A p-adaptive Matrix-Free Discontinuous Galerkin Method for the Implicit LES of Incompressible Transitional Flows

(Article begins on next page)

28 December 2024

A p-adaptive matrix-free Discontinuous Galerkin method for the Implicit LES of incompressible transitional flows

F. Bassi · L. Botti · A. Colombo · A. Crivellini · M. Franciolini · A. Ghidoni · G. Noventa

Abstract In recent years Computational Fluid Dynamics (CFD) has become a widespread practice in industry. The growing need to simulate off-design conditions, characterized by massively separated flows, strongly promoted research on models and methods to improve the computational efficiency and to bring the practice of Scale Resolving Simulations (SRS), like the Large Eddy Simulation (LES), to an industrial level. Among the possible approaches to the SRS, an appealing choice is to perform Implicit LES (ILES) via a high-order Discontinuous Galerkin (DG) method, where the favourable numerical dissipation of the space discretization scheme plays directly the role of a subgrid-scale model. To reduce the large CPU time for ILES, implicit time integrators, that allows for larger time steps than explicit schemes, can be considered. The main drawbacks of implicit time integration in a DG framework are represented by the large memory footprint, the large CPU time for the operator assembly and the difficulty to design highly scalable preconditioners for the linear solvers. In this paper, which aims to significantly reduce the memory requirement and CPU time without spoiling the high-order accuracy of the method, we rely on a p-adaptive algorithm suited for the ILES of turbulent flows and an efficient matrix-free iterative linear solver based on a cheap p-multigrid preconditioner

F. Bassi, L. Botti, A. Colombo* (corresponding author)
University of Bergamo, Dalmine (BG), 24044, Italy
E-mail: {francesco.bassi, lorenzo.botti, alessandro.colombo}@unibg.it

A. Crivellini
Polytechnic University of Marche, Ancona, 60131, Italy
E-mail: a.crivellini@univpm.it

M. Franciolini
NASA Ames Research Center, Moffett Field (CA), 94035, United States
E-mail: matteo.franciolini@nasa.gov

A. Ghidoni, G. Noventa
University of Brescia, Brescia, 25123, Italy
E-mail: {antonio.ghidoni, gianmaria.noventa}@unibs.it

and a Flexible GMRES method. The performance and accuracy of the method have been assessed by considering the following test cases: *i*) the T3L test case of the ERCOFTAC suite, a rounded leading edge flat plate at $\mathrm{Re}_D = 3\,450$; *ii*) the flow past a sphere at $Re_D = 300$; *iii*) the flow past a circular cylinder at $Re_D = 3\,900$.

Keywords Discontinuous Galerkin · ILES · incompressible flows · Rosenbrock-type schemes · matrix-free · *p*-multigrid preconditioner · *p*-adaptation

1 Motivation

Many industrial applications, *e.g.*, aircraft and turbomachinery, are characterized by chaotic flow regimes, and are often simulated by using statistical models. These models solve the set of the Navier-Stokes equations averaged in time, *i.e.*, the Reynolds averaged Navier-Stokes (RANS) equations. RANS approach allows accurately simulating attached turbulent flows around complex geometries but it usually fails the prediction of massively separated flows, which is crucial for the off-design performance in engineering practice. To overcome this limitation, Scale-Resolving Simulations (SRS) based on the Large Eddy Simulation (LES) can be used. However, their computational cost is still too large for a routine use in industry, and is promoting the research on innovative physical models and numerical methods to enhance their computational efficiency.

In this paper, some promising strategies are investigated and developed to increase the computational efficiency of a discontinuous Galerkin (DG) solver [7,9] for scale-resolving simulations based on the Implicit Large Eddy Simulation (ILES), or under-resolved Direct Numerical Simulation (uDNS), approach. In fact, high-order DG finite elements methods proved to be very well suited [66,67,48,8,42,47,56,9,30] for ILES, where the unfiltered incompressible Navier-Stokes equations are solved, and the numerical dissipation introduced by the discretizaton itself, *e.g.*, by the Riemann interface fluxes and the viscous stabilization, plays the role of an explicit subgrid-scale (SGS) model that dissipates the smallest scale eddies.

The incompressible DG solver adopted for this work is based on a peculiar formulation for the inviscid interface numerical flux [15], computed as the exact solution of the Riemann problem modified by an artificial compressibility perturbation, which provides the necessary coupling between the discrete incompressibility constraint and the rest of the governing equations. The non linear system of differential algebraic equations arising from the spatial discretization is advanced in time using an implicit high-order time integration scheme [11], which allows to implicitly satisfy the incompressibility constraint and increase the time step size with respect to semi-implicit or projection methods, commonly employed for incompressible flows. However, the main drawbacks of such schemes are a large memory footprint, high computational time to assemble implicit operators, and the difficulty to design highly scalable

preconditioners. To alleviate such disadvantages, the following strategies are considered: *i*) the local adaptation of the polynomial degree of the solution over the mesh, *ii*) a multi-constraint domain decomposition algorithm to ensure the computational balance over the processes, *iii*) the adaptation of the degree of exactness for quadrature rules, and *iv*) a *p*-MG multilevel preconditioner for the linear solver. Since DG methods represent the numerical solution as polynomial functions inside the elements of a mesh with no continuity requirements at the cells interfaces, one can locally vary the order of accuracy very efficiently by using a different degree of the polynomial representation according to some error indicator. This practice is known in the literature as *p*-adaptation [49, 50, 14, 65]. The accuracy is increased/reduced automatically to capture the flow features only where needed, reducing dramatically the memory cost and the CPU time for the operator assembly. Moreover, the tedious task of the mesh generation is strongly simplified, as the local mesh refinements related to *a priori* assumptions on the flow features are avoided. This strategy can be fruitfully exploited in a parallel computing environment by the use of a balanced re-partitioning of the computational grid after each adaptation cycle, to overcome the natural imbalance of floating point operations per domain induced by the adaptation. In our implementation we also consider that algorithms for high-order mesh generation can produce computational grids made of both strongly curved- and (almost) straight-sided elements. The linear elements within the mesh are often not known *a-priori* from the grid format, *e.g.*, in the case of agglomerated high-order meshes built on top of block-structured grids, but their over-integration should be avoided to end up with an efficient implementation of the DG method. To this purpose, in this work we use an approach to locally adapt the degree of exactness of quadrature rules according to the mesh characteristics with significant saving in the operator assembly. Finally, we make use of a multilevel preconditioner recently developed in [27], based on element-wise approximations of the implicit operators obtained by neglecting off-diagonal blocks coupled with a matrix-free (MF) [24, 28] FGMRES iterative solver, to strongly reduce the memory requirements and the operation count. However, due to the variable polynomial degree over the mesh, we extend the original approach taking into account for this peculiarity on the finest level, while coarser levels are assembled through uniform-degree matrix-based operators.

    The performance and accuracy of the method have been assessed by considering the following test cases: *i*) the T3L ERCOFTAC case with different levels of inlet turbulence, *ii*) the flow past a sphere at $Re_D = 300$, and *iii*) the flow past a circular cylinder at $Re_D = 3\,900$. The T3L case has been used to show how a *p*-multigrid preconditioner for the linear systems solution can improve the solver efficiency. Moreover, this test case has been also used to demonstrate for different turbulence intensity values the good properties of the present DG method for the ILES approach. After demonstrating the computational efficiency of the underlying DG method, the effect of the *p*-adaptive strategy on the solver performance was evaluated by computing the unsteady laminar flow past a sphere at $Re_D = 300$. In particular, the *p*-adaptive sim-

ulation shows better performance for a given target accuracy with respect to uniform-degree simulations, both in terms of the overall number of DOFs and CPU time. Finally, as a proof of concept, the *p*-adaptive ILES of the turbulent flow past a circular cylinder at Reynolds number $\mathrm{Re}_D = 3\,900$ is considered.

2 The numerical framework

In this section the spatial and temporal discretizations will be presented with special emphasis on: *i*) a linear systems solution strategy via a Flexible GM-RES method preconditioned with a *p*-multigrid strategy; *ii*) a *p*-adaptation algorithm suited for scale-resolving simulation of turbulent flows.

2.1 DG spatial discretization

The set of governing equations for the DNS and Implicit LES (ILES) is simply the set of the discretized Navier-Stokes equations. In compact form the incompressible Navier-Stokes (INS) equations can be written as

$$\mathbf{P}\frac{\partial \mathbf{w}}{\partial t} + \nabla \cdot \mathbf{F}_{\mathrm{c}}\left(\mathbf{w}\right) + \nabla \cdot \mathbf{F}_{\mathrm{v}}\left(\mathbf{w}, \nabla \mathbf{w}\right) = \mathbf{0}, \tag{1}$$

where m is the number of variables, d the number of dimensions, $\mathbf{w} = [p, u_i]^T \in \mathbb{R}^m$ the unknown solution vector, $\mathbf{F}_c, \mathbf{F}_v \in \mathbb{R}^m \otimes \mathbb{R}^d$ the convective and viscous flux functions. $\mathbf{P} = \mathbf{I} - \mathbf{J}^{11}$ is the difference between the identity and a single-entry matrix, $\mathbf{J}^{11}$, defined as the matrix with 1 at the $(1,1)$ entry and zero elsewhere.

The governing equations are spatially discretized according to the DG method [9], where the weak form of the equations is obtained by multiplying Eq. (1) by an arbitrary test function and integrating by parts. The solution and the test function are then replaced with a finite element approximation and a discrete test function, which belong to $\mathbf{V}_h := [\mathbb{P}_d^k(\mathcal{K}_h)]^m$, where $\mathbb{P}_d^k(\mathcal{K}_h) := \{v_h \in L^2(\Omega) \,|\, v_{h|K} \in \mathbb{P}_d^k(K), \forall K \in \mathcal{K}_h\}$ is the discrete polynomial space in physical coordinates. $\mathbb{P}_d^k(K)$ denotes the restriction of the polynomial functions of d variables and degree at most k to the element K, part of the discretization $\mathcal{K}_h = \{K\}$ of the flow domain $\Omega \in \mathbb{R}^d$. In case of adaptive computations, the polynomial space will be denoted as $\mathbb{P}_d^{k_m \to k_M}$, where k_m and k_M are the minimum and maximum polynomial degree over the discretization, respectively. k_K denotes the polynomial degree local to any element $K \in \mathcal{K}_h$.

In this work we use a set of hierarchical and orthonormal basis functions. For all the $K \in \mathcal{K}_h$ the basis $\Phi_K^k = \left\{\phi_i^K\right\}$, where $i \in \{1, \ldots, N_{dof}^K\}$, is obtained by applying the modified Gram–Schmidt (MGS) orthogonalization algorithm to a set of monomials defined in a reference frame centered in the element barycenter and aligned with the principal axes of inertia of the element. The orthonormalization procedure can be expressed as the "composition" of the

monomial basis with a set of coefficients for the MGS algorithm, see for details Bassi et al. [10].

Each component $j = 1, \ldots, m$ of the numerical solution $\mathbf{w}_h$ over any element K can be expressed, in terms of the elements of the global vector $\mathbf{W}$ of unknown degrees of freedom, as $\mathbf{w}_{h,j}^K = \phi_l^K W_{j,l}^K$ where $\mathbf{W}^K$ is the local part of the global vector storing the degrees of freedom of the element K and the repeated index l implies summation over the $l = 1, \ldots, N_{dof}^K$. According to this notation, the DG spatial discretization of the system in Eq. (1) consists in seeking, for $j = 1, \ldots, m$, the elements of $\mathbf{W}$ such that

$$\sum_{K \in \mathcal{T}_h} \int_K \phi_i^K P_{j,k} \, \phi_l^K \frac{dW_{k,l}^K}{dt} d\mathbf{x} - \sum_{K \in \mathcal{T}_h} \int_K \frac{\partial \phi_i^K}{\partial x_n} F_{j,n} \left(\mathbf{w}_h, \nabla_h \mathbf{w}_h + \mathbf{r} \left([\![\mathbf{w}_h]\!] \right) \right) d\mathbf{x}$$

$$+ \sum_{F \in \mathcal{F}_h} \int_F [\![\phi_i^K]\!]_n \, \widehat{F}_{j,n} \left(\mathbf{w}_h^\pm, \left(\nabla_h \mathbf{w}_h + \eta_F \mathbf{r}_F \left([\![\mathbf{w}_h]\!] \right) \right)^\pm \right) d\sigma = 0, \quad (2)$$

for $i = 1, \ldots, N_{dof}^K$, where $\mathbf{F}$ is the sum of the convective and viscous flux functions, $\mathbf{n}$ is the unit vector normal to the boundary, $[\![\cdot]\!]$ and $\{ \cdot \}$ are the jump and average trace operators [9], respectively. In Eq. (2) repeated indices imply summation over the ranges $k = 1, \ldots, m$, $l = 1, \ldots, N_{dof}^K$, $n = 1, \ldots, d$. As the solution is discontinuous at mesh faces, the convective and viscous fluxes are not uniquely defined and need to be replaced with numerical flux functions, $\hat{\mathbf{F}}$. For the convective part, the approach proposed in [15] is used, whereby the numerical flux is computed from the exact solution of local Riemann problems suitably modified by means of an artificial compressibility perturbation. Viscous fluxes are discretized according to the BR2 method, proposed in [16] and theoretically analyzed in [20,3]. The operators $\mathbf{r}_F$ and $\mathbf{r}$ are the local and global lifting operators arising from the BR2 scheme [16]. The numerical dissipation associated to the Riemann fluxes and viscous stabilization replace in the ILES context the subgrid scale model of LES.

2.2 Linearly-implicit time discretization

Numerical integration of Eq. (2) via Gauss quadrature rules leads to a system of nonlinear Differential Algebraic Equations (DAEs) that can be written as

$$\mathbf{M_P} \frac{d\mathbf{W}}{dt} + \mathbf{R} \left(\mathbf{W} \right) = \mathbf{0}, \quad (3)$$

where $\mathbf{R} \left(\mathbf{W} \right)$ is the vector of residuals of the spatial discretization and $\mathbf{M_P}$ is a global block diagonal matrix arising from the discretization of the first term in Eq. (2). The choice of orthonormal basis functions defined in the mesh space allows reducing $\mathbf{M_P}$ to a modified identity matrix with zeros in the diagonal positions corresponding to the pressure degrees of freedom.

In this work accurate time integration is performed via linearly-implicit Rosenbrock-type Runge-Kutta schemes as follows

$$\mathbf{W}^{n+1} = \mathbf{W}^n + \sum_{j=1}^{s} m_j \mathbf{Y}_j, \tag{4}$$

$$\left[\frac{\mathbf{M_P}}{\gamma \Delta t} + \frac{\partial \mathbf{R}(\mathbf{W}^n)}{\partial \mathbf{W}} \right] \mathbf{Y}_i = -\mathbf{R} \left(\mathbf{W}^n + \sum_{j=1}^{i-1} a_{ij} \mathbf{Y}_j \right) + \frac{\mathbf{M_P}}{\Delta t} \sum_{j=1}^{i-1} c_{ij} \mathbf{Y}_j,$$
$$i = 1, \dots, s, \tag{5}$$

where m_j, γ, a_{ij} and c_{ij} are real coefficients, and s is the number of stages. Among the Rosenbrock schemes available in the literature [11,45], we rely on the ROS3P (third order, three stages) scheme of Lang and Verwer [40] or the ROSI2PW (third order, four stages) scheme of Rang and Angermann [60].

