Available online at www.sciencedirect.com
ScienceDirect
Journal of Computational Physics 227 (2007) 1024–1045
JOURNAL OF COMPUTATIONAL PHYSICS
www.elsevier.com/locate/jcp

h-Multigrid for space-time discontinuous Galerkin discretizations of the compressible Navier–Stokes equations

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

[a] *University of Twente, Department of Applied Mathematics, P.O. Box 217, 7500 AE Enschede, The Netherlands*
[b] *Centrum voor Wiskunde en Informatica, P.O. Box 94079, 1090 GB Amsterdam, The Netherlands*
[c] *National Aerospace Laboratory NLR, P.O. Box 90502, 1006 BM Amsterdam, The Netherlands*

Received 13 April 2007; received in revised form 8 August 2007; accepted 21 August 2007
Available online 14 September 2007

Abstract

Being implicit in time, the space-time discontinuous Galerkin discretization of the compressible Navier–Stokes equations requires the solution of a non-linear system of algebraic equations at each time-step. The overall performance, therefore, highly depends on the efficiency of the solver. In this article, we solve the system of algebraic equations with a *h*-multigrid method using explicit Runge–Kutta relaxation. Two-level Fourier analysis of this method for the scalar advection–diffusion equation shows convergence factors between 0.5 and 0.75. This motivates its application to the 3D compressible Navier–Stokes equations where numerical experiments show that the computational effort is significantly reduced, up to a factor 10 w.r.t. single-grid iterations.

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

Keywords: Space-time discontinuous Galerkin method; Pseudo-time stepping methods; Multigrid; Two-level fourier analysis

1. Introduction

Discontinuous Galerkin (DG) methods have grown very popular over recent years because of the relative ease with which the mesh and/or the polynomial order of the basis functions can be (locally) adapted. Contrary to continuous finite element methods, DG methods do not require any inter-element continuity, making such local *hp*-refinement less difficult. In this article, we consider the compressible Navier–Stokes equations discretized by a second order accurate DG method both in the spatial direction and in the time direction [17]. Although originally developed for hyperbolic equations, DG methods were successfully extended to (incompletely) parabolic equations, and the feasibility and benefits of DG methods for the compressible

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

0021-9991/$ - see front matter
doi:10.1016/j.jcp.2007.08.034

Navier–Stokes equations have been demonstrated by various authors including Bassi and Rebay [2,4], Bauman and Oden [6], Dolejší [9], Fidkowski et al. [10] and Hartmann and Houston [12].

Recently, attention shifted to the development of efficient solvers for the system of algebraic equations arising from DG discretizations, notably multigrid methods because of their expected optimal efficiency [21,26]. In the context of DG discretizations, multigrid methods with block iterative relaxation schemes were analyzed for model problems such as the Laplace equation and the advection–diffusion equation by Gopalakrishnan and Kantschatz in [11] and by Hemker, Hoffmann and Van Raalte in [13–15,25]. These multigrid methods use a sequence of meshes (h-multigrid) and are based on the embedding of function spaces associated with these meshes. On non-uniform grids where the embedding of spaces does not formally hold, an alternative is the approach followed by Fidkowski et al. [10], who keep the mesh fixed and use a sequence of different order polynomials (p-multigrid).

In [22], h- and p-multigrid were combined to solve the non-linear system of algebraic equations arising from the space-time DG discretization of the (hyperbolic) Euler equations: the discretization is second order on the fine grid and first order on the coarse grids. For the (incompletely parabolic) Navier–Stokes equations, however, this approach proves inadequate. Therefore, in this article, we consider h-multigrid with a second order DG discretization on all grid levels. As relaxation schemes for the multigrid algorithm, we use the explicit Runge–Kutta methods presented in [16]. With explicit relaxation the iterative solution process remains local and does not need large data storage, thus ensuring low computational costs.

In order to predict the multigrid behavior, we introduce a similar two-level local mode Fourier analysis as described in [13,24] for a model problem: the scalar advection–diffusion equation. Although we limit ourselves to a second order space-time discretization, the resulting analysis can be used for arbitrary polynomial basis and is directly extendable to higher-dimensional problems by the tensor product principle [15]. For various Courant numbers and cell Reynolds numbers, we compute multigrid convergence rates and we find that explicit Runge–Kutta smoothing is efficient for solving time-dependent advection–diffusion equations: two-level convergence factors between 0.5 and 0.75 are obtained. This motivates us to apply the h-multigrid method to the space-time DG discretization of the compressible Navier–Stokes equations. The performance of the h-multigrid method is then investigated using numerical experiments for laminar, (un)steady flow in 2D and 3D. We find that the computational effort is significantly reduced, up to a factor 10 w.r.t. single grid iteration.

The outline of this article is as follows. In Section 2, we summarize the space-time DG discretization [17] and discuss the Runge–Kutta methods [16] used for the pseudo-time integration of the non-linear system of algebraic equations. The multigrid algorithm is presented in Section 3 and studied in Section 4 with two-level Fourier analysis for a model problem. Appendix A contains the two-level analysis for the (deprecated) approach with constant basis functions on the coarse grid. The results for the compressible Navier–Stokes equations in 2D and 3D simulations are presented in Section 5. Conclusions are drawn in Section 6.

2. Summary of space-time DG method for the compressible Navier–Stokes equation

In this section, we first summarize the space-time discontinuous Galerkin method presented in [17]. The compressible Navier–Stokes equations are considered directly in the space-time domain; which implies that the basis-functions are discontinuous in space-time. The discretization results in a non-linear system of algebraic equations. Second, we summarize the pseudo-time integration with explicit Runge–Kutta methods [16]. These will serve as relaxation schemes in our multigrid method later on.

2.1. Space-time formulation

In [17], the compressible Navier–Stokes equations are directly considered in an open domain $\mathcal{E} \subset \mathbb{R}^4$. The Cartesian coordinates (x_0, x_1, x_2, x_3) of a point in this domain give the position $\bar{x} = (x_1, x_2, x_3)$ at time $t = x_0$. At time t, the flow domain $\Omega(t)$ is defined as $\Omega(t) \equiv \{\bar{x} \in \mathbb{R}^3 : (t, \bar{x}) \in \mathcal{E}\}$. Considering the time interval $t_0 < t < T$, the boundary $\partial\mathcal{E}$ of the space-time domain is given by the hypersurfaces $\Omega(t_0) \equiv \{x \in \partial\mathcal{E} : x_0 = t_0\}$, $\Omega(T) \equiv \{x \in \partial\mathcal{E} : x_0 = T\}$, and $\mathcal{Q} \equiv \{x \in \partial\mathcal{E} : t_0 < x_0 < T\}$. Using this notation, we can write the compressible Navier–Stokes equations as:

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

with the summation convention on repeated indices i, $r = 1, \ldots, 5$ and k, $s = 1, 2, 3$. The comma notation refers to partial differentiation to the indicated Cartesian coordinate direction. Here, $U \in \mathbb{R}^5$ is the vector of conservative variables and $F^{\mathrm{e}} \in \mathbb{R}^{5\times 3}$ the inviscid flux. The homogeneity tensor $A \in \mathbb{R}^{5\times 3\times 5\times 3}$ is defined as the derivative of the viscous flux $F^{\mathrm{v}} \in \mathbb{R}^{5\times 3}$ with respect to the gradient $\nabla U \in \mathbb{R}^{5\times 3}$ of the conservative variables:

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

and is given in [17]. The initial flow field is $U_0 \in \mathbb{R}^5$ and the boundary operator $\mathcal{B} \in \mathbb{R}^5$ depends on the internal data U and the prescribed boundary data U^{b}. The conservative variables, the inviscid flux and the viscous flux are given by:

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

with ρ the density, $\rho\vec{u}$ the momentum density vector, ρE the total energy density, p the pressure, δ the Kronecker delta function. The shear stresses τ are defined as: $\tau_{jk} = \lambda u_{i,i}\delta_{jk} + \mu(u_{j,k} + u_{k,j})$ with the viscosity coefficients μ and λ related through the Stokes hypothesis $3\lambda + 2\mu = 0$. The heat flux q is defined as: $q_k = -\kappa T_{,k}$. The system is closed with the following equations of state:

$$p = (\gamma - 1)\left(\rho E - \frac{1}{2}\rho u_i u_i\right), \quad T = \frac{1}{c_v}\left(E - \frac{1}{2}u_i u_i\right),$$

where $\gamma = c_p/c_v$ is the ratio of specific heats.

2.2. Discretization

The approximation $\Omega_h(t_n)$ of $\Omega(t_n)$ is divided into N_n non-overlapping hexahedral spatial elements $K^n_j = K_j(t_n)$. Each element K^n_j is related to the master element $\widehat{K} = (-1, 1)^3$ through the mapping F^n_K:

$$F^n_K : \widehat{K} \to K^n_j : \bar{\xi} \mapsto \bar{x} = \sum_{i=1}^{8} x_i(K^n_j)\chi_i(\bar{\xi}),$$

with x_i the spatial coordinates of the vertices of the hexahedron K^n_j and χ_i the usual tri-linear finite element shape functions for hexahedra. A similar approach is followed for $\Omega(t_{n+1})$. Linear interpolation in time connects $K_j(t_n)$ with $K_j(t_{n+1})$, defining the space-time elements $\mathcal{K}^n_j$ of $\mathcal{E}^n$. This is expressed by the mapping $G_{\mathcal{K}}$ from the master element $\widehat{\mathcal{K}} = (-1, 1)^4$ to the space-time element $\mathcal{K}^n_j$:

$$G^n_{\mathcal{K}} : \widehat{\mathcal{K}} \to \mathcal{K}^n_j : \xi \mapsto (t, \bar{x}) = \left(\frac{1}{2}(t_{n+1} + t_n) + \frac{1}{2}(t_{n+1} - t_n)\xi_0, \frac{1}{2}(1 - \xi_0)F^n_K(\bar{\xi}) + \frac{1}{2}(1 + \xi_0)F^{n+1}_K(\bar{\xi})\right). \tag{1}$$

The flow domain $\mathcal{E}$, limited to the time interval (t_n, t_{n+1}), defines a space-time slab which is divided into space-time elements $\mathcal{K}$ according to the tessellation $\mathcal{T}^n_h = \{\mathcal{K}\}$. We will need the corresponding function space:

$$W_h \equiv \{W \in (L^2(\mathcal{E}_h))^5 : W|_{\mathcal{K}} \circ G_{\mathcal{K}} \in (P^m(\widehat{\mathcal{K}}))^5, \quad \forall \mathcal{K} \in \mathcal{T}_h\},$$

where $P^m(\widehat{\mathcal{K}})$ denotes the space of polynomials of degree at most m on the master element $\widehat{\mathcal{K}} = (-1, 1)^4$, mapped to element $\mathcal{K} \in \mathcal{T}^n_h$ by $G_{\mathcal{K}}$. In a space-time slab, we distinguish a set $\mathcal{S}^n_{\mathrm{I}}$ of internal faces and a set $\mathcal{S}^n_{\mathrm{B}}$ of boundary faces. On an internal face $\mathcal{S} = \overline{\mathcal{K}}^{\mathrm{L}} \cap \overline{\mathcal{K}}^{\mathrm{R}}$, the traces from the left and right element are denoted by $(\cdot)^{\mathrm{L}}$ and $(\cdot)^{\mathrm{R}}$, respectively, and we define the average operator as $\{\!\{ \cdot \}\!\} = ((\cdot)^{\mathrm{L}} + (\cdot)^{\mathrm{R}})/2$ and the jump operator as $[\![\cdot]\!]_k = (\cdot)^{\mathrm{L}} n^{\mathrm{L}}_k + (\cdot)^{\mathrm{R}} n^{\mathrm{R}}_k$, with n the outward normal vector of the element.

