6th European Conference on Computational Mechanics (ECCM 6)
7th European Conference on Computational Fluid Dynamics (ECFD 7)
11-15 June 2018, Glasgow, UK

P-MULTIGRID PRECONDITIONERS APPLIED TO HIGH-ORDER DG AND HDG DISCRETIZATIONS

M. FRANCIOLINI[1,2], K. J. FIDKOWSKI[1] and A. CRIVELLINI[2]

[1] Department of Aerospace Engineering, University of Michigan
1320 Beal Ave, 48109, Ann Arbor (MI), USA
{mfrancio,kfid}@umich.edu

[2] Department of Industrial Engineering and Mathematical Science,
Polytechnic University of Marche
Via Brecce Bianche 12, 60131, Ancona, Italy
{m.franciolini@pm.,a.crivellini@}univpm.it

Key words: high-order, DG, HDG, p-multigrid, preconditioners, parallel efficiency

Abstract. In this work the use of a p-multigrid preconditioned flexible GMRES solver to deal with the solution of stiff linear systems arising from high order time discretization is explored in the context of two high-order spatial discretizations. The first one is a standard modal discontinuous Galerkin method, while the second one is an hybridizable discontinuous Galerkin method, which for high order has fewer globally-coupled degrees of freedom compared to DG. The efficiency of the proposed solution strategy is assessed on low-Mach, two-dimensional, compressible flow problems. The numerical results highlight that a considerable reduction in the number of GMRES iterations can be achieved for both space discretizations, but that only with DG is this gain reflected in the CPU time. Moreover, a comparison of the performance shed light on the convenience of using the former or the latter space discretization.

1 INTRODUCTION

In recent years, high-order discontinuous Galerkin (DG) methods have become increasingly popular in the field of Computational Fluid Dynamics. This fact is certainly ascribed to their convenient dispersion and diffusion properties, the ease of parallelization thanks to their compact stencil, and their accuracy in arbitrary complex geometries. However, the implementation of an efficient solution strategy is still a subject of active research, especially for unsteady flow problems [1] involving the solution of the Navier–Stokes (NS) equations.

Several previous studies demonstrated that implicit schemes in the context of high-order space discretizations are one of the most viable ways to overcome the strict stability limits of explicit time integration schemes [2]. Such strategies require the solution of a large system of linear/non-linear equations, which is typically performed with iterative solvers such as the generalised minimial residual method (GMRES). The choice of the preconditioner is a key aspect of the strategy and it has been explored extensively in the literature, see for example [3, 4]. Among those, the use of multilevel algorithms to precondition a flexible GMRES solver has been demonstrated to be an appealing choice for both compressible [3] and incompressible flow problems [5].

More recently, hybridizable discontinuous Galerkin (HDG) methods have been considered as an alternative to the standard discontinous Galerkin discretization [6, 7]. HDG methods, which introduce an additional trace variable defined on the mesh element faces, can reduce the globally coupled degrees of freedom if compared to DG when a high order of polynomial approximation is employed. In fact, exploiting the block-structured nature of the matrix, the system size can be reduced using static condensation. Moreover, HDG methods exhibit superconvergence properties of the gradient variable in diffusion-dominated regimes. On the other hand, they increase the amount of operations local to each element and the workload between and after the linear system solution. While several works aimed to compare the accuracy of HDG both versus CG and DG, a comparison of the iterative solvers is missing in this context. Additionally, the use of multilevel solution strategies in HDG contexts has been introduced only partially by [8], in the context of an h-multigrid strategy with trace variable projections on an underlying continuous finite element space.

The present work focuses on the linear solution process. In particular, the use of a p-multigrid preconditioned flexible GMRES algorithm to deal with the solution of stiff linear systems arising from a high-order time discretization is explored in the context of DG and HDG high-order spatial discretizations. The scalability of the algorithm is also considered and compared to standard single-grid preconditioners like ILU(0). The efficiency of the different solution strategies is assessed on two-dimensional laminar compressible flow problems. The results of such cases suggest that (i) similar error levels can be obtained by the two solvers, (ii) the use of a multilevel strategy reduces considerably the number of linear iterations, and (iii) only for the DG discretizations this advantage is reflected in the CPU time.

2 DISCRETIZATION

The set of compressible Navier–Stokes (NS) equations can be written in compact form as

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

where $\mathbf{u} \in \mathbb{R}^M$ is the state vector and $\mathbf{F} \in \mathbb{R}^{M\times N}$ is the sum of the inviscid and viscous fluxes, with M being the number of conservative variables and N the number of space dimensions. For HDG, (1) is written as a system of first-order partial differential equations, by introducing $\mathbf{q} \in \mathbb{R}^{M\times N}$ such that

$$\begin{aligned} \mathbf{q} - \nabla \mathbf{u} &= \mathbf{0}, \\ \frac{\partial \mathbf{u}}{\partial t} + \boldsymbol{\nabla} \cdot \mathbf{F}(\mathbf{u}, \mathbf{q}) &= \mathbf{0}. \end{aligned} \tag{2}$$

The space and time discretization is outlined in the following sections.