The PETSc library [4,5,6] is used for linear solvers, here preconditioned Generalized Minimal RESidual (GMRES) methods, and to devise a pure MPI parallel implementation for arrays, matrices and the communication among them.

2.3 The matrix-free iterative linear solver

The solution of the linear system through the GMRES iterative method requires only the evaluation of a matrix-vector product at each linear iteration. The Matrix-based (MB) GMRES approach calculates and stores both the complete implicit operator matrix and its preconditioner, while its matrix-free (MF) version approximates the matrix-vector products in Eq. (5) [36] as

$$\left[\frac{\mathbf{M_P}}{\gamma \Delta t} + \frac{\partial \mathbf{R}(\mathbf{W}^n)}{\partial \mathbf{W}} \right] \Delta \mathbf{W} \simeq \frac{\mathbf{G}(\mathbf{W}^n + h \Delta \mathbf{W}) - \mathbf{G}(\mathbf{W}^n)}{h}, \tag{6}$$

where

$$h = \varepsilon \frac{\sqrt{1 + \|\mathbf{W}^n\|_{L^2}}}{\|\Delta \mathbf{W}\|_{L^2}} \tag{7}$$

is a numerical perturbation [58], ε an user defined parameter here set to $\varepsilon = 10^{-9}$, and

$$\mathbf{G}(\mathbf{W}) = \frac{\mathbf{M_P}}{\gamma \Delta t} \mathbf{W} + \mathbf{R}(\mathbf{W}). \tag{8}$$

The finite differentiation error can show up, decreasing the efficiency of the solution process, only for small tolerances of the linear system solution when its residuals norm gets close to h [28]. However, such tight tolerances are not necessary for practical simulations [11,28] and the MF or MB GMRES need the same number of iterations to satisfy the target convergence.

The implicit operator assembly could be completely avoided in MF-GMRES approach, but the Jacobian evaluation may still be required for preconditioning in order to speed up the solution process on real applications. One of the

main advantages of a matrix-free implementation is that the computational cost for the matrix assembly can be reduced by approximating and/or "freezing" (lagging) the preconditioner for a given number of consecutive iterations without affecting the formal accuracy of the time integration scheme. This strategy can be exploited with some confidence when dealing with flow problems characterized by high-order polynomial approximation and small time steps, as in the case of the SRS of turbulent flows, see [28].

Widely used preconditioners for GMRES based on the factorization of the Jacobian matrix are: *i*) ASM(*i*,ILU(*j*)) - the Additive Schwarz domain decomposition Method (ASM) preconditioner with *i* levels of overlap between sub-domains and a block ILU decomposition for each sub-domain matrix with *j* levels of fill; *ii*) the Block Jacobi (BJ) - a block ILU decomposition for each sub-domain matrix with same fill level of the original matrix without overlapping elements; *iii*) EWBJ - the Element-wise Block Jacobi, a BJ preconditioner obtained by neglecting the off-diagonal blocks from the global matrix and applying the LU factorization of the diagonal blocks local to each element. Unfortunately, iterative solvers coupled with these standard preconditioners are not able to efficiently reduce the linear system solution error when applied to the incompressible Navier–Stokes equations, as a large number of iterations are typically required. The error can be expressed as a combination of "modes", and standard solvers are able to damp efficiently, in few iterations, only the oscillatory part, while smooth modes are damped very slowly. This behaviour results in a stall of the convergence as thoroughly discussed in Franciolini et al. [27] and summarized by the numerical results in Sec. 3.1.

2.4 The *p*-multigrid preconditioner

To speed up the linear system convergence, linear multigrid (MG) strategies can be exploited for the preconditioning [17, 18, 27, 29]. The basic idea of multigrid methods is to use coarser problems to smooth separately all the modes of the error. Indeed, low-frequency modes when transferred to coarser spaces become part of the high-frequency range and can be damped by standard iterative solvers (smoothers) [46, 32, 61, 26]. In multigrid methods the solution of the fine space problem can be accelerated by computing corrections on coarser spaces and then projecting them back to the finest space. The algorithm need proper operators to transfer functions from the fine to the coarse space (restriction) and vice versa (prolongation). In the *p*-multigrid context, coarser levels are defined by lower polynomial degree discretizations, see Ref. [27] for details.

Let L denote the number of multigrid levels, spanned by index $l = 1, \ldots, L$; where $l = 1$ and $l = L$ represent the finest and coarsest levels, respectively. Any coarse problem of Eq. (5) can be written in compact form as

$$\mathbf{A}_l \mathbf{W}_l = \mathbf{b}_l, \tag{9}$$

Algorithm 1 V-cycle p-MG algorithm, $\breve{\mathbf{W}}_l = MG_V(l, \mathbf{b}_l, \mathbf{W}_l)$

1: **if** $l = L$ **then**
2:      SOLVE $\mathbf{A}_l \breve{\mathbf{W}}_l = \mathbf{b}_l$
3: **else**
4:      $\breve{\mathbf{W}}_l$=SMOOTH($\mathbf{W}_l, \mathbf{A}_l, \mathbf{b}_l, \nu_1$)
5:      $\mathbf{r}_l = \mathbf{b}_l - \mathbf{A}_l \breve{\mathbf{W}}_\ell$
6:      $\mathbf{r}_{l+1} = \mathbf{I}_l^{l+1} \mathbf{r}_l$
7:      $\mathbf{e}_{l+1}$=$MG_V(l+1, \mathbf{r}_{l+1}, \mathbf{0})$
8:      $\hat{\mathbf{W}}_l = \breve{\mathbf{W}}_l + \mathbf{I}_{l+1}^l \mathbf{e}_{l+1}$
9:      $\breve{\mathbf{W}}_l$=SMOOTH($\hat{\mathbf{W}}_l, \mathbf{A}_l, \mathbf{b}_l, \nu_2$)
10: **end if**
11: **return** $\breve{\mathbf{W}}_l$

Algorithm 2 Full V-cycle p-MG algorithm

1: **for** $l = L, 1, -1$ **do**
2:      **if** $l = L$ **then**
3:          $\mathbf{b}_l = \mathbf{I}_1^l \mathbf{b}_1$
4:          SOLVE $\mathbf{A}_l \mathbf{W}_l^{FMG} = \mathbf{b}_l$
5:      **else**
6:          $\mathbf{b}_l = \mathbf{I}_1^l \mathbf{b}_1$
7:          $\tilde{\mathbf{W}}_l = \mathbf{I}_{l+1}^l \mathbf{W}_{l+1}^{FMG}$
8:          $\mathbf{W}_l^{FMG} = MG_V(l, \mathbf{b}_l, \tilde{\mathbf{W}}_l)$
9:      **end if**
10: **end for**
11: **return** $\mathbf{W}_l^{FMG}$

---

where $\mathbf{A}_l$ is the restriction to level l of the finest level matrix ($\mathbf{A}_1 = \mathbf{M}_\mathbf{P}/\gamma\Delta t + \partial\mathbf{R}/\partial\mathbf{W}$), $\mathbf{W}_l$ is the vector of the coefficients of the unknown polynomial functions on the coarse level l and $\mathbf{b}_l$ is the restriction of the finest level right hand side (RHS). In this work we adopted an inherited multigrid strategy, where coarse level operators are obtained by means of projection operators to avoid the explicit assembly of the residual and Jacobian operators on the coarse levels.

During the solution process the various levels can be used according to different paths. A common choice is the so called V-cycle described by the Algorithm 1. Starting from the finest level, a number ν_1 of pre-smoothing iterations is performed at each level prior to restricting the solution to the next coarser level. Once reached the coarsest level L, the problem is solved with a larger residual drop. On the way back to the finest level, a number ν_2 of post-smoothing iterations is performed at each level after updating the solution with the prolongated correction. In Algorithm 1, $\mathbf{I}_{l+1}^l$ and $\mathbf{I}_l^{l+1}$ denote the prolongation and restriction operators, and $\mathbf{e}_{l+1}$ is the coarse grid correction.

While satisfactory multigrid performance can be obtained by the use of standard V-Cycle, we observed a substantial improvement of the performance when the full multigrid (FMG) strategy is employed. As described in the Algorithm 2, in the FMG algorithm a V-Cycle is called in each level starting from the coarsest. At the end of each cycle, the solution is prolongated and used as initial guess in the next fine level V-Cycle. The process is terminated

when the convergence on the finest level is reached. We remark that the use of an orthonormal and hierarchical basis greatly simplify the transfer operators. In particular, the restriction consists in discarding the high-frequency modes of the solution that exceed those of the coarse space representation, while the prolongation consists in adding null high-order components to the low-order functions [27]. In this work multigrid is used as a preconditioner for a flexible GMRES (FGMRES) solver applied to the numerical solution of the linear systems arising from the temporal integration of the DG discretized INS equations. The use of FGMRES is mandatory since the action of the preconditioner changes at each time step and it needs to be stored in memory to compute accurately the final solution using Krylov subspaces. Several parameters of the implemented algorithms can be set by the user, such as the number of multigrid levels, L, and the degree of the polynomial approximation of the coarse levels, k_l, with $l = \{2, \dots, L\}$. For each level a different type of smoother and preconditioner can be chosen, as well as different tolerances. As suggested by previous numerical experiments [27], the following configuration is adopted, as it optimizes the computational efficiency of the *p*-MG algorithm:

– matrix-free GMRES smoother coupled with a EWBJ preconditioner on the finest space to reduce the memory footprint;
– matrix-based GMRES smoother on each coarse level;
– BJ or ASM preconditioners on the coarser levels to ensure a good enough convergence rate;
– lag all the matrix-based operators over some time step.

This approach allows computing the off-diagonal blocks of the Jacobian matrix, not required by the finest-level EWBJ preconditioner, at the lower order corresponding to the coarse $l = 2$ level, thus saving CPU time in the expensive task of the operator assembly.

2.5 Adaptation of the degree of exactness for quadrature rules

Algorithms for high-order mesh generation can produce computational grids where both curved- and straight-sided cells coexist. The linear elements within the mesh are often not known *a priori* from the grid format, as, for example, in the case of agglomerated high-order meshes built on top of block-structured grids. In some cases, cells are marked as curved although they are (almost) straight-sided. For an efficient implementation of the DG method, over-integration of linear elements should be avoided. Indeed, on curved elements the number of points needed for an accurate integration dramatically increases when dealing with high-order approximations of the solution.

In this work, inspired by Refs. [64,10], we locally adapt the degree of exactness (DOE) of quadrature rules according to the mesh characteristics. To this end, an estimate for the integration error of quadrature rules with a "reduced" number of points with respect to that formally needed to exactly integrate the mass matrix, is introduced. For an i-th diagonal entry of the mass matrix m_{ii},

we define

$$\varepsilon_{i,K} = \frac{|m_{ii}^* - m_{ii}^{ex}|}{|m_{ii}^{ex}|}, \quad \forall K \in \mathcal{K}_h, \tag{10}$$

where m_{ii}^{ex} and m_{ii}^* are the integral values computed using quadrature rules with an "exact" and a reduced DOE, respectively.

According to this definition, we can take into account of the actual curvature of cells and avoid over-integrating during operator assembly, with significant CPU time saving. In practice, for each element $K \in \mathcal{K}_h$, we pick an integration rule with the minimum degree of exactness that satisfies

$$\max_{i \in 1, \ldots, N_{dof}^K} \varepsilon_{i,K} \leq tol_q, \quad \forall K \in \mathcal{K}_h, \tag{11}$$

where tol_q is a user-defined tolerance. A similar procedure is used for surface integrals over each mesh face. The value formally needed to exactly integrate the mass matrix in case of straight-sided cells is set as a lower bound for the DOE.

Numerical tests on the L_2-projection of a given function on the polynomial space were performed to give some insight about the tolerance value. In particular, the projection error obtained with different sets of quadrature points, corresponding to different values for tol_q, shows a safe value around 10^{-10}, which allows reducing significantly the number of Gauss points without spoiling the quadrature accuracy.

Since in the present numerical framework, the system matrix accuracy is guaranteed by the matrix-free implementation of the linear solver, hence two different values for the tolerance can be considered. The former, $tol_{q,RHS}$, is used for the assembly of the system's RHS, while the latter, $tol_{q,pre}$, is employed for the assembly of the preconditioner matrices. According to the numerical experiments, the tolerance for the RHS has been set to $tol_{q,RHS} = 10^{-10}$ while a more aggressive value for the preconditioner, here $tol_{q,pre} = 10^{-2}$, has been used to further speed up the assembly operations. Notice that the DOE adaptation process is accomplished only once during pre-processing and after each p-adaptation cycle.

In this work the reduction algorithm has been only used for the computations of the flow past a sphere and the flow around a circular cylinder, in Secs. 3.2 and 3.3, respectively.

2.6 The p-adaptive algorithm

An order-adaptive (p-adaptation) strategy identifies, according to an error estimator, the elements where the local polynomial degree of the approximation need to be refined or coarsened. The present strategy has been borrowed from the compressible version of the solver, see Bassi et al. [14,22]. Since we are interested in statistically stationary unsteady flow problems, the estimator is computed from the time-averaged solution and combines two contributions: *i*) a measure of the solution jumps at grid cells interfaces [38,31]; *ii*) the decay

rate of the modal coefficients of the polynomial expansion [59]. Both indicators aim at revealing a lack/excess of spatial accuracy in the numerical discretization and are here applied to the pressure variable.

The jump indicator is calculated as the maximum normalized pressure difference at the interfaces of the element K

$$\eta_K^{JMP} = \max_{if} \max_j \left| \frac{\left(p\left(x_j \right) - p\left(x_j \right)^+ \right)_{if}}{\left(p\left(x_j \right) + p\left(x_j \right)^+ \right)_{if}} \right|, \tag{12}$$

where $p(x_j)$ is the pressure value at the j-th surface quadrature point x_j on the if-th element's interface and computed with the degrees of freedom of the solution belonging to the cell K, while $p(x_j)^+$ is the pressure value at the same location but computed with the degrees of freedom belonging to the cell sharing the face if with K. Our numerical experiments revealed that this indicator, probably due to its stencil, marks for adaptation large regions of the domain, especially when dealing with low-order solutions.

The Spectral Decay Indicator (SDI) correlates the amplitudes of highest modes of the solution to the amplitude of the total modes, and is defined as

$$\eta_K^{SDI} = \frac{\int_K (p^K - \tilde{p}^K)^2 d\mathbf{x}}{\int_K (p^K)^2 d\mathbf{x}}, \tag{13}$$

$$p^K(\mathbf{x}) = \sum_{l=1}^{N_{dof}^K} W_{p,l}\phi_l^K(\mathbf{x}), \qquad \tilde{p}^K(\mathbf{x}) = \sum_{l=1}^{L_{dof}^K} W_{p,l}\phi_l^K(\mathbf{x}), \tag{14}$$

where $W_{p,l}$ are the coefficients of the modal expansion related to the pressure variable, N_{dof}^K the number of degrees of freedom associated to the local polynomial degree k_K, and L_{dof}^K the number of degrees of freedom associated to the polynomial degree $k_K - 1$.

