Available online at www.sciencedirect.com
ScienceDirect
Journal of Computational Physics 219 (2006) 622–643
JOURNAL OF COMPUTATIONAL PHYSICS
www.elsevier.com/locate/jcp

Pseudo-time stepping methods for space–time discontinuous Galerkin discretizations of the compressible Navier–Stokes equations

C.M. Klaij [a,*], J.J.W. van der Vegt [a], H. van der Ven [b]

[a] University of Twente, Department of Applied Mathematics, P.O. Box 217, 7500 AE Enschede, The Netherlands
[b] National Aerospace Laboratory NLR, P.O. Box 90502, 1006 BM Amsterdam, The Netherlands

Received 25 November 2005; received in revised form 31 March 2006; accepted 4 April 2006
Available online 24 May 2006

Abstract

The space–time discontinuous Galerkin discretization of the compressible Navier–Stokes equations results in a non-linear system of algebraic equations, which we solve with pseudo-time stepping methods. We show that explicit Runge–Kutta methods developed for the Euler equations suffer from a severe stability constraint linked to the viscous part of the equations and propose an alternative to relieve this constraint while preserving locality. To evaluate its effectiveness, we compare with an implicit–explicit Runge–Kutta method which does not suffer from the viscous stability constraint. We analyze the stability of the methods and illustrate their performance by computing the flow around a 2D airfoil and a 3D delta wing at low and moderate Reynolds numbers.

PACS: 02.60.Cb; 02.70.Dh; 03.40.Gc

Keywords: Compressible Navier–Stokes equations; Discontinuous Galerkin finite element methods; Pseudo-time integration; Implicit–explicit Runge–Kutta methods

1. Introduction

Discontinuous Galerkin (DG) methods are nowadays applied to a wide range of problems because of their accuracy and favorable properties related to the locality of the discretization, such as flexibility in mesh adaptation and efficient parallelization. In fact, the discretization in each element only involves its direct neighbors, even for higher order of accuracy, making the method most local. The usual approach is to apply discontinuous basis-functions in space and a Runge–Kutta method for the time integration, resulting in the so-called RKDG method, see for example the survey by Cockburn and Shu [10]. Thanks to the work by Bassi and

* Corresponding author.
E-mail addresses: c.m.klaij@math.utwente.nl (C.M. Klaij), j.j.w.vandervegt@math.utwente.nl (J.J.W. van der Vegt), venvd@nlr.nl (H. van der Ven).

0021-9991/$ - see front matter © 2006 Elsevier Inc. All rights reserved.
doi:10.1016/j.jcp.2006.04.003

Rebay [3], Baumann and Oden [6] and Cockburn and Shu [9], DG methods were successfully extended from hyperbolic to (incompletely) parabolic equations, see [1,7] for the detailed analysis of purely elliptic problems and [2,5,11] for applications to the compressible Navier–Stokes equations.

In [14], we presented a space–time discontinuous Galerkin method for the compressible Navier–Stokes equations, which is an extension of the space–time DG method for the Euler equations [23,24] designed for flow problems on moving and deforming meshes. The main idea is to use discontinuous basis-functions both in space *and* time, and to introduce a numerical time-flux to ensure causality in time. The viscous flux is treated by extending the approach presented in [3,5] to the space–time context. The method is fully implicit in *physical time* and results in a system of non-linear algebraic equations [14]. This paper focuses on solving the algebraic system.

The algebraic system may be solved with a fully implicit method, such as the implicit Euler method [4] with GMRES to solve the nonlinear system or the Newton method [13] for the steady-state equations. The advantage of implicit methods is that the number of iterations needed to solve the system is significantly reduced in comparison with explicit methods. Their main disadvantage, however, is the increased cost per iteration, linked to the global linear system (based on the Jacobian of the flux) which has to be assembled and solved. The storage of this (sparse) matrix can become problematic in 3D 'real-life' cases which typically require millions of elements. Another issue is the small basin of attraction which demands an accurate initial guess in order to converge. In practice, the initial guess is based on the free stream conditions, and special attention is needed to overcome this problem, either by initially using small physical time-steps [4] before increasing the step-size or by grid sequencing [13].

For the space–time discretization of the Euler equations [23], the algebraic system was solved with an explicit *pseudo-time* stepping Runge–Kutta method (with the correction by Melson et al. [18]). When applied as a smoother in a full approximation multigrid scheme, this approach proved very efficient. The main advantage of the pseudo-time stepping method is its locality, which matches the locality of the discontinuous Galerkin method and dispenses with the assembly and storage of a global matrix. Also, the pseudo-time stepping method is insensitive to the initial condition and remains stable for large physical time steps. Therefore, we will aim at extending the pseudo-time stepping approach in [23] to the space–time DG discretization of the Navier–Stokes equations presented in [14].

Extending the pseudo-time stepping method to the space–time DG discretization of the compressible Navier–Stokes equations is not trivial for two reasons. First, for the Euler equations, the pseudo-time Courant–Friedrichs–Levy (CFL) condition must be satisfied for stability of the Runge–Kutta method [23]. Applying the same method to the space–time discretization of the compressible Navier–Stokes equations requires an additional stability constraint, the Von Neumann condition, which is more restrictive than the CFL condition in flow regions with small *cell* Reynolds numbers, i.e. boundary layers. Therefore, the Runge–Kutta method will converge much slower. Second, in [23], the Runge–Kutta method is used as a smoother in a full approximation multigrid algorithm. Since the efficiency of the multigrid algorithm greatly depends on the quality of the smoother, we will limit ourselves in this paper to finding an effective Runge–Kutta method for viscous flows, to be combined with multigrid in our future work [15].

To relieve the stability constraint without losing locality, we turn to the family of explicit Runge–Kutta methods derived by Kleb et al. [16]. These schemes are specially designed for viscous flows and have stability domains which are much more stretched along the negative real axis than the Runge–Kutta schemes used for hyperbolic equations. Even though the Von Neumann condition still has to be satisfied for stability, it may no longer be prohibitively restrictive. Since in aerodynamic applications, we encounter both inviscid flow (far-field) and viscous flow (boundary layers) in the same simulation, we apply *local* pseudo-time stepping and combine the scheme developed for the Euler equations in [23], which is optimal in the inviscid regime, with the scheme presented in [16] for the viscous regime. This is feasible because both methods are local and permissible because accuracy is not an issue in pseudo-time.

We investigate the effectiveness of this combination in relieving the viscous stability constraint by comparing with an implicit–explicit Runge–Kutta method. In this method, the inviscid part is treated explicitly and the viscous part implicitly, such that only the pseudo-time CFL condition needs to be satisfied for stability. In that sense, it represents the ideal situation where viscosity does not affect stability, against which we can test the influence of the Von Neumann condition on fully explicit methods. Since the implicit–explicit method

shares the disadvantage of fully implicit methods (i.e. a global linear system) without sharing the advantage (i.e. a small the number of iterations) it is only recommended for this type of testing.

The outline of this paper is the following. In Section 2, we briefly summarize the space–time discontinuous Galerkin discretization and give the weak formulation, the basis-functions and the resulting system of non-linear algebraic equations. The different pseudo-time stepping methods are described in Section 3 and their stability is analyzed using the scalar advection–diffusion equation as a model problem in Section 4. In Section 5, we compare the performance of the methods by computing steady and unsteady viscous flow around the 2D NACA0012 airfoil and a 3D delta wing before drawing conclusions in Section 6.

2. Summary of the space–time DG discretization

This section summarizes the space–time discontinuous Galerkin discretization of the compressible Navier–Stokes equations as presented in [14], to which we refer for further details. The main idea is to consider the equations directly in the space–time domain using discontinuous basis-functions in space–time and introduce a numerical time-flux to ensure causality in time. The treatment of the viscous terms in [3,5] was extended to the space–time context.

2.1. Space–time formulation

The space–time discontinuous Galerkin finite element method does not distinguish between space and time variables: the equations are considered in an open domain $\mathscr{E} \subset \mathbb{R}^4$, where a point with position $\bar{x} = (x_1, x_2, x_3)$ at time $t = x_0$ has Cartesian coordinates (x_0, x_1, x_2, x_3). The flow domain $\Omega(t)$ at time t is defined as $\Omega(t) := \{\bar{x} \in \mathbb{R}^3 : (t, \bar{x}) \in \mathscr{E}\}$. Let t_0 and T be the initial and final time of the evolution of the flow domain, then the space–time domain boundary $\partial\mathscr{E}$ consists of the hypersurfaces $\Omega(t_0) := \{x \in \partial\mathscr{E} : x_0 = t_0\}$, $\Omega(T) := \{x \in \partial\mathscr{E} : x_0 = T\}$, and $\mathscr{Q} := \{x \in \partial\mathscr{E} : t_0 < x_0 < T\}$. Using this notation, the compressible Navier–Stokes equations can be written as:

$$\begin{cases} U_{i,0} + F^{\mathrm{e}}_{ik}(U)_{,k} - (A_{ikrs}(U)U_{r,s})_{,k} = 0 & \text{on } \mathscr{E}, \\ U = U_0 & \text{on } \Omega(t_0), \\ U = \mathscr{B}(U, U^b) & \text{on } \mathscr{Q}, \end{cases}$$

with $U \in \mathbb{R}^5$ the vector of conservative variables, $F^{\mathrm{e}} \in \mathbb{R}^{5\times 3}$ the inviscid flux, $A \in \mathbb{R}^{5\times 3\times 5\times 3}$ the homogeneity tensor, $U_0 \in \mathbb{R}^5$ the initial flow field and $\mathscr{B} \in \mathbb{R}^5$ the boundary operator. The summation convention on repeated indices is used with $i, r = 1, \ldots, 5$ and $k, s = 1, 2, 3$. The conservative variables, the inviscid flux and the viscous flux $F^{\mathrm{v}} \in \mathbb{R}^{5\times 3}$ are defined as:

$$U = \begin{bmatrix} \rho \\ \rho u_j \\ \rho E \end{bmatrix}, \quad F^{\mathrm{e}}_k = \begin{bmatrix} \rho u_k \\ \rho u_j u_k + p\delta_{jk} \\ u_k(\rho E + p) \end{bmatrix}, \quad F^{\mathrm{v}}_k = \begin{bmatrix} 0 \\ \tau_{jk} \\ \tau_{kj} u_j - q_k \end{bmatrix}$$

with ρ the density, $\rho\vec{u}$ the momentum density vector, ρE the total energy density, p the pressure, δ the Kronecker delta function, τ the shear stresses and q the heat flux. The viscous flux F^{v} is homogeneous with respect to the gradient of the conservative variables ∇U. This defines the homogeneity tensor A as:

$$A_{ikrs}(U) = \frac{\partial F^{\mathrm{v}}_{ik}(U, \nabla U)}{\partial(U_{r,s})}.$$

This property is essential for the treatment of the viscous terms in the space–time formulation of the compressible Navier–Stokes equations [14].

