20th AIAA Computational Fluid Dynamics Conference
27 - 30 June 2011, Honolulu, Hawaii

AIAA 2011-3696

A Robust Adaptive Solution Strategy for High-Order Implicit CFD Solvers

Marco Ceze* and Krzysztof J. Fidkowski†

Department of Aerospace Engineering, University of Michigan, Ann Arbor, MI 48109

This paper presents a solution approach for steady state forms of nonlinear systems of partial differential equations with physical realizability constraints. The main advantage of the approach is the insertion of feasibility constraints in the iterative solution path such that intermediate states before convergence are physically realizable. The method is specially suited for coarse meshes where regular pseudo-transient methods may lead to non-physical states. We present a technique for including the constraints in the solution path that seeks to improve the robustness, with respect to physical realizability, of the iteration to zero residual. In addition, we introduce an adaptive indicator that attempts to localize the cells that are preventing convergence when the solver fails to obtain a *zero-residual* solution. Results are presented in the context of adaptive mesh refinement for laminar and Reynolds-Averaged Navier-Stokes (RANS) flows with highly-under-resolved starting meshes.

I. Introduction

The presence of Computational Fluid Dynamics (CFD) tools in the engineering environment has steadily increased in the past few decades. With the evolution of algorithms and the substantial enhancement of computational power, CFD tools provide the ability to explore new configurations and test flow conditions that may be otherwise difficult to produce experimentally. As the range of applications becomes wider and the number of simulations increases, requirements of high-accuracy and robustness present challenges for the CFD development community.[1] Accuracy is usually assessed by a mesh convergence (*i.e.* grid refinement) study and even then, errors may be large;[2–4] robustness is seldom addressed.

Mesh adaptation methods present an attractive alternative for robust and accurate calculations on affordable grid sizes. These methods rely on the definition of an adaptive indicator which localizes the regions of the computational domain that need mesh modification through refinement, coarsening, or node movement. An effective indicator is obtained through output-based error estimation methods, which have already been demonstrated for many complex problems, including those in aerospace applications.[5–10] The goal of these methods is to provide confidence measures in the form of error bars for scalar outputs of engineering interest. As a by-product of those measures, one can use the error contributions of different elements or volumes of the computational mesh as an adaptive indicator that specifically targets error in the output of interest.[8,9,11–14]

*PhD Candidate, AIAA Member
†Assistant Professor, AIAA Member

Despite the availability of grid coarsening algorithms, it is generally desirable to start mesh adaptation routines with the coarsest mesh that still enables a solution[1] and to only add spatial resolution in areas of the computational domain that are relevant for the particular calculation being performed. Moreover, general gridding guidelines for complex problems are not widely available which presents the question of what is a *fine-enough* mesh that allows the flow solver to converge. In the context of mesh adaptation, this solution does not have to be accurate, and most likely will not be on the initial meshes, but it should allow the adaptive algorithm to proceed to meshes that allow accurate predictions of engineering quantities.

Under-resolved flow features such as shocks, boundary layers, etc. often cause oscillations in the numerical solution whose amplitudes may lead to non-physical values.[1] The advent of limiters in finite-volume schemes substantially increases the robustness of flow solvers at the cost of loss of accuracy. Cockburn *et al*[15] extended the idea of slope limiters, originally proposed by van Leer,[16] to the discontinuous Galerkin finite-element discretization with Runge-Kutta time stepping. This method is know as RKDG and it preserves monotonicity of mean values. More recently, Kuzmin[17] proposed a form of RKDG that uses a hierarchical derivative limiting approach. This is convenient with Taylor basis functions since the limiter acts directly on the degrees of freedom by a process that is equivalent to p-coarsening the cells where the solution is not monotone.

Despite the robustness in time-accurate calculations, schemes with limiters generally do not modify the residual expression[18,19] which means that the final monotone solution does not necessarily satisfy the steady-state flow equations. Conversely, the *zero-residual* solution may also not be monotonic.[19] From the robustness standpoint, numerical oscillations are a problem when they lead to a violation of the physical realizability of the local thermodynamic state.

