Adaptation strategies for high order discontinuous Galerkin methods based on Tau-estimation

Moritz Kompenhans*, Gonzalo Rubio*, Esteban Ferrer, Eusebio Valero

ETSIAE (School of Aeronautics), Universidad Politécnica de Madrid, Plaza de Cardenal Cisneros 3, 28040 Madrid, Spain

In this paper three p-adaptation strategies based on the minimization of the truncation error are presented for high order discontinuous Galerkin methods. The truncation error is approximated by means of a τ-estimation procedure and enables the identification of mesh regions that require adaptation. Three adaptation strategies are developed and termed *a posteriori*, quasi-*a priori* and quasi-*a priori corrected*. All strategies require fine solutions, which are obtained by enriching the polynomial order, but while the former needs time converged solutions, the last two rely on non-converged solutions, which lead to faster computations. In addition, the high order method permits the spatial decoupling for the estimated errors and enables anisotropic p-adaptation.

These strategies are verified and compared in terms of accuracy and computational cost for the Euler and the compressible Navier–Stokes equations. It is shown that the two quasi-*a priori* methods achieve a significant reduction in computational cost when compared to a uniform polynomial enrichment. Namely, for a viscous boundary layer flow, we obtain a speedup of 6.6 and 7.6 for the quasi-*a priori* and quasi-*a priori corrected* approaches, respectively.

1. Introduction

Finite volume, finite difference and finite element numerical methods are regarded as the mainstream and well accepted techniques to solve the Navier–Stokes equations that govern fluid flow (see Peiro and Sherwin [35]). However, during the last decades the interest of the fluid dynamics community has shifted towards high order methods, such as spectral methods or discontinuous Galerkin, see Wang et al. [50] for recent advances. These methods enable the use of high degree polynomials inside each computational element to approximate the numerical solution. By doing so, the accuracy of the solution is improved and the numerical error is shown to decrease exponentially for smooth flows.

Discontinuous Galerkin (DG) methods were developed by the pioneering work of Reed and Hill [38] in 1973 in the framework of hyperbolic partial differential equations for the neutron transport equation, and it was only in the late 1990s that the method was generalized to elliptic and convection–diffusion problems; e.g. Bassi and Rebay in 1997 [6], Baumann in 1997 [7] or Cockburn and Shu in 1998 [13]. Since, DG methods have proven useful in solving the compressible, e.g. [6,31,33,34,30], and the incompressible NS equations, e.g. [5,39,12,45,18,19,17].

The high order DG approach relaxes the continuity requirement needed in continuous methods and allows discontinuities between elements, e.g. non-conforming meshes with hanging nodes or varying polynomial orders. This characteristic

* Corresponding authors.

E-mail addresses: moritz.kompenhans@upm.es (M. Kompenhans), g.rubio@upm.es (G. Rubio), esteban.ferrer@upm.es (E. Ferrer), eusebio.valero@upm.es (E. Valero).

enhances the flexibility of the method when dealing with complex flows and eases the incorporation of adaptation strategies. Since discontinuities are allowed in the numerical flow, local adaptation can be performed either by locally refining the mesh (h-refinement) or by a local increase in the polynomial order in certain elements (p-refinement). Examples of both strategies may be found in Mavriplis [32], Van der Vegt et al. [49] or Roy et al. [40]. The flexibility for h or p-adaptation raises the question of which strategy provides better solutions with a minimal cost. For a broad range of cases with smooth numerical solutions, an increase of the polynomial order has shown to outperform h-refinement, in terms of accuracy for a given number of degrees of freedom (DoF), e.g. Karniadakis and Sherwin [29], Ferrer et al. [16].

More important than the local accuracy of the method is to identify the flow regions that require refinement to obtain an optimal global accuracy. An efficient refinement strategy would guarantee high accuracy whilst keeping the computational cost and number of DoF low. The classical and almost naive approach considers creating denser meshes in the flow regions that present geometric complexities and high flow gradients to maximize the accuracy. To identify these zones, density or pressure gradients are often used as the criterion for refinement. This so called "feature based adaptation" has been broadly applied to improve the accuracy around shock waves, expansion fans, contact discontinuities and boundary layers, e.g. Dwight et al. [15], Ainsworth et al. [1]. However, the main limitation of this approach resides in that there is no clear and direct relation between the computed feature (used as adaptation criterion) and the numerical errors and therefore the overall accuracy is generally difficult to predict and control.

To overcome these drawbacks, adjoint based adaptation procedures were developed and have gained popularity during recent years, e.g. Hartmann [27], Balasubramanian and Newman [3]. These techniques require the selection of a functional target (e.g. lift or drag in airfoil computations) to provide information about the regions that require refinement.

An alternative to adjoint based methods is provided by the truncation error (see Fraysse et al. [22] or Derlaga et al. [14] for comparisons to adjoints). This technique, which is the main topic of our work, is seen as a promising methodology to avoid the high computational cost associated to adjoint methods whilst keeping the favorable fast computation properties and accuracy inherited from error estimation methods. In addition, let us note that truncation error based methods do not require the selection of a particular functional and target the numerical accuracy of all functionals.

The truncation error is defined as the difference between the discrete partial differential equation (PDE) and the exact PDE operator, both applied to the exact solution of the problem, as detailed by Phillips and Roy [37]. Mathematically, this can be written as

$$\tau^N = \mathcal{R}^N(u) - \mathcal{R}(u), \tag{1}$$

where τ^N denotes the truncation error for a polynomial order N, $\mathcal{R}$ is the partial differential operator, $\mathcal{R}^N$ the discrete partial differential operator (of order N) and u represents the exact solution.

The discretization error (i.e. the difference between the exact solution to the PDE and the exact solution to the discretized PDE) and the truncation error are related through the Discretization Error Transport Equation (DETE), see Roy [41]. Roy has shown that the truncation error acts as a local source for the discretization error, which is convected and diffused through the domain with the flow. Hence, the resulting discretization error is a combination of the locally generated error and the error transported by flow advection and diffusion. This relationship provides a valid argument for the use of the truncation error as a sensor for a mesh adaptation algorithm, e.g. Syrakos et al. [47], Frayssee et al. [20,21]. Moreover, in high order discretizations, an accurate estimate for the error also enables modification of the polynomial order via a procedure known as τ-extrapolation to better capture the numerical solution (see Bernert [9]).

The truncation error can be estimated using evaluations of the discrete PDE operator on a hierarchy of meshes. Shih and Williams [46] proposed a multiple grid method with interpolation from the coarse to the fine grid. More recently, Gao and Wang [25] proposed a similar approach with interpolation from low to high polynomial order. The main drawback of the coarse to fine grid approaches is that they tend to over-predict the truncation error, Phillips [36]. The truncation error may be also estimated by means of the τ-estimation method of Brandt [10]. This estimate relies on the evaluation of the fine grid solution (e.g. higher order solution) on a coarse mesh (e.g. space spanned by lower order functions). The fine to coarse approach is more accurate and more costly than the coarse to fine one. The seminal works of Berger [8], Bernert [9] and Fulton [24] posed the fundamentals of the τ-estimation method and studied the conditions on the restriction operators to transfer solutions from fine to coarse and vice-versa, mainly for finite difference uniform meshes. Syrakos and Goulas [48] successfully implemented the τ-estimation method for a finite volume discretization and the incompressible Navier–Stokes equations. Fraysse et al. [20,23] extended these analyses to finite volume discretizations on any kind of meshes, with an interesting extension to non-converged temporal solutions. More recently, Rubio et al. [43] extended this methodology to continuous high-order methods using a spectral collocation method. It was shown that some of the fundamental assumptions about the error tendency, that are well established for low-order methods, are no longer valid when dealing with high-order schemes.

The extensions of τ-estimation to high order discontinuous Galerkin methods has been recently performed by Rubio et al. [44] for a simple advection equation. Additionally in [44], a quasi-*a priori* idea introduced by Fraysse et al. [23] was adapted to high order discontinuous discretizations using scalar partial differential equations. This quasi-*a priori* approach enables the computation of the truncation error using solutions that are not fully converged in time. Note that the solution is considered converged (or steady) when the iterative errors are below a certain threshold, normally close to machine roundoff. Obtaining the truncation error before the solution is totally converged, saves computational resources while providing valuable information to perform adaptation.

The objective of this work is to extend the achievements of Rubio et al. [44] for simple differential equations to the compressible Navier–Stokes equations in multiple dimensions. In addition, we propose efficient and robust algorithms for local p-adaptation using discontinuous Galerkin discretizations. Converged and non-converged solutions are considered in our analysis and named *a posteriori*, quasi-*a priori* and quasi-*a priori corrected* adaptation strategies, respectively. The accuracy, efficiency and computational costs of the different strategies are included. Finally, let us note that the high order technique used in this work enables the anisotropic treatment of the error in terms of the flow direction, hence allowing the p-adaptation refinement to be performed differently depending on the flow orientation.

The paper is organized as follows. In section 2 the discontinuous Galerkin method is detailed. In section 3 the different sources of errors are explained, followed by a description on the means for estimating the truncation error in section 4. The complete adaptation process is detailed in section 5 and verified using a manufactured solution test case with the Euler equations. A viscous boundary layer case is detailed in section 6, where the adaptation procedures are compared for the compressible Navier–Stokes equations. Finally, the efficiency of the methods is quantified in section 7.

2. Discontinuous Galerkin spectral element method

Discontinuous Galerkin methods were first developed [38] to solve conservation laws of the form

$$u_t + \nabla \cdot \mathbf{f} = 0. \tag{2}$$

A particular nodal variant of the discontinuous Galerkin technique is used here, the Discontinuous Galerkin Spectral Element Method (DGSEM), see Kopriva [30], which solves Eq. (2) in general three-dimensional geometries in which the domain Ω is divided into k non-overlapping quadrilateral elements Ω^k. In this paper, the approximation is restricted, without loss of generality, to two-dimensional problems. Each element in the domain is mapped individually onto a unit square by an iso-parametric transformation. This mapping between the unit square and the physical space is described generically by $\mathbf{x} = \mathbf{r}(\xi, \eta)$ where ξ, η are the computational coordinates on the unit square. For a complete derivation of the DGSEM method, the reader is referred to Kopriva [30]. On each element the solution is approximated by a series of orthogonal (w.r.t. the L_2 inner product) polynomials P_N of degree N, for more information see Canuto et al. [11]. As a basis for this approximation, a set of Lagrange interpolating polynomials $l_j(\xi)$, $j = 0, \ldots, N$ is used and can be written as

$$l_j(\xi) = \prod_{\substack{i=j \\ i \neq j}}^{N} \frac{\xi - \xi_i}{\xi_j - \xi_i}. \tag{3}$$

The nodal points ξ_i, which represent the grid points of the scheme, are chosen to be the nodes of the Legendre–Gauss quadrature. Multiple space dimensions are spanned by tensor products of these polynomials, so that we write $P_{N,N} = P_N \times P_N$. For simplicity of exposition only, we will take the same polynomial order in each direction, though this is not required in practice. As a matter of fact, we will use different polynomial orders in each direction in other sections of this paper. From this point onwards and to simplify the notation, we consider that no mapping is performed, i.e. only the reference element in the computational space is used. In two dimensions, the spectral element method approximates the solution and the fluxes element-by-element by the polynomials

$$u^N(\xi, \eta) = \sum_{\mu,\nu=0}^{N} u^N_{\mu,\nu} \phi_{\mu,\nu}, \quad \mathbf{f}^N(\xi, \eta) = \sum_{\mu,\nu=0}^{N} \mathbf{f}^N_{\mu,\nu} \phi_{\mu,\nu}, \tag{4}$$

where $\phi_{\mu,\nu} = \ell_\mu(\xi)\, \ell_\nu(\eta)$. The nodal (grid point) values of the fluxes are computed from the grid point values of the solution, i.e. $\mathbf{f}^N_{\mu,\nu} = \mathbf{f}\left(u^N_{\mu,\nu}\right)$. Note that $u^N_{\mu,\nu}$ is not the nodal value of $u(\xi, \eta)$, but the result of solving the discretized PDE. Therefore we distinguish

$$u^N(\xi, \eta) = \sum_{\mu,\nu=0}^{N} u^N_{\mu,\nu} \phi_{\mu,\nu}, \quad \text{and} \quad I_N u(\xi, \eta) = \sum_{\mu,\nu=0}^{N} u_{\mu,\nu} \phi_{\mu,\nu}. \tag{5}$$

The former is the solution of the discretized PDE while the latter is the spectral interpolation of the exact solution of the PDE. The same applies to the fluxes

$$\mathbf{f}^N(\xi, \eta) = \sum_{\mu,\nu=0}^{N} \mathbf{f}^N_{\mu,\nu} \phi_{\mu,\nu}, \quad \text{and} \quad I_N \mathbf{f}(\xi, \eta) = \sum_{\mu,\nu=0}^{N} \mathbf{f}_{\mu,\nu} \phi_{\mu,\nu}. \tag{6}$$

As generally imposed in variational formulations, the PDE residual is required to be orthogonal to the approximation space locally within an element. Thus,

$$\left(u^N_t, \phi_{i,j}\right) + \left(\nabla \cdot \mathbf{f}^N, \phi_{i,j}\right) = 0 \quad i, j = 0, 1, \ldots, N, \tag{7}$$

where (a, b) represents the usual $L_2(\Omega)$ inner product. Integration by parts of Eq. (7) gives

$$\left(u_t^N, \phi_{i,j}\right) + \sum_{e\in\partial\Omega} \int_e \mathbf{f}^N \cdot \mathbf{n}\phi_{i,j} dS - \left(\mathbf{f}^N, \nabla\phi_{i,j}\right) = 0 \quad i, j = 0, 1, \ldots, N, \tag{8}$$

where $\mathbf{n}$ is the outward normal unit vector, $\partial\Omega$ represents the boundary of the element and the summation is extended over the edges e of $\partial\Omega$.

