INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN FLUIDS
Int. J. Numer. Meth. Fluids 2009; **60**:847–865
Published online 24 September 2008 in Wiley InterScience (www.interscience.wiley.com). DOI: 10.1002/fld.1917

High-order accurate p-multigrid discontinuous Galerkin solution of the Euler equations

F. Bassi[1], A. Ghidoni[2,*,†], S. Rebay[2] and P. Tesini[1]

[1]*Dipartimento di Ingegneria Industriale, Università degli Studi di Bergamo, Bergamo, Italy*
[2]*Dipartimento di Ingegneria Meccanica e Industriale, Università degli Studi di Brescia, via Branze 38, 25138 Brescia, Italy*

SUMMARY

Discontinuous Galerkin (DG) methods have proven to be perfectly suited for the construction of very high-order accurate numerical schemes on arbitrary unstructured and possibly nonconforming grids for a wide variety of applications, but are rather demanding in terms of computational resources. In order to improve the computational efficiency of this class of methods a p-multigrid solution strategy has been developed, which is based on a semi-implicit Runge–Kutta smoother for high-order polynomial approximations and the implicit Backward Euler smoother for piecewise constant approximations. The effectiveness of the proposed approach is demonstrated by comparison with p-multigrid schemes employing purely explicit smoothing operators for several 2D inviscid test cases. Copyright © 2008 John Wiley & Sons, Ltd.

Received 1 February 2008; Revised 4 June 2008; Accepted 3 August 2008

KEY WORDS: high-order accurate discontinuous Galerkin method; Euler equations; explicit/implicit time integration; p-multigrid

1. INTRODUCTION

Discontinuous Galerkin (DG) methods, originally introduced for purely advective problems [1–5] and successively extended to advection–diffusion [6–9] and to purely elliptic problems [10], are nowadays adopted in a wide variety of applications.

DG methods are finite element methods in which the solution is approximated by means of piecewise continuous functions inside elements with no global continuity requirement. The lack

*Correspondence to: A. Ghidoni, Dipartimento di Ingegneria Meccanica e Industriale, Università degli Studi di Brescia, via Branze 38, 25138 Brescia, Italy.
†E-mail: antonio.ghidoni@ing.unibs.it

Contract/grant sponsor: European Union

of global continuity constraints leads to a discrete approximation characterized by geometrical flexibility. High-order accurate schemes can in fact be constructed on arbitrary and possibly nonconforming grids. Elements of different order of accuracy can be easily accommodated in the same grid, thus opening the way to a straightforward implementation of '*hp*' adaptive solution strategies. The compactness of the scheme is particularly advantageous when an implicit time advancement scheme is employed and/or for a parallel implementation of the method. The price to pay for robustness, accuracy, and flexibility offered by DG methods is their relatively high computational cost and storage requirement.

The DG space discretized equations can be advanced in time using different time integration schemes. Explicit Runge–Kutta (RK) methods are very effective for the solution of unsteady problems characterized by strong discontinuities and/or fast space–time oscillations, and can match in time the high-order accuracy of the DG discretization while retaining total variation stability, see e.g. [3, 4, 11]. However, for large-scale simulations and/or for high-order polynomial approximations, the rate of convergence can be extremely slow, resulting in inefficient solution techniques. In order to improve the computational efficiency of DG methods, implicit time discretization schemes [12, 13], h- and p-multigrid strategies [14–18] have been considered.

Implicit methods require considerable computational resources due to the solution of the linear system arising at each time step and a considerable amount of memory to store the Jacobian matrix, which may be prohibitive for a large-scale problem and high-order solutions. Multigrid methods offer an alternative and efficient approach to steady-state solution. While in the classical h-multigrid method the discrete equations are solved on a series of recursively coarsened grids, in the p-multigrid algorithm the equations are solved by considering a series of progressively lower-order approximations on the same grid. Several p-multigrid algorithms for DG space approximations have been recently proposed in the literature for the inviscid [16, 18] as well as for viscous flows [17], showing that this solution approach is perfectly suited to the DG method. Notice, however, that this work is focused on DG p-multigrid algorithms for purely inviscid flows.

In the p-multigrid context the performance of the RK explicit schemes as smoother is quite disappointing [18], especially for higher-order schemes. Differently from previous methods, in this paper we propose an improvement of the p-multigrid scheme originally introduced in [16], which employs a semi-implicit RK smoother for P^k polynomial approximation if $k>0$, and the implicit backward Euler smoother for P^0 polynomial approximation. The proposed approach has been compared with three different p-multigrid smoothers:

- semi-implicit RK;
- explicit RK;
- explicit RK for P^k polynomial approximation if $k>0$, and the implicit backward Euler for P^0 polynomial approximation.

The comparison shows a considerable reduction of the computational time required by the proposed method with respect to other smoothers.

The organization of this paper is as follows: in Section 2 the DG discretization of the Euler equations is briefly presented, Section 3 describes the p-multigrid algorithm, and Section 4 shows the computed results for the subsonic inviscid flow through a channel with a bump, for the subsonic inviscid flow around a circle, and for the subsonic inviscid flow around a NACA0012 airfoil.

DOI: 10.1002/fld

2. DG FORMULATION

We will first briefly present the DG space discretization of the Euler equations, which can be written in conservation form as

$$\frac{\partial \mathbf{u}}{\partial t}+\nabla\cdot\mathbf{F}(\mathbf{u})=0 \tag{1}$$

with suitable initial and boundary conditions. The conservative variables $\mathbf{u}$ and the Cartesian components $\mathbf{f}(\mathbf{u})$ and $\mathbf{g}(\mathbf{u})$ of the flux function $\mathbf{F}(\mathbf{u})$ are given in the 2D case by

$$\mathbf{u}=\begin{Bmatrix}\rho\\ \rho e_0\\ \rho v_x\\ \rho v_y\end{Bmatrix};\quad \mathbf{f}(\mathbf{u})=\begin{Bmatrix}\rho v_x\\ \rho h_0 v_x\\ \rho v_x^2+P\\ \rho v_y v_x\end{Bmatrix};\quad \mathbf{g}(\mathbf{u})=\begin{Bmatrix}\rho v_y\\ \rho h_0 v_y\\ \rho v_x v_y\\ \rho v_y^2+P\end{Bmatrix} \tag{2}$$

where ρ is the fluid density, v_x and v_y are the Cartesian velocity components, P is the pressure, e_0 is the total (or stagnation) internal energy, and $h_0=e_0+P/\rho$ is the total enthalpy. By assuming that the fluid behaves as a polytropic ideal gas, the pressure can be computed as $P=(\gamma-1)[\rho e_0-\rho(v_x^2+v_y^2)/2]$, where $\gamma=C_p/C_v$ indicates the ratio between the specific heats of the fluid.

Multiplying Equation (1) by a test function $\mathbf{v}$ and integrating by parts over the domain Ω leads to the weak formulation

$$\int_\Omega \mathbf{v}\frac{\partial \mathbf{u}}{\partial t}\,\mathrm{d}\Omega+\oint_{\partial\Omega}\mathbf{v}\mathbf{F}(\mathbf{u}_\star)\cdot\mathbf{n}\,\mathrm{d}\sigma-\int_\Omega \nabla\mathbf{v}\cdot\mathbf{F}(\mathbf{u})\,\mathrm{d}\Omega=0\quad\forall\mathbf{v} \tag{3}$$

where $\partial\Omega$ denotes the boundary of Ω, $\mathbf{n}$ the unit outward normal vector to the boundary, and $\mathbf{u}_\star$ denotes a 'boundary state,' which depends on the prescribed boundary data.

Let Ω_h be an approximation of the domain Ω, and $\mathscr{T}_h=\{e\}$ a triangulation of Ω_h, i.e. a collection of N 'finite elements' e of domain Ω_e and boundary $\partial\Omega_e$, and let $\mathscr{V}_h$ denote the space of piecewise polynomial functions on the element e of $\mathscr{T}_h$, i.e.

$$\mathscr{V}_h=\{\mathbf{v}_h\in L^2(\Omega_h)^{d+2}:\mathbf{v}_h|_{\Omega_e}\in\mathscr{P}^k,\forall e\in\mathscr{T}_h\}$$

where $\mathscr{P}^k(\Omega_e)$ is the space of polynomials of degree at most k in the element e. Notice that the functions in $\mathscr{V}_h$ are in general discontinuous at element interfaces, and that the polynomial order k may in general be different from element to element, thus opening the way to a straightforward implementation of a p-adaptive solution strategy.