Alternatively, artificial dissipation is often used as an attempt to smooth out oscillations. Originally proposed by Von Neumann and Richtmyer[20] for capturing shocks and explored by many others, artificial dissipation methods generally use discontinuity sensors that control even-order derivative terms that damp wave-lengths of the order of the local mesh size. This is achieved by augmenting the residual expression with dissipative terms that are negligible in smooth regions of the flow and are triggered at regions with certain features, such as strong gradients or lack of smoothness. Because of the residual modification, these methods do not prevent Newton-based methods from converging to steady-state. The challenge, however, is to determine the level of artificial dissipation that is adequate for robustness but not too large to destroy solution accuracy. In the finite volume community, this balance was found in a seminal paper by Jameson *et al.*[21] In high-order finite elements discretizations, robust artificial dissipation methods are still being pursued for complex problems.[18]

Full nonlinear convergence of the residual to machine precision levels is not strictly necessary for most flow simulations in the design environment. However, in some practical cases of the aeronautical industry, quantities such as drag and moment vary significantly despite the residual being reduced by several orders of magnitude.[1] Additionally, the theory of error estimation makes use of Galerkin orthogonality which is only theoretically valid if the discrete residual is zero. Therefore, coarse meshes with under-resolved flow features may prevent the solver from providing a solution for use in the mesh adaptation algorithm. Limiting methods can help in explicit time-marching, however they are not mature yet in higher-order implicit formulations.

We propose an optimization approach to solve the steady-state form of the governing equations that reduces reliance on meshing guidelines. The technique is based on a minimization argument that stems from Newton's method and allows the inclusion of physical feasibility constraints in the

solution path.

The structure of this paper is as follows. In Section II, we discuss the pseudo-time marching method and present the motivation for an optimization-based solver that is described in Section III. Section IV presents the mesh adaptation strategy followed by a description of the implementation aspects in Section V and results in Section VI. We conclude in Section VII with discussions of extensions and ongoing work.

II. Problem Statement

Let $\mathbf{U}$ denote a discrete state vector and consider the semi-discrete form of the flow equations,

$$\mathbf{U}_t = -\mathbf{R}(\mathbf{U}) = -\mathbf{R}. \tag{1}$$

The boundary conditions and details of the spatial discretization are embedded in the residual $\mathbf{R}$. No single discretization method is considered at this point, and the equations may include convective, diffusive, and source terms. However, it is assumed that the discrete state vector $\mathbf{U}$ is used to approximate the physical state of conserved variables, $\mathbf{u}$, as a field $\mathbf{u}_H(t, x)$.

In finite element methods, the field representation of the state is given by an expansion in terms of basis functions $\phi_j(x)$,

$$\mathbf{u}_H(t, x) = \sum_j \mathbf{U}_j(t)\phi_j(x). \tag{2}$$

In finite volume methods, the field representation $\mathbf{u}_H(t, x)$ is implicitly defined in a reconstruction operator involving the discrete state of the cell containing the point x and the neighboring cells.

For physical realizability, the field $\mathbf{u}_H(t, x)$ is subject to feasibility constraints c_i which, in the case of fluid dynamics, correspond to valid local thermodynamic states, *i.e.*, pressure and density being positive,

$$c_i(\mathbf{u}_H(t, x)) > 0, \tag{3}$$

where i indexes the constraints. Note that when complementing equations, *e.g.* turbulence models, are included in the residual operator, additional constraints may be necessary.

Even though our interest is in the steady state solution of Eqn. 1, the time derivative term is included to improve the initial transient behavior of the solver. We consider the case when the left hand-side is discretized with an implicit scheme with an appropriate strategy for increasing the time step.

A. Constraints

1. Laminar

The constraints for laminar fluid flow are:

$$\begin{aligned} \frac{p(\mathbf{u}_H(t, x))}{p_\infty} &> 0, \\ \frac{\rho(\mathbf{u}_H(t, x))}{\rho_\infty} &> 0, \end{aligned} \tag{4}$$

where p_∞ and ρ_∞ refer to free-stream pressure and density, respectively. Note that ρ is a conserved variable, therefore, its extrema match the extrema of the corresponding position in the vector $\mathbf{u}_H$. Pressure, however, does not have this property. In fact, its curvature along a spatial direction ζ is given by

$$\frac{\partial^2 p}{\partial \zeta^2} = \left(\frac{\partial \mathbf{u}_H}{\partial \zeta}\right)^T \underbrace{\frac{\partial^2 p}{\partial \mathbf{u}_H \partial \mathbf{u}_H^T}}_{\mathcal{H}_p} \left(\frac{\partial \mathbf{u}_H}{\partial \zeta}\right) + \left(\frac{\partial p}{\partial \mathbf{u}_H}\right)^T \frac{\partial^2 \mathbf{u}_H}{\partial \zeta^2}. \tag{5}$$

