An entropy-adjoint p-adaptive discontinuous Galerkin method for the under-resolved simulation of turbulent flows

Francesco Bassi[1*], Alessandro Colombo[1†], Andrea Crivellini[2‡], Krzysztof J. Fidkowski[3§], Matteo Franciolini[3,4¶], Antonio Ghidoni[5‖], and Gianmaria Noventa[5**]

[1] *University of Bergamo, Dalmine (BG), 24044, Italy*

[2] *Polytechnic University of Marche, Ancona, 60131, Italy*

[3] *University of Michigan, Ann Arbor (MI), 48109, United States*

[4] *NASA Ames Research Center, Moffett Field (CA), 94035, United States*

[5] *University of Brescia, Brescia, 25123, Italy*

The paper presents an approach to mesh adaptation suitable for scale-resolving simulations. The methodology is based on the entropy adjoint approach, which corresponds to a standard output-based adjoint method with the output functional targeting areas of spurious generation of entropy. The method shows several advantages over standard output-based error estimation: *i*) it is computationally inexpensive, *ii*) it does not require the solution of a fine-space adjoint problem, and *iii*) it is nonlinearly stable with respect to the primal solution for chaotic dynamical systems. In addition, the work reports on the parallel efficiency of the solver, which has been optimized through a multi-constraint domain decomposition algorithm available within the Metis 5.0 library. The reliability, accuracy, and efficiency of the approach are assessed by computing three test cases: the

[*] Full Professor, Department of Engineering and Applied Sciences, francesco.bassi@unibg.it

[†] Assistant Professor, Department of Engineering and Applied Sciences, alessandro.colombo@unibg.it

[‡] Associate Professor, Department of Industrial Engineering and Mathematical Sciences, a.crivellini@staff.univpm.it

[§] Associate Professor, Department of Aerospace Engineering, kfid@umich.edu

[¶] NPP/USRA Research Fellow, matteo.franciolini@nasa.gov

[‖] Associate Professor, Department of Mechanical and Industrial Engineering, antonio.ghidoni@unibs.it

[**] Research Fellow, Department of Mechanical and Industrial Engineering, gianmaria.noventa@unibs.it

two-dimensional, laminar, chaotic flow around a square at $Re = 3\,000$, the implicit Large Eddy Simulation (LES) of the flow past a circular cylinder at $Re = 3\,900$ and past the a square cylinder at $Re = 22\,000$. The results show a significant reduction in the number of DoFs with respect to uniform order-refinement with a good agreement with experimental data.

I. Introduction

The growing need for highly accurate flow simulations for the prediction of problem-specific output quantities has paved the way for higher-order methods such as the discontinuous Galerkin (DG) methods. Although for many flow problems, with typical engineering error tolerances, DG methods are still generally less efficient than standard codes, the industrial interest in high-fidelity simulation tools is strongly fostering research on efficient high-order CFD solvers.

Previous works contributed to the development of efficient high-order numerical methods for steady and unsteady flow problems involving adaptation of the spatial discretization by varying the order of polynomial approximation throughout the domain[1,2] or by performing mesh adaptation.[3–6] In this paper, a p-adaptation strategy has been adopted to increase the computational efficiency of a DG solver[7] for scale-resolving simulations performed according to the Implicit Large Eddy Simulation (ILES), or under-resolved Direct Numerical Simulation (uDNS), approach. The proposed strategy allows changing the polynomial order of the solution representation within each element according to an error estimate, thus reducing the CPU time and memory usage, while not spoiling the spectral resolution required by this class of simulations. In the ILES approach,[8–14] the unfiltered 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.

As the work concerns unsteady flow simulations, an adaptation procedure suitable for time-dependent problems must be devised. Many error indicators have been proposed for adaptation in a LES context, which are based on energy dissipation,[15] small-scale kinetic energy,[16] and on more comprehensive definitions including all sources of error (discretization and modelling) in the LES framework.[17,18] In addition, recently published algorithms, *e.g.* ensemble-adjoint[19,20] and the Least Square Shadowing (LSS)[21] approaches, allow to compute accurately the sensitivities of target output quantities with respect to given parameters for flows exhibiting chaotic regimes, where standard unsteady adjoint methods provide unbounded flow sensitivities which diverge backwards in time. The ensemble-adjoint approach involves the computation of the sensitivity as the ensemble-average of those obtained using

standard adjoint-based techniques applied over a short enough time window that prevents divergence phenomena. Despite showing mathematically consistent results, theoretical estimates[22] reveal that the computational complexity of the method makes it unfeasible for relevant engineering applications. On the other hand, implementations of the LSS approach demonstrated to be able to compute bounded and accurate sensitivities for turbulent flow simulations,[23,24] but the computational cost is extremely high, as the solution of an optimization problem within the adjoint problem is required. For these reasons, alternative and more practical approaches to mesh adaptation for turbulent flows are still of interest to improve the computational efficiency of simulations involving chaotic dynamics.

Our previous research efforts[25,26] aimed at adapting the polynomial degree by combining two simple element-wise indicators based on interface pressure jumps and on the decay of the coefficients of the modal expansion as in.[27] These sensors are coupled to guarantee a reasonable behavior both for high- and low-degree polynomial approximations. Despite not targeting any output quantity, the strategy performed reasonably well in the solution of statistically steady turbulent flows.

The objective of the present work is to extend the previous approach by using error estimates provided by the so called entropy-adjoint approach introduced by Fidkowski and Roe.[28] The main idea of the method is to measure the balance of entropy throughout the domain, including the inflow, outflow, and generation. The output functional is associated to the adjoint problem, whose solution is the set of entropy variables associated with the state. By targeting the source of spurious entropy generation, which is closely related to the numerical error (the only source of error in the ILES context), the approach overcomes the drawback of many heuristic sensors, which are generally not robust for controlling numerical discretization errors. The approach is able to target areas of spurious entropy generation, which mainly affect the scalar output that measures net entropy balance throughout the domain. The method shows several advantages over standard, output-based adaptive simulations for steady flows,[29,30] since the adjoint solution is easily obtained by manipulating the state vector. This property can be exploited also for unsteady flow problems, avoiding an often prohibitively expensive backward-in-time integration, typical of unsteady adjoint applications. Moreover, the adjoint solution is nonlinearly stable, meaning that the accuracy of the error estimate is not spoiled by the chaotic nature of the flow.

Output-based error estimation typically involves the computation of a fine-space adjoint solution[6,17,18,31–34] which can be accomplished by solving an approximate fine-space problem[35,36] using few smoothing iterations. However, building the fine space problem can be considerably expensive for a three dimensional LES case. Therefore, with the idea of minimizing the costs associated with the fine space solution on three dimensional cases, we here implement a higher-order reconstruction approach. The method, hereby called patch recon-

struction, consists of interpolating the variables from an extended stencil, made of each mesh element and its neighbours, into an enriched polynomial space.

The reliability, accuracy and efficiency of the approach are assessed by computing two and three dimensional test cases involving compressible, low Mach number, unsteady flows. We first report numerical experiments to show the optimal parallel efficiency of the method on multi-core machines, which is obtained through a dynamic load balancing based on the Metis library and its capability of generating weighted graphs. Such an implementation handles imbalances in the degrees of freedom on each partition due to the application of the adaptation algorithm, and provides a nearly optimal scalability, comparable to that obtained using a fixed number of degrees of freedom. Secondly, we assess the implementation on a two-dimensional test case, the laminar chaotic flow past a square at $Re = 3\,000$. Finally, we report ILES of two span-wise periodic turbulent cases, *i.e.*, the flow around the circular cylinder at $Re = 3\,900$ and around a square cylinder at $Re = 22\,000$. The results show a significant reduction in the overall number of DoFs with respect to uniform order approximations while delivering a good agreement with the literature.

II. The numerical framework

In this work two different DG solvers have been used for the 2D and 3D computations, i.e., the XFLOW solver[28] and the code MIGALE,[7] respectively.

A. Space and time discretization

The compressible Navier-Stokes equations for m variables in d dimensions can be written in compact form as

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

where $\mathbf{w} \in \mathbb{R}^m$ is the unknown solution vector, $\mathbf{F}_c, \mathbf{F}_v \in \mathbb{R}^m \otimes \mathbb{R}^d$ are the convective and viscous flux functions, and $\mathbf{P}\left(\mathbf{w}\right) \in \mathbb{R}^m \otimes \mathbb{R}^m$ is a transformation matrix that takes into account the possible use of a set of unknowns $\mathbf{w}$ different from the conservative set $\mathbf{w}_c = [\rho, \rho u_i, \rho E]^T$. Both XFLOW and MIGALE discretize in space the governing equations according to the DG approach[7,37] but with some differences in the implementation of the method. For example, the XFLOW solver relies on the set of conservative variables ($\mathbf{P} = \mathbf{I}$) while code MIGALE uses a primitive set given by $\mathbf{w} = [p, u_i, T]^T$. To perform the spatial discretization, the weak form of the governing equations is first obtained by multiplying Eq. (1) by an arbitrary, smooth 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, both belonging to the finite-dimensional set $\mathcal{V}_h := [\mathbb{P}_d^k(\mathcal{T}_h)]^m$, where $\mathbb{P}_d^k(\mathcal{T}_h) := \{v_h \in L^2(\Omega) \,|\, v_{h|K} \in \mathbb{P}_d^k(K),\, \forall K \in \mathcal{T}_h\}$ is the discrete polynomial space and $\mathbb{P}_d^k(K)$ denotes the restriction of the polynomial functions of $d = 2, 3$ variables and total degree k to the element K belonging to a tessellation $\mathcal{T}_h = \{K\}$ of the computational domain. While the XFLOW solver uses a set of Lagrange basis functions defined in reference elements that can be mapped onto the cells of the triangulation, MIGALE employs a set of hierarchical and orthonormal functions computed in the physical (mesh) space according to Bassi *et al.*[38] As the functional approximation space is discontinuous, the flux functions over mesh faces are not uniquely defined and the convective and viscous fluxes need to be replaced with numerical counterparts. In this work, both solvers use the BR2 scheme of Bassi and Rebay[39] for the viscouos flux discretization, while for the convective part XFLOW uses the approximate Riemann solver of Roe[40] and MIGALE implements the exact Riemann solver of Gottlieb and Groth.[41]

By assembling together all the elemental contributions of the DG discretization, the system of ordinary differential equations governing the evolution in time of the discrete solution can be written as

$$\frac{d\mathbf{W}}{dt} + \widetilde{\mathbf{R}}\left(\mathbf{W}\right) = \mathbf{0}, \quad \text{with} \quad \widetilde{\mathbf{R}}\left(\mathbf{W}\right) = \mathbf{M_P}^{-1}\left(\mathbf{W}\right)\mathbf{R}\left(\mathbf{W}\right), \tag{2}$$

where $\mathbf{W}$ is the global vector of unknown degrees of freedom, $\mathbf{M_P}$ is the global block diagonal mass matrix, and $\mathbf{R}\left(\mathbf{W}\right)$ is the vector of spatial residuals. Both the solvers implement several highly accurate time integration schemes.[7,37] For the computations presented in this paper the XFLOW solver uses a fifth-order Explicit Singly Diagonal Implicit Runge-Kutta (ESDIRK) scheme while MIGALE uses a multi-stage, linearly implicit, Rosenbrock-type, Runge-Kutta scheme with three stages and third-order accuracy (ROS3P).[42,43] In both implementations, the Jacobian matrix is computed analytically and a preconditioned GMRES algorithm is used to solve the linear systems arising from the temporal discretization. Code MIGALE exploits the PETSc library for linear algebra and to deal with distributed arrays and communication.[44]

B. Efficient adaptive quadrature rules

A key aspect of solver efficiency is represented by a proper choice of the degree of exactness (DOE) of the quadrature rules. In fact, the number of integration points rapidly increases when dealing with high-degree polynomial approximations and curved mesh elements. It is worth noting that the use of curved mesh elements is typically limited to the proximity of a curved solid boundary, while in the remainder of the domain straight-sided cells and faces are employed. To avoid over-integration on those elements, it is of primary importance to

recognize the minimum quadrature requirements of each element and face.

Inspired by Bassi et al.,[38] a simple indicator for the integration error, which allows for identification of the actual curved-sided elements in a mesh, is defined as

$$\varepsilon_{i,K} = \frac{|m^{*}_{ii} - m^{ex}_{ii}|}{|m^{ex}_{ii}|}, \quad \forall K \in \mathcal{K}_h, \tag{3}$$

where m^{ex}_{ii} and m^{*}_{ii} are the values of the i-th diagonal entry of the local mass matrix computed using quadrature rules with an exact and a "reduced" DOE, respectively. According to this definition and to a user-defined tolerance tol_q, for each element K, an integration rule with the minimum degree of exactness that satisfies

$$\max_{i \in 1,\dots,N^{K}_{DoFs}} \varepsilon_{i,K} \leq tol_q, \quad \forall K \in \mathcal{K}_h, \tag{4}$$

is used with significant CPU time savings. A similar procedure, considering the surface integrals of the same term is used to reduce the DOE for the mesh faces. The DOE adaptation process is performed only once during pre-processing and after each adaptation step during the solution.