The DG approximation of Equation (3) for a generic element $e\in\mathscr{T}_h$ reads as follows: find $\mathbf{u}_h\in\mathscr{V}_h$ so that

$$\int_{\Omega_e}\mathbf{v}_h\frac{\partial \mathbf{u}_h}{\partial t}\,\mathrm{d}\Omega+\oint_{\partial\Omega_e}\mathbf{v}_h\mathbf{h}(\mathbf{u}_h,\mathbf{u}_h^+,\mathbf{n})\,\mathrm{d}\sigma-\int_{\Omega_e}\nabla\mathbf{v}_h\cdot\mathbf{F}(\mathbf{u}_h)\,\mathrm{d}\Omega=0 \tag{4}$$

holds for an arbitrary function $\mathbf{v}_h\in\mathscr{V}_h$. Notice that the flux on $\partial\Omega_e$ is evaluated by means of a 'numerical' flux function $\mathbf{h}(\mathbf{u}_h,\mathbf{u}_h^+,\mathbf{n})$, which depends on the solution vector $\mathbf{u}_h$ associated to e, on the solution vector $\mathbf{u}_h^+$ associated to the neighbors of e (or on the boundary data for a boundary side), and on the unit vector $\mathbf{n}$. The numerical flux function is introduced in order to

DOI: 10.1002/fld

uniquely define the flux at the element interfaces—thus obtaining a consistent and conservative approximation of (3)—and to weakly prescribe the boundary data. For an internal interface, any of the numerical flux functions commonly considered in the finite volume method can be used. In the present work we have used the 'exact' Godunov flux function, i.e. the exact solution of a planar Riemann problem in the direction normal to the boundary. For a boundary face, the numerical flux function is simply set as $\mathbf{h}=\mathbf{F}(\mathbf{u}_\star)\cdot\mathbf{n}$.

By introducing a basis for $\mathscr{V}_h$, i.e. a set of M polynomial shape functions $\phi_{ie}(\mathbf{x})$, $i=1,\ldots,M$ for each element $e\in\mathscr{T}_h$, i.e. $e=1,\ldots,N$, the discrete functions $\mathbf{u}_h$ and $\mathbf{v}_h$ in Ω_e can be expressed as

$$\mathbf{u}_h(\mathbf{x},t)|_{\Omega_e}=\sum_{i=1}^{M}\mathbf{u}_{ie}(t)\phi_i(\mathbf{x}),\quad \mathbf{v}_h|_{\Omega_e}=\sum_{i=1}^{M}\mathbf{v}_{ie}\phi_i(\mathbf{x}),\quad \mathbf{x}\in\Omega_e \tag{5}$$

The DG formulation (4) must be satisfied for any element e and for any function $\mathbf{v}_h$, which is in general a linear combination of the MN shape function ϕ_{ie}, and is therefore equivalent to the system of MN equations

$$\sum_{i=1}^{M}\left[\int_{\Omega_e}\phi_{je}\phi_{ie}\,\mathrm{d}\Omega\right]\frac{\mathrm{d}\mathbf{u}_{ie}(t)}{\mathrm{d}t}+\oint_{\partial\Omega_e}\phi_{je}\mathbf{h}(\mathbf{u}_h,\mathbf{u}_h^+,\mathbf{n})\,\mathrm{d}\sigma-\int_{\Omega_e}\nabla\phi_{je}\cdot\mathbf{F}(\mathbf{u}_h)\,\mathrm{d}\Omega=0 \tag{6}$$

that can be used to compute the time evolution of the MN unknown expansion coefficients $\mathbf{u}_{ie}(t)$. By assembling together all the elemental contributions, the system of ordinary differential equations governing the evolution in time of the discrete solution can be written as

$$\mathbf{M}\frac{\mathrm{d}\mathbf{u}}{\mathrm{d}t}+\mathbf{r}(\mathbf{u})=0 \tag{7}$$

where $\mathbf{M}$ denotes the mass matrix, $\mathbf{u}$ the global vector of the degree of freedom, and $\mathbf{r}(\mathbf{u})$ the residual vector.

In this work, nodal shape functions on equispaced interpolation points have been used. All the integrals appearing in the elemental equations are evaluated by means of Gauss numerical quadrature formulae with a number of integration points consistent with the accuracy required.

3. *P*-MULTIGRID

The convergence rate of standard iterative solvers has a tendency to 'stagnate,' i.e. to fail in effectively reducing the error after a few iterations. This behavior, which is more prominent for fine meshes, is strictly related to the frequency content of the error in the solution. If the error is distributed in the high frequency modes, the convergence rate is fast. However, after the first few iterations, the high frequencies of the error are smoothed out and the convergence rate deteriorates. In h-multigrid methods, a good convergence rate is obtained by adopting a sequence of progressively coarser grids, which allows for an effective reduction of the solution error over the entire frequency field. The p-multigrid approach is based on the same concepts as the standard h-multigrid method except that lower-order approximations on a single grid serve as coarse levels. The p variant of the multigrid idea is perfectly suited for high-order accurate methods and can be easily adopted for both structured and unstructured grids since the interpolation needed to restrict the solution and to prolongate the error between different levels is local to an individual element and there is no need for the complex interpolations procedures as in the h-multigrid case.

DOI: 10.1002/fld

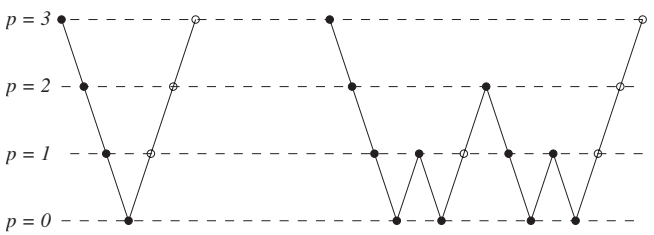

Figure 1. V-cycle and W-cycle for $p=3$ (•: pre-smoothing; ○: post-smoothing).

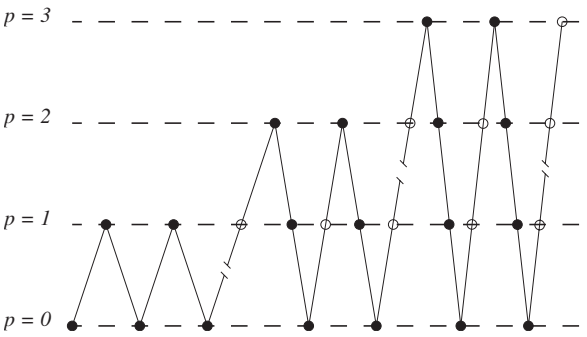

Figure 2. V-cycle full multigrid for $p=3$ (•: pre-smoothing; ○: post-smoothing).

Multigrid methods have proven to be effective techniques for accelerating convergence to steady state for both linear and nonlinear problems [19, 20] and can be applied with many existing relaxation techniques (smoother). The variant of the multigrid method suited to nonlinear problems such as system (2) is called full approximation storage scheme, see e.g. [21].

The various levels can be visited following different paths. In the commonly considered V-cycle and W-cycle, the algorithm visits the various levels as depicted in Figure 1. At each level, a number v_1 of pre-smoothing iterations is performed prior to restricting the solution to the next coarser level (bullets), while, on the way back to 'finer' levels, a number v_2 of post-smoothing iterations is performed after prolongation (circles). As a further improvement of the algorithm, in the full multigrid (FMG) algorithm the coarser level solutions are exploited to obtain good initial guess to initialize the computation of the finer grids. However, converging the solution fully on each level is not practical because the discretization error on the coarser level is usually above machine zero. In the proposed algorithm the solution is prolongated to the finer level when a residual-based criterion is met, as described in [17]. The FMG V-cycle *p*-multigrid strategy adopted in this work is depicted in Figure 2.