The eigenvalues of the Hessian of the pressure with respect to the state are

$$\mathrm{eig}(\mathcal{H}_p) = \left(0,\ 0,\ -(\gamma-1)\frac{\rho^2 + (\rho V)^2}{(\rho E)^3}\right). \tag{6}$$

Note that for a linear distribution of state quantities along ζ, the only local extremum possible in pressure between two points is a maximum since the eigenvalues of $\mathcal{H}_p$ are non-positive. Consequently, the pressure constraint will be violated first on the corners of a triangular element with linear Lagrange basis functions. On quadrilateral elements, however, the state distributions are not necessarily linear for a generic direction even for a bilinear Lagrange basis (*e.g.* along the diagonals of the element). Therefore, the pressure constraint can be violated in the interior of a cell despite the distribution being linear along the edges of the element.

2. RANS

In this work we use the Spallart-Allmaras (SA) turbulence model with the modifications proposed by Oliver in Ref. 22. Physical intuition indicates that eddy viscosity should be constrained similarly to pressure and density, *i.e.* $\nu_t > 0$. However, it has been noticed in the literature[22] that mildly negative values of ν_t may be present in a discrete solution computed with the SA model even in a highly-resolved mesh. Therefore, forcing ν_t to be strictly positive may limit convergence to unreasonably-fine meshes.

Alternatively, we impose a constraint of positive total viscosity,

$$\frac{\nu + \nu_t}{\nu} > 0. \tag{7}$$

The kinematic viscosity ν is used in the denominator of Eqn. 7 to make the left hand-side $\mathcal{O}(1)$ in areas where the eddy viscosity is small.

In the SA model, the turbulent viscosity relates to the turbulent state variable $\tilde{\nu}$ through the relationship

$$\nu_t = \tilde{\nu} f_{v1}, \quad \text{where} \quad f_{v1} = \frac{\left(\frac{\tilde{\nu}}{\nu}\right)^3}{\left(\frac{\tilde{\nu}}{\nu}\right)^3 + c_{v1}^3}. \tag{8}$$

B. Pseudo-transient Continuation (PTC)

Since we are interested in the steady-state solution of the flow equations, high-accuracy is not required for discretizing the unsteady term of Eqn. 1. Instead, stability is the main attribute which makes backward Euler an attractive choice. The fully discrete form of Eqn. 1 is then:

$$\mathcal{M}\frac{1}{\Delta t}(\mathbf{U}^{n+1} - \mathbf{U}^{n}) + \mathbf{R}(\mathbf{U}^{n+1}) = 0, \tag{9}$$

where $\mathcal{M}$ is the mass matrix that accounts for the different cell sizes in the mesh. In standard finite volume schemes, this matrix is diagonal and in higher order discretizations such as spectral finite volumes and DG, it is block diagonal.

In time-accurate calculations, Eqn. 9 is solved for the future state using a nonlinear solver such as Newton-Raphson. For steady calculations, the residual at the future state in Eqn. 9 is expanded about the current state and the steps in the iterative procedure require linear solves for the update $\Delta\mathbf{U}^{k} = \mathbf{U}^{k+1} - \mathbf{U}^{k}$,

$$\left(\mathcal{M}\frac{1}{\Delta t} + \frac{\partial \mathbf{R}}{\partial \mathbf{U}}\Big|_{\mathbf{U}^{k}}\right)\Delta\mathbf{U}^{k} = -\mathbf{R}(\mathbf{U}^{k}), \tag{10}$$

where k is used for the linear iteration number to distinguish the method from the time-accurate backward Euler case.

The linearization of the residual operator may involve simplifications due to non-differentiable terms in numerical flux functions and artificial dissipation sensors. Additionally, the sparse structure of the linear system given in Eqn. 10 depends on the type of spatial scheme used for $\mathbf{R}$, and an appropriate choice of iterative solver and preconditioner must be made. The Generalized Minimal Residual (GMRES) algorithm with incomplete Lower-Upper (ILU) preconditioner are typical choices. Note that for $\Delta t \to \infty$ the iterative procedure of Eqn. 10 reduces to Newton's root-finding method.