2.1 Spatial

This work considers two implementations of the discontinuous Galerkin finite element method. The first one is a modal-based DG solver operating on a triangulation $\mathcal{T}_h$ of the domain Ω. The state vector is approximated by a polynomial expansion with no continuity constraints imposed between adjacent elements: $\mathbf{u}_h \in [\mathcal{V}_h]^M$ where $\mathcal{V}_h = \{u \in L_2(\Omega) : u|_K \in \mathbb{P}_k, \forall K \in \mathcal{T}_h\}$ and k is the order of polynomial approximation. The weak-form of (1) follows from multiplying the PDE by test functions in the same approximation space, integrating by parts, and coupling elements via consistent and stable numerical fluxes. By following this procedure a system of ordinary differential equations for the degrees of freedom (DoFs) of the problem can be written in the form

$$\mathbf{M}\frac{d\mathbf{U}}{dt} + \mathbf{R} = \mathbf{0}, \tag{3}$$

where $\mathbf{M}$ is the mass matrix, $\mathbf{U}$ the vector of DoFs and $\mathbf{R}$ the residuals vector.

The second spatial discretization is the HDG method [7]. The HDG discretization approximates the variables $\mathbf{u}_h, \mathbf{q}_h$ with $\mathbf{u}_h \in [\mathcal{V}_h]^M$ and $\mathbf{q}_h \in [\mathcal{V}]^{M\times N}$. Moreover, an additional variable $\boldsymbol{\lambda}_h \in [\mathcal{M}_h]^M$ is defined in the space $\mathcal{M}_h = \{\lambda \in L_2(\mathcal{F}_h) : \lambda|_{\sigma_f} \in \mathbb{P}^k, \forall \sigma_f \in \mathcal{F}_h\}$, where $\mathcal{F}_h$ is the set of interior faces σ_f, and $\mathbb{P}_k$ is the space of polynomials of order k on face σ_f. The weak form is obtained in this case by weighting the equations in (2) with appropriate test functions, integrating by parts, and using the interface variable $\boldsymbol{\lambda}_h$ for the face state. A consistent and stable flux function is introduced at the mesh element interfaces, where continuity of the flux is ensured by additional equations required to close the system. See [7] for further details. Defining $\mathbf{R}^Q$, $\mathbf{R}^U$ and $\mathbf{R}^\Lambda$ the residuals vectors arising from the gradient equation, NS equations and flux-consistency equations, the ODE system of equations can be written as

$$\begin{aligned} \mathbf{R}^Q &= \mathbf{0}, \\ \mathbf{M}^U\frac{d\mathbf{U}}{dt} + \mathbf{R}^\mathbf{U} &= \mathbf{0}, \\ \mathbf{R}^\Lambda &= \mathbf{0}. \end{aligned} \tag{4}$$

where $\mathbf{M}^U$ is the elemental mass matrix. Defining the solution vector of the discrete unkown of the approximation as $\mathbf{W} = [\mathbf{Q}; \mathbf{U}; \boldsymbol{\Lambda}]$, and the vector of agglomerated residuals $\mathbf{R} = [\mathbf{R}^Q; \mathbf{R}^U; \mathbf{R}^\Lambda]$, the compact form of (4) becomes

$$\mathbf{M}\frac{d\mathbf{W}}{dt} + \mathbf{R} = \mathbf{0}, \tag{5}$$

where the matrix $\mathbf{M}$ and the vector $\mathbf{W}$ follow directly from equation (4). It is worth noticing that, for a DG discretization, $\mathbf{W} = \mathbf{U}$ and $\mathbf{M} = \mathbf{M}^U$.

2.2 Temporal

The temporal discretization is an explicit-single-diagonal-implicit Runge–Kutta (ESDIRK) scheme. The general formulation of the scheme for equation (5) is

$$\begin{aligned} \mathbf{M}\mathbf{W}^i &= \mathbf{M}\mathbf{W}^n - \Delta t \sum_{j=1}^{i} a_{ij}\mathbf{R}(\mathbf{W}^j), \\ \mathbf{W}^{n+1} &= \mathbf{W}^n + \sum_{i=1}^{s} \beta_i \Delta t \mathbf{W}^i, \end{aligned} \tag{6}$$

for $i = 1, ..., s$ where s is the number of stages, a_{ij} and b_i are the coefficients of the scheme, and n is the time index. Within each stage, the solution of a non-linear system is required; to this end, the Newton-Krylov method is used and the k^{th} Newton–Krylov iteration assumes the form

$$\left(\frac{\mathbf{M}}{a_{ii}\Delta t} + \frac{\partial \mathbf{R}}{\partial \mathbf{W}}\right)(\mathbf{W}^i_{k+1} - \mathbf{W}^i_k) = -\frac{\mathbf{M}}{a_{ii}\Delta t}(\mathbf{W}^i_k - \mathbf{W}^n) - \sum_{j=1}^{i-1} \frac{a_{ij}}{a_{ii}}\mathbf{R}(\mathbf{W}^j) - \mathbf{R}(\mathbf{W}^i_k), \tag{7}$$

with $i = 1, ..., s$. In this work the order-three ESDIRK3 scheme [9] will be employed.