Unlike the jump indicator, the SDI does not involve in its definition the solution on the neighbouring cells. Moreover, while the jump indicator can be computed for any polynomial degree, the SDI definition is not suited for piece-wise constant approximations, *i.e.*, $k_K = 0$.

The error estimator, inspired by [31], has been implemented by combining the two indicators as

$$\eta_K^{TOT} = \eta_K^{SDI} + \frac{1}{\max\left(1, k_K\right)}\eta_K^{JMP}, \quad \forall K \in \mathcal{K}_h, \tag{15}$$

where, according to its definition and our numerical experiments, η_K^{SDI} is set to 0 for $k_K = \{0,1\}$. Before the coupling, both the indicators η^{JMP} and η^{SDI} are normalized over the domain according to their maximum and minimum values.

The adaptation process is triggered by a simple indicator, trg, defined as the norm of the relative increment, computed at each time step, of the vector of the degrees of freedom of the run-time time-averaged solution $\overline{\mathbf{W}}$.

The pseudo code for the p-adaptation strategy is reported in the Algorithm 3, where ℓ is the index denoting the ℓ-th adaptive cycle. The user-defined parameters for the process are:

- k_{ini}, the value for the uniform initialization of the polynomial degree over the mesh;
- n_{adp}, the maximum number of adaptation cycles;
- tol_{trg}, a threshold tolerance for the activation of the adaptation process;
- $\mathcal{N}_{adp}$, the minimum number of time steps, i_{cyc}, between two adaptation cycles;
- k_{max}, the maximum allowable polynomial degree;
- k_K, the polynomial degree on element K;
- $\mathcal{G}_r$, the percentage of elements with the higher estimated error that are marked for refinement at each adaptation cycle;
- $\mathcal{G}_c$, the percentage of elements with the lower estimated error that are marked for coarsening at each adaptation cycle (coarsening);
- POS_K, the position of the element K in an array numbered from zero and sorted in increasing order according to the estimator η_K^{TOT} value.
- N_{cyc}, the total number of time step.

Algorithm 3 p-adaptation algorithm

1: $\ell = 0$
2: $j_{cyc} = 1$
3: $k_K = k_{ini} \ \forall K \in \mathcal{K}_h$
4: **for** $i_{cyc} = 1$ **to** N_{cyc} **do**
5:       advance the solution in time
6:       evaluate the run-time time-averaged solution, $\overline{\mathbf{W}}$
7:       compute trg: the relative increment of $\overline{\mathbf{W}}$
8:       **if** $(j_{cyc} \geq \mathcal{N}_{adp}$ **and** $trg \leq tol_{trg}$ **and** $\ell \leq n_{adp})$ **then**
9:           $\ell \leftarrow \ell + 1$
10:          **if** $(\ell = 1$ **and** $k_{ini} = 0)$ **then**
11:              **for** $K \in \mathcal{K}_h$ **do**
12:                  $k_K \leftarrow 1$
13:              **end for**
14:          **else**
15:              compute and normalize the estimators $\eta_K^{TOT} \ \forall K \in \mathcal{K}_h$
16:              **for** $K \in \mathcal{K}_h$ **do**
17:                  **if** $POS_K \geq (1 - \mathcal{G}_r)card(\mathcal{K}_h)$ **then**
18:                      $k_K \leftarrow \min(k_K + 1, k_{max})$
19:                  **else if** $POS_K < (\mathcal{G}_c)card(\mathcal{K}_h)$ **then**
20:                      $k_K \leftarrow \max(k_K - 1, 1)$
21:                  **end if**
22:              **end for**
23:              balance the load among processors via re-partitioning
24:          **end if**
25:          L_2 projection of the solution on the new polynomial space
26:          $j_{cyc} = 0$
27:      **end if**
28:      $j_{cyc} \leftarrow j_{cyc} + 1$
29: **end for**

The algorithm has been written in a general form, as we consider both refining and coarsening. However, all the simulations performed in this papers uses very coarse meshes, where polynomial degree coarsening is not necessary, *i.e.*, $\mathcal{G}_c = 0$. As suggested by numerical experiments, the following initialization strategy has been adopted to optimize the effectiveness and the computational efficiency of the algorithm: the computations start from a piece-wise free-stream solution ($k_{ini} = 0$) and the polynomial degree is increased uniformly to $k_K = 1, \forall K \in \mathcal{T}_h$, during the first adaptation cycle ($\ell = 1$).

Adaptation leads to a variable order distribution over the mesh, and, as a consequence, the finest level implicit operator is a partitioned matrix with blocks of variable size. In the adaptive framework different possibilities arise for the definition of the coarse levels for the p-MG preconditioner. In this work the coarse operators are made of equal-order approximations over the mesh, where the polynomial degree of the levels is set by the end user. Since the p-adaptation process can generate elements with a degree which can be lower/equal than the ones set for the intermediate/coarse levels by the user, some *ad-hoc* strategy needs to be devised. In particular, for the elements K where $k_K < k_l$, with $l = \{2, \ldots, L\}$, the coarse levels are locally set to $\min(k_K, k_l)$.

Parallel implementation The use of a p-adaptation strategy for parallel simulations induces an imbalance of the load per partition and a drastic reduction of the parallel efficiency. To overcome this issue, a balanced re-partitioning of the computational grids is implemented, which is used after every polynomial adaptation. The strategy exploits the Metis [35] library to optimize the decomposed mesh by enforcing multiple constraints on a weighted graph. Different weights are associated to the vertices of the graph, corresponding to the mesh elements. These weights are computed to balance the computational cost of the residual, the Jacobian assembly, and matrix-vector products of the (F)GMRES. The definition of the weights can be found in the Appendix of [13]. As in the present work we rely on an adaptation strategy for quadrature rules that can lead to a different set of integration points for the evaluation of the RHS and preconditioner, respectively, the overall local number of points is used in the weights definition.

3 Numerical results

The performance of the proposed algorithms has been demonstrated by computing the following test cases: (i) the flow over a rounded leading edge flat plate at $\mathrm{Re}_D = 3\,450$ under different levels of free-stream turbulence (the T3L test case of the ERCOFTAC test case suite), (ii) the laminar flow past a sphere at $\mathrm{Re}_D = 300$, and (ii) the flow around a circular cylinder at Reynolds number $\mathrm{Re}_D = 3\,900$.

The scope of the T3L case is twofold: assess the improvement of the solver efficiency due to the use of the p-multigrid preconditioner and demonstrate the suitability of the underlying DG method for the ILES.

The flow past a sphere at $\mathrm{Re}_D = 300$ test case is used to investigate the improvement in terms of performance due to the p-adaptive strategy with respect to the original uniform degree spatial discretizations. Although this test case can be considered as a simple problem, it allowed to clearly assess the performance enhancement provided by the adaptive method, and to give confidence for its use in more complex flow problems.

Finally, as a proof of concept, the p-adaptive ILES of the turbulent flow past a circular cylinder at $\mathrm{Re}_D = 3\,900$ was considered to demonstrate the potential of our promising adaptive DG method.

3.1 Flow around the T3L plate at $\mathrm{Re}_D = 3\,450$

The first test case is the transitional flow over a rounded leading edge flat plate, the T3L test case of the ERCOFTAC suite [23], at $\mathrm{Re}_D = 3\,450$ based on the leading edge diameter D and the free-stream velocity u_{ref}. This flow problem is used to demonstrate the potential of our DG discretization in the ILES of incompressible flows as well as the performance of the preconditioning strategy for the linear solver regardless of any p-adaptation strategy. The simulations have been performed in parallel using 540 cores on a hybrid mesh of $38\,320$ elements with quadratic edges and a uniform $\mathbb{P}^6$ solution approximation. The first cell height is $10^{-2}\,D$ and the mesh is refined near the reattachment region, where the minimum dimension along x axis is $2{\cdot}10^{-2}\,D$. The domain extension on the $x-y$ plane is $28D \times 17D$, and it is extruded using 10 elements along the spanwise direction z, assuming spanwise periodicity equal to $2D$.

The numerical resolution of this simulation (DOFs$=3\,218\,880$) is comparable to the LES with dynamic Smagorinsky model and second-order discretization of Yang and Voke [69] and Langari and Yang [41]. For the same test case, Lamballais et al. [39] perform a marginally-resolved DNS using a sixth-order finite difference scheme. The number of DoFs of such simulation is much larger, *i.e.* roughly 64 times those of the present DG solution on a domain 3.9 times smaller, due to the use of a symmetry boundary condition at $y = 0$. Although obtained for a slightly higher Reynolds number, *i.e.* $\mathrm{Re}_D = 4\,000$, we consider the results from Lamballais et al. as a good reference to highlights the computational savings of our approach with respect to a DNS.

An implicit time integration scheme, here the ROSI2PW, has been used that, in general, overcomes the strict explicit stability limit of a high-order DG discretization, which scales, for a model problem, as $\Delta t \sim h/k^2$, where h is the element size and k is the polynomial degree [34]. This choice allows using a time step for the $Tu = 0.2\%$ case that is 16 and 8 times larger than those reported in [69] and [41], respectively. Unfortunately, details on the time step size are not available for the DNS of Lamballais et al. [39].

The flowfield is very sensitive to the free-stream turbulence intensity at the inlet (Tu) and the choice of the way to inject the turbulence is of paramount importance. The generation of a free-stream turbulence at inlet has not been considered because it would be probably damped before reaching the leading

edge of the plate, as the mesh at the farfield is very coarse. As a consequence, the turbulent fluctuations have been synthetically injected, via a spatially-supported random forcing term [27], in regions where the mesh is fine enough to avoid the dissipation of small scales. The following turbulence intensity values have been considered: $Tu = \{0.0\%, 0.2\%, 0.65\%, 2.3\%, 5.6\%\}$.

The forcing term is a vector $\mathbf{f}_h$ added as a source term to the discrete momentum equations (2). This approach, instead of perturbing the flow velocity directly, guarantees a divergence-free perturbation, consistent with the incompressibility constrain. The vector is obtained by multiplying a random three-dimensional discrete vector $\mathbf{r}_h$ to a function of the spatial coordinates x, y, z. This function is chosen to be a gaussian distribution in the x direction and homogeneous in $y-z$ such that

$$\sum_{K \in \mathcal{T}_h} \int_K \phi_i^K f_{h,j} d\mathbf{x} = \sum_{K \in \mathcal{T}_h} \int_K \phi_i^K \left(A e^{-\left(\frac{x-\overline{x}}{2\alpha} \right)^2} \right) r_{h,j} d\mathbf{x}, \qquad (16)$$

where A, $\overline{x}$ and α are the amplitude coefficient, the location of the forcing plane, and the amplitude of the gaussian support. The random vector $r_{h,j}$ is computed at each Gauss-quadrature point separately during the residual assembly, and normalized such that $\|\mathbf{r}_h\| = 1$. It is worth pointing out that the discrete forcing vector changes not only in space, but also in time, but it has been verified that the results are insensitive with respect to temporal refinements. In our computations, the gaussian function was centered in $\overline{x}/D = -3$, and the constants A, α were adjusted, via a trial and error approach, to meet the experimental Tu value within the proximity of the leading edge of the plate. We avoided a fine control algorithm of the turbulent length-scale since the reattachment length is pretty insensitive to this value, see [33,51].

The performance of the different strategies to solve the linear system arising from the temporal discretization, see Secs. 2.3 and 2.4, has been first assessed over 6 time steps. In particular the flowfield at $Tu = 0.2\%$ has been computed using the p-MG MF-FGMRES, the MB- and MF-GMRES approaches, assembling the preconditioner every time step or every 3 time steps. For this test case no adaptive strategy for the selection of the DOE of the quadrature rules over the grid was used. For all the cases the relative tolerance for convergence is $tol_r = 10^{-5}$, while the p-MG preconditioner settings are summarized in the following:

- full multigrid V-cycle;
- number of levels $L = 3$;
- coarse level: $k_c = 1$, GMRES smoother with ASM(1) preconditioner, maximum number of GMRES iterations $n_{it}^{max} = 60$;
- intermediate levels: $k_1 = 2$, GMRES smoother with EWBJ preconditioner, maximum number of GMRES iterations $n_{it}^{max} = 8$;
- fine level: $k_K = 6$, $\forall K \in \mathcal{K}_h$, GMRES smoother with EWBJ preconditioner, maximum number of GMRES iterations $n_{it}^{max} = 8$.

The results of the comparison are summarized in Tab. 1, and, as expected [27], the p-MG strategy reduces the CPU time by $\sim 50\%$ with respect

Table 1 T3L case – Assessment of different strategies [MB, MF and p-MG MF-FGMRE (operator assembly at each time step)] for the linear system solution in terms of non dimensional CPU time, memory footprint of the solution process, average number of GMRES iterations, and the percentage ratio time$_{mat}$/time% of the CPU time needed for the operator assembly with respect to the overall simulation time.

	BJ-MB	ASM(1)-MB	BJ-MF	EWBJ-MF	p-MG
CPU [%]	100	111	95	101	47
Memory [%]	100	160	60	21	15
GMRES iterations	115	72	115	229	3
time$_{mat}$/time%	18.8	17.5	21.0	5.5	35.1

to a standard matrix-based method, and the memory footprint by 85%. Notice that, since the simple EWBJ is not a very effective preconditioner, a large number of linear iteration is required to reach a given convergence level, thus resulting in the allocation of large GMRES Krylov subspacs, which compromise, at least partially, the strong memory saving proper of the EWBJ operator. The superior effectiveness of the p-MG preconditioner allows maintaining a small number of GMRES iterations even when the matrix operator is lagged over some time steps, without any significant influence on the memory allocation. In particular, when the assembly is lagged over $lg = 3$ time steps, the number of linear iterations increases by $\sim 10\%$ only. This behaviour can be related to one of the main advantages of the matrix-free algorithm, which automatically maintains the finest level operator always updated also within the lagged p-MG preconditioner. Indeed, the re-assembly of the p-MG preconditioner over some time steps is avoided by lagging the iteration matrix and the preconditioner of the coarse levels smoothers, while the finest level smoother uses the same (block-diagonal) preconditioner together with an always updated matrix-free operator. In practice, a p-MG preconditioner in a MF-GMRES framework automatically keeps trace in its structure of the solution changes even when the operator assembly is lagged. For this flow problem, the lag of the operator assembly allows further improving the saving in terms of CPU time, which is 69% lower than the value measured for a standard BJ-MB algorithm. However, the other preconditioners when coupled with the matrix-free algorithm have different performance. The BJ-MF strategy with $lg = 3$ shows an increase of the GMRES iterations, $\sim +25\%$, with a slight performance degradation. In an unsteady context, the lag of the operator assembly for a MB solution algorithm can not be always exploited, since the accuracy of some classes of temporal schemes, *e.g.*, the linearly-implicit methods [11,28], can be guaranteed only by computing the iteration matrix at each time step.

Concerning the time needed for the assembly of the Jacobian matrix, it is worth noting that, even if the p-MG shows a larger relative value, time$_{mat}$/time, the absolute value is slightly smaller than that of the BJ operator thanks to the direct assembly of the off-diagonal blocks at the (lower) polynomial degree of the coarse levels.

The T3L case shows smaller relative time for the assembly of the operator compared to the values reported in the next Section for the laminar unsteady flow past a sphere, cfr. Tab. 4. This behaviour is motivated by the very few curved elements within the mesh, located close to leading-edge, that use quadrature rules as if they were straight-sided cells, with a significant reduction of the assembly cost for the Jacobian matrix. All the computations presented in the rest of this Section have been performed using the p-MG MF-FGMRES approach.