In the first stages of calculations initialized by states that do not satisfy all boundary conditions, strong transients are observed due to the propagation of boundary information into the domain. To alleviate those transients and to avoid robustness problems, small time steps are used in an attempt to make the solution follow a physical path. This causes a diagonal dominance in the coefficient matrix in Eqn. 10 and makes the calculation closer to time-accurate if Δt does not vary spatially. As an alternative to global time stepping, element-wise time steps can be used by setting a global CFL number defined as:

$$\mathrm{CFL} = \frac{\lambda_{\max}\Delta t}{L_e}, \tag{11}$$

where $\lambda_{\max}$ is the maximum wave speed and L_e is a measure of element size, *e.g.* hydraulic diameter.

The flow state vector $\mathbf{U}$ is updated with $\Delta\mathbf{U}$. For robustness purposes, an under-relaxation parameter is commonly used to keep the solution physical by satisfying the constraints given in Eqn. 3.

$$\mathbf{U}^{k+1} = \mathbf{U}^{k} + \omega^{k}\Delta\mathbf{U}^{k}. \tag{12}$$

1. CFL evolution strategies

Switched Evolution Relaxation

Many strategies for increasing the time step are available.[23,24] Amongst them, a widely used strategy is the *Switched Evolution Relaxation* (SER) method proposed by Mulder and van Leer.[25] The general idea of SER is to change the time step or the CFL number based on a measure of convergence which is inferred from the reduction in a residual norm between consecutive iterations. The algorithm reads as follows:

$$\Delta t^k = \min\left(\Delta t^{k-1}\frac{|\mathbf{R}^{k-1}|}{|\mathbf{R}^k|}, \Delta t_{\max}\right). \tag{13}$$

For local time-stepping, Δt is substituted by CFL in Eqn. 13.

SER is an effective time step evolution strategy. However the constraints (Eqn. 3) are verified after the direction $\Delta\mathbf{u}$ is computed and the relaxation parameter ω must be such that Eqn. 3 is satisfied. In the event of ω becoming too small and the time step not changing significantly, a contingency plan needs to be designed so that the direction $\Delta\mathbf{U}$ changes.

Exponential progression with under-relaxation - EXPur

Alternatively, the CFL evolution can be based on the value of the under-relaxation parameter. Specifically, the CFL increases by a factor $\beta > 1$ if a full update ($\omega = 1$) happened in the previous step of the solver. On the other end, if $\omega < \omega_{\min}$ the CFL is reduced by multiplying it by $\kappa < 1$ and the solver step is repeated. The relaxation factor is limited such that the changes in pressure and density at selected limit points of the interpolated field $\mathbf{u}_H(t,x)$ are within a fraction, $\eta_{\max}$, of the current values. This strategy accounts for the physical feasibility constraints for the next update. However, it is an indirect way of avoiding non-physical states in the flow field since the direction $\Delta\mathbf{U}$ may still produce states that are closer to becoming non-physical even at the minimum CFL. In particular, this is observed in highly under-resolved meshes.

The CFL strategy is summarized below:

$$\mathrm{CFL}^{k+1} = \begin{cases} \beta\cdot\mathrm{CFL}^k & \text{for } \beta>1 \quad \text{if} \quad \omega^k = 1 \\ \mathrm{CFL}^k & \text{if} \quad \omega_{\min} < \omega^k < 1 \\ \kappa\cdot\mathrm{CFL}^k & \text{for } \kappa<1 \quad \text{if} \quad \omega^k < \omega_{\min} \end{cases}. \tag{14}$$

C. Pseudo-transient as an optimizer

This section presents an optimization aspect of the pseudo-transient continuation method. We assume the coefficient matrix in Eqn. 10 is real and non-singular and the update $\Delta\mathbf{U}$ is not zero. Additionally, we drop the index k for brevity.

Multiplying the left-hand side of Eqn. 10 by its transpose gives:

$$\Delta\mathbf{U}^T\left(\mathcal{M}\frac{1}{\Delta t}+\frac{\partial\mathbf{R}}{\partial\mathbf{U}}\right)^T\left(\mathcal{M}\frac{1}{\Delta t}+\frac{\partial\mathbf{R}}{\partial\mathbf{U}}\right)\Delta\mathbf{U} = -\Delta\mathbf{U}^T\underbrace{\left(\mathcal{M}\frac{1}{\Delta t}+\frac{\partial\mathbf{R}}{\partial\mathbf{U}}\right)^T\mathbf{R}(\mathbf{U})}_{\frac{\partial f}{\partial\mathbf{U}}} > 0. \tag{15}$$