To solve this set of equations, the integrals are replaced by Legendre–Gauss quadratures, which in two-dimensional rectangular domains become

$$\int_{-1}^{1}\int_{-1}^{1} v(\xi, \eta)\, d\xi d\eta = \sum_{i,j=0}^{N} v\left(\xi_i, \eta_j\right) w_i w_j \quad \forall v \in P_{2N+1,2N+1}, \tag{9}$$

where ξ_i, η_j are the nodes of the Legendre–Gauss quadrature and w_i, w_j the corresponding weights. This replacement is exact provided that the element sides are straight. Finally, substituting Eq. (4) into Eq. (8), taking into account Eq. (9) and the discrete orthogonality of the Lagrange interpolating polynomials (Canuto et al. [11]) yields

$$u_{t_{i,j}}^N w_i w_j + \sum_{e\in\partial\Omega} \int_e^N \mathbf{f}^N \cdot \mathbf{n}\phi_{i,j} dS - \sum_{\mu,\nu=0}^{N} \mathbf{f}_{\mu,\nu}^N \cdot \nabla\phi_{i,j} w_\mu w_\nu = 0 \quad i, j = 0, 1, \ldots, N. \tag{10}$$

In Eq. (10), $\sum_{e\in\partial\Omega} \int_e^N \mathbf{f}^N \cdot \mathbf{n}\phi_{i,j} dS$ is the sum of all the integrals over all the edges of the element approximated by quadrature. The boundary term can be written as follows

$$\begin{aligned}\sum_{e\in\partial\Omega} \int_e^N \mathbf{f}^N \cdot \mathbf{n}\phi_{i,j} dS = {} & \mathbf{f}^N\left(1, \eta_j\right)\phi_{i,j}\left(1, \eta_j\right) w_j - \mathbf{f}^N\left(-1, \eta_j\right)\phi_{i,j}\left(-1, \eta_j\right) w_j \\ & + \mathbf{f}^N\left(\xi_i, 1\right)\phi_{i,j}\left(\xi_i, 1\right) w_i - \mathbf{f}^N\left(\xi_i, -1\right)\phi_{i,j}\left(\xi_i, -1\right) w_i.\end{aligned} \tag{11}$$

Finally, we define the discrete partial differential operator for each element as

$$\mathcal{R}^N(u) = \sum_{e\in\partial\Omega} \int_e^N I_N \mathbf{f}^* \cdot \mathbf{n}\phi_{i,j} dS - \sum_{\mu,\nu=0}^{N} \mathbf{f}_{\mu,\nu} \cdot \nabla\phi_{i,j} w_\mu w_\nu \quad i, j = 0, 1, \ldots, N, \tag{12}$$

where $\mathbf{f}^*$ is the approximation of the Riemann problem (e.g. Roe's method is selected in this work). Having obtained a suitable discrete expression for each elemental contribution, it suffices to sum over all elements in the mesh and apply the boundary conditions weakly to finalize the DGSEM method, see details in Korpiva [30].

The τ-estimation procedure requires to interpolate the solution from a fine (using a high polynomial order) to a coarse grid (using a low polynomial order). Since the DGSEM works with the values of polynomial expansions from a set of nodes, the interpolant from order P to N is

$$I_P^N \mathbf{f}^P\left(\xi_i, \eta_j\right) = \sum_{\mu,\nu=0}^{P} \mathbf{f}\left(\xi_\mu, \eta_\nu\right)\phi_{\mu,\nu}\left(\xi_i, \eta_j\right), \quad i, j = 0, \ldots, N, \tag{13}$$

where $\left(\xi_i, \eta_j\right)$ are the $(N+1)\times(N+1)$ Gauss–Legendre nodal points of order N and $\left(\xi_\mu, \eta_\nu\right)$ the $(P+1)\times(P+1)$ Gauss–Legendre nodal points of order P. $I_P^N f^P$ is the polynomial of order N whose values in the Gauss–Legendre nodes of order N match $\mathbf{f}^P$. To apply the discrete operator $\mathcal{R}^N$ to a solution of different order u^P, it is necessary to evaluate this solution at the Gauss–Legendre nodes of order N, i.e. to interpolate to a lower polynomial order coarse grid. For compactness, the notation in this work omits the interpolant such that $\mathcal{R}^N u^P = \mathcal{R}^N I_P^N u^P$.

3. Definition of errors

In this section we define three types of errors that are necessary to understand the error estimation procedure and the following adaptation strategies.

3.1. Discretization error

The discretization error is the difference between the exact solution of the problem, u, and the approximate solution, u^N,

$$\epsilon^N \equiv u - u^N. \tag{14}$$

In the asymptotic range, for sufficiently smooth functions, we can assume that the convergence of this error is spectral (or exponential). This means that for a fixed size h_k of the elements, the behavior in each element Ω_k with the polynomial order N_k is bounded (in the L_∞ norm) by

$$\|\epsilon_k^N\|_{L_\infty} \leq \sum_{k=1}^{K} C_k \exp\left(-\eta_k N_k\right), \tag{15}$$

where C_k and η_k are constants that depend on the smoothness of the functions (see Canuto et al. [11] or Hesthaven and Warburton [28]) and K is the total number of elements. From Eq. (15) it may be deduced that to obtain an accurate solution, it is necessary not only to properly resolve the element k, but also the surrounding neighbors. Let us note that throughout this work, if a vector or a system of equations is normed, then the L_∞ norm of the system requires finding the L_∞ norm for each equation and retain the maximum of all.

For several dimensions, a "tensor-product"-type error bound is valid for the discretization error

$$\|\epsilon_k^N\|_{L_\infty} \leq \sum_{k=1}^{K} \sum_{i=1}^{N_{dim}} C_{ik} \exp\left(-\eta_{ik} N_{ik}\right), \tag{16}$$

where N_{dim} is the number of spatial dimensions of the problem. See [44,26] for a more detailed explanation of anisotropic error bounds in discontinuous Galerkin methods.

3.2. Iteration error

We define the iteration error as the difference between the steady, converged approximate solution u^N and the current approximation of the solution (not-converged) $\tilde{u}^N$,

$$\epsilon_{it}^N \equiv u^N - \tilde{u}^N. \tag{17}$$

The iteration error is directly related to the residual of the iterative method, e.g. Runge–Kutta in a pseudo time iterative procedure, used to advance in time the solution of the discrete set of equations. Indeed, this can be seen using Taylor series:

$$\mathcal{R}^N\left(\tilde{u}^N\right) = \mathcal{R}^N\left(u^N - \epsilon_{it}^N\right) = \cancel{\mathcal{R}^N\left(u^N\right)} - \left.\frac{\partial \mathcal{R}^N}{\partial u^N}\right|_{u^N} \epsilon_{it}^N + \mathcal{O}\left(\epsilon_{it}^N\right)^2, \tag{18}$$

where the iteration error is clearly linked to the residual of the discrete operator.

3.3. Truncation error

The truncation error is defined as the difference between the discrete operator and the exact continuous operator applied to the exact solution,

$$\tau^N \equiv \mathcal{R}^N(u) - \mathcal{R}(u). \tag{19}$$

When u is the steady exact solution, then $\mathcal{R}(u) = 0$ and the truncation error becomes

$$\tau^N = \mathcal{R}^N(u). \tag{20}$$

The assumption of u being the steady exact solution of the problem $\mathcal{R}(u) = 0$ means that it is only valid for steady exact solutions. The truncation error defined in Eq. (20) provides a measure of the suitability of the spatial discretization to solve the steady problem. Unsteady problems are not considered in this analysis.

The truncation error, Eq. (20), and the discretization error, Eq. (14), are linked through the Discretization Error Transport Equation (DETE) equation [41]. This equation can be derived by substituting the definition of the discretization error, Eq. (14), into the definition of the truncation error, Eq. (20), and expanding using Taylor series, to obtain

$$\tau^N \approx \left.\frac{\partial \mathcal{R}^N}{\partial u^N}\right|_{u^N} \epsilon^N. \tag{21}$$

Eq. (21) is the general expression for $\mathcal{R}$ being a non-linear operator, however a similar result can be written for linear operators. As a consequence of Eq. (21), the truncation error follows an exponential convergence law, similar to the discretization error convergence law, described by Eqs. (15) and (16). See [44] for a more detailed analysis.

Furthermore, it is possible to use Eq. (21) to find a lower bound for the truncation error. Assuming that the problem is well posed (i.e. invertible Jacobian)

$$\left(\left.\frac{\partial \mathcal{R}^N}{\partial u^N}\right|_{u^N}\right)^{-1} \tau^N \approx \epsilon^N. \tag{22}$$

Taking norms

$$\left\|\left(\left.\frac{\partial \mathcal{R}^N}{\partial u^N}\right|_{u^N}\right)^{-1} \tau^N\right\|_{L_\infty} \approx \|\epsilon^N\|_{L_\infty}, \tag{23}$$

and using the Cauchy–Schwarz inequality, we have

$$\left\|\left(\left.\frac{\partial \mathcal{R}^N}{\partial u^N}\right|_{u^N}\right)^{-1}\right\|_{L_\infty} \|\tau^N\|_{L_\infty} \gtrsim \|\epsilon^N\|_{L_\infty}, \tag{24}$$

and rearranging, we obtain

$$\|\tau^N\|_{L_\infty} \gtrsim \frac{\|\epsilon^N\|_{L_\infty}}{\left\|\left(\left.\frac{\partial \mathcal{R}^N}{\partial u^N}\right|_{u^N}\right)^{-1}\right\|_{L_\infty}} = \frac{\|\epsilon^N\|_{L_\infty}}{\left\|\left(\left.\frac{\partial \mathcal{R}^N}{\partial u^N}\right|_{u^N}\right)\right\|_{L_\infty} \kappa\left(\left.\frac{\partial \mathcal{R}^N}{\partial u^N}\right|_{u^N}\right)}, \tag{25}$$

where $\kappa\left(\left.\frac{\partial \mathcal{R}^N}{\partial u^N}\right|_{u^N}\right) = \left\|\left(\left.\frac{\partial \mathcal{R}^N}{\partial u^N}\right|_{u^N}\right)^{-1}\right\|_{L_\infty} \left\|\left(\left.\frac{\partial \mathcal{R}^N}{\partial u^N}\right|_{u^N}\right)\right\|_{L_\infty}$ denotes the condition number (in the L_∞ norm) of the system Jacobian. Eq. (25) shows that there exists a direct relationship between truncation and discretization errors. The latter equation may be used to set a threshold for the truncation error based adaptation.

Additionally, it is possible to show that the truncation error controls functional errors. Let us consider a functional output $\mathcal{J}^N(u)$ (e.g. lift or drag coefficient) and expand using Taylor series about u:

$$\mathcal{J}^N(u) = \mathcal{J}^N\left(u^N\right) + \left.\frac{\partial \mathcal{J}^N}{\partial u^N}\right|_{u^N}\left(u - u^N\right) + \mathcal{O}\left(u - u^N\right)^2. \tag{26}$$

Rearranging Eq. (26) and taking norms we obtain

$$\|\mathcal{J}^N(u) - \mathcal{J}^N\left(u^N\right)\|_{L_\infty} \approx \left\|\left.\frac{\partial \mathcal{J}^N}{\partial u^N}\right|_{u^N}\left(u - u^N\right)\right\|_{L_\infty}. \tag{27}$$

Applying the Cauchy–Schwarz inequality to the right hand side of Eq. (27) and rearranging, we find a lower bound for the discretization error:

$$\frac{\|\mathcal{J}^N(u) - \mathcal{J}^N\left(u^N\right)\|_{L_\infty}}{\left\|\left.\frac{\partial \mathcal{J}^N}{\partial u^N}\right|_{u^N}\right\|_{L_\infty}} \lesssim \|\left(u - u^N\right)\|_{L_\infty} = \|\epsilon^N\|_{L_\infty}. \tag{28}$$

Finally combining Eq. (25) and Eq. (28) it is easy to show that the truncation error controls not only the discretization error but also the functional error

$$\|\tau^N\|_{L_\infty} \gtrsim \frac{\|\epsilon^N\|_{L_\infty}}{\left\|\left(\left.\frac{\partial \mathcal{R}^N}{\partial u^N}\right|_{u^N}\right)^{-1}\right\|_{L_\infty}} \gtrsim \frac{\|\mathcal{J}^N(u) - \mathcal{J}^N\left(u^N\right)\|_{L_\infty}}{\left\|\left(\left.\frac{\partial \mathcal{R}^N}{\partial u^N}\right|_{u^N}\right)^{-1}\right\|_{L_\infty} \left\|\left.\frac{\partial \mathcal{J}^N}{\partial u^N}\right|_{u^N}\right\|_{L_\infty}}. \tag{29}$$

This brief proof shows that by controlling the truncation error, one can control any derived functional. Therefore an adaption process based on the truncation error should enhance the accuracy of all functionals.

