al. [46] for turbulent flow around the RAE2822 airfoil. For Case 1, the flow field remains

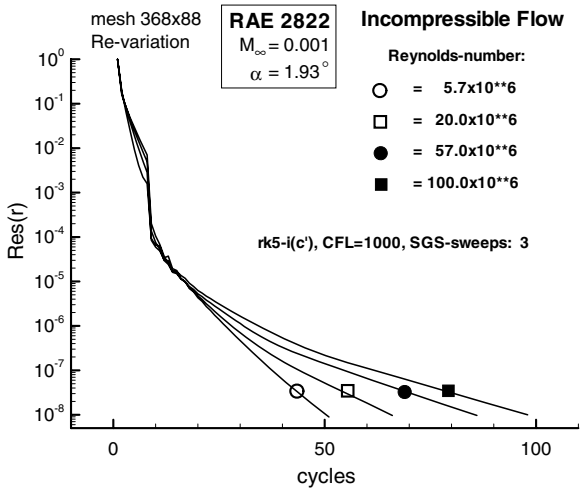

Fig. 7. Convergence histories for Reynolds number variation for incompressible flow (rk5-i(c'): present method with artificial speed of sound, CFL = 1000).

Table 2
Free stream conditions of test cases for RAE2822 airfoil

Case	M_∞	α [deg]	Re	x_{tr}/l
Case 1	0.676	1.93	5,700,000	0.11
Case 9	0.730	2.80	6,500,000	0.03
Case 10	0.750	2.80	6,200,000	0.03

mainly subsonic and no shock wave occurs. Case 9 exhibits a fairly strong shock wave on the upper surface of the airfoil, and for Case 10, the shock strength is increased, leading to substantial separation behind the shock, which often severely degrades convergence of numerical methods. The different free stream conditions are listed in Table 2. The computational grid with 320×64 cells is identical to that used for the Mach number variation with results displayed in Figs. 3 and 4.

5.3.1. Comparison of basic and present method

To establish a sound basis for comparison of computational efficiency, all methods are implemented into the same computational framework, and computations are always made on the same SGI Octane workstation. Thus, the only difference between the present and the basic method is the employment of Eq. (26) instead of Eq. (22), all other components of the code are identical. In the present method, after start-up a CFL number of CFL = 1000 could be used for all cases, which is a clear improvement with respect to the basic method, where the allowable CFL number could only be increased to a maximum of CFL = 160 [17]. The pressure distributions obtained with the present method are very similar to those presented in Ref. [17] with the basic method and will not be repeated here. Table 3 displays total force coefficients obtained with both, the basic method [17], and the method proposed in this study using the artificial speed of sound in Eq. (26) and Table 1. Results marked with 'rk5-i' denote the use of a 5-stage Runge–Kutta scheme combined with the implicit system of Eq. (24), corresponding to the results of Ref. [17] with the basic Runge–Kutta/Implicit method. The indication 'rk5-i(c')' denotes computations using the present method with artificial speed of sound according to Eq. (26). The results of both methods are quite similar and agree well with numerical results presented elsewhere in the literature [16,33,48]. Note that the differences in steady state results are caused by the use of the artificial speed of sound in the spatial discretization of Table 1 and not by employing Eq. (26) instead of Eq. (22), since the converged solution is independent from time integration [36]. Using an artificial speed of sound in the spatial discretization is necessary when employing low speed preconditioning for the computation of nearly incompressible flows [19–22]. In the transonic speed regime this influence becomes negligible, as confirmed by the similarity in steady state results of present and basic method.

Figs. 8–10 show the corresponding convergence histories for a residual reduction by 13 orders of magnitude. As was outlined in Ref. [17], the basic method reduced necessary computation times by more than a factor of 2 compared to conventional, well tuned methods. For the present method, the implementation of the artificial speed of sound c' into the discrete equations is performed with almost no additional computational effort, and the reduction in the number of multigrid cycles by about 15% using the present method therefore directly corresponds to a similar reduction of CPU time when compared to the basic method. Table 4 gives a detailed comparison of the basic method 'rk5-i' and the present method 'rk5-i(c')'with respect to CPU time, number of iterations, and convergence rate. Additionally, corresponding data of a conventional, well tuned method following Refs. [32,33,30] with Runge–Kutta time stepping, implicit residual smoothing [12], and mul-