Using this notation, the weak formulation of the compressible Navier–Stokes equations becomes: *find a* $U \in W_h$*, such that for all* $W \in W_h$*:*

$$
\begin{aligned}
&-\sum_{\mathcal{K}\in\mathcal{T}_h^n}\int_{\mathcal{K}}(W_{i,0}U_i + W_{i,k}(F_{ik}^{\mathrm{e}} - A_{ikrs}(U_{r,s}+\mathcal{R}_{ik})))\,\mathrm{d}\mathcal{K} + \sum_{K\in\mathcal{T}_h^n}\left(\int_{K(t_{n+1}^-)} W_i^{\mathrm{L}}U_i^{\mathrm{L}}\,\mathrm{d}K - \int_{K(t_n^+)} W_i^{\mathrm{L}}U_i^{\mathrm{R}}\,\mathrm{d}K\right)\\
&+\sum_{\mathcal{S}\in\mathcal{S}_{\mathrm{I}}^n}\int_{\mathcal{S}}(W_i^{\mathrm{L}} - W_i^{\mathrm{R}})H_i\,\mathrm{d}\mathcal{S} + \sum_{\mathcal{S}\in\mathcal{S}_{\mathrm{B}}^n}\int_{\mathcal{S}} W_i^{\mathrm{L}}H_i^{\mathrm{b}}\,\mathrm{d}\mathcal{S} - \sum_{\mathcal{S}\in\mathcal{S}_{\mathrm{I}}^n}\int_{\mathcal{S}}[\![W_i]\!]_k\{\!\{A_{ikrs}(U_{r,s}-\eta_{\mathcal{S}}\mathcal{R}_{ik}^{\mathcal{S}})\}\!\}\,\mathrm{d}\mathcal{S}\\
&-\sum_{\mathcal{S}\in\mathcal{S}_{\mathrm{B}}^n}\int_{\mathcal{S}} W_i^{\mathrm{L}}(A_{ikrs}^{\mathrm{b}}(U_{r,s}^{\mathrm{b}}-\eta_{\mathcal{S}}\mathcal{R}_{ik}^{\mathcal{S}}))\bar{n}_k^{\mathrm{L}}\,\mathrm{d}\mathcal{S} = 0. \qquad (2)
\end{aligned}
$$

The inviscid numerical flux $H \in \mathbb{R}^5$ is based on the HLLC approximate Riemann solver for moving meshes [22]. The stabilization parameter for the discretization of the viscous terms is denoted by $\eta_{\mathcal{S}}$ and the superscript b indicates dependence on the prescribed boundary data U^{b}. The weak form (2) is slightly different from the one presented in [17]: the homogeneity tensor A is no longer included in the definition of the lifting operators $\mathcal{R}$ and $\mathcal{R}^{\mathcal{S}}$. This makes its computation less expensive. The current weak form can be obtained by following the derivation in [17] with the auxiliary variables defined as $U_{r,s}$ instead of $A_{ikrs}U_{r,s}$. This viscous flux discretization is the generalization to the space-time context of the approach originally proposed by Bassi et al. [3,5] for $\eta = 1$ and analyzed for an elliptic model problem in [1,7]. A detailed theoretical analysis of the present space-time DG algorithm for the advection–diffusion equation, including an *hp*-error analysis, can be found in [20].

The definition of the local lifting operator $\mathcal{R}^{\mathcal{S}} \in \mathbb{R}^{5\times 3}$ requires the function space:

$$
V_h \equiv \left\{V \in (L^2(\mathcal{E}_h))^{5\times 3} : V|_{\mathcal{K}} \circ G_{\mathcal{K}} \in (P^m(\widehat{\mathcal{K}}))^{5\times 3}, \quad \forall \mathcal{K} \in \mathcal{T}_h\right\},
$$

such that $\nabla_h W_h \subset V_h$ with $(\nabla_h W_h)|_{\mathcal{K}} = \nabla(W_h|_{\mathcal{K}})$ the broken derivative. Then, the local lifting operator is defined as [17]: *find an* $\mathcal{R}^{\mathcal{S}} \in V_h$*, such that for all* $V \in V_h$*:*

$$
\sum_{\mathcal{K}\in\mathcal{T}_h^n}\int_{\mathcal{K}} V_{ik}\mathcal{R}_{ik}^{\mathcal{S}}\,\mathrm{d}\mathcal{K} = \begin{cases} \int_{\mathcal{S}}\{\!\{V_{ik}\}\!\}[\![U_i]\!]_k\,\mathrm{d}\mathcal{S} & \text{for } \mathcal{S}\in\mathcal{S}_{\mathrm{I}}^n,\\ \int_{\mathcal{S}} V_{ik}^{\mathrm{L}}(U_i^{\mathrm{L}} - U_i^{\mathrm{b}})\bar{n}_k\,\mathrm{d}\mathcal{S} & \text{for } \mathcal{S}\in\mathcal{S}_{\mathrm{B}}^n,\end{cases}
$$

and relates to the global lifting operator $\mathcal{R} \in \mathbb{R}^{5\times 3}$ through: $\mathcal{R} = \sum_{\mathcal{S}\in\mathcal{S}_{\mathrm{I}}^n\cup\mathcal{S}_{\mathrm{B}}^n}\mathcal{R}^{\mathcal{S}}$. The stabilization parameter $\eta_{\mathcal{S}}$ is constant and (at least) equal to the number of spatial faces of an element [7,20]: four in 2D and six in 3D for hexahedra. We refer to [17] and [22] for a more complete description of the space-time discretization.

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. In this paper, we limit ourselves to linear polynomials to represent the trial function U and the test function W in each element $\mathcal{K} \in \mathcal{T}_h^n$:

$$
U(t,\bar{x})|_{\mathcal{K}} = \widehat{U}_m\psi_m(t,\bar{x}), \qquad (3)
$$

$$
W(t,\bar{x})|_{\mathcal{K}} = \widehat{W}_l\psi_l(t,\bar{x}), \qquad (4)
$$

with $m, l = 0, \ldots, 4$. The expansion coefficients are denoted by $\widehat{(\cdot)}$ and the basis functions ψ are given by:

$$
\psi_m = \begin{cases} 1, & \text{for } m = 0\\ \phi_m(t,\bar{x}) - \frac{1}{|K_j(t_{n+1}^-)|}\int_{K_j(t_{n+1}^-)}\phi_m(t,\bar{x})\,\mathrm{d}K, & \text{for } m = 1,\ldots,4\end{cases}
$$

where the functions ϕ in an element $\mathcal{K}$ are related to the basis functions $\hat{\phi}$ on the master element $\widehat{\mathcal{K}}$ through the mapping (1) as $\phi_m = \hat{\phi}_m \circ G_{\mathcal{K}}^{-1}$ with $\hat{\phi}_m(\xi) = \xi_m$ for $m = 1, \ldots, 4$ and ξ the local coordinates in $\widehat{\mathcal{K}}$. This polynomial basis is of interest because of two reasons: the basis functions are chosen such that the test and trial functions can be split into an element mean $\overline{U}$ at $t = t_{n+1}$ and a fluctuating part $\widetilde{U}$ [22]:

$$
U(t,\bar{x}) = \overline{U} + \widetilde{U}(t,\bar{x}), \quad \forall (t,\bar{x}) \in \mathcal{K}
$$

with $\overline{U} = \widehat{U}_0$ and

$$\int_K \widetilde{U}(t,\bar{x})\,\mathrm{d}x = 0.$$

As a consequence the relation between DG and finite volume discretizations is exposed: the equations for the element mean in the space-time DG discretization are the same as those of a finite volume discretization. The second reason is that it suits the definition of the artificial dissipation operator used in [22] as an alternative for slope limiters to guarantee monotone solutions around discontinuities and sharp gradients.

2.3. Pseudo-time integration

For each physical time step the system of algebraic equations can be written as [17]:

$$\mathcal{L}(\widehat{U}^n; \widehat{U}^{n-1}) = 0.$$

This system is then solved using pseudo-time integration, i.e. we add a pseudo-time derivative:

$$|K^n|\frac{\partial \widehat{U}}{\partial \tau} = -\frac{1}{\Delta t}\mathcal{L}(\widehat{U}; \widehat{U}^{n-1}), \tag{5}$$

and iterate in pseudo-time τ to steady-state using explicit Runge–Kutta methods. Here, $\Delta t = t_{n+1} - t_n$ and $|K^n|$ is the diagonal matrix with entries $|K_j(t_{n+1})|$. At steady-state we have $\widehat{U}^n = \widehat{U}$.

In [16], we proposed a combination of two explicit Runge–Kutta schemes for the pseudo-time integration, one designed for the inviscid part of the flow domain, the other for the viscous part. Redefining $\mathcal{L}$ as $|K^n|^{-1}\mathcal{L}$, the scheme for the inviscid part is given by:

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

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

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

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

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 factor λ is the ratio between the pseudo-time step and the physical time step: $\lambda = \Delta\tau/\Delta t$. The scheme for the viscous part is given by:

Algorithm 2 (*EXV*). Explicit Runge–Kutta method for viscous flows.

(1) Initialize $\widehat{V}^0 = \widehat{U}$.
(2) For all stages $s = 1$ to 4 compute $\widehat{V}^s$ as:

$$\widehat{V}^s = \widehat{V}^0 - \alpha_s\lambda\mathcal{L}(\widehat{V}^{s-1}; \widehat{U}^{n-1}).$$

(3) Return $\widehat{U} = \widehat{V}^4$.

Here, the Runge–Kutta coefficients at stage s are defined as: $\alpha_1 = 0.0178571$, $\alpha_2 = 0.0568106$, $\alpha_3 = 0.174513$ and $\alpha_4 = 1.0$.

Since accuracy is not important in pseudo-time, we can apply local pseudo-time stepping and deploy whichever scheme gives the mildest stability constraint. The EXI scheme has the mildest stability constraint for relatively high cell Reynolds numbers and the EXV scheme for relatively low cell Reynolds numbers. For further details on the stability of both methods and the threshold between them we refer to [16].

Now that the discretization and the pseudo-time integration are set, we are ready to introduce the h-multigrid method for space-time DG discretizations.

3. h-Multigrid method

In this section, we present the h-multigrid pseudo-time integration method for solving the non-linear system of algebraic equations arising from the space-time discretization of the compressible Navier–Stokes equations.

3.1. Two-level algorithm

At the core of any multigrid method is the two-level algorithm, which we consider first. Let the subscripts $(\cdot)_h$ and $(\cdot)_H$ denote a quantity $(\cdot)$ on the fine and coarse grid, respectively. Let $\widehat{U}$ denote an approximation of the solution $\widehat{U}^n$ of (5). Let R denote the restriction operator for the solution, $\overline{R}$ the restriction operator for the residuals and P the prolongation operator, to be defined later on. The two-level algorithm which iterates system (5) to steady-state in pseudo-time can be written as:

Algorithm 3 (TLA). Two-level algorithm.