3.1. P-multigrid cycle

The entire multigrid strategy is based on a recursive application of the so-called two-level algorithm—in which the 'exact' solution on the coarser grid is used to accelerate the solution on the finer grid. In practice, to avoid the prohibitively expansive exact solution on the coarse grid,

DOI: 10.1002/fld

the two-level algorithm is recursively applied to progressively coarser grids thus arriving at the previously described V-cycle, W-cycle, and FMG algorithm.

In order to illustrate the two-level algorithm, let us consider a generic nonlinear problem $\mathbf{A}^p(\mathbf{u}^p)=\mathbf{f}^p$, where $\mathbf{u}^p$ is the discrete solution vector on a given grid, $\mathbf{A}^p(\mathbf{u}^p)$ is the associated nonlinear algebraic operator and the superscript p indicates the level. Let $\mathbf{v}^p$ be an approximation to the solution vector $\mathbf{u}^p$ and define the discrete residual $\mathbf{r}(\mathbf{v}^p)$ by

$$\mathbf{r}^p(\mathbf{v}^p)=\mathbf{f}^p-\mathbf{A}^p(\mathbf{v}^p)$$

In the basic two-level multigrid method, the exact solution on the coarse level is used to correct the solution on the fine level. The correction is performed according to the following steps:

- restrict the solution and the residual to the coarse level,

$$\mathbf{v}_0^{p-1}=\widetilde{\mathbf{I}}_p^{p-1}\mathbf{v}^p, \quad \mathbf{r}^{p-1}=\mathbf{I}_p^{p-1}\mathbf{r}^p(\mathbf{v}^p) \tag{8}$$

  where $\widetilde{\mathbf{I}}_p^{p-1}$ and $\mathbf{I}_p^{p-1}$ are the solution and the residual restriction operators from level p to level $p-1$, respectively, to be defined in the following;
- compute the forcing term for the coarse level:

$$\mathbf{s}^{p-1}=\mathbf{A}^{p-1}(\mathbf{v}_0^{p-1})-\mathbf{r}^{p-1} \tag{9}$$

- solve the coarse level problem:

$$\mathbf{A}^{p-1}(\mathbf{v}^{p-1})=\mathbf{I}_p^{p-1}\mathbf{f}^p+\mathbf{s}^{p-1} \tag{10}$$

- calculate the coarse grid error:

$$\mathbf{e}^{p-1}=\mathbf{v}^{p-1}-\mathbf{v}_0^{p-1} \tag{11}$$

- prolongate the coarse grid error and correct the fine level approximation:

$$\mathbf{v}^p=\mathbf{v}^p+\widetilde{\mathbf{I}}_{p-1}^p\mathbf{e}^{p-1} \tag{12}$$

  where $\widetilde{\mathbf{I}}_{p-1}^p$ is the error prolongation operator.

3.2. Solution and error transfer operators

The restriction and prolongation operators are simply L^2 projections onto the low-order and high-order spaces $\mathscr{V}^{p-1}$ and $\mathscr{V}^p$, respectively. The low-order level solution is obtained from the high-order solution by requiring that

$$\int_\Omega \mathbf{w}_h^{p-1}\mathbf{v}_h^{p-1}\,\mathrm{d}\Omega=\int_\Omega \mathbf{w}_h^{p-1}\mathbf{v}_h^{p}\,\mathrm{d}\Omega, \quad \forall \mathbf{w}_h^{p-1}\in\mathscr{V}^{p-1} \tag{13}$$

By introducing the matrices

$$\mathbf{M}^{p-1}=[M_{ij}]^{p-1}=\int_\Omega \phi_i^{p-1}\phi_j^{p-1}\,\mathrm{d}\Omega, \quad \mathbf{M}_p^{p-1}=[M_{ij}]_p^{p-1}=\int_\Omega \phi_i^{p-1}\phi_j^{p}\,\mathrm{d}\Omega$$

Equation (13) can be rewritten as

$$\mathbf{M}^{p-1}\mathbf{v}^{p-1}=\mathbf{M}_p^{p-1}\mathbf{v}^{p-1}$$

DOI: 10.1002/fld

which shows that $\mathbf{v}^{p-1}$ can be obtained from $\mathbf{v}^p$ as

$$\mathbf{v}^{p-1} = \widetilde{\mathbf{I}}_p^{p-1}\mathbf{v}^p, \quad \widetilde{\mathbf{I}}_p^{p-1} \equiv (\mathbf{M}^{p-1})^{-1}\mathbf{M}_p^{p-1} \tag{14}$$

which provides the definition for the solution restriction operator $\widetilde{\mathbf{I}}_p^{p-1}$. In a similar fashion, the L^2 prolongation of $\mathbf{e}^{p-1}$ to the higher-order space $\mathscr{V}^p$ can be written as

$$\mathbf{M}^p\mathbf{e}^p = \mathbf{M}_{p-1}^p\mathbf{e}^{p-1}$$

and consequently

$$\mathbf{e}^p = \widetilde{\mathbf{I}}_{p-1}^p\mathbf{e}^{p-1}, \quad \widetilde{\mathbf{I}}_{p-1}^p \equiv (\mathbf{M}^p)^{-1}\mathbf{M}_{p-1}^p = (\mathbf{M}^p)^{-1}(\mathbf{M}_p^{p-1})^{\mathrm{T}} \tag{15}$$

3.3. Residual restriction operator

The residual operator at level p can be written in abstract form as $\mathbf{r}_j^p = B(\phi_j^p, \mathbf{v}^p)$, where $B(\cdot,\cdot)$ is linear in its first argument. The restricted residual $\mathbf{r}^{p-1}$ is defined as $\mathbf{r}_j^{p-1} \equiv B(\phi_j^{p-1}, \mathbf{v}^p)$. An explicit expression of the residual restriction operator $\mathbf{I}_p^{p-1}$ can be obtained following the approach proposed by Fidkowski, see e.g. [17]. Provided that the space $\mathscr{V}^{p-1}$ is a subspace of $\mathscr{V}^p$, a linear relation exists between the basis functions ϕ^{p-1} and ϕ^p, which can be used to express any basis function ϕ^{p-1} as a linear combination of the basis functions ϕ_j^p, say

$$\phi_i^{p-1} = \alpha_{ij}\phi_j^p \tag{16}$$

where α_{ij} are constant coefficients. The matrix $\boldsymbol{\alpha} = [\alpha_{ij}]$ can be related to the solution prolongation operator $\widetilde{\mathbf{I}}_{p-1}^p$ by considering Equation (15) and by expressing ϕ_i^{p-1} in terms of ϕ_i^p using Equation (16), thus obtaining

$$\begin{aligned}\widetilde{\mathbf{I}}_{p-1}^p = (\mathbf{M}^p)^{-1}\mathbf{M}_{p-1}^p &= (\mathbf{M}^p)^{-1}\int_\Omega \phi_i^p\phi_j^{p-1}\,\mathrm{d}\Omega \\ &= (\mathbf{M}^p)^{-1}\alpha_{jk}\int_\Omega \phi_i^p\phi_k^p\,\mathrm{d}\Omega = (\mathbf{M}^p)^{-1}\mathbf{M}^p\boldsymbol{\alpha}^{\mathrm{T}} = \boldsymbol{\alpha}^{\mathrm{T}}\end{aligned} \tag{17}$$

where $\boldsymbol{\alpha}$ denotes the matrix of coefficients α_{ij}. By exploiting the linearity of $B(\cdot,\cdot)$ in its first argument we then have

$$\mathbf{r}_i^{p-1} = B(\phi_i^{p-1}, \mathbf{v}^p) = B(\alpha_{ij}\phi_j^p, \mathbf{v}^p) = \alpha_{ij}B(\phi_j^p, \mathbf{v}^p) = \alpha_{ij}\mathbf{r}_j^p \tag{18}$$

i.e. in matrix notation

$$\mathbf{r}^{p-1} = \boldsymbol{\alpha}\mathbf{r}^p = (\widetilde{\mathbf{I}}_{p-1}^p)^{\mathrm{T}}\mathbf{r}^p$$

which shows that in fact $\mathbf{I}_p^{p-1} = (\widetilde{\mathbf{I}}_{p-1}^p)^{\mathrm{T}}$.