Table 3
Total forces computed for RAE2822 airfoil

Case	Method	c_l	c_{d_total}	$c_{d\_pressure}$	$c_{d\_friction}$
Case 1	$rk5-i$	0.60875	0.00840	0.00246	0.00594
	$rk5-i(c')$	0.60863	0.00832	0.00242	0.00590
Case 9	$rk5-i$	0.85260	0.01784	0.01223	0.00561
	$rk5-i(c')$	0.85249	0.01778	0.01221	0.00557
Case 10	$rk5-i$	0.84503	0.02866	0.02316	0.00550
	$rk5-i(c')$	0.84535	0.02866	0.02319	0.00547

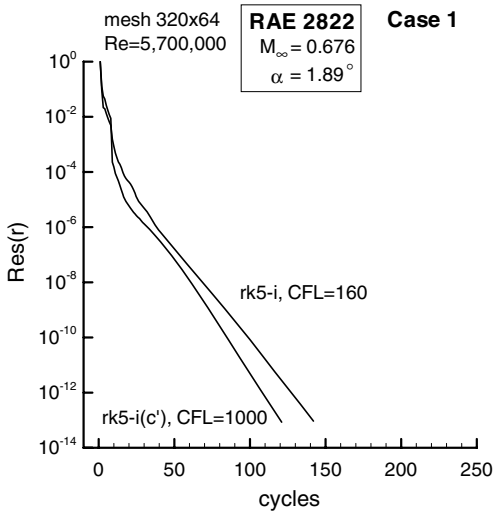

Fig. 8. Convergence history for computation of Case 1 with 4-level multigrid (rk5-i(c'): present method with artificial speed of sound, CFL = 1000; rk5-i: basic method without artificial speed of sound, CFL = 160).

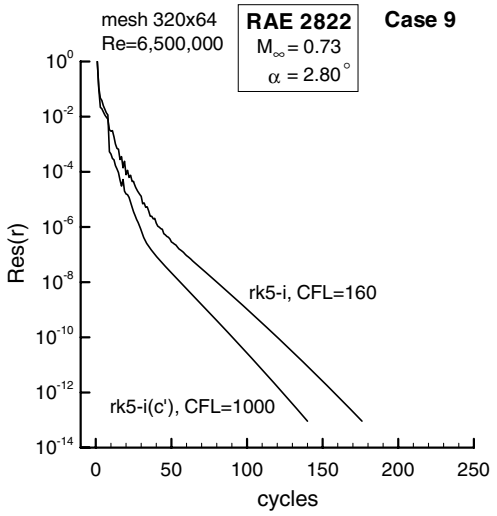

Fig. 9. Convergence history for computation of Case 9 with 4-level multigrid (rk5-i(c'): present method with artificial speed of sound, CFL = 1000; rk5-i: basic method without artificial speed of sound, CFL = 160).

tigrid [16] are included in Table 4. For consistent comparison, this well tuned conventional method is implemented into the same computational framework as basic and present method, and efficiency is measured on the same machine. These results are marked by 'rk5/3-s', denoting a 5-stage Runge–Kutta scheme with the standard smoothing of Ref. [12], where the artificial viscosity is evaluated only at every odd stage to increase efficiency [12,16], and CFL = 7.5 [16,30,33]. The spatial discretization of method 'rk5/3-s' is the same as that of method 'rk5-i, therefore steady state results of the two methods are identical [17]. With respect to the reference method 'rk5/3-s', for all cases the present method reduces CPU time by roughly a factor of 2.5.

5.3.2. Influence of number of Runge–Kutta stages, artificial speed of sound, and initialization of implicit system

In Ref. [17], a 3 stage Runge–Kutta scheme with coefficients $[0.15, 0.4, 1.0]$ was employed for further reduction of CPU-time with respect to the basic 5 stage scheme. However, in Ref. [17] the admissible CFL number had to be reduced from CFL = 160 to CFL = 100, leading to an increased number of iterations, see results

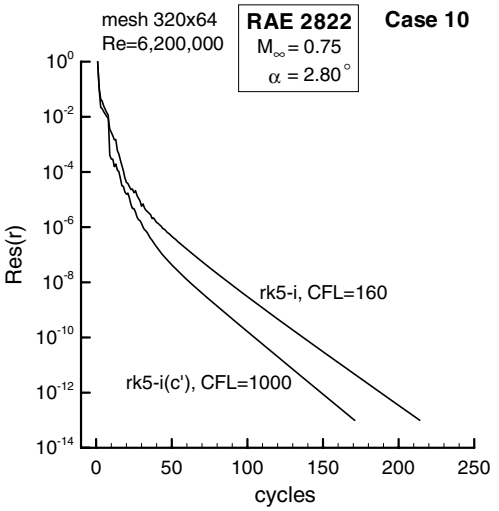

Fig. 10. Convergence history for computation of Case 10 with 4-level multigrid (rk5-i(c'): present method with artificial speed of sound, CFL = 1000; rk5-i: basic method without artificial speed of sound, CFL = 160).