(1) Take one pseudo-time step on the fine grid with the combined EXI and EXV methods, this gives the approximation $\widehat{U}_h$.
(2) Restrict this approximation to the coarse grid: $\widehat{U}_H = R(\widehat{U}_h)$.
(3) Compute the forcing:

$$F_H \equiv \mathcal{L}(\widehat{U}_H; \widehat{U}_H^{n-1}) - \overline{R}(\mathcal{L}(\widehat{U}_h; \widehat{U}_h^{n-1})).$$

(4) Solve the coarse grid problem for the unknown $\widehat{U}_H^*$:

$$\mathcal{L}(\widehat{U}_H^*; \widehat{U}_H^{n-1}) - F_H = 0,$$

(5) Compute the coarse grid error $E_H = \widehat{U}_H^* - \widehat{U}_H$ and correct the fine grid approximation: $\widehat{U}_h \leftarrow \widehat{U}_h + P(E_H)$.

Here, the value $\widehat{U}^{n-1}$ is retained from the previous physical time step (see (5)); for the first iteration ($n = 1$) it is obtained from the initial condition U_0. When converged in pseudo-time, this algorithm yields $\widehat{U}^n$ and the next physical time step can be taken.

Solving the coarse grid problem at stage four of Algorithm 3 can again be done with the two-level algorithm. This recursively defines the V-cycle multi-level algorithm in terms of the two-level algorithm. It is common practice to take ν_1 pseudo-time steps at stage one of Algorithm 3 and another ν_2 pseudo-time steps after stage five. In that case, ν_1 and ν_2 are called the number of pre- and post-relaxations, respectively. In practice, the exact solution of the problem on the coarsest grid is not always feasible; instead one simply takes $\nu_1 + \nu_2$ relaxation steps.

Next, we define the inter-grid transfer operators R, $\overline{R}$ and P.

3.2. Inter-grid transfer operators

The inter-grid transfer operators stem from the L_2-projection of the coarse grid solution U_H in an element $\mathcal{K}_H$ on the corresponding set of fine elements $\{\mathcal{K}_h\}$. The solution U_h in element $\mathcal{K}_h$ can be found by solving:

$$\int_{\mathcal{K}_h} W_i U_i^h \,\mathrm{d}\mathcal{K} = \int_{\mathcal{K}_h} W_i U_i^H \,\mathrm{d}\mathcal{K}, \quad \forall W \in W_h. \tag{6}$$

This relation supposes the embedding of spaces, i.e. $W_H \subset W_h$, to ensure that U_H is defined on $\mathcal{K}_h$. As illustrated in Fig. 1, the embedding of spaces does not hold for curvilinear grids with iso-parametric mapping: the fine elements overlap only partially with the coarse element. However, in order to construct the inter-grid transfer operators we will assume that the integral on the r.h.s. of (6) exists and proceed as follows. Replacing the test and trial functions in (6) by their polynomial expansions (3) gives:

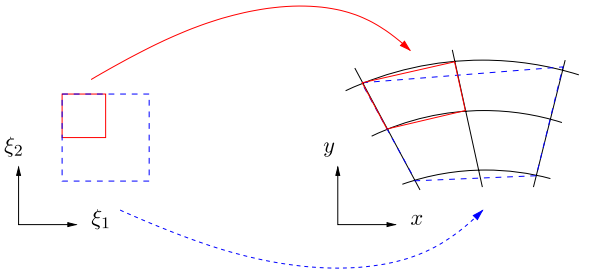


Fig. 1. Relation between two grid levels in computational and physical space.

$$\left(\int_{\mathcal{K}_h} \psi_l^h \psi_m^h \,\mathrm{d}\mathcal{K}\right) \widehat{U}_{im}^h = \left(\int_{\mathcal{K}_h} \psi_l^h \psi_n^H \,\mathrm{d}\mathcal{K}\right) \widehat{U}_{in}^H,$$

with $l, m, n = 0, \ldots, 4$. On the l.h.s., we recognize the mass matrix M_h of element $\mathcal{K}_h$ so the projection of U_H onto $\mathcal{K}_h$ can be computed in terms of the expansion coefficients as:

$$\widehat{U}_{im}^h = (M_h^{-1})_{ml} \left(\int_{\mathcal{K}_h} \psi_l^h \psi_n^H \,\mathrm{d}\mathcal{K}\right) \widehat{U}_{in}^H. \tag{7}$$

Consider the integral on the r.h.s. of (7) and transform to computational space using the mapping $G_{\mathcal{K}}$:

$$\int_{\mathcal{K}_h} \psi_l^h \psi_n^H \,\mathrm{d}\mathcal{K} = \int_{\widehat{\mathcal{K}}_h} \widehat{\psi}_l^h \widehat{\psi}_n^H |J_G| \,\mathrm{d}\xi,$$

with J_G the Jacobian determinant of the transformation (1). Note that in computational space we have:

$$\xi_i^h = \frac{1}{2}\xi_i^H \pm \frac{1}{2}, \quad i = 1, \ldots, 4$$

where the sign depends on the position of the fine element within the coarse element. In Fig. 1 for example, $\xi_1^h = \frac{1}{2}\xi_1^H - \frac{1}{2}$ and $\xi_2^h = \frac{1}{2}\xi_2^H + \frac{1}{2}$. Since the basis functions $\hat{\psi}$ are expressed in terms of ξ (Section 2.2), the integral (7) can easily be computed and thereby the L_2 projection of the coarse grid solution onto the fine grid. This defines the prolongation operator P. The restriction operator for the residuals is then defined as the transpose of the prolongation operator: $\overline{R} = P^{\mathrm{T}}$. The restriction operator R for the solution is defined as $R = P^{-1}$ such that the property $U_H = R(P(U_H))$ holds, meaning that the inter-grid transfer does not modify the solution.

Remark 1. In this article, we limit ourselves to h-multigrid in a single space-time slab. The time-step Δt is equal on both levels; the coefficients $\widehat{U}_{i4}$ which correspond to the gradient in time are therefore identical: $\widehat{U}_{i4}^H = \widehat{U}_{i4}^h$. Multi-time multigrid methods are also feasible, see [23], but not considered in this article.

Now that the h-multigrid method is well defined, we continue by analyzing its stability and performance.

4. Two-level fourier analysis for a model problem

In this section, the convergence behavior of Algorithm 3 is studied with Fourier analysis for the space-time discontinuous Galerkin discretization of the scalar advection–diffusion equation.

4.1. Discretization of the model problem

Consider the time-dependent scalar advection–diffusion equation:

$$\begin{cases} u_t + a u_x - d u_{xx} = 0, & x \in \mathbb{R}, \quad t \in \mathbb{R}^+ \\ u(x,0) = u_0, & x \in \mathbb{R}, \end{cases}$$

where $a, d > 0$ denote the advection and diffusion constants, respectively. The flow domain $\mathcal{E}_h = \mathbb{R} \times \mathbb{R}^+$ restricted to the time interval (t^n, t^{n+1}) has a tessellation $\mathcal{T}_h^n$ consisting of uniform elements $\mathcal{K} = (x_j, x_{j+1}) \times (t^n, t^{n+1})$ with $j \in \mathbb{Z}$ and $n \in \mathbb{N}$. The corresponding functions space is:

$$w_h \equiv \left\{ w \in L^2(\mathcal{E}_h) : w|_{\mathcal{K}} \circ G_{\mathcal{K}} \in P^m(\widehat{\mathcal{K}}), \quad \forall \mathcal{K} \in \mathcal{T}_h^n \right\}.$$

The weak form becomes the following. Find a $u \in w_h$, such that for all $w \in w_h$:

$$-\sum_{\mathcal{K}\in\mathcal{T}_h^n}\int_{\mathcal{K}}(w_t u + w_x(au - d(u_x - \mathcal{R})))\,\mathrm{d}\mathcal{K} + \sum_{K\in\mathcal{T}_h^n}\left(\int_{K(t_{n+1}^-)} w^{\mathrm{L}}u^{\mathrm{L}}\,\mathrm{d}K - \int_{K(t_n^+)} w^{\mathrm{L}}u^{\mathrm{R}}\,\mathrm{d}K\right)$$
$$+\sum_{\mathcal{S}\in\mathcal{S}_{\mathrm{I}}^n}\int_{\mathcal{S}}[\![w]\!](a\hat{u} - d\{\!\{u_x - \eta\mathcal{R}^{\mathcal{S}}\}\!\})\mathrm{d}\mathcal{S} = 0,$$

where standard upwinding is used for the numerical flux $a\hat{u}$. The definition of the local lifting operator $\mathcal{R}^{\mathcal{S}} \in \mathbb{R}$ now becomes:

$$\sum_{\mathcal{K}\in\mathcal{T}_h^n}\int_{\mathcal{K}} w\mathcal{R}^{\mathcal{S}}\,\mathrm{d}\mathcal{K} = \int_{\mathcal{S}}\{\!\{w\}\!\}[\![u]\!]\,\mathrm{d}\mathcal{S} \quad \text{for } \mathcal{S}\in\mathcal{S}_{\mathrm{I}}^n,$$

with global lifting operator $\mathcal{R} = \sum_{\mathcal{S}\in\mathcal{S}_{\mathrm{I}}^n}\mathcal{R}^{\mathcal{S}}$. For stability of the discretization we take $\eta = 2$. The mapping (1) from the master element $\widehat{\mathcal{K}} = (-1,1)^2$ to the element $\mathcal{K}$ reduces to:

$$G_{\mathcal{K}} : \widehat{\mathcal{K}} \to \mathcal{K} : (\xi_1,\xi_2) \mapsto (x,t) = \left(\frac{1}{2}(x_{j+1}+x_j) + \frac{1}{2}(x_{j+1}-x_j)\xi_1, \frac{1}{2}(t_{n+1}+t_n) + \frac{1}{2}(t_{n+1}-t_n)\xi_2\right),$$

and the linear basis functions are $\psi = \hat{\psi}\circ G_{\mathcal{K}}^{-1}$ with $\hat{\psi}_0 = 1$, $\hat{\psi}_1 = \xi_1$ and $\hat{\psi}_2 = \xi_2 - 1$. The polynomial expansions (3) and (4) of the trial and test functions now read:

$$u(t,x)|_{\mathcal{K}} = \hat{u}_m\psi_m(t,x), \quad w(t,x)|_{\mathcal{K}} = \widehat{w}_l\psi_l(t,x),$$

with $l, m = 0, 1, 2$.

Replacing u and w in the weak form by these expansions yields a discrete system[1] for the vector of expansion coefficients $\hat{u}$ of u at time level n:

$$\mathcal{L}_h(\hat{u}^n;\hat{u}^{n-1}) \equiv (\mathcal{L}_h^{\mathrm{a}} + \mathcal{L}_h^{\mathrm{d}})\hat{u}^n + \mathcal{L}_h^t\hat{u}^{n-1} = 0,$$

with $h = x_{j+1} - x_j$. This $3\mathbb{Z}\times 3\mathbb{Z}$ system has a block Toeplitz structure with 3×3 blocks and its stencil has the form:

$$\mathcal{L}_h \cong [L_h \mid D_h \mid U_h], \tag{8}$$

where L_h represents the left block, D_h the diagonal block and U_h the right block. The advective part $\mathcal{L}_h^{\mathrm{a}}$ of the discretization depends on the Courant number

$$\sigma = \frac{a\Delta t}{h}, \tag{9}$$

and gives the following block tridiagonal contribution to the system:

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

The right block is zero because the advective numerical flux is upwind ($a > 0$). The diffusive part $\mathcal{L}_h^{\mathrm{d}}$ of the discretization depends on the Courant number, the stabilization constant η and the cell Reynolds number:

$$Re_h = \frac{ah}{d}, \tag{10}$$

[1] The scaling with $|K^n|^{-1}$ from Section 2.3 amounts to a division by h which yields the expression for $\mathcal{L}_h$ presented here.

and also gives a block tridiagonal contribution to the system:

$$\mathcal{L}_h^{\mathrm{d}} \cong \frac{\sigma}{Re_h}\left[\begin{array}{ccc|ccc|ccc} -2\eta & 1-2\eta & 2\eta & 4\eta & 0 & -4\eta & -2\eta & -1+2\eta & 2\eta \\ -1+2\eta & -2+2\eta & 1-2\eta & 0 & 4\eta & 0 & 1-2\eta & -2+2\eta & -1+2\eta \\ 2\eta & -1+2\eta & -\frac{13}{6}\eta & -4\eta & 0 & \frac{13}{3}\eta & 2\eta & 1-2\eta & -\frac{13}{6}\eta \end{array}\right].$$

The contribution $\mathcal{L}_h^t$ related to the previous space-time slab is block diagonal:

$$\mathcal{L}_h^t \cong \left[\begin{array}{ccc|ccc|ccc} 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -\frac{1}{3} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 2 & 0 & 0 & 0 & 0 & 0 \end{array}\right].$$

The linear system associated with the space-time DG discretization of the scalar advection–diffusion equation must be solved for each time slab. We do so with the multigrid method presented in Section 3.