Figure 1 shows the time-averaged x-component velocity u_x/u_{ref} and the velocity fluctuations $u'_x u'_{xrms}/u_{ref}$ for increasing Tu values. The main flow feature that characterizes this testcase is the laminar separation bubble, $i.e.$, the laminar flow detaches at the plate leading edge, and reattaches after becoming turbulent. As the turbulence intensity increases, the extent of the separation bubble decreases, as shown by the x-component velocity, and the reattachment point moves towards the leading edge. The turbulent transition of the flow is marked by the increasing velocity fluctuations.

The instantaneous flowfield, see Fig 2, shows some differences for $Tu = 0.0\%$ (left) and $Tu = 5.6\%$ (right). At $Tu = 0\%$, the quasi two-dimensional Kelvin-Helmholtz instabilities in the shear-layer region above the separation bubble and their convection downstream are observed. Differently, at the highest turbulence level, streaky like structures streamwise oriented are visible close to the leading edge. The development of hairpin vortices after the reattachment and the breakdown to turbulence are similar for both cases.

The pressure, c_p, (left) and skin friction, c_f, (right) coefficients distributions along the plate are depicted in Fig. 3. The plateau along the c_p distribution and the zone where c_f is negative represent the extent of the laminar separation bubble. The length of the laminar separation bubbles for the different Tu is reported in Tab. 2, and compared with the experimental data, explicitly available only for $Tu = 0.2\%$. The solver predicts a value of $2.69D$, which is in good agreement with the experiment ($2.75D$), while it is slightly different in comparison to reference LES results, $i.e.$ $2.6D$ [69] and $3.0D$ [41]. Predicted values are reported in Fig. 4, where the DNS data of Lamballais et al. [39] at $\mathrm{Re}_D = 4\,000$, the experimental data from Coupland reported as personal communication in Yang and Voke [69], and the experimental data available in Palikaras et al. [55] (same geometry with a slightly different flow configuration, $i.e.$, $Tu = 7\%$ and $\mathrm{Re}_D = 3\,330$) have been also added.

The computed curve is similar to the DNS curve for low value of Tu, even if shifted upward for the lower Reynolds number. The experimental data at $Tu = 7\%$ confirms the trend of the computed curve for higher values of Tu and the greater accuracy of the computed bubble length for $Tu = 5.6\%$, $i.e.$ $l/D = 1.08$, with respect to the value reported in the LES of Langari and Yang [41], $i.e.$ $l/D = 1.8$.

Figures 5 and 6 compare the computed velocity profiles with the experimental data. The mean stream-wise velocity u_x/u_{max}, and the velocity fluctuation (or velocity RMS), $u'_x u'^{+}_{xrms}/u_{max}$, are reported as a function of the normal direction for different stations. The velocity is normalized by the local
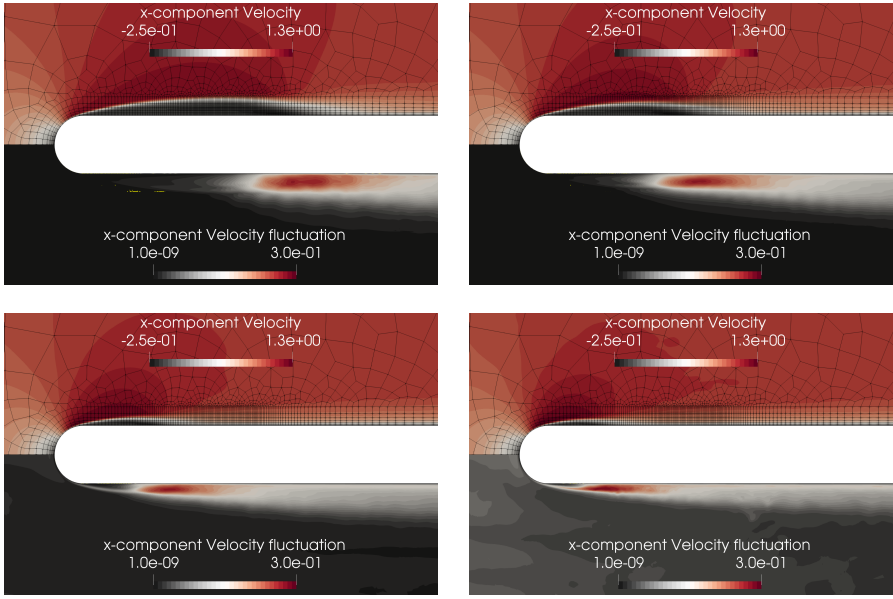
Fig. 1 T3L case – Spanwise- and time-averaged x-component velocity u_x/u_{ref} and velocity fluctuations $u'_x u'_{x_{rms}}/u_{ref}$ with $Tu = 0\%$ (top left), $Tu = 0.2\%$ (top right), $Tu = 2.3\%$ (bottom left) and $Tu = 5.6\%$ (bottom right), $\mathbb{P}^6$ solution
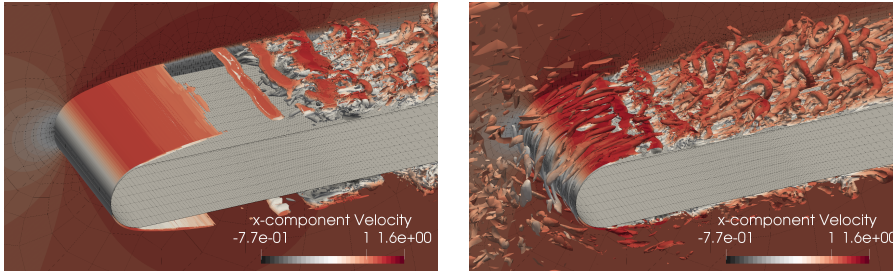

Fig. 2 T3L case – Iso-contour with $\lambda_2 = -1$ and periodic plane coloured by the x-component velocity u_x/u_{ref} with $Tu = 0\%$ (left) and $Tu = 5.6\%$ (right), $\mathbb{P}^6$ solution

Table 2 T3L case – Length of the laminar separation bubble as a function of the free-stream turbulence intensity. Comparison of the computed results with available experimental data

	T3L0	T3L1	T3L2	T3L3	T3L4
Tu	0.00%	0.20%	0.65%	2.30%	5.60%
l/D	3.90	2.69	2.00	1.49	1.08
l/D_{exp}	-	2.75	-	-	-
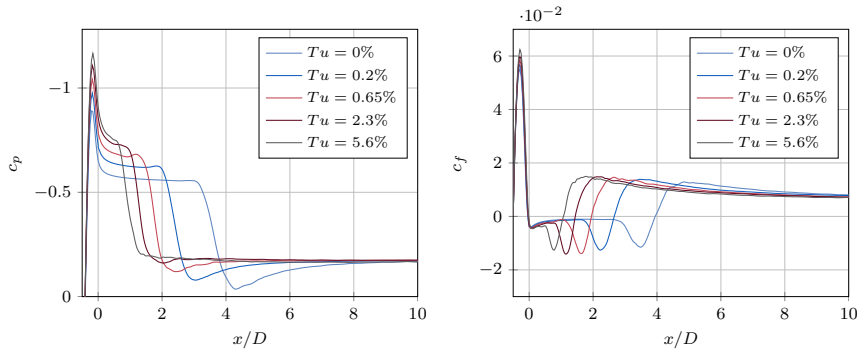
Fig. 3 T3L case – Spanwise- and time-averaged pressure c_p (left) and skin friction c_f (right) coefficients distribution on the wall, $\mathbb{P}^6$ solution
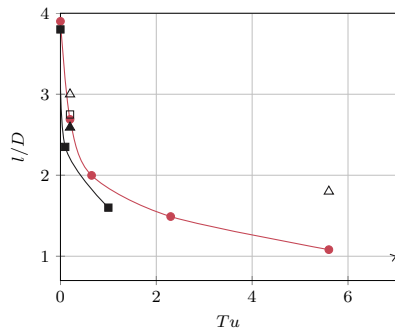

Fig. 4 T3L case – Length of the laminar separation bubble as a function of the free-stream turbulence intensity in comparison with available numerical and experimental data, $\mathbb{P}^6$ solution. —— DG-$\mathbb{P}^6$, ▲ Yang and Voke LES [69], △ Langari and Yang LES [41], —■— Lamballais et al. DNS $\mathrm{Re}_D = 4000$ [39], □ Copland Exp. reported in Yang and Voke [69], ⋆ Palikaras et al. Exp. [55].

maximum velocity u_{max} for each station, as suggested in literature [69, 41]. In general, a good agreement is observed with the experimental data, especially for mean velocity profiles, which suggests that the bubble length is correctly captured for all turbulence intensities. The fact that our results match with experimental data near the plate stagnation point confirms the effectiveness of the random forcing term, even if a fine tuning of the turbulent length-scale was not performed. The result confirms that the length-scale value does not influence the reattachment length, as conjectured by [33, 51].

Figure 7 shows the averaged velocity profiles expressed in wall units computed at the station $x/D = 12.1$ for $Tu = 5.6\%$, where the momentum thickness Reynolds number Re_θ is 295. The non dimensional stream-wise velocity profiles compare fairly well with the law of the wall, while both stream-wise velocity and velocity fluctuations are in good agreement with the $\mathrm{Re}_\theta = 300$ boundary layer DNS results of Spalart [63]. We remark that for $y^+ > 100$ the fluctuations show some differences due to a different value of the free-stream
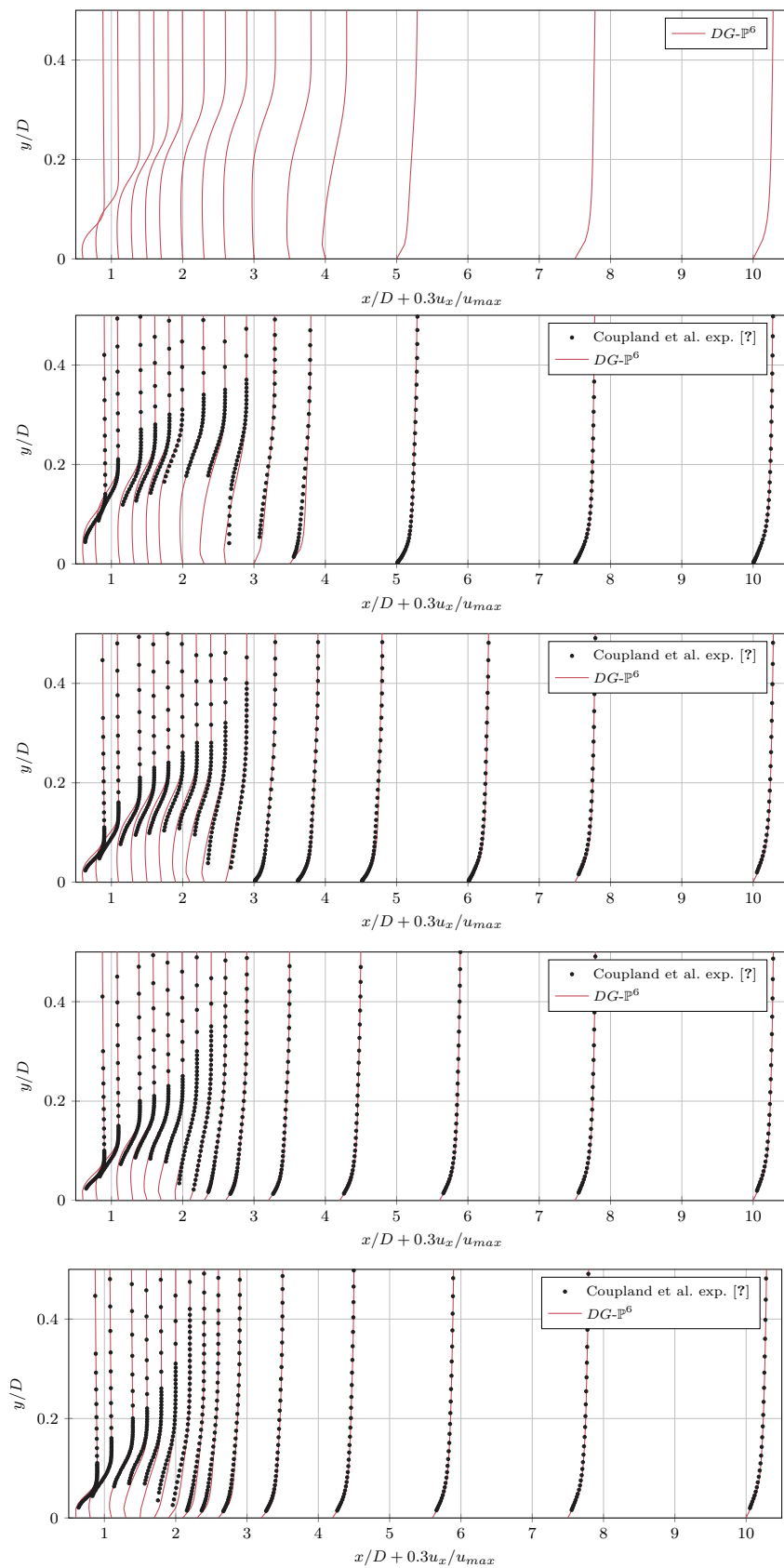
Fig. 5 T3L case – Spanwise- and time-averaged x-component velocity u_x/u_{max} at different locations with $Tu = \{0\%, 0.2\%, 0.65\%, 2.3\%, 5.6\%\}$ from top to bottom
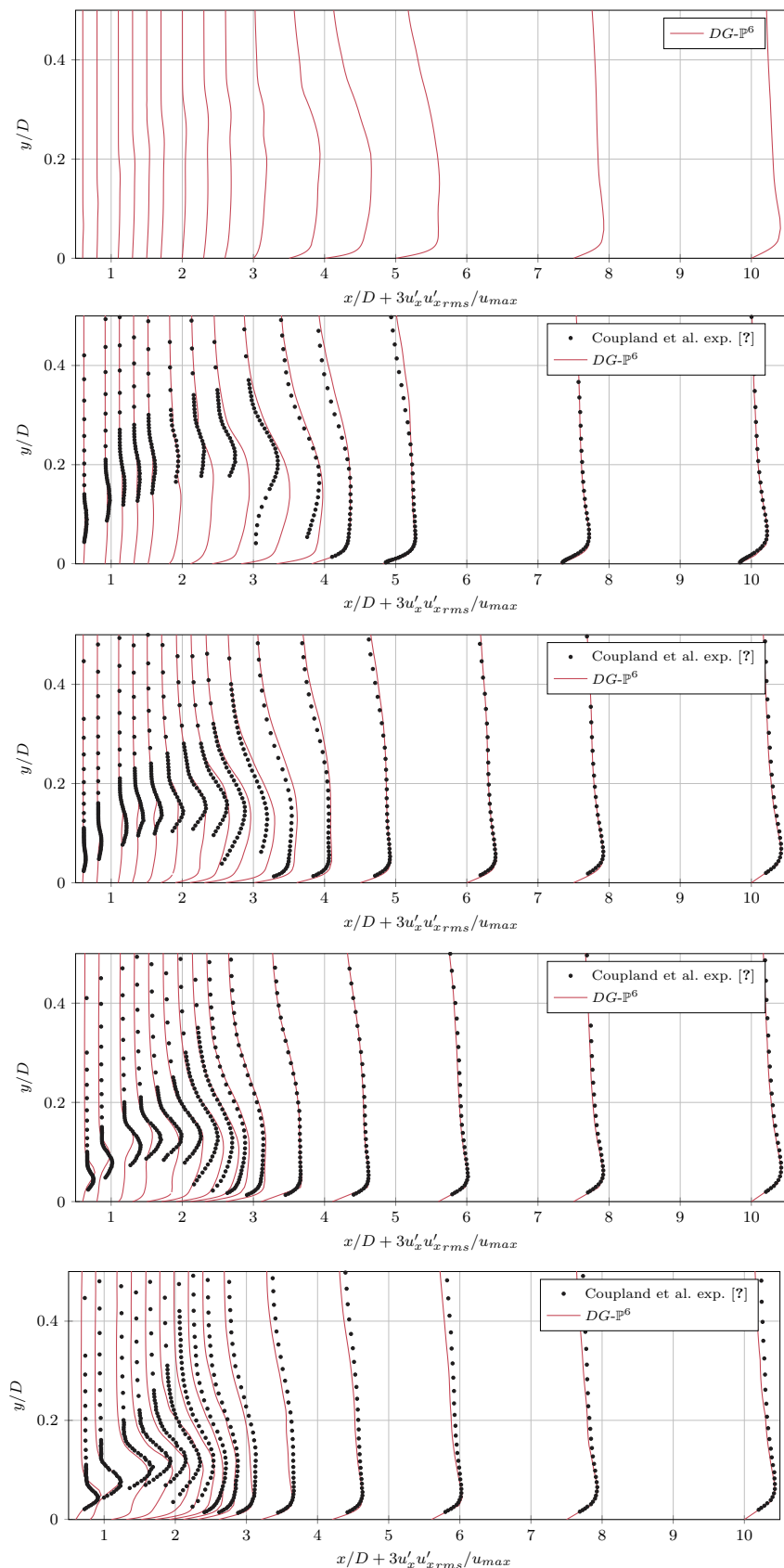
Fig. 6 T3L case – Spanwise- and time-averaged x-component velocity fluctuations $u'_x u'_{x_{rms}}/u_{max}$ at different locations with $Tu = \{0\%, 0.2\%, 0.65\%, 2.3\%, 5.6\%\}$ from top to bottom
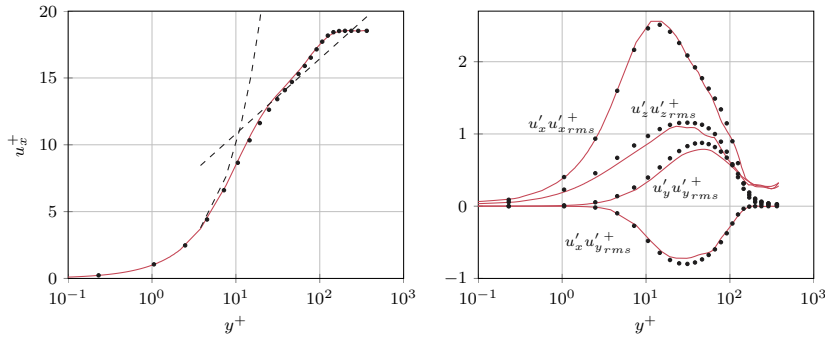
Fig. 7 T3L case – Non dimensional x-component velocity profile (left) and x-, y-, z-component velocity fluctuations profile (right) in the boundary layer at $x/D = 12.1$ with $Tu = 5.6\%$, $\mathbb{P}^6$ solution. —— DG-$\mathbb{P}^6$, • Spalart $\mathrm{Re}_\theta = 300$ DNS [63]