Table 4
Computational effort required for RAE 2822 test cases

	CPU time [s]	# iterations	Convergence rate
Case 1			
rk5/3-s	2349	1191	0.975
rk5-i	1206	142	0.801
rk5-i(c')	1047	121	0.779
rk3-i	1023	190	0.854
rk3-i(c')	702	117	0.776
Case 9			
rk5/3-s	3129	1585	0.981
rk5-i	1495	176	0.843
rk5-i(c')	1210	140	0.807
rk3-i	1337	248	0.886
rk3-i(c')	842	137	0.803
Case 10			
rk5/3-s	3929	2019	0.985
rk5-i	1818	214	0.869
rk5-i(c')	1480	171	0.839
rk3-i	1552	288	0.901
rk3-i(c')	997	168	0.836

denoted by 'rk3-i' in Table 4. Regarding the increased computational efficiency of the three stage scheme, a similar study is performed for the present method with artificial speed of sound. Using the same 3 stage scheme as in Ref. [17] in combination with the implicit system of Eqs. (24) and (26), the CFL number can still be kept at CFL = 1000, as for the 5 stage scheme. This indicates that in contrast to the basic method, the present method essentially removes the CFL-restriction on the explicit Runge–Kutta scheme. Due to the identical CFL number of 3 stage and 5 stage scheme, the corresponding convergence rates are also almost identical, with the 3 stage scheme requiring even a little fewer multigrid cycles, see Figs. 11–13. This leads to a significant reduction in CPU time, as can be seen from the corresponding results denoted by 'rk3-i(c')' in Table 4. Note that the convergence of lift is also identical for 3 stage and 5 stage scheme. With respect to the conventional reference scheme 'rk5/3-s', computation times are reduced by factors of 3.5–4 with method 'rk3-i(c')'.

From these results, the question arises whether the possible increase in CFL number may be attributed to the introduction of the artificial speed of sound. However at transonic speeds, the difference between physical

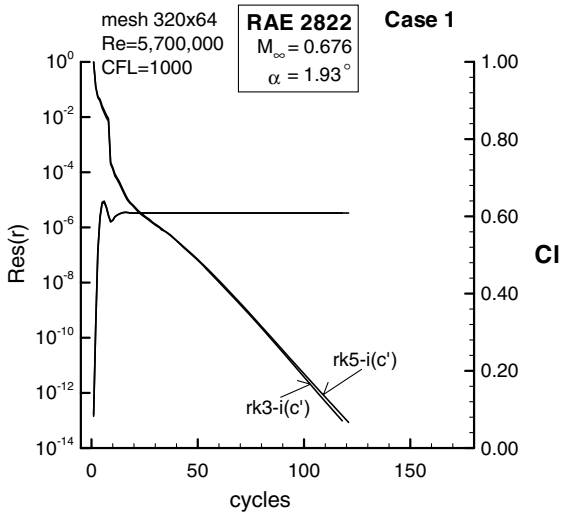

Fig. 11. Comparison of different Runge–Kutta schemes for Case 1 with 4-level multigrid (rk5-i(c'): 5 stage scheme, CFL = 1000; rk3-i(c'): 3 stage scheme, CFL = 1000).