4.2. The two-level algorithm

At the core of any multigrid method is the two-level algorithm. Multilevel methods are obtained by recursively applying the two-level algorithm in, for example, a V-cycle. Therefore, we study the error amplification operator of the two-level algorithm M_h^{TLA}, which is given by [13,24]:

$$M_h^{\mathrm{TLA}} = M_h^{\mathrm{CGC}} M_h^{\mathrm{REL}},$$

with M_h^{REL} the error amplification operator associated with either the EXI or the EXV scheme presented in Section 2. The explicit form of these operators is obtained from their recursive definitions (Algorithm 1 and 2) and reads:

$$M_h^{\mathrm{EXI}} = \frac{I}{1+\alpha_5\lambda} + \frac{\alpha_5\lambda(I-\mathcal{L}_h)}{(1+\alpha_4\lambda)(1+\alpha_5\lambda)} + \cdots + \frac{\alpha_2\alpha_3\cdots\alpha_5(\lambda(I-\mathcal{L}_h))^4}{(1+\alpha_2\lambda)(\cdots)(1+\alpha_5\lambda)} + \frac{\alpha_1\alpha_2\cdots\alpha_5(\lambda(I-\mathcal{L}_h))^5}{(1+\alpha_1\lambda)(\cdots)(1+\alpha_5\lambda)}. \tag{11}$$

and

$$M_h^{\mathrm{EXV}} = I - \alpha_4\lambda\mathcal{L}_h + \alpha_3\alpha_4(\lambda\mathcal{L}_h)^2 - \cdots + \alpha_1\alpha_2\alpha_3\alpha_4(\lambda\mathcal{L}_h)^4, \tag{12}$$

with I the identity matrix. The coarse grid correction (CGC) of Algorithm 3 is given by:

$$M^{\mathrm{CGC}} = I - P\mathcal{L}_H^{-1}\overline{R}\mathcal{L}_h.$$

On the uniform grid, the prolongation operator defined in Section 3.2 becomes:

$$P \cong \left[\begin{array}{ccc|ccc|ccc} 1 & \frac{1}{2} & 0 & 1 & -\frac{1}{2} & 0 & 0 & 0 & 0 \\ 0 & \frac{1}{2} & 0 & 0 & \frac{1}{2} & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \end{array}\right].$$

The right block is zero because the coarse grid element $\mathcal{K}_H^n = (x_{j-1}, x_{j+1}) \times (t^n, t^{n+1})$ corresponds to the fine elements $(\mathcal{K}_h^n)_L = (x_{j-1}, x_j) \times (t^n, t^{n+1})$ and $(\mathcal{K}_h^n)_D = (x_j, x_{j+1}) \times (t^n, t^{n+1})$. This choice is arbitrary considering the infinite domain. Note the block Toeplitz structure with 3×3 blocks; since $\overline{R} = P^{\mathrm{T}}$ the restriction operator for the residuals is also block Toeplitz.

Remark 2. The parameter η has a significant effect on the stability of the Runge–Kutta methods: as η increases, the permissible pseudo-timestep decreases proportionally. Therefore η should be taken as small as allowed in the discontinuous Galerkin discretization, in general equal to the number of faces of an element [7,20].

The convergence behaviour of the two-level algorithm for the space-time DG discretization is given by the spectral radius of the error amplification operator, i.e. $\rho(M_h^{\mathrm{TLA}})$, which represents the expected convergence factor per iteration. In the following section, we will apply Fourier analysis to compute the eigenvalue spectra of the two-level algorithm.

4.3. Fourier analysis of the two-level algorithm

As shown in [24,25], the eigenvalue spectra of the two-level algorithm is $\{\lambda_i(\omega)\}$ with $i = 1, \ldots, 6$ and $\lambda_i(\omega)$ the eigenvalues of the Fourier transform $\widehat{M_h^{\mathrm{TLA}}}$ for $\omega \in [-\pi/H,\pi/H)$, . The Fourier[2] transform $\widehat{\mathcal{L}_h}$ of the block Toeplitz operator $\mathcal{L}_h$ with stencil (8) for a frequency ω is:

$$\widehat{\mathcal{L}_h}(\omega) = L_h \mathrm{e}^{-\mathrm{i}\omega h} + D_h + U_h \mathrm{e}^{+\mathrm{i}\omega h},$$

with $\mathrm{i} = \sqrt{-1}$. Since the operators M_h^{REL}, P and $\overline{R}$ are also block Toeplitz, their Fourier transforms $\widehat{P}$ and $\widehat{\overline{R}}$ are computed similarly. The Fourier transform of the two-level error amplification operator is then given by [24,25]:

$$\widehat{M_h^{\mathrm{TLA}}}(\omega) = \begin{bmatrix} I_h & 0 \\ 0 & I_h \end{bmatrix} - \begin{bmatrix} \widehat{P}(\omega) \\ \widehat{P}(\omega + \pi/h) \end{bmatrix} \left[\widehat{\mathcal{L}_H}(\omega)^{-1} \right] \begin{bmatrix} \widehat{\overline{R}}(\omega) & \widehat{\overline{R}}(\omega + \pi/h) \end{bmatrix} \begin{bmatrix} \widehat{\mathcal{L}_h}(\omega) & 0 \\ 0 & \widehat{\mathcal{L}_h}(\omega + \pi/h) \end{bmatrix} \times \begin{bmatrix} \widehat{M_h^{\mathrm{REL}}}(\omega) & 0 \\ 0 & \widehat{M_h^{\mathrm{REL}}}(\omega + \pi/h) \end{bmatrix},$$

with I_h the 3×3 identity matrix. Here, $\omega \in [-\pi/H,\pi/H)$ corresponds to the low frequencies and $\omega + \pi/h$ to the associated high frequencies. The Fourier transforms of the error amplification operators (11) and (12) are:

$$\widehat{M_h^{\mathrm{EXI}}}(\omega) = \frac{I_h}{1 + \alpha_5\lambda} + \frac{\alpha_5\lambda(I_h - \widehat{\mathcal{L}_h}(\omega))}{(1 + \alpha_4\lambda)(1 + \alpha_5\lambda)} + \cdots + \frac{\alpha_2\alpha_3 \cdots \alpha_5(\lambda(I_h - \widehat{\mathcal{L}_h}(\omega)))^4}{(1 + \alpha_2\lambda)(\cdots)(1 + \alpha_5\lambda)} + \frac{\alpha_1\alpha_2 \cdots \alpha_5(\lambda(I_h - \widehat{\mathcal{L}_h}(\omega)))^5}{(1 + \alpha_1\lambda)(\cdots)(1 + \alpha_5\lambda)},$$

and

$$\widehat{M_h^{\mathrm{EXV}}}(\omega) = I_h - \alpha_4\lambda\widehat{\mathcal{L}_h}(\omega) + \alpha_3\alpha_4(\lambda\widehat{\mathcal{L}_h}(\omega))^2 - \cdots + \alpha_1\alpha_2\alpha_3\alpha_4(\lambda\widehat{\mathcal{L}_h}(\omega))^4.$$

Now, the eigenvalue spectra and radii of the two-level algorithm can be computed, depending on the Courant number (9) and cell Reynolds number (10) occuring in $\mathcal{L}_h$. The Courant number expresses the time-accuracy of the discretization and the cell Reynolds number the importance of diffusion relative to advection. Since the space-time DG discretization is *implicit* in physical time, the method is unconditionally stable [20] for any physical time step. This allows us to take the Courant number $\sigma = 100$ for steady-state cases and $\sigma = 1$ for time-dependent cases. We will further consider cell Reynolds numbers between $Re_h = 0.01$ and $Re_h = 100$, which represent the diffusion and advection dominated cases, respectively. The Runge–Kutta methods are explicit in pseudo time and their stability depends on the ratio λ between the pseudo timestep and the physical timestep $\lambda = \Delta\tau/\Delta t$. It is often convenient to express the stability condition in terms of the pseudo-time CFL number $\sigma_{\Delta\tau}$ and the pseudo-time diffusive Von Neumann condition $\delta_{\Delta\tau}$:

$$\Delta\tau \leqslant \Delta\tau^{\mathrm{a}} \equiv \frac{\sigma_{\Delta\tau} h}{a} \quad \text{and} \quad \Delta\tau \leqslant \Delta\tau^{\mathrm{d}} \equiv \frac{\delta_{\Delta\tau} h^2}{d}.$$

The pseudo-time CFL number is given by $\sigma_{\Delta\tau} = \lambda\sigma$ and the pseudo-time diffusive Von Neumann number by $\delta_{\Delta\tau} = \lambda\sigma/Re_h$.

In Table 1, the spectral radii of the two-level algorithm with EXI smoother for steady cases ($\sigma = 100$) are given for various values of Re_h. We see that as Re_h decreases, the spectral radius of the two-level algorithm increases; in other words as diffusion becomes more important, the rate of convergence deteriorates. This is to be expected as the EXI method was optimized for inviscid cases. The EXV method, on the other hand, was optimized for diffusion dominated cases and in Table 2, we see that the two-level algorithm with EXV smoother maintains good convergence factors in the diffusion dominated cases. For unsteady cases ($\sigma = 1$),

[2] In this section, the $\widehat{(\cdot)}$ notation indicates Fourier transform of a block Toeplitz operator, not to be confused with the expansion coefficients $\hat{u}$ or the basis functions $\hat{\psi}$ on the master element $\hat{\mathcal{K}}$.

Table 1
Spectral radii of the two-level algorithm with the EXI smoother for steady cases ($\sigma = 100$)

Re_h	$\Delta\tau/\Delta t$	$\rho(M_h^{\mathrm{EXI}})$	$\rho(M_h^{\mathrm{TLA}})$
100	1.8e−02	0.991	0.622
10	8.0e−03	0.996	0.716
1	1.4e−03	0.999	0.906
0.1	1.6e−04	0.999	0.932
0.01	1.6e−05	0.999	0.935

The TLA convergence with EXI smoothing is better than with EXV smoothing (Table 2) for $Re_h = 100$ and 10.