Additionally, we note that introducing the adjoint vector field $(\psi^N)^T \left.\frac{\partial \mathcal{R}^N}{\partial u^N}\right|_{u^N} = \left.\frac{\partial \mathcal{J}^N}{\partial u^N}\right|_{u^N}$ (and using again the Cauchy–Schwarz inequality) one may simplify Eq. (29) to

$$\|\tau^N\|_{L_\infty} \gtrsim \frac{\|\mathcal{J}^N(u) - \mathcal{J}^N\left(u^N\right)\|_{L_\infty}}{\|(\psi^N)^T\|_{L_\infty} \kappa\left(\left.\frac{\partial \mathcal{R}^N}{\partial u^N}\right|_{u^N}\right)}, \tag{30}$$

where, the previously introduced condition number (in the L_∞ norm) of the system Jacobian, is used again.

Finally, it should be noticed that the definition of the truncation error, Eq. (20), includes the exact solution of the problem. Since the exact solution is not available in general, an estimation for Eq. (20) is necessary. This is covered in the next section.

4. Error estimation

In this section, we introduce the *a posteriori* and the quasi-*a priori* methodology to estimate the error. The former relates to the estimation of the error for a converged solution, while the latter shows an approach to estimate the error based on a solution that is not converged in time.

Since the exact solution, that is required to calculate the truncation error in Eq. (20) is generally not available, the τ-estimation method uses an approximate solution instead to estimate the error. This approximate solution is obtained by solving the same problem with a higher polynomial order on each element. Then, the error can be estimated for all polynomial orders lower than specified, e.g. τ_P^N is the truncation error estimation using a fine simulation with order P to estimate the coarse error with polynomial order N, when $N < P$. This estimation is called *a posteriori* since it requires a fully converged solution of order P.

This approach can be extended to non-converged solutions. The quasi-*a priori* method permits accurate estimations of the truncation error, τ^N, on a fine mesh using a not necessarily converged solution, $\tilde{u}^P$. Furthermore, it is possible to derive a correction for this solution which can be incorporated into the method to overcome the lack of temporal convergence.

The expressions to estimate the truncation error were first deduced by Fraysse et al. [20] for finite volume schemes and extended later to spectral Chebyshev collocation methods and the DGSEM by Rubio et al. [43,44]. Here, we summarize their main conclusions for non-converged solutions. The approximated truncation error becomes

$$\tau_P^N \equiv \mathcal{R}^N\left(\tilde{u}^P\right) - \bar{I}_P^N \mathcal{R}^P\left(\tilde{u}^P\right). \tag{31}$$

Here, $\bar{I}_P^N$ is the transfer operator of the residual from order P to N, defined as

$$\bar{I}_P^N \equiv \tilde{\mathcal{R}}^N I_P^N \left(\tilde{\mathcal{R}}^P\right)^{-1}, \tag{32}$$

for linear operators. Note that the operator $\mathcal{R}^N(u) = \tilde{\mathcal{R}}^N(u) + S^N$ may be decomposed in a sum of a homogeneous operator $\tilde{\mathcal{R}}^N$ that is a function of u, and an independent term S^N, that accounts for the source terms and the value of the boundary conditions. The full derivation can be found in Rubio et al. [44] and has been included as an appendix at the end of this paper. The difference between the exact, Eq. (19), and the approximate truncation error, Eq. (31), is (Rubio et al. [44]):

$$\tau_P^N = \tau^N - \tilde{\mathcal{R}}^N\left(\epsilon^P\right). \tag{33}$$

Likewise, for non-linear operators the transfer operator is defined as

$$\bar{I}_P^N \equiv \left.\frac{\partial \mathcal{R}^N}{\partial u^N}\right|_{u^N} I_P^N \left(\left.\frac{\partial \mathcal{R}^P}{\partial u^P}\right|_{u^P}\right)^{-1} \tag{34}$$

and the difference between the exact and the approximate truncation error reads

$$\tau_P^N = \tau^N - \left.\frac{\partial \mathcal{R}^N}{\partial u^N}\right|_{u^N} \epsilon^P + \mathcal{O}\left(\epsilon^P\right)^2 + \mathcal{O}\left(\epsilon_{it}^P\right)^2. \tag{35}$$

In the case of non-converged solutions, the second term on the right hand side (RHS) of Eq. (31) acts as a correction term for the iteration error, ϵ_{it}^P. Indeed, from Eq. (18), it can be seen that

$$\epsilon_{it} \approx \left(\left.\frac{\partial \mathcal{R}^P}{\partial u^P}\right|_{u^P}\right)^{-1} \mathcal{R}^N(\tilde{u}^P). \tag{36}$$

However, the computation of this correction term is computationally expensive, since it requires the solution of a linear system, therefore we perform quasi-*a priori* estimations with and without this correction term. If no correction term is used, i.e. $\bar{I}_P^N \mathcal{R}^P(\tilde{u}^P)$ is not computed, then the difference between the exact and the approximate truncation error becomes

$$\tau_P^N = \tau^N - \left.\frac{\partial \mathcal{R}^N}{\partial u^N}\right|_{u^N} \epsilon^P - \underbrace{\left.\frac{\partial \mathcal{R}^N}{\partial u^N}\right|_{u^N} \epsilon_{it}^P}_{\text{first order iteration error}} + \mathcal{O}\left(\epsilon^P\right)^2 + \mathcal{O}\left(\epsilon_{it}^P\right)^2. \tag{37}$$

We will check the accuracy and computational time of both approaches.

For the *a posteriori* method (with converged solution), we have $u^P = \tilde{u}^P$, and the second term on the RHS of Eq. (31) reduces to:

$$\tau_P^N = \mathcal{R}^N\left(u^P\right). \tag{38}$$

Part A – Estimation
Integrate in time on the fine mesh P (with a high polynomial order) until steady state;
for $N < P$ **do**
  Estimate the truncation error τ_P^N with *no correction* term, Eq. (33);
end
Part B – Adaptation
for *each spatial direction* $d \leq N_{dim}$ **do**
  Set $P_{new}^d = 0$
  for $N = 1, P-1$ **do**
    if $\|\tau_P^N\|_{L_\infty} \leq$ *required threshold*(τ_{max}) **then**
      $P_{new}^d = N$
    end
  end
  if $P_{new}^d = 0$ **then**
    Determine interpolation parameters η and C using least squares,
    Calculate P_{new}^d using Eq. (41): $P_{new}^d = \frac{C - \log(\text{required threshold}(\tau_{max}))}{\eta}$
  end
end
Part C – Simulation
Interpolate converged solution to new p-adapted mesh;
Continue the simulation;

Algorithm 1: *A posteriori* τ-estimation adaptation for each element.

5. Adaptation process

While DG methods have been used extensively in recent years together with different adaptation strategies (Hartmann [27], Mavriplis [32] among others), limited work exists where high order mesh adaptation is combined with truncation error estimations. In this section three novel adaptation algorithms are presented for DG methods, an *a posteriori* approach, based on a converged solution, a quasi-*a priori* and a quasi-*a priori corrected* approach, the last two based on a non-converged temporal solution.

5.1. A posteriori τ-estimation

The proposed adaptation process based on *a posteriori* τ-estimation is summarized in Algorithm 1. The first part, A, of the algorithm encompasses the error estimation process that was explained earlier in section 4. Assuming a simulation that is converged until its steady state with a polynomial order P, the error estimates can be calculated for all orders N smaller than P, e.g. τ_P^N following expression Eq. (33), with $N = 1, \ldots, (P-1)$. It should be noticed that, for problems with several spatial dimensions, N_{dim}, the number of estimations provided by the method is $(P-1)^{N_{dim}}$. Once these errors are estimated for each element, they can be used to determine the new polynomial order needed to fulfill the predetermined truncation error threshold.

For illustration, Fig. 1 shows a typical plot of the estimation of the truncation error in log scale. The error $\|\tau_P^N\|_{L_\infty}$ in each element is shown as a function of the polynomial order P. Now, for a desired error threshold τ_{max} (e.g. 10^{-5}, horizontal dashed line in Fig. 1), two different possibilities appear depending on whether the finest solution u^P has been chosen to be accurate enough. In the first case (squares), the required error can be reached with a polynomial of order $P = 7$, which agrees with the estimation already performed in the element, this is known as *optimal scaling*. In the second case (triangles), the required error is not reached by any of the estimates with a lower order than P, and a higher polynomial order is needed (i.e. *suboptimal scaling*). In the last case, one may extrapolate the estimate to determine the polynomial order that satisfies the error for the given threshold. Based on Rubio et al. [44], it can be proved that the locally generated truncation error follows an exponential law

$$\|\tau_k^N\| \leq \sum_{i=1}^{N_{dim}} C_{ik} \exp\left(-\eta_{ik} N_{ik}\right), \tag{39}$$

where N_{dim} is the number of spatial dimensions of the problem while C_k and η_k are constants that depend on the smoothness of the function (Canuto et al. [11], Hesthaven and Warburton [28]).

It is important to notice that, for two dimensional problems with isotropic solutions and assuming without loss of generality that $N_x > N_y$, Eq. (39) can be approximated by

$$\|\tau_k^N\| \lesssim C_{2k} \exp\left(-\eta_{2k} N_{2k}\right), \tag{40}$$

where the sub-index 2 denotes the y-direction. Taking logarithms in Eq. (40), the following asymptotic behavior may be inferred (for the element k):

$$\log(\|\tau_k^N\|_{L_\infty}) \approx C_{2k} - \eta_{2k} N_{2k}. \tag{41}$$

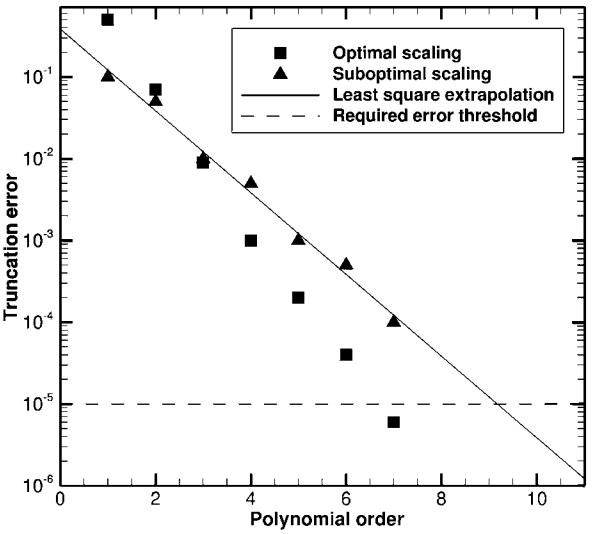


Fig. 1. Example of the extrapolation process within the adaptation procedure, to determine the polynomial order based on the truncation error: optimal scaling (squares) and suboptimal scaling (triangles). Dashed horizontal line: truncation error threshold and black line: least square extrapolation.

The two constants, C_{2k} and η_{2k}, depend on the element and can be easily approximated by least squares fitting in each spatial dimension.

For anisotropic problems Eq. (40) is only valid for $N_x >> N_y$ (i.e. assuming that N_x is the spatial dimension where the highest flow complexity, such as large flow gradient, are present). For very anisotropic problems, even with $N_x > N_y$ the truncation error may have contributions of both spatial dimensions. Therefore only the values of the truncation error where $N_x >> N_y$ should be used for the least square fitting.

The selection of optimal or suboptimal scaling is described in part B of Algorithm 1 and determines the appropriate polynomial order based on the estimation or the extrapolation. In particular, the desired truncation threshold, τ_{max}, is set before starting the adaptation procedure.

Finally in part C, the simulation is continued using the locally p-adapted mesh (i.e. adaptation is performed for each element and each direction). Note that the obtained converged solution on the non-adapted mesh can be interpolated to the p-adapted mesh and used as initial condition. In addition, let us note that besides the beneficial effect of an increase of the polynomial order in terms of accuracy, also a decrease of polynomial order in certain areas can be expected, which decreases the degrees of freedom and reduces the computational cost.

5.2. Quasi-a priori and quasi-a priori corrected τ-estimation

The adaptation process for the non-converged quasi-*a priori* τ-estimation, described in Algorithm 2, is similar to the *a posteriori* τ-estimation. The main difference with respect to the previous algorithm is that the solution used for the estimation is only partially converged in time. In this work, the infinite norm of the residual (tolerance) is chosen as the criterion to stop the computation. According to Eq. (35), the difference between the estimated and the exact truncation error is caused by two different sources. The first term is proportional to the discretization error in a finer mesh ϵ^P, which is negligible due to the spectral convergence of the method if $P > N$. The second term is proportional to the square of the iteration error (if the correction is applied) or to the iteration error (if no correction is applied). However, for smooth solutions, the iteration error is proportional to the residual, Eq. (36), which is the typical parameter monitored in the convergence process. With these considerations in mind and once the desired maximum error threshold is defined (τ_{max}), the following criteria can be used:

1. Quasi-*a priori*: If no correction term is considered (to reduce the computational cost), the maximum value of the residual, is chosen such that the solution converges until $tolerance < \tau_{max}/F$, namely, the residual is an order of magnitude lower than the desired maximum threshold. A typical value for F is 10. This guarantees that Eq. (35) holds when retaining first order terms and then Eq. (31) provides an accurate estimation of the truncation error.
2. Quasi-*a priori corrected*: If the correction term is applied, Eq. (35) applies with $\mathcal{O}\left(\epsilon_{it}^P\right)^2$. Therefore, the value of the residual can be relaxed to $tolerance < (\tau_{max}/F)^{1/2}$.