2.2. Discretization

The space–time discretization starts with the tessellation $\mathscr{T}^n_h = \{\mathscr{K}\}$ in a space–time slab, which is the flow domain $\mathscr{E}$ in the time interval (t_n, t_{n+1}). The associated function spaces are defined as:

$$W_h := \{W \in (L^2(\mathscr{E}_h))^5 : W|_{\mathscr{K}} \circ G_{\mathscr{K}} \in (P^m(\hat{\mathscr{K}}))^5 \ \forall \mathscr{K} \in \mathscr{T}_h\},$$
$$V_h := \{V \in (L^2(\mathscr{E}_h))^{5\times 3} : V|_{\mathscr{K}} \circ G_{\mathscr{K}} \in (P^m(\hat{\mathscr{K}}))^{5\times 3} \ \forall \mathscr{K} \in \mathscr{T}_h\},$$

where $G_{\mathscr{K}}$ denotes the mapping of the master element $\hat{\mathscr{K}} = (-1,1)^4$ to element $\mathscr{K}$ and $P^m(\hat{\mathscr{K}})$ denotes the space of polynomials of degree at most m. Notice that $\nabla_h W_h \subset V_h$ where ∇_h is the broken gradient: $(\nabla_h W_h)|_{\mathscr{K}} = \nabla(W_h|_{\mathscr{K}})$. The set of internal faces in a space–time slab is denoted by $\mathscr{S}_I^n$ and the set of boundary faces by $\mathscr{S}_B^n$. The traces from the left and right are denoted by $(\cdot)^{\mathrm{L}}$ and $(\cdot)^{\mathrm{R}}$, respectively. The average operator is defined as $\{\{\cdot\}\} = 1/2((\cdot)^{\mathrm{L}} + (\cdot)^{\mathrm{R}})$ and the jump operator as $[\![\cdot]\!]_k = (\cdot)^{\mathrm{L}} n_k^{\mathrm{L}} + (\cdot)^{\mathrm{R}} n_k^{\mathrm{R}}$, with n the outward normal vector of the element under consideration. Using this notation, the weak formulation of the compressible Navier–Stokes equations can be written as follows.

Find a $U \in W_h$, such that for all $W \in W_h$:

$$\begin{aligned}
&-\sum_{\mathscr{K}\in\mathscr{T}_h^n}\int_{\mathscr{K}}(W_{i,0}U_i + W_{i,k}(F_{ik}^{\mathrm{e}} - A_{ikrs}U_{r,s} + \mathscr{R}_{ik}))\,\mathrm{d}\mathscr{K} + \sum_{K\in\mathscr{T}_h^n}\left(\int_{K(t_{n+1}^-)} W_i^{\mathrm{L}}U_i^{\mathrm{L}}\,\mathrm{d}K - \int_{K(t_n^+)} W_i^{\mathrm{L}}U_i^{\mathrm{R}}\,\mathrm{d}K\right)\\
&+\sum_{\mathscr{S}\in\mathscr{S}_I^n}\int_{\mathscr{S}}(W_i^{\mathrm{L}} - W_i^{\mathrm{R}})H_i\,\mathrm{d}\mathscr{S} + \sum_{\mathscr{S}\in\mathscr{S}_B^n}\int_{\mathscr{S}} W_i^{\mathrm{L}}H_i^b\,\mathrm{d}\mathscr{S} - \sum_{\mathscr{S}\in\mathscr{S}_I^n}\int_{\mathscr{S}}[\![W_i]\!]_k\{\{A_{ikrs}U_{r,s} - \eta\mathscr{R}_{ik}^{\mathscr{S}}\}\}\,\mathrm{d}\mathscr{S}\\
&-\sum_{\mathscr{S}\in\mathscr{S}_B^n}\int_{\mathscr{S}} W_i^{\mathrm{L}}(A_{ikrs}^b U_{r,s}^b - \eta\mathscr{R}_{ik}^{\mathscr{S}})\bar{n}_k^{\mathrm{L}}\,\mathrm{d}\mathscr{S} = 0.
\end{aligned}$$

Here, $H \in \mathbb{R}^5$ is the inviscid numerical flux from the HLLC approximate Riemann solver with the extension needed for moving meshes (cf. [23]) and $(\cdot)^{\mathrm{b}}$ indicates dependence on the prescribed boundary data. The stability constant is $\eta > N_{\mathrm{f}}$, with N_{f} the number of faces per element. The local lifting operator is denoted by $\mathscr{R}^{\mathscr{S}} \in \mathbb{R}^{5\times 3}$ and defined [14] as:

Find an $\mathscr{R}^{\mathscr{S}} \in V_h$, such that for all $V \in V_h$:

$$\sum_{\mathscr{K}\in\mathscr{T}_h^n}\int_{\mathscr{K}} V_{ik}\mathscr{R}_{ik}^{\mathscr{S}}\,\mathrm{d}\mathscr{K} = \begin{cases} \int_{\mathscr{S}}\{\{V_{ik}A_{ikrs}\}\}[\![U_r]\!]_s\,\mathrm{d}\mathscr{S} & \text{for } \mathscr{S}\in\mathscr{S}_I^n,\\ \int_{\mathscr{S}} V_{ik}^{\mathrm{L}}A_{ikrs}^{\mathrm{L}}(U_r^{\mathrm{L}} - U_r^b)\bar{n}_s\,\mathrm{d}\mathscr{S} & \text{for } \mathscr{S}\in\mathscr{S}_B^n,\end{cases}$$

The global lifting operator $\mathscr{R} \in \mathbb{R}^{5\times 3}$ is obtained from the local lifting operator $\mathscr{R}^{\mathscr{S}}$ using the relation:

$$\mathscr{R} = \sum_{\mathscr{S}\in\mathscr{S}_I^n\cup\mathscr{S}_B^n}\mathscr{R}^{\mathscr{S}}.$$

The upwind character of the numerical time-flux in the integrals over the time faces $K(t_n^+)$ and $K(t_{n+1}^-)$ ensures causality in time. The trial function U and the test function W in each element $\mathscr{K} \in \mathscr{T}_h^n$ are represented as polynomials:

$$U(t,\bar{x})|_{\mathscr{K}} = \hat{U}_j\psi_j(t,\bar{x}), \quad \text{and} \quad W(t,\bar{x})|_{\mathscr{K}} = \hat{W}_l\psi_l(t,\bar{x}),$$

with $\hat{(\cdot)}$ the expansion coefficients and ψ the basis functions described in [14].

The system of algebraic equations for the expansion coefficients of U is obtained by replacing U and W in the weak formulation with their polynomial expansions and using the fact that the test functions W are arbitrary. For each physical time step the system can be written as:

$$\mathscr{L}(\hat{U}^n; \hat{U}^{n-1}) = 0. \tag{1}$$

This paper focuses on solving system (1) using pseudo-time stepping methods. We add a pseudo-time derivative:

$$\frac{\partial\hat{U}}{\partial\tau} = -\frac{1}{\Delta t}\mathscr{L}(\hat{U}; \hat{U}^{n-1}), \tag{2}$$

with $\Delta t = t_{n+1} - t_n$ and iterate in pseudo-time τ to steady-state using Runge–Kutta methods. At steady-state we have $\hat{U}^n = \hat{U}$.

In this paper we will investigate a combination of two explicit Runge–Kutta schemes, one designed for inviscid flows and the other for viscous flows. To evaluate the effect of viscosity on the stability and perfor-

mance of this method, we will compare with an implicit–explicit Runge–Kutta method, where the viscous terms are treated implicitly and the inviscid terms explicitly. In our future work the explicit Runge–Kutta method will be used as a smoother in a full approximation multigrid scheme to enhance the overall efficiency of the method.

3. Pseudo-time stepping methods

In this section, the different Runge–Kutta methods for the pseudo-time integration of system (2) are described.

First, we consider the explicit 5-stage Runge–Kutta method, which was successfully used to solve the system arising from the space–time discretization of the Euler equations in [23]. It is derived from a 5-stage Runge–Kutta method using the correction proposed by Melson et al. [18] to enhance the stability of the pseudo-time integration. For details of the derivation and the stability analysis for the Euler case we refer to [23]. This scheme is given by:

Algorithm 1 (*EXI*). Explicit Runge–Kutta method for inviscid flow with Melson correction.

(1) Initialize $\hat{V}^0 = \hat{U}$.
(2) For all stages $s = 1$ to 5 compute $\hat{V}^s$ as:

$$(I + \alpha_s \lambda I)\hat{V}^s = \hat{V}^0 + \alpha_s \lambda(\hat{V}^{s-1} - \mathscr{L}(\hat{V}^{s-1}; \hat{U}^{n-1})).$$

(3) Return $\hat{U} = \hat{V}^5$.

The Runge–Kutta coefficients at stage s are denoted by α_s and defined as: $\alpha_1 = 0.0791451$, $\alpha_2 = 0.163551$, $\alpha_3 = 0.283663$, $\alpha_4 = 0.5$ and $\alpha_5 = 1.0$. The matrix I represents the identity matrix. The coefficients were optimized to ensure rapid convergence to steady state. The factor λ is the ratio between the pseudo-time step $\Delta\tau$ and the physical time step: $\lambda = \Delta\tau/\Delta t$. The Melson correction consists in treating $\hat{V}$ semi-implicitly, without this the scheme would become unstable for values of λ around one.

Second, we consider the implicit–explicit version of the EXI method. The residual $\mathscr{L}$ defined in (1) consist of two parts: $\mathscr{L} = \mathscr{L}^{\mathrm{e}} + \mathscr{L}^{\mathrm{v}}$, where $\mathscr{L}^{\mathrm{e}}$ stems from the inviscid part of the compressible Navier–Stokes equations and $\mathscr{L}^{\mathrm{v}}$ from the viscous part. The implicit–explicit method can be derived by introducing a Newton matrix $\mathscr{D}$, which approximates the Jacobian of the viscous part of the residual:

$$\mathscr{D}\hat{V}^s \cong \mathscr{L}^{\mathrm{v}}.$$

Here, the approximation consists of freezing the (non-linear) homogeneity tensor A at the previous Runge–Kutta stage $s - 1$. This approximation is relatively inexpensive compared with the Jacobian of the inviscid flux which would be required by a Newton solver, since A is readily available in the discretization. The implicit–explicit Runge–Kutta method can thus be written as:

Algorithm 2 (*IMEX*). Implicit–explicit Runge–Kutta method.

(1) Initialize $\hat{V}^0 = \hat{U}$.
(2) For all stages $s = 1$ to 5 compute $\hat{V}^s$ by solving:

$$(I + \alpha_s \lambda(I + \mathscr{D}))\hat{V}^s = \hat{V}^0 + \alpha_s \lambda((I + \mathscr{D})\hat{V}^{s-1} - \mathscr{L}(\hat{V}^{s-1}; \hat{U}^{n-1})).$$

(3) Return $\hat{U} = \hat{V}^5$.

The coefficients α_s are the same as in Algorithm 1. Note that the diffusive terms $\mathscr{L}^{\mathrm{v}}$ in the residual $\mathscr{L}$ are *not* replaced by the approximation, both methods solve the same non-linear system $\mathscr{L} = 0$ at steady-state. Clearly, the l.h.s. of the equation for $\hat{V}^s$ is no longer a diagonal matrix, but a global sparse block matrix, therefore $\hat{V}^s$ must be computed by solving the sparse linear system. We do so using the sparse iterative GMRES solver with Jacobi preconditioning, available in the PETSc package [20].