III. Adaptation strategy

The adaptation of the elemental polynomial degree is driven by an error estimator that identifies regions of the domain that lack/exceed a required resolution. These regions can be refined/coarsened by increasing/decreasing the degree of the polynomial approximation of the solution. Regardless of the adopted error estimator and CFD solver, the pseudo code of the adaptation procedure is reported in Algorithm 1. Here, $\hat{k}$ is the polynomial degree at the beginning of the computation, N_{cyc} is the total number of time steps of the simulation, k_{max} is the maximum allowable polynomial degree defined by the user, $\mathcal{N}$ is the number of time steps between two adaptation cycles or between the simulation beginning and the first adaptation cycle, $\mathcal{G}_r$ is the percentage of the total number of elements that will be marked for refinement, $\mathcal{G}_c$ is the percentage of the total number of elements that will be marked for coarsening, n_{adp} is the number of adaptation cycles to be performed, POS_K is the position of the element numbered from zero and sorted in increasing order according to the estimator η^{ψ}_{K} (see Eq. (9)). The algorithm has been written in a general form, considering refining and coarsening. However, the simulations have been performed starting from coarse meshes, where coarsening is not necessary. This strategy, as shown by some numerical investigations, improves the computational efficiency of the simulation. The value of G_r has been set to 0.15 for the XFLOW code (2D simulations) and to 0.2

Algorithm 1 Adaptation algorithm

$l = 0$
$k_K = \hat{k}\ \forall K \in \mathcal{T}_h$
for $i_{cyc} = 1$ **to** N_{cyc} **do**
  integrate the governing equation in time
  evaluate the time-averaged solution, $\overline{\mathbf{W}}$
  if $mod(i_{cyc}, \mathcal{N}) = 0$ **and** $l \leq n_{adp}$ **then**
    $l \leftarrow l + 1$
    compute the estimators $\eta_K^{\psi}\ \forall K \in \mathcal{T}_h$
    for $K \in \mathcal{T}_h$ **do**
      if $POS_K \geq (1 - \mathcal{G}_r) card(\mathcal{T}_h)$ **then**
        $k_K \leftarrow \min(k_K + 1, k_{max})$
      else if $POS_K \leq (\mathcal{G}_c) card(\mathcal{T}_h)$ **then**
        $k_K \leftarrow \max(k_K - 1, 1)$
      end if
    end for
    balance the load among processors via re-partitioning
    L_2 projection of the solution on the new space
  end if
end for

for the MIGALE code (3D simulations). Higher values could refine zones not necessary to capture the main fluid dynamic features, increasing only the computational cost. We remark that orthonormal and hierarchical modal bases greatly simplify the L_2 projection operators: the DoFs of the restricted solution are equal to the low-order subset of their high-order representations, while the DoFs of the prolongated solution are the same as the low-order solution with zero high-order components.

In the context of adaptive LES we can distinguish two different sources of error: one arising from the numerical discretization, and the other from the subgrid-scale model.[17] Minimizing the errors, as already pointed out by Sagaut,[45] would yield asymptotically a DNS resolution, and would remove the advantage of performing an under-resolved simulation. In fact, the objective of adaptive LES should be to adapt the resolution and the size of the LES filter so as to resolve only a prescribed amount of the turbulent scales, while modelling the others. According to these observations, many error indicators have been proposed for LES adaptation, which are based on energy dissipation,[15] small-scale kinetic energy,[16] and on more comprehensive definitions including all sources of error in the LES framework.[17,18] However, when an error indicator specifically designed for LES is combined with adaptive ILES, we obtain at the same time a finer space resolution and a smaller LES filter that will lead to a different corresponding LES with more resolved turbulent scales, but not necessarily with lower error. This behaviour will lead to further refinement, reaching asymptotically a DNS resolution. To avoid the resolution of all turbulent scales, the use of an indicator

based only on the numerical error, such as spurious entropy generation, can represent a viable solution. Whereas more sophisticated indicators, based on adjoint solutions, have been successfully applied to non-chaotic problems, their suitability for LES is not yet clear, as these problems elicit more fundamental questions of convergence of adaptive methods.

A. Output-adjoint sensitivity

Techniques based on output-adjoint sensitivity proved to be efficient and reliable for the analysis of some classes of unsteady flow problems.[4,46] However, it has been shown that adjoint approaches compute very large sensitivities when applied to chaotic dynamical systems such as those arising from scale resolving simulations. Such behaviour is not only driven by turbulence itself, but it is also observed in two dimensional flow simulations. The reason for this resides in the very high sensitivity with respect to the initial conditions in the context of chaotic dynamical systems,[20,47,48] leading to sensitivity parameters and error estimates which are not anymore effective. Additionally, backward time integration increases considerably the computational costs of the the procedure, which then drastically increases the overall CPU time of the simulation.

To circumvent those problems, we propose to compute the unsteady adjoint equation by neglecting the unsteady term and following a steady-state approach, being interested in adapting for time-integrated statistically-steady quantities. The idea of using a steady-state method to target adaptation of quasi-stationary governing equations was already employed in the literature, see for example Braak *et al.*[49] This approach shows favorable results for mesh adaptation, *i.e.* when an accurate error estimator is not required but rather relative magnitudes are used for error localization. The adjoint in this case is obtained through the solution of the following linear system

$$\mathbf{R}^{\mathbf{\Psi}}(\widehat{\mathbf{W}}, \mathbf{\Psi}) = \left(\frac{\widehat{\partial \mathbf{R}}}{\partial \mathbf{W}}\right)^T \mathbf{\Psi} + \left(\frac{\widehat{\partial J}}{\partial \mathbf{W}}\right)^T = \mathbf{0}, \tag{5}$$

where J is a functional of the unsteady state vector $\mathbf{W}(t)$, and the residual Jacobian and the output linearization are evaluated using a pre-computed time-averaged solution $\widehat{\mathbf{W}}$. We remark that this approach reduces the computational cost as the steady-state adjoint is obtained through the solution of a single linear system. However, it does not provide exact sensitivities with respect to the output, since the average field $\widehat{\mathbf{W}}$ is not a solution of the space-time problem, $\overline{\mathbf{R}}(\mathbf{W}(t), t) = \mathbf{0}$.

B. Entropy-adjoint sensitivity

The entropy-adjoint approach can be derived from the unsteady adjoint differential equation by choosing an output functional such that

$$J = \int_{\partial\Omega} \mathbf{f}_e \cdot \mathbf{n} d\sigma + \int_{\Omega} \nabla \mathbf{v}^T \cdot \mathbf{F}_v d\Omega - \int_{\partial\Omega} \mathbf{v}^T \mathbf{F}_v \cdot \mathbf{n} d\sigma, \tag{6}$$

where $\mathbf{f}_e$ is the entropy flux associated with an entropy function $U(\mathbf{w}_c)$, and $\mathbf{w}_e$ are the entropy variables associated with U. The main idea behind this approach is to choose entropy variables that symmetrize inviscid and viscous terms of the compressible Navier–Stokes equations. This can be achieved by setting

$$U = -\frac{\rho S}{\gamma - 1}, \qquad S = c_p \ln p - c_v \ln \rho, \tag{7}$$

where S is the physical entropy, and c_p and c_v are the heat capacities at constant pressure and volume, respectively. The entropy variables are obtained by differentiating U with respect to the conservative state $\mathbf{w}_c$,

$$\mathbf{w}_e = U^T_{\mathbf{w}_c} = \left[\frac{\gamma}{\gamma - 1} - \frac{S}{R} - \frac{1}{2}\frac{\rho u_i u_i}{p}, \frac{\rho u_i}{p}, -\frac{\rho}{p} \right]^T \tag{8}$$

while the corresponding entropy flux is $f_{e,i} = u_i U$.

The three integrals appearing in Eq. (6) can be associated to the flow physics by noting that the first represents the convective outflow of entropy across the domain boundary, the second term is the generation of entropy due to the viscous dissipation, and the third is the entropy diffusion across the boundary. Since U has opposite sign from S, the first term measures the net convective inflow of physical entropy across the boundary, while the third term computes the net diffusive inflow of physical entropy into the domain. Since the second integral evaluates the generation of entropy inside the domain, the theoretical value of the functional J for an exact solution should be zero, meaning that the production of entropy is balanced from the entropy flux throughout the boundaries. Since the balance is not strictly enforced in a discrete sense, the output functional J targets areas of spurious entropy production by the numerical discretization.

The choice of such output functional and entropy variable is particularly attractive. First, it is worth pointing out that, under the assumption of limited entropy generation over the domain $\Delta S/R \ll 1$, where R is the gas constant coefficient, adapting using the objective function defined in Eq. (6) can be connected to a drag-adaptation, where the objective function is calculated integrating entropy as proposed by Oswatitsch.[50] Second, Fidkowski and Roe[28] demonstrated that the solution Ψ is obtained from the state variable directly, and

hence at extremely low computational cost. The residual's linearization is not required for its computation, meaning that the adjoint solution remains bounded for chaotic flow problems. Finally, the average adjoint value can be obtained directly from the average state, such that $\widehat{\mathbf{\Psi}} = \widehat{\mathbf{w}}_e$.

As demonstrated by Fidkowski *et al.*,[29] adapting using entropy variables can be considered equivalent to a farfield drag output adjoint for inviscid cases. For viscous cases, the equivalence of the error estimate can be recovered through an additional fine space residual evaluation involving inviscid terms, which is however neglected in the current study.

C. Error localization

In the results, we compare entropy-adjoint and output-adjoint adaptive indicators for a two-dimensional problem. These indicators take the following form for element K

$$\eta_K^{\psi} \equiv \left| \delta\widehat{\mathbf{\Psi}}_{K,h}^T \widehat{\mathbf{R}}_h(\mathbf{W}_h^H) \right|, \tag{9}$$

where $\delta\widehat{\mathbf{\Psi}}_{K,h}^T$ is the steady-state, fine-space entropy or output adjoint solution computed from the coarse-space, time-averaged primal solution injected into the fine-space. The δ indicates that the coarse-space projection of this output or entropy adjoint is removed prior to error estimation. $\widehat{\mathbf{R}}_h(\mathbf{W}_h^H)$ is the time-averaged unsteady residual, computed during the unsteady primal solution by injecting the coarse-space solution into the fine-space and calculating the residual. The residual is then averaged in time, and cancellations between residuals at different times are allowed. The purpose of using an averaged, unsteady residual is to obtain an accurate representation of the extent to which the fine-space unsteady equations are not satisfied using the coarse-space solution. A static sensitivity field computed from the steady-state entropy and output adjoint then provides the weight on this residual. The error indicator drives a fixed-fraction adaptive strategy, and a burn-time is used prior to each unsteady simulation to remove the transient phase from the average process. We remark that this strategy, which computes the error estimate on the average quantities, is preferred to compute the average of the instantaneous error estimate. In fact, the latter approach, even if more rigorous from a mathematical point of view, increases the computational cost per time step, as a fine space reconstruction of the solution needs to be computed at each iteration.

Equation (9) shows that a fine space adjoint solution is necessary to compute $\delta\widehat{\mathbf{\Psi}}_{K,h} = \widehat{\mathbf{\Psi}}_{K,h} - \widehat{\mathbf{\Psi}}_{K,H}$. In general, two methods are commonly used to this end:[31,51]

- the computation of a fine space adjoint $\widehat{\mathbf{\Psi}}_{K,h}$ on a finer space or using higher order polynomials, $\mathcal{V}_h \in \mathcal{V}_H$, and then subtracting $\widehat{\mathbf{\Psi}}_{K,H}$ obtained through standard Galerkin projection operators;

- the interpolation of the finer space solution through interpolation on a wider patch, also known as patch reconstruction, as $\widehat{\mathbf{\Psi}}_{K,h} = \mathbf{I}_H^h \widehat{\mathbf{\Psi}}_{K_p,h}$, where K_p is the union of the elements sharing a face with K.

In this work we solve the auxiliary problem on a finer space only for the two dimensional test cases. For the computationally intensive three dimensional flow problems, a time-averaged solution on a higher polynomial degree ($k_K + 1, \forall\, K \in \mathcal{T}_h$) is reconstructed element-by-element using information on the patch. The global vector of degrees of freedom of entropy variables is then computed on the reconstructed fine space.

Code MIGALE implements the simple procedure reported in the following. For any $K \in \mathcal{T}_h$, let K_p be a set of elements, a patch, consisting of the cell K itself and the nn elements K_n sharing a face with K. To locally reconstruct each component g of the solution vector $\mathbf{w}$, *e.g.*, pressure, to a higher polynomial degree $k_K + 1$, we simply L_2-project the numerical solution available at any cell $K_n \in K_p$ onto the local polynomial space $\mathbb{P}_d^{k_K+1}(K_p)$. As a basis for this space, we employ a continuous extension over K_p of the shape functions ϕ_i^K of K, directly defined on the mesh space according to Bassi *et al.*[38] The set of coefficients $\hat{\mathbf{G}}$ for the reconstructed solution $\hat{g}(\mathbf{x}) = \sum_{i=1}^{nv} \phi_i^K(\mathbf{x})\hat{G}_i$, where $nv = \mathrm{card}(\mathbb{P}_d^{k_K+1}(K_p))$, is then computed from

$$\int_{K_p} \phi_i^K \hat{g}(\mathbf{x})\, d\Omega = \int_K \phi_i^K\, g(\mathbf{x})\, d\Omega + \sum_{n=1}^{nn} \int_{K_n} \phi_i^K g(\mathbf{x})\, d\Omega, \quad \forall\, i = 1, \ldots, nv. \tag{10}$$

Although the set of basis functions ϕ_i^K is orthonormal in the physical space over the element K, we remark that the mass matrix arising from the LHS term in Eq. (10) is not an identity matrix and a local linear system needs to be solved for any of the $m = 5$ components of the solution vector $\mathbf{w}$. To maintain the solution's mean value fixed through the reconstruction process, the first coefficient of the reconstructed polynomial expansion is set to be equal to the first coefficient of the numerical solution over K.

IV. Load balancing approach

Any adaptation algorithm applied to parallel simulations generally induces a strong imbalance of the load per partition and a drastic reduction of the parallel efficiency. In such cases, an effective re-decomposition of the computational grid is mandatory to retain good computational efficiency. The proposed approach is based on the Metis[52] library. It exploits Metis's capabilities to optimize the partition of a weighted graph (graph vertices correspond to the mesh elements), enforcing multiple constraints, and to minimize the number of faces of the partition boundaries.