Once the tolerance is reached, the truncation error is estimated and the appropriate polynomial order is chosen in part B of the algorithm, as detailed in the previous section. In the final step the non-converged solution is interpolated onto the p-adapted mesh and the simulation is continued.

Part A – Estimation
while $\|\mathcal{R}(\tilde{u})\|_{L_\infty} > tolerance$ **do**
  Integrate in time in the fine mesh P (with a high polynomial order);
end
if *Correction* **then**
  Calculate $\epsilon_{it}^P \approx (\frac{\partial \mathcal{R}^P}{\partial u^P}|_{u^P})^{-1}\mathcal{R}^P(\tilde{u}^P)$
end
for $N < P$ **do**
  if *Correction* **then**
    Estimate non-converged truncation error τ_p^N *with correction* term Eq. (31);
  else
    Estimate non-converged truncation error τ_p^N *without correction*;
  end
end
Part B – Adaptation
for *each spatial direction* $d \leq N_{dim}$ **do**
  Set $P_{new}^d = 0$
  for $N = 1, P-1$ **do**
    if $\|\tau_P^N\|_{L_\infty} \leq required\ threshold(\tau_{max})$ **then**
      $P_{new}^d = N$
    end
  end
  if $P_{new}^d = 0$ **then**
    Determine interpolation parameters η and C using least squares,
    Calculate P_{new}^d using Eq. (41): $P_{new}^d = \frac{C - \log(required\ threshold(\tau_{max}))}{\eta}$
  end
end
Part C – Simulation
Interpolate non-converged solution to new p-adapted mesh;
Continue the simulation;

Algorithm 2: Quasi-*a priori* τ-estimation adaptation for each element.

Let us note that when anisotropic p-adaptation is considered, then Algorithms 1 and 2 are performed for each cell in the computational domain. The principal directions in the computational domain correspond to the physical x and y-directions only when the mesh is aligned with the Cartesian coordinate system.

5.3. Note on the selection of the τ_{max} threshold

The τ_{max} threshold controls the level of refinement or coarsening for all elements in the adapted mesh and hence determines the accuracy of the final solution. Here, we introduce two possibilities for its selection.

First, Eq. (25) in section 3.3 introduced an explicit relation between the truncation and the discretization error. This inequality may be used to estimate the value of τ_{max} based on a maximum allowable value for the discretization error. Alternatively, one could use Eq. (30), to estimate τ_{max} based on a particular output functional error. Although these relations can be used in simple computations, let us note that both Eq. (25) and Eq. (30) have denominators that may be difficult or computationally expensive to evaluate (e.g. Jacobian condition number), hence making the estimation of τ_{max} through these relations impractical.

A second option to estimate τ_{max} is to consider instead the maximum number of degrees of freedom DoF_{max} associated to the τ_{max} threshold. In computational physics a common limitation is the maximum allowable number of DoF for a given simulation (e.g. due to memory or computational time constraints). An alternative to setting τ_{max}, is to set a maximum number of DoF (DoF_{max}) for a given simulation. As shown, the truncation error controls both discretization and functional errors, and hence an adaptation based on the truncation error approach, given maximum number of DoF, may be argued to be optimal. Optimality is argued on the basis of minimizing all errors (and not a particular functional).

If the degrees of freedom DoF_{max} are selected as an alternative to τ_{max}, then, after part A and part B of Algorithms 1 or 2 are completed, the τ_{max} (or required threshold) is set such that the DoF of the adapted mesh are below the threshold for the maximum number of degrees of freedom DoF_{max}. The resulting mesh has an optimum distribution of DoF. Besides, if control of the error is required, an approximation of the discretization error, or output functional error, can be computed *a posteriori* using Eq. (22), or Eq. (25) and once part C of algorithms is completed. If the obtained solution does not reach the required error, a lower value of τ_{max} can then be selected.

Henceforth and for the sake of simplicity, the value of τ_{max} will be assumed as an input for the algorithms.

6. Numerical experiments

In order to check the accuracy and efficiency of the described methodology, two test cases are presented. First, the truncation error is estimated for a manufactured solution problem, where an exact solution is available. The estimated error and the efficiency of the different strategies for adaptation (*a posteriori*, quasi-*a priori* and quasi-*a priori corrected* approaches)

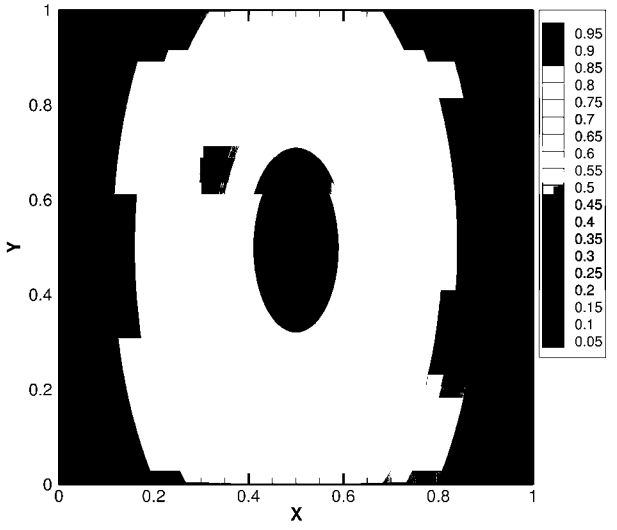


Fig. 2. The function $\rho(x, y) = p(x, y) = e^{-5(4(x-0.5)^2+(y-0.5)^2)}$ is used for the source term of the manufactured solution.

are analyzed. Second, a boundary layer (or flat plate) problem is used to show the properties of the error estimation for test cases with a higher number of degrees of freedom and to show the potential of the anisotropic p-adaptation.

In both test cases and unless otherwise specified, the desired threshold truncation error is chosen to be $\tau_{max} = 10^{-5}$. A solution is assumed to be converged when the residual of the solution (*tolerance*) is below 10^{-10}. For non-converged solutions, a value of *tolerance* $= 10^{-6}$ is chosen, if no correction term is applied, and 10^{-3} for the cases with correction.

6.1. Error estimation

6.1.1. Euler equations: manufactured solution test case

The manufactured solution technique can be used to determine the ability of an error estimation method, see Roy et al. [42]. This technique requires forcing terms to drive the differential partial equations to a predetermined solution. These source terms are incorporated in the code, then the modified governing equations (including the source terms) are discretized and solved numerically and compared to the exact solution.

Based on Roy [42], the six steps to implement the method of manufactured solutions are: 1. Choose the form of the governing equations; 2. Choose the form of the manufactured solution; 3. Apply the governing equations to the manufactured solution to generate analytical source terms; 4. Discretize the equations using analytical boundary conditions and source terms from the manufactured solution; 5. Evaluate the truncation error in the numerical solution and 6. Determine whether the observed order of accuracy matches the formal order of accuracy.

In our case, the previous steps are applied to the 2D inviscid Euler equations

$$\mathbf{Q}_t + \mathbf{F}^a_x + \mathbf{G}^a_y = \boldsymbol{\varphi}(x, y), \tag{42}$$

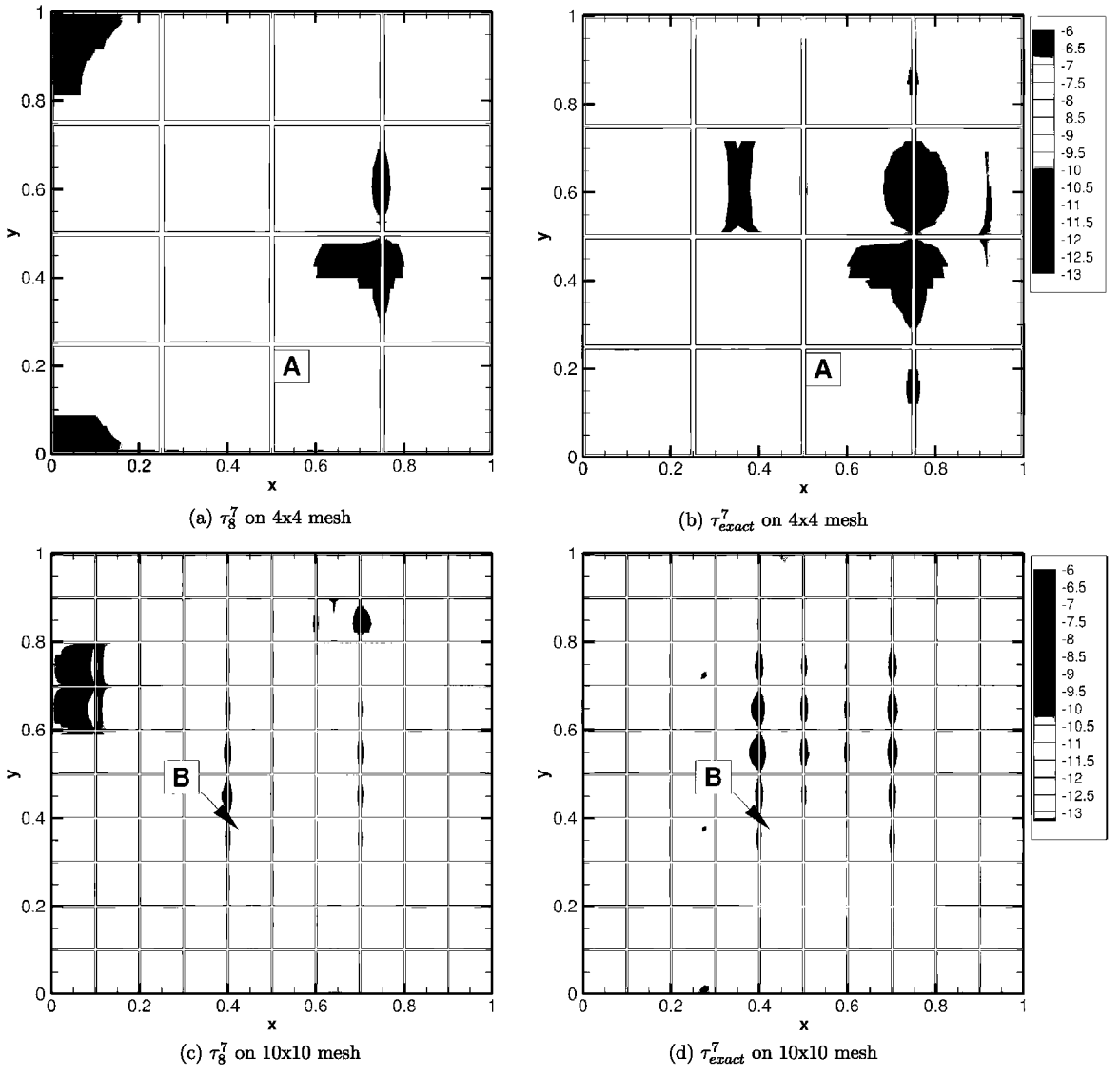
where $\mathbf{Q}$ is $(\rho, \rho u, \rho v, \rho e)^T$ and ρ, u, v, e denote the density, velocity components and energy. The inviscid flux vectors $\mathbf{F}^a$ and $\mathbf{G}^a$ are

$$\mathbf{F}^a = \begin{bmatrix} \rho u \\ p + \rho u^2 \\ \rho uv \\ u(\rho e + p) \end{bmatrix}, \mathbf{G}^a = \begin{bmatrix} \rho v \\ \rho uv \\ p + \rho^2 \\ v(\rho e + p) \end{bmatrix}, \tag{43}$$

where the pressure p is assumed to follow an ideal gas equation. For the imposed source term the horizontal and vertical velocities are set constant ($u = v = 1$), while the pressure and density distribution are chosen to follow an exponential distribution:

$$\rho(x, y) = p(x, y) = e^{-5(4(x-0.5)^2+(y-0.5)^2)}. \tag{44}$$

This function is steep with large gradients in the x-direction while it is relatively flat in the y-direction, see Fig. 2. This shape is used to test the error estimation as it is expected that a denser refinement in x-direction is needed while a less stringent refinement is required in the y-direction. The test case is performed with a Mach number of $M = 0.8$.



(a) τ_8^7 on 4x4 mesh (b) τ_{exact}^7 on 4x4 mesh

(c) τ_8^7 on 10x10 mesh (d) τ_{exact}^7 on 10x10 mesh

Fig. 3. The truncation error estimation τ_8^7 (left) and the exact truncation error τ_{exact}^7 (right) on a 4×4 mesh (top) and 10×10 mesh (bottom) for polynomial order $N_x = N_y = 7$. Colored contours show logarithmic values for the errors.