turbulent kinetic energy. These differences reduce for lower values of Tu, whose results are here not reported for the sake of compactness. Nevertheless, the values of ${u'_x u'_y}^+_{rms}$ agree very well with the reference boundary layer data and suggest that the free-stream turbulence is isotropic. The agreement points out that at about 10 bubble lengths downstream the reattachment the boundary layer recovers an almost standard shape, somehow "forgetting" the separation induced transition. This behaviour confirms the finding of Alam and Sandham [2], who stated that the logarithmic law of the wall is re-established after about 7 bubble lengths.

As stated by Castro and Epik [21], a distance greater than 70 boundary layer thickness, evaluated at the flow reattachment, is required to perfectly form the standard wake-law structure of the outer layer. The value extrapolated from the simulations is slightly smaller (roughly 55). However, we consider this behaviour acceptable since the current Re_θ is lower than the value reported in [21], as well as what is usually considered an equilibrium turbulent boundary layer.

Figure 8 shows the kinetic energy spectra evaluated for $Tu = \{0.65\%, 2.3\%, 5.6\%\}$ at two probe points: $(x, y) = (0.9D, 0.71D)$ that is above the bubble, and $(x, y) = (6.92D, 0.65D)$ that is well behind the reattachment zone. For the first point a peak at $St = fD/u_{ref} \simeq 1.3$ is observed mainly for the lower Tu values, due to the Kelvin-Helmholtz instability. For the second probe point, located roughly at $y^+ = 35$, the peak is absent and all the spectra collapse in one curve. These results confirms the flow visualization of Fig. 2 and the by-pass of the Kelvin–Helmholtz instability for Tu values sufficiently high.

3.2 Laminar flow past a sphere at $\mathrm{Re}_D = 300$ – A performance assessment

In this paragraph the performance improvement of the p-adaptive strategy over the uniform degree spatial discretizations is assessed by computing the
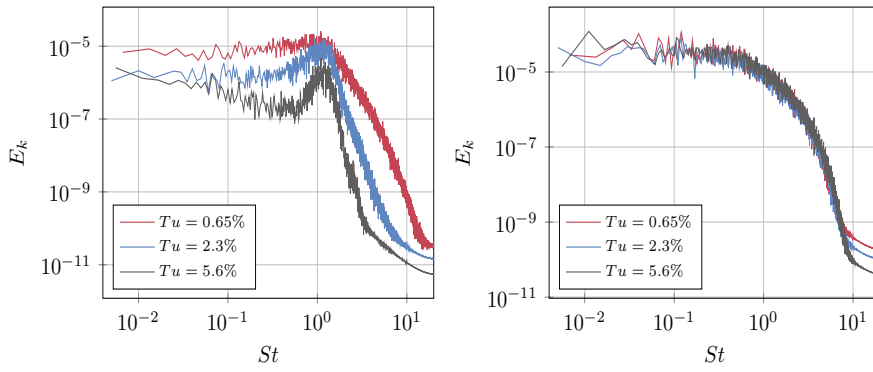
Fig. 8 T3L case – The kinetic energy spectra evaluated for $Tu = \{0.65\%, 2.3\%, 5.6\%\}$ at $(x, y) = (0.9D, 0.71D)$ (left) and $(x, y) = (6.92D, 0.65D)$ (right), $\mathbb{P}^6$ solution.

unsteady laminar flow past a sphere at $\mathrm{Re}_D = 300$. This test case exhibits a periodic behaviour, with the flow maintaining a plane of symmetry [25]. The computational mesh is made of $3\,560$ hexahedral elements with quadratic edges and we apply a symmetry condition to simulate half of the sphere. Solutions are computed for uniform polynomial degree discretizations, $\mathbb{P}^3$ up to $\mathbb{P}^5$, and compared with the results from a *p*-adaptation process with $k_{ini} = 0$, $k_{max} = 5$, $\mathcal{G}_r = 0.2$, $\mathcal{G}_c = 0$ (coarsening disabled), $\mathcal{N}_{adp} = 1\,000$ and $n_{adp} = 9$. The tol_{trg} parameter was deliberately set to a large value to force adaptation to be activated only when the $\mathcal{N}_{adp}$ criteria is satisfied, and to monitor the force coefficients over roughly 10 shedding periods between two subsequent adaptations. For all the simulations, ROS3P time integration scheme was adopted with a constant time step size $\Delta t = 0.125 D/U_\infty$. For the linear system solution the MF-FGMRES method is used with a relative tolerance $tol_r = 10^{-3}$. Differently from the T3L case (see Sec. 3.1), for low-Reynolds flow problems the viscous effects are dominating. As the diffusive terms are linear, the Jacobian is exact even when the evaluation of the matrix is lagged, see [28]. As a consequence, the number of GMRES iterations does not significantly increase with the size of the lagging window and the relatively large $lg = 5$ value was considered. The remaining settings of the *p*-MG preconditioner are the same of the T3L case. For the matrix-free cases, the tolerances for the adaptation of the quadrature rules DOE are set to $tol_{q,RHS} = 10^{-10}$ and $tol_{q,pre} = 10^{-2}$ for the residual and preconditioner assembly, respectively. When considering matrix-based implementations, *e.g.*, the BJ-MB, to achieve the formal accuracy of the employed time integration scheme, the Jacobian matrix must be exact. For this reason, only in these cases we use quadrature rules for the matrix assembly of the same adapted DOE of the residual, *i.e.*, $tol_{q,RHS} = 10^{-10}$.

For the same $p-MG$ preconditioning technique, the numerical experiments are summarized in Tab. 3 in terms of number of DOFs per equation; percentage ratio of DOFs and CPU time, for a single time step, with respect to the

Table 3 Flow past a sphere – Performance assessment in terms of number of DOFs per equation; percentage ratio of DOFs and CPU time with respect the reference $\mathbb{P}^5$ values (DOFs_r and time_r); relative percentage error on the time-averaged force coefficients, $\overline{c_D}$ and $\overline{c_L}$, on the oscillation semi-amplitudes of the force coefficients, Δc_D and Δc_L, and on the Strouhal number $St = fD/U_\infty$.

		DOFs	$\frac{\text{DOFs}}{\text{DOFs}_r}\%$	$\frac{\text{time}}{\text{time}_r}\%$	relative error, $err(\cdot)$, [%]				
					$\overline{c_D}$	$\overline{c_L}$	Δc_D	Δc_L	St
uniform	$\mathbb{P}^3$	71 200	35.7	28.6	0.70	10.18	29.26	3.35	-1.98
	$\mathbb{P}^4$	124 600	62.5	58.5	0.00	0.03	15.64	6.13	-1.79
	$\mathbb{P}^5$	199 360	100.0	100.0	–	–	–	–	–
$\mathbb{P}^{1\to5}$	$\ell=6$	44 911	22.5	24.8	0.17	-0.06	-4.84	-3.54	0.00
	$\ell=7$	53 603	26.9	30.2	0.08	0.01	-1.44	-0.80	-0.38
	$\ell=8$	62 725	31.5	32.3	0.06	0.03	-0.92	-1.13	0.05
	$n_{adp}=9$	72 038	36.1	35.9	0.02	0.01	-0.45	-0.60	0.05

reference $\mathbb{P}^5$ values (DOFs_r and time_r); relative percentage error on the time-averaged force coefficients, $\overline{c_D}$ and $\overline{c_L}$, on the oscillation semi-amplitudes of the force coefficients, Δc_D and Δc_L, and on the Strouhal number $St = fD/U_\infty$. The reference $\mathbb{P}^5$ values for the coefficients are $\overline{c_D} = 0.6572$ and $\overline{c_L} = 0.06590$, $\Delta c_D = 0.00278$ and $\Delta c_L = 0.0145$, $St = 0.1309$. The $\overline{c_L}$ coefficient is evaluated along the direction normal to the symmetry plane. Results from the p-adapted $\mathbb{P}^{1\to5}$ discretizations from $\ell = 6$ to $\ell = n_{adp} = 9$ are tabulated in Tab. 3.

All the simulations reported in Tab. 3 are in good agreement, in terms of time-averaged force coefficients, with the reference values except for the $\overline{c_L}$ of the uniform $\mathbb{P}^3$ computation that shows an error of $\sim$10%. Large error values are observed for the coefficients oscillation semi-amplitudes: Δc_D value predicted by the uniform $\mathbb{P}^3$ and $\mathbb{P}^4$ computations shows an error $\sim$29% and $\sim$15.5%, respectively, while the computation with p-adaptation ($\mathbb{P}^{1\to5}$, ℓ=6) shows an error $\sim$5%.

The adaptive procedure, from $\ell = 7$ onward, is in general able to deliver an overall accuracy comparable to the reference $\mathbb{P}^5$ computation. For the adaptive $n_{adp} = 9$ simulation the relative errors on all the monitored coefficients are within the 1%, with significant savings in terms of both CPU time for a single time step, $\sim -64.1\%$, and degrees of freedom, $\sim -64\%$. Notice that, for this flow problem, the CPU time reduction for the adaptive solutions directly reflect the percentage reduction of the number of DOFs with respect to the reference uniform $\mathbb{P}^5$ computation.

Figure 9 shows the overall relative error err_a, defined as the arithmetic mean of the absolute values of the relative errors, against the percentage number of DOFs ($\text{DOFs}/\text{DOFs}_r\%$) and the CPU time for a single time step ($\text{time}/\text{time}_r\%$). For the p-adaptive computations and $\ell \geq 6$, the err_a value is always lower than the uniform $\mathbb{P}^4$ case, and, in particular, the error for the $\ell = n_{adp} = 9$ solution is one order of magnitude lower than $\mathbb{P}^4$.
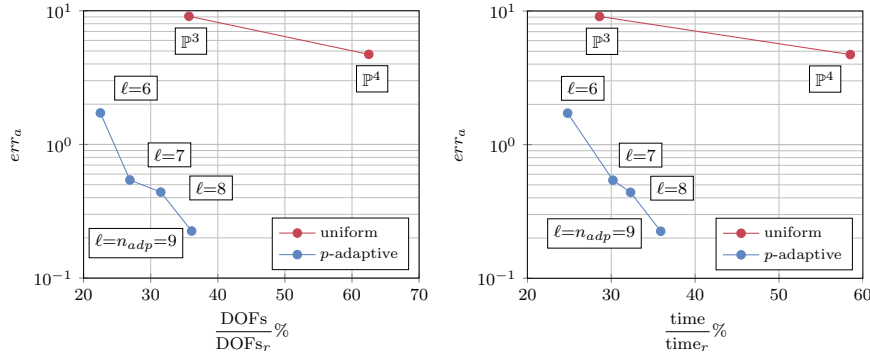
Fig. 9 Flow past a sphere – Averaged relative error err_a against the percentage number of DOFs (DOFs/DOFs$_r$%) (left) and CPU time ratios (time/time$_r$%) (right).