The multi-constrained mesh partitioning approach has a significant impact on the parallel efficiency of simulations, as the floating-point operation count in different phases of the solution scales differently with the polynomial degree. For instance, the Jacobian matrix evaluation and factorization scales approximately as k^{3d} while the residual evaluation as k^{2d}. However, a naive partitioning approach which balances those two components is likely to not be optimal: the reason lies in the missing information about quadrature rules, which scale differently with the polynomial order according to the volume and face integrals. It turns out that taking into account additional and simultaneous balancing of the operation count related to the volume and surface integrals evaluation improves the parallel efficiency. For the three dimensional cases on hybrid meshes, a good balance of the computational cost related to the different solution phases was achieved by using five constraints. The first four constraints aim at balancing the floating point operations of residual evaluation and Jacobian assembly on different element and face types, which require different numbers of Gauss integration points for a given polynomial order. The last constraint balances the computational cost of matrix-vector products. The weights w_i used to enforce these constraints are reported in Appendix A.

The parallel efficiency of the p-adaptation strategy has been assessed on two different 3D test cases to demonstrate the effectiveness of the load balancing strategy. In the first test case, a grid with 768 elements was used to perform the simulations on a small Linux cluster consisting of four AMD Opteron 6276 sixteen core CPUs. The solutions were advanced in time using the backward-Euler scheme, which requires the solution of one nonlinear system per time step. The solution of the linear system required in the Newton-Raphson approach was obtained using a block-Jacobi preconditioned GMRES iterative solver. Figure 1 compares how the DoFs per processor are distributed across the MPI processes without load balancing (left) and with the multi-constraint algorithm (right). As expected, the result of the proposed optimal load balancing procedure is not a purely uniform distribution of DoFs.

Figure 2 (left) shows the strong scalability for the AMD Opteron 6276 CPUs, up to 64 cores. The curves labelled $k = 1$ and $k = 4$ refer to the integration of solutions for 10 time steps using first- and fourth-degree polynomials, and serve as reference. For a mesh with a small number of elements, the lowest-order computations scale poorly (see the $k = 1$ curve), as expected. The p-adaptive solutions were advanced in time for 100 time steps by using the backward-Euler scheme, starting from a $k = 1$ approximation and performing 6 adaptation cycles (each time adapting 20% of the elements marked for adaptation), every 10 time steps and setting to $k = 4$ the maximum polynomial degree. In the last 40 time steps no further p-adaptation was performed. The scalability of the p-adaptive solver, running on the unweighted partition of the grid and without load balancing (Adp), is clearly quite poor.

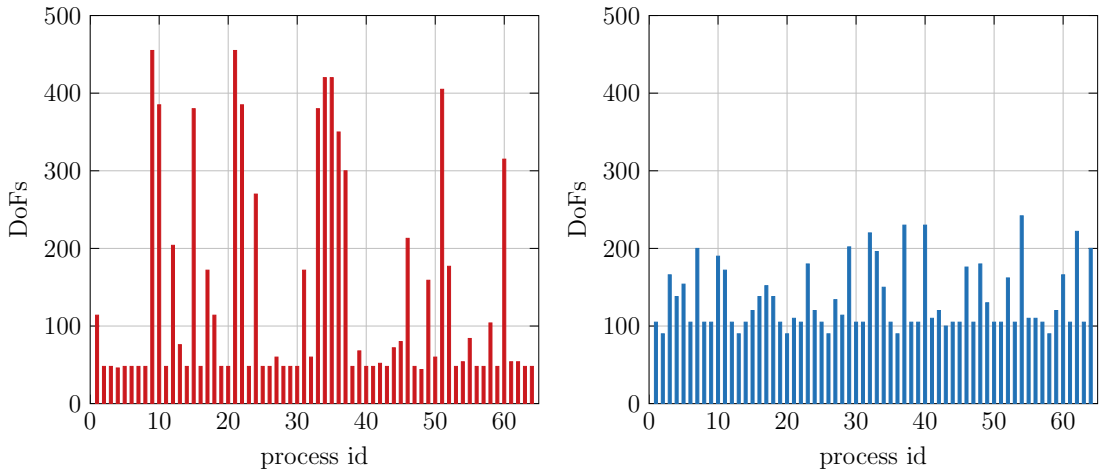


Figure 1: Load of the processors without (left) and with (right) the use of the load balancing approach after six p-refinements, AMD Opteron 6276 (sixteen core) CPUs, up to 64 cores

Activating the load balancing ($LB - Adp$), the scalability improves and the curve gets closer to the uniform $k = 4$ polynomial degree case computed on the unweighted partitioned grid, even if the final number of DoFs is more than three times smaller for the adaptive computation. The graph in this case was obtained using the multi-constraint partitioning strategy. It is worth noting that the parallel efficiency of the p-adaptive computation, evaluated over the last 40 time-steps, when the solution is already adapted, is even higher, as shown by the $LB - Adp(40)$ curve.

In the second test case we evaluate the strong scalability of the p-adaptive solver performing ILES on the circular cylinder testcase described in Sec. V. The grid consists of 38 320 elements and the time integration scheme was the third-order ROS3P scheme of Lang and Verwer.[43] The starting point of our investigation was a solution already adapted and characterized by a polynomial degree distribution $k \leq 6$. The parallel performance is evaluated over 10 time steps without performing any further adaptation. For this problem we used up to 32 Intel KNL nodes (up to 2178 cores) of the MARCONI A2 HPC system and the results are shown in Fig. 2 (right). In all the p-adaptive computations, we use the best single node performance as reference, obtained using a weighted $LB - Adp$ algorithm on 68 cores. The multi-constraint partitioning algorithm guarantees a behaviour similar to that achieved by the $k = 3$ fixed polynomial order computation which involves almost the same total number of DoFs. On the other hand, it is worth noting that a naive algorithm balancing only the number of DoFs of each partition (labelled $LB - Adp(DoFs)$) scales very poorly, together with the unbalanced Adp algorithm.

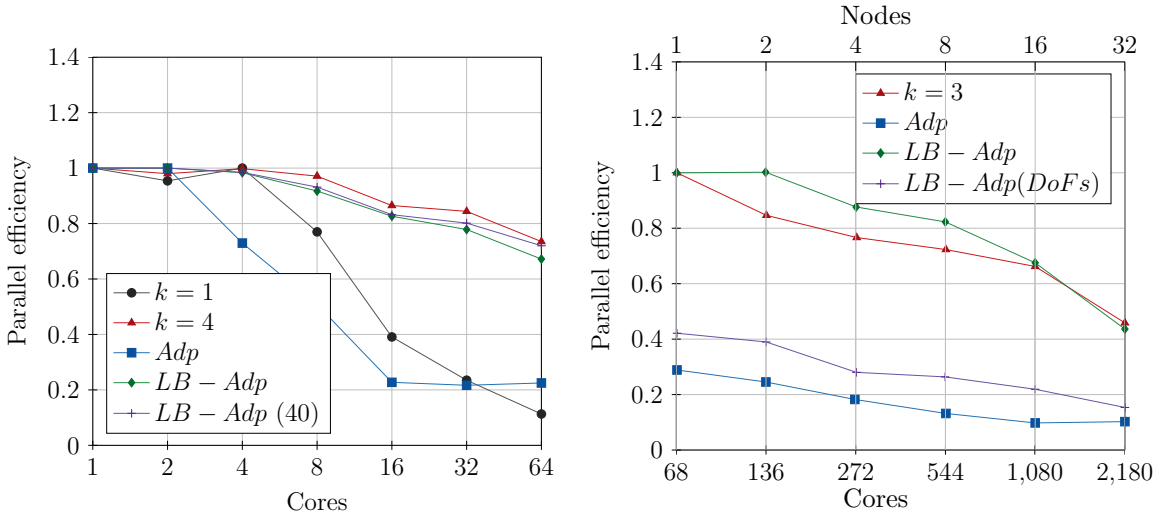


Figure 2: Parallel efficiency of the solver on AMD Opteron 6276 CPUs (left) and MARCONI A2 HPC system (right). $k = const$ stands for the fixed-p solver, Adp stands for the adaptive solution without employing the load balancing algorithm, $LB - Adp$ refers to the adaptive solution with multi-constraint load balancing, $LB - Adp(DoFs)$ refers to load balancing of only the DoFS, $LB - Adp(40)$ refers to parallel efficiency evaluated over the last 40 time-steps

V. Numerical results

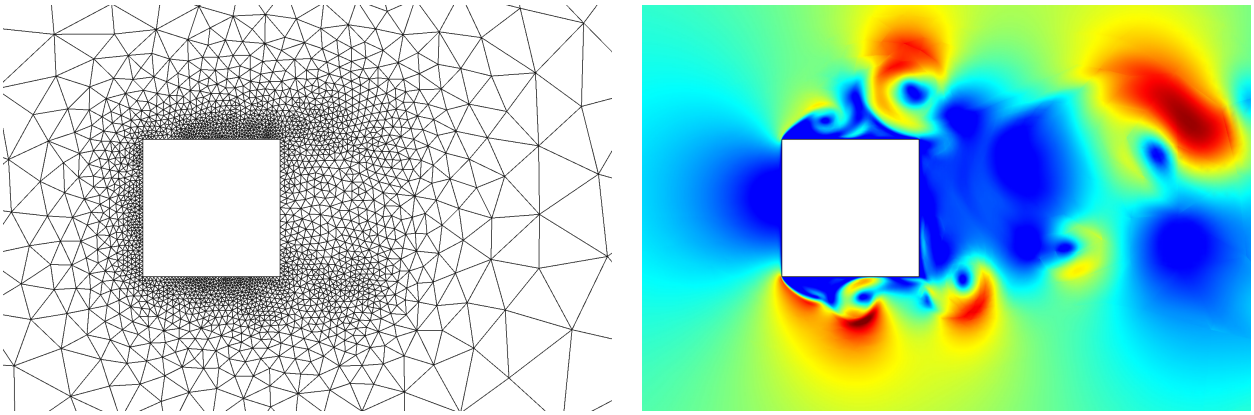
Numerical experiments are performed to assess the robustness, accuracy and efficiency of the proposed methodology in the context of unsteady, chaotic flow problems. First, we consider the two-dimensional flow around a square at $Re = 3\,000$, where the performances of the averaged entropy and output adjoint sensors are compared. Second, we validate the entropy-adjoint adaptive strategy on the implicit LES of the flow around a circular cylinder at $Re = 3\,900$. Third, as proof of concept, we report results of a higher-Reynolds number test case, the implicit Large Eddy Simulation of the flow around a rectangular cylinder at $\mathrm{Re} = 22\,000$.

A. Two-dimensional square test case

In this section, we compare the performance of the averaged entropy and output adjoint indicators for a two-dimensional simulation. The problem of interest is a unit square in horizontal flow at Mach number $M = 0.1$ and Reynolds number $Re = 3\,000$. The flow is initialized to free-stream and allowed to develop into a statistically-steady regime, as illustrated in Fig. 3, which also shows the mesh used for all of the runs with 4454 elements. The output of interest is the time-averaged drag coefficient on the square, defined as

$$J = \int_T \int_{\partial\Omega_w} (-p\mathbf{n} + \mathbf{T} \cdot \mathbf{n}) \cdot \mathbf{x} d\sigma dt, \tag{11}$$

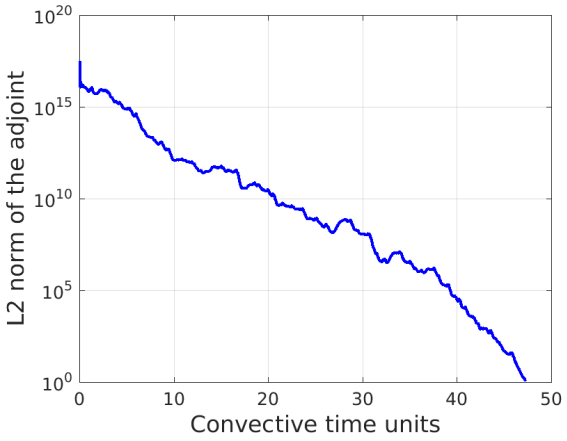
where $\mathbf{T}$ is the stress tensor, $\mathbf{n}$ and $\mathbf{x}$ are the wall-normal and free-stream directions, respectively. We remark that standard unsteady adjoint solutions for this output quickly grows without bound and becomes impossible to use for adaptation, as shown in Fig. 4.



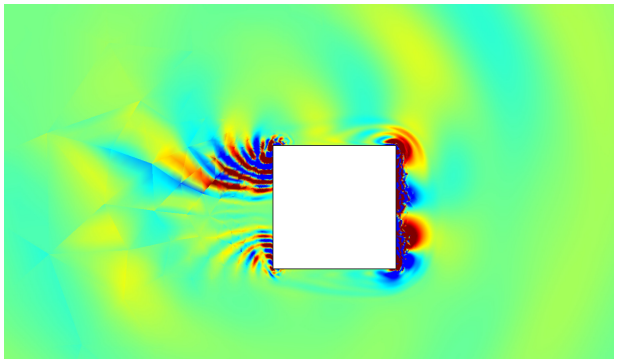
(a) Mesh

(b) Mach contours (0, 0.2)

Figure 3: Flow around a square - Mesh and instantaneous Mach contours



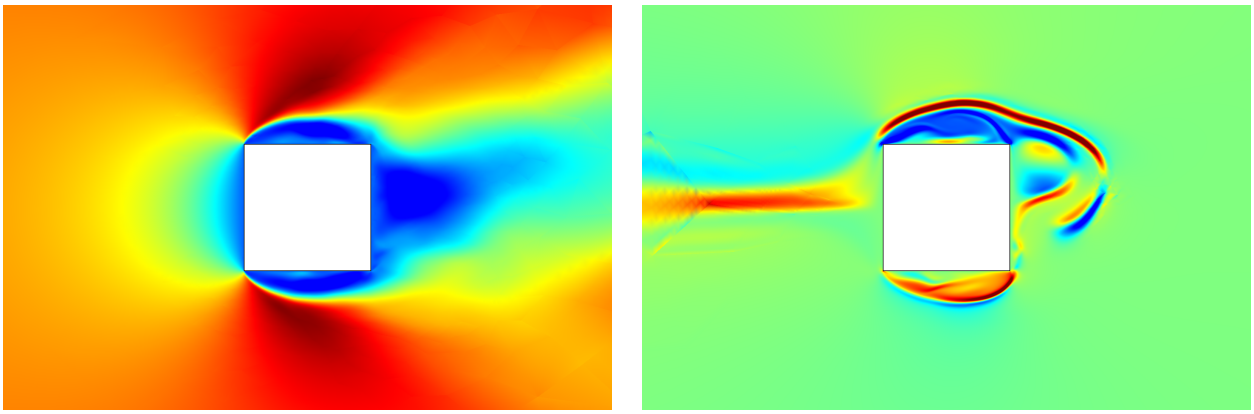
(a) Adjoint norm time history



(b) Conservation of mass component of the adjoint at $t = 0$. Scale is $O(10^{17})$.