3 LINEAR SOLVER

For a DG discretization, system (7) is solved iteratively without any additional operations, and assumes the form

$$\mathbf{A}\mathbf{x} + \mathbf{b} = \mathbf{0}, \tag{8}$$

with $\mathbf{A} = (\mathbf{M}/(a_{ii}\Delta t) + \partial\mathbf{R}/\partial\mathbf{W})$ the iteration matrix, $\mathbf{x} = \Delta\mathbf{W}$ the vector of the degrees of freedom update and $\mathbf{b}$ the right-hand side.

On the other hand, the HDG discretization exploits the introduction of face unknowns in order to reduce the size of the matrix to be allocated, see [7] for further details. In fact, system (7) can be conveninetly arranged using the definition of gradients residuals, state residuals and trace variables residuals as

$$\begin{bmatrix} \mathbf{A}^{QQ} & \mathbf{A}^{QU} & \mathbf{B}^{Q\Lambda} \\ \mathbf{A}^{UQ} & \mathbf{A}^{UU} & \mathbf{B}^{U\Lambda} \\ \mathbf{C}^{\Lambda Q} & \mathbf{C}^{\Lambda U} & \mathbf{D} \end{bmatrix} \begin{pmatrix} \Delta\mathbf{Q} \\ \Delta\mathbf{U} \\ \Delta\mathbf{\Lambda} \end{pmatrix} + \begin{pmatrix} \mathbf{b}^Q \\ \mathbf{b}^U \\ \mathbf{b}^\Lambda \end{pmatrix} = \mathbf{0}, \tag{9}$$

where the block-structure of the iteration matrix and the right-hand side can be easily derived from (4).

The solution of the system is obtained by statically condensing out the element-interior variables, resulting in the following problem

$$\begin{aligned} &\left(\mathbf{D} - \begin{bmatrix} \mathbf{C}^{\Lambda Q}\mathbf{C}^{\Lambda U} \end{bmatrix} \begin{bmatrix} \mathbf{A}^{QQ} & \mathbf{A}^{QU} \\ \mathbf{A}^{UQ} & \mathbf{A}^{UU} \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{B}^{Q\Lambda} \\ \mathbf{B}^{U\Lambda} \end{bmatrix}\right) \Delta\mathbf{\Lambda} + \\ &\left(\mathbf{b}^\Lambda - \begin{bmatrix} \mathbf{C}^{\Lambda Q}\mathbf{C}^{\Lambda U} \end{bmatrix} \begin{bmatrix} \mathbf{A}^{QQ} & \mathbf{A}^{QU} \\ \mathbf{A}^{UQ} & \mathbf{A}^{UU} \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{b}^Q \\ \mathbf{b}^U \end{bmatrix}\right) = \mathbf{0}, \end{aligned} \tag{10}$$

which assumes the same form as system (8). It is worth noticing that in HDG the memory allocation and the time spent on global solve are lower than that of a DG solver due to the smaller amount of globally coupled degrees of freedom. On the other hand, the number of elemental operations is higher. In fact, the inversion of the $\mathbf{A}^{ij}$ block-structured matrix of equation (10), although being local to each element is still not a trivial cost. Note that the element-interior states have to be recovered after the linear solve to evaluate the residuals vector in the successive iteration using a back-solve strategy. See [7] for further details.

The linear system of ODEs can be solved numerically using iterative solvers. To this end, we employ the generalized minimal residual method (GMRES). Three preconditioner matrices are considered in the remaining of the paper to speed-up the iterative process. The first one is element-wise block-Jacobi (BJ), which extracts the block-diagonal part of the iteration matrix and takes the LU factorization in a local-to-each element fashion. The second one [10] is line-Jacobi (LJ), which creates within the mesh lines of elements of maximum coupling and solves implicitly the degrees of freedom for each line. The last one is the incomplete lower-upper factorization with zero-fill, $\text{ILU}_0(\mathbf{A})$, and minimum discarded fill reordering [11]. When applied in parallel, the $\text{ILU}_0(\mathbf{A})$ is performed on each square, partition-wise block of the iteration matrix.

An important characteristics of GMRES is that any iterative linear solver, such as multigrid, can be used as a preconditioner. In such case, the flexible implementation of GMRES will be employed.

4 P-MULTIGRID PRECONDITIONING

In this work, the use of a p-multigrid strategy is explored in the context of the space discretizations presented. Coarser linear systems $\mathbf{A}_i\mathbf{x}_i = \mathbf{b}_i$ are built using lower-order polynomial spaces. Subspace inheritance [5] is used to generate coarse grid operators, *i.e.* the matrices $\mathbf{A}_i$, both for the DG and HDG discretizations. This choice involves projection of the iteration matrix, which is computed only once on the finest level. Compared to subspace non-inheritance, which requires the recomputation of the Jacobian in proper coarser-space discretizations of the problem, inheritate is cheaper in processing and memory. Although previous work has shown lower convergence rates when using such cheaper operators [12], especially in the context of elliptic problems and incompressible flows, we found these operators efficient enough for our target problems involving the compressible NS equations.