Therefore, $\Delta\mathbf{U}$ is a descent direction for the scalar function $f(\mathbf{U})$ given by:

$$f(\mathbf{U}) = \int \mathcal{M}\frac{1}{\Delta t}\mathbf{R}(\mathbf{U})\ d\mathbf{U} + \frac{1}{2}\mathbf{R}(\mathbf{U})^T\mathbf{R}(\mathbf{U}). \tag{16}$$

In the limit of infinite time-step, $f(\mathbf{U})$ corresponds to the square of the L_2-norm of the residual:

$$|\mathbf{R}(\mathbf{U})|^2_{L_2} = \frac{1}{2}\mathbf{R}(\mathbf{U})^T\mathbf{R}(\mathbf{U}). \tag{17}$$

Note that Newton's method for finding the roots of $\mathbf{R}(\mathbf{U})$ is a quasi-Newton method for minimizing the function given in Eqn. 17 since the Hessian of $|\mathbf{R}(\mathbf{U})|^2_{L_2}$ includes second derivatives of $\mathbf{R}$.

III. Residual Optimization Technique

We present a solution technique that includes the physical realizability constraints in the solution path to provide the ability to circumvent the non-physical regions of $\mathbf{u}$-space. First, we use the minimization argument in Section II. This allows us to use constraint handling methods from the optimization field.

Since the residual operator is not defined for non-physical states (*e.g.* negative pressure), we need to keep the iterates within the physical region of the solution space. Therefore, interior penalty methods[26] are attractive because of their simplicity and efficiency in acknowledging feasibility constraints in the solution path. These methods augment a scalar objective function with a term that tends to infinity as the solution path approaches a feasibility boundary creating a repelling effect with respect to prohibited regions of the domain.

A simple way of employing interior penalty methods for the flow equations is to augment Eqn. 17 with a barrier function of the constraints in Eqn. 3 and use a quasi-Newton optimizer as a solver. However, this leads to a highly ill-conditioned Hessian of the augmented objective function due to the squaring of the residual vector.

Alternatively, we propose augmenting the residual with a penalty vector to account for the constraints:

$$\mathbf{R}_p(\mathbf{U}) = \mathbf{R}(\mathbf{U}) + \mathbf{P}(\mathbf{U}). \tag{18}$$

In order to have the repelling effect with respect to non-feasible regions of the domain, the penalization vector $\mathbf{P}$ must have a positive projection on the direction of the residual vector $\mathbf{R}$. To satisfy this requirement, we define the penalization vector as:

$$\mathbf{P} = \Phi\ \mathbf{R}, \tag{19}$$

where Φ is a diagonal matrix with the elemental penalties $\mathbb{P}_e$ for each row corresponding to an element "e":

$$\Phi_{ij} = \begin{cases} \mathbb{P}_e & \text{if } \ i = j \in e \\ 0 & \end{cases}. \tag{20}$$

The elemental penalty is given by:

$$\mathbb{P}_e(\mathbf{U}_e(t), \mu) = \mu \sum_i^{N_i} \sum_j^{N_j} \frac{1}{c_i(\mathbf{u}_h(t, x_j))} \quad \forall x_j \in \mathcal{S}_e, \tag{21}$$

where $\mathcal{S}_e$ is the geometric space occupied by element "e", N_i is the number of constraints and μ is a scaling factor, referred to as the penalty factor throughout the text. Note that if the constraints are evaluated on all the points of an element, the summation over "j" becomes an integral in a reference element with unitary volume. In finite element methods, such an integral can be approximated using quadrature rules. Additionally, the projection of $\mathbf{P}$ – as defined in Eqn. 19 – onto the residual vector is always positive for non-zero $\mathbf{R}$ since the elemental penalties are strictly positive in the feasible domain.

A root of the residual operator corresponds to a root of $\mathbf{R}_p$, so that the steady-state solution is independent of the values of the elemental penalties. Additionally, one can use the pseudo-transient continuation method presented in Section II-B to evolve the solution from the initial condition to steady state. The equation for the update direction becomes:

$$\left(\underbrace{(\mathbf{I}+\Phi)^{-1}\frac{\mathcal{M}}{\Delta t}}_{a} + \frac{\partial \mathbf{R}}{\partial \mathbf{U}} + \underbrace{(\mathbf{I}+\Phi)^{-1}\left(\frac{\partial \Phi}{\partial \mathbf{U}}\mathbf{R}(\mathbf{U})\right)}_{b} \right) \Delta \mathbf{U} = -\mathbf{R}(\mathbf{U}). \tag{22}$$

The terms "a" and "b" in Eqn. 22 are block diagonal for discontinuous Galerkin methods. Additionally, the CFL number gets amplified by $(1+\mathbb{P}_e)$ for each element and in the limit of an infinite time step, the solution path seeks a minimum of $|\mathbf{R}_p|_{L_2}$. Similarly, the time continuation term vanishes at elements where the solution approaches a non-physical region while the penalization term "b" does not vanish because the function value of inverse barrier penalties (Eqn. 21) tends to infinity at a slower rate than the magnitude of its derivative.