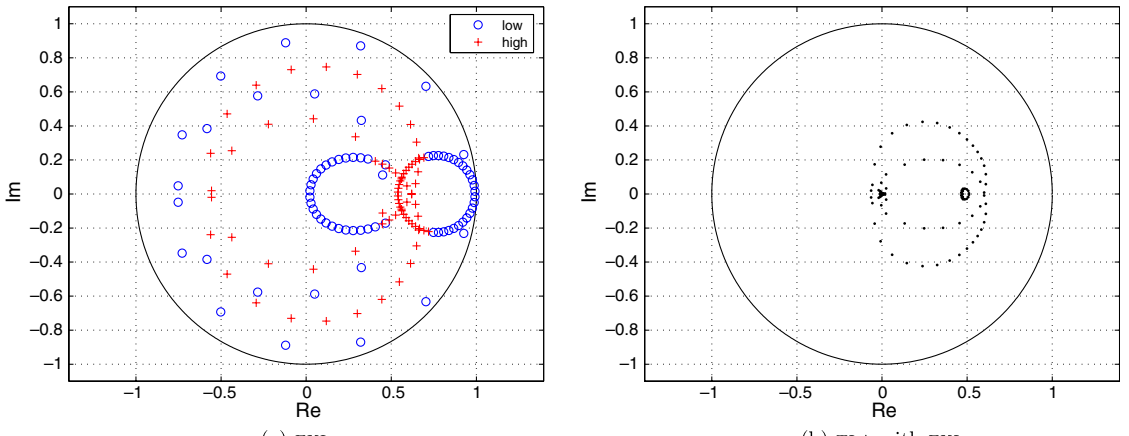
Table 2
Spectral radii of the two-level algorithm with the EXV smoother for steady cases ($\sigma = 100$)

Re_h	$\Delta\tau/\Delta t$	$\rho(M_h^{\mathrm{EXV}})$	$\rho(M_h^{\mathrm{TLA}})$
100	2.0e−03	0.999	0.914
10	3.0e−03	0.998	0.871
1	7.0e−03	0.996	0.697
0.1	8.0e−04	0.999	0.753
0.01	8.0e−05	0.999	0.744

The TLA convergence with EXV smoothing is better than with EXI smoothing (Table 1) for $Re_h = 1$, 0.1 and 0.01.

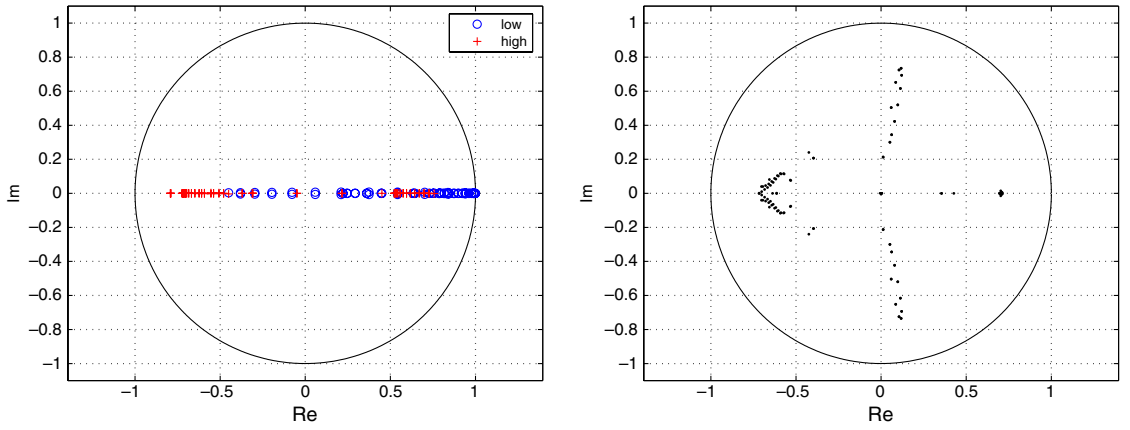
the values are different (see Tables 3 and 4) but the trend is similar: EXI smoothing is preferable for advection dominated cases ($Re_h > 1$) and EXV smoothing for diffusion dominated cases ($Re_h \leqslant 1$). This allows us to choose the optimal scheme depending on the cell Reynolds number.

In Figs. 2–5, we show the eigenvalue spectra of the preferable smoother and of the two-level algorithm for steady and unsteady, advection and diffusion dominated cases. We have plotted the eigenvalues corresponding to a discrete series of low frequencies $\omega_i = -\pi/H, -0.96\pi/H, \ldots, \pi/H$ and associated high frequencies $\omega_i + \pi/h$. For the smoothers, the eigenvalues corresponding to low frequencies are denoted by $\circ$; those corresponding to high frequencies by $+$. In the eigenvalue spectra of two-level algorithms we do not distinguish between low and high frequencies: the two-level algorithm must damp all frequencies. From these figures we see that the Runge–Kutta methods have the smoothing property, i.e. the high frequencies are damped. The observed smoothing factor of approximately *0.8* (which is often used as an estimate for multigrid convergence [8]) is rather inaccurate in comparison to the true smoothing factor obtained with two-level analysis (see Tables 1–4).



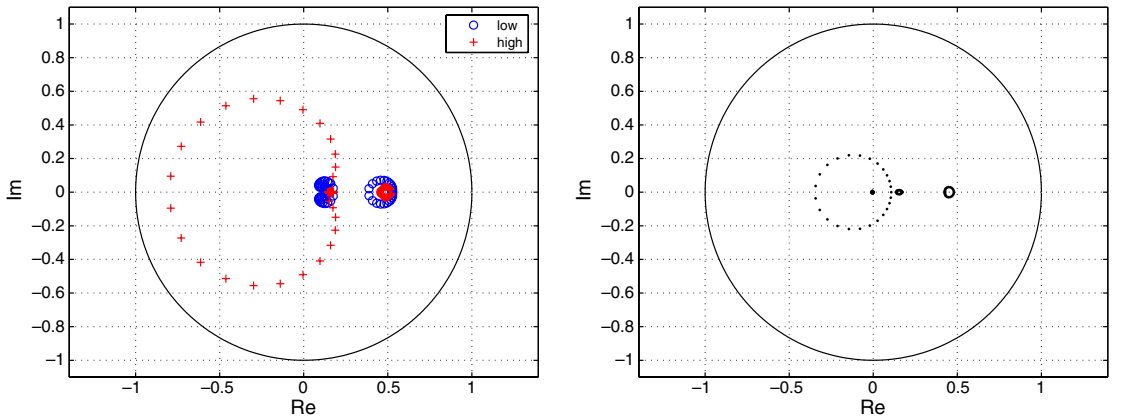
(a) EXI (b) TLA with EXI

Fig. 2. Eigenvalue spectra of the EXI smoother and two-level algorithm in the steady advection dominated case ($\sigma = 100$ and $Re_h = 100$, first row of Table 1).



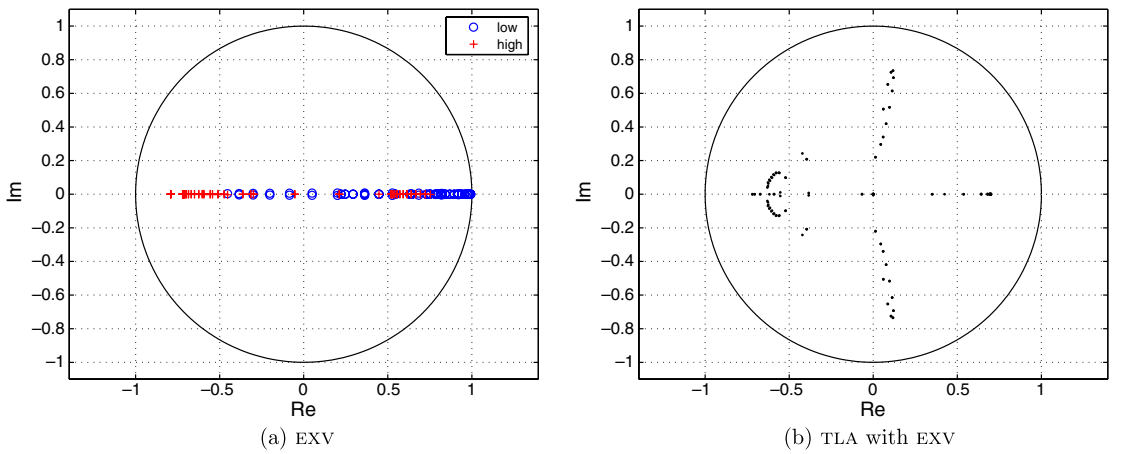
(a) EXV (b) TLA with EXV

Fig. 3. Eigenvalue spectra of the EXV smoother and two-level algorithm in the steady diffusion dominated case ($\sigma = 100$ and $Re_h = 0.01$, last row of Table 2).



(a) EXI (b) TLA with EXI

Fig. 4. Eigenvalue spectra of the EXI smoother and two-level algorithm in the unsteady advection dominated case ($\sigma = 1$ and $Re_h = 100$, first row of Table 3).



(a) EXV (b) TLA with EXV

Fig. 5. Eigenvalue spectra of the EXV smoother and two-level algorithm in the unsteady diffusion dominated case ($\sigma = 1$ and $Re_h = 0.01$, last row of Table 4).

Table 3
Spectral radii of the two-level algorithm with the EXI smoother for unsteady cases ($\sigma = 1$)

Re_h	$\Delta\tau/\Delta t$	$\rho(M_h^{\mathrm{EXI}})$	$\rho(M_h^{\mathrm{TLA}})$
100	1.6e−00	0.796	0.479
10	8.0e−01	0.918	0.599
1	1.4e−01	0.904	0.837
0.1	1.6e−02	0.987	0.923
0.01	1.6e−03	0.998	0.934

The TLA convergence with EXI smoothing is better than with EXV smoothing (Table 4) for $Re_h = 100$ and 10.

Table 4
Spectral radii of the two-level algorithm with the EXV smoother for unsteady cases ($\sigma = 1$)

Re_h	$\Delta\tau/\Delta t$	$\rho(M_h^{\mathrm{EXV}})$	$\rho(M_h^{\mathrm{TLA}})$
100	1.0e−00	0.924	0.660
10	7.0e−01	0.812	0.704
1	7.0e−01	0.805	0.719
0.1	8.0e−02	0.936	0.755
0.01	8.0e−03	0.993	0.744

The TLA convergence with EXV smoothing is better than with EXI smoothing (Table 3) for $Re_h = 1$, 0.1 and 0.01.

The two-level analysis of h-multigrid iteration for the space-time DG discretization of the advection–diffusion equation shows significantly improved convergence factors w.r.t. single-grid iteration. Multigrid convergence factors range between 0.5 and 0.75, whereas single-grid convergence factors range between 0.8 and 0.99, depending on the case.[3] This motivates the application of multigrid to the compressible Navier–Stokes equations in Section 5.