For generic basis functions ϕ_{ie}^p, the restriction and prolongation operators defined above are dense matrices. However, for an orthogonal and hierarchical base, the operators $\widetilde{\mathbf{I}}_p^{p-1}$ and $\widetilde{\mathbf{I}}_{p-1}^p$ are simply given by

$$\widetilde{\mathbf{I}}_p^{p-1} = \delta_{p,p-1}, \quad \widetilde{\mathbf{I}}_{p-1}^p = \delta_{p-1,p}$$

DOI: 10.1002/fld

where δ_{ij} is the Kronecker symbol. In practice, the dofs of the restricted solution are equal to the low-order subset of the high-order solution, and the low-order subset of the prolongated error are the same as the low-order error with null high-order dofs. If M^p and M^{p-1} denote the number of basis functions of a generic element e of order p and $p-1$, respectively, the restriction and prolongation are simply achieved as

$$\mathbf{u}_{ie}^{p-1} = \mathbf{u}_{ie}^{p}, \quad i = 1, \ldots, M^{p-1}$$

$$\mathbf{e}_{ie}^{p} = \mathbf{e}_{ie}^{p-1}, \quad i = 1, \ldots, M^{p-1}$$

$$\mathbf{e}_{ie}^{p} = 0, \quad i = M^{p-1}+1, \ldots, M^{p}$$

3.4. Semi-implicit RK smoother

In the p-multigrid context the performance of explicit RK schemes as smoothers is quite disappointing [18], especially for high-order polynomial approximations. This led us to introduce some implicitness in the m-stage explicit RK scheme

$$\begin{aligned}
&\mathbf{u}^0 = \mathbf{u}^n \\
&\texttt{DO} \quad k = 1,\ m \\
&\qquad \mathbf{u}^k = \mathbf{u}^0 - \alpha_k \Delta t \mathbf{M}^{-1} \mathbf{r}(\mathbf{u}^{k-1}) \\
&\texttt{END DO} \\
&\mathbf{u}^{n+1} = \mathbf{u}^m
\end{aligned} \tag{19}$$

by modifying $\mathbf{r}(\mathbf{u}^{k-1})$ appearing in the above explicit RK scheme with the linearly semi-implicit approximation defined in the following. Let us first consider the linearization $\widehat{\mathbf{r}}(\mathbf{u})$ of the residual vector

$$\begin{aligned}
\widehat{\mathbf{r}}(\mathbf{u}^k) &\approx \mathbf{r}(\mathbf{u}^0) + \frac{\partial \mathbf{r}(\mathbf{u}^0)}{\partial \mathbf{u}} (\mathbf{u}^k - \mathbf{u}^0) \\
&\approx \mathbf{r}(\mathbf{u}^0) + \frac{\partial \mathbf{r}(\mathbf{u}^0)}{\partial \mathbf{u}} (\mathbf{u}^{k-1} - \mathbf{u}^0) + \frac{\partial \mathbf{r}(\mathbf{u}^0)}{\partial \mathbf{u}} (\mathbf{u}^k - \mathbf{u}^{k-1}) \\
&\approx \mathbf{r}(\mathbf{u}^{k-1}) + \frac{\partial \mathbf{r}(\mathbf{u}^0)}{\partial \mathbf{u}} (\mathbf{u}^k - \mathbf{u}^{k-1})
\end{aligned} \tag{20}$$

which allows to write the linearized residual increment $\delta\widehat{\mathbf{r}} = \widehat{\mathbf{r}}(\mathbf{u}^k) - \mathbf{r}(\mathbf{u}^{k-1})$ as

$$\delta\widehat{\mathbf{r}} = \frac{\partial \mathbf{r}(\mathbf{u}^0)}{\partial \mathbf{u}} (\mathbf{u}^k - \mathbf{u}^{k-1}) = [D(\mathbf{u}^0) + O(\mathbf{u}^0)](\mathbf{u}^k - \mathbf{u}^{k-1}) = \delta\widehat{\mathbf{r}}_d + \delta\widehat{\mathbf{r}}_o$$

where

$$\delta\widehat{\mathbf{r}}_d = D(\mathbf{u}^0)(\mathbf{u}^k - \mathbf{u}^{k-1}), \quad \delta\widehat{\mathbf{r}}_o = O(\mathbf{u}^0)(\mathbf{u}^k - \mathbf{u}^{k-1})$$

and $D(\mathbf{u}^0)$ and $O(\mathbf{u}^0)$ are the block diagonal and off-diagonal parts of the full Jacobian matrix $\partial \mathbf{r}(\mathbf{u}^0)/\partial \mathbf{u}$, respectively.

DOI: 10.1002/fld

The semi-implicit scheme is obtained by replacing the occurrence of $\mathbf{r}(\mathbf{u}^{k-1})$ in (19) with the linearized residual corresponding to the diagonal part of the full Jacobian matrix

$$\widehat{\mathbf{r}}_d(\mathbf{u}^k)=\mathbf{r}(\mathbf{u}^{k-1})+\delta\widehat{\mathbf{r}}_d=\mathbf{r}(\mathbf{u}^{k-1})+D(\mathbf{u}^0)(\mathbf{u}^k-\mathbf{u}^{k-1})$$

so that the semi-implicit RK scheme can be written as

$$\begin{aligned}
&\mathbf{u}^0=\mathbf{u}^n\\
&\texttt{DO } k=1,\ m\\
&\quad [\mathbf{M}+\alpha_k\Delta t\,D(\mathbf{u}^0)]\delta\mathbf{u}^k=-\mathbf{M}(\mathbf{u}^{k-1}-\mathbf{u}^0)-\alpha_k\Delta t\,\mathbf{r}(\mathbf{u}^{k-1})\\
&\quad \mathbf{u}^k=\mathbf{u}^{k-1}+\delta\mathbf{u}^k\\
&\texttt{END DO}\\
&\mathbf{u}^{n+1}=\mathbf{u}^m
\end{aligned}\tag{21}$$

As a matter of fact, the implicit coupling of the degrees of freedom inside the elements greatly improves the stability and the smoothing property of the scheme. Further improvement of the damping properties of the scheme in a multigrid approach is expected from our current research work on specifically tuned scheme coefficients.

3.5. Backward Euler smoother

At level $p=0$ an implicit iterative smoother based on the backward Euler scheme is used and Equation (7) can be linearized in time and written as

$$\left(\frac{\mathbf{M}^0}{\Delta t}+\frac{\partial\mathbf{r}(\mathbf{u}^0)}{\partial\mathbf{u}}\right)\Delta\mathbf{u}^0+\mathbf{r}(\mathbf{u}^0)=0\tag{22}$$

where $\mathbf{M}^0$ denotes the mass matrix, $\mathbf{u}^0$ the vector of the unknowns and $\mathbf{r}(\mathbf{u}^0)$ the residual vector at level $p=0$. The fully coupled linear system is solved by means of the GMRES algorithm and the incomplete LU factorization preconditioner. As demonstrated in the numerical results, this fully implicit smoother at the $p=0$ level, allows for a significant improvement in the convergence speed.

4. NUMERICAL RESULTS

This section presents the results for three shockless test cases, the inviscid flow through a channel with a bump, the inviscid flow around a circle, and the inviscid flow around a NACA0012 airfoil.

All the test cases have been computed with the FMG V-cycle and the solution is prolongated to the finer level when a residual-based criterion is met, as described in Section 3. The various explicit and semi-implicit smoothers described in Section 3.4 have been analyzed. In all cases, the performance displayed by the smoothers based on the explicit RK is rather poor, especially for higher-order accurate polynomial approximations, while the performance of the semi-implicit RK smoother is instead very satisfactory, especially when associated with the implicit backward Euler smoother for piecewise constant approximations.

DOI: 10.1002/fld

Four different smoothing strategies have been compared:

ERK: explicit three stages RK scheme (19) with coefficient values $\alpha_1=0.25$, $\alpha_2=0.6667$, $\alpha_3=1.0$, and with $\nu_1=12$ and $\nu_2=12$ pre- and post-smoothing iterations at each level, respectively.