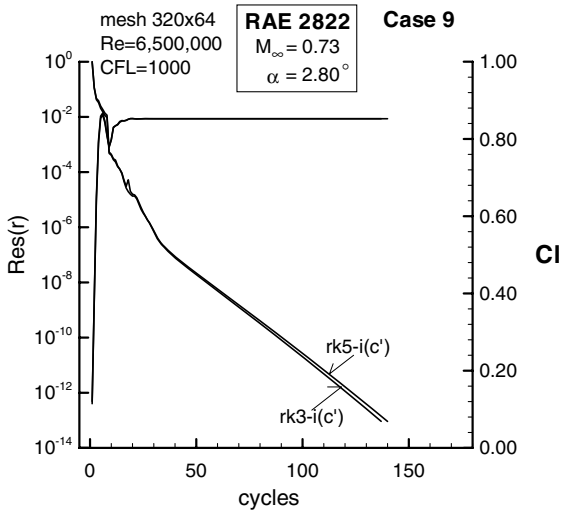

Fig. 12. Comparison of different Runge–Kutta schemes for Case 9 with 4-level multigrid (rk5-i(c'): 5 stage scheme, CFL = 1000; rk3-i(c'): 3 stage scheme, CFL = 1000).

speed of sound and artificial speed of sound becomes smaller with increasing Mach number, and for $M > 1$, both speeds are identical. Thus, another reason is suspected, which is related to the initialization of the approximate solution of Eq. (24) by SGS-sweeps. Fig. 14 shows convergence histories for computation of Case 9 with the basic method using the original initialization, the basic method with the initialization of the present method, and with the present method. For the basic method of Ref. [17] with the initialization of the SGS-sweeps by explicit residuals, the CFL number was limited to CFL = 160. Changing this initialization to that of the present method by setting the unknowns to zero, the CFL number could be raised to CFL = 1000. The convergence behavior observed in Fig. 14 for Case 9 with the different methods is similar for analogous computations of Cases 1 and 10, thus those results are not presented.

In Figs. 11–13 it is demonstrated that for computation of compressible flows the convergence behavior of the present method for a 3 stage and 5 stage scheme is similar with CFL = 1000. To confirm this for a broader range of applications, the cases of Mach number variation for inviscid and viscous flow, and the cases for Rey-

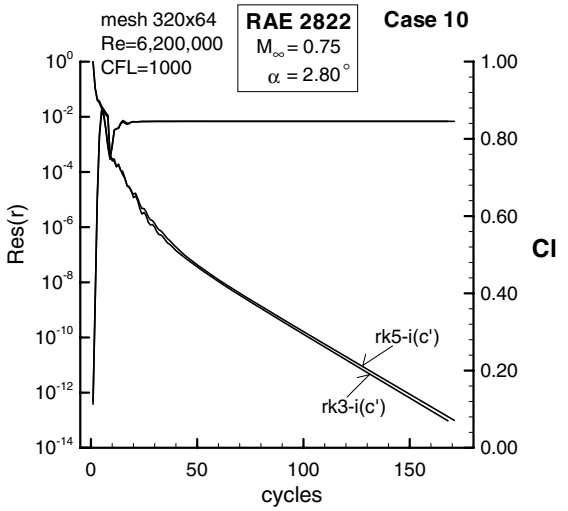

Fig. 13. Comparison of different Runge–Kutta schemes for Case 10 with 4-level multigrid (rk5-i(c'): 5 stage scheme, CFL = 1000; rk3-i(c'): 3 stage scheme, CFL = 1000).

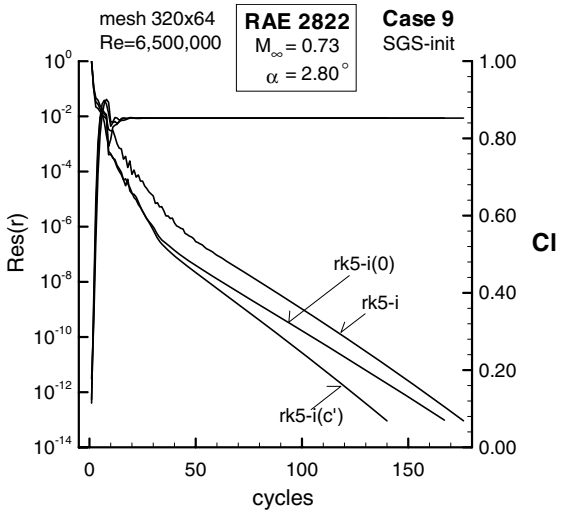

Fig. 14. Influence of SGS-initialization for computation of Case 9 with 4-level multigrid (rk5-i: 5 stage scheme, basic, SGS-initialization = explicit residuals, CFL = 160; rk5-i(0): 5 stage scheme, basic, SGS-initialization = 0, CFL = 1000; rk5-i(c'): 5 stage scheme, present, SGS-initialization = 0, CFL = 1000).

nolds number variation are repeated with the 3 stage scheme. The convergence histories obtained are again almost identical to those of the 5 stage scheme, resulting in the substantial gains in computational efficiency noted above.