The multigrid strategy requires the definition of both restriction and prolongation operators to project the vectors of degrees of freedom between the polynomial spaces. To do so, a distinction between the DG and HDG space discretization has to be performed. In fact, while in the former case the linear solver works with element-wise unknowns, in the latter the element-interior coefficients are statically condensed out, and the system is solved for the face unknowns only. As regards DG, let us define a sequence of approximation spaces $\mathcal{V}_\ell \supseteq \mathcal{V}_h$ on the same triangulation $\mathcal{T}_h$, ℓ being a multigrid level, with $\mathcal{V}_h = \mathcal{V}_1 \subset \mathcal{V}_2 \subset ... \subset \mathcal{V}_{N_{LVL}}$ and N_{LVL} the total number of levels. Note that $\mathcal{V}_{N_{LVL}}$ denotes the coarsest space. In this context, the prolongation operator can be defined as

$\mathcal{I}_{\ell+1}^{\ell} : \mathcal{V}_{\ell+1} \to \mathcal{V}_{\ell}$ such that

$$\sum_{K\in\mathcal{T}_h} \int_K \left(\mathcal{I}_{\ell+1}^{\ell} u_{\ell+1} - u_{\ell+1}\right) = 0, \quad \forall u_{\ell+1} \in \mathcal{V}_{\ell+1}. \tag{11}$$

Similarly, the restriction operator can be defined as the L_2 projection $\mathcal{I}_{\ell}^{\ell+1} : \mathcal{V}_{\ell} \to \mathcal{V}_{\ell+1}$ such that

$$\sum_{K\in\mathcal{T}_h} \int_K \left(\mathcal{I}_{\ell}^{\ell+1} u_{\ell} - u_{\ell}\right) v_{\ell+1} = 0, \quad \forall (u_{\ell}, v_{\ell+1}) \in \mathcal{V}_{\ell} \times \mathcal{V}_{\ell+1}. \tag{12}$$

Such definitions can be easily extended to build discrete matrix operators $\mathbf{I}_i^j$ that project the vectors and matrices in the coarser levels.

Similar considerations can be done for the face unknowns of the HDG discretization. In this case the sequence of approximation spaces $\mathcal{M}_{\ell} \supseteq \mathcal{M}_h$ is properly defined on the interior mesh element faces $\mathcal{F}_h$, being ℓ a multigrid level. To this end, let us define $\mathcal{M}_h = \mathcal{M}_1 \subset \mathcal{M}_2 \subset ... \subset \mathcal{M}_{N_{LVL}}$. The prolongation operator is now $\mathcal{I}_{\ell+1}^{\ell} : \mathcal{M}_{\ell+1} \to \mathcal{M}_{\ell}$ such that

$$\sum_{F\in\mathcal{F}_h} \int_F \left(\mathcal{I}_{\ell+1}^{\ell} \lambda_{\ell+1} - \lambda_{\ell+1}\right) = 0, \quad \forall \lambda_{\ell+1} \in \mathcal{M}_{\ell+1}. \tag{13}$$

On the other hand, the restriction is defined as $\mathcal{I}_{\ell}^{\ell+1} : \mathcal{M}_{\ell} \to \mathcal{M}_{\ell+1}$ such that

$$\sum_{F\in\mathcal{F}_h} \int_F \left(\mathcal{I}_{\ell}^{\ell+1} \lambda_{\ell} - \lambda_{\ell}\right) \mu_{\ell+1} = 0, \quad \forall (\lambda_{\ell}, \mu_{\ell+1}) \in \mathcal{M}_{\ell} \times \mathcal{M}_{\ell+1}. \tag{14}$$

These definitions can also be extended to build matrix operators $\mathbf{I}_i^j$ that transform the vectors, residuals, and block matrices shown in equation (10). Note that the projection of the fine-space condensed matrix and the right hand side differs from performing a static condensation of the projected matrices and vectors in coarse space. Despite this discrepancy, the results show that a considerably large reduction of the number of iterations is achieved.

We employ a full multigrid (FMG) V-Cycle solver, outlined in Algorithm 1. The FMG cycle constructs a good initial guess for a V-Cycle iteration which starts on the fine grid. To do so, the solution is initially solved on the coarsest level (N_{LVL}), and then prolongated to the next refined one ($N_{LVL}-1$). At this point, a standard V-Cycle is called, such that an improved approximation of the solution can be used for the V-Cycle at level $N_{LVL}-2$. This procedure is repeated until the V-Cycle on the finest level is completed. The single V-Cycle is outlined in Algorithm 2. Starting from a level ℓ, the solution is initially smoothed using an iterative solver (SMOOTH). The residual of the solution $\mathbf{r}_{\ell}$ is then computed and projected in the coarser level $\ell+1$, where another V-Cycle is recursively called to obtain a coarse-grid correction $\mathbf{e}_{\ell+1}$. The quantity is then prolongated on to level ℓ and used to correct the solution to be smoothed again. When the coarsest level is reached, the problem is solved with a higher number of iterations to decrease as much as possible the solution error.