4.4. Numerical illustration

To illustrate the results of the multigrid analysis, we consider the space-time discretization of the scalar advection–diffusion equation for the following simple initial boundary value problem:

$$\begin{cases} u_t + a u_x = d u_{xx}, & x \in (0,1), \quad t \in \mathbb{R}^+, \\ u(0,t) = 1, \quad u(1,t) = 0, & t \in \mathbb{R}^+, \\ u(x,0) = 1 - x, & x \in (0,1). \end{cases}$$

The exact (steady state) solution is given by:

$$u(x) = \frac{\mathrm{e}^{a/d} - \mathrm{e}^{ax/d}}{\mathrm{e}^{a/d} - 1},$$

and features an exponential boundary layer near $x = 1$. Such a case is best solved on a so-called Shishkin mesh [19]. With N elements, this mesh is piecewise equidistant with nodes x_j given by:

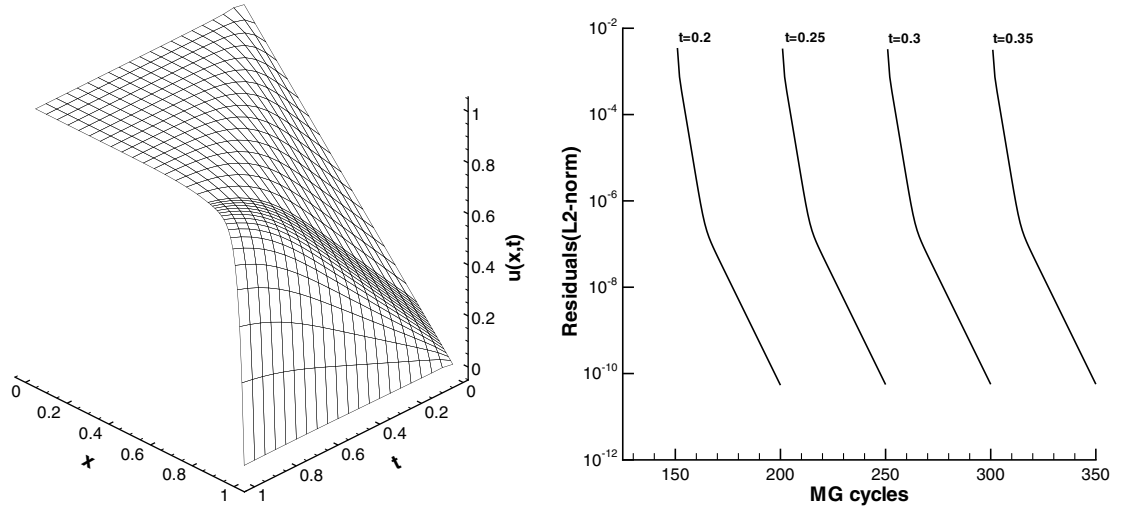
$$x_j = \begin{cases} 2(1-c)j/N & \text{for } j = 0, 1, \ldots, N/2, \\ 1 - c + 2c/N(j - N/2) & \text{for } j = N/2, N/2+1, \ldots, N, \end{cases}$$

where $c = (2/a)d\ln(N)$. For our example, we take $a = 1$, $d = 0.025$ and $N = 32$. Advection dominates in the first part of the mesh, so we use the EXI scheme there and the EXV scheme in the second part. We use three level multigrid in a V-cycle with two pre- and post-relaxations. The coarse grid problem is solved approxi-

[3] Note that the single-grid convergence factors for time-dependent, advection dominated cases, being around 0.8, are already quite good.

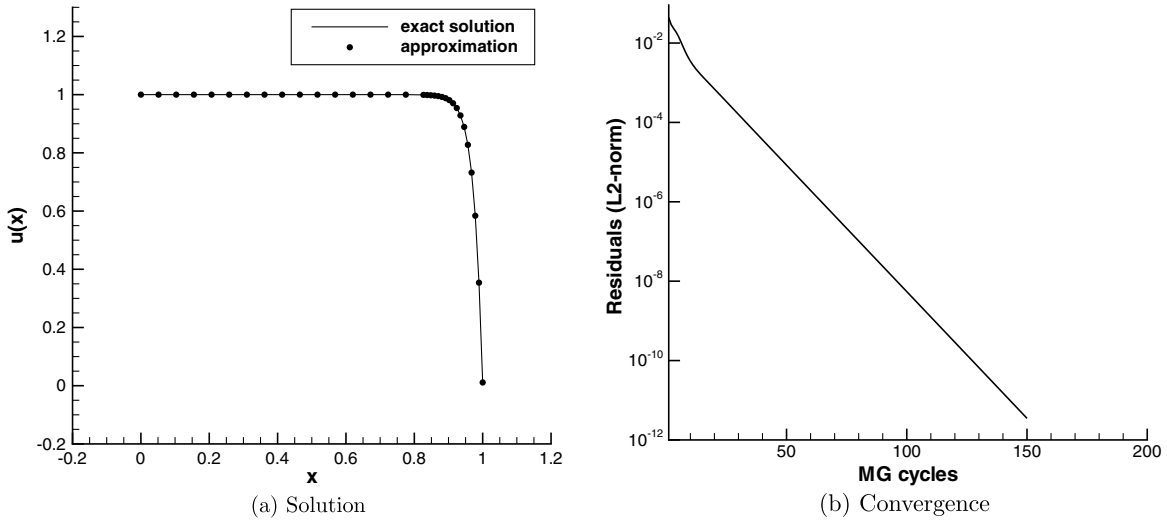
mately with four relaxations, which is more realistic in view of applications to complex problems where the exact coarse grid solution cannot be attained.

The problem can be solved in two ways: time accurate with $\Delta t = 0.05$ which corresponds to $C_{\Delta t} \approx \mathcal{O}(1)$ or directly as a steady-state problem with $\Delta t = 5$ which corresponds to $C_{\Delta t} \approx \mathcal{O}(100)$. In Fig. 6, the space-time solution and the convergence in pseudo-time for a few physical time steps are shown. With eight orders of convergence in 50 cycles, an effective damping factor of 0.7 is achieved. In Fig. 7, the steady-state solution is shown. With a single time step the convergence in pseudo-time is 10 orders in 150 cycles which corresponds to a damping factor of 0.85. Despite the presence of boundary conditions and the inaccurate solution of the coarse grid problem, these convergence rates are in agreement with the rate obtained from the analysis.



(a) Solution (b) Convergence

Fig. 6. The space-time solution of the advection–diffusion equation ($a = 1$, $d = 0.025$) on a Shishkin mesh with 32 elements and the convergence in pseudo-time of the MG algorithm for a few physical time steps $\Delta t = 0.05$.



(a) Solution (b) Convergence

Fig. 7. The steady-state solution of the advection–diffusion equation ($a = 1$, $d = 0.025$) on a Shishkin mesh with 32 elements and the convergence in pseudo-time of the MG algorithm for a single physical time step $\Delta t = 5$.

These results show that the EXI and EXV methods can indeed be combined to form a cheap local smoother for a full multigrid setting as expected from the analysis.

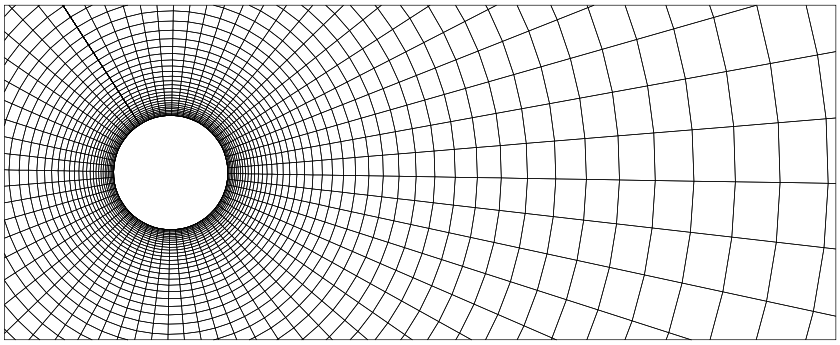
5. Numerical simulations

In this section, we verify through numerical experiments whether the improved convergence predicted by the two-level analysis of the advection–diffusion equation also holds for the compressible Navier–Stokes equations.

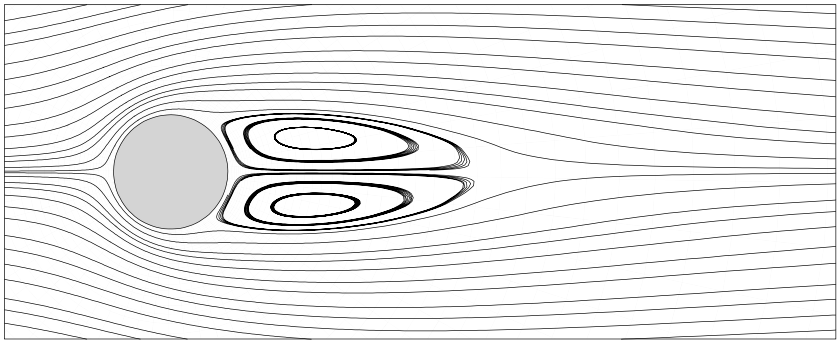
5.1. Definition of work units

To measure the efficiency of the multigrid algorithm, we have to define a basic work unit. The CPU time does not reflect the true work load as it is greatly affected by the implementation, optimization and the machine the code runs on. Therefore, we employ a more transparent definition: one work unit corresponds to one Runge–Kutta step on the fine grid. To account for the work done on the coarse grids in terms of this work unit, we make use of the following observation.

In a well written code, the computational effort of an explicit Runge–Kutta step is proportional to the number of degrees of freedom (DoF). The number of DoF on the fine mesh is $N_e N_q N_c$ with N_e the number of elements, N_q the number of equations and N_c the number of expansion coefficients. On the coarse mesh, the number of elements is N_e/f_e with f_e the mesh coarsening factor. Therefore, the number of DoF on the coarse mesh is $1/f_e$ with respect to fine mesh. For example, $f_e = 8$ in 3D, hence eight coarse grid Runge–Kutta steps are counted as one work unit. A similar counting is done for multiple levels. The prolongation and restriction are trivial and this effort is neglected.



(a) grid



(b) streamlines

Fig. 8. Slice through the 3D grid with $64 \times 64 \times 4$ elements around the circular cylinder and steady-state streamlines at $M_\infty = 0.3$ and $Re_\infty = 40$.

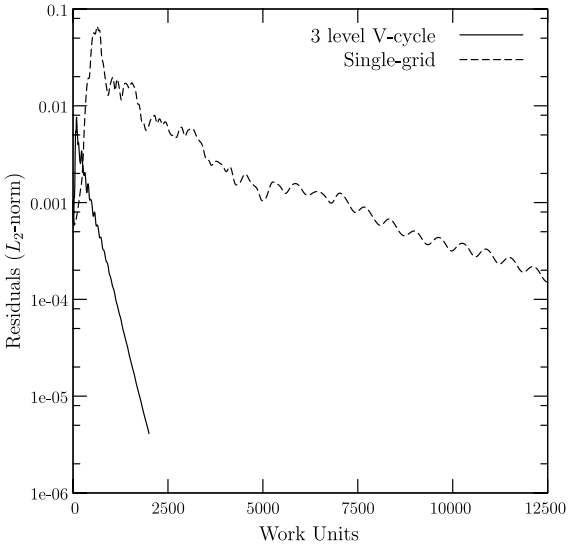


Fig. 9. Convergence to steady-state for the cylinder at $M_\infty = 0.3$ and $Re_\infty = 40$ on the $64 \times 64 \times 4$ grid.

5.2. Flow around a circular cylinder

First, we consider the flow around a circular cylinder with Reynolds number $Re_\infty = 40$ based on the diameter of the cylinder. We solve the compressible Navier–Stokes equations with the space-time discontinuous Galerkin method summarized in Section 2, taking the Mach number $M_\infty = 0.3$. The flow is laminar, steady and characterized by a closed near-wake region with separation and recirculation forming twin eddies, see for example [27]. In Fig. 8, we show a slice through the computational grid with $64 \times 64 \times 4$ elements and the streamlines at steady-state.