5.3.3. Variation of number of symmetric Gauss–Seidel sweeps

To investigate further possibilities for increasing computational efficiency, the number of Symmetric Gauss–Seidel sweeps when solving Eq. (24) is varied. As a test case, the conditions of Case 9 are used computing the flow around the RAE2822 airfoil on the mesh with 320x64 cells, and the scheme 'rk3-i(c')' is used with a CFL number of CFL = 1000. The number of SGS-sweeps is varied between 1 and 8, and the corresponding results with respect to CPU-time, number of iterations, and convergence rate are listed in Table 5. As can be seen from Table 5, raising the number of SGS-sweeps above 3 only has a moderate effect on

Table 5
Variation of number of Symmetric Gauss–Seidel (SGS) sweeps using 3 stage Runge–Kutta/Implicit scheme for computation of Case 9

# SGS-iterations	CPU time [s]	# iterations	Convergence rate
1	565	180	0.846
2	635	147	0.815
3	842	137	0.803
4	884	132	0.796
6	1244	128	0.791
8	1522	126	0.788

Table 6
Efficiency of 3 stage Runge–Kutta/Implicit scheme with variation of SGS-sweeps for RAE2822 test cases

Case, # SGS-sweeps	CPU time [s]	# iterations	Convergence rate
Case 1, #SGS = 1	505	145	0.813
Case 1, #SGS = 3	702	117	0.776
Case 9, #SGS = 1	565	180	0.846
Case 9, #SGS = 3	842	137	0.803
Case 10, #SGS = 1	736	211	0.867
Case 10, #SGS = 3	997	168	0.836

improving convergence; however the CPU-time required increases significantly. In contrast to that, reducing the number of SGS-sweeps only moderately increased the number of iterations, however CPU-time decreased substantially, with best efficiency obtained when using only one SGS-sweep. In Table 6, a comparison is made for computation of Case 1, Case 9, and Case 10 using one and three SGS-sweeps. Using one SGS-sweep proves to be the most efficient strategy for all cases. With respect to the conventional, well tuned method 'rk5/3-s', the required CPU time is reduced by a factor of almost 5 for the subsonic and by more than 5 for the transonic cases, as can be seen by comparison with Table 4.

5.3.4. Influence of mesh refinement on convergence behavior

The implicit step of the Runge–Kutta/Implicit method employs Symmetric Gauss–Seidel iteration to approximately solve Eq. (24), and the question arises whether mesh density may influence convergence properties. To assess this effect, viscous flow is computed on three successively refined meshes, where the mesh with 320×64 cells used previously represents the medium mesh. Analogously to the Reynolds number variation, the subsonic conditions of Case 1 from Ref. [40] are used, and pure second order reconstruction of variables without limiting is employed. The Runge–Kutta/Implicit scheme is used with the parameters previously determined as most efficient, namely the 3 stage scheme with only one SGS sweep to solve Eq. (24) combined with artificial speed of sound in Eq. (26). Fig. 15 displays the corresponding convergence histories, and Table 7 summarizes the computational performance with respect to CPU-time, number of iterations, and convergence rate. As can be seen from Fig. 15 and Table 7, convergence properties are almost unaffected by mesh refinement, and the present Runge–Kutta/Implicit method yields an average convergence rate of about 0.825 on all meshes.

6. Concluding remarks

A computational approach was derived which addresses two sources of stiffness when solving fluid dynamic equations, namely discrete stiffness associated with high aspect ratio cells, and analytical stiffness occurring when the Mach number approaches zero. The Runge–Kutta/Implicit method, where an explicit Runge–Kutta time integration was combined with the solution of an implicit system of equations, had already demonstrated its efficiency for computations on highly stretched meshes in earlier work. To also address analytical stiffness arising from the disparity of eigenvalues when approaching the incompressible limit, an artificial speed of sound was introduced into the implicit system of discrete equations. This allowed the extension of the