Table 4 demonstrates that the adopted preconditioning strategy, *i.e.*, the *p*-MG, outperforms the BJ-MB or BJ-MF methods for both the *p*-adaptive and the uniform degree simulations, as shown by the percentage CPU time ratio time/time$_{bj}$%. Notice that, when considering *p*-adaptive/uniform degree computations the reference time$_{bj}$ value is set equal to the CPU time for the BJ-MB *p*-adaptive/uniform computation. The *p*-MG preconditioner provides a greater CPU time saving for the uniform-degree cases compared to adaptive simulations, 54% and 30%, respectively. This difference can be attributed to the matrix-free algorithm that shows better performance for high-order polynomial degrees approximations. Indeed, as reported in Franciolini *et al.* [28], the number of operations for the matrix-vector product (matrix-based) and the residual evaluation (matrix-free), scales equally with the polynomial degree only asymptotically, thus suggesting the use of very high-order discretizations. For the present *p*-adaptive computation ($n_{adp} = 9$), the number of low-order cells, *i.e.* with a polynomial degree less than or equal to 1, is large, and represents 45% of the global number of mesh elements, thus partially reducing the overall performance of matrix-free algorithm. This effect can be also clearly observed in the BJ-MF *p*-adaptive case, where, differently from the uniform degree discretization, no gain is obtained even when the matrix assembly is lagged over five iterations ($lg = 5$). When matrix-free is coupled to *p*-multigrid, the large reduction of GMRES iterations ensured by this efficient preconditioner alleviates the higher cost of each GMRES iteration, thus obtaining a memory-saving but efficient strategy.

The different preconditioning techniques are characterized by different cost breakdowns. This is shown by the time$_{mat}$/time percentage ratio of the CPU time needed for the matrices assembly with respect to the overall simulation time. For the BJ-MB case, a similar time percentage is devoted to the matrix assembly for the fixed-degree ($\sim$54%) and the *p*-adaptive ($\sim$60%) simulations. When considering the BJ-MF implementation, as expected, the percentage time for the matrix assembly reduces due to the increased cost of the linear

Table 4 Flow past a sphere – Performance assessment for different preconditioners. For each strategy time$_{mat}$/time% is the percentage ratio of the CPU time needed for the operator assembly with respect to the overall simulation time. For the adaptive and uniform simulations, time/time$_{bj}$% is the percentage ratio of the CPU time of each preconditioning strategy compared with the BJ-MB strategy for the p-adaptive and uniform $\mathbb{P}^5$ runs, respectively. time$_{mat}$/time$_{mat,bu}$% is the percentage ratio of the CPU time needed for the operator assembly of each preconditioning strategy compared with the best performing method for the uniform $\mathbb{P}^5$ case, *i.e.*, p-MG with the matrices assembly lagged over lg=5 iterations. time$_{mat}$/time$_{mat,bu}$% is the percentage ratio of the CPU time of each preconditioning strategy compared with the best performing method for the uniform $\mathbb{P}^5$ case, *i.e.*, p-MG with lg=5.

		$\dfrac{\text{time}_{mat}}{\text{time}}\%$	$\dfrac{\text{time}}{\text{time}_{bj}}\%$	$\dfrac{\text{time}_{mat}}{\text{time}_{mat,bu}}\%$	$\dfrac{\text{time}}{\text{time}_{bu}}\%$
$\mathbb{P}^5$	BJ-MB	53.79	100	1739.7	216.1
	BJ-MF	26.87	112.7	979.6	243.6
	lg=5	7.49	85.1	206.2	184.0
	p-MG	24.46	63.9	505.4	138.1
	lg=5	6.68	46.3	100	100
$\mathbb{P}^{1\to5}$	BJ-MB	59.61	100	453.2	50.8
	BJ-MF	18.64	136.2	193.1	69.2
	lg=5	4.59	114.0	39.8	57.9
	p-MG	17.45	87.9	116.6	44.7
	lg=5	4.38	70.7	23.5	35.9

system solution and the higher cost of the matrix assembly for the BJ-MB strategy. Notice that the cost of the operator assembly for the p-MG preconditioner is smaller than the BJ-MF. As pointed out in Sec. 3.1, this is motivated by an implementation of the multilevel strategy that only retains the diagonal blocks on the finest level matrix and mitigates the overhead related to the assembly and positioning of coarser levels off-diagonal blocks by directly assembling them at a lower polynomial degree. In general, the p-MG outperforms BJ-MF in both the fixed- and variable-order cases, as demonstrated by a smaller time/time$_{bj}$% ratio, *i.e.* (time/time$_{bj}$%)$_{pMG} \sim 0.5$ (time/time$_{bj}$%)$_{BJ-MF}$.

Finally, the time/time$_{bu}$% percentage ratio of the CPU time for each preconditioning strategy compared with the best performing method (bu) for the uniform $\mathbb{P}^5$ case, *i.e.*, p-MG with lg=5, demonstrates the superior performance delivered by putting together the multilevel preconditioner with the p-adaptive algorithm. This coupling allows for a $\sim 64\%$ reduction of the CPU time with respect to the best performing uniform $\mathbb{P}^5$ approach, while not spoiling the overall accuracy, cfr. Tab. 3. Together with a significant CPU time reduction, a large memory saving over the BJ is always guaranteed by assembling the diagonal blocks only. The possible reduction of each block size due to the use of the p-adaptive procedure further improves the memory footprint as shown by the strong saving, $\sim -76.5\%$, in the CPU time for the operator evaluation time/time$_{bu}$% when compared to the best performing fixed-degree strategy.

3.3 Flow past a circular cylinder at $\mathrm{Re}_D = 3\,900$ – *p*-adaptive simulations

The transitional turbulent flow around a circular cylinder at Reynolds number $\mathrm{Re}_D = 3\,900$ based on the diameter D and the free-stream velocity u_{ref} is a very common flow problem used as benchmark in the scientific community [19, 37,54]. This test case was also part of the suite of the International Workshop on high-order methods [1]. The simulations have been performed in parallel using $1\,020$ CPU cores on two different meshes made of $44\,856$ (mesh A) and $67\,466$ (mesh B) elements with quadratic-edges, respectively. Differently from mesh B, mesh A was deliberately generated without any wake refinement, see Figs. 10-12, to assess the performance of the *p*-adaptation strategy in recovering this lack of spatial resolution. Both the grids have been obtained by extruding a two-dimensional mesh using 14 elements in the spanwise direction with a $2D$ periodicity. The cylindrical far-field boundary has been placed at $50D$.

The computations have been initialized with a uniform $\mathbb{P}^0$ solution over the mesh, *i.e.*, $k_{ini} = 0$. According to Algorithm 3, when starting from the piece-wise constant approximation, the devised adaptation strategy brings all the elements of the mesh to $\mathbb{P}^1$ at the first adaptation cycle. The total number of adaptation cycles performed during the computations was set to $n_{adp} = 7$ and to $n_{adp} = 6$ for the mesh A and mesh B, respectively. These values guarantee a comparable number of DOFs per equation at the end of the process, *i.e.*, $653\,679$ and $681\,327$ for mesh A and B, respectively. Adaptation was triggered according to the tolerance $tol_{trg} = 0.02$ and a minimum number of time steps between two adaptation cycles equal to $\mathcal{N}_{adp} = 500$. This last value corresponds to roughly $5\,Tc$, where Tc is a reference convective time defined according to the free-stream velocity and the reference length. ROS3P time integration scheme was adopted with the constant time step size $\Delta t = 0.01Tc$. Coarsening has been disabled, *i.e.*, $\mathcal{G}_c = 0$, while the 20% of elements with the higher estimated error are marked to be refined at each adaptation cycle, *i.e.*, $\mathcal{G}_r = 0.2$. The maximum allowable polynomial degree is $k_{max} = 5$.

For the linear system solution the MF-FGMRES method is used with the relative tolerance $tol_r = 10^{-4}$. The settings for the *p*-MG preconditioner are the same of the T3L case, see Sec. 3.1, but with the preconditioner lagged over $lg = 5$ time steps. The tolerances for the adaptation of the quadrature rules DOE are set to $tol_{q,RHS} = 10^{-10}$ and $tol_{q,pre} = 10^{-2}$ for the residual and preconditioner assembly, respectively.

Once all the n_{adp} adaptation steps have been performed, the solutions have been integrated in time to compute a time-averaged solution over roughly 40 shedding periods. The algorithm was able to increase the resolution at the shear layer and in the wake region as shown in Figs. 11 and 12 by the polynomial degree distribution over the domain and in Fig. 10 by the instantaneous Mach number and pressure contours over the two meshes. As expected, high-order polynomials elements are differently distributed in mesh A and B, with a larger number of $\mathbb{P}^4$ and $\mathbb{P}^5$ cells in the wake region of mesh A. The refine-
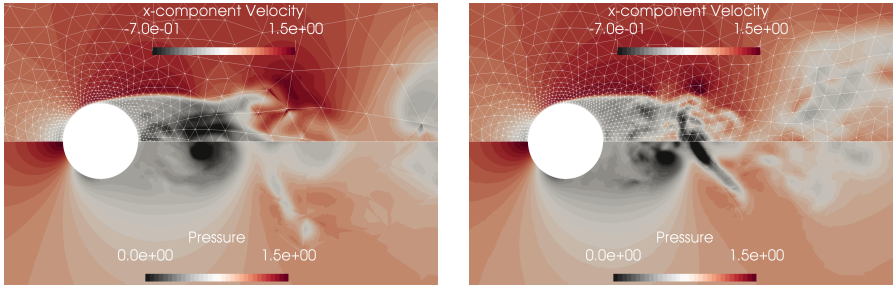
Fig. 10 Flow past a circular cylinder – Instantaneous Mach number and pressure contours for mesh A (left) and B (right), $\mathbb{P}^{1 \to 5}$ solution
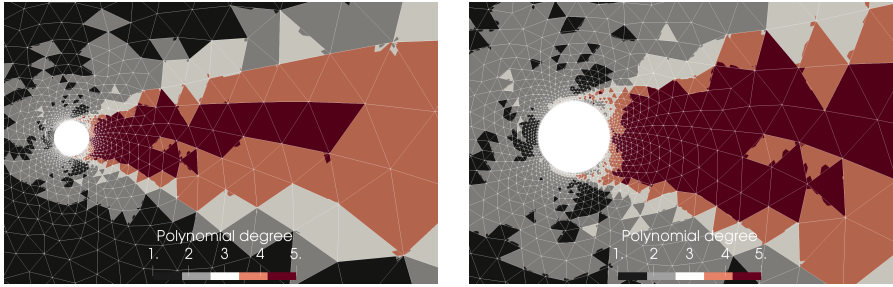

Fig. 11 Flow past a circular cylinder – Polynomial degree distribution after 7 adaptation cycles, mesh A (651 3679 DoFs), $\mathbb{P}^{1 \to 5}$ solution

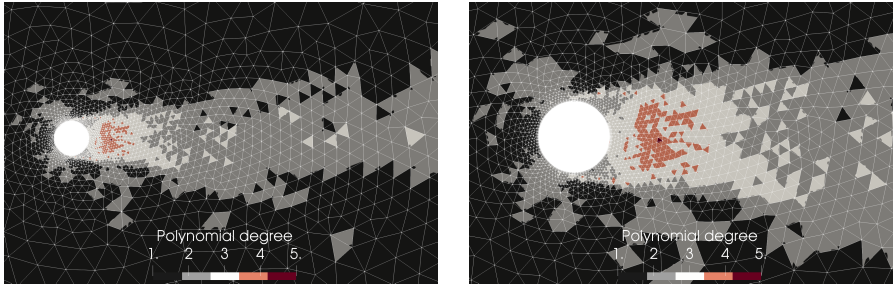**Fig. 12** Flow past a circular cylinder – Polynomial degree distribution after 6 adaptation cycles, mesh B (681 327 DoFs), $\mathbb{P}^{1 \to 5}$ solution

ment in the wake is not symmetric with respect to the centerline due to the asymmetry of the unstructured meshes.

Figure 13 depicts the spanwise- and time-averaged distributions over the cylinder for the pressure coefficient c_p and the non-dimensional wall vorticity $\Omega/2Re^{0.5}$. The c_p distribution is in good agreement with the experimental data from Norberg [52] and other numerical results, such as the DNS of Ma et al. [44] (DOFs=100 000 000) and the LES of Lysenko et al. [43] (DOFs=5 760 000). The vorticity profile is quite different in the first half of the cylinder from the
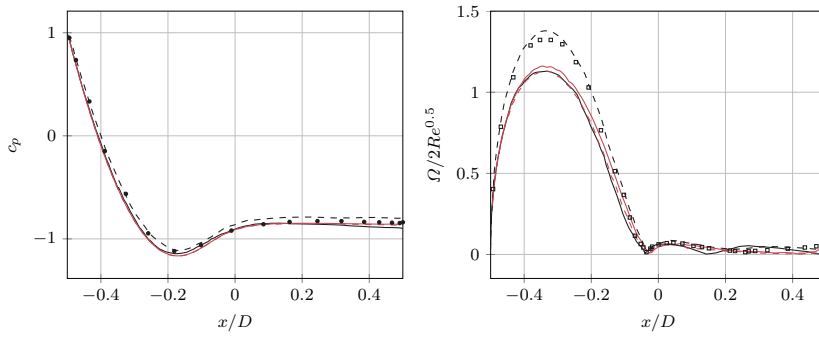
Fig. 13 Flow past a circular cylinder – Spanwise- and time-averaged pressure coefficients c_p (left) and non-dimensional wall vorticity $\Omega/2Re^{0.5}$ (right) distribution on the wall of the cylinder, $\mathbb{P}^{1\to5}$ solution. • Norberg exp $Re = 3000$ [52], ◦ Son and Hanratty exp [62], —— Ma et al. DNS [44], - - - Lysenko et al. LES k-eq. [43], —— DG-$\mathbb{P}^n$ mesh A, - - - DG-$\mathbb{P}^{1\to5}$ mesh B

experimental data of Son and Hanratty [62], but the distribution is almost superimposed with DNS of Ma et al. [44]. The *p*-adaptation strategy allows to provide almost "mesh independent" time-averaged solutions with a number of DOFs roughly one and two order of magnitude lower than LES [43] and DNS [44] computations, respectively.

Figure 14 compares the spanwise- and time-averaged streamwise u_x/u_{ref} and crosswise u_y/u_{ref} velocity profiles at different locations in the wake of the cylinder $x/D = \{1.06, 1.54, 2.02\}$ with the numerical results available in the literature [43,68] and the experiments from Parnaudeau et al. [57]. These quantities confirm a good matching between experimental and numerical results. In this case only streamwise velocities can be considered almost grid independent, while some discrepancies between mesh A and B are observed for the crosswise velocities. However, our numerical experiments on the compressible case [12] suggest that this difference could be ascribed to an insufficient averaging period/window.

Fig. 15 compares the streamwise u_x/u_{ref} velocity along the centerline in the wake of the cylinder with the numerical results available in the literature [43,68] and the experiments from Parnaudeau et al. [57] and Ong et al. [53]. The length of the recirculation bubble is predicted correctly, and it is in agreement with the experiment of Parnaudeau et al. [57]. The value of u_x/u_{ref} far from the circular cylinder is in good agreement with the experimental value of Ong et al. [53].