Algorithm 1 *FMG*

```
 1: for $\ell = N_{LVL}, 1, -1$ do
 2:     if $\ell = N_{LVL}$ then
 3:         $\mathbf{b}_\ell = \mathbf{I}_1^\ell \mathbf{b}_1$
 4:         SOLVE $\mathbf{A}_\ell \mathbf{x}_\ell^{FMG} = \mathbf{b}_\ell$
 5:     else
 6:         $\mathbf{b}_\ell = \mathbf{I}_1^\ell \mathbf{b}_1$
 7:         $\mathbf{x}_\ell^0 = \mathbf{I}_{\ell+1}^\ell \mathbf{x}_{\ell+1}^{FMG}$
 8:         $\mathbf{x}_\ell^{FMG} = MG_V(\ell, \mathbf{b}_\ell, \mathbf{x}_\ell^0)$
 9:     end if
10: end for
11: return $\mathbf{x}_\ell^{FMG}$
```

Algorithm 2 $MG_V(\ell, \mathbf{b}_\ell, \mathbf{x}_\ell)$

```
 1: if $\ell = N_{LVL}$ then
 2:     SOLVE $\mathbf{A}_\ell \overline{\mathbf{x}}_\ell = \mathbf{b}_\ell$
 3: else
 4:     $\overline{\mathbf{x}}_\ell$=SMOOTH($\mathbf{x}_\ell, \mathbf{A}_\ell, \mathbf{b}_\ell$)
 5:     $\mathbf{r}_\ell = \mathbf{b}_\ell - \mathbf{A}_\ell \overline{\mathbf{x}}_\ell$
 6:     $\mathbf{r}_{\ell+1} = \mathbf{I}_\ell^{\ell+1} \mathbf{r}_\ell$
 7:     $\mathbf{e}_{\ell+1}$=$MG_V(\ell+1, \mathbf{r}_{\ell+1}, \mathbf{0})$
 8:     $\hat{\mathbf{x}}_\ell = \overline{\mathbf{x}}_\ell + \mathbf{I}_{\ell+1}^\ell \mathbf{e}_{\ell+1}$
 9:     $\overline{\mathbf{x}}_\ell$=SMOOTH($\hat{\mathbf{x}}_\ell, \mathbf{A}_\ell, \mathbf{b}_\ell$)
10: end if
11: return $\overline{\mathbf{x}}_\ell$
```

5 NUMERICAL RESULTS

We present numerical experiments to assess the behaviour of the p-multigrid preconditioner for the DG and HDG discretizations. First, NS solutions of a vortex transported by uniform flow at $M = 0.05$ and $Re = 100$ are reported to show the convergence rates in space and time. The second test case deals with the solution of a two-dimensional circular cylinder at $Re = 100$ and $M = 0.2$, and it is used to evaluate parallel efficiency.

5.1 Convected vortex

The test case is a modified version of the VI1 case studied in the 5th International Workshop on High Order CFD Methods [13], and consists of a two-dimensional mesh on the domain $(x, y) \in [0, 0.1] \times [0, 0.1]$ with periodic boundary conditions on each side. See the online web page for information about the flow initialization. Here the set of NS equations are solved instead of the Euler equations. Numerical experiments have been performed to assess the output error, both in space and time. The meshes were obtained in a structured-like manner using regular quadrilaterals, starting from 2×2 up to 32×32. The polynomial spaces employed were $\mathbb{P}_4$, $\mathbb{P}_5$ and $\mathbb{P}_6$. The L_2 state error was computed relative to the solution on a 128×128, $\mathbb{P}_6$ space discretization, after one convective period T. The contour plot of the solutions at the initial and final states are shown in Figure 1. Table 1 reports space discretization errors. The tests were performed using a very small time step size, $T/\Delta t = 4000$, and the ESDIRK3 scheme to ensure a negligible time discretization error. An absolute tolerance on the non-linear system of 10^{-10}, and a relative tolerance of 10^{-5} on the GMRES, were employed to ensure a low time discretization error, Even though DG suffers less than HDG of pre-asympthotic behaviour on such a smooth solution, both the implementations show comparable error levels and converge with the theoretical convergence rates for every polynomial approximation shown. As a consequence of such analysis, and considering that both the DG and HDG implementations share the same code base, we will consider only the CPU time as a measure of the time-to-solution efficiency.