Figure 4: Flow around a square - Growth of the adjoint for the time-averaged drag output in reverse time

Order-adaptive simulations are performed starting from $p = 1$ on every element. At each adaptive iteration, 15% of the elements with the highest error indicator have their order incremented by 1. A burn time of 10 convective time units (here CTU, where 1 CTU is the time taken by the flow at free-stream speed to traverse the length of the square) is used prior to the averaging for each unsteady run. The averaging time is also 10 CTU. A sufficiently small time-step (.05 CTU) and high-order time-stepping scheme (ESDIRK5) are used to minimize temporal errors, so that the dominant source of error is spatial. The temporal



(a) Averaged state (x-momentum) (b) Steady adjoint (conservation of x-momentum)

Figure 5: Flow around a square - Averaged state and the steady-state adjoint computed about this state

discretization is therefore not adapted.

Figure 5 shows the time-averaged state and corresponding steady-state adjoint solution from the last adjoint-based adaptive simulation. Note the lack of symmetry, particularly in the adjoint solution, which can be addressed to some extent by increasing the averaging time.

Five adaptive iterations were run using three indicators: the entropy-adjoint weighted residual, the output-adjoint weighted residual, and an unweighted residual. The latter was computed by using the L_1 norm of the time-averaged residual on each element as the indicator, without any adjoint weight. The other adaptive indicators also use the time-averaged residual, but with entropy or output adjoint weights. Figures 6 and 7 show the order distributions on the final adapted meshes for each indicator. We see that a common area to refine includes the corners of the square, particularly in the front and extending along the dominant flow direction towards the rear of the square, where a shear layer is present. The entropy-adjoint indicator additionally targets elements behind the square, whereas the output-adjoint indicator targets elements ahead of the square. This difference is caused by the opposite orientation of the primal versus adjoint wake – the adjoint wake extends ahead of the square.

Once the adapted order fields were obtained, they were tested in long-time simulations. In each such simulation, a burn time of about 100 CTU was first run starting with the final solution from the adaptation sequence. Then, an averaging time of about 300 CTU was simulated to compute the average drag coefficient.

Finally, Fig. 8 compares the average drag coefficient results for all of the adaptive simulations and plots them versus the spatial degrees of freedom of the order field. In this plot the convergence of uniform order refinement is also included. Note that this plot shows the drag

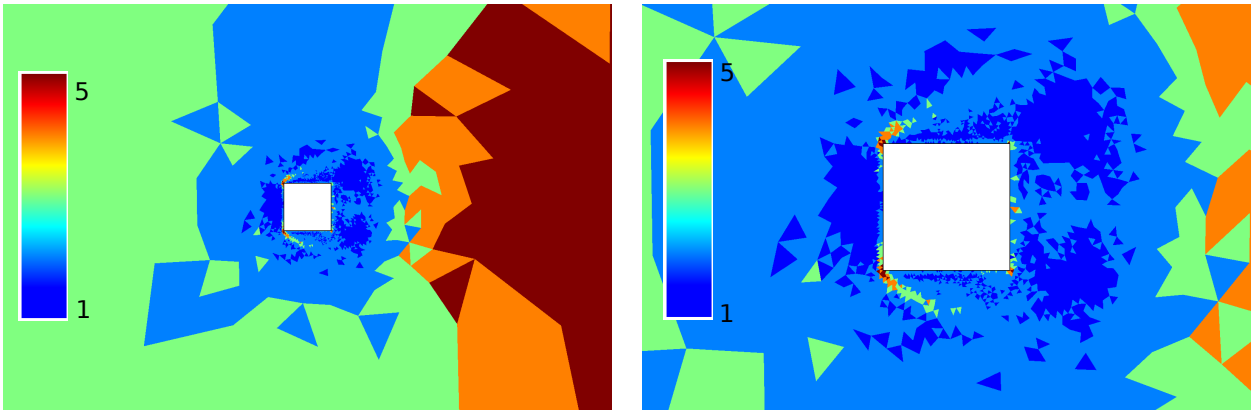


Figure 6: Flow around a square - Polynomial degree distribution (1–5) on the final adapted meshes with the entropy adjoint

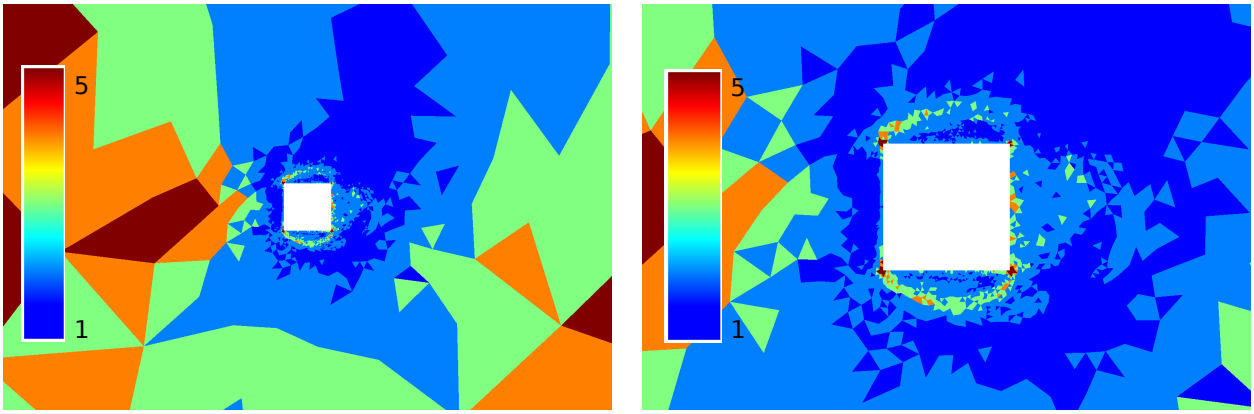


Figure 7: Flow around a square - Polynomial degree distribution (1–5) on the final adapted meshes with the drag adjoint

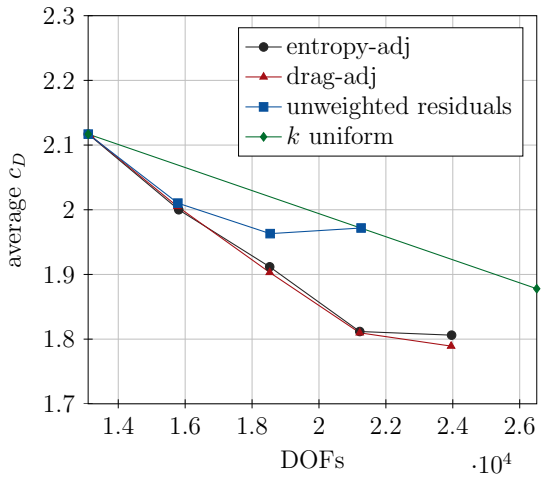


Figure 8: Flow around a square - Convergence of the average drag coefficient output on adapted and uniformly-refined order fields

coefficient itself, not the error. We see that both the entropy and output adjoint adaptations yield order fields that quickly converge to an average drag coefficient in the vicinity of what appears to be the true value (which is lower than the starting point). The unweighted residual, on the other hand, does not converge well, and this is due to the lack of a weight that cuts off adaptation far away from the square: many of those areas still have high residuals due to large elements and passing transient/acoustic waves, even though their order has little impact on the output. These results suggest that for this class of simulations involving bluff bodies and separated flow regions, there exist a similarity between entropy and output adjoint indicators for statistically-steady, yet chaotic simulations. We note however that although the outputs are similar, the order fields differ in certain regions, suggesting that the subset of the given order refinements common to both fields is the principally-important region for refinement.

B. Three dimensional test cases

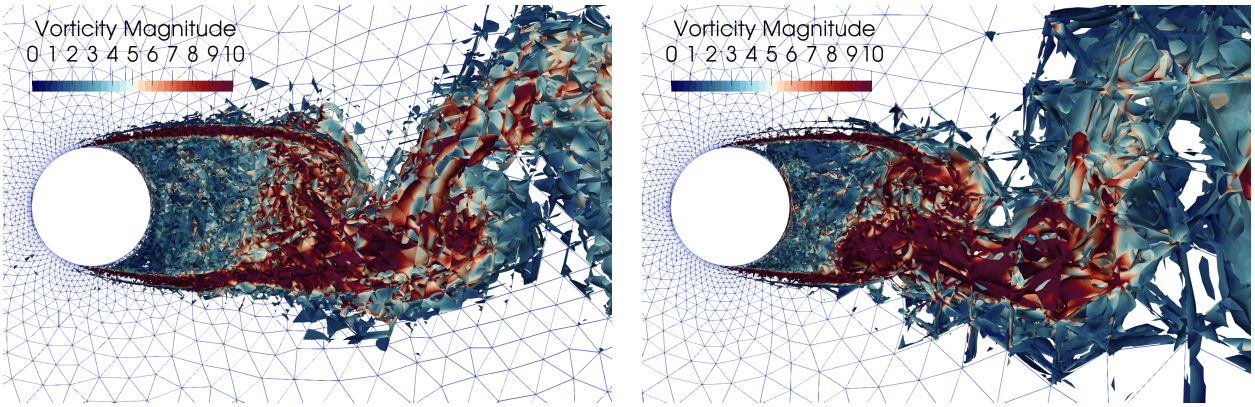
Here, we report the results obtained by computing the implicit LES of two test cases: *i*) the three-dimensional flow past a circular cylinder at $Re = 3\,900$ and Mach number $M = 0.1$, and *ii*) the turbulent flow over a square cylinder at $\alpha = 0°$, Reynolds number $\mathrm{Re} = 22\,000$ and Mach number $M = 0.1$. Span-wise periodicity is in both cases assumed. The meshes used in this work have been generated with a 2D high-order version of a fully-automated hybrid mesh generator based on the advancing-Delaunay strategy[53] and extruded in the span-wise direction.

In both computations, coarsening was disabled, *i.e.*, $\mathcal{G}_c = 0$, and $\mathcal{G}_r$ was set to 0.2 to refine the 20% of elements with the highest estimated error. Both simulations were performed using 15 KNL nodes on the MARCONI A2 HPC system at CINECA.

1. Turbulent flow around a circular cylinder

The transitional turbulent flow around a circular cylinder at $Re_D = 3\,900$, where D is the diameter of the cylinder, has been analyzed in several papers and was also part of the test case suite of the International Workshop on High-Order Methods.[54] In this section we report adaptive DG computations for this flow problem on two grids made of 67 466 (mesh A) and 44 856 (mesh B) elements, respectively, with quadratic edges near the solid walls. The grids were generated by extruding two-dimensional meshes with a circular ($50D$) farfield boundary for a $2D$ length along the direction perpendicular to the plane. Both meshes enforce span-wise periodicity and discretize the span-wise direction with 14 cells. Note that Mesh A includes a refinement in the wake region with respect to Mesh B, see Figure 10(a). Details of the computational grids are shown in Figure 9 together with the isosurfaces of the Q-criterion

coloured with the vorticity magnitude.

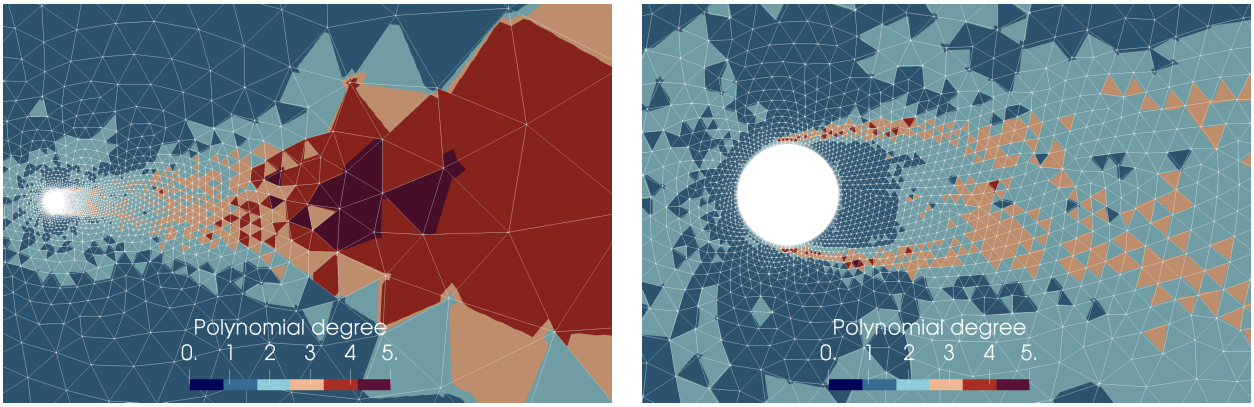


(a) mesh A (b) mesh B

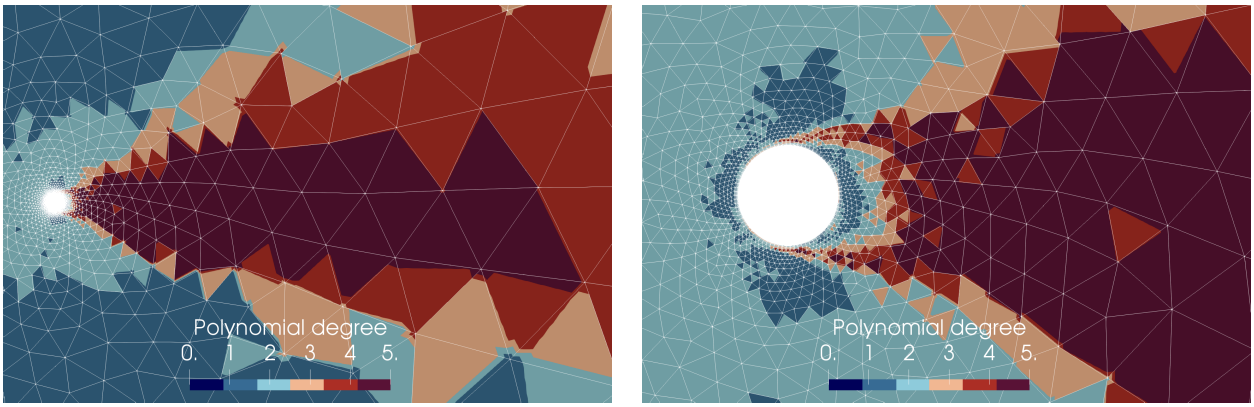
Figure 9: Flow around a circular cylinder - Mesh details and instantaneous isosurfaces of the Q-criterium coloured with the vorticity magnitude