Figure 16 compares the spanwise- and time-averaged streamwise $u'_x u'_x/u^2_{ref}$ and crosswise $u'_y u'_y/u^2_{ref}$ velocity fluctuations with the numerical results available in the literature [43,68] and the experiments from Parnaudeau et al. [57]. The agreement with the reference data is generally good, even if locally some discrepancies can be observed. Some differences between mesh A and B are also present, but they could be ascribed also in this case to a not sufficient time of average. Finally, Fig. 17 presents the comparison between all the veloc-
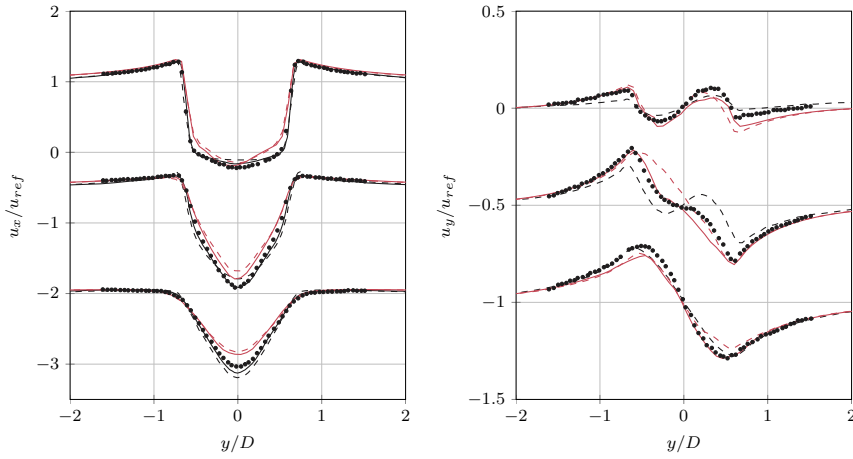
Fig. 14 Flow past a circular cylinder – Spanwise- and time-averaged streamwise u_x/u_{ref} and crosswise u_y/u_{ref} velocity at different locations in the wake of the cylinder $x/D = \{1.06, 1.54, 2.02\}$, $\mathbb{P}^{1\rightarrow5}$ solution. • Parnaudeau et al. exp [57], - - - Lysenko et al. LES k-eq. [43],, —— Wissink and Rodi DNS [68], —— DG-$\mathbb{P}^{1\rightarrow5}$ mesh A, - - - DG-$\mathbb{P}^{1\rightarrow5}$ mesh B
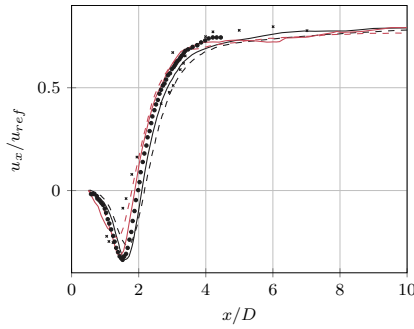

Fig. 15 Flow past a circular cylinder – Spanwise- and time-averaged streamwise velocity u_x/u_{ref} along the centerline ($y/D = 0$) in the wake of the cylinder, $\mathbb{P}^{1\rightarrow5}$ solution. • Parnaudeau et al. exp [57], · Ong et al. exp [53], - - - Lysenko et al. LES k-eq. [43], —— Wissink and Rodi DNS [68], —— DG-$\mathbb{P}^{1\rightarrow5}$ mesh A, - - - DG-$\mathbb{P}^{1\rightarrow5}$ mesh B

ities and velocity fluctuations profiles, at different locations in the wake of the cylinder $x/D = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10\}$, with the DNS performed by Wissink and Rodi [68]. In this case a very good matching is observed. Comparing the results of grids A and B, an almost mesh independent behaviour is shown with the exception of the first station. At the first station, the polynomial distribution of the mesh A seems to guarantee a better resolution.
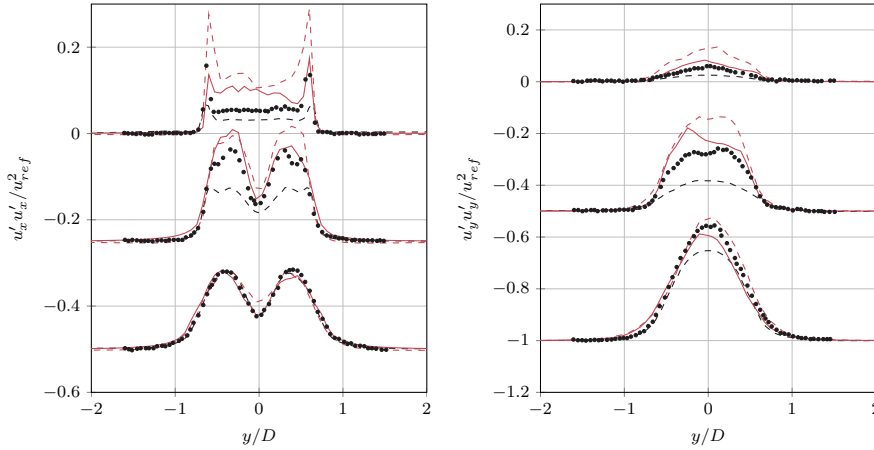
Fig. 16 Flow past a circular cylinder – Spanwise- and time-averaged streamwise $u'_x u'_x / u^2_{ref}$ and crosswise $u'_y u'_y / u^2_{ref}$ velocity fluctuations at different locations in the wake of the cylinder $x/D = \{1.06, 1.54, 2.02\}$, $\mathbb{P}^{1\to5}$ solution. • Parnaudeau et al. exp [57], - - - Lysenko et al. LES *k*-eq. [43], —— *DG*-$\mathbb{P}^{1\to5}$ mesh A, - - - *DG*-$\mathbb{P}^{1\to5}$ mesh B

4 Conclusion

The paper presents a strategy to enhance the computational efficiency of an implicit DG solver for the scale resolving simulation of incompressible turbulent flows. The proposed approach exploits the coupling of a *p*-adaptive solution process and a *p*-MG preconditioner.

The *p*-adaptive algorithm aims at reducing the spatial discretization error by locally varying the polynomial degree of the solution according to proper error estimators. The algorithm also takes advantage of *i*) a multi-constraint domain decomposition algorithm to mitigate the computational load imbalance induced by adaptation, and *ii*) an adaptive algorithm for the degree of exactness of the quadrature rules to reduce, without loss of accuracy, the operator assembly cost.

To further improve the solver performance in terms of memory footprint and the CPU time, a matrix-free FGMRES implementation has been considered together with a *p*-MG preconditioner that can exploit element-wise approximations of the implicit operator. This multilevel preconditioner, proposed in [27], was here extended to a variable polynomial degree framework, where the finest level operator is a partitioned matrix with blocks of variable size, by using coarser levels of (almost) uniform degree over the mesh.

The performance and accuracy of the method have been assessed by considering the following test cases: *i*) the T3L ERCOFTAC case with different levels of inlet turbulence, *ii*) the flow past a sphere at $Re_D = 300$, and *iii*) the flow past a circular cylinder at $Re_D = 3\,900$.

The T3L case has been used to show the dramatic improvement of the solver efficiency due to the only use of an effective preconditioner for the
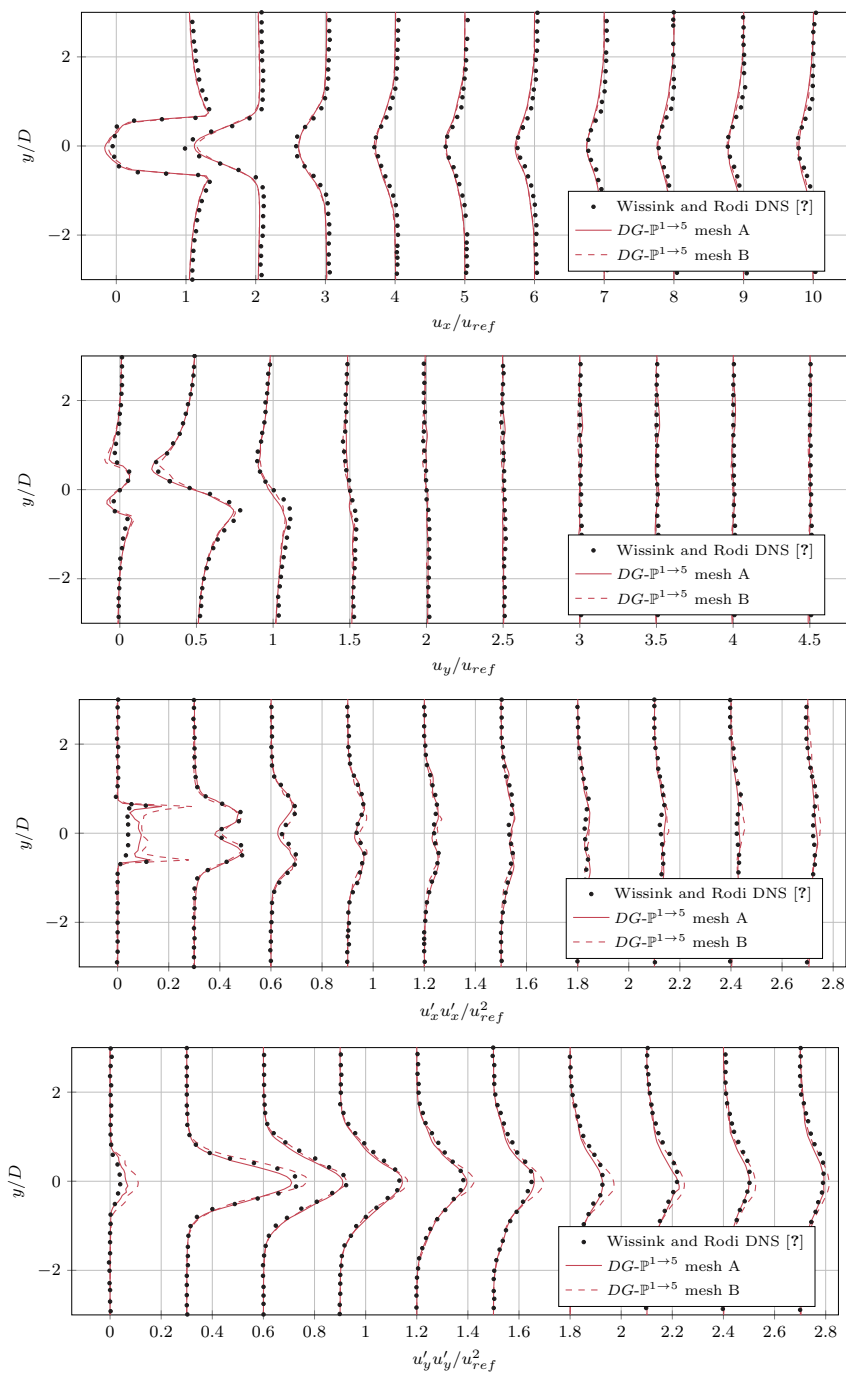
Fig. 17 Flow past a circular cylinder – Spanwise- and time-averaged streamwise and crosswise velocity, u_x/u_{ref} and u_y/u_{ref}, and streamwise and crosswise velocity fluctuations, $u'_x u'_x/u^2_{ref}$ and $u'_y u'_y/u^2_{ref}$, at different locations in the wake of the cylinder $x/D = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10\}$

linear systems solution, here the p-multigrid preconditioner. This flow problem has been also used to demonstrate, in detail and for different Tu values, the promising predicting capabilities of the numerical method when applied to the ILES.

Once the the good properties of the underlying DG method have been proved, the positive impact of the p-adaptive strategy on the solver performance was quantified by computing the unsteady laminar flow past a sphere at $\mathrm{Re}_D = 300$. Indeed, for a given target accuracy, the overall number of DOFs and the CPU time are always in favor of the adaptive strategy when compared to uniform-degree discretizations.

Finally, as a proof of concept, the p-adaptive ILES of the turbulent flow past a circular cylinder was considered. The proposed approach was able to deliver an almost mesh-independent solution for many quantities of interest (c_p, c_f, and averaged velocities) when considering two very coarse unstructured meshes. The results are in reasonable agreement with the literature but with a number of DOFs roughly 1-2 order of magnitude smaller than the references.

Future works involve a thorough investigation of the parameters of the p-multigrid preconditioner, *e.g.*, number of levels and orders distributions over the levels, as well as the study of alternatives strategies for the definition of the coarser levels operators, *e.g.*, strongly non-uniform order coarser levels. More complex flow problems will be considered in a forthcoming paper to demonstrate the effective potential of the promising p-adaptive method devised in this work.

Compliance with Ethical Standards:

Conflict of Interest: The authors declare that they have no conflict of interest.

References

1. 4th International workshop on High-Order CFD Methods. `https://how4.cenaero.be/news`
2. Alam, M., Sandham, N.D.: Direct numerical simulation of 'short' laminar separation bubbles with turbulent reattachment. Journal of Fluid Mechanics **410**, 1–28 (2000). DOI 10.1017/S0022112099008976
3. Arnold, D.N., Brezzi, F., Cockburn, B., Marini, D.: Unified analysis of discontinuous Galerkin methods for elliptic problems. SIAM J. Numer. Anal. **39**(5), 1749–1779 (2002)
4. Balay, S., Abhyankar, S., Adams, M.F., Brown, J., Brune, P., Buschelman, K., Dalcin, L., Dener, A., Eijkhout, V., Gropp, W.D., Karpeyev, D., Kaushik, D., Knepley, M.G., May, D.A., McInnes, L.C., Mills, R.T., Munson, T., Rupp, K., Sanan, P., Smith, B.F., Zampini, S., Zhang, H., Zhang, H.: PETSc Web page. `https://www.mcs.anl.gov/petsc` (2019). URL `https://www.mcs.anl.gov/petsc`