ERK+BE: explicit three stage RK scheme with the same coefficients and the same number of pre-/post-smoothing iteration as above at each level $p>0$, but with $\nu_0=1$ the implicit backward Euler iterations at level $p=0$.

S-IRK: semi-implicit five stage RK scheme (19) with coefficient values $\alpha_1=0.2$, $\alpha_2=0.25$, $\alpha_3=0.3333$, $\alpha_4=0.4$ $\alpha_5=1.0$, and with $\nu_1=3$ and $\nu_2=3$ pre- and post-smoothing iterations at each level, respectively.

S-IRK+BE: semi-implicit five stage RK scheme with the same coefficients as above with $\nu_1=1$ and $\nu_2=1$ pre- and post-smoothing iterations at each level $p>0$, but with $\nu_0=1$ the implicit backward Euler iterations at level $p=0$.

The values of ν_1, ν_2, and ν_0 have been empirically determined in order to minimize the computer time needed to perform the considered test cases. Notice that the values obtained are independent of the test case considered. The value of the RK coefficients is instead taken equal to those that maximize the time accuracy of the method, even if it is well known that these values are not particularly well suited for a multigrid solution strategy. The L^2 norm of the density residual has been used as a convergence indicator. All the simulations have been run on a Laptop equipped with an Intel Centrino Duo 1.8 GHz system with 2 GB RAM.

4.1. Inviscid flow through a channel with a bump

For this test case the domain extends from $-1\leqslant x\leqslant 1$ in the stream-wise direction and $0\leqslant y\leqslant 0.5$ in the cross-stream direction. The shape of the bump is given by

$$y=\begin{cases}[1+\cos(2\pi x)]/8 & \text{if } -1/2\leqslant x\leqslant 1/2\\ 0 & \text{elsewhere}\end{cases}$$

The bump geometry is represented using piecewise quadratic polynomials for all solution approximation orders. Slip boundary conditions are imposed on the top and bottom of the channel. At the inflow, the total temperature, total pressure, and flow angle (0°) are prescribed and at the outflow the static pressure is set, resulting in a free-stream Mach number of $M=0.4$. Two grids have been generated for this configuration, consisting of 128 (coarse) and 512 (fine) triangular elements, respectively, as depicted in Figure 3. Figure 4 illustrates the corresponding Mach contours computed on the fine mesh with a $p=9$ spatial discretization.

The performance of the smoothers previously described is presented in the following: Figure 5 shows the FMG convergence history of the residual L^2 norm plotted versus CPU time for all the smoothing strategies on the fine mesh with a $p=5$ spatial discretization. The timing for the $p=0$ level is not reported because the converged $p=0$ solution serves as the starting point for the FMG

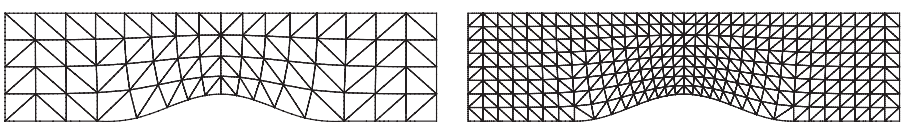

Figure 3. Meshes used to compute a compressible channel flow over a bump: 128 (left) and 512 (right) triangular elements.

DOI: 10.1002/fld

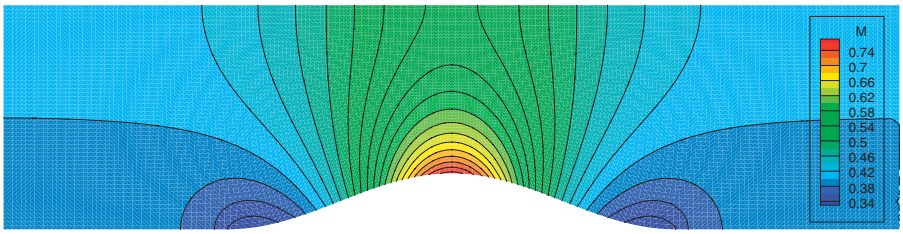

Figure 4. Channel test case: Mach contours with a $p=9$ spatial discretization on the fine mesh (512 triangles).

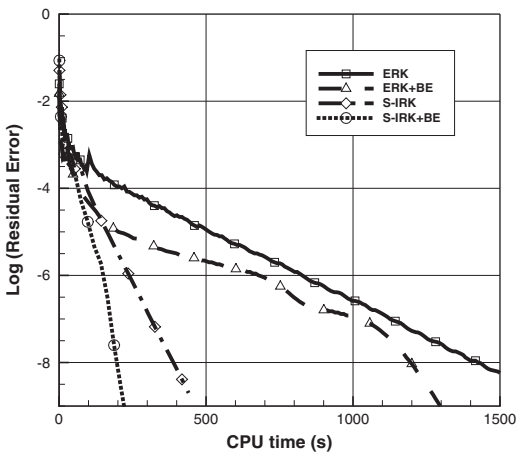

Figure 5. Channel test case: residual L^2 norm convergence history on the fine mesh. Solid line: ERK strategy. Dashed line: ERK+BE strategy. Dash-dot line: S-IRK strategy. Dotted line: S-IRK+BE.

algorithm at the higher levels. Note that the previously mentioned residual based criterion is used to switch to higher levels. As shown in Figure 5, the 'S-IRK+BE' solution strategy is the most efficient in this case. Table I presents the comparison of the performance of the various smoothers for the channel test case, and, in particular, the differences of cpu time and FMG cycles needed to converge with a $p=5$ spatial discretization. The 'S-IRK+BE' strategy is the most efficient regardless of the used mesh and a computational time reduction of $\approx 85\%$ can be observed with respect to the 'ERK' case.

To study the asymptotic behavior of the residual L^2 norm, the FMG solution with the 'S-IRK+BE' strategy has been computed for a $p=9$ spatial discretization with full convergence on each level, as depicted in Figure 6. The convergence history is presented in Table II, where the number of iterations needed to fully converge on each level $N_{\mathrm{FMG}}(128)$ and $N_{\mathrm{FMG}}(512)$, and the slope of a linear regression of each convergence curve $s(128)$ and $s(512)$, has been given for the coarse and the fine mesh, respectively. The table shows the p-multigrid order independent property since the slope of the convergence curves is almost constant on each level. The table also shows that the convergence history is h-dependent since the number of iterations required for convergence increases with the number of elements, as can be expected for a p-multigrid solution strategy.

DOI: 10.1002/fld

Table I. Channel test case: comparison of different p-multigrid smoothing strategies.

Smoother	N_e	v_0	$v_{1,2}$	N_{FMG}	$t_c(s)$	Δt
ERK	128	12	12	109	188	0
ERK+BE	128	1	12	100	181	−4%
S-IRK	128	3	3	37	69	−64%
S-IRK+BE	128	1	1	29	38	−80%
ERK	512	12	12	228	1710	0
ERK+BE	512	1	12	156	1591	−7%
S-IRK	512	3	3	34	453	−74%
S-IRK+BE	512	1	1	19	235	−87%

N_e is the number of mesh elements; v_0 is the number of smoothing iteration at level $p=0$; $v_{1,2}$ is the number of pre-/post-smoothing iterations on each level $p>0$; N_{FMG} is the number of FMG cycles to converge; t_c is the computational time; Δt is the computational time reduction with respect to 'ERK' strategy which is taken as the reference case.