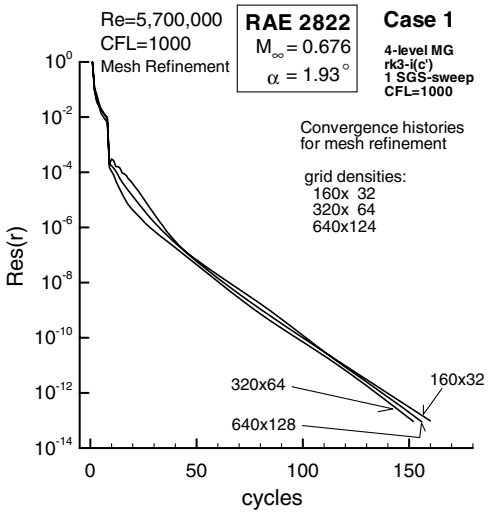

Fig. 15. Convergence histories for mesh refinement study of Case 1 (4-level multigrid, 3 stage scheme, artificial speed of sound, 1 SGS-iteration, CFL = 1000).

Table 7
Convergence behavior for mesh refinement study of Case 1

Grid	CPU time [s]	# iterations	Convergence rate
160×32	111	160	0.829
320×64	450	152	0.820
640×128	1996	156	0.825

Runge–Kutta/Implicit method to incompressible flows without compromising its efficiency. The artificial speed of sound is of the order of the main flow speed, where its definition followed the practice of low speed preconditioning when extending compressible codes toward incompressible flows. In contrast to common low speed preconditioning however, no additional matrix for preconditioning was required, since the artificial speed of sound was consistently introduced into both, the explicit residual on the right hand side and into the implicit terms on the left hand side of the discrete system of equations. Due to the formulation of the inviscid flux Jacobians in terms of Mach number, the implementation of the artificial speed of sound could be achieved with negligible additional effort, and the efficient, storage reducing evaluation of flux Jacobians in the implicit step was not compromised.

The present Runge–Kutta/Implicit method was applied for solving different cases of compressible and incompressible, inviscid and viscous turbulent airfoil flow. Favorable convergence rates were achieved ranging from 0.45 to 0.65 for inviscid flow and from 0.75 to 0.85 for viscous flow. Convergence was observed to be almost independent of Mach number, confirming the successful extension of the method to address analytical stiffness. For the investigation with respect to discrete stiffness, a Reynolds number variation by more than an order of magnitude was performed. Despite cell aspect ratios increasing from 3000 to 50,000, convergence was only affected by roughly a factor of 2, confirming the ability to address discrete stiffness. A reduction of free stream Mach number to 0.001 in the Reynolds number variation study demonstrated that this capability was not impaired for incompressible flows. The implicit step of the Runge–Kutta/Implicit method thus acts as a preconditioner to the explicit Runge–Kutta which proved to effectively address both, discrete stiffness caused by high aspect ratio cells, and stiffness in the analytical equations when the Mach number approaches zero.

It was further shown that appropriate initialization for solving the implicit system allowed a CFL limit of CFL = 1000 not only for a 5 stage scheme but also for a 3 stage scheme, resulting in a substantial reduction of computation time while maintaining similar convergence rates. Additional increase in computational efficiency was achieved by using just one SGS-sweep per Runge–Kutta stage, and compared to a well tuned conven-

tional method CPU times were reduced by half an order of magnitude. Finally it was demonstrated that convergence properties of the method were independent of mesh density.

Acknowledgement

The author likes to express his thanks to Charles Swanson, NASA LaRC, and Eli Turkel, Tel Aviv University, for many helpful discussions and e-mail conversations.