The adaptation process has been performed 6 times for both simulations. The computations have been initialized from freestream values with a piece-wise constant polynomial representation of the solution. During the first adaptive cycle all elements were refined from DG-$\mathbb{P}^0$ to DG-$\mathbb{P}^1$. The maximum polynomial degree allowed to the process was set to DG-$\mathbb{P}^5$. The polynomial degree distribution resulting from the sixth adaptation cycle was finally used to compute an average solution over 100 shedding cycles. Adaptation led to an overall number of DoFs of 6.61×10^5 for mesh A and 5.56×10^5 for mesh B, corresponding to roughly one order of magnitude lower than other LES computations.[55] Similarly to what happens in the two-dimensional case of Sec. A, the approach based on the averaged entropy variables and the averaged residual vector increases the spatial resolution at the shear layer and in the wake region, as shown in Figures 10 and 11. The polynomial degree distributions over the two grids, *i.e.*, Mesh A and Mesh B, are significantly different due to the different mesh densities. In Mesh B (Figs. 10(c) and 10(d)) the algorithm almost uniformly places high-order polynomials elements along the wake to compensate for the original lack of spatial resolution due to the use of very large cells downstream the body. Over Mesh A (Figs. 10(a) and 10(b)) the error indicator is more selective, and is still able to localize high-order elements in the wake region and in the shear layer. On both grids, the spanwise direction refinement is mainly driven by the streamwise resolution.

Figure 12 shows the span-wise and time-averaged pressure coefficient c_p and non-dimensional wall vorticity $\Omega/2Re^{0.5}$ distributions on the cylinder. The c_p is in good agreement with the experimental measurements of Norberg[56] and two numerical simulations published in the literature: an LES from Lysenko et al.[55] with k-eq. and Smagorinsky subgrid scale model, and a DNS from Ma et al.[57] Although the vorticity profile is quite different in the first

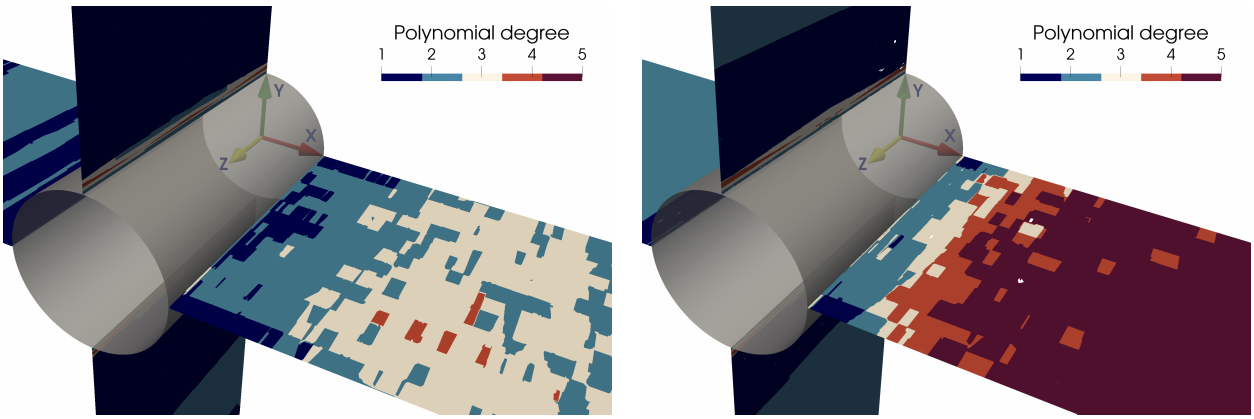


(a) mesh A (b) mesh A (zoom)



(c) mesh B (d) mesh B (zoom)

Figure 10: Flow around a circular cylinder - Polynomial degree distribution on the midspan plane perpendicular to the cylinder axis



(a) mesh A (b) mesh B

Figure 11: Flow around a circular cylinder - Polynomial degree distribution on the planes passing trough the origin (center of the circular cylinder) and perpendicular to the x- and y-axis

half of the cylinder when compared to the experimental data from Son and Hanratty,[58] the distribution is almost superimposed with the DNS from Ma et al.[57]

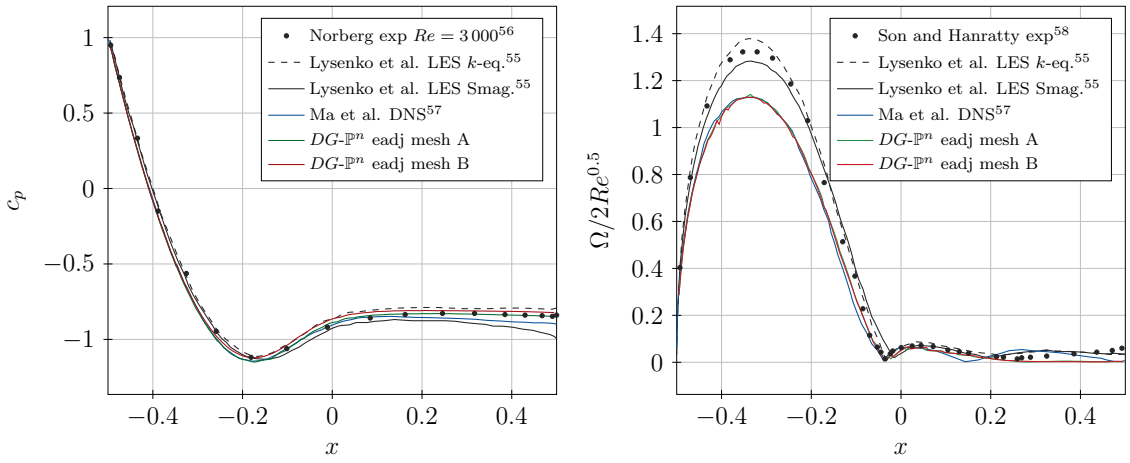


Figure 12: Flow around a circular cylinder - Span-wise and time-averaged pressure coefficients c_p (left) and non-dimensional wall vorticity $\Omega/2Re^{0.5}$ (right) along the cylinder

Figure 13 shows the span-wise and time-averaged stream-wise u_x/u_{ref} and cross-wise u_y/u_{ref} velocity profiles at different locations in the near wake of the cylinder $x/D = \{1.06, 1.54, 2.02\}$. The computations are compared with experimental data of Parnaudeau et al.,[59] DNS data of Wissink and Rodi,[60] and LES from Lysenko et al.[55] with k-equation and Smagorinsky subgrid scale models (SGS). For both the meshes the velocity profiles u_x/u_{ref} are in good agreement with experimental data, DNS and LES k-equation/Smagorinsky results. The profiles for u_y/u_{ref} compare very well with LES k-equation/Smagorinsky results but show some discrepancy with respect to experiments at $x/D = 1.54$, where the shape of the profile is similar but the magnitude is slightly different.

Figure 14 extends the comparison to the span-wise and time-averaged stream-wise $u'_x u'_x/u^2_{ref}$ and cross-wise $u'_y u'_y/u^2_{ref}$ velocity fluctuations. The computations are in good agreement with the experimental data from Parnaudeau for both distributions, although some discrepancies with respect the Lysenko results, regardless of the adopted SGS model, are observed.

Figure 15 shows the comparison of stream- and cross-wise velocities, and velocity fluctuations distributions in the wake up to $x/D = 10$. We compare the results to the DNS from Wissink and Rodi[60] at the following locations: $x/D = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10\}$. An almost perfect agreement with the reference DNS is observed, where only small differences between the profiles obtained on mesh A and B can be seen for the velocity fluctuations at $x/D = \{1, 2\}$, due to a slightly different local spatial resolution. In general, the results on the two grids are very similar, and possibly suggest the capability of the present adaptation strategy to deliver almost "mesh independent" average solutions.

We finally report the assessment of the computational performance gain due to the use of

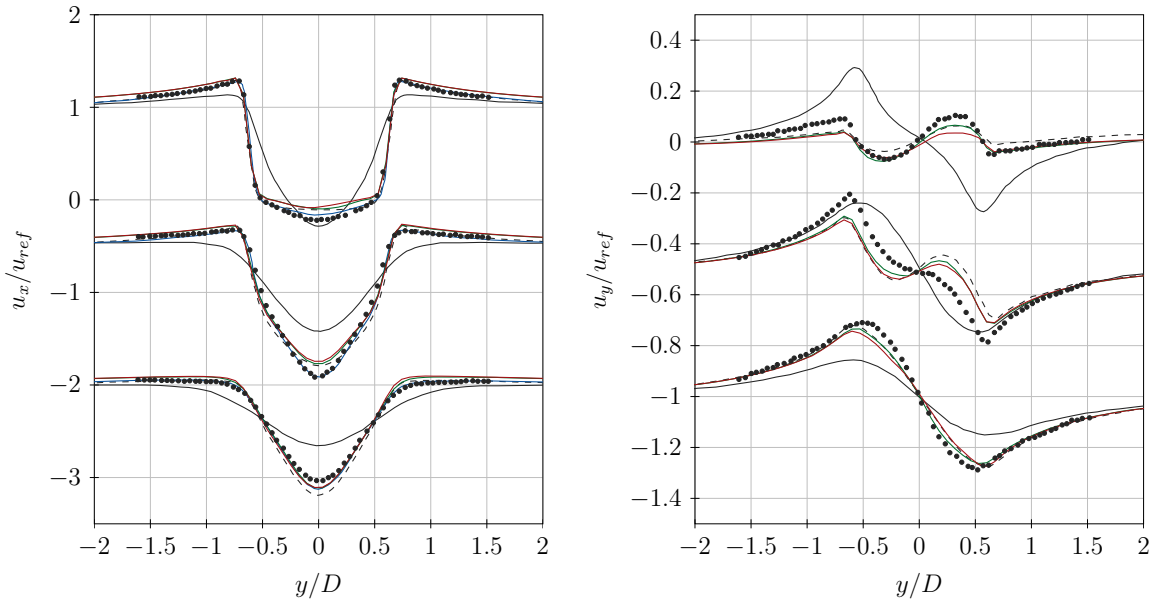


Figure 13: Flow around a circular cylinder - Span-wise and time-averaged stream-wise u_x/u_{ref} and cross-wise u_y/u_{ref} velocity at different locations in the wake of the cylinder $x/D = \{1.06, 1.54, 2.02\}$. • Parnaudeau et al. exp,[59] - - - Lysenko et al. LES k-eq.,[55] —— Lysenko et al. LES Smag.,[55] —— Wissink and Rodi DNS,[60] —— *DG*-$\mathbb{P}^n$ eadj mesh A, —— *DG*-$\mathbb{P}^n$ eadj mesh B

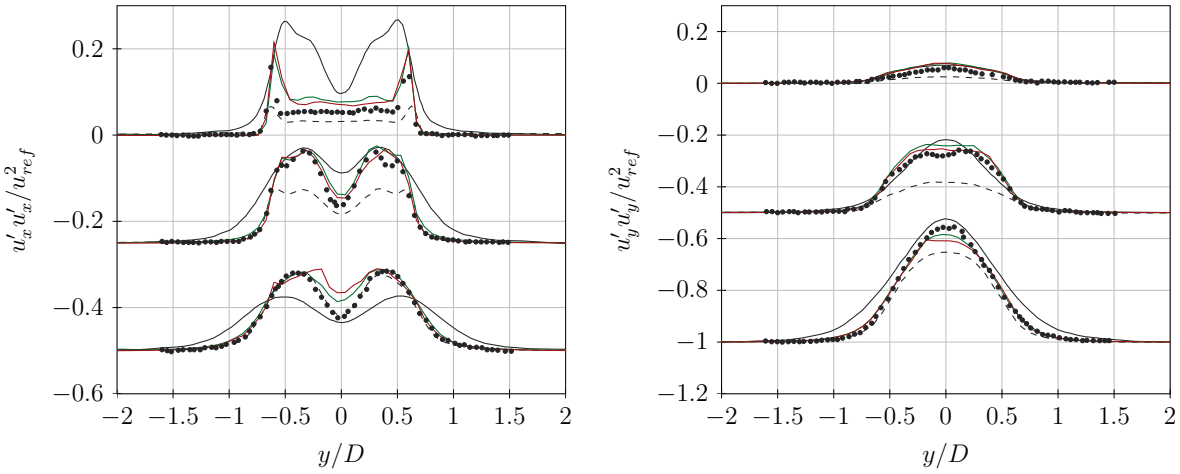


Figure 14: Flow around a circular cylinder - Span-wise and time-averaged stream-wise $u'_x u'_x/u^2_{ref}$ and cross-wise $u'_y u'_y/u^2_{ref}$ velocity fluctuations at different locations in the wake of the cylinder $x/D = \{1.06, 1.54, 2.02\}$. • Parnaudeau et al. exp,[59] - - - Lysenko et al. LES k-eq.,[55] —— Lysenko et al. LES Smag.,[55] —— *DG*-$\mathbb{P}^n$ eadj mesh A, —— *DG*-$\mathbb{P}^n$ eadj mesh B