Estimated error vs. exact error We firstly compute the estimation of the truncation error using the *a-posteriori* approach. As already mentioned, the manufactured solution test case is converged when the residual infinite norm falls below the prescribed tolerance: $\|\mathcal{R}^N(u^N)\|_{L_\infty} < tolerance = 10^{-10}$. The calculation is obtained with a polynomial order of $P_x = P_y = 8$ on each element. Based on this calculation, and using Eq. (33) the truncation error is estimated for all combinations of the polynomial orders $N_x = 1, \ldots, 7$ and $N_y = 1, \ldots, 7$. Note that the DGSEM method enables different polynomial orders to be used in each spatial direction.

In Fig. 3, the results obtained for the truncation estimates are depicted for a 4×4 (top) and 10×10 (bottom) grid. The left side of the picture shows the estimated map τ_8^7 and the right side shows the exact truncation error τ_{exact}^7 (i.e. the exact solution interpolated using a uniform polynomial order 7). It can be seen that the estimate agrees very well with the exact solution for both meshes and that the maximum relative error: $\|\tau_8^7 - \tau_{exact}^7\|_{L_\infty} / \|\tau_{exact}^7\|_{L_\infty}$, is below 0.106 for the 4×4 mesh and below $5.399 \cdot 10^{-2}$ for the 10×10 mesh. In addition, we select two individual elements for each grid, A and B, to explore how the polynomial order influences the error. Fig. 4 shows the errors for Element A of the 4×4 mesh for different polynomial orders. As expected, the error decreases for higher orders. Furthermore, the distribution is not symmetric along the diagonal (i.e. $N_x = N_y$). This could have been predicted since the truncation error is stretched in the x-direction and thus a more stringent refinement is needed in this direction when compared to the y-direction.

The same tendency can be observed for the element B of the 10×10 mesh, Fig. 5. In this case, the error decreases significantly faster, with a difference of 8 orders of magnitude, when using polynomial orders N_x, N_y ranging from 1 to 7.

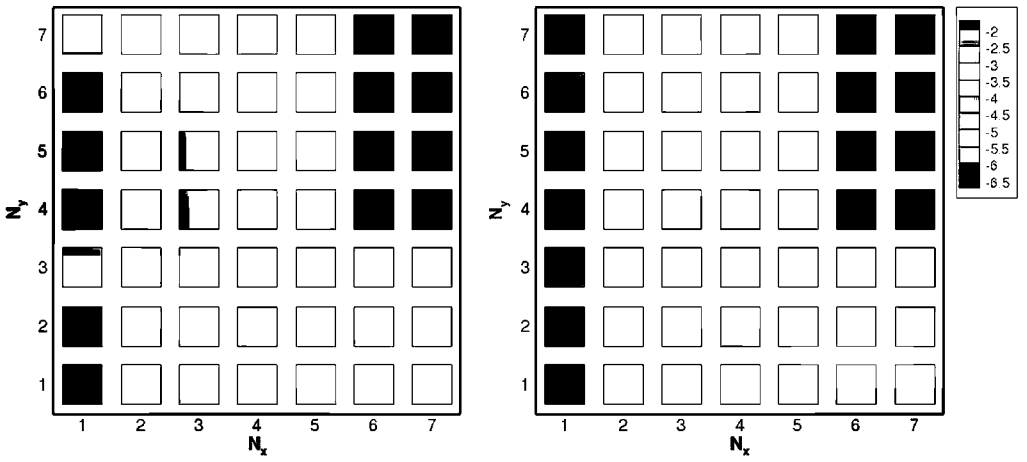


Fig. 4. Scatter plot (logarithmic scale) for the truncation error τ_8^N for varying polynomial orders N_x, N_y in the 4×4 grid and the manufactured solution case, only showing element A: left shows the estimated error and right shows the exact error.

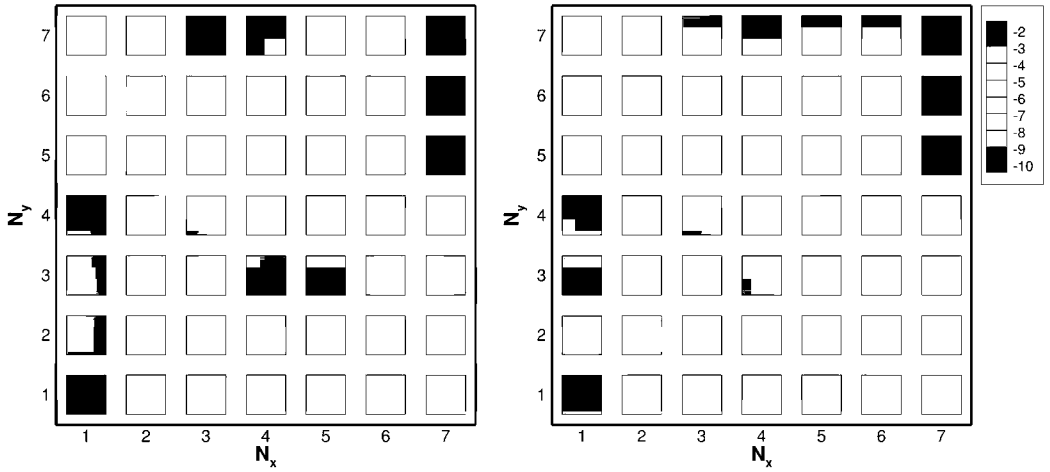


Fig. 5. Scatter plot (logarithmic scale) for the truncation error τ_8^N for varying polynomial orders N_x, N_y in the 10×10 grid and the manufactured solution case, only showing element B: left shows the estimated error and right shows the exact error.

Finally, the truncation error dependence with the polynomial order is shown in Fig. 6. The polynomial order in the y-direction is fixed to $N_y = 7$ and only the error related to N_x is considered. In both cases the estimates are in the asymptotic range, but due to a higher density of interpolation points, the error for the 10×10 mesh (right side) is lower for the same polynomial when compared to the 4×4 mesh (left side). The estimates are validated using the exact solution, Eq. (44), gray line in Fig. 6, which shows a very good agreement with the estimation.

As described in the previous section, for a defined truncation error threshold, these curves are used to estimated the polynomial order needed to obtain the desired accuracy.

A posteriori and quasi-a priori estimates In this section, the previously computed *a posteriori* approach is compared to the quasi-*a priori* one. Fig. 7 shows the truncation error on one particular element for the manufactured solution test case (10×10 mesh). For this plot, the truncation error was estimated in the *a posteriori* approach after the simulation was converged until a tolerance of 10^{-10}. It can be seen that the truncation error for this simulation is of the same order of magnitude and that it decreases asymptotically with high polynomial orders. The quasi-*a priori* approach was converged until 10^{-3}, while the truncation error was estimated once with and another time without applying the correction. Considering the estimations without the correction term, the error is stagnating around 10^{-3} while the estimations with the correction term flattens out around 10^{-6} to 10^{-7}. Indeed, applying the correction term will cancel out the first order iteration error highlighted in Eq. (35).

6.1.2. Navier–Stokes equations: boundary layer test case

The previous section proved the favorable properties of the truncation error. In particular, it was shown that the error estimated agrees well with the exact error.

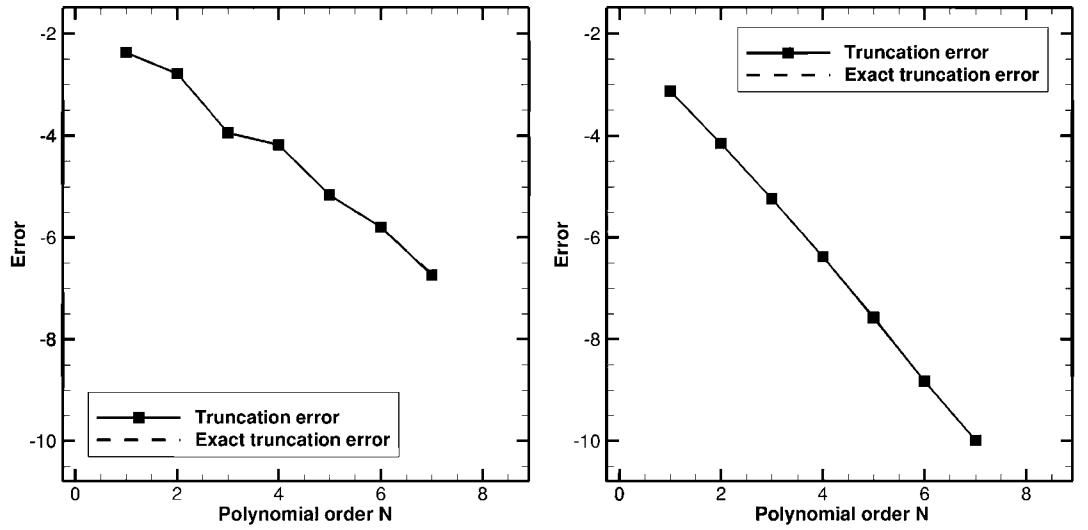


Fig. 6. Logarithm of the estimated and exact truncation errors as a function of the polynomial order N for $N = 1, \ldots, 7$. The estimation is performed with a fine polynomial order $P = 8$. Results are shown for the highlighted element A of the 4×4 mesh (left) and element B of the 10×10 mesh, see Fig. 3.

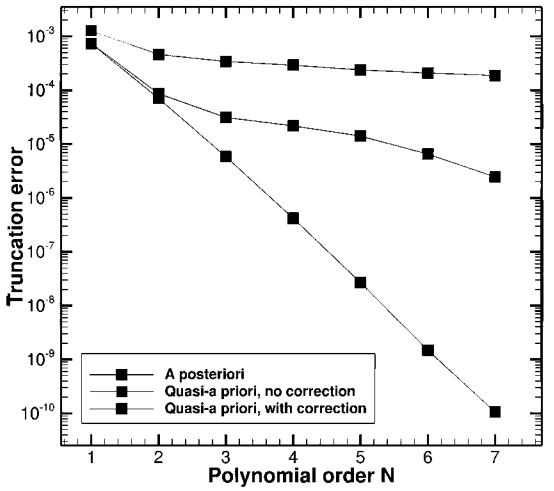


Fig. 7. Truncation error estimation τ_8^N for $N = 1, \ldots, 7$: *a posteriori*, quasi-*a priori* without correction and quasi-*a priori* with correction for the manufactured solution test case (10×10 mesh).

We now turn our attention to a more complex application with higher number of degrees of freedom. To simulate the viscous compressible Navier–Stokes equations, the DGSEM method is modified to account for viscous effects. To this end, we incorporate an Interior Penalty method as detailed in Arnold et al. [2]. For this purpose a boundary layer simulation is performed for Reynolds number per unit length $\mathrm{Re} = 500$ and Mach number $M = 0.2$. The governing equations are the viscous compressible Navier–Stokes equations, written in non-dimensional form:

$$\mathbf{Q}_t + \mathbf{F}_x^a + \mathbf{G}_y^a = \frac{1}{\mathrm{Re}}(\mathbf{F}_x^v + \mathbf{G}_y^v) \tag{45}$$

where $\mathbf{Q}$ is $(\rho, \rho u, \rho v, \rho e)^T$, $\mathbf{F}_x^a$ and $\mathbf{G}_y^a$ are the advective fluxes, defined as,

$$\mathbf{F}^a = \begin{bmatrix} \rho u \\ p + \rho u^2 \\ \rho uv \\ u(\rho e + p) \end{bmatrix}, \mathbf{G}^a = \begin{bmatrix} \rho v \\ \rho uv \\ p + \rho v^2 \\ v(\rho e + p) \end{bmatrix}. \tag{46}$$

The pressure p is defined through the ideal gas equation. On the other hand, the diffusive fluxes are defined as

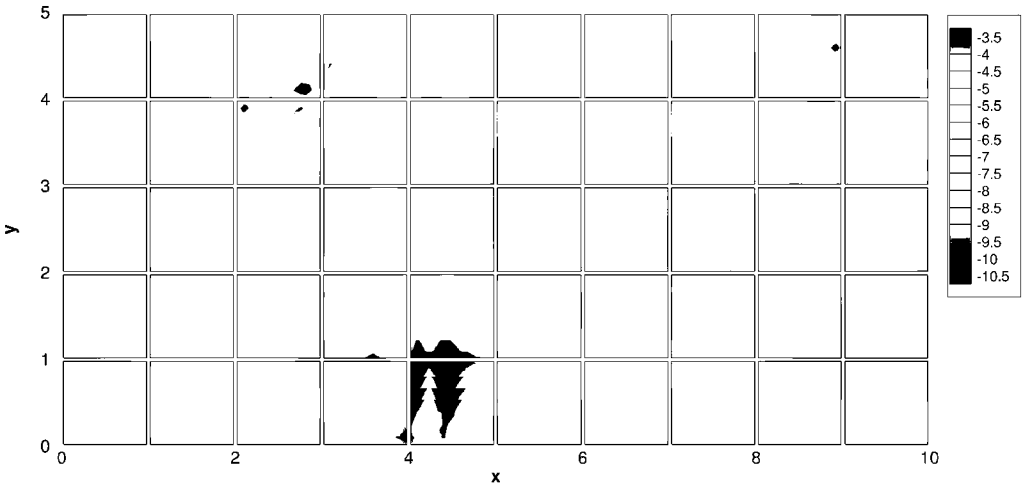


Fig. 8. Truncation error τ_8^7 based on the density (logarithmic scale), for boundary layer simulation, singularity at $x=4$.

$$\mathbf{F}^v = \begin{bmatrix} 0 \\ \tau_{xx} \\ \tau_{xy} \\ u\tau_{xx} + v\tau_{xy} + \frac{\kappa}{(\gamma-1)\,\mathrm{Pr}\,M^2} T_x \end{bmatrix}, \mathbf{G}^v = \begin{bmatrix} 0 \\ \tau_{yx} \\ \tau_{yy} \\ u\tau_{yx} + v\tau_{yy} + \frac{\kappa}{(\gamma-1)\,\mathrm{Pr}\,M^2} T_y \end{bmatrix}, \tag{47}$$

and the Stokes hypothesis

$$\begin{aligned} \tau_{xx} &= 2\mu(u_x - (u_x + v_y)/3), \\ \tau_{yy} &= 2\mu(v_y - (u_x + v_y)/3), \\ \tau_{xy} &= \tau_{yx} = \mu(v_x + u_y), \end{aligned} \tag{48}$$

where T the temperature, μ is the viscosity, γ is the heat capacity ratio, κ is the thermal diffusivity and the non-dimensional parameters Re the Reynolds number, Pr the Prandtl number and M the Mach number. The Prandtl number and the heat capacity ratio are set to the usual values for air $\mathrm{Pr}=0.72$ and $\gamma=1.4$. The viscosity, μ, and the thermal diffusivity, κ are calculated using Sutherland's law.