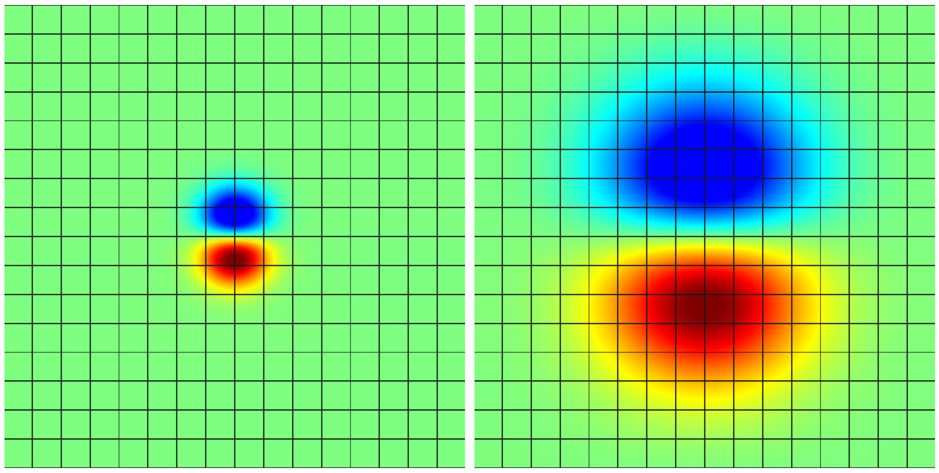

Figure 1: Convected vortex at $Re = 100$, $M = 0.05$. Mach number contours. Solution at $t = 0$ (left) and $t = T$ (right).

		DG		HDG	
order	grid	$\Vert err\Vert_{L_2}$	k	$\Vert err\Vert_{L_2}$	k
$\mathbb{P}_4$	2×2	1.12E-07		1.18E-07	
	4×4	5.25E-09	4.42	5.42E-09	4.45
	8×8	1.18E-10	5.48	1.99E-10	4.77
	16×16	6.56E-12	4.17	7.31E-12	4.77
	32×32	4.98E-12	0.40	5.22E-12	0.49
$\mathbb{P}_5$	2×2	4.27E-08		4.50E-08	
	4×4	6.65E-10	6.00	7.82E-10	5.85
	8×8	1.15E-11	5.85	9.60E-11	3.03
	16×16	5.07E-12	1.18	4.85E-12	4.31
	32×32	4.99E-12	0.03	5.46E-12	-0.17
$\mathbb{P}_6$	2×2	1.42E-08		1.35E-08	
	4×4	9.71E-11	7.19	3.44E-10	5.30
	8×8	5.12E-12	4.24	2.46E-11	3.80
	16×16	5.08E-12	0.01	5.10E-12	2.27
	32×32	5.03E-12	0.02	5.31E-12	-0.06

Table 1: L_2 solution error. Laminar vortex test case at $Re = 100$, $M = 0.05$. Convergence rates for the DG and HDG discretizations.

The convergence rates of the ESDIRK3 time integration scheme are also reported in Table 2, and have been obtained using the 16×16, $\mathbb{P}_6$ space discretization. A large non-dimensional time step of $\Delta t = 0.1$, was selected to assess the efficiency of the preconditioning strategies (BJ, LJ, ILU(0) and p-MG). A three-level Full multigrid V-Cycle was employed, with parameters determined empirically for best convergence. The coarser space discretizations were built using $\mathbb{P}_1$ and $\mathbb{P}_3$, while GMRES smoothers were used in each level. 10 iterations of BJ-preconditioned GMRES smoothers have been employed for

	DG		HDG	
$T/\Delta t$	$\Vert err\Vert_{L_2}$	k	$\Vert err\Vert_{L_2}$	k
4	5.39E-07		5.39E-07	
10	1.37E-07	1.50	1.37E-07	1.50
20	2.58E-08	2.40	2.58E-08	2.40
40	3.77E-09	2.78	3.77E-09	2.77
100	2.62E-10	2.91	2.60E-10	2.92
200	2.48E-10	0.08	2.55E-10	0.03

Table 2: Laminar vortex test case at $Re = 100$, $M = 0.05$. Time convergence rates for the DG and HDG discretizations and ESDIRK3 scheme.

	DG			HDG		
Prec.	CPU time	IT_a	ρ_a	CPU time	IT_a	ρ_a
ILU	6.67E+03	24.87	0.59277	6.04E+03	19.40	0.53163
LJ	4.12E+03	84.88	0.8453	6.15E+03	124.59	0.85552
BJ	3.63E+03	129.25	0.88282	6.10E+03	109.09	0.86254
p-MG	2.66E+03	2.01	0.00243	6.06E+03	1.40	0.00017

Table 3: Performance of the preconditioners using DG and HDG discretizations. Laminar vortex test case at $Re = 100$, $M = 0.05$, discretized on a 16×16 mesh with $\mathbb{P}_6$ polynomials, $T/\Delta t = 10$, ESDIRK3 scheme. IT_a stands for the average number of GMRES iterations while ρ_a the average convergece rate.