References

[1] N. Kroll, J.K. Fassbender (Eds.), MEGAFLOW—Numerical Flow Simulation for Aircraft Design, Notes on Numerical Fluid Mechanics and Multidisciplinary Design, vol. 89, Springer, 2005.
[2] J.B. Vos, A. Rizzi, D. Darraq, E.H. Hirschel, Navier–Stokes solvers in European aircraft industry, Prog. Aerosp. Sci. 38 (2002) 601–697.
[3] F. Le Chuiton, Chimera simulation of a complete helicopter with rotors as actuator disks, in: H.J. Rath, C. Holze, H.-J. Heinemann, R. Henke, H. Hoenlinger (Eds.), Contributions to 14th STAB/DGLR Symposium Bremen, 2004, Notes on Numerical Fluid Mechanics and Multidisciplinary Design, vol. 92, Springer Publishers, Berlin, Heidelberg, New York, 2006.
[4] Th. Eggers, Aerothermodynamics – Longitudinal Stability and Trim of an Ariane V Fly-Back Booster, AIAA-Paper 2003-7055, 2003.
[5] C. Farhat, M. Lesoinne, M. Maman, Mixed explicit/implicit time integration of coupled aeroelastic problems: three field formulation, geometric conservation and distributed solution, Int. J. Numer. Meth. Fluids 21 (1995) 807–835.
[6] R. Heinrich, R. Ahrem, G. Guenther, H.-P. Kersken, W.-R. Krueger, J. Neumann, Aeroelastic computation using the AMANDA simulation environment, in: Proc. CEAS Conf. Multidisciplinary Design and Optimization, Cologne, 2001, pp. 19–30.
[7] A. Schuette, G. Einarsson, B. Schoening, A. Raichle, W. Moennich, Th. Alrutz, Numerical simulation of maneuvring combat aircraft, in: H.J. Rath, C. Holze, H.-J. Heinemann, R. Henke, H. Hoenlinger (Eds.), Contributions to 14th STAB/DGLR Symposium Bremen, 2004, Notes on Numerical Fluid Mechanics and Multidisciplinary Design, vol. 92, Springer Publishers, Berlin, Heidelberg, New York, 2006.
[8] A. Brandt, Multi-level adaptive solutions to boundary-value problems, Math. Comp. 31 (138) (1977) 333–390.
[9] A. Jameson, Multigrid algorithms for compressible flow calculations, in: W. Hackbusch, U. Trottenberg (Eds.), Second European Conference on Multigrid Methods, Cologne, 1985, Lecture Notes in Mathematics, vol. 1228, Springer Verlag, Berlin, 1986.
[10] A. Jameson, D.A., Caughey, How many steps are required to solve the Euler equations of steady, compressible flow: in search of a fast solution algorithm, AIAA-Paper 2001-2673, 2001.
[11] N.A. Pierce, M.B. Giles, Preconditioned multigrid methods for compressible flow calculations on stretched meshes, Comput. Phys. 136 (1997) 425–445.
[12] L. Martinelli, Calculations of Viscous Flow with a Multigrid Method, Ph.D. Dissertation, Princeton University, 1987.
[13] S.R. Allmaras, Analysis of a local matrix preconditioner for the 2-D Navier–Stokes Equations, AIAA-Paper 93-3330, 1993.
[14] W.L. Kleb, W.A. Wood, B. van Leer, Efficient multi-stage time marching for viscous flows via local preconditioning, AIAA-Paper 99-3267, 1999.
[15] D.A. Caughey, A. Jameson, Fast preconditioned multigrid solution of the Euler and Navier–Stokes equations for steady, compressible flows, AIAA-Paper 2002-0963, 2002.
[16] R. Radespiel, C.-C. Rossow, R.C. Swanson, An efficient cell-vertex multigrid scheme for the three-dimensional Navier–Stokes equations, AIAA J. 28 (1998) 423–459.
[17] C.-C. Rossow, Convergence acceleration for solving the compressible Navier–Stokes equations, AIAA J. 44 (2006) 345–352.
[18] C.-C. Rossow, Toward efficient solution of the compressible Navier–Stokes equations, in: H.J. Rath, C. Holze, H.-J. Heinemann, R. Henke, H. Hoenlinger (Eds.), Contributions to 14th STAB/DGLR Symposium Bremen, 2004, Notes on Numerical Fluid Mechanics and Multidisciplinary Design, vol. 92, Springer Publishers, Berlin, Heidelberg, New York, 2006.
[19] E. Turkel, Preconditioning techniques in computational fluid dynamics, Annu. Rev. Fluid Mech. 31 (1999) 385–416.
[20] E. Turkel, Preconditioned methods for solving the incompressible and low speed compressible equations, J. Comp. Phys. 72 (1987) 277–298.
[21] Y.-H. Choi, C.L. Merkle, The application of preconditioning to viscous flows, J. Comp. Phys. 105 (1993) 207–223.
[22] J.M. Weiss, W.A. Smith, Preconditioning applied to variable and constant density flows, AIAA J. 33 (1995) 2050–2057.
[23] D.L. Darmofal, B. van Leer, Local preconditioning: manipulating mother nature to fool father time, in: D. Caughey, M. Hafez (Eds.), Computing the Future 2: Computational Fluid Dynamics and Transonic Flow, 1998.
[24] R. Rudnik, S. Melber, A. Ronzheimer, O. Brodersen, Three-dimensional Navier–Stokes simulations for transport aircraft high lift configurations, AIAA J. 38 (2001) 895–903.
[25] K.C. Karki, S.V. Patankar, Pressure based calculation procedure for viscous flows at all speed in arbitrary configurations, AIAA J. 27 (1989) 1167–1174.
[26] I. Demirdzic, Z. Lilek, M. Peric, A collocated finite volume method for predicting flows at all speeds, Int. J. Numer. Meth. Fluids 16 (1993) 1029–1050.
[27] H. Bijl, P. Wesseling, A unified method for computing incompressible and compressible flows in boundary fitted coordinates, J. Comp. Phys. 141 (1998) 153–177.