Uniform boundary conditions are used at the inflow, while at the bottom boundary, a symmetric boundary conditions is used up to the stagnation point located at $x=4$ and an adiabatic wall assumed further downstream. Pressure exit boundary conditions are applied for the outflow and the far field. A steady solution is calculated with $P_x = P_y = 8$ and converged to a residual of 10^{-10}. The estimation for the truncation error for τ_8^7 is shown in Fig. 8.

As expected, the truncation error is large around the singularity at $x=4$ and close to the wall in the downstream region. Certainly, this is the region that contains complex flow features and consequently requires higher resolution e.g. needs to be adapted using a higher polynomial order. Furthermore, the error is slightly higher at the inlet than in the outer region due to the imposed inflow boundary conditions and the interaction with the singularity which could be avoided by extending the distance between the inflow and the begin of the boundary layer.

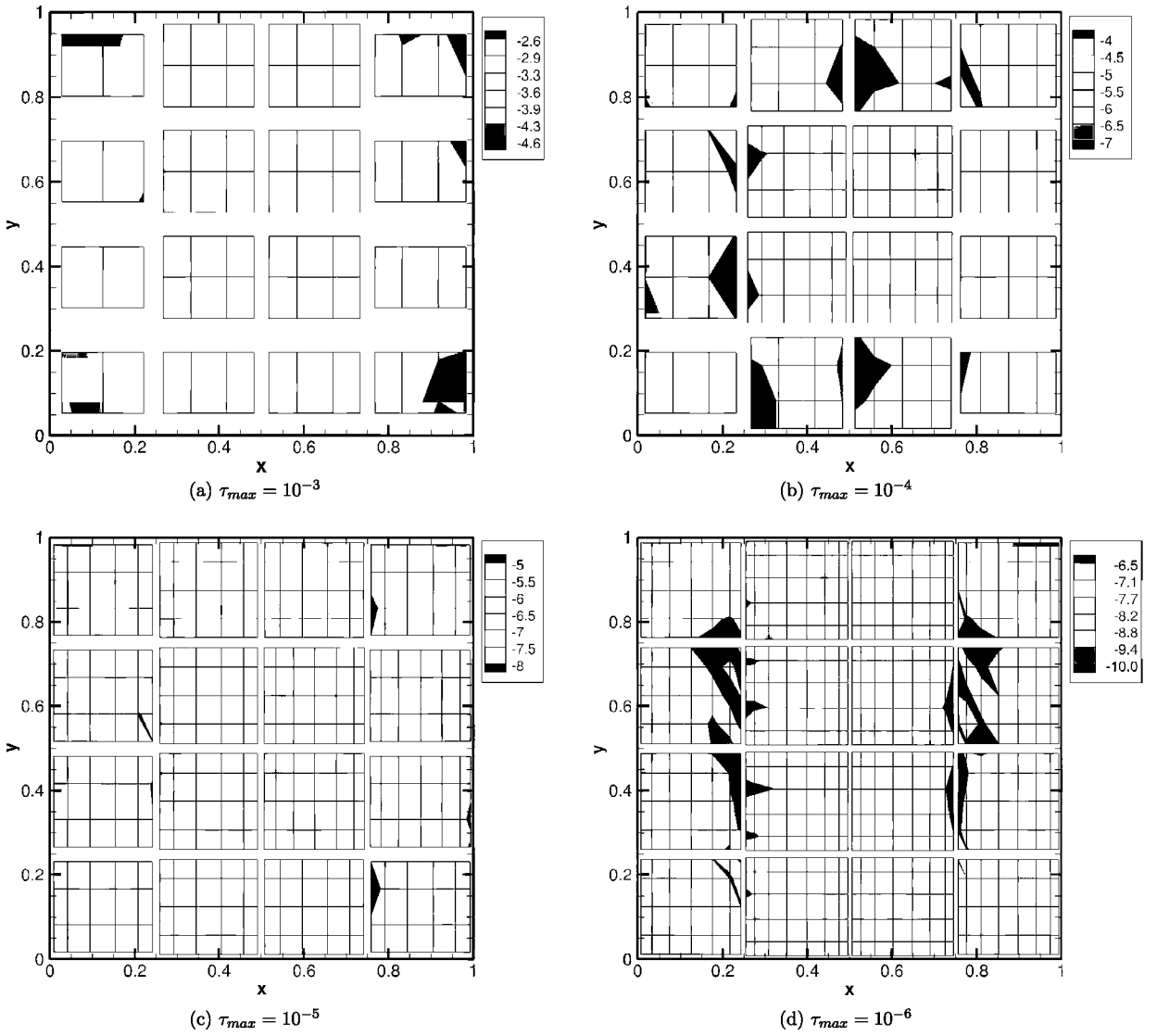
6.2. Adaptation process

Having demonstrated the validity of the error estimation, let us now test our adaptation algorithms.

6.2.1. Euler equations: manufactured solution test case

Once the truncation error has been obtained, we follow the *a posteriori* Algorithm 1, described in section 5, to adapt the mesh for different values of the truncation error threshold τ_{max}. The required polynomial order in the x and y directions is obtained through interpolation (optimal scaling) or extrapolation (suboptimal scaling) of the truncation error estimates curves computed at each element (see the example in Fig. 4).

Fig. 9 shows the p-adapted meshes based on different threshold levels for the required truncation error. After the adaptation, the error is checked with the exact truncation error. The final error from the adapted mesh always achieves the required order of accuracy. Let us note the p-adapted mesh is finer in the x-direction than in the y-direction, which shows the potential of the method for anisotropic refinements. Note that the figure only shows interior element nodes. In the DGSEM formulation, these correspond to Legendre–Gauss nodes. Hence for a polynomial of order P, we show $P+1$ nodes per direction that do not include the edges of the element.



(a) $\tau_{max} = 10^{-3}$ (b) $\tau_{max} = 10^{-4}$

(c) $\tau_{max} = 10^{-5}$ (d) $\tau_{max} = 10^{-6}$

Fig. 9. Adaptation results for different required truncation errors (logarithmic scale); the computation Legendre–Gauss nodes are shown in each element.

6.2.2. Navier–Stokes equations: boundary layer test case

The adaptation process is applied to the boundary layer test case using a mesh of 5×10 elements and an initial polynomial order of $P_x = P_y = P = 8$. Now, only the quasi-*a priori* adaptation process without the correction term is detailed since no significant differences are observed when adding the correction. The main differences appear in the computational costs, which will be explained later in section 7.

We first explore the effect of varying the truncation error threshold (τ_{max}). In Fig. 10 we show three meshes where we have varied the threshold levels: Fig. 10 (top-left) $\tau_{max} = 10^{-1}$, Fig. 10 (top-right) $\tau_{max} = 10^{-3}$ and Fig. 10 (bottom) $\tau_{max} = 10^{-2}$. Comparing these meshes it can be seen that decreasing the threshold level results in finer meshes. More interesting is the clear anisotropic refinement enabled by the high order discontinuous Galerkin discretization as shown in Fig. 10 (bottom). Note that when selecting $\tau_{max} = 10^{-3}$, Fig. 10 (top-right), the maximum allowable polynomial $P = 10$ is reached and hence the anisotropic refinement is masked.

Having shown the effect of varying the truncation error threshold, we retain the threshold of $\tau_{max} = 10^{-2}$, mesh depicted in Fig. 10 (bottom). This threshold implies that the convergence is stopped when the simulation reaches a residual of 10^{-3} (i.e. $F = 10$ in section 5.2).

Following Algorithm 2, the estimates are calculated for $N_x, N_y = 1, \ldots, 7$. This estimation is shown in Fig. 11 for three particular elements A, B and C, depicted in Fig. 10 (bottom). The first plot Fig. 11 (left) shows the adaptation for element A, which is located far from the leading edge singularity. In this case the estimates for the different polynomial orders show that a polynomial order of 1 suffices to reach the desired threshold $\tau_{max} = 10^{-2}$. Fig. 11 (right), considers Element C (located

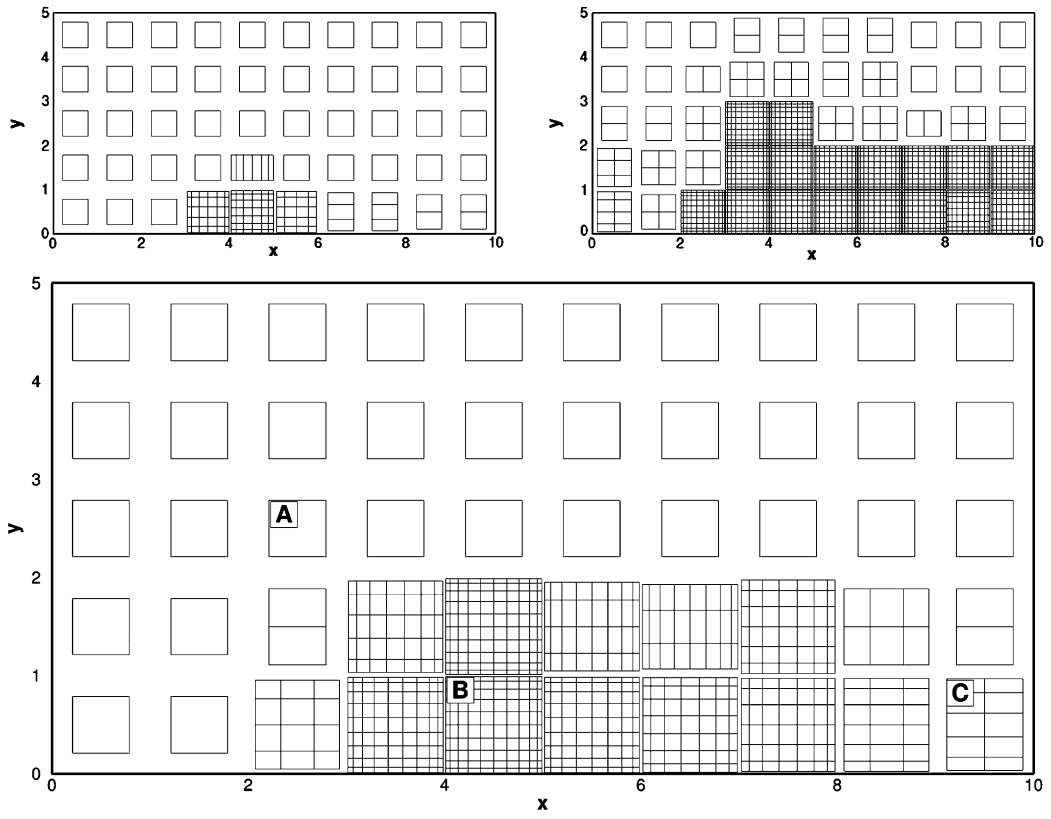


Fig. 10. Adapted boundary layer meshes based on various truncation error thresholds: 10^{-1} (upper-left), 10^{-2} (lower-left) and 10^{-3} (upper-right). The computation Legendre–Gauss nodes are shown in each element.