all the levels except from the coarsest, where 30 ILU$_0$-preconditioned GMRES iterations were performed. Such a setting was found adequate for all of the tests presented in this work, and for both space discretizations. The numerical experiments are reported in Table 3, which gives the CPU time, the average and maximum number of GMRES iterations though the time integration, as well as the average convergence rate (CR). The CR is defined as $\rho = (r_{IT}/r_0)^{1/IT}$ with IT the number of iteratons, r_0 and r_{IT} the residuals at the first and IT^{th} iteration respectively. We see that the use of a p-multigrid preconditioner reduces considerably the number of outer GMRES iterations both for the DG and the HDG space discretizations, but only in the DG case is this gain reflected on the CPU time. In fact, the costs of condensing-out the element-interior variables before the solution of the system, as well as the back-solve for their evaluation using the face unknowns after the linear solve, increase the amount of local operations and hide the benefits of having a faster global solver in HDG.

5.2 Circular cylinder

The laminar flow around a circular cylinder at Mach number $M = 0.2$ and Reynolds number $Re = 100$ has been solved on a grid made by $N_e = 960$ mesh elements using a $\mathbb{P}_6$ space discretization. The time scheme employed is ESDIRK3. Figure 2 shows a snapshot of the Mach number contours. The solution accuracy was assessed in comparison with literature data [14]. To this end, Table 4 shows the averaged drag, lift coefficients and Strouhal number (C_d, C_l, St) for several temporal refinements on the same grid. The

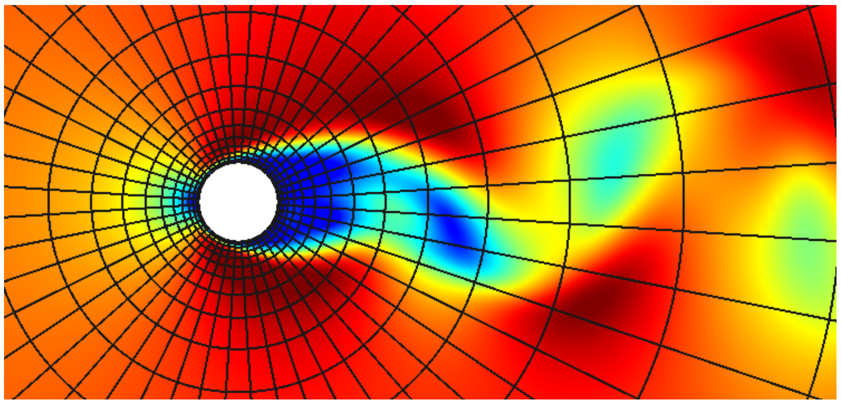

Figure 2: Laminar flow around a circular cylinder at $Re = 100$, $M = 0.2$. Mach number contours.

$(U/L)\Delta t$	C_d	C_l	St
0.5	1.3468	3.383e-03	0.16327
0.25	1.3519	-1.441e-03	0.16410
0.125	1.3527	-1.400e-04	0.16410
0.05	1.3528	-1.718e-04	0.16410
0.025	1.3528	-6.353e-06	0.16410

Table 4: Laminar vortex test case at $Re = 100$, $M = 0.05$. Time convergence rates for the DG and HDG discretizations and ESDIRK3 scheme.

coefficients were obtained by averaging a fully developed solution over ten shedding periods. An overall good agreement has been found, and temporal convergence can be seen by using a non-dimensional time step of $\Delta t \leq 0.25$. The numerical experiments are thus performed using $\Delta t = 0.25$ to maximize the advantage of using an implicit scheme for such type of problem.

The parallel performance of the p-multigrid preconditioner strategy introduced in the previous section is assessed by considering the effects of domain decomposition. To evaluate the CPU time, a fully-developed solution is advanced in time for 10 time steps to compute the average number of GMRES iterations and the convergence rates during the non-linear solution. The computations are performed using 1 to 48 cores (NP) on a platform based on Intel Xeon X5650 processors aranged in a two-processor (12 cores) per-node fashion. A fixed relative tolerance of 10^{-6} to stop the GMRES solver is used, as well as an absolute tolerance of 10^{-5} for the Newton-Raphson method.

Tables 5 and 6 report the results of the computations. For the ILU$_0$ preconditioner, an increase in the number of GMRES iterations is observed for both the DG and HDG space discretizations with an increasing number of cores. This behavior is attributed to the incomplete lower-upper factorization, which is performed on the squared, partition-wise block of the iteration matrix. In this case, the preconditioner effectiveness naturally decreases as NP grows, as the amount of off-diagonal blocks neglected increases with NP. On the other hand, by tuning the parameters of the multigrid preconditioner it is possible to achieve an ideal algorithmic scalability, *i.e.* the number of iterations required to solve

	DG			HDG		
NP	CPU time	IT_a	ρ_a	CPU time	IT_a	ρ_a
1	2.59E+04	20.507	0.47417	2.59E+04	17.807	0.43142
12	2.86E+03	76.767	0.80356	3.03E+03	38.08	0.67155
24	1.48E+03	83.867	0.81811	1.66E+03	43.827	0.70301
36	1.03E+03	94.68	0.83113	1.20E+03	46.713	0.71824
48	8.27E+02	112.567	0.85091	1.00E+03	50.913	0.7337

Table 5: Performance of the ILU_0 preconditioner using DG and HDG discretizations.