[28] C.-C. Rossow, A blended pressure/density based method for the computation of incompressible and compressible flows, J. Comp. Phys. 185 (2003) 375–398.
[29] C.-C. Rossow, Extension of a compressible code toward the incompressible limit, AIAA J. 41 (2003) 2379–2386.
[30] C.-C. Rossow, A flux splitting scheme for compressible and incompressible flows, J. Comp. Phys. 164 (2000) 104–122.
[31] P.L. Roe, Approximate Riemann solvers, parameter vectors and difference schemes, J. Comp. Phys. 43 (1981) 357–372.
[32] A. Jameson, Artificial diffusion, upwind biasing, limiters and their effect on accuracy and multigrid performance in transonic and hypersonic flow, AIAA-Paper 93-3359, 1993.
[33] R.C. Swanson, R. Radespiel, E. Turkel, On some numerical dissipation schemes, J. Comp. Phys. 147 (1998) 518–544.
[34] J. Blazek, N. Kroll, R. Radespiel, C.-C. Rossow, Upwind implicit residual smoothing method for multi-stage schemes, AIAA-Paper 91-1533, 1991.
[35] J. Blazek, N. Kroll, C.-C. Rossow, R.C. Swanson, A comparison of several implicit smoothing methods in combination with multigrid, in: Int. Conf. on Numerical Methods in Fluid Dynamics, Rome, Italy, 1992.
[36] J. Blazek, Computational Fluid Dynamics: Principles and Applications, second ed., Elsevier Publishers, 2005, ISBN 0-080-44506-3.
[37] A. Jameson, E. Turkel, Implicit schemes and LU decompositions, Math. Comp. 37 (156) (1981) 385–397.
[38] A. Jameson, S. Yoon, Multigrid solutions of the Euler equations using implicit schemes, AIAA J. 24 (1986) 1737–1743.
[39] L. Hong, J.D. Baum, R. Löhner, A fast, matrix-free implicit method for compressible flows on unstructured grids, J. Comp. Phys. 146 (1998) 664–690.
[40] S. Abarbanel, D. Gottlieb, Optimal splitting for two and three dimensional Navier–Stokes equations with mixed derivatives, ICASE Report No. 80-6, February 1980.
[41] B. Baldwin, H. Lomax, Thin layer approximation and algebraic turbulence model for separated turbulent flows, AIAA-Paper 78-257, 1978.
[42] E. Turkel, Preconditioning and the limit of the compressible to the incompressible flow equations for finite difference schemes, in: D. Caughey, M. Hafez (Eds.), Frontiers of Computational Fluid Dynamics, John Wiley & Sons, 1994, pp. 215–234.
[43] H. Guillard, C. Viozat, On the behaviour of upwind schemes in the low Mach number limit, Comp. Fluids 28 (1999) 63–86.
[44] B. Van Leer, C.H. Tai, K.G. Powell, Design of optimally smoothing multi-stage schemes for the Euler equations, AIAA-Paper 89-1933, 1989.
[45] J. Feng, C.L. Merkle, Evaluation of preconditioning methods for time-marching systems, AIAA-Paper 90-0016, 1990.
[46] P.H. Cook, M.A. McDonald, M.C.P. Firmin, Aerofoil RAE2822 pressure distributions and boundary layer and wake measurements, AGARD-AR-138, 1979.
[47] J. Faßbender, Improved robustness of numerical simulation of turbulent flows around civil transport aircraft at flight Reynolds numbers, Ph.D. dissertation, Technical University of Braunschweig, 2004. Available from: <http://opus.tu-bs.de/opus/volltexte/2004/579>.
[48] R. Rudnik, Untersuchung der Leistungsfähigkeit von Zweigleichungs-Turbulenzmodellen bei Profilumströmungen, Ph.D. dissertation, Technical University of Berlin, 1997.