5. Balay, S., Abhyankar, S., Adams, M.F., Brown, J., Brune, P., Buschelman, K., Dalcin, L., Dener, A., Eijkhout, V., Gropp, W.D., Karpeyev, D., Kaushik, D., Knepley, M.G., May, D.A., McInnes, L.C., Mills, R.T., Munson, T., Rupp, K., Sanan, P., Smith, B.F., Zampini, S., Zhang, H., Zhang, H.: PETSc users manual. Tech. Rep. ANL-95/11 - Revision 3.12, Argonne National Laboratory (2019). URL `https://www.mcs.anl.gov/petsc`
6. Balay, S., Gropp, W.D., McInnes, L.C., Smith, B.F.: Efficient management of parallelism in object oriented numerical software libraries. In: E. Arge, A.M. Bruaset, H.P. Langtangen (eds.) Modern Software Tools in Scientific Computing, pp. 163–202. Birkhäuser Press (1997)
7. Bassi, F., Botti, L., Colombo, A., Crivellini, A., De Bartolo, C., Franchina, N., Ghidoni, A., Rebay, S.: Time Integration in the discontinuous Galerkin Code MIGALE - Steady Problems. In: N. Kroll, C. Hirsch, F. Bassi, C. Johnston, K. Hillewaert (eds.) IDIHOM: Industrialization of High-Order Methods - A Top-Down Approach, *Notes on Numerical Fluid Mechanics and Multidisciplinary Design*, vol. 128, pp. 179–204. Springer International Publishing, Cham (2015)
8. Bassi, F., Botti, L., Colombo, A., Crivellini, A., Ghidoni, A., Massa, F.: On the development of an implicit high-order Discontinuous Galerkin method for DNS and implicit LES of turbulent flows. European Journal of Mechanics - B/Fluids **55**, 367–379 (2016). DOI https://doi.org/10.1016/j.euromechflu.2015.08.010
9. Bassi, F., Botti, L., Colombo, A., Crivellini, A., Ghidoni, A., Nigro, A., Rebay, S.: Time Integration in the discontinuous Galerkin Code MIGALE - Unsteady Problems. In: N. Kroll, C. Hirsch, F. Bassi, C. Johnston, K. Hillewaert (eds.) IDIHOM: Industrialization of High-Order Methods - A Top-Down Approach, *Notes on Numerical Fluid Mechanics and Multidisciplinary Design*, vol. 128, pp. 205–230. Springer International Publishing (2015). URL `http://dx.doi.org/10.1007/978-3-319-12886-3_11`
10. Bassi, F., Botti, L., Colombo, A., Di Pietro, D.A., Tesini, P.: On the flexibility of agglomeration based physical space discontinuous Galerkin discretizations. Journal of Computational Physics **231**(1), 45–65 (2012)
11. Bassi, F., Botti, L., Colombo, A., Ghidoni, A., Massa, F.: Linearly implicit Rosenbrock-type Runge–Kutta schemes applied to the discontinuous Galerkin solution of compressible and incompressible unsteady flows. Computers & Fluids **118**, 305–320 (2015)
12. Bassi, F., Colombo, A., Crivellini, A., Fidkowski, K., Franciolini, M., Ghidoni, A., Noventa, G.: An entropy-adjoint *p*-adaptive discontinuous Galerkin method for the under-resolved simulation of turbulent flows. DOI 10.2514/6.2019-3418
13. Bassi, F., Colombo, A., Crivellini, A., Fidkowski, K., Franciolini, M., Ghidoni, A., Noventa, G.: An entropy-adjoint *p*-adaptive discontinuous Galerkin method for the under-resolved simulation of turbulent flows. *Accepted for publication on AIAA journal* (2020)
14. Bassi, F., Colombo, A., Crivellini, A., Franciolini, M., Ghidoni, A., Manzinali, G., Noventa, G.: Under-resolved simulation of turbulent flows using a *p*-adaptive discontinuous Galerkin method. pp. 157–162 (2019). DOI 10.1007/978-3-030-22196-6_25
15. Bassi, F., Crivellini, A., Di Pietro, D.A., Rebay, S.: An artificial compressibility flux for the discontinuous Galerkin solution of the incompressible Navier-Stokes equations. J. Comput. Phys. **218**, 794–815 (2006)
16. 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: R. Decuypere, G. Dibelius (eds.) Proceedings of the 2nd European Conference on Turbomachinery Fluid Dynamics and Thermodynamics, pp. 99–108. Technologisch Instituut, Antwerpen, Belgium (1997)
17. Botti, L., Colombo, A., Bassi, F.: *h*-multigrid agglomeration based solution strategies for discontinuous Galerkin discretizations of incompressible flow problems. Journal of Computational Physics **347**, 382–415 (2017)
18. Botti, L., Colombo, A., Crivellini, A., Franciolini, M.: {h-p-hp}-Multilevel discontinuous Galerkin solution strategies for elliptic operators. International Journal of Computational Fluid Dynamics pp. 1–9 (2019)
19. Breuer, M.: Large eddy simulation of the subcritical flow past a circular cylinder: numerical and modeling aspects. International Journal for Numerical Methods in Fluids **28**(9), 1281–1302 (1998)

20. Brezzi, F., Manzini, G., Marini, D., Pietra, P., Russo, A.: Discontinuous Galerkin approximations for elliptic problems. Numer. Methods Partial Differential Equations **16**, 365–378 (2000)
21. Castro, I.P., Epik, E.: Boundary layer development after a separated region. Journal of Fluid Mechanics **374**, 91–116 (1998). DOI 10.1017/S0022112098002420
22. Colombo, A., Manzinali, G., Ghidoni, A., Noventa, G., Franciolini, M., Crivellini, A., Bassi, F.: A *p*-adaptive implicit discontinuous Galerkin method for the under-resolved simulation of compressible turbulent flows. pp. 4159–4170 (2020)
23. Coupland, J., Brierley, D.: Transition in turbomachinery flows. final report. BRITE/EURAM Project AERO-CT92-0050. Measurements available at the ERCOFTAC site (1996)
24. Crivellini, A., Bassi, F.: An implicit matrix-free discontinuous Galerkin solver for viscous and turbulent aerodynamic simulations. Computers and Fluids **50**(1), 81–93 (2011). DOI 10.1016/j.compfluid.2011.06.020
25. Crivellini, A., D'Alessandro, V., Bassi, F.: Assessment of a high-order discontinuous galerkin method for incompressible three-dimensional navier-stokes equations: Benchmark results for the flow past a sphere up to $re = 500$. Computers & Fluids **86**, 442–458 (2013). DOI https://doi.org/10.1016/j.compfluid.2013.07.027. URL `http://www.sciencedirect.com/science/article/pii/S004579301300306X`
26. Diosady, L.T., Darmofal, D.L.: Preconditioning methods for discontinuous Galerkin solutions of the Navier–Stokes equations. Journal of Computational Physics **228**(11), 3917–3935 (2009)
27. Franciolini, M., Botti, L., Colombo, A., Crivellini, A.: *p*-multigrid matrix-free discontinuous Galerkin solution strategies for the under-resolved simulation of incompressible turbulent flows. *Accepted for publication in Computers & Fluids* (2020)
28. Franciolini, M., Crivellini, A., Nigro, A.: 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). DOI 10.1016/j.compfluid.2017.10.008
29. Franciolini, M., Fidkowski, L., Crivellini, A.: Efficient discontinuous Galerkin implementations and preconditioners for implicit unsteady compressible flow simulations. Computers & Fluids **203**, 104542 (2020). DOI https://doi.org/10.1016/j.compfluid.2020.104542
30. Garai, A., Diosady, L., Murman, S., Madavan, N.: Scale-resolving simulations of bypass transition in a high-pressure turbine cascade using a spectral element discontinuous Galerkin method. Journal of Turbomachinery **140**(3) (2018). DOI 10.1115/1.4038403
31. Gassner, G., Staudenmaier, M., Hindenlang, F., Atak, M., Munz, C.D.: A space–time adaptive discontinuous Galerkin scheme. Computers & Fluids **117**, 247 – 261 (2015). DOI https://doi.org/10.1016/j.compfluid.2015.05.002. URL `http://www.sciencedirect.com/science/article/pii/S0045793015001498`
32. Helenbrook, B., Mavriplis, D., Atkins, H.: Analysis of *p*-multigrid for continuous and discontinuous finite element discretizations. In: 16th AIAA Computational Fluid Dynamics Conference, p. 3989 (2003)
33. Hillier, R., Cherry, N.: The effects of stream turbulence on separation bubbles. Journal of Wind Engineering and Industrial Aerodynamics **8**(1), 49 – 58 (1981). DOI https://doi.org/10.1016/0167-6105(81)90007-6
34. Jan S. Hesthaven, T.W.: Nodal Discontinuous Galerkin Methods. Springer (2008)
35. Karypis, G., Kumar, V.: METIS: Unstructured graph partitioning and sparse matrix ordering system, version 5.0. Tech. rep. (2009). URL `http://www.cs.umn.edu/~metis`
36. Knoll, D., Keyes, D.: Jacobian-free Newton-Krylov methods: a survey of approaches and applications. J. Comput. Phys. **193**(2), 357–397 (2004)
37. Kravchenko, A.G., Moin, P.: Numerical studies of flow over a circular cylinder at Red=3900. Physics of Fluids **12**(2), 403–417 (2000). DOI 10.1063/1.870318
38. Krivodonova, L., Xin, J., Remacle, J.F., Chevaugeon, N., Flaherty, J.E.: Shock detection and limiting with discontinuous Galerkin methods for hyperbolic conservation laws. Applied Numerical Mathematics **48**(3-4), 323–338 (2004)
39. Lamballais, E., Silvestrini, J., Laizet, S.: Direct numerical simulation of flow separation behind a rounded leading edge: Study of curvature effects. International Journal of Heat and Fluid Flow **31**(3), 295 – 306 (2010). DOI https://doi.org/10.1016/

j.ijheatfluidflow.2009.12.007. URL `http://www.sciencedirect.com/science/article/pii/S0142727X09001726`. Sixth International Symposium on Turbulence and Shear Flow Phenomena

40. Lang, J., Verwer, J.: ROS3P-An accurate third-order Rosenbrock solver designed for parabolic problems. BIT Numerical Mathematics **41**(4), 731–738 (2001). DOI 10.1023/A:1021900219772

41. Langari, M., Yang, Z.: Numerical study of the primary instability in a separated boundary layer transition under elevated free-stream turbulence. Physics of Fluids **25**(7), 074106 (2013). DOI 10.1063/1.4816291

42. de la Llave Plata, M., Couaillier, V., le Pape, M.C.: On the use of a high-order discontinuous Galerkin method for DNS and LES of wall-bounded turbulence. Computers and Fluids **176**, 320–337 (2018). DOI 10.1016/j.compfluid.2017.05.013

43. Lysenko, D.A., Ertesvåg, I.S., Rian, K.E.: Large-Eddy Simulation of the flow over a circular cylinder at Reynolds number 3900 using the OpenFOAM toolbox. Flow, Turbulence and Combustion **89**(4), 491–518 (2012). DOI 10.1007/s10494-012-9405-0. URL `https://doi.org/10.1007/s10494-012-9405-0`

44. Ma, X., Karamanos, G.S., Karniadakis, G.: Dynamics and low-dimensionality of a turbulent near wake. Journal of Fluid Mechanics **410**, 29–65 (2000). DOI 10.1017/S0022112099007934

45. Massa, F., Noventa, G., Lorini, M., Bassi, F., Ghidoni, A.: High-order linearly implicit two-step peer schemes for the discontinuous Galerkin solution of the incompressible Navier–Stokes equations. Computers & Fluids **162**, 55–71 (2018). DOI 10.1016/j.compfluid.2017.12.003

46. Mavriplis, D.J.: An assessment of linear versus nonlinear multigrid methods for unstructured mesh solvers. Journal of Computational Physics **175**(1), 302 – 325 (2002). DOI https://doi.org/10.1006/jcph.2001.6948

47. Moura, R., Mengaldo, G., Peiró, J., Sherwin, S.: On the eddy-resolving capability of high-order discontinuous Galerkin approaches to implicit LES / under-resolved DNS of Euler turbulence. Journal of Computational Physics **330**, 615–623 (2017). DOI 10.1016/j.jcp.2016.10.056

48. Moura, R., Sherwin, S., Peiró, J.: Linear dispersion–diffusion analysis and its application to under-resolved turbulence simulations using discontinuous Galerkin spectral/hp methods. Journal of Computational Physics **298**, 695–710 (2015)

49. Naddei, F., De La Llave Plata, M., Couaillier, V.: A comparison of refinement indicators for p-adaptive discontinuous Galerkin methods for the Euler and Navier-Stokes equations (2018). DOI 10.2514/6.2018-0368

50. Naddei, F., de la Llave Plata, M., Couaillier, V., Coquel, F.: A comparison of refinement indicators for p-adaptive simulations of steady and unsteady flows using discontinuous Galerkin methods. Journal of Computational Physics **376**, 508–533 (2019). DOI 10.1016/j.jcp.2018.09.045

51. Nakamura, Y., Ozono, S.: The effects of turbulence on a separated and reattaching flow. Journal of Fluid Mechanics **178**, 477–490 (1987). DOI 10.1017/S0022112087001320

52. Norberg, C.: Flow around a circular cylinder: aspects of fluctuating lift. Journal of Fluids and Structures **15**(3), 459 – 469 (2001)

53. Ong, L., Wallace, J.: The velocity field of the turbulent very near wake of a circular cylinder. Experiments in Fluids **20**(6), 441–453 (1996). DOI 10.1007/BF00189383. URL `https://doi.org/10.1007/BF00189383`

54. Ouvrard, H., Koobus, B., Dervieux, A., Salvetti, M.: Classical and variational multiscale LES of the flow around a circular cylinder on unstructured grids. Computers and Fluids **39**(7), 1083–1094 (2010). DOI 10.1016/j.compfluid.2010.01.017

55. Palikaras, A., Yakinthos, K., Goulas, A.: The effect of negative shear on the transitional separated flow around a semi-circular leading edge. International Journal of Heat and Fluid Flow **24**(3), 421 – 430 (2003)

56. Park, J., Witherden, F., Vincent, P.: High-order implicit large-eddy simulations of flow over a NACA0021 aerofoil. AIAA Journal **55**(7), 2186–2197 (2017). DOI 10.2514/1.J055304

57. Parnaudeau, P., Carlier, J., Heitz, D., Lamballais, E.: Experimental and numerical studies of the flow over a circular cylinder at Reynolds number 3900. Physics of Fluids **20**(8), 085101 (2008). DOI 10.1063/1.2957018. URL `https://doi.org/10.1063/1.2957018`

58. Pernice, M., Walker, H.: NITSOL: A Newton iterative solver for nonlinear systems. SIAM Journal on Scientific Computing **19**(1), 302–318 (1998)
59. Persson, P.O., Peraire, J.: Sub-cell shock capturing for discontinuous Galerkin methods. In: 44th AIAA Aerospace Sciences Meeting and Exhibit, p. 112 (2006)
60. Rang, J., Angermann, L.: New Rosenbrock methods of order 3 for PDAEs of index 2. In: Proceedings of Equadiff-11 2005, pp. 385–394. Comenius University Press (Bratislava) (2007)
61. Shahbazi, K., Mavriplis, D.J., Burgess, N.K.: Multigrid algorithms for high-order discontinuous Galerkin discretizations of the compressible Navier–Stokes equations. Journal of Computational Physics **228**(21), 7917–7940 (2009)
62. Son, J.S., Hanratty, T.J.: Velocity gradients at the wall for flow around a cylinder at Reynolds numbers from 5×103 to 105. Journal of Fluid Mechanics **35**(2), 353–368 (1969). DOI 10.1017/S0022112069001157
63. Spalart, P.R.: Direct simulation of a turbulent boundary layer up to $Re_\theta = 1410$. Journal of Fluid Mechanics **187**, 61–98 (1988). DOI 10.1017/S0022112088000345
64. Tesini, P.: An *h*-Multigrid Approach for High-Order discontinuous Galerkin Methods. Ph.D. thesis, Università degli Studi di Bergamo (2008)
65. Tugnoli, M., Abbà, A., Bonaventura, L., Restelli, M.: A locally *p*-adaptive approach for Large Eddy simulation of compressible flows in a DG framework. Journal of Computational Physics **349**, 33–58 (2017). DOI 10.1016/j.jcp.2017.08.007
66. Uranga, A., Persson, P.O., Drela, M., Peraire, J.: Implicit large eddy simulation of transition to turbulence at low Reynolds numbers using a discontinuous Galerkin method. International Journal for Numerical Methods in Engineering **87**(1-5), 232–261 (2011). DOI 10.1002/nme.3036
67. de Wiart, C., Hillewaert, K., Bricteux, L., Winckelmans, G.: Implicit LES of free and wall-bounded turbulent flows based on the discontinuous Galerkin/symmetric interior penalty method. International Journal for Numerical Methods in Fluids **78**(6), 335–354 (2015). DOI 10.1002/fld.4021
68. Wissink, J., Rodi, W.: Numerical study of the near wake of a circular cylinder. International Journal of Heat and Fluid Flow **29**, 1060–1070 (2008). DOI 10.1016/j.ijheatfluidflow.2008.04.001
69. Yang, Z., Voke, P.R.: Large-eddy simulation of boundary-layer separation and transition at a change of surface curvature. Journal of Fluid Mechanics **439**, 305–333 (2001). DOI 10.1017/S0022112001004633