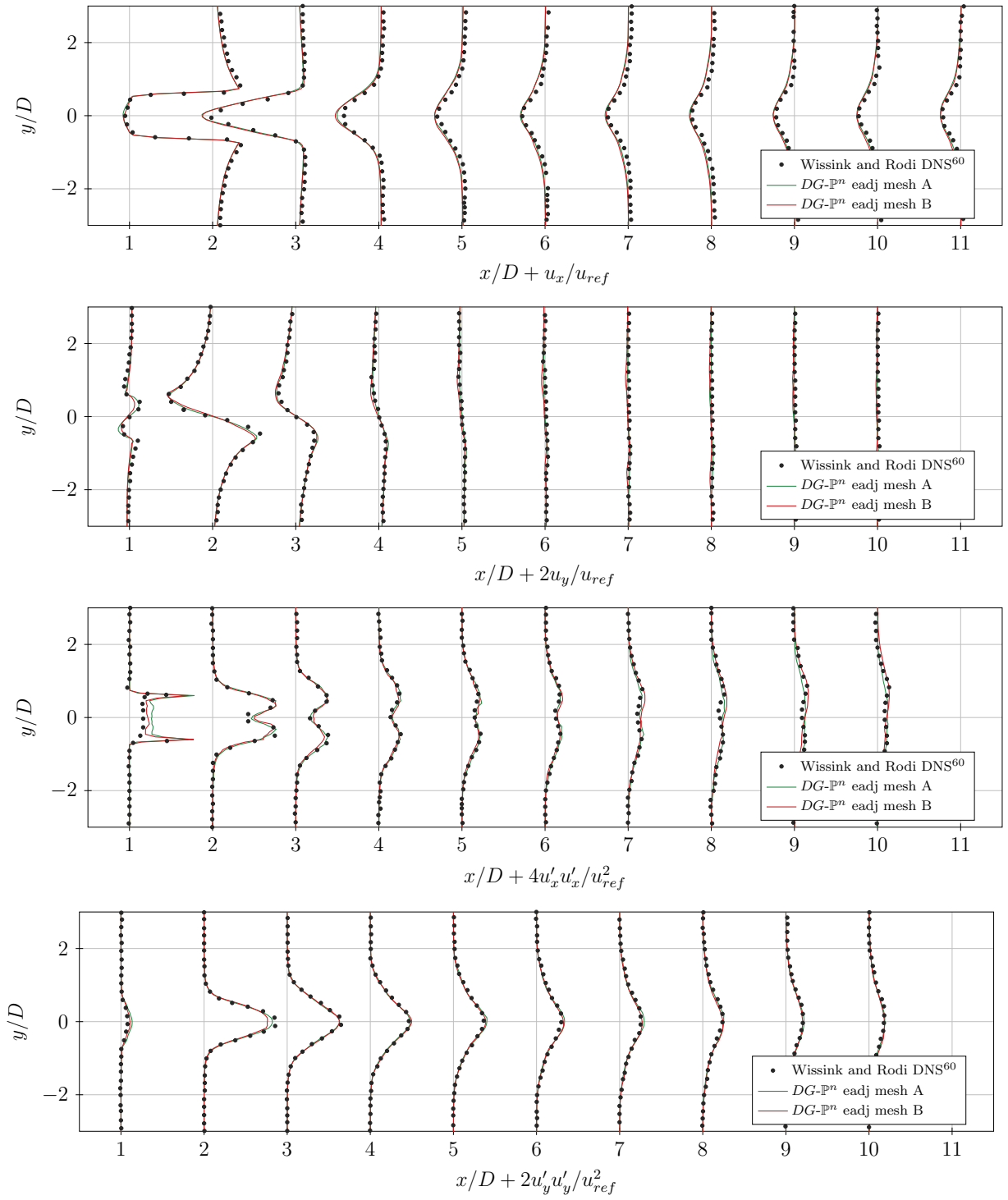


Figure 15: Flow around a circular cylinder - Span-wise and time-averaged stream-wise and cross-wise velocity, u_x/u_{ref} and u_y/u_{ref}, and stream-wise and cross-wise velocity fluctuations, $u'_x u'_x/u^2_{ref}$ and $u'_y u'_y/u^2_{ref}$, at different locations in the wake of the cylinder $x/D = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10\}$

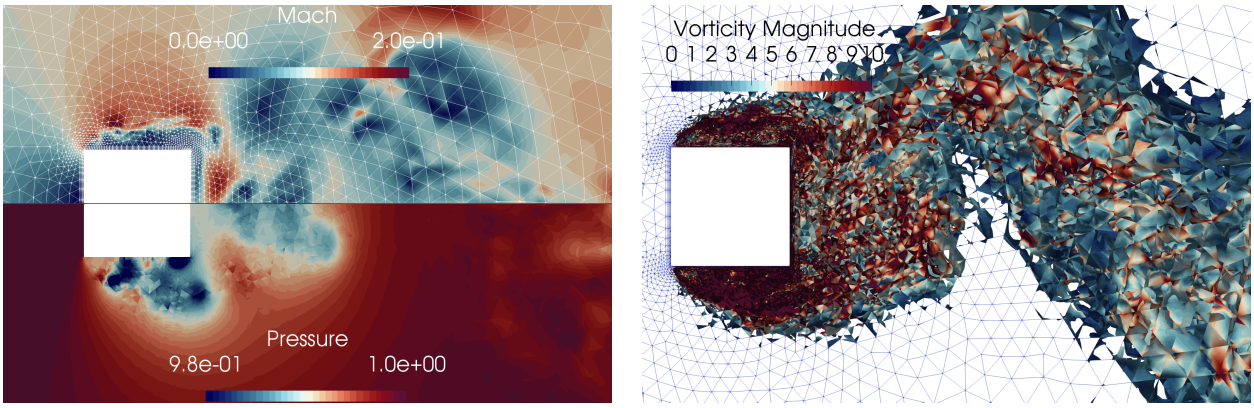
Table 1: Flow around a circular cylinder (Mesh B, 10 time steps) - Impact on the computational performance of the p-adaptive computation $\mathbb{P}^n$ (5.56×10^5 DoFs) when using ($aDOE$) or not ($fDOE$) the automatic algorithm for the choice of the quadrature rules degree of exactness (see Sec. B).

	[%]		$100 \cdot \frac{CPU^{time}_{aDOE}}{CPU^{time}_{fDOE}}$
	$aDOE$	$fDOE$	
residuals assembly	4.9	5.9	12.4
matrices assembly	76.1	90.7	12.6
GMRES solution	19.0	3.4	84.5
execution	100.0	100.0	15.0

the adaptive algorithm for the choice of degree of exactness of the quadrature rules. Table 1 shows, in terms of percentage ("% execution"), how the overall execution time is split over the different computational tasks, *i.e.*, residual and matrix assembly and solution time. The CPU time ratio between a 10 time steps simulation that uses ($aDOE$) or not ($fDOE$) the DOE adaptation technique is reported in the same table. The tolerance value for the reduction process is $tol_q = 10^{-10}$, see Sec. B. The computational gain in using the adaptive DOE algorithm is clear and is measured by a reduction of the overall CPU time of 85%. As expected the main saving is related to the assembly of the implicit operator.

2. Turbulent flow around a square cylinder

The performance of the averaged entropy adjoint approach as an adaptive indicator was also assessed by computing the flow past a square cylinder for the relatively high Reynolds number $\mathrm{Re_D} = 22\,000$, where D is the edge of the square, an angle of attack $\alpha = 0°$, and a Mach number $M = 0.1$. An hybrid 3D mesh consisting of 83 016 elements with linear edges was used. The farfield is rectangular of extension $[-10.5D, 20D]\times[-21D, 21D]$. The mesh was extruded in the span-wise direction of $4D$ using 12 elements and enforcing span-wise periodicity. Details of the computational mesh are shown in Figures 16(a) and 16(b) together with the instantaneous pressure and Mach number contours and the isosurfaces of the Q-criterion coloured with the vorticity magnitude, respectively. During the simulation, 5 polynomial adaptations were performed. The computation has been initialized from freestream values with a piece-wise constant polynomial representation of the solution. During the first adaptive cycle all the elements were refined from DG-$\mathbb{P}^0$ to DG-$\mathbb{P}^1$. The maximum polynomial degree allowed to the algorithm was set to DG-$\mathbb{P}^3$. The final number of DoFs obtained from the adaptation process and used to compute the average solution over 20 shedding cycles is 7.34×10^5 and roughly 20% of the elements are $\mathbb{P}^3$. The polynomial degree distribution over the domain is shown in Fig. 17. We remark that high-order polynomials are used in the shear



(a) (b)

Figure 16: Flow around a square cylinder - Instantaneous Mach number and pressure contours (a), and isosurfaces of the Q-criterion coloured with the vorticity magnitude contours (b)

layer and in the wake, which is shifted upstream with respect to the circular cylinder test case due to the higher Reynolds number. In the spanwise direction, as observed for the circular cylinder test case, the refinement is driven by the resolution in the streamwise direction. To assess the accuracy of the solution delivered by the proposed adaptation algorithm ($\mathbb{P}^n$), we compare the results to those of a uniform DG-$\mathbb{P}^2$ discretization resulting in a comparable number of DoFs, *i.e.*, 8.30×10^5. Figure 18 shows the span-wise and time-averaged pressure

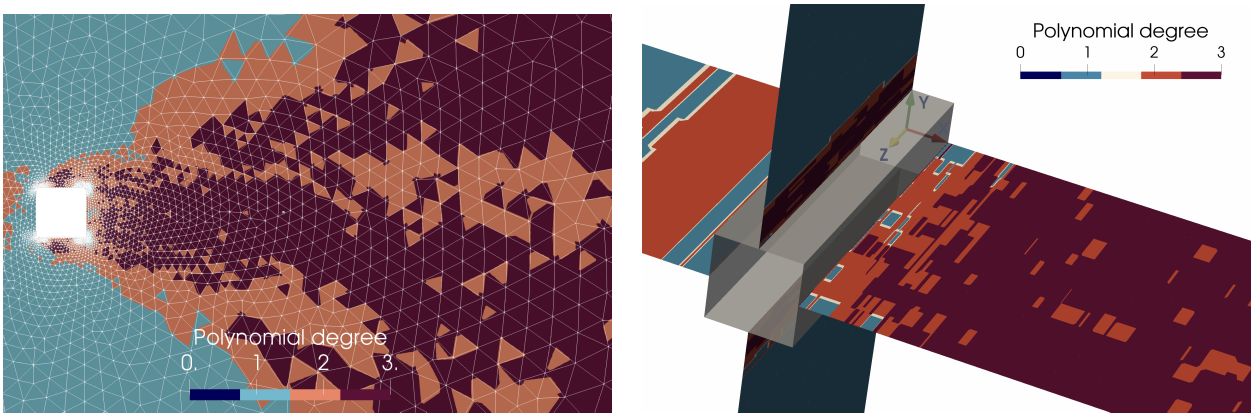


Figure 17: Flow around a square cylinder - Polynomial degree distribution in the plane perpendicular to the cylinder axis and on the planes passing for the origin (center of the square cylinder) and perpendicular to the x- and y-axis

coefficient c_p distribution on the body. The result compares favourably, except for the rear part, with experimental data from Bearman and Obasaju[61] and the DNS from Trias et al.[62] The discrepancy on the downstream side, as reported by Sohankar et al.[63] for a LES with dynamic Smagorinsky, can be ascribed to a different predicted stream-wise velocity distri-

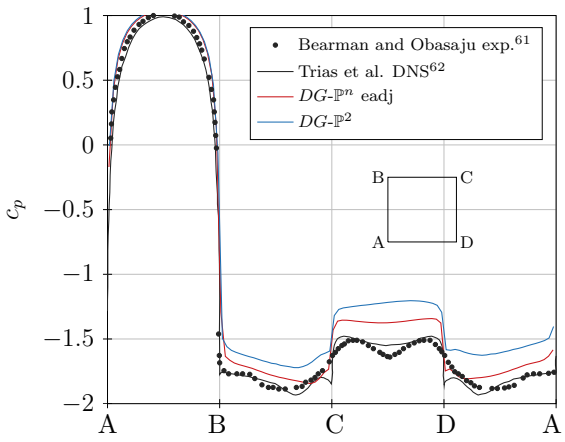


Figure 18: Flow around a square cylinder - Span-wise and time-averaged pressure coefficient c_p along the cylinder

bution in the near wake along the centerline, as shown in Fig. 20. The comparison between adapted DG-$\mathbb{P}^n$ and uniform DG-$\mathbb{P}^2$ solution demonstrates a better accuracy provided by the adaptation algorithm.