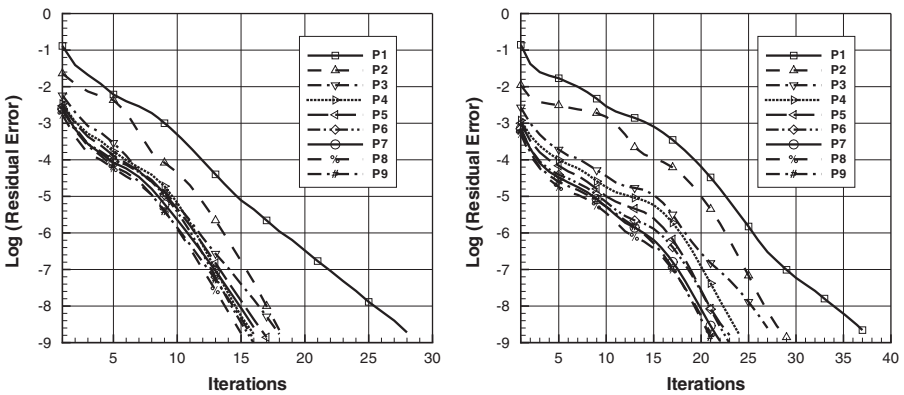

Figure 6. Channel test case: residual convergence rate (full convergence on each level) as a function of FMG cycles for $p=9$ on a mesh of 128 triangles (left) and of 512 triangles (right). Solid line with square symbols: P^1 curve. Dashed line with delta symbols: P^2 curve. Dash-dot line with gradient symbols: P^3 curve. Dotted line with right-triangle symbols: P^4 curve. Long dashed line with left-triangle symbols: P^5 curve. Dashed-dot-dot line with diamond symbols: P^6 curve. Solid line with circle symbols: P^7 curve. Dashed line with percentage symbols: P^8 curve. Dashed-dot line with '#' symbols: P^9 curve.

4.2. Inviscid flow around a circle

In the second test case, the flow around a circle is computed for a farfield Mach number $M_\infty=0.38$, which is close to the limiting value to have a completely subsonic and therefore shockless flow. The impermeability wall boundary condition is prescribed on the circle, and the circle geometry is represented using piecewise quadratic polynomials for all solution approximation orders. Details of the coarse (128 triangles) and of the fine mesh (512 triangles) are displayed in Figure 7, while Figure 8 illustrates the Mach contours computed on the coarse mesh with a $p=9$ spatial discretization.

The performance of the various smoothing strategies is analyzed in Figure 9, which presents the residual L^2 norm convergence behavior as a function of cpu time for each smoother on the coarse

DOI: 10.1002/fld

Table II. Channel test case: FMG cycles needed to converge on each level and slope of the convergence curves for the 'S-IRK+BE' smoothing strategy.

	P^1	P^2	P^3	P^4	P^5	P^6	P^7	P^8	P^9
$N_{\mathrm{FMG}}(128)$	28	18	18	16	17	16	16	15	16
$s(128)$	−0.29	−0.4	−0.38	−0.40	−0.38	−0.4	−0.38	−0.39	−0.39
$N_{\mathrm{FMG}}(512)$	37	29	27	24	23	23	22	21	21
$s(512)$	−0.22	−0.23	−0.21	−0.22	−0.23	−0.23	−0.24	−0.23	−0.23

P^k is the polynomial order with $1 \leqslant k \leqslant 9$; $N_{\mathrm{FMG}}(128)$ and $N_{\mathrm{FMG}}(512)$ are the number of iterations needed to converge on the coarse and fine mesh; $s(128)$ and $s(512)$ are the slopes of the linear regression of each convergence curve.

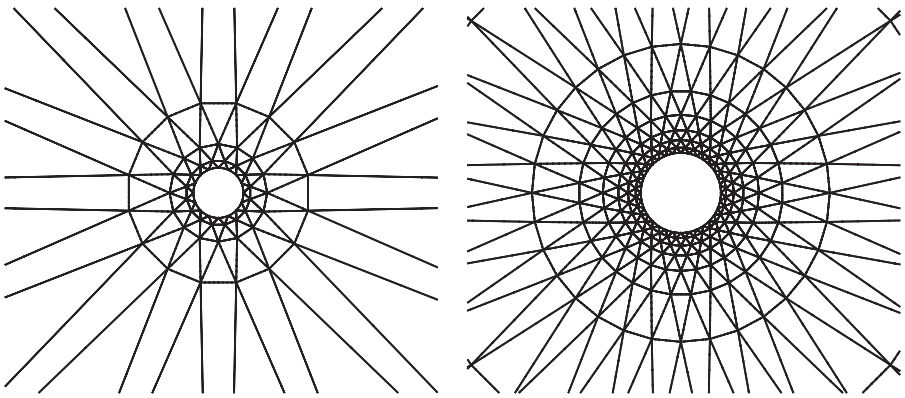

Figure 7. Meshes used to compute an inviscid flow around a circle: 128 (left) and 512 (right) triangular elements.

mesh with a $p=5$ spatial discretization, and in Table III, which shows the cpu time and the FMG cycles needed to converge with a $p=5$ spatial discretization for both the coarse and fine meshes. Also in this case, the 'S-IRK+BE' strategy is the most efficient regardless of the adopted mesh, and a computational time reduction of $\approx 88\%$ can be observed with respect to the 'ERK' case.

Figure 10 illustrates the residual convergence rate as a function of FMG cycles at $p=9$ obtained by using 'S-IRK+BE' strategy. In order to put in evidence the p-multigrid order independent property, the residual is in this case fully converged on each approximation level. Table IV shows both the number of iterations and the slope of the convergence curves for the coarse and fine meshes. Notice that in this case the p-multigrid convergence rate slightly improves with increasing approximation order.

4.3. Inviscid subsonic flow around a NACA0012 airfoil

In the last test case the inviscid flow around a NACA0012 airfoil is computed for a farfield Mach number $M_\infty = 0.5$ and an angle of attack $\alpha = 2°$. The impermeability condition is prescribed on the airfoil surface, and the boundary is represented using piecewise quadratic polynomials for all solution approximation orders. The mesh (512 triangular elements) and the Mach contour obtained with a $p=9$ spatial discretization are depicted in Figure 11.

DOI: 10.1002/fld

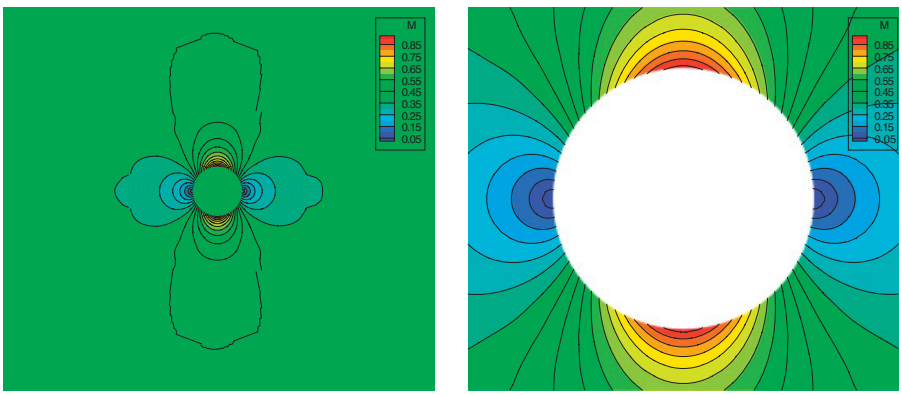

Figure 8. Circle test case: Mach contours for $p=9$ on a mesh of 128 triangles.

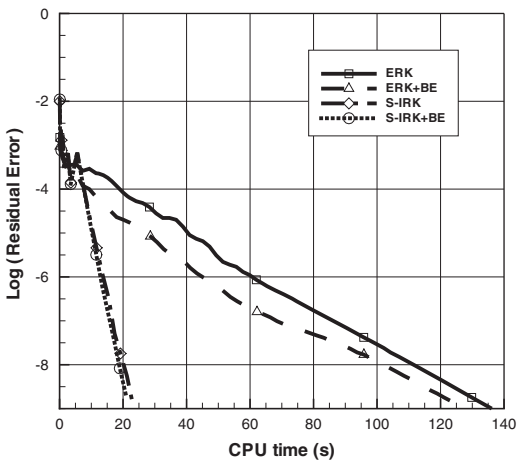

Figure 9. Circle test case: residual L^2 norm convergence history on the coarse mesh. Solid line: ERK strategy. Dashed line: ERK+BE strategy. Dash-dot line: S-IRK strategy. Dotted line: S-IRK+BE.