Finally, we consider the family of methods proposed by Kleb et al. [16]. Based on an explicit 4-stage Runge–Kutta method, Kleb et al. proposed a procedure to optimize the Runge–Kutta coefficients for various situations depending on the cell Reynolds number. The method is given by:

Algorithm 3 (*EXV*). Explicit Runge–Kutta method for viscous flows.

(1) Initialize $\hat{V}^0 = \hat{U}$.
(2) For all stages $s = 1$ to 4 compute $\hat{V}^s$ as:
$$\hat{V}^s = \hat{V}^0 - \alpha_s \lambda \mathscr{L}(\hat{V}^{s-1}; \hat{U}^{n-1}).$$
(3) Return $\hat{U} = \hat{V}^4$.

For viscous flows, the Runge–Kutta coefficients at stage s are defined as: $\alpha_1 = 0.0178571$, $\alpha_2 = 0.0568106$, $\alpha_3 = 0.174513$ and $\alpha_4 = 1$. A summary of the derivation of these values is given in Appendix A. With these coefficients, the stability domain of the Runge–Kutta method is very different from the one associated with the classic 4-stage Runge–Kutta method for inviscid flows. Notice that we do not apply the Melson correction to this scheme because we will not use it for values of λ around one, for reasons which will become clear in the next section.

The EXI method is designed for inviscid flows, while the EXV method is designed for viscous flows. In aerodynamical applications, however, one encounters both flow regimes in the same simulation: the flow is inviscid in the far-field *and* viscous in boundary layers. Therefore, we will seek to combine both methods, based on their stability properties. The advantage of such a combination is that it remains local, contrary to implicit–explicit methods or fully implicit methods.

4. Stability analysis

The methods discussed in the previous section can all be applied to solve the system of non-linear equation (2) given by the space–time discretization of the compressible Navier–Stokes equations, provided a suitable pseudo-time stability constraint is satisfied. In this section, we derive these constraints.

4.1. The model problem

Rigorous stability analysis of numerical methods for the Navier–Stokes equations is extremely difficult and rarely attempted. Instead, in order to derive practical stability constraints, the method is required to be stable for the scalar advection–diffusion equation [16,25]:

$$u_t + a u_x = d u_{xx}, \quad (x,t) \in \mathbb{R} \times \mathbb{R}^+,$$

with $a > 0$ the advection constant and $d > 0$ the diffusion constant. The domain is divided into uniform rectangular elements Δt by Δx. The space–time discontinuous Galerkin method using the linear basis functions described in [14] gives the following discrete system for the vector of expansion coefficients $\hat{u}^n$ at time level n:

$$\mathscr{L}(\hat{u}^n; \hat{u}^{n-1}) \equiv (\mathscr{L}^{\mathrm{a}} + \mathscr{L}^{\mathrm{d}})\hat{u}^n + \mathscr{L}^t \hat{u}^{n-1} = 0. \tag{3}$$

The (block tridiagonal) inviscid part of the stencil depends on the Courant number:

$$\sigma = \frac{a\Delta t}{\Delta x}, \tag{4}$$

and is given by:

$$\mathscr{L}^{\mathrm{a}} = \left[\begin{array}{ccc|ccc|ccc} -\sigma & -\sigma & \sigma & 1+\sigma & \sigma & -\sigma & 0 & 0 & 0 \\ \sigma & \sigma & -\sigma & -\sigma & \frac{1}{3}+\sigma & \sigma & 0 & 0 & 0 \\ \sigma & \sigma & -\frac{4}{3}\sigma & -2-\sigma & -\sigma & 2+\frac{4}{3}\sigma & 0 & 0 & 0 \end{array}\right].$$

The right block is zero because the advective numerical flux is upwind ($a > 0$). The (block tridiagonal) viscous part of the stencil depends on the Von Neumann number:

$$\delta = \frac{d\Delta t}{(\Delta x)^2}, \tag{5}$$

as well as on the stabilization constant η and is given by:

$$\mathscr{L}^{\mathrm{d}} = \delta \left[\begin{array}{ccc|ccc|ccc} -2\eta & 1-2\eta & 2\eta & 4\eta & 0 & -4\eta & -2\eta & -1+2\eta & 2\eta \\ -1+2\eta & -2+2\eta & 1-2\eta & 0 & 4\eta & 0 & 1-2\eta & -2+2\eta & -1+2\eta \\ 2\eta & -1+2\eta & -\frac{13}{6}\eta & -4\eta & 0 & \frac{13}{3}\eta & 2\eta & 1-2\eta & -\frac{13}{6}\eta \end{array}\right].$$

The (block diagonal) part of the stencil related to the previous space–time slab is given by:

$$\mathscr{L}^{t} = \begin{bmatrix} -1 & 0 & 0 \\ 0 & -\frac{1}{3} & 0 \\ 2 & 0 & 0 \end{bmatrix}.$$

Note that the matrix has a periodic block Toeplitz structure with 3×3 blocks, written symbolically as:

$$\mathscr{L} = [L|D|U],$$

with L the block-lower, D the block-diagonal and U the block-upper part of the matrix. The system of algebraic equations (3) resulting from the space–time discontinuous Galerkin discretization of the model problem is solved using the pseudo-time stepping methods described in the previous section. Since the stability in pseudo-time of the Runge–Kutta methods is only affected by the transients, we only consider the homogeneous part of the linear system (3). Thus, the pseudo-time equation for the model problem becomes:

$$\frac{\partial \hat{u}^n}{\partial \tau} = -\frac{1}{\Delta t}(\mathscr{L}^{\mathrm{a}} + \mathscr{L}^{\mathrm{d}})\hat{u}^n. \tag{6}$$

4.2. Stability of the EXI and EXV method

The stability analysis of the EXI and EXV method is similar and therefore treated simultaneously in this section. The vector of expansion coefficients in element j is assumed to be a Fourier mode:

$$\hat{u}_j^n = \hat{u}^{\mathrm{F}} \exp(\imath\theta j)$$

with $\hat{u}^{\mathrm{F}}$ the amplitude of the mode, $\imath = \sqrt{-1}$ and $\theta \in (-\pi, \pi]$. With this assumption, the Fourier transform of the discrete system becomes:

$$\mathbf{FT}(\mathscr{L})(\theta) = L\exp(-\imath\theta) + D + U\exp(\imath\theta),$$

with L the block-lower, D the block-diagonal and U the block-upper part of the matrix. We begin by noticing that the matrix $\mathbf{FT}(\mathscr{L})(\theta)$ is non-singular and can be diagonalized as QMQ^{-1}, with Q the matrix of right eigenvectors and M the diagonal matrix with the (complex) eigenvalues $\mu_i(\theta)$ with $i = 1, 2, 3$. Using this property and introducing the new vector $w = Q^{-1}\hat{u}^n$ reduces Eq. (6) to the simple scalar test model:

$$\frac{\partial w_i}{\partial \tau} = -\frac{\mu_i(\theta)}{\Delta t} w_i \quad \text{for } i = 1, 2, 3. \tag{7}$$

Note that the summation convention does not apply here and to avoid confusion, we consider the generic scalar model problem of the form:

$$\frac{\partial w}{\partial \tau} = -\frac{\mu}{\Delta t} w,$$

with the understanding that the scalars w and μ can be any of the three components of the corresponding vectors. When applying the EXI method to this model equation, the Runge–Kutta stages w^s are computed as:

$$(1+\alpha_s\lambda)w^s = w^0 + \alpha_s\lambda(1-\mu)w^{s-1},$$

with $\lambda = \Delta\tau/\Delta t$ and for the EXV as:

$$w^s = w^0 - \alpha_s\lambda\mu w^{s-1}.$$

Using these equations the relation between two consecutive pseudo-time steps can easily be derived and is written in generic form as:

$$w^n = G(-\lambda\mu)w^{n-1},$$

with G the algorithm dependent amplification factor. Starting with an initial condition w^{init}, we obtain after n steps:

$$w^n = G(-\lambda\mu)^n w^{\text{init}}.$$

In stability analysis, we are interested in the behavior of a perturbation of the initial condition (see for example [25]), and, due to linearity, the amplification of the perturbation is the same as the amplification of w. Clearly, the perturbation is bounded if $\|G^n\|$ is bounded, where $\|\cdot\|$ denotes the Euclidian (or discrete l_2) norm [12,25]. Therefore, in view of (7), a sufficient condition for stability is that the values $-\lambda\mu_i(\theta)$ for $i = 1,2,3$ and $\theta \in (-\pi,\pi]$ all lie inside the stability domain S given by:

$$S = \{z \in \mathbb{C} : |G(z)| \leqslant 1\}.$$

Remember that the discretization of the advection–diffusion equation only depends on the Courant number (4), the Von Neumann number (5) and the constant η. For given values of these numbers, the factor λ of the Runge–Kutta algorithm should be chosen such that $-\lambda\mu_i(\theta)$ lies inside the stability domain S. Once a suitable λ is found, it is convenient to express the stability in terms of the *pseudo-time* Courant and Von Neumann numbers: $\sigma_{\Delta\tau} = \lambda\sigma$ and $\delta_{\Delta\tau} = \lambda\delta$. Hence, for stability, the pseudo-time step $\Delta\tau$ must satisfy the Courant–Friedrichs–Levy (CFL) condition and the Von Neumann condition:

$$\Delta\tau \leqslant \Delta\tau^{\text{a}} \equiv \frac{\sigma_{\Delta\tau}\Delta x}{a} \quad \text{and} \quad \Delta\tau \leqslant \Delta\tau^{\text{d}} \equiv \frac{\delta_{\Delta\tau}(\Delta x)^2}{d}.$$

We distinguish between flow regimes by introducing the *cell* Reynolds number, defined as:

$$Re_{\Delta x} \equiv \frac{a\Delta x}{d}. \tag{8}$$

In aerodynamical computations, the flow is inviscid in most of the domain, yet significant viscous effects occur in the boundary layer near the airfoil. Therefore, we will consider the following regimes:

(1) steady-state, inviscid: $\sigma = 100$ and $Re_{\Delta x} = 100$,
(2) steady-state, viscous: $\sigma = 100$ and $Re_{\Delta x} = 0.01$,
(3) time-dependent, inviscid: $\sigma = 1$ and $Re_{\Delta x} = 100$,
(4) time-dependent, viscous: $\sigma = 1$ and $Re_{\Delta x} = 0.01$.

The Von Neumann condition can be expressed in terms of the pseudo-time Courant number and the cell Reynolds number as:

$$\Delta\tau^{\text{d}} = \frac{\delta_{\Delta\tau}Re_{\Delta x}}{\sigma_{\Delta\tau}}\Delta\tau^{\text{a}}.$$

Thus, for the inviscid flow regime the CFL condition is the most restrictive, for the viscous flow regime the Von Neumann condition and the threshold between both is given by $\delta_{\Delta\tau}Re_{\Delta x} = \sigma_{\Delta\tau}$.