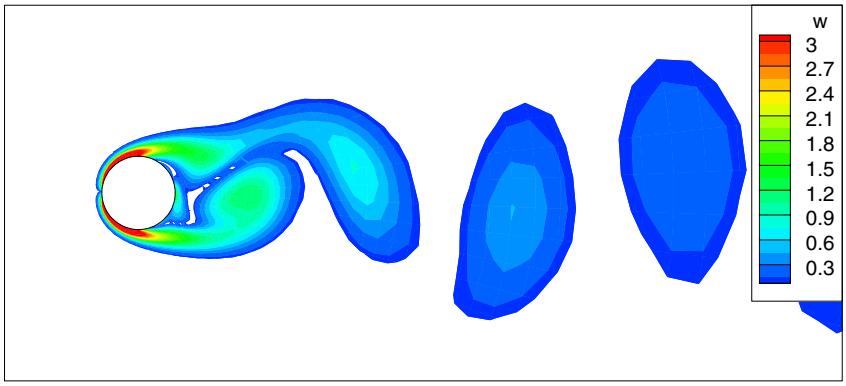
To evaluate the performance of the h-multigrid iteration (Algorithm 3) for solving the system of algebraic equations, we express the convergence in terms of the previously defined work units and compare with single-grid iteration in Fig. 9. With two pre- and post-relaxations on each level[4], we find that multigrid attains three orders of convergence in 2000 WU whereas single-grid only attains two orders of convergence in 12,500 WU. In this case, multigrid iteration is approximately 10 times cheaper than single-grid iteration.

Second, we increase the Reynolds number to $Re_\infty = 200$ and refine the grid to $80 \times 84 \times 4$ elements. The flow now becomes unsteady and is characterized by periodic vortex shedding. The Strouhal number is $St \equiv fd/u_\infty = 0.2$ with f the frequency of the vortex shedding, d the diameter of the cylinder and u_∞ the far-field velocity [18,27]. This gives us the corresponding period $T = 1/f$ and we choose our time-step Δt such that we have 32 time-steps per period. In Fig. 10 we show a snapshot of the vorticity and the streamlines and in Fig. 11 the periodic evolution of the lift and drag coefficients C_L and C_D. In this time-dependent case, we solve the algebraic system for every physical time-step using Algorithm 3 with 2 pre- and post-relaxations on each of the three levels of the $80 \times 84 \times 4$ grid. In Fig. 12, we show the typical convergence in pseudo-time. Single-grid iteration stagnates after 30 work units while multigrid iteration has already met our convergence criterion of order 10^{-6} residuals.

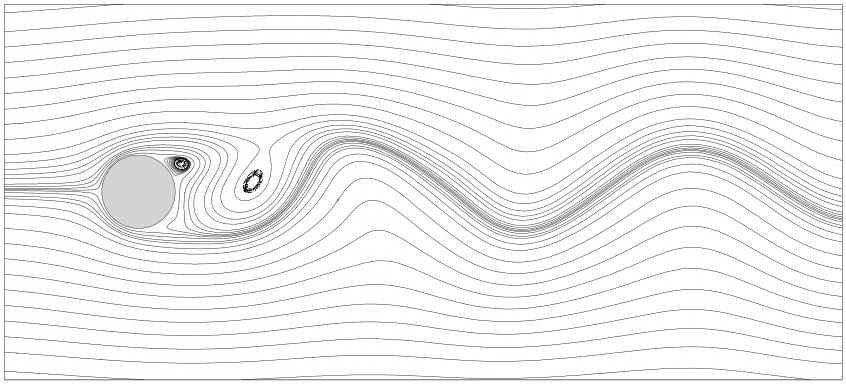
5.3. Flow around an ONERA M6 wing

Finally, we consider the steady laminar flow around an ONERA M6 wing at $M_\infty = 0.4$, $Re_\infty = 10^4$ and angle of attack $\alpha = 1°$. The fine grid consists of 125,000 hexahedral elements. An impression of the grid is given in Fig. 13 where we also show the Mach number isolines in the plane perpendicular to the wing and the pres-

[4] Increasing the number of relaxations and/or changing from V-cycle to W-cycle did not significantly improve the performance in terms of work units.



(a) vorticity magnitude $\omega = ||\nabla \times \vec{u}||$



(b) streamlines

Fig. 10. Snapshot of the vorticity and streamlines around the circular cylinder and streamlines at $M_\infty = 0.3$ and $Re_\infty = 200$.

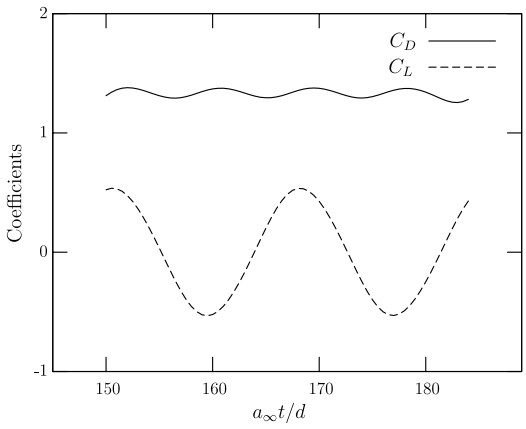


Fig. 11. Periodic evolution of the lift and drag coefficients C_L and C_D for the cylinder at $M_\infty = 0.3$ and $Re_\infty = 200$.

sure coefficient C_p on the wing. This simulation was done with the NLR DG algorithm HEXADAP, in which the algorithms discussed in this article have been implemented.

In Fig. 14, we compare a multigrid iteration consisting of three level V- and W-cycles with single-grid iteration. The V-cycle has a total of four relaxations on each grid level, while the W-cycle has four relaxations on the fine grid and 8 on the medium and coarse grid. In terms of work units both attain residuals of order 10^{-6} in

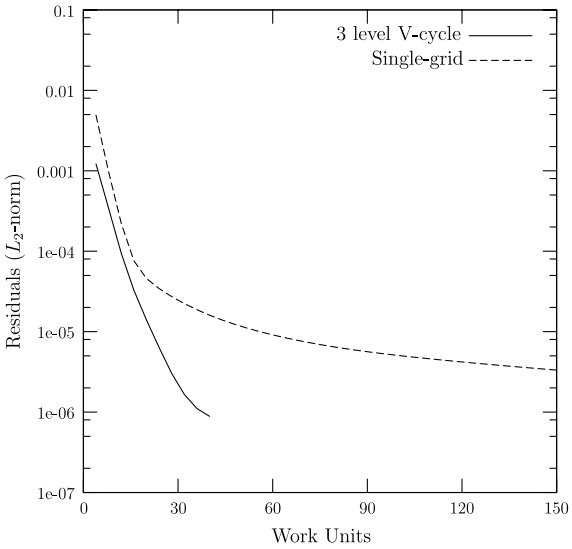


Fig. 12. Typical convergence in pseudo-time for a physical time-step $a_\infty \Delta t/d = 0.5$ for the cylinder at $M_\infty = 0.3$ and $Re_\infty = 200$ on the $80 \times 84 \times 4$ grid.

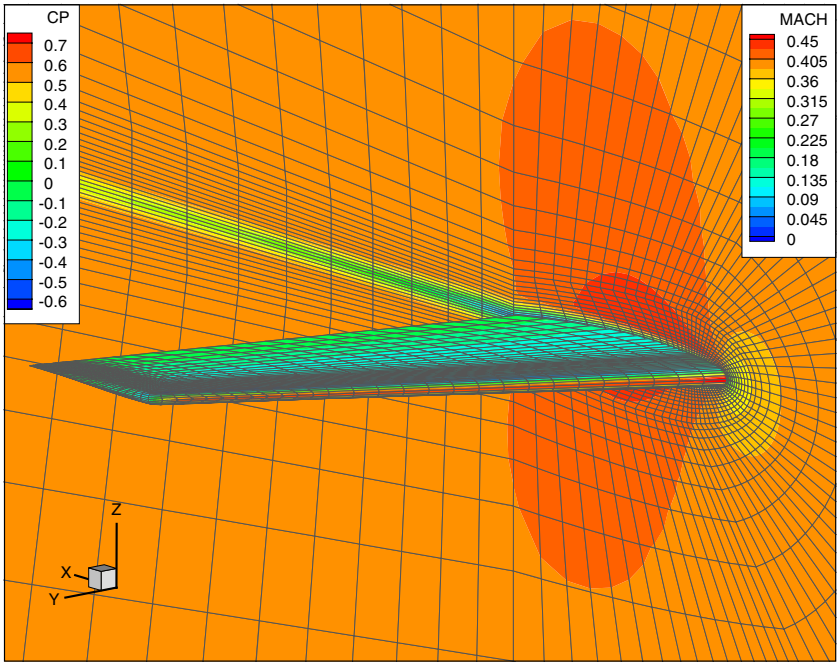


Fig. 13. Impression of the grid, the Mach number isolines and the pressure coefficient C_p on the ONERA M6 wing at $M_\infty = 0.4$, $Re_\infty = 10^4$ and $\alpha = 1°$.

2000 work units, while the residuals with single grid iteration are still of order 10^{-4} after 5000 work units. With the *h*-multigrid algorithm, this simulation can be run with HEXADAP in six hours on a single CPU of NEC SX-8R at 5.3 Gflop/s.

The performance of the single- and multigrid iteration for the numerical experiments presented in this section is summarized in Table 5. For these cases, multigrid iteration significantly reduces the computational effort, up to a factor of 10 w.r.t. the single-grid iteration.

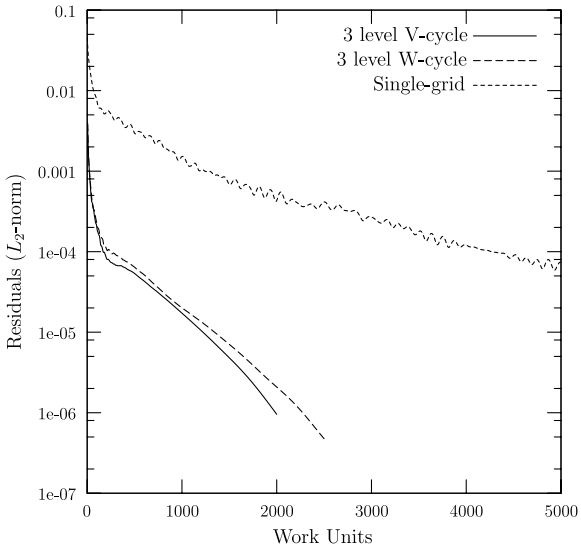


Fig. 14. Convergence in pseudo-time for the ONERA M6 wing at $M_\infty = 0.4$, $Re_\infty = 10^4$ and $\alpha = 1°$.

Table 5
Summary of the computational effort for the numerical experiments

Case	Single-grid performance	Multigrid performance	Cost reduction
Cylinder (steady)	2 orders in 12,500 WU	3 orders in 2000 WU	9.4
Cylinder (unsteady)	3 orders in 150 WU	3 orders in 30 WU	5.0
ONERA M6	2 orders in 5000 WU	3 orders in 2000 WU	3.7

Multigrid iteration reduces the costs by a factor 4–10 w.r.t. single-grid iteration.

6. Discussion and conclusions

In this article, we investigated a *h*-multigrid algorithm for the pseudo-time integration of the system of nonlinear equations arising from the space-time DG discretization of the compressible Navier–Stokes equations. The discretization is second order accurate on all grid levels and the relaxation is done with explicit Runge–Kutta methods in order to keep the multigrid algorithm local. The locality of the solver thereby matches the locality of the discretization, which may be important in view of real-life applications where the assembly and storage of global systems may not be feasible.