The cpu time and the number of FMG cycles needed to converge with a $p=5$ spatial discretization are shown in Table V. The 'S-IRK+BE' strategy is again the most efficient with a computational time reduction of $\approx 78\%$ with respect to the 'ERK' case. This behavior can be observed also in the left part of Figure 12, which presents the residual L^2 norm convergence curves of the various smoothing strategies as a function of cpu time. The right part of Figure 12, instead, illustrates the residual L^2 norm convergence rate as a function of FMG cycles obtained by using the 'S-IRK+BE' strategy at $p=9$. Also in this case, the residual is fully converged on each level in order to show the p-multigrid order independent property. The convergence history is finally given in Table VI both as a function of number of FMG cycles needed for convergence and of convergence slope. Also in this case the p-multigrid convergence rate is perfectly satisfactory.

DOI: 10.1002/fld

Table III. Circle test case: comparison of different p-multigrid smoothing strategies.

Smoother	N_e	v_0	$v_{1,2}$	N_{FMG}	$t_c(s)$	Δt
ERK	128	12	12	71	149	0
ERK+BE	128	1	12	68	136	−9%
S-IRK	128	3	3	19	24	−84%
S-IRK+BE	128	1	1	18	21	−86%
ERK	512	12	12	111	892	0
ERK+BE	512	1	12	62	492	−45%
S-IRK	512	3	3	30	247	−73%
S-IRK+BE	512	1	1	22	98	−90%

N_e is the number of mesh elements; v_0 is the number of smoothing iterations at level $p=0$; $v_{1,2}$ is the number of pre-/post-smoothing iterations at each level $p>0$; N_{FMG} is the number of FMG cycles; t_c is the computational time; Δt is the computational time reduction with respect to 'ERK' strategy which is taken as reference case.

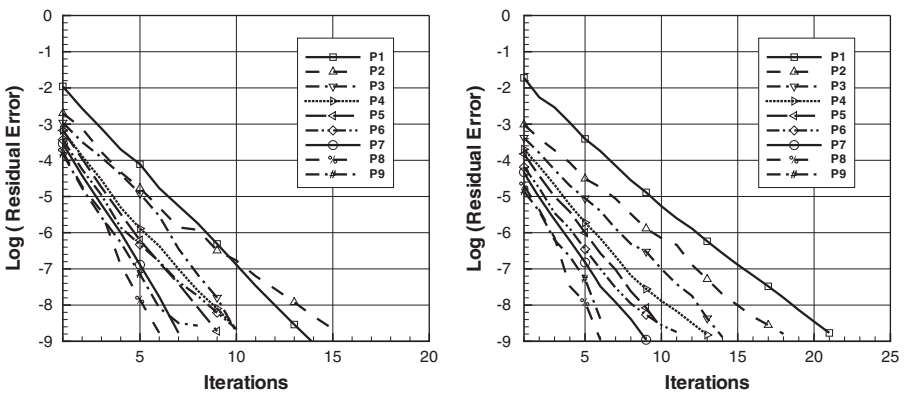

Figure 10. Circle test case: residual convergence rate (full convergence on each level) as a function of FMG cycles for $p=9$ on a mesh of 128 triangles (left) and of 512 triangles (right). Solid line with square symbols: P^1 curve. Dashed line with delta symbols: P^2 curve. Dash-dot line with gradient symbols: P^3 curve. Dotted line with right-triangle symbols: P^4 curve. Long dashed line with left-triangle symbols: P^5 curve. Dashed-dot-dot line with diamond symbols: P^6 curve. Solid line with circle symbols: P^7 curve. Dashed line with percentage symbols: P^8 curve. Dashed-dot line with '#' symbols: P^9 curve.

Table IV. Circle test case: FMG cycles needed to converge on each level and slope of the convergence curves for the 'S-IRK+BE' smoothing strategy.

	P^1	P^2	P^3	P^4	P^5	P^6	P^7	P^8	P^9
$N_{\mathrm{FMG}}(128)$	14	15	10	10	9	10	7	6	8
$s(128)$	−0.54	−0.42	−0.64	−0.61	−0.69	−0.57	−0.86	−1.02	−0.72
$N_{\mathrm{FMG}}(512)$	21	18	14	13	10	11	9	6	6
$s(512)$	−0.35	−0.35	−0.41	−0.43	−0.53	−0.47	−0.57	−0.86	−0.68

P^k is the polynomial order with $1\leqslant k\leqslant 9$; $N_{\mathrm{FMG}}(128)$ and $N_{\mathrm{FMG}}(512)$ are the number of iterations needed to converge on the grid with 128 and 512 elements, respectively; $s(128)$ and $s(512)$ are the slopes of the convergence curves.

DOI: 10.1002/fld

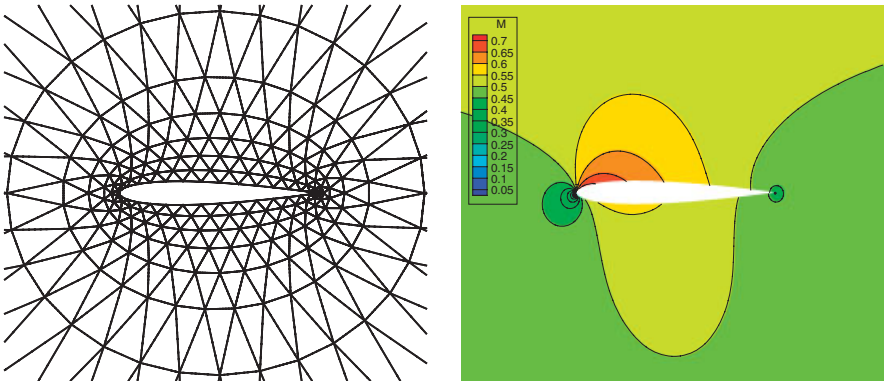

Figure 11. NACA0012 airfoil test case: mesh (512 triangular elements) used to compute an inviscid flow around a NACA0012 airfoil (left) and the corresponding Mach contours with a $p=9$ spatial discretization (right).

Table V. NACA0012 airfoil test case: comparison of different p-multigrid smoothing strategies.

Smoother	ν_0	$\nu_{1,2}$	N_{FMG}	$t_c(s)$	Δt
ERK	12	12	106	822	0
ERK+BE	1	12	69	575	−31%
S-IRK	3	3	20	212	−75%
S-IRK+BE	1	1	14	187	−78%

ν_0 is the number of smoothing iterations at level $p=0$; $\nu_{1,2}$ is the number of pre-/post-smoothing iterations at each level $p>0$; N_{FMG} is the number of FMG cycles; t_c is the computational time; Δt is the computational time reduction with respect to 'ERK' strategy which is taken as reference.

5. CONCLUSIONS

A p-multigrid discontinuous Galerkin (DG) algorithm for the solution of the steady-state Euler equations has been proposed, which employs either explicit or semi-implicit Runge–Kutta (RK) smoothers for the high-order polynomial approximations and possibly the implicit backward Euler smoother for the piecewise constant approximation.

The performance of the various p-multigrid smoothers considered has been evaluated by comparing the results obtained in the computation of three different shockless test cases. In all the test cases considered, the best performance has been obtained with the S-IRK+BE smoother, which allows for a very significant reduction of computing time with respect to the simple ERK smoother. The saving can be as high as 90% and is always greater than 78%. It is, however, to be noticed that the computational resources needed by the backward Euler scheme at the $p=0$ level are not significant for the small 2D test cases here considered, but might become considerable for complex 3D problems. In the latter case, the computational resources required by the solution of the linear system at the $p=0$ level could be a significant fraction of the overall computational cost, and the simpler S-IRK smoother could be more efficient than S-IRK+BE. Notice in fact that, even for the small problems considered here, the computational gain offered by the S-IRK