The stability domains of the EXI and EXV method and the values $-\lambda\mu_i(\theta)$ are plotted in Figs. 1–4 for $i = 1,2,3$ and a discrete series of $\theta = -0.96\pi, 0.92\pi, \ldots, 1.00\pi$. For inviscid flow regimes with pseudo-time step Courant number around $\sigma_{\Delta\tau} = 1.7$, the EXI method is stable and the EXV is unstable, but for viscous flow regimes with pseudo-time step Von Neumann number $\delta_{\Delta\tau} = 0.8$, the converse holds. Stability constraints for which both methods are stable are given in Table 1, confirming that the EXI method is preferable in

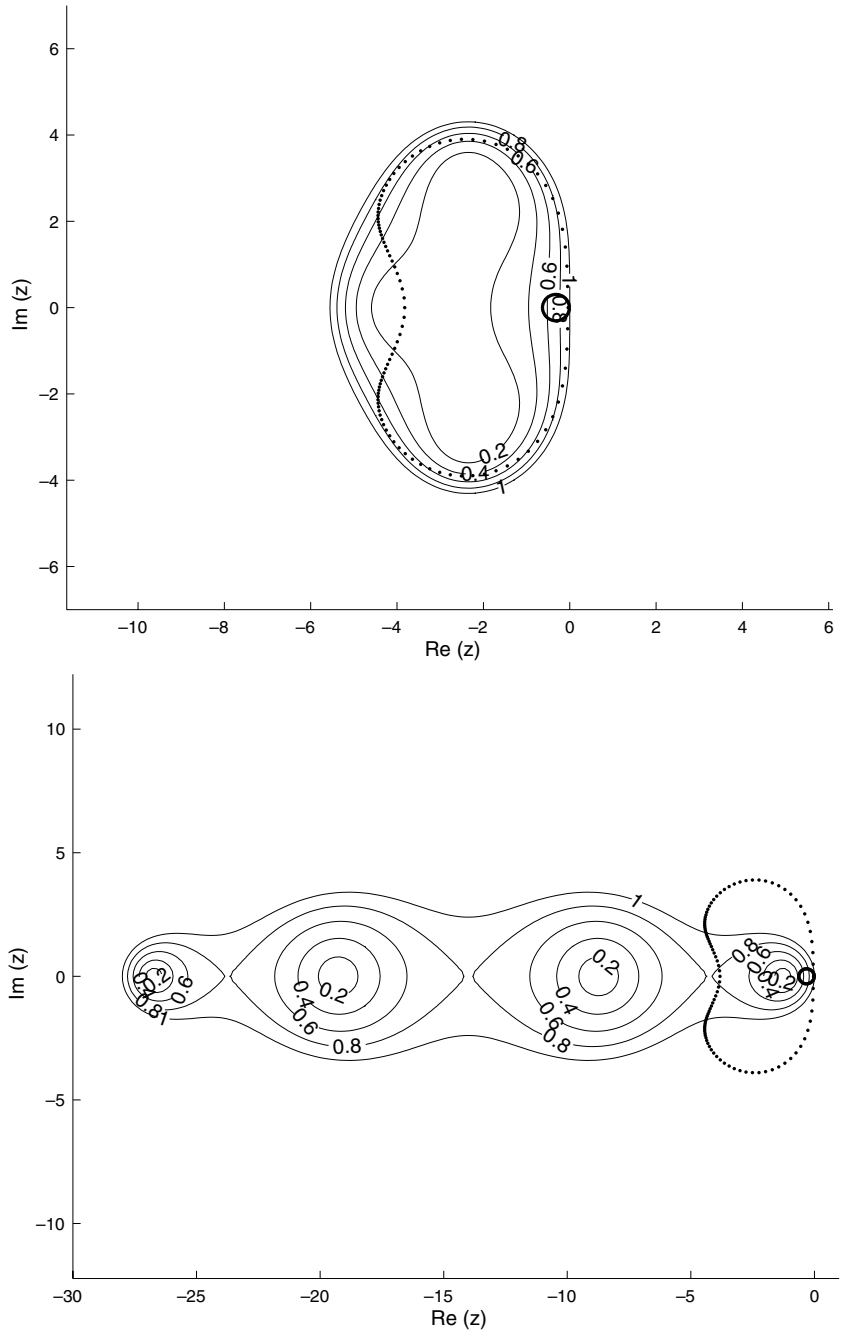


Fig. 1. The stability domain S and values $-\lambda\mu_i$ (dots) for the EXI method (top) and EXV method (bottom) in the steady-state inviscid flow regime with $\lambda = 1.8 \times 10^{-2}$. The pseudo-time CFL number is 1.8 and for this constraint only the EXI method is stable.

the inviscid regime and the EXV in the viscous regime. Therefore, we combine the EXI and EXV by looking at the cell Reynolds number, and, for that particular cell, deploy whichever scheme has the mildest stability restriction.

Remark 1. The stabilization parameter η has a significant effect on the stability of the pseudo-time integration: as η increases, the pseudo-time Von Neumann number decreases proportionally. Therefore, η

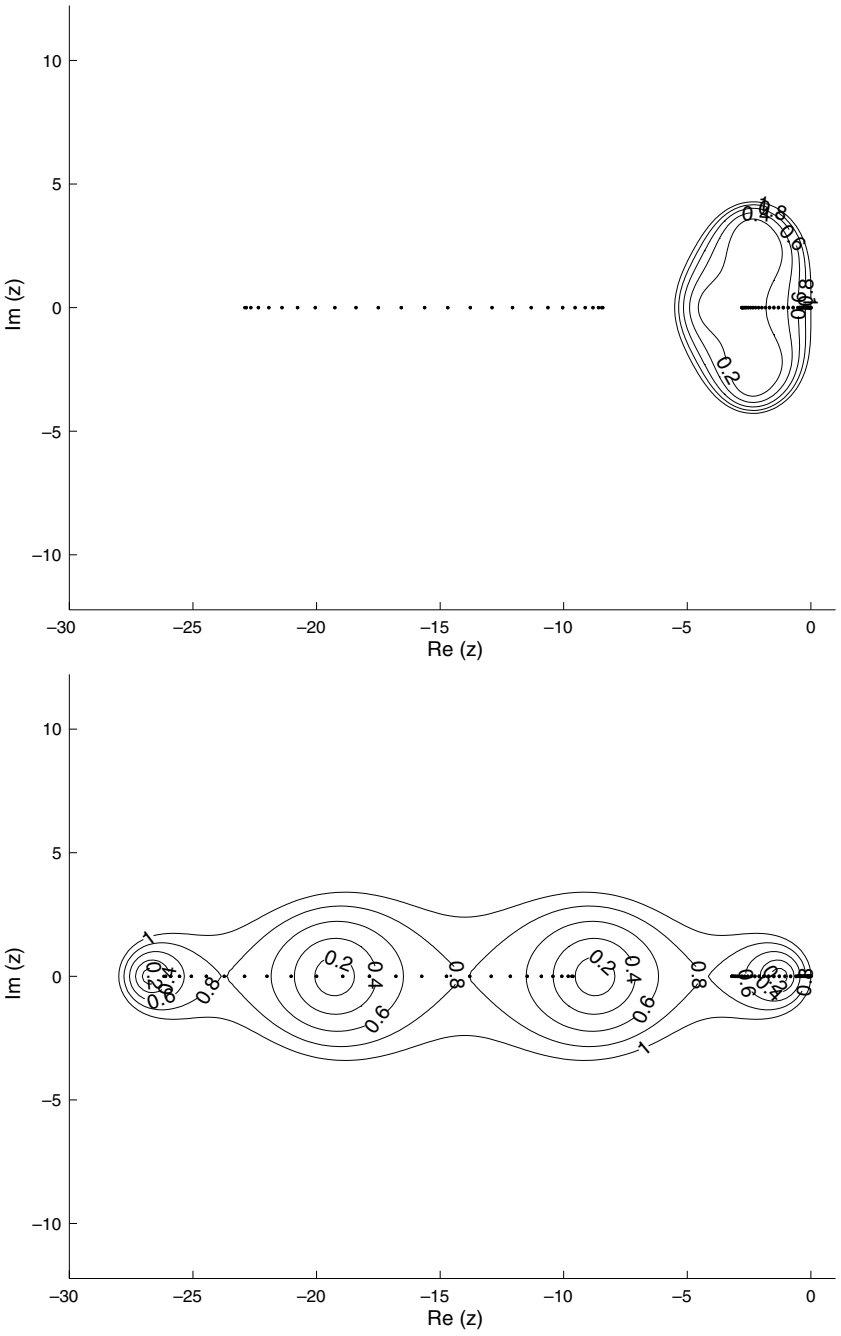


Fig. 2. The stability domain S and values $-\lambda\mu_i$ (dots) for the EXI method (top) and EXV method (bottom) in the steady-state viscous flow regime with $\lambda = 8 \times 10^{-5}$. The pseudo-time Von Neumann number is 0.8 and for this constraint only the EXV method is stable.

should be taken as small as allowed in the discontinuous Galerkin discretization, in general equal to the number of faces of an element [7,22].

Remark 2. The Melson correction is applied to the EXI scheme to ensure stability for values of λ around one, which is the case for the time-dependent inviscid flow regime (Fig. 3). For all other flow regimes, λ is small and the Melson correction vanishes. Since we only apply the EXV scheme in the viscous flow regime, the Melson correction is unnecessary for this scheme.