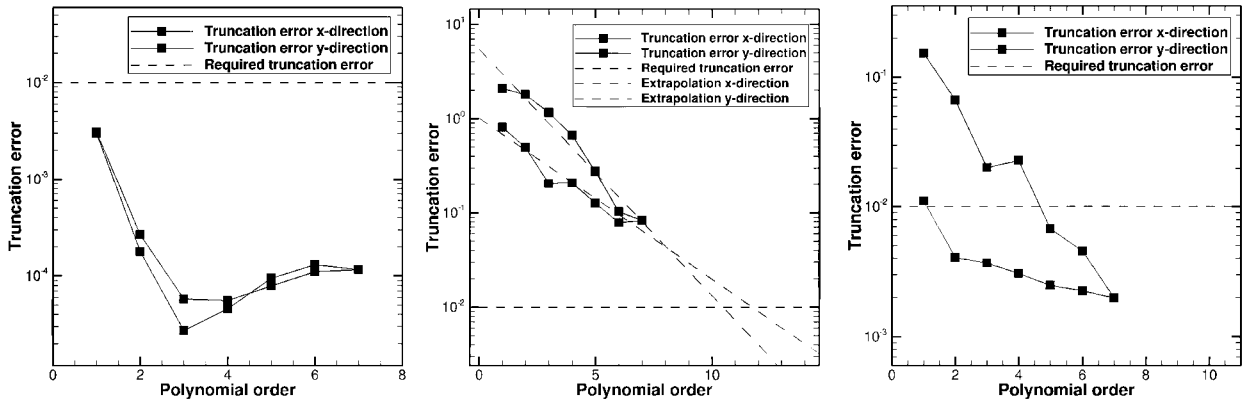


Fig. 11. Error estimation and required error for elements A (left), B (middle) and C (right) of Fig. 10, linear error extrapolation is used on elements B and C. Horizontal dashed line shows the 10^{-3} threshold.

near the outflow and the wall), these error estimates show the anisotropic character expected form a mesh element located near the wall, where large gradient in the wall-normal (y-direction) dominate. In this case the polynomial order that fulfills the truncation threshold is $P_x = 2$ and $P_y = 6$. Depending on the test case it may happen that the estimates of certain elements do not behave accordingly to the asymptotic range. This is the case for element B (located near the boundary layer singularity or leading point). Fig. 10 shows that this element has a high error due to its location very close to the leading edge singularity. To resolve this type of elements we follow the extrapolation procedure as depicted in Fig. 11 (center). If the extrapolated value provides a polynomial order that is below the maximum allowable polynomial then the extrapolated is selected (this is the case of the red curve or y-direction). However, if the extrapolation shows that a polynomial higher than possible (above the allowable maximum) is needed, the polynomial order is set to the maximum in the corresponding direction for this element ($P = 10$ in this case). Another possibility would be to recalculate the entire underlying simulation with a higher polynomial order or to refine the area around the singularity with smaller elements (h-refinement).

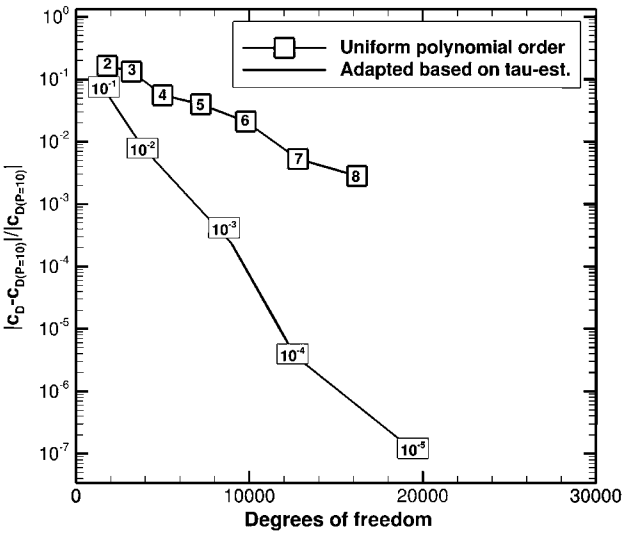


Fig. 12. c_D error based on different simulations on the corresponding DoF; uniform polynomial order solutions (blue with polynomial order in the box) and solutions obtained by the τ-truncation error adaptation process (red with the adaptation criteria τ_{max} in the box); Plotted value $\frac{|c_D - c_{D(P=10)}|}{|c_{D(P=10)}|}$, where $c_{D(P=10)}$ is calculated on a uniformly refined mesh with $P = 10$. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Finally and for completeness, we show in Fig. 12 the relative error for the drag coefficient, $|c_D - c_{D(P=10)}|/|c_{D(P=10)}|$ for the boundary layer case. It can be seen that when decreasing the truncation error thresholds (values within the boxes for the red line) the relative error for the drag decreases. In addition, we include drag-errors issued from non-adapted meshes where a uniform polynomial order is used (blue line and boxes show the polynomial order). When comparing the drag-errors issued from the p-adapted meshes to the errors obtained using uniform polynomial, it can be seen that for the same number of degrees of freedoms, the error is lower if the anisotropic adaptation based on tau-estimates is used. These last results show that, at least for the boundary layer test case, the accuracy of the drag is governed by the truncation error.

7. Computational cost

In this section, the computational cost of the different methods is compared for the boundary layer test case. A truncation error threshold of $\tau_{max} < 10^{-2}$ is defined as the objective. As previously explained, the *a posteriori* method converges the solution until the maximum residual (L_∞ norm) is below 10^{-10} (i.e. *pre-adaptation step*), then a new p-adapted mesh is computed following Algorithm 1. The solution is subsequently interpolated from the original to the new p-adapted mesh and converged again (i.e. *post-adaptation step*). In this section, we test the *a posteriori* method for a polynomial $P = 8$. For the quasi-*a priori* method, we include results following the two approaches with and without correction. If no correction is applied, the solution is converged until a tolerance of 10^{-3} is reached (*pre-adaptation step*), using this solution the truncation error is estimated and a new p-adapted mesh is obtained and converged until 10^{-10} (*post-adaptation step*). Obviously this solution is computationally more efficient than the *a posteriori* method, since the initial solution is relaxed to a lower tolerance and the rest of the algorithm is equivalent. Finally the tolerance is set to $10^{-1.5}$, which clearly shortens the computational time, and the correction term is applied afterwards, this is the quasi-*a priori corrected* method. However, it must be bared in mind that the calculation of the correction and the accurate estimation of the truncation error is expensive (i.e. requires the solution of a linear system). The method can, nonetheless, be computationally efficient when considering the overall computational cost, depending on the time required in each step and the time needed to compute the correction factor. In our cases, the Jacobian, $\left.\frac{\partial \mathcal{R}^P}{\partial u^P}\right|_{u^P}$, is stored in sparse format and the solution of the linear system, required by the correction step, is provided by a GMRES iterative solver with block Jacobi preconditioning. Our implementation relies on the PETSc libraries, Balay et al. [4], to solve the linear problem in the quasi-*a priori corrected* method. As a final step, the transfer operator, $\left.\frac{\partial \mathcal{R}^N}{\partial u^N}\right|_{u^N} I_P^N$, for the iteration error from the fine to the coarse mesh is calculated.

For reference, the adaptation strategies are compared to the computational time of a non-adapted solution with a constant polynomial order $P = 10$. This polynomial is chosen such that it equals the maximum polynomial that is used in the adaptation algorithm to obtain the required truncation error threshold of $\tau_{max} < 10^{-2}$.

Table 1 summarizes the run times of the *a posteriori* and the two quasi-*a priori* approaches (with and without correction). These computational costs are relative to the calculation with a homogeneous polynomial $P = 10$. It can be seen that the *a posteriori* achieves a speedup of 2.26 times which is lower than the speedup of the quasi-*a priori* method without correction (speedup of 6.59). The most important time gains are provided by the quasi-*a priori corrected* method, which provides a speedup of 7.61.

Table 1
Runtime and speedup of the *a posteriori* and quasi-*a priori* adaptation approaches, non-dimensionalized with respect to a homogeneous polynomial $P = 10$ (no adaptation), time convergence until $\|\mathcal{R}(\tilde{u})\|_{L_\infty} < 10^{-10}$.

	Runtime (% w.r.t. $P = 10$)	Speedup (w.r.t. $P = 10$)
0) homogeneous $P = 10$	100	0
I) *a posteriori* $P = 8$	44.13	2.26
II) quasi-*a priori*	15.18	6.59
III) quasi-*a priori corrected*	13.14	7.61

Table 2
Computational cost of *a posteriori* and quasi-*a priori* adaptation algorithms non-dimensionalized with respect to a homogeneous polynomial $P = 10$ (no adaptation).

	Pre adaptation (%)	Adaptation (%)	Post adaptation (%)
I) *a posteriori* $P = 8$	38.618	0.0166	5.492
II) quasi-*a priori*	9.104	0.0182	6.059
III) quasi-*a priori corrected*	2.830	3.5895	6.724

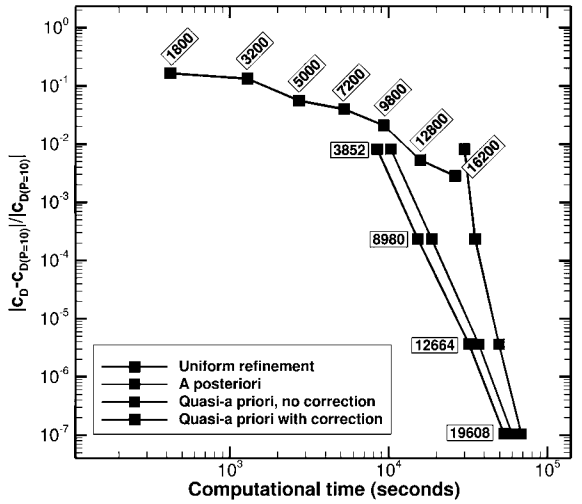


Fig. 13. Relative error in drag coefficient $\frac{|c_D - c_{D(P=10)}|}{|c_{D(P=10)}|}$ and computational time (in seconds). The reference drag $c_{D(P=10)}$ is calculated on a uniformly refined mesh with $P = 10$. The blue line shows uniform polynomial order (non-adapted meshes). Adapted simulations include: *a posteriori* approach (orange), quasi-*a priori* approach without correction (green) and quasi-*a priori corrected* (red). The numbers of DoF for each simulation are shown in the boxes. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Table 2 details the relative amount of time spent by the methods in each part of the algorithm (i.e. *pre-adaptation*, *adaptation* and *post-adaptation*) which are non-dimensionalized with respect to the total time of the simulation with homogeneous polynomial $P = 10$ (no adaptation). The main differences between the three adaptation strategies can be seen for the *pre-adaptation* times. Indeed, the quasi-*a priori* approach requires longer *pre-adaptation* simulations than the quasi-*a priori corrected* method ($9.1/2.8 \sim 3.3$ times longer). Even though the time to calculate the correction in the adaptation part of the quasi-*a priori corrected* method is not negligible 3.6, the overall time remains lower (13.14 of the reference computation as shown in Table 1) making this adaptation strategy the most efficient. In addition, it can be seen that the *post-adaptation* times are similar in the two quasi-*a priori* approaches. Finally, let us note that the three anisotropic p-adaptation strategies lead to identical adapted meshes as depicted in the previous section Fig. 10 (bottom).

We can conclude that the three adaptation strategies reduce the computational cost significantly when compared to a simulation with the same accuracy and where the polynomial order is fixed everywhere in the domain. In addition, the two quasi-*a priori* approaches show significant time reductions, and in particular the quasi-*a priori corrected* algorithm enables significant speedups.

We have summarized the advantages of the adaptation process in Fig. 13, where the accuracy and computational cost are depicted for all the computed cases. These include the uniform polynomial meshes (non-adapted) and the adapted meshes using the three adaptation strategies. We also include in the figure the degrees of freedom (DoF) used for each simulation. The outperforming results of the adaptation strategies over the uniform polynomial are clear. Furthermore, the quasi-*a priori* methods (with and without correction) show cost improvements for the same accuracy over the *a posteriori* technique. Finally, the quasi-*a priori corrected* shows the best performance among all the proposed techniques.

8. Conclusions

Three novel and efficient anisotropic p-adaptation strategies have been presented in this paper. The truncation error estimation has been successfully used to select the elements and directions that require adaptation. Indeed, its direct relation to the numerical error makes it an excellent criterion for mesh refinement.

An *a posteriori* algorithm and two quasi-*a priori* approaches have been presented and have shown to provide faster converged solutions than when a uniformly high polynomial order is selected. The quasi-*a priori* techniques enable an accurate and reliable adaptation process based on not fully time-converged solutions. It has been shown that the incorporation of a correction term, to the non-converged solution, significantly improves the error estimate by canceling out the first order iteration error.

Remarkable savings in computational cost are achieved based on the proposed anisotropic adaptation algorithms. In particular, the calculations using the quasi-*a priori* approach with correction term enables a speedup of 7.6 when compared to the non-adapted solution.

Acknowledgements

The authors would like to thank the anonymous reviewers for their valuable comments and suggestions. The authors would like to acknowledge the European Commission for the financial support of the ANADE project (Advances in Numerical and Analytical tools for DEtached flow prediction) under grant contract PITN-GA-289428 and the NNATAC project (New Numerical and Analytical Tools for Aerodynamic flow Control) under grant agreement PIAP-GA-2012-324298. In addition, the authors would like to thank Professor David A. Kopriva for his technical advice and cooperation.