Figure 19 show the span-wise and time-averaged velocity profiles, u_x/u_{ref} and u_y/u_{ref}, on the cylinder and in the near wake at different locations x/D. The agreement with the DNS from Trias[62] and with the experimental measurements of Lyn[64] is in general good. However, some discrepancies are observed for the u_x/u_{ref} profile at wall in proximity of the upstream corner. These differences can be ascribed to a zone near the corner where the adaptation algorithm places $\mathbb{P}^1$ cells, which are probably not enough in terms of spatial resolution. In the near wake, u_x/u_{ref} profiles for the adaptive DG solution are similar to the reference DNS, while experimental data shows slightly lower values approaching the centerline. u_y/u_{ref} profiles are almost coincident with DNS data near wall, while some discrepancy is observed at $x/D = 1$, the first location in the near wake. In particular the profile has higher values near the downstream corner, probably due to a lack of local resolution. Figure 20 shows the stream-wise velocity u_x/u_{ref} along the centerline in the wake of the cylinder. For the adaptive simulation the peak location and its value is well predicted, if compared to DNS data,[62] while the distribution shows some discrepancy with both experimental[64] and DNS data. The velocity along the centerline reaches a plateau, which is slightly delayed, *i.e.* the length of rear recirculation bubble is higher, and overpredicted with respect to DNS and experiment. However the predicted distribution seems to be globally better than other LES results.[63] The uniform DG-$\mathbb{P}^2$ solution predicts a different peak position and value. Moreover, the distribution is far from the DNS profile. Figure 21 extends the comparison to the span-wise and time-averaged velocity fluctuations, $u'_x u'_x/u^2_{ref}$, on the square body and in the near wake. The profiles from adaptive DG correctly capture the peak location,

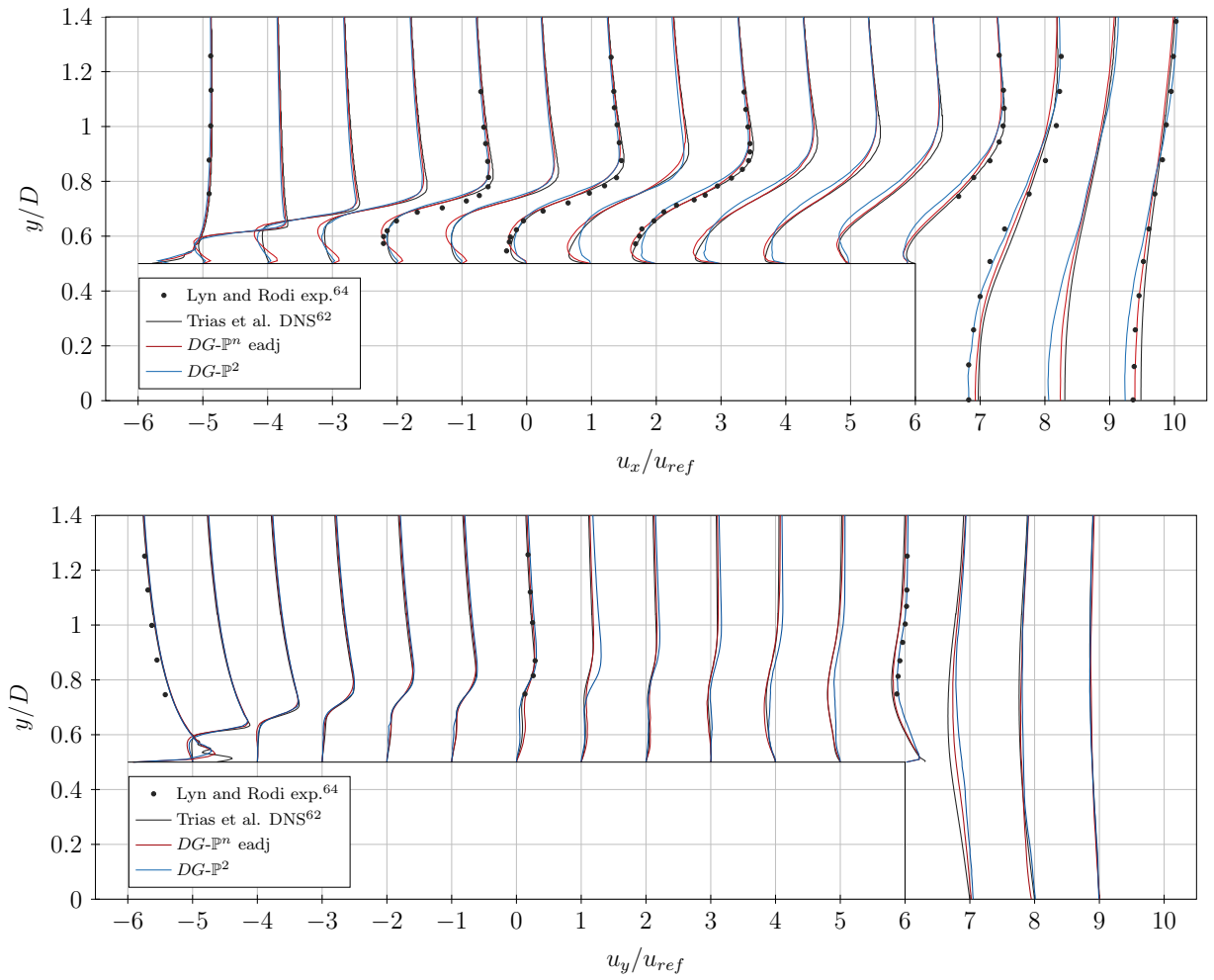


Figure 19: Flow around a square cylinder - Span-wise and time-averaged stream-wise u_x/u_{ref} and cross-wise u_y/u_{ref} velocity at different locations on the cylinder and in the wake $x/D = \{-0.5, -0.4, -0.3, -0.2, -0.125, -0.1, 0, 0.1, 0.125, 0.2, 0.3, 0.4, 0.5, 1, 1.5, 2\}$

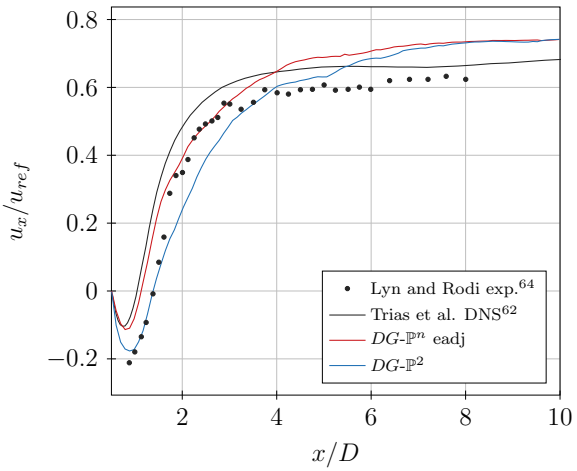


Figure 20: Flow around a square cylinder - Span-wise and time-averaged stream-wise velocity u_x/u_{ref} along the centerline ($y/D = 0$) in the wake of the cylinder

but overpredict the values of DNS[62] and experimental[64,65] data. However, a reasonable agreement with the LES from Minguez[65] is observed. In general, the comparison between the adapted DG-$\mathbb{P}^n$ and the uniform DG-$\mathbb{P}^2$ solutions reveal a better accuracy delivered by the adaptation algorithm.

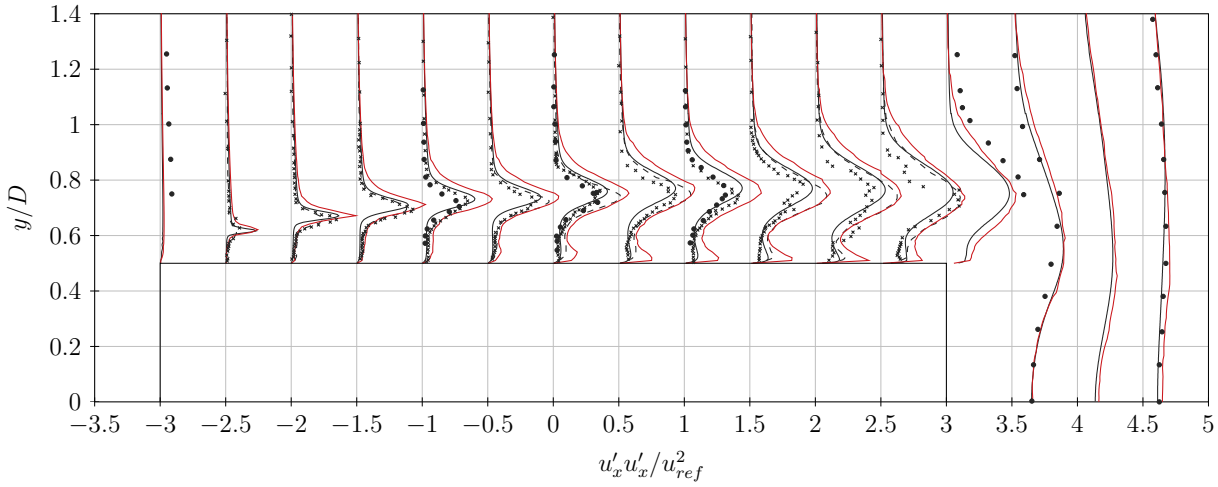


Figure 21: Flow around a square cylinder - Span-wise and time-averaged stream-wise $u'_x u'_x/u^2_{ref}$ velocity fluctuations at different locations on the cylinder and in the wake $x/D = \{-0.5, -0.4, -0.3, -0.2, -0.125, -0.1, 0, 0.1, 0.125, 0.2, 0.3, 0.4, 0.5, 1, 1.5, 2\}$. • Lyn and Rodi exp.,[64] ⋆ Minguez et al. exp.,[65] —— Trias et al. DNS,[62] - - - Minguez et al. LES,[65] —— DG-$\mathbb{P}^n$ eadj.

Finally, in Table 2 the performance of the adaptive DG algorithm are compared with uniform $\mathbb{P}^2$ and $\mathbb{P}^3$ discretizations. For each of the main task of the computations, *i.e.*, residuals and matrices assembly and time spent in the iterative linear solver, the percentage

Table 2: Flow around a square cylinder (10 time steps) - Comparison of the performance of the solver: adaptive $\mathbb{P}^n$ (7.34×10^5 DoFs), uniform $\mathbb{P}^2$ (8.3×10^5 DoFs) and $\mathbb{P}^3$ (1.66×10^6 DoFs) solutions

	$\mathbb{P}^n$	$\mathbb{P}^2$		$\mathbb{P}^3$	
	execution [%]	execution [%]	$\mathbb{P}^n$ [%]	execution [%]	$\mathbb{P}^n$ [%]
residuals assembly	6.2	7.2	91.9	5.5	305.8
matrices assembly	64.7	60.3	74.2	60.0	322.8
GMRES solution	29.1	32.5	89.0	34.6	414.8
execution	100.0	100.0	79.6	100.0	350.0

CPU time with respect to the adaptive computation is shown (column $\mathbb{P}^n$ [%]). Results show that the less accurate $\mathbb{P}^2$ simulation is only 20% faster than the adaptive one, while the uniform $\mathbb{P}^3$ computation results in a computational effort which is almost four times larger. In this work, due to the high computational cost required by the simulation, we were not able to compute a statistically converged uniform $\mathbb{P}^3$ solution. For each run we also tabulated in terms of percentage how the CPU time splits across the main tasks (column "execution [%]"). The values are quite similar for the different polynomial approximation and are well in line with those reported in Table 1.

VI. Conclusions

In this work, a strategy based on an adjoint p-adaptation approach is presented to increase the computational efficiency of a high-order DG solver in the context of unsteady compressible flows simulations. In particular, the entropy-adjoint approach[28] is applied to run-time averaged field to drive the adaptation. For the scale-resolving simulations, the fine-space adjoint solution is computed using a patch reconstruction approach, which avoids the need of assembling a finer space problem and keeps the computational costs low. A dynamic load balancing strategy after each adaptation iteration is also proposed, based on a multi-constraint domain decomposition algorithm. The approach, applied to the two-dimensional flow around a square cylinder, performs similarly to a drag-based averaged adjoint adaptation method at a cheaper computational cost, demonstrating the benefits of using the proposed adaptation method. Three dimensional simulations involving the implicit LES of compressible turbulent flows show the suitability of the solver to reduce the computational costs of scale-resolving simulations involving bluff bodies and separated flows using massively parallel platforms. Comparisons of output quantities of engineering interest are also considered, highlighting the accuracy of the adapted discretization. Future work will be devoted to the extension of the current methodologies for the simulation of incompressible flows, the

use of matrix-free iterative solution strategies[66] and preconditioners[67,68] to further increase the computational efficiency.

Appendix A

In this work, load balancing is obtained via an optimized re-partitioning of the mesh performed by the Metis library.[52] Metis allows to drive the graph partitioning according to user-defined constraints that can be applied as weights at the graph vertices, here corresponding to computational elements. This Appendix includes the definition of the weights.

1. $w_1 = {N_v}^2$, where $N_v = \Pi_{i=1}^{d} (k+i)/i$ is the number of DoFs related to the polynomial degree k, local to the mesh element, and thus to the vertex of the graph. The w_1 estimate represents the leading-order term in the operation count for the matrix-vector product required by an iteration of the GMRES linear solver.

2. $w_2 = N_v N_{g,v}$, where $N_{g,v}$ is the number of Gauss points involved in the computation of the volume integrals (see Secs. II and B) part of the residuals vector $\mathbf{R}(\mathbf{W})$.

   The w_2 estimate represents the leading-order term in the operation count for the assembly of the volume contribution to the residuals vector.

3. $w_3 = \sum_{s=1}^{N_f} N_v N_{g,s}$, where N_f is the number of faces belonging to one element and $N_{g,s}$ is the number of Gauss points involved in the computation of the surface integrals (see Secs. II and B) part of the residuals vector $\mathbf{R}(\mathbf{W})$. The w_3 estimate represents the leading-order term in the operation count for the assembly of the surface contribution to the residuals vector.

4. $w_4 = N_v^2 N_{g,v}$ represents the leading-order term in the operation count for for the assembly of the volume contribution to the Jacobian $\mathbf{J} = \partial\mathbf{R}/\partial\mathbf{W}$.

5. $w_5 = \sum_{s=1}^{N_f} N_v^2 N_{g,v}$. w_5 represents the leading-order term in the operation count for the surface contribution to the Jacobian $\mathbf{J} = \partial\mathbf{R}/\partial\mathbf{W}$. Notice that $N_{g,v}$ is used instead of $N_{g,s}$ in the definition of w_5. This choice is motivated by the evaluation of the volume lift term in a loop over all the faces, see Crivellini and Bassi [69] for a discussion about the computational cost of this contribution.

An extended derivation of the operation count here reported can be found in Renac et al.[70]

VII. Acknowledgement

We acknowledge the CINECA award, under the ISCRA initiative (grant number HP10BEE6C4), for the availability of high performance computing resources and support. K. Fidkowski acknowledges support from the U.S. Department of Energy under grant DE-FG02-13ER26146/DE-SC0010341.