	DG			HDG		
NP	CPU time	IT_a	ρ_a	CPU time	IT_a	ρ_a
1	1.24E+04	3.68	0.01961	2.61E+04	2.52	0.00195
12	1.32E+03	3.69	0.01962	3.04E+03	2.59	0.00222
24	6.99E+02	3.69	0.01958	1.68E+03	2.58	0.00227
36	4.84E+02	3.69	0.01961	1.21E+03	2.60	0.00234
48	3.72E+02	3.69	0.01948	1.01E+03	2.60	0.00234

Table 6: Performance of the p-MG preconditioner using DG and HDG discretizations.

the system doesn't grow by partitioning the domain. This consideration holds true for both the DG and HDG discretizations. In terms of CPU time, in all the runs the p-MG preconditioned DG solver outperforms both the ILU_0-preconditioned DG solver as well as HDG. In particular, the computational time of HDG is still higher despite fewer linear iterations required on average to reach the same levels of accuracy.

6 CONCLUSIONS

An inherited p-multigrid strategy has been proposed and assessed in the context of DG and HDG discretizations. The algorithm has been employed as a preconditioner for an FGMRES solver. Compared to standard single-grid preconditioners, the approach is able to reduce significantly the number of iterations required for the linear system solution. This reduction is reflected in the overall CPU time only in the case of DG. For HDG, the static condensation and back-solve operations are expensive and hide the benefits of using such a strategy. Future works will be devoted to assess the influence of different multigrid settings, as well as to extend the current work by considering stiff three-dimensional cases.

REFERENCES

[1] M. Franciolini, A. Crivellini, and A. Nigro. On the efficiency of a matrix-free linearly implicit time integration strategy for high-order discontinuous Galerkin solutions of incompressible turbulent flows. *Computers & Fluids*, 159:276–294, 2017.

[2] J. S. Hesthaven and T. Warburton. *Nodal discontinuous Galerkin methods: algorithms, analysis, and applications*. Springer Science & Business Media, 2007.

[3] L. Wang and D. J. Mavriplis. Implicit solution of the unsteady Euler equations for high-order accurate discontinuous Galerkin discretizations. *Journal of Computational Physics*, 225(2):1994 – 2015, 2007.

[4] P. Birken, G. Gassner, M. Haas, and C. D. Munz. Preconditioning for modal discontinuous Galerkin methods for unsteady 3D Navier–Stokes equations. *Journal of Computational Physics*, 240:20–35, 2013.

[5] L. Botti, A. Colombo, and F. Bassi. *h*-multigrid agglomeration based solution strategies for discontinuous Galerkin discretizations of incompressible flow problems. *Journal of Computational Physics*, 347:382–415, 2017.

[6] N. C. Nguyen, J. Peraire, and B. Cockburn. An implicit high-order hybridizable discontinuous Galerkin method for nonlinear convection–diffusion equations. *Journal of Computational Physics*, 228(23):8841–8855, 2009.

[7] K. J. Fidkowski. A hybridized discontinuous Galerkin method on mapped deforming domains. *Computers & Fluids*, 139:80–91, 2016.

[8] B. Cockburn, O. Dubois, J. Gopalakrishnan, and S. Tan. Multigrid for an HDG method. *IMA Journal of Numerical Analysis*, 34(4):1386–1425, 2014.

[9] H. Bijl, M. H. Carpenter, V. N. Vatsa, and C. A. Kennedy. Implicit time integration schemes for the unsteady compressible Navier–Stokes equations: laminar flow. *Journal of Computational Physics*, 179(1):313–329, 2002.

[10] K. J. Fidkowski, T. A. Oliver, J. Lu, and D. L. Darmofal. *p*-Multigrid solution of high-order discontinuous Galerkin discretizations of the compressible Navier–Stokes equations. *Journal of Computational Physics*, 207(1):92–113, 2005.

[11] P.-O. Persson and J. Peraire. Newton-GMRES preconditioning for discontinuous Galerkin discretizations of the Navier–Stokes equations. *SIAM Journal on Scientific Computing*, 30(6):2709–2733, 2008.

[12] P. F. Antonietti, M. Sarti, and M. Verani. Multigrid algorithms for *hp*-discontinuous Galerkin discretizations of elliptic problems. *SIAM Journal on Numerical Analysis*, 53(1):598–618, 2015.

[13] 5^{th} International Workshop on High–Order CFD Methods. https://how5.cenaero.be/.

[14] J. R. Meneghini, F. Saltara, C. L. R. Siqueira, and J. A. Ferrari Jr. Numerical simulation of flow interference between two circular cylinders in tandem and side-by-side arrangements. *Journal of Fluids and Structures*, 15(2):327–350, 2001.