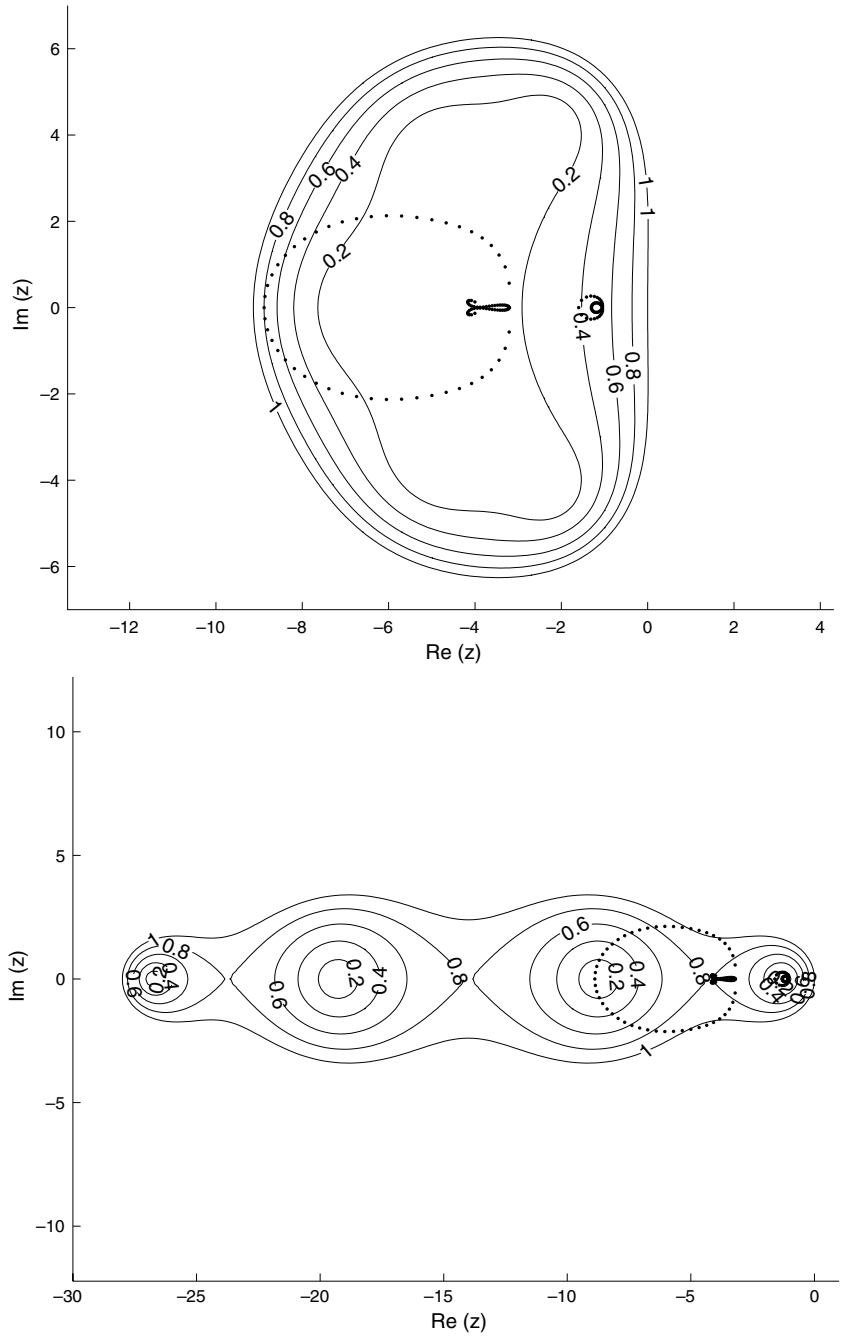


Fig. 3. The stability domain S and values $-\lambda\mu_i$ (dots) for the EXI method (top) and EXV method (bottom) in the time-dependent inviscid flow regime with $\lambda = 1.6$. The pseudo-time CFL number is 1.6 and for this constraint only the EXI method is stable.

4.3. Stability of the IMEX method

The IMEX method solves the inviscid part of the equations with the EXI method and treats the viscous part implicitly. The main idea is that the stability should now only depend on the inviscid part, so only the CFL condition has to be satisfied. This represents the ideal situation where viscosity does not affect stability, needed to assess the influence of the Von Neumann condition on the fully explicit methods. Unfortunately, the

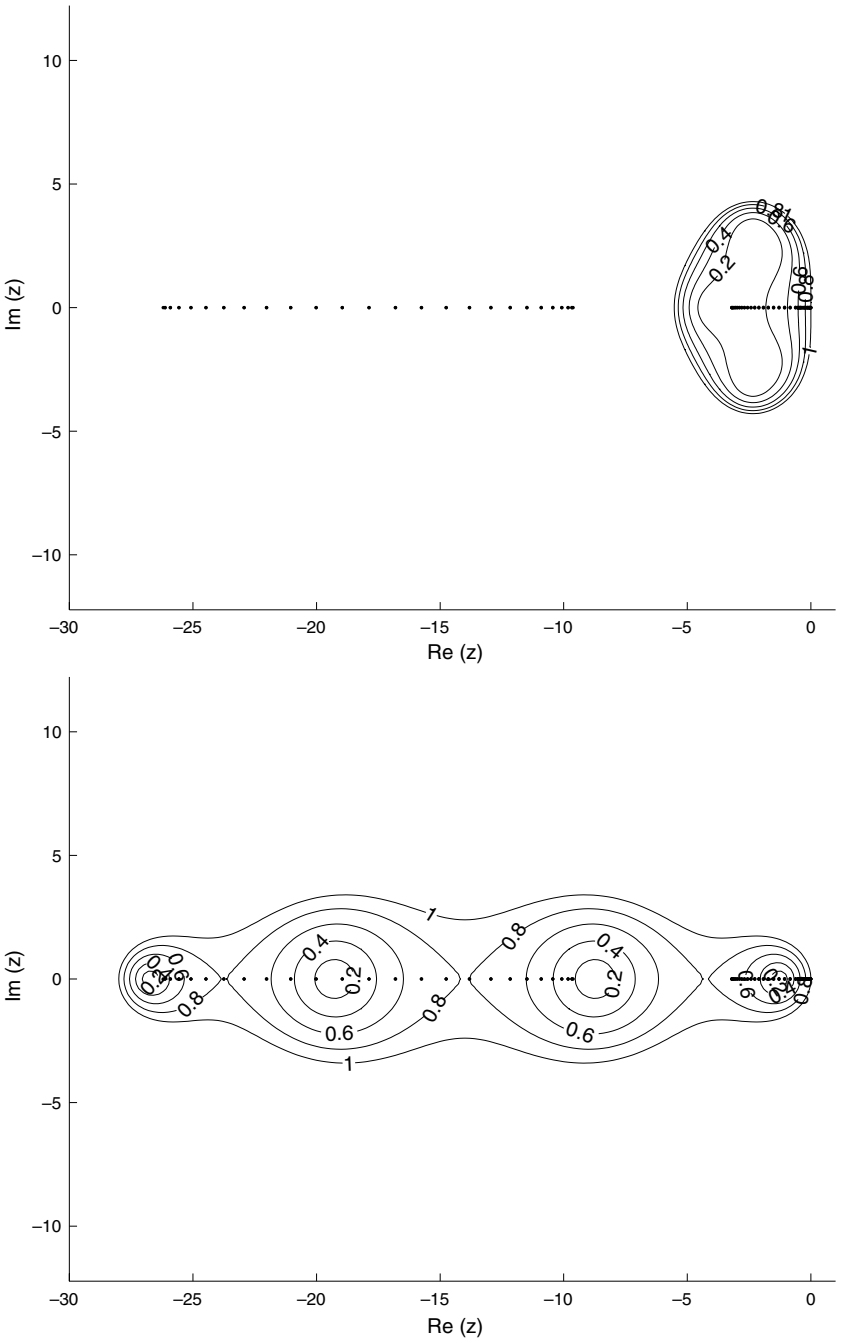


Fig. 4. The stability domain S and values $-\lambda\mu_i$ (dots) for the EXI method (top) and EXV method (bottom) in the time-dependent viscous flow regime with $\lambda = 8 \times 10^{-3}$. The pseudo-time Von Neumann number is 0.8 and for this constraint only the EXV method is stable.

matrices $\mathscr{L}^{\mathrm{a}}$ and $\mathscr{L}^{\mathrm{d}}$ in (6) do not commute, making it impossible to obtain a scalar model problem through diagonalization, as was done for the explicit method. Stability analysis of IMEX methods for general non-commuting matrices is still largely an open problem, although recently, for the related W-methods, results have been presented by Ostermann [19]. In this section, we will prove stability of the IMEX method by directly estimating the norm of the amplification factor G.

Table 1
Stability constraints of the EXI and EXV methods

	Flow regime		Stability restriction	
	σ	$Re_{\Delta x}$	EXI	EXV
Steady-state, inviscid	100	100	$\sigma_{\Delta\tau} \leqslant 1.8$	$\sigma_{\Delta\tau} \leqslant 0.3$
Steady-state, viscous	100	0.01	$\delta_{\Delta\tau} \leqslant 0.1$	$\delta_{\Delta\tau} \leqslant 0.8$
Time-dependent, inviscid	1	100	$\sigma_{\Delta\tau} \leqslant 1.6$	$\sigma_{\Delta\tau} \leqslant 1.0$
Time-dependent, viscous	1	0.01	$\delta_{\Delta\tau} \leqslant 0.1$	$\delta_{\Delta\tau} \leqslant 0.8$

For the IMEX method the Runge–Kutta stages $\hat{v}^s$ are computed by solving the sparse linear system:

$$(I + \alpha_s\lambda(I + \mathscr{L}^{\mathrm{d}}))\hat{v}^s = \hat{v}^0 + \alpha_s\lambda(I - \mathscr{L}^{\mathrm{a}})\hat{v}^{s-1}. \tag{9}$$

The starting point of our analysis is the fact that $\mathscr{L}^{\mathrm{d}}$ is a Hermitian matrix, therefore $\mathscr{L}^{\mathrm{d}} = QMQ^{\mathrm{T}}$ where Q is a unitary matrix and M the diagonal matrix with the eigenvalues μ_i of $\mathscr{L}^{\mathrm{d}}$. The eigenvalues of $\mathscr{L}^{\mathrm{d}}$ are real and positive, and can be computed as the eigenvalues $\mu_i(\theta)$ with $i = 1, 2, 3$ and $\theta \in (-\pi, \pi]$ of the corresponding Fourier transform:

$$\mathbf{FT}(\mathscr{L}^{\mathrm{d}})(\theta) = L^{\mathrm{d}}\exp(-\imath\theta) + D^{\mathrm{d}} + U^{\mathrm{d}}\exp(\imath\theta),$$

with L^{d} the block-lower, D^{d} the block-diagonal and U^{d} the block-upper part of the matrix $\mathscr{L}^{\mathrm{d}}$. For a unitary matrix $Q^{-1} = Q^{\mathrm{T}}$ and the l.h.s. of (9) can be written as:

$$I + \alpha_s\lambda(I + \mathscr{L}^{\mathrm{d}}) = Q(I + \alpha_s\lambda(I + M))Q^{\mathrm{T}} = QM_sQ^{\mathrm{T}}, \tag{10}$$

with M_s the diagonal matrix with values $1 + \alpha_s\lambda\,(1 + \mu_i)$. Introducing the decomposition (10) into (9) gives:

$$M_s w^s = w^0 + \alpha_s\lambda Q^{\mathrm{T}}(I - \mathscr{L}^{\mathrm{a}})Qw^{s-1} = w^0 + \alpha_s\lambda P_a w^{s-1}, \tag{11}$$

with $w^s = Q^{\mathrm{T}}\hat{v}^s$ and $P_a = Q^{\mathrm{T}}(I - \mathscr{L}^{\mathrm{a}})Q$. Therefore, the relation between two consecutive pseudo-time steps is: $w^n = G\,w^{n-1}$ with the amplification matrix G defined as:

$$G = M_5^{-1}(I + \alpha_5\lambda P_a M_4^{-1}(I + \alpha_4\lambda P_a \cdots M_1^{-1}(I + \alpha_1\lambda P_a))).$$

If $\|G\| \leqslant 1$, then $\|G^n\| \leqslant 1$ and the method is stable. Our stability analysis aims at a direct estimation of this norm, therefore we consider the following upper bound:

$$\|G\| \leqslant \|M_5^{-1}\|(1 + \alpha_5\lambda\|P_a\|\|M_4^{-1}\|(1 + \alpha_4\lambda\|P_a\| \cdots \|M_1^{-1}\|(1 + \alpha_1\lambda\|P_a\|))).$$

The matrices M_s^{-1} are equal to:

$$M_s^{-1} = \mathrm{diag}\left(\frac{1}{1 + \alpha_s\lambda(1 + \mu_1)}, \ldots, \frac{1}{1 + \alpha_s\lambda(1 + \mu_n)}\right),$$

with μ_i the eigenvalues of $\mathscr{L}^{\mathrm{d}}$. The Euclidian norm of M_s^{-1} can be estimated as:

$$\|M_s^{-1}\| = \max_{i\in\{1,\ldots,n\}} \frac{1}{1 + \alpha_s\lambda(1 + \mu_i)} < \frac{1}{1 + \alpha_s\lambda},$$

since $\mu_i, \alpha_s, \lambda > 0$. Using this estimation, the upper bound for the Euclidian norm of G is then provided by the following estimate:

$$\|G\| \leqslant \frac{1}{1 + \alpha_5\lambda}\left(1 + \alpha_5\lambda\|P_a\|\frac{1}{1 + \alpha_4\lambda}\left(1 + \alpha_4\lambda\|P_a\| \cdots \frac{1}{1 + \alpha_1\lambda}(1 + \alpha_1\lambda\|P_a\|)\right)\right).$$