We applied two-level Fourier analysis to the space-time DG discretization of the scalar advection–diffusion equation to get an impression of the stability and convergence of the algorithm. This analysis shows that convergence factors between 0.5 and 0.75 can be obtained, depending on the case under consideration, which is quite good for a fully explicit multigrid algorithm.

The construction of intergrid transfer operators is based on the L_2 projection of the coarse grid solution on the fine grid and assumes the embedding of spaces. Although the embedding of spaces does not formally hold on curvilinear grids, we found that the multigrid algorithm still performs well: three level multigrid iteration with two pre- and post-relaxations is up to 10 times less expensive than single-grid iteration for our 2D and 3D test cases. These include steady and unsteady laminar flow around a circular cylinder and steady laminar flow around an ONERA M6 wing.

Although there is a clear performance gain compared to single-grid computations, the full potential of multigrid was not yet reached, in view of the multigrid results obtained for second-order finite volume schemes (see [21] and the references therein) or other DG schemes (for example [10]). The main difference is in the smoother: the line or plane smoothers commonly used in multigrid literature involve the iterative solution of a linear system in an inner loop. This reduces the number of fine mesh residual evaluations,

but adds the costs of constructing and solving a linear system. Whether the advantage of improved convergence rates outweighs the disadvantage of the additional cost associated with such smoothers is very difficult to answer. Most important is the fact that our approach maintains the locality of the DG algorithm, thereby preserving well-known benefits as efficient parallelization and straightforward *hp*-adaptation. Future research should therefore be directed to the development of more efficient smoothers which respect the locality of the DG scheme. A performance comparison with other smoothers or with the Newton-GMRES approach would be highly desirable. Other techniques, such as semi-coarsening in viscous boundary layers or optimization of the coefficients in the Runge–Kutta schemes should also be considered.

Although we applied the *h*-multigrid pseudo-time integration method in the space-time DG context, we would like to point out that it is equally suitable for solving the system of non-linear algebraic equations arising from spatial DG discretizations of the steady compressible Navier–Stokes equations.

Acknowledgments

The work of Christiaan Klaij and Jaap van der Vegt was carried out within the European project ADIGMA on the development of innovative solution algorithms for aerodynamic simulations.

The work of Marc van Raalte has been supported by the Computational Life Science program of the Netherlands Organisation of Scientific Research (NWO 635.100.009) (C-pump project).

The work of Harmen van der Ven is partially carried out within the European project ADIGMA on the development of innovative solution algorithms for aerodynamic simulations, and partially by NLR's programmatic research program.

Marc van Raalte and Christiaan Klaij would like to thank Ben Sommeijer (CWI) for the suggestions and discussions on two-level analysis.

Harmen van der Ven would like to thank Koen Hillewaert for the discussions on multigrid theory for discontinuous Galerkin finite element discretizations.

Appendix A. First order discretization on the coarse grids

For the space-time discontinuous Galerkin discretization of the Euler equations, constant basis functions were used on the coarse grids in the multigrid algorithm presented in [22]. This approach, however, is inadequate for the Navier–Stokes equations, as we show in this section.

The basis functions in Section 2.2 are such that the test and trial functions are split into an element mean at time t_{n+1} and a fluctuating part. In [22], only the element means are used on the coarse grids, which proved effective for the discretization of the Euler equations. An additional benefit is the simplicity of the associated inter-grid transfer operators, which facilitates the implementation for non-uniform, locally refined grids. The restriction and prolongation operators for the solution are defined as [22]:

$$R(\widehat{U})|_{\mathcal{K}_H} = \frac{\sum \widehat{U}_0(\mathcal{K}_h)|\mathcal{K}_h|}{\sum |\mathcal{K}_h|}, \quad P(\widehat{U})|_{\mathcal{K}_h} = \hat{U}_0(\mathcal{K}_H), \tag{A.1}$$

where the coarse grid element $\mathcal{K}_H$ corresponds to a set $\{\mathcal{K}_h\}$ of fine grid elements. The restriction operator $\overline{R}$ for the residual is the same as for the solution ($\overline{R} = R$).

We can analyze this approach for the scalar advection–diffusion equation with the method presented in Section 4. The discretization on the coarse grid with $H = 2h$ now only involves the element means $\hat{u} = \hat{u}_0$ and reduces to:

$$\mathcal{L}_H(\bar{u}^n; \bar{u}^{n-1}) \equiv (\mathcal{L}_H^{\mathrm{a}} + \mathcal{L}_H^{\mathrm{d}})\bar{u}^n + \mathcal{L}_H^t \bar{u}^{n-1} = 0,$$

with

$$\mathcal{L}_H^{\mathrm{a}} \cong [-a\Delta t \mid a\Delta t + H \mid 0], \quad \mathcal{L}_H^{\mathrm{d}} \cong \frac{\mathrm{d}\Delta t}{H}[-2\eta_S \mid 4\eta_S \mid -2\eta_S],$$

Table A.1
Spectral radii in the advection dominated cases

Physics		Stability	Convergence	
σ	Re_h	$\Delta\tau/\Delta t$	$\rho(M_h^{\mathrm{EXI}})$	$\rho(M_h^{\mathrm{TLA}})$
100	100	1.8e−02	0.991	0.979
1	100	1.6e−00	0.796	0.794

Table A.2
Spectral radii in the diffusion dominated cases

Physics		Stability	Convergence	
σ	Re_h	$\Delta\tau/\Delta t$	$\rho(M_h^{\mathrm{EXV}})$	$\rho(M_h^{\mathrm{TLA}})$
100	0.01	8.0e−05	0.999	0.998
1	0.01	8.0e−03	0.993	0.985

and

$$\mathcal{L}_H^t \cong [0 \mid -H \mid 0].$$

This $\mathbb{Z}\times\mathbb{Z}$ system has a block Toeplitz structure with 1×1 blocks, with associated stencil:

$$\mathcal{L}_H \cong [L_H \mid D_H \mid U_H].$$

On this uniform grid, the $3\mathbb{Z}\times\mathbb{Z}$ system associated with the prolongation P defined in (A.1) has a block Toeplitz structure with 3×1 blocks:

$$P = \left[\begin{array}{c|c|c} 1 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{array}\right],$$

and the restriction operator for the residual is $\overline{R} = P^{\mathrm{T}}$. The spectral radius of the two-level operator can now be computed in the same manner as described in Section 4.

The spectral radii of the relaxation schemes and the two-level algorithm with constant basis functions on the coarse grid are given in Tables A.1 and A.2. For steady-state cases, the spectral radius of the relaxation scheme is typically 0.99 and the TLA hardly improves the situation: only in the advection dominated case the spectral radius of the TLA is 0.98. For the other cases, the TLA does not improve the convergence factor, but note that the EXI method is already very efficient for the unsteady advection dominated cases: its spectral radius is 0.79.

Based on this analysis, we do not expect the multigrid algorithm with constant basis functions on the coarse grids to significantly improve the convergence. This was confirmed by numerical experiments, both for the advection–diffusion equation and the compressible Navier–Stokes equations.

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, 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.
[3] F. Bassi, S. Rebay, A high-order discontinuous Galerkin method for compressible turbulent flow, Lecture Notes in Computational Science and Engineering, vol. 11, Springer-Verlag, 2000.
[4] F. Bassi, S. Rebay, Numerical evaluation of two discontinuous Galerkin methods for the compressible Navier–Stokes equations, Int. J. Num. Meth. Fluids 40 (2002) 197–207.
[5] F. Bassi, S. Rebay, G. Mariotti, S. Pedinotti, M. Savini, A high-order accurate discontinuous finite element method for inviscid and viscous turbomachinery flow, in: R. Decuypere, G. Dibelius (Eds.), Proceedings of the 2nd European Conference on Turbomachinery, Fluid Dynamics and Thermodynamics, Technologisch Instituut, Antwerpen, 1997, pp. 99–108.
[6] C.E. Baumann, J.T. Oden, A discontinuous *hp* finite element method for the Euler and Navier–Stokes equations, in: Tenth International Conference on Finite Elements in Fluids (Tucson, AZ, 1998), Internat. 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] W.L. Briggs, Van Emden Henson, S.F. McCormick, A Multigrid Tutorial, SIAM, 2000.
[9] 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.
[10] K.J. Fidkowski, T.A. Oliver, J. Lu, D.L. Darmofal, *p*-Multigrid solution of high-order discontinuous Galerkin discretizations of the compressible Navier–Stokes equations, J. Comput. Phys. 207 (1) (2005) 92–113.
[11] J. Gopalakrishnan, G. Kanschat, A multilevel discontinuous Galerkin method, Numer. Math. 95 (2003) 527–550.
[12] 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.
[13] P.W. Hemker, W. Hoffmann, M.H. van Raalte, Two-level Fourier analysis of a multigrid approach for discontinuous Galerkin discretization, SIAM J. Sci. Comput. 25 (3) (2003) 1018–1041.
[14] P.W. Hemker, W. Hoffmann, M.H. van Raalte, Fourier two-level analysis for discontinuous Galerkin discretization with linear elements, Numer. Linear Algebra Appl. 11 (2004) 473–491.
[15] P.W. Hemker, M.H. van Raalte, Fourier two-level analysis for higher dimensional discontinuous Galerkin discretization, Comput. Vis. Sci. 7 (2004) 159–172.
[16] C.M. Klaij, J.J.W. van der Vegt, H. van der Ven, Pseudo-time stepping methods for space-time discontinuous Galerkin discretizations of the compressible Navier–Stokes equations, J. Comput. Phys. 219 (2) (2006) 622–643.
[17] C.M. Klaij, J.J.W. van der Vegt, H. van der Ven, Space-time discontinuous Galerkin method for the compressible Navier–Stokes equations, J. Comput. Phys. 217 (2) (2006) 589–611.
[18] M. Rosenfeld, D. Kwak, M. Vinokur, A fractional step solution method for the unsteady incompressible Navier–Stokes equations in generalized coordinate systems, J. Comput. Phys. 94 (1991) 102–137.
[19] M. Stynes, E. O'Riordan, A uniformly convergent Galerkin method on a Shishkin mesh for convection–diffusion problem, J. Math. Anal. Appl. 214 (1997) 36–54.
[20] J.J. Sudirham, J.J.W. van der Vegt, R.M.J. van Damme, Space-time discontinuous Galerkin method for advection–diffusion problems on time-dependent domains, Appl. Numer. Math. 56 (12) (2006) 1491–1518.
[21] U. Trottenberg, C.W. Oosterlee, A. Schüller, Multigrid, Academic Press, London, 2001.
[22] 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.
[23] H. van der Ven, O.J. Boelens, B. Oskam, Multitime multigrid convergence acceleration for periodic problems with future applications to rotor simulations, in: Parallel Computational Fluid Dynamics Conference, Egmond aan Zee, Netherlands, May 21–23, 2001.
[24] M.H. van Raalte, Multigrid analysis and embedded boundary conditions for discontinuous Galerkin discretization, PhD Thesis, Korteweg-de Vries Institute, University of Amsterdam, 2004.
[25] M.H. van Raalte, P.W. Hemker, Two-level multigrid analysis for the convection–diffusion equation discretized by a discontinuous Galerkin method, Numer. Linear Algebra Appl. 12 (2005) 563–584.
[26] P. Wesseling, A robust and efficient multigrid method, in: W. Hackbush, U. Trottenberg (Eds.), Multigrid Methods, Springer-Verlag, New York, 1982, pp. 614–630.
[27] M.M. Zdravkovich, Flow Around Circular Cylinder, Vol. 1: Fundamentals, Oxford Science Publications, 1997.