Appendix A

A.1. Proof of the quasi-a priori τ-estimation formula, Eq. (31)

Substituting the definitions of the iteration error, $\tilde{u}^P = u^P - \epsilon_{it}^P$, and the discretization error, $u^P = u - \epsilon^P$, onto the estimate of the truncation error Eq. (31) and, using Taylor series, we obtain

$$\tau_P^N = \mathcal{R}^N(u) - \left.\frac{\partial \mathcal{R}^N}{\partial u^N}\right|_{u^N} \epsilon^P - \left.\frac{\partial \mathcal{R}^N}{\partial u^N}\right|_{u^N} \epsilon_{it}^P - \bar{I}_P^N \mathcal{R}^P\left(\tilde{u}^P\right) + \mathcal{O}\left(\epsilon^P\right)^2 + \mathcal{O}\left(\epsilon_{it}^P\right)^2, \tag{49}$$

or equivalently

$$\tau_P^N = \tau^N - \left.\frac{\partial \mathcal{R}^N}{\partial u^N}\right|_{u^N} \epsilon^P - \left.\frac{\partial \mathcal{R}^N}{\partial u^N}\right|_{u^N} \epsilon_{it}^P - \bar{I}_P^N \mathcal{R}^P\left(\tilde{u}^P\right) + \mathcal{O}\left(\epsilon^P\right)^2 + \mathcal{O}\left(\epsilon_{it}^P\right)^2. \tag{50}$$

Using Eq. (18), we modify Eq. (50) to

$$\tau_P^N = \tau^N - \left.\frac{\partial \mathcal{R}^N}{\partial u^N}\right|_{u^N} \epsilon^P - \left.\frac{\partial \mathcal{R}^N}{\partial u^N}\right|_{u^N} \epsilon_{it}^P + \bar{I}_P^N \left.\frac{\partial \mathcal{R}^P}{\partial u^P}\right|_{u^P} \epsilon_{it}^P + \mathcal{O}\left(\epsilon^P\right)^2 + \mathcal{O}\left(\epsilon_{it}^P\right)^2. \tag{51}$$

Taking into account that, by definition $u - \tilde{u}^P = \epsilon^P + \epsilon_{it}^P$, it can be seen that for $\bar{I}_P^N = \left.\frac{\partial \mathcal{R}^N}{\partial u^N}\right|_{\tilde{u}^P} I_P^N \left(\left.\frac{\partial \mathcal{R}^P}{\partial u^P}\right|_{\tilde{u}^P}\right)^{-1}$ we have

$$\tau_P^N = \tau^N - \left.\frac{\partial \mathcal{R}^N}{\partial u^N}\right|_{u^N} \epsilon^P + \mathcal{O}\left(\epsilon^P\right)^2 + \mathcal{O}\left(\epsilon_{it}^P\right)^2. \tag{52}$$

Eq. (52) holds for nonlinear equations. However, it is also valid for linear equations taking into account that, for linear equations the Jacobian $\left.\frac{\partial \mathcal{R}^N}{\partial u^N}\right|_{u^N}$ is substituted by the homogeneous discrete partial differential operator $\tilde{\mathcal{R}}^N$ and that the Taylor expansions are exact taking only one term, such that $\left(\epsilon^P\right)^2 = 0$ and $\left(\epsilon_{it}^P\right)^2 = 0$.

References

[1] M. Ainsworth, A posteriori error estimation for discontinuous Galerkin finite element approximation, SIAM J. Numer. Anal. 45 (4) (June 2007) 1777–1798.

[2] D.N. Arnold, F. Brezzi, B. Cockburn, L.D. Marini, Unified analysis of discontinuous Galerkin methods for elliptic problems, SIAM J. Numer. Anal. 39 (5) (2001) 1749–1779.

[3] R. Balasubramanian, J.C. Newman, Adjoint-based error estimation and grid adaptation for functional outputs: application to two-dimensional, inviscid, incompressible flows, Comput. Fluids 38 (2) (2009) 320–332.

[4] Satish Balay, Shrirang Abhyankar, Mark F. Adams, Jed Brown, Peter Brune, Kris Buschelman, Lisandro Dalcin, Victor Eijkhout, William D. Gropp, Dinesh Kaushik, Matthew G. Knepley, Lois Curfman McInnes, Karl Rupp, Barry F. Smith, Stefano Zampini, Hong Zhang, PETSc users manual, Technical Report ANL-95/11 – Revision 3.6, Argonne National Laboratory, 2015.

[5] F. Bassi, A. Crivellini, D.A. Di Pietro, S. Rebay, An artificial compressibility flux for the discontinuous Galerkin solution of the incompressible Navier–Stokes equations, J. Comput. Phys. 218 (2) (2006) 794–815.
[6] F. Bassi, S. Rebay, A high-order accurate discontinuous finite element method for the numerical solution of the compressible Navier–Stokes equations, J. Comput. Phys. 131 (2) (1997).
[7] C.E. Baumann, An hp-adaptive discontinuous finite element method for computational fluid dynamics, PhD thesis, University of Texas at Austin, 1997.
[8] M.J. Berger, Adaptive Finite Difference Methods in Fluid Dynamics, Courant Institute of Mathematical Sciences, New York University, 1987.
[9] K. Bernert, τ-Extrapolation – theoretical foundation, numerical experiment, and application to Navier–Stokes equations, SIAM J. Sci. Comput. 18 (2) (1997) 460–478.
[10] A. Brandt, O.E. Livne, Multigrid Techniques, Society for Industrial and Applied Mathematics, 2011.
[11] C.G. Canuto, Y. Hussaini, A. Quarteroni, T.A. Zang, Spectral Methods: Fundamentals in Single Domains, Scientific Computation, Springer, 2010.
[12] B. Cockburn, G. Kanschat, D. Schötzau, An equal-order DG method for the incompressible Navier–Stokes equations, J. Sci. Comput. 40 (1) (2009) 188–210.
[13] B. Cockburn, C.W. Shu, The local discontinuous Galerkin method for time dependent convection–diffusion systems, SIAM J. Numer. Anal. 35 (1998) 2440–2463.
[14] J.M. Derlaga, T. Phillips, C.J. Roy, J. Borggaard, Adjoint and truncation error based adaptation for finite volume schemes with error estimates, in: 53rd AIAA Aerospace Sciences Meeting, AIAA SciTech, American Institute of Aeronautics and Astronautics, January 2015.
[15] R.P. Dwight, Heuristic a posteriori estimation of error due to dissipation in finite volume schemes and application to mesh adaptation, J. Comput. Phys. 227 (5) (February 2008) 2845–2863.
[16] E. Ferrer, A high order discontinuous Galerkin–Fourier incompressible 3D Navier–Stokes solver with rotating sliding meshes for simulating cross-flow turbines, PhD thesis, University of Oxford, 2012.
[17] E. Ferrer, D. Moxey, R.H.J. Willden, S. Sherwin, Stability of projection methods for incompressible flows using high order pressure-velocity pairs of same degree: continuous and discontinuous Galerkin formulations, Commun. Comput. Phys. 16 (3) (2014) 817–840.
[18] E. Ferrer, R.H.J. Willden, A high order discontinuous Galerkin finite element solver for the incompressible Navier–Stokes equations, Comput. Fluids 46 (1) (2011) 224–230.
[19] E. Ferrer, R.H.J. Willden, A high order discontinuous Galerkin–Fourier incompressible 3D Navier–Stokes solver with rotating sliding meshes, J. Comput. Phys. 231 (21) (2012) 7037–7056.
[20] F. Fraysse, J. de Vicente, E. Valero, The estimation of truncation error by τ-estimation revisited, J. Comput. Phys. 231 (9) (May 2012) 3457–3482.
[21] F. Fraysse, G. Rubio, J. de Vicente, E. Valero, Quasi-a priori mesh adaptation and extrapolation to higher order using tau-estimation, Aerosp. Sci. Technol. 38 (2014) 76–87.
[22] F. Fraysse, E. Valero, J. Ponsin, Comparison of mesh adaptation using the adjoint methodology and truncation error estimates, AIAA J. 50 (9) (2012) 1920–1932.
[23] F. Fraysse, E. Valero, G. Rubio, Quasi-a priori truncation error estimation and higher order extrapolation for non-linear partial differential equations, J. Comput. Phys. 253 (2013) 389–404.
[24] S.R. Fulton, On the accuracy of multigrid truncation error estimates, Electron. Trans. Numer. Anal. 15 (2003) 29–37.
[25] H. Gao, Z.J. Wang, A residual-based procedure for hp-adaptation on 2d hybrid meshes, AIAA Pap. 492 (2011).
[26] E.H. Georgoulis, Discontinuous Galerkin methods on shape-regular and anisotropic meshes, PhD thesis, University of Oxford, 2003, D. Phil. Thesis.
[27] R. Hartmann, Error estimation and adjoint-based adaptation in aerodynamics, in: P. Wesseling, E. Onate, J. Périaux (Eds.), Proceedings of the ECCOMAS CFD 2006, September 5–8, Egmond aan Zee, The Netherlands, 2006.
[28] J.S. Hesthaven, T. Warburton, Nodal Discontinuous Galerkin Methods: Algorithms, Analysis, and Applications, 1st edition, Springer, 2007.
[29] G. Karniadakis, S.J. Sherwin, Spectral/hp Element Methods for CFD, Numerical Mathematics and Scientific Computation, Oxford University Press, 1999.
[30] D.A. Kopriva, Implementing Spectral Methods for Partial Differential Equations: Algorithms for Scientists and Engineers, 1st edition, Springer, 2009.
[31] B. Landmann, M. Kessler, S. Wagner, E. Krämer, A parallel high-order discontinuous Galerkin code for laminar and turbulent flows, Comput. Fluids 37 (4) (2008) 427–438.
[32] C. Mavriplis, Adaptive mesh strategies for the spectral element method, Comput. Methods Appl. Mech. Eng. 116 (14) (1994) 77–86.
[33] N.C. Nguyen, P.O. Persson, J. Peraire, RANS solutions using high order discontinuous Galerkin methods, in: 45th AIAA Aerospace Science Meeting and Exhibit, Reno, Nevada, 2007.
[34] T.A. Oliver, D.L. Darmofal, An unsteady adaptation algorithm for discontinuous Galerkin discretizations of the RANS equations, in: 18th AIAA Computational Fluid Dynamics Conference, Reno, Nevada, 2007.
[35] J. Peiro, S.J. Sherwin, Finite difference, finite element and finite volume methods for partial differential equations, in: Sidney Yip (Ed.), Handbook of Materials Modeling, Springer, Netherlands, 2005, pp. 2415–2446.
[36] T. Phillips, Residual-based discretization error estimation for computational fluid dynamics, PhD thesis, Virginia Polytechnic Institute and State University, 2014.
[37] T. Phillips, C. Roy, Residual methods for discretization error estimation, in: 20th AIAA Computational Fluid Dynamics Conference, 2011.
[38] H. Reed, T.R. Hill, Triangular mesh methods for the neutron transport equation, Technical Report LA-UR-73-479, Los Alamos Scientific Laboratory, 1973.
[39] B. Riviere, Discontinuous Galerkin Methods for Solving Elliptic and Parabolic Equations: Theory and Implementation, Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2008.
[40] C.J. Roy, Strategies for driving mesh adaptation in CFD, AIAA 2009-1302, invited paper for session on Error Estimation and Control, in: 47th AIAA Aerospace Sciences Meeting, Orlando, Florida, January 5–8, 2009.
[41] C.J. Roy, Review of discretization error estimators in scientific computing, AIAA Pap. 126 (2010) 2010.
[42] C.J. Roy, T.M. Smith, C.C. Ober, Verification of a compressible CFD code using the method of manufactured solutions, in: 32nd AIAA Fluid Dynamics Conference, 2002.
[43] G. Rubio, F. Fraysse, J. de Vicente, E. Valero, The estimation of truncation error by τ-estimation for Chebyshev spectral collocation method, J. Sci. Comput. 57 (1) (2013) 146–173.
[44] G. Rubio, F. Fraysse, D.A. Kopriva, E. Valero, Quasi-a priori truncation error estimation in the DGSEM, J. Sci. Comput. (2014) 1–31.
[45] K. Shahbazi, P.F. Fischer, C.R. Ethier, A high-order discontinuous Galerkin method for the unsteady incompressible Navier–Stokes equations, J. Comput. Phys. 222 (1) (2007) 391–407.
[46] T.I.P. Shih, B.R. Williams, Development and evaluation of an a posteriori method for estimating and correcting grid-induced errors in solutions of the Navier–Stokes equations, AIAA Pap. 1499 (2009) 2009.
[47] A. Syrakos, G. Efthimiou, J.G. Bartzis, A. Goulas, Numerical experiments on the efficiency of local grid refinement based on truncation error estimates, J. Comput. Phys. 231 (20) (2012) 6725–6753.
[48] A. Syrakos, A. Goulas, Finite volume adaptive solutions using SIMPLE as smoother, Int. J. Numer. Methods Fluids 52 (11) (2006) 1215–1245.
[49] J.J.W. van der Vegt, H. van der Ven, Space time discontinuous Galerkin finite element method with dynamic grid motion for inviscid compressible flows: I. General formulation, J. Comput. Phys. 182 (2) (2002) 546–585.
[50] Z.J. Wang, K. Fidkowski, R. Abgrall, F. Bassi, D. Caraeni, A. Cary, H. Deconinck, R. Hartmann, K. Hillewaert, H.T. Huynh, N. Kroll, G. May, P. Persson, B. van Leer, M. Visbal, High-order CFD methods: current status and perspective, Int. J. Numer. Methods Fluids 72 (8) (January 2013) 811–845.