The r.h.s. of this equation is called the stability function, denoted by $f(\lambda, \|P_a\|)$ and plotted for $\|P_a\| = 1$ in Fig. 5. If $\|P_a\| < 1$ we find ourselves below the curve in Fig. 5, therefore: $\|P_a\| \leqslant 1 \Rightarrow f(\lambda, \|P_a\|) \leqslant 1 \Rightarrow \|G\| \leqslant 1$ meaning $\|P_a\| \leqslant 1$ is a sufficient condition for stability of the implicit–explicit method. Since the matrix P_a is defined as $P_a = Q^{\mathrm{T}}(I - \mathscr{L}^{\mathrm{a}})Q$, with Q a unitary matrix (hence $\|Q\| = 1$), this implies that the stability of the IMEX method is only determined by the following condition:

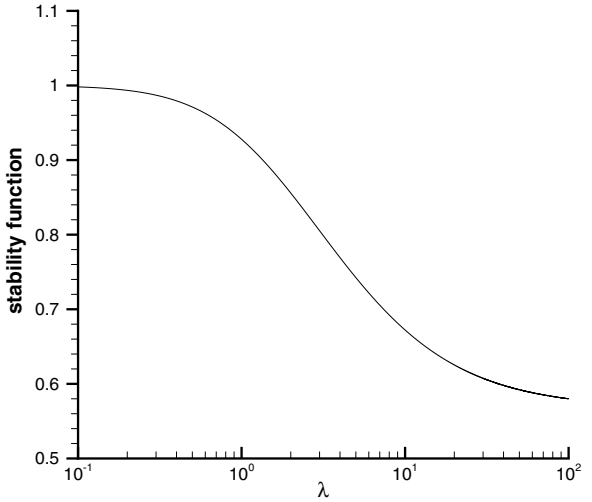


Fig. 5. The stability function f for $\|P_d\| = 1$.

$$\|I - \mathscr{L}^{\mathrm{a}}\| \leqslant 1.$$

Since $\mathscr{L}^{\mathrm{a}}$ only depends on the Courant number (4), this condition implies that the IMEX method is stable independently of the Von Neumann number, and only the CFL condition has to be satisfied.

The fact that the IMEX method does not require the Von Neumann condition for stability makes it suitable to evaluate (by comparison) the effect of viscosity on the stability and performance of fully explicit methods. We can thus assess whether the combined EXI and EXV method effectively relieves the Von Neumann constraint. We will address this issue in the following section through numerical experiments.

5. Results

In this section, three benchmark problems for the compressible Navier–Stokes equations are considered. First, we explain how to compute the local pseudo-time step size. Then, we compare the performance of the EXI method with the combined EXI and EXV method and the IMEX method to assess the effect of viscosity on the stability and performance of the explicit methods. Finally, we show the feasibility of the explicit methods for the flow around a 3D delta wing.

5.1. Local pseudo-time step-size

The space–time method is unconditionally stable in *physical* time, which allows us to take any desired physical time step Δt and solve the non-linear system using the pseudo-time stepping methods discussed in this paper. Since accuracy is not an issue in *pseudo-time* we can use local steps $(\Delta\tau)_K$, which are determined for each element K as:

$$(\Delta\tau)_K = \begin{cases} \min\{(\Delta\tau)^{\mathrm{i}}_K, (\Delta\tau)^{\mathrm{v}}_K\} & \text{for EXI and EXV,} \\ (\Delta\tau)^{\mathrm{i}}_K & \text{for IMEX.} \end{cases}$$

The local inviscid and viscous pseudo-time steps are computed as:

$$(\Delta\tau)^{\mathrm{i}}_K = \frac{\sigma_{\Delta\tau} d_K}{\lambda^{\mathrm{i}}_K} \quad \text{with} \quad \lambda^{\mathrm{i}}_K = \max\{|u_K - a_K|, |u_K + a_K|\},$$

$$(\Delta\tau)^{\mathrm{v}}_K = \frac{\delta_{\Delta\tau} (d_K)^2}{\lambda^{\mathrm{v}}_K} \quad \text{with} \quad \lambda^{\mathrm{v}}_K = \max\left\{\frac{1}{c_v}\frac{\kappa_K}{\rho_K}, \frac{4}{3}\frac{\mu_K}{\rho_K}\right\},$$

where $\sigma_{\Delta\tau}$ is the pseudo-time step Courant number, $\delta_{\Delta\tau}$ the pseudo-time step Von Neumann number (both from Table 1) and d_K the diameter of the circle inscribed in element K. The cell Reynolds number Re_K is defined as:

$$Re_K = \frac{\lambda_K^{\mathrm{i}} d_K}{\lambda_K^{\mathrm{v}}},$$

and λ^{i} represents the absolute maximum of the eigenvalues of the inviscid Jacobian and λ^{v} of the viscous Jacobian. These wave-speeds are computed at the element faces during the flux evaluation and we take the maximum over all faces belonging to element K, where u is the flow speed, a the speed of sound and ρ the density. Note that the specific heat at constant volume c_{v} is constant throughout the domain but the dynamic viscosity μ and the thermal conductivity coefficient κ depend on the temperature in element K, see [14,17]. Even though the stability analysis was only done for the advection–diffusion equation on a periodic domain, the resulting stability constraints proved also adequate in case of the compressible Navier–Stokes equations.

5.2. NACA0012 airfoil

5.2.1. Steady-state case

To test the effect of viscosity on the stability and performance of the explicit methods, we have chosen the A1 case described in [8] for the viscous flow past a NACA0012 airfoil. This case has become a standard benchmark for discontinuous Galerkin methods for the compressible Navier–Stokes equations as it was treated in the seminal paper by Bassi and Rebay [3]. The Prandtl number is fixed at $Pr = 0.72$, the far-field Mach and Reynolds numbers at $M_\infty = 0.8$ and $Re_\infty = 73$, respectively. With an angle of attack $\alpha = 12°$ this gives a highly viscous flow. Since it is a steady-state flow, we take one huge physical time step $a_\infty\,\Delta t/c = 10^{10}$ with a_∞ the far-field speed of sound and c the chord of the airfoil. For laminar viscous flow, the boundary layer thickness at the nose of the airfoil is estimated as [21]:

$$b/c \approx 5/\sqrt{Re_\infty},$$

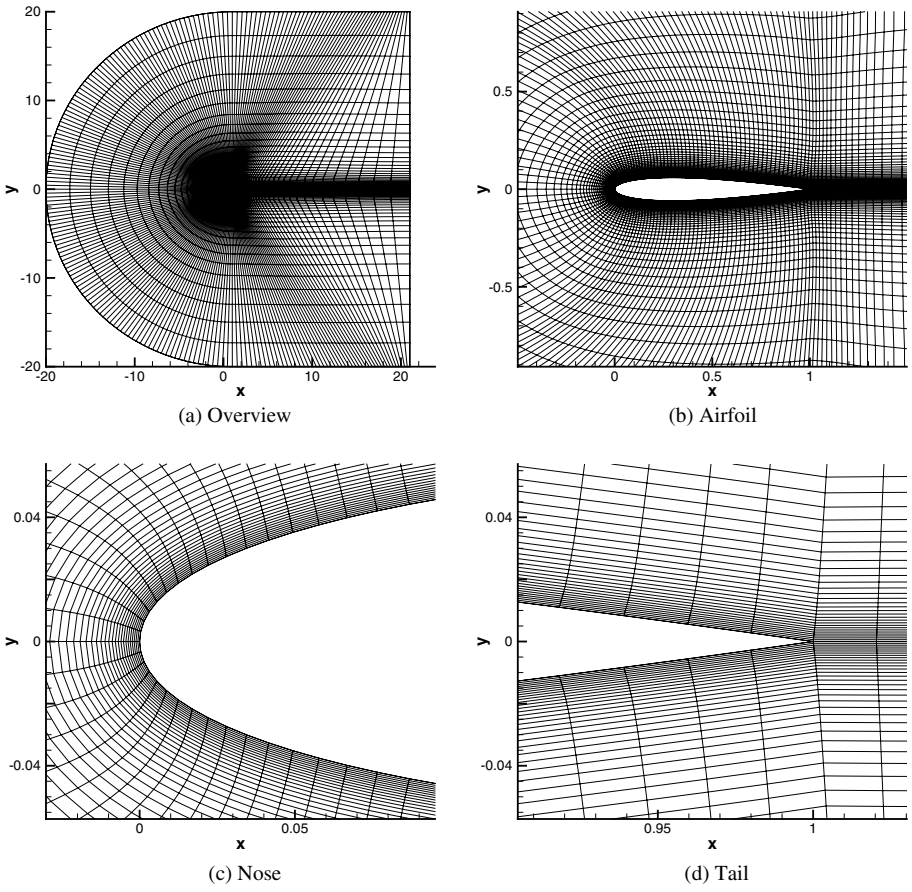
with c the airfoil chord length, which means that $b/c \approx 0.6$ in this case. To compute the boundary layer with reasonable accuracy, we have chosen a C-type mesh with 112×38 elements, which is a coarsening of the 224×76 grid shown in Fig. 6.

The local Mach number isolines and the convergence results are presented in Fig. 7. The cell Reynolds number varies between 0.09 and 88 which explains why the convergence of the EXI method is very slow: one order of convergence in 80,000 pseudo-time iterations. For the combined EXI and EXV method, one order of convergence requires ten times less iterations, which is roughly what we expect from the stability analysis as the Von Neumann number for the EXV method is eight times larger (Table 1). Furthermore, the combined method achieves seven orders of convergence within 50,000 steps. The IMEX method performs much better in terms of iterations: six orders of convergence in 3000 pseudo-time steps. Therefore, we conclude that combining the EXI method with the EXV method significantly improves the convergence in pseudo-time but does not completely rule out the effect of viscosity on the stability restriction and thus on the convergence of explicit methods.

Remark 3. For the IMEX method, the implicit linear system must be solved at each Runge–Kutta stage. We do so using the block sparse GMRES solver with Jacobi preconditioner available in the PETSc package [20]. In terms of CPU-time, the IMEX method converges significantly slower than the combined EXI and EXV method in all considered cases and is only pursued to evaluate the effect of the viscosity on the fully explicit methods.

5.2.2. Time-dependent case

To test the performance of the explicit pseudo-time stepping methods for time-dependent simulations, we have chosen the A7 case described in [8]. The Prandtl number is fixed at $Pr = 0.72$, the far-field Mach and Reynolds numbers at $M_\infty = 0.85$ and $Re_\infty = 10^4$, respectively, and the angle of attack is $\alpha = 0°$, which gives



(a) Overview (b) Airfoil

(c) Nose (d) Tail

Fig. 6. Details of the NACA0012 C-grid with 224×76 elements.

a time-dependent viscous flow with a shock and vortex shedding. To capture the vortex shedding, we have estimated the velocity of the vortices to be around $u/a_\infty = 0.8$ and the diameter to be around $D/c = 0.1$, which, together with a physical time step $a_\infty \Delta t/c = 0.05$, gives a physical Courant number $u\Delta t/D = 0.4$: small enough for accuracy in time. To compute the boundary layer in the A7 case with reasonable accuracy, we use the C-type grid for viscous flows shown in Fig. 6 with 224×76 elements which offers more than 30 elements in the boundary layer with thickness $b/c \approx 0.05$.