References

[1]M. Tugnoli, A. Abbà, L. Bonaventura, and M. Restelli. A locally p-adaptive approach for Large Eddy Simulation of compressible flows in a DG framework. *Journal of Computational Physics*, 349:33–58, 2017.

[2]F. Naddei, M. de la Llave Plata, V. Couaillier, and F. Coquel. 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.

[3]R. Hartmann and P. Houston. Adaptive discontinuous Galerkin finite element methods for the compressible Euler equations. *Journal of Computational Physics*, 183(2):508–532, 2002.

[4]K. J. Fidkowski and Y. Luo. Output-based space–time mesh adaptation for the compressible Navier–Stokes equations. *Journal of Computational Physics*, 230(14):5753–5773, 2011.

[5]M. Ceze and K. J. Fidkowski. Drag prediction using adaptive discontinuous finite elements. *Journal of Aircraft*, 51(4):1284–1294, 2014.

[6]K. J. Fidkowski. Output-based space–time mesh optimization for unsteady flows using continuous-in-time adjoints. *Journal of Computational Physics*, 341:258–277, 2017.

[7]F. Bassi, L. Botti, A. Colombo, A. Crivellini, A. Ghidoni, A. Nigro, and S. Rebay. Time integration in the discontinuous Galerkin code MIGALE - Unsteady problems. In *IDIHOM: Industrialization of High-Order Methods-A Top-Down Approach*, pages 205–230. Springer, 2015.

[8]A Uranga, P-O Persson, M Drela, and J Peraire. 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.

[9]C Carton Wiart, Koen Hillewaert, Laurent Bricteux, and Grégoire Winckelmans. 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.

[10]RC Moura, SJ Sherwin, and Joaquim Peiró. 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.

[11]R.C. Moura, G. Mengaldo, J. Peiró, and S.J. Sherwin. 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.

[12]A. D. Beck, T. Bolemann, D. Flad, H. Frank, G. J. Gassner, F. Hindenlang, and C. D. Munz. High-order discontinuous Galerkin spectral element methods for transitional and turbulent flow simulations. *International Journal for Numerical Methods in Fluids*, 76(8):522–548, 2014.

[13]F Bassi, L Botti, A Colombo, A Crivellini, A Ghidoni, and F Massa. 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.

[14]A. Garai, L.T. Diosady, S.M. Murman, and N.K. Madavan. 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.

[15]I. B. Celik, Z. N. Cehreli, and I. Yavuz. Index of Resolution Quality for Large Eddy Simulations. *Journal of Fluids Engineering*, 127(5):949–958, 09 2005.

[16]Toosi S. and Larsson J. Anisotropic grid-adaptation in Large Eddy Simulations. *Computers & Fluids*, 156:146 – 161, 2017.

[17]J. Hoffman. Computation of mean drag for bluff body problems using adaptive DNS/LES. *SIAM Journal on Scientific Computing*, 27(1):184–207, 2005.

[18]T.J. Barth. Space-time error representation and estimation in Navier-Stokes calculations. *Lecture Notes in Computational Science and Engineering*, 56:29–48, 2007.

[19]GL Eyink, TWN Haine, and DJ Lea. Ruelle's linear response formula, ensemble adjoint schemes and lévy flights. *Nonlinearity*, 17(5):1867, 2004.

[20]Y. Shimizu and K. J. Fidkowski. Output error estimation for chaotic flows. In *46th AIAA Fluid Dynamics Conference*, page 3806, 2016.

[21]Qiqi Wang, Rui Hu, and Patrick Blonigan. Least squares shadowing sensitivity analysis of chaotic limit cycle oscillations. *Journal of Computational Physics*, 267:210–224, 2014.

[22]Nisha Chandramoorthy, Pablo Fernandez, Chaitanya Talnikar, and Qiqi Wang. Feasibility analysis of ensemble sensitivity computation in turbulent flows. *AIAA Journal*, 57(10):4514–4526, 2019.

[23]Patrick J Blonigan. Adjoint sensitivity analysis of chaotic dynamical systems with non-intrusive least squares shadowing. *Journal of Computational Physics*, 348:803–826, 2017.

[24]Angxiu Ni and Chaitanya Talnikar. Adjoint sensitivity analysis on chaotic dynamical systems by non-intrusive least squares adjoint shadowing (nilsas). *Journal of Computational Physics*, 395:690–709, 2019.

[25]F. Bassi, A. Colombo, A. Crivellini, M. Franciolini, A. Ghidoni, G. Manzinali, and G. Noventa. Under-resolved simulation of turbulent flows using a p-adaptive discontinuous Galerkin method. *Springer Proceedings in Physics*, 226:157–162, 2019.

[26]A. Colombo, G. Manzinali, A. Ghidoni, G. Noventa, M. Franciolini, A. Crivellini, and F. Bassi. A p-adaptive implicit discontinuous galerkin method for the under-resolved simulation of compressible turbulent flows. In *Proceedings of the 6th European Conference on Computational Mechanics: Solids, Structures and Coupled Problems, ECCM 2018 and 7th European Conference on Computational Fluid Dynamics, ECFD 2018*, pages 4159–4170, 2020.

[27]G. Gassner, M. Staudenmaier, F. Hindenlang, M. Atak, and C. D. Munz. A space-time adaptive discontinuous Galerkin scheme. *Computers & Fluids*, 117:247 – 261, 2015.

[28]K. J. Fidkowski and P. L. Roe. An entropy adjoint approach to mesh refinement. *SIAM Journal on Scientific Computing*, 32(3):1261–1287, 2010.

[29]K. J. Fidkowski, M. Ceze, and P. L. Roe. Entropy-based drag-error estimation and mesh adaptation in two dimensions. *Journal of Aircraft*, 49(5):1485–1496, 2012.

[30]K. T. Doetsch and K. J. Fidkowski. A combined entropy and output-based adjoint approach for mesh refinement and error estimation. AIAA Paper 2018–0918, 2018.

[31]K. J. Fidkowski and D. L. Darmofal. Review of output-based error estimation and mesh adaptation in computational fluid dynamics. *AIAA journal*, 49(4):673–694, 2011.

[32]M. Ceze, L. Diosady, and S.M. Murman. Development of a high-order space-time matrix-free adjoint solver. In *54th AIAA Aerospace Sciences Meeting*, page 0833, 2016.

[33]G. Zenoni, T. Leicht, A. Colombo, and L. Botti. An agglomeration-based adaptive discontinuous Galerkin method for compressible flows. *International Journal for Numerical Methods in Fluids*, 85(8):465–483, 2017.

[34]Francesco Bassi, Alessandro Colombo, Andrea Crivellini, Krzysztof Fidkowski, Matteo Franciolini, Antonio Ghidoni, and Gianmaria Noventa. An entropy-adjoint p-adaptive discontinuous galerkin method for the under-resolved simulation of turbulent flows. In *AIAA Aviation 2019 Forum*, page 3418, 2019.

[35]J. Kim, D. J. Bodony, and J. B. Freund. Adjoint-based control of loud events in a turbulent jet. *Journal of Fluid Mechanics*, 741:28–59, 2014.

[36]Least squares shadowing sensitivity analysis of chaotic limit cycle oscillations. *Journal of Computational Physics*, 267:210 – 224, 2014.

[37]K.J. Fidkowski. Output-based space-time mesh optimization for unsteady flows using continuous-in-time adjoints. *Journal of Computational Physics*, 341:258–277, 2017.

[38]F. Bassi, L. Botti, A. Colombo, D. A. Di Pietro, and P. Tesini. On the flexibility of agglomeration based physical space discontinuous Galerkin discretizations. *Journal of Computational Physics*, 231(1):45–65, 2012.

[39]F. Bassi, S. Rebay, G. Mariotti, S. Pedinotti, and M. Savini. A high-order accurate discontinuous finite element method for inviscid and viscous turbomachinery flows. In R. Decuypere and G. Dibelius, editors, *Proceedings of the 2nd European Conference on Turbomachinery Fluid Dynamics and Thermodynamics*, pages 99–108, Antwerpen, Belgium, March 5–7 1997. Technologisch Instituut.

[40]P. L. Roe. Approximate riemann solvers, parameter vectors, and difference schemes. *Journal of computational physics*, 43(2):357–372, 1981.

[41]J. J. Gottlieb and C. P. T. Groth. Assessment of Riemann solvers for unsteady one-dimensional inviscid flows of perfect gases. *Journal of Computational Physics*, 78(2):437–458, 1988.

[42]F. Bassi, L. Botti, A. Colombo, A. Ghidoni, and F. Massa. 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.

[43]J. Lang and J. Verwer. ROS3P—An accurate third-order Rosenbrock solver designed for parabolic problems. *BIT*, 41(4):731–738, 2001.

[44]S. Balay, M. F. Adams, J. Brown, P. Brune, K. Buschelman, V. Eijkhout, W. D. Gropp, Kaushik D., M. G. Knepley, L. C. McInnes, F. Barry, K. R. Smith, and H. Zhang. PETSc Web page. `http://www.mcs.anl.gov/petsc`, 2014.

[45]P. Sagaut. *Large Eddy Simulation for Incompressible Flows: An Introduction.* 2006.

[46]E. J. Nielsen and B. Diskin. Discrete adjoint-based design for unsteady turbulent flows on dynamic overset unstructured grids. *AIAA journal*, 51(6):1355–1373, 2013.

[47]Y. Shimizu and K. J. Fidkowski. Output-based error estimation for chaotic flows using reduced-order modeling. In *2018 AIAA Aerospace Sciences Meeting*, page 0826, 2018.

[48]P. J. Blonigan, P. Fernandez, S. M. Murman, Q. Wang, G. Rigas, and L. Magri. Toward a chaotic adjoint for LES. *arXiv preprint arXiv:1702.06809*, 2017.

[49]M. Braack, E. Burman, and N. Taschenberger. Duality based a posteriori error estimation for quasi-periodic solutions using time averages. *SIAM Journal on Scientific Computing*, 33(5):2199–2216, 2011.

[50]D. Destarac. Far-field/near-field drag balance and applications of drag extraction in CFD. *VKI Lecture Series*, 2:3–7, 2003.

[51]R. Rannacher. Adaptive Galerkin finite element methods for partial differential equations. *Journal of Computational and Applied Mathematics*, 128(1-2):205–233, 2001.

[52]G. Karypis and V. Kumar. METIS: Unstructured graph partitioning and sparse matrix ordering system, version 5.0. Technical report, 2009.

[53]A. Ghidoni, E. Pelizzari, S. Rebay, and V. Selmin. 3D anisotropic unstructured grid generation. *International journal for numerical methods in fluids*, 51(9-10):1097–1115, 2006.

[54]4$^{\text{th}}$ international workshop on high order methods. `https://how4.cenaero.be`.

[55]D. A. Lysenko, I. S. Ertesvåg, and K. E. Rian. 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, Dec 2012.

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

[57]X. Ma, G.-S. Karamanos, and G. E. Karniadakis. Dynamics and low-dimensionality of a turbulent near wake. *Journal of Fluid Mechanics*, 410:29–65, 2000.

[58]J. S. Son and T. J. Hanratty. Velocity gradients at the wall for flow around a cylinder at Reynolds numbers from 5×10^3 to 10^5. *Journal of Fluid Mechanics*, 35(2):353–368, 1969.

[59]P. Parnaudeau, J. Carlier, D. Heitz, and E. Lamballais. Experimental and numerical studies of the flow over a circular cylinder at Reynolds number 3900. *Physics of Fluids*, 20, 08 2008.

[60]J. Wissink and W. Rodi. Numerical study of the near wake of a circular cylinder. *International Journal of Heat and Fluid Flow*, 29:1060–1070, 08 2008.

[61]P. W. Bearman and E. D. Obasaju. An experimental study of pressure fluctuations on fixed and oscillating square-section cylinders. *Journal of Fluid Mechanics*, 119:297–321, 1982.

[62]F.X. Trias, A. Gorobets, and A. Oliva. Turbulent flow around a square cylinder at Reynolds number 22,000: A DNS study. *Computers & Fluids*, 123:87 – 98, 2015.

[63]A. Sohankar, L. Davidson, and C. Norberg. Large Eddy Simulation of Flow Past a Square Cylinder: Comparison of Different Subgrid Scale Models . *Journal of Fluids Engineering*, 122(1):39–47, 11 1999.

[64]D. A. Lyn, S. Einav, W. Rodi, and J.-H. Park. A laser-doppler velocimetry study of ensemble-averaged characteristics of the turbulent near wake of a square cylinder. *Journal of Fluid Mechanics*, 304:285–319, 1995.

[65]M. Minguez, C. Brun, R. Pasquetti, and E. Serre. Experimental and high-order LES analysis of the flow in near-wall region of a square cylinder. *International Journal of Heat and Fluid Flow*, 32(3):558 – 566, 2011. 8th International Symposium on Engineering Turbulence Modelling and Measurements, Marseille, France,June 9 to 11, 2010.

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

[67]M. Franciolini, L. Botti, A. Colombo, and A. Crivellini. p-Multigrid matrix-free discontinuous Galerkin

solution strategies for the under-resolved simulation of incompressible turbulent flows. *arXiv preprint arXiv:1809.00866*, 2018.

[68]L. Botti, A. Colombo, A. Crivellini, and M. Franciolini. {*h*-*p*-*hp*}-multilevel discontinuous Galerkin solution strategies for elliptic operators. *International Journal of Computational Fluid Dynamics*, 33(9):362–370, 2019.

[69]A. Crivellini and F. Bassi. An implicit matrix-free discontinuous Galerkin solver for viscous and turbulent aerodynamic simulations. *Computers & fluids*, 50(1):81–93, 2011.

[70]F. Renac, S. Gérald, C. Marmignon, and F. Coquel. Fast time implicit–explicit discontinuous Galerkin method for the compressible Navier–Stokes equations. *Journal of Computational Physics*, 251:272–291, 2013.