DOI: 10.1002/fld

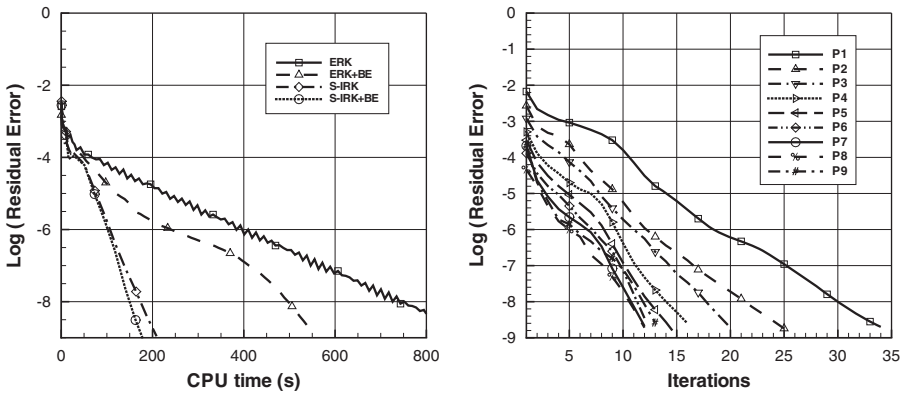

Figure 12. NACA0012 airfoil test case: residual convergence history as a function of cpu time of different smoothers (left). Solid line: ERK strategy. Dashed line: ERK+BE strategy. Dash-dot line: S-IRK strategy. Dotted line: S-IRK+BE. Residual convergence rate (full convergence on each level) as a function of FMG cycles for the 'S-IRK+BE' strategy (right). Solid line with square symbols: P^1 curve. Dashed line with delta symbols: P^2 curve. Dash-dot line with gradient symbols: P^3 curve. Dotted line with right-triangle symbols: P^4 curve. Long dashed line with left-triangle symbols: P^5 curve. Dashed-dot-dot line with diamond symbols: P^6 curve. Solid line with circle symbols: P^7 curve. Dashed line with percentage symbols: P^8 curve. Dashed-dot line with '#' symbols: P^9 curve.

Table VI. NACA0012 airfoil test case: FMG cycles needed to converge on each level and slope of the convergence curves for the 'S-IRK+BE' smoothing strategy. P^k is the polynomial order with $1 \leqslant k \leqslant 9$; $N_{\mathrm{FMG}}(512)$ is the number of FMG cycles needed to converge; $s(512)$ is the slope of each convergence curve.

	P^1	P^2	P^3	P^4	P^5	P^6	P^7	P^8	P^9
$N_{\mathrm{FMG}}(512)$	34	25	20	16	15	12	12	12	13
$s(512)$	−0.2	−0.26	−0.3	−0.35	−0.38	−0.38	−0.4	−0.37	−0.34

smoother is still remarkable, and ranges from a minimum of 64% to a maximum of 84%, as shown in the previous section.

The performance of the p-multigrid algorithm is here assessed on simple shockless flows, which can be computed with the 'standard' DG space discretization method described in Section 2. The computation of shocked flows requires instead some sort of oscillation control mechanism, however, an effective oscillation control technique for very high-order approximation such as those considered here is still an open problem, which is currently under investigation by many research groups, even if some effective methods to deal with shocked flows for moderately high-order approximations have been proposed in the literature. The effectiveness of the proposed method for more complex shocked flows, which can be expected to be strictly connected with the type of oscillation control mechanism employed, will be the object of a future research effort.

DOI: 10.1002/fld

Further significant saving in the computational resources required to reach a prescribed accuracy could be obtained by combining the proposed p-multigrid solution method with a p-adaptive solution strategy, but this issue has not been studied in the present work.

Work is under way to extend this algorithm to grids of quadrilateral elements and to the 3D case, to investigate other possible smoothing procedures, and to apply the proposed method to the solution of the compressible Navier–Stokes and RANS equations.

ACKNOWLEDGEMENTS

This work has been supported by the European Union, under the ADIGMA project on the development of innovative solution algorithms for aerodynamic simulations. The authors would further like to thank the reviewers for their helpful comments.

REFERENCES

1. Reed WH, Hill TR. Triangular mesh methods for the neutron transport equation. *Technical Report LA-UR-73-479*, Los Alamos Scientific Laboratory, 1973.
2. LeSaint P, Raviart PA. On a finite element method for solving the neutron transport equation. In *Mathematical Aspects of Finite Elements in Partial Differential Equations*, de Boor C (ed.). Academic Press: New York, 1974; 89–145.
3. Cockburn B, Shu CW. TVB Runge–Kutta local projection discontinuous Galerkin finite element method for conservation laws II: general framework. *Mathematics of Computation* 1989; **52**:411–435.
4. Cockburn B, Shu CW. TVB Runge–Kutta local projection discontinuous Galerkin finite element for conservation laws III: one dimensional systems. *Journal of Computational Physics* 1989; **84**(1):90–113.
5. Cockburn B, Hou S, Shu CW. The Runge–Kutta local projection discontinuous Galerkin finite element method for conservation laws IV: the multidimensional case. *Mathematics of Computation* 1990; **54**:454–581.
6. Cockburn B, Shu CW. The local discontinuous Galerkin method for time-dependent convection–diffusion systems. *SIAM Journal on Numerical Analysis* 1998; **35**(6):2440–2463.
7. Bassi F, Rebay S. A high-order accurate discontinuous finite element method for the numerical solution of the compressible Navier–Stokes equations. *Journal of Computational Physics* 1997; **131**(2):267–279.
8. Bassi F, Rebay S, Mariotti G, Pedinotti S, Savini M. A high-order accurate discontinuous finite element method for inviscid and viscous turbomachinery flows. In *2nd European Conference on Turbomachinery Fluid Dynamics and Thermodynamics*, Decuypere R, Dibelius G (eds). Technologisch Instituut: Antwerpen, Belgium, 1997; 99–108.
9. Bassi F, Crivellini A, Rebay S, Savini M. Discontinuous Galerkin solution of the Reynolds averaged Navier–Stokes and k–ω turbulence model equations. *Computers and Fluids* 2005; **34**(4–5):507–540. Electronic version available at: www.sciencedirect.com.
10. Arnold DN, Brezzi F, Cockburn B, Marini LD. Unified analysis of discontinuous Galerkin method for elliptic problems. *SIAM Journal on Numerical Analysis* 2002; **39**(5):1749–1779.
11. Cockburn B, Lin SY, Shu CW. TVB Runge–Kutta local projection discontinuous Galerkin finite element for conservation laws V: multidimensional systems. *Journal of Computational Physics* 1998; **141**(2):199–224.
12. Bassi F, Rebay S. GMRES discontinuous Galerkin solution of the compressible Navier–Stokes equations. *Discontinuous Galerkin Methods*: *Theory*, *Computation and Applications*. Lecture Notes in Computational Science and Engineering. Springer: Berlin, 2000.
13. Rasetarinera P, Hussaini MY. An efficient implicit discontinuous spectral Galerkin method. *Journal of Computational Physics* 2001; **172**(2):718–738.
14. Helenbrook BT, Mavriplis DJ, Atkins HA. Analysis of p-multigrid for continuous and discontinuous finite element discretizations. *16th AIAA Computational Fluid Dynamics Conference*, *AIAA*. AIAA: Orlando, FL, 2003.
15. van der Vegt JJW, van der Ven H. Space–time discontinuous Galerkin finite element method with dynamic grid motion for inviscid compressible flows: I. General formulation. *Journal of Computational Physics* 2002; **182**(2):546–585.
16. Bassi F, Rebay S. Numerical solution of the Euler equations with a multiorder discontinuous finite element method. In *Computational Fluid Dynamics 2002*: *Proceedings of the Second International Conference on Computational Fluid Dynamics*, Armfield S, Morgan P, Srinivas K (eds). Springer: Sydney, 2002; 199–204.

DOI: 10.1002/fld

17. Fidkowski KJ, Oliver TA, Lu J, Darmofal L. *p*-multigrid solution of high-order discontinuous Galerkin discretizations of the compressible Navier–Stokes equations. *Journal of Computational Physics* 2005; **207**(1): 92–113.
18. Luo H, Baum JD, Löhner R. A *p*-multigrid discontinuous Galerkin method for the Euler equations on unstructured grids. *Journal of Computational Physics* 2006; **211**(21):767–783.
19. Trottenberg U, Oosterlee CW, Schüller A. *Multigrid*. Academic Press: London, U.K., 2001.
20. Briggs WL, Henson VE, McCormick SF. *A Multigrid Tutorial* (2nd edn). SIAM: Philadelphia, PA, 2000.
21. Brandt A. *Guide to Multigrid Development*. Springer: Berlin, 1982.

DOI: 10.1002/fld