In the A7 case, for each physical-time step, the EXI method achieves three orders of convergence in 1000 pseudo-time steps, see Fig. 8. The physical time-step is already fairly small in order to capture the vortex shedding, which explains the relatively small number of pseudo-time steps needed to solve the system. In this case, the cell Reynolds number varies between 2 and 14,000, which explains why the difference between the EXI and the combined EXI and EXV method is less than in the A1 case. Still, the convergence is twice as fast. The IMEX method requires 200 iterations, which is two and half times faster in terms of iterations than the combined EXI and EXV method. Therefore, as in the steady-state case A1, we conclude that combining the EXI method with the EXV method significantly improves the convergence in pseudo-time but viscosity still has some influence on the convergence of the combined EXI and EXV method.

5.3. 3D delta wing

In [14], we presented results for the steady-state flow around a 3D delta wing at $M_\infty = 0.3$, $Re = 4 \times 10^4$ and $\alpha = 12.5°$ on an adapted mesh. The mesh adaptation was based on the vorticity in order to capture the primary and secondary vortices along both sides of the sharp leading edge of the wing, see [14] for the wing

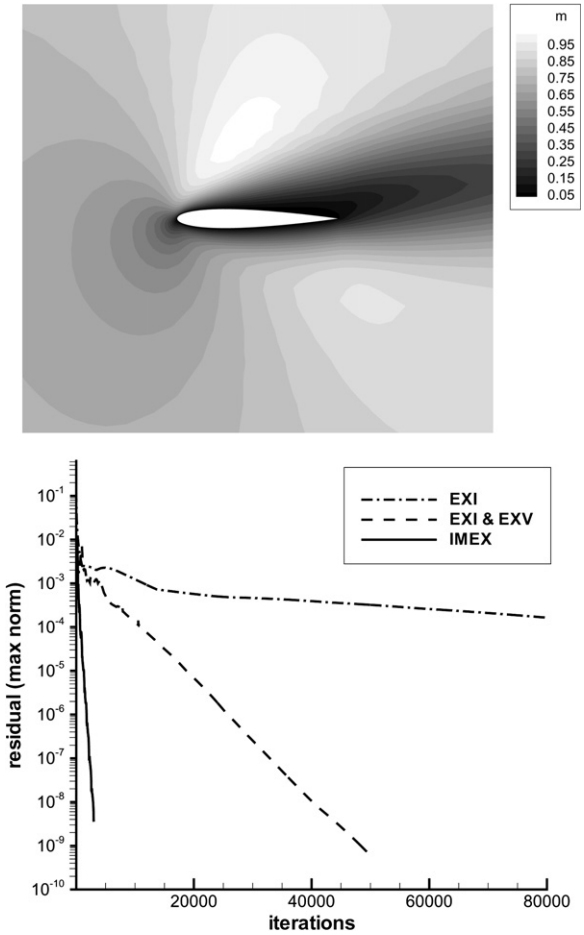


Fig. 7. Local Mach numbers for the A1 case ($M_\infty = 0.8$, $Re_\infty = 73$, $\alpha = 12°$) on the 112×38 grid and convergence to steady-state for the different pseudo-time stepping methods.

geometry, an impression of the mesh and the flow field. The results for the pressure coefficient and the helicity showed good agreement with those obtained on a fine mesh (1,650,000 elements). Therefore, we concluded that the solution adaptive space–time method results in significant cost savings for this application. Here, we show the convergence for this case using the EXI and combined EXI and EXV method. The IMEX method was not attempted in this case due to the size of the linear system.

We start by computing the solution on the coarse mesh with 208,896 elements, with uniform flow as initial condition. Based on a comparison with the solution on the fine mesh, we consider the case solved after three orders of convergence, see Fig. 9 for an impression of the solution. In Fig. 10, we see that the combined EXI and EXV method meets our convergence criterion in 5000 pseudo-time steps while the EXI method takes 15,000 steps. Next, anisotropic mesh refinement is applied in vortex regions in such a way that eventually the mesh in the region around a vortex is as uniform as possible. A cell is refined in any of the three directions whenever the vorticity is greater than $2a_\infty/c$ and the mesh width in the given direction is greater than $0.01c$. In this way, the mesh around the primary vortices will eventually have a uniform resolution of $0.005c$. Mesh adaptation is based on the (more accurate) solution obtained with the combined EXI and EXV method. After each mesh adaptation, the residual has to be decreased again. We do so with the EXI and the combined EXI and EXV method and show the convergence results in Fig. 10. The advantage of the combined EXI and EXV method becomes even more apparent, since, at each adaptation, the percentage of relatively small cells (and thus of small cell Reynolds numbers) increases. On the final adapted grid, the cell Reynolds number varies

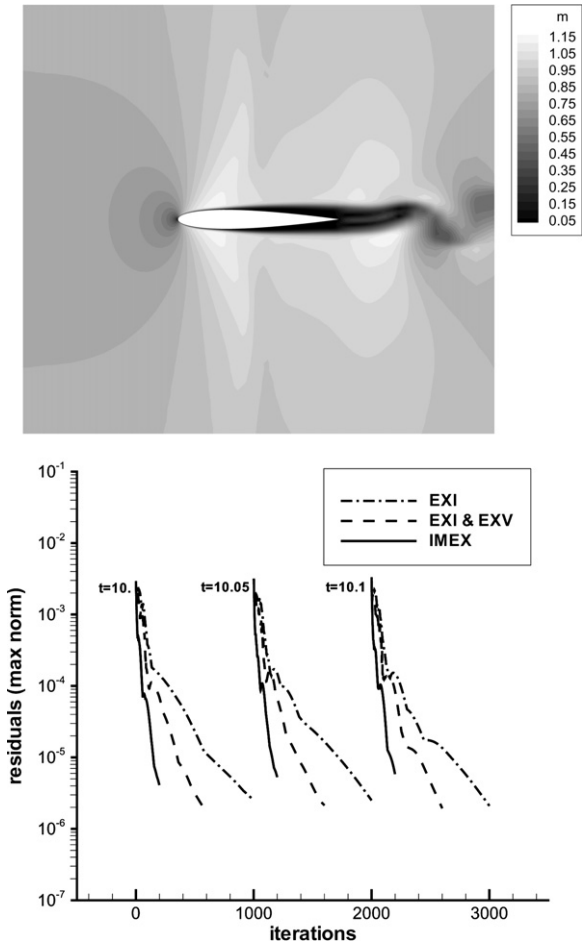


Fig. 8. Local Mach numbers for the A7 case ($M_\infty = 0.85$, $Re_\infty = 10^4$, $\alpha = 0°$) on the 224×76 grid at $t = 10$ and convergence in pseudo-time for three physical time steps with the different pseudo-time stepping methods.

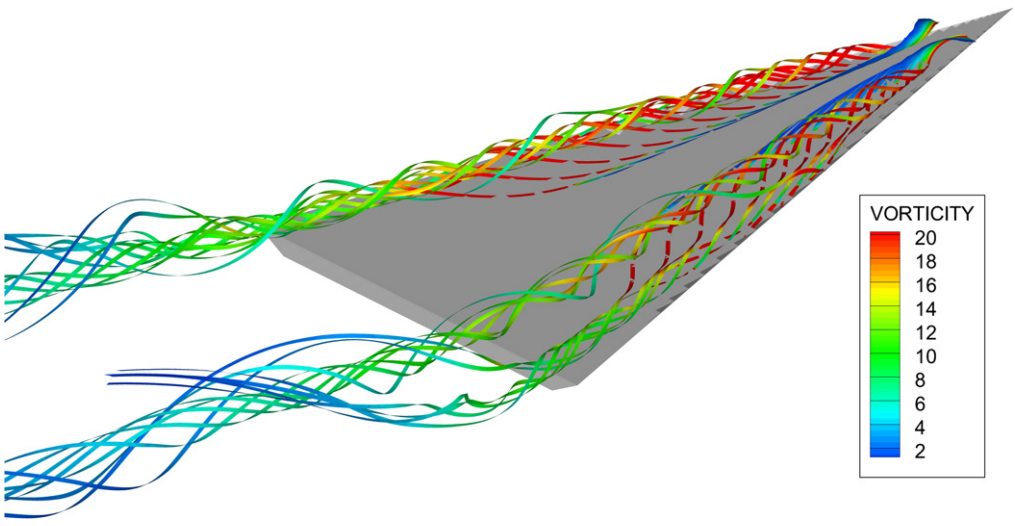
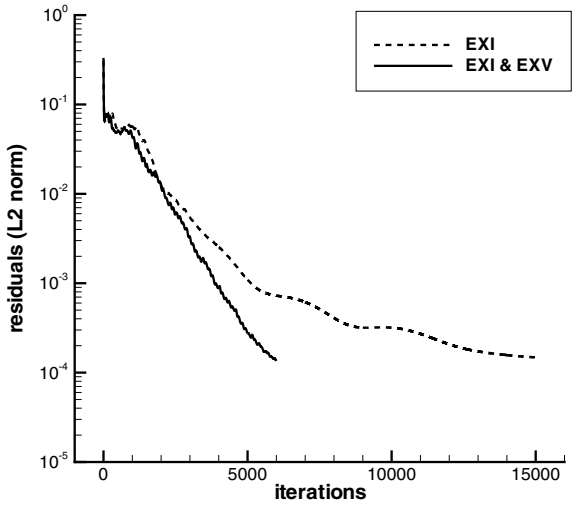
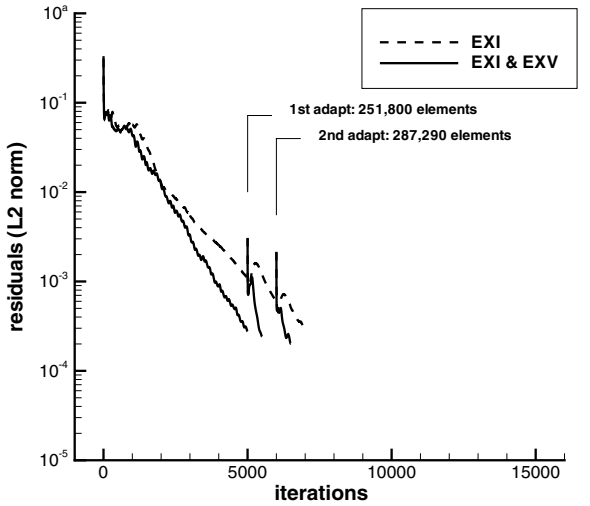


Fig. 9. Streamlines around the 3D delta wing ($M_\infty = 0.3$, $Re_\infty = 4 \times 10^4$, $\alpha = 12.5°$).



(a) coarse mesh with 208 896 elements



(b) mesh adaptations

Fig. 10. Convergence to steady-state with the explicit pseudo-time stepping methods for the 3D delta wing ($M_\infty = 0.3$, $Re_\infty = 4 \times 10^4$, $\alpha = 12.5°$) starting from uniform flow on a coarse mesh (a) and with two successive mesh adaptations (b).

between 0.4 and 118,000, confirming the robustness of the combined EXI and EXV method for a wide range of cell Reynolds numbers.

6. Discussion and conclusions

When applying the space–time discontinuous Galerkin method to the compressible Navier–Stokes equations one obtains a system of non-linear algebraic equations. To solve this system we developed a fully explicit pseudo-time stepping method suitable for a wide range of applications. This was achieved by combining two explicit Runge–Kutta methods: the EXI method for the inviscid part of the flow domain and the EXV method for the viscous part.

We showed that these methods are stable if either the pseudo-time CFL or the Von Neumann condition is satisfied, depending on the cell Reynolds number. In the inviscid flow regime, the cell Reynolds numbers are large and the CFL condition is the most restrictive, while in the viscous flow regime low cell Reynolds numbers

occur so that the Von Neumann condition is the most restrictive. To assess the influence of viscosity on the performance of the explicit methods, we also considered the implicit–explicit version of the EXI method where the viscous terms are treated implicitly and the inviscid terms explicitly. We showed that the stability of the IMEX method only depends on the inviscid part, thereby effectively relieving us of the Von Neumann condition, and providing a method against which we could test our explicit algorithms.

The combination of the EXI and EXV method significantly improved the convergence in pseudo-time compared to the EXI method, yet viscosity is still limiting the convergence when compared to the IMEX results in terms of number of iterations. In terms of CPU-time, the IMEX method is significantly slower due to the linear system which must be solved at each iteration. The main advantage of the combined EXI and EXV method is its locality which matches the locality of the space–time discontinuous Galerkin discretization and dispenses with the assemblage and storage of a global (sparse) matrix. We showed that the method is feasible for the 3D flow around a delta wing involving 300,000 elements. In our current work, we are focusing on further reducing the computational effort by applying the combined EXI and EXV method as a smoother in a multigrid algorithm [15].

Acknowledgments

The advice and support of Dr. A. Bell concerning the PETSc package and its sparse linear solvers is greatly appreciated. This research has been conducted in the STW project TWI.5541, entitled *Advanced simulation techniques for vortex dominated flows in aerodynamics*. The financial support from STW and the National Aerospace Laboratory NLR is gratefully acknowledged.

Appendix A. Details of EXV method

In [16] a family of Runge–Kutta schemes for efficient time-marching of viscous flow problems is presented. We used a member of this family, the EXV method, for local pseudo-time stepping in flow regions with low cell Reynolds numbers. In this appendix, we summarize the derivation of the entire family.

Consider the following N stage Runge–Kutta scheme:

(1) Initialize $\hat{v}^0 = \hat{u}$.
(2) For all stages $s = 1$ to N compute $\hat{v}^s$ as:

$$\hat{v}^s = \hat{v}^0 - \alpha_s \lambda \mathscr{L}(\hat{v}^{s-1}; \hat{u}^{n-1}).$$

(3) Return $\hat{u} = \hat{v}^N$.

When applied to the simple model problem:

$$\frac{\partial u}{\partial \tau} = -\frac{\mu}{\Delta t} u,$$

the stages s are updated according to: $v^s = v^0 - \alpha_s \lambda \mu v^{s-1}$ and therefore the amplification factor G is of the form:

$$G_N(z) = 1 + \alpha_N z + \alpha_N \alpha_{N-1} z^2 + \cdots + \alpha_N \cdots \alpha_1 z^N, \tag{A.1}$$

with $z = -\lambda\mu \in \mathbb{C}$. The family of Runge–Kutta schemes proposed in [16] can be derived by choosing the coefficients α_s in such a way that the amplification factor equals Manteuffel's transformation of Tchebyshev polynomials:

$$G_N(z) = \frac{T_N((d-z)/\epsilon)}{T_N(d/\epsilon)},$$

where T_N denotes the Nth Tchebyshev polynomial defined recursively as:

$$T_{n+1}(z) = 2zT_n(z) - T_{n-1}, \quad n \in \mathbb{N},$$

with $T_0(z) = 1$ and $T_1(z) = z$. Here, the parameter d defines the family of N stage Runge–Kutta schemes and the parameter ϵ is chosen such that:

$$G_N(0) = 1 \quad \text{and} \quad \left.\frac{\mathrm{d}G_N}{\mathrm{d}z}\right|_{z=0} = 1,$$

which ensures that the stability region touches the imaginary axis and is symmetric w.r.t. the real axis. The parameter d controls the scaling of the stability region.

The family member used in this paper is the 4-stage Runge–Kutta scheme with $d = -14$. For this scheme, we use the fourth order Tchebyshev polynomial: $T_4(z) = 8z^4 - 8z^2 + 1$ and obtain the following stability region:

$$G_4(z) = 1 + \frac{16\epsilon^2 d - 32d^3}{D} z + \frac{48d^2 - 8\epsilon^2}{D} z^2 - \frac{32d}{D} z^3 + \frac{8}{D} z^4, \tag{A.2}$$

with $D = 8d^4 - 8d^2\epsilon^2 + \epsilon^4$. The coefficients α_s can now be computed by equating (A.2) with (A.1) which gives:

$$\alpha_1 = -\frac{1}{4d}, \quad \alpha_2 = \frac{4d}{\epsilon^2 - 6d^2}, \quad \alpha_3 = \frac{6d^2 - \epsilon^2}{2d(\epsilon^2 - 2d^2)}, \quad \alpha_4 = \frac{16d(\epsilon^2 - 2d^2)}{D}.$$

The condition $G_4(0) = 1$ is already satisfied and the condition on the derivative of G becomes:

$$\frac{16\epsilon^2 d - 32d^3}{D} = 1,$$

which has four solutions for ϵ from which we choose the following:

$$\epsilon = \sqrt{4d(d+2) - 2\sqrt{16d^2 + 8d^3 + 2d^4}}.$$

In the same way, we can derive the other members of the family. Note, however, that only even N produces consistent schemes.

References

[1] D. Arnold, F. Brezzi, B. Cockburn, D. Marini, Unified analysis of discontinuous Galerkin methods for elliptic problems, SIAM J. Numer. Anal. 39 (2002) 1749–1779.

[2] F. Bassi, A. Crivellini, S. Rebay, M. Savini, Discontinuous Galerkin solution of the Reynolds-averaged Navier–Stokes and k–ω turbulence model equations, Comput. Fluids 34 (4–5) (2004) 507–540.

[3] F. Bassi, S. Rebay, A high-order accurate discontinuous finite element method for the numerical solution of the compressible Navier–Stokes equations, J. Comput. Phys. 131 (1997) 267–279.

[4] F. Bassi, S. Rebay, GMRES discontinuous Galerkin solution of the compressible Navier–Stokes equations, in: B. Cockburn, G. Karniadakis, C.-W. Shu (Eds.), Discontinuous Galerkin Methods, vol. 11, Springer, Berlin, 1999, pp. 197–208.

[5] F. Bassi, S. Rebay, Numerical evaluation of two discontinuous Galerkin methods for the compressible Navier–Stokes equations, Int. J. Numer. Meth. Fluids 40 (2002) 197–207.

[6] C.E. Baumann, J.T. Oden, A discontinuous *hp* finite element method for the Euler and Navier–Stokes equations. Tenth international conference on finite elements in fluids (Tucson, AZ, 1998), Int. J. Numer. Methods Fluids 31 (1) (1999) 79–95.

[7] F. Brezzi, G. Manzini, D. Marini, P. Pietra, A. Russo, Discontinuous Galerkin approximations for elliptic problems, Numer. Meth. Part. Diff. Eq. 16 (4) (2000) 365–378.

[8] M.O. Bristeau, R. Glowinski, J. Periaux, H. Viviand (Eds.), Numerical Simulation of Compressible Navier–Stokes Flows: A GAMM Workshop, 4–6 December 1985, Nice, France, Braunschweig, Vieweg, 1987.

[9] B. Cockburn, C.-W. Shu, The local discontinuous Galerkin method for time-dependent convection–diffusion systems, SIAM J. Numer. Anal. 35 (1998) 2240–2463.

[10] B. Cockburn, C.-W. Shu, Runge–Kutta discontinuous Galerkin methods for convection-dominated problems, J. Sci. Comput. 16 (3) (2001) 173–261.

[11] V. Dolejší, On the discontinuous Galerkin method for the numerical solution of the Navier–Stokes equations, Int. J. Numer. Meth. Fluids 45 (2004) 1083–1106.

[12] E. Hairer, G. Wanner, Solving Ordinary Differential Equations II. Stiff and Differential-Algebraic Problems, second revised ed., Springer, Berlin, 2002.

[13] R. Hartmann, P. Houston, Symmetric interior penalty DG methods for the compressible Navier–Stokes equations I: method formulation, Int. J. Numer. Anal. Model. 3 (1) (2006) 1–20.

[14] C.M. Klaij, J.J.W. van der Vegt, H. van der Ven, A space–time discontinuous Galerkin discretization for the compressible Navier–Stokes equations, J. Comput. Phys. (in press).
[15] C.M. Klaij, M.H. van Raalte, J.J.W. van der Vegt, H. van der Ven, *hp*-multigrid for space–time discontinuous Galerkin discretizations of the compressible Navier–Stokes equations, J. Comput. Phys. (in preparation).
[16] W.L. Kleb, W.A. Wood, B. van Leer, Efficient multi-stage time marching for viscous flows via local preconditioning, AIAA J. 99-3267 (1999) 181–194.
[17] J. Kok, An industrially applicable solver for compressible, turbulent flows, Ph.D. Thesis, Delft University of Technology, Delft, The Netherlands, 1998.
[18] N.D. Melson, M.D. Sanetrik, H.L. Atkins, Time-accurate Navier–Stokes calculations with multigrid acceleration, in: Proceedings of the 6th Copper Mountain Conference on Multigrid Methods, 1993.
[19] A. Ostermann, Stability of W-methods with applications to operator splitting and to geometric theory, Appl. Numer. Math. 42 (2002) 353–366.
[20] PETSc package. Available from: <http://www-unix.mcs.anl.gov/petsc/petsc-2>.
[21] S.B. Pope, Turbulent Flows, Cambridge University Press, Cambridge, 2000.
[22] J.J. Sudirham, J.J.W. van der Vegt, R.M.J. van Damme, Space–time discontinuous Galerkin method for advection–diffusion problems. Application to wet-chemical etching processes, Appl. Numer. Math. (in press).
[23] J.J.W. van der Vegt, H. van der Ven, Space–time discontinuous Galerkin finite element method with dynamic grid motion for inviscid compressible flows. I. General formulation, J. Comput. Phys 182 (2002) 546–585.
[24] H. van der Ven, J.J.W. van der Vegt, Space–time discontinuous Galerkin finite element method with dynamic grid motion for inviscid compressible flows. II. Efficient flux quadrature, Comput. Meth. Appl. Mech. Eng. 191 (2002) 4747–4780.
[25] Pieter Wesseling, Principles of Computational Fluid Dynamics, Springer, Berlin, 2000.