The CFL evolves according to the strategies described in Section II-B-1 and at each solver step the state is updated according to Eqn. 12. The penalty factor μ is evolved using a form of SER:

$$\mu^{n+1} = \min\left(\mu^n \frac{\langle 1+\mathbb{P}_e(\mathbf{U}^n,\mu^n)\rangle}{\langle 1+\mathbb{P}_e(\mathbf{U}^{n-1},\mu^{n-1})\rangle}, \mu_{\max}\right), \tag{23}$$

where $\langle\cdot\rangle$ indicates an average over all elements. This evolution strategy for μ tries to make the solver acknowledge the presence of a feasibility constraint by increasing its repelling effect as the solution path goes towards a non-physical state. Conversely, if the solution path is moving away from a feasibility boundary the repelling effect decreases to make the linear solves easier to compute since the effect of the time-continuation matrix increases.

The penalty factor is initialized such that $\mathcal{O}(\langle 1+\mathbb{P}_e\rangle) = 1$. This keeps the pseudo-transient term active and alleviates the initial solution transients while helping the spectral conditioning of initial linear systems.

1. Iterative procedure

The iterative procedure for the vector penalization approach consists of:

1. Choose an initial CFL and its evolution strategy.
2. Choose μ^0 such that $\mathcal{O}(\langle 1+\mathbb{P}_e\rangle) = 1$.
3. If $|\mathbf{R}(\mathbf{U}^k)| \leq \varepsilon_{\text{res}}$, then CONVERGED.
4. Compute the search direction $\Delta\mathbf{U}^k$ by solving Eqn. 22 using GMRES.

5. Compute the under-relaxation factor ω^k that keeps the changes in primitive variables limited to a fraction $\eta_{\max}$ of the current values.
6. If $\omega^k \geq \omega_{\min}$, then
   (a) Update the state with $\mathbf{U}^{k+1} = \mathbf{U}^k + \omega^k \Delta U^k$.
   (b) Evolve CFL with chosen strategy.
   (c) GO TO 4.

   Else
   (a) Reduce CFL.
   (b) GO TO 4.

A maximum number of iterations is set in case the convergence criterion is not met for the residual. Additionally, we set a maximum number of reductions of the CFL number, $N_{\kappa,\max}$.

IV. Mesh Adaptation Strategy

The mesh adaptation strategy has two modes. One of them is a robustness mode in which we attempt to localize the cells that are preventing the solver from converging to a physical root of the residual operator. The other mode consists of output-based adaptation which identifies for refinement the cells that are most contributing to the error in an output of interest.

A. Robustness Indicator

This indicator is computed when the solver is not able to meet the convergence criterion for the residual norm. Since both PTC and ROT methods retain the time continuation term, they are expected to be globally convergent in the unconstrained sense.[24,27,28] Therefore, it is reasonable to assume that the solver is not proceeding due to physical constraints. Based on this assumption, we propose a mesh adaptation indicator that is the projection of the search direction $\Delta\mathbf{U}$ onto the gradient of the penalty function as an attempt to localize the cells that are trying to violate the physical realizability constraints.

The search direction is obtained by solving Equations 10 or 22 depending on the method being used. The adaptive indicator θ is given by the following inner product:

$$\theta = \sum_{e}^{N_e} \underbrace{\Delta\mathbf{U}_e \cdot \frac{\partial \mathbb{P}_e}{\partial \mathbf{U}_e}}_{\theta_e}, \tag{24}$$

where $\Delta\mathbf{U}_e$ is the update vector for element "e" and we drop the superscript index k for clarity.

Note that $\theta_e > 0$ suggests that the element "e" is preventing the solver from proceeding because the solution path may be approaching a non-physical state for that element. Moreover, only the elements with positive indicators should be considered for refinement since negative θ_e are related to cells that are not immediately trying to violate the feasibility constraints.

B. Output-error Indicator

The theory behind output error estimation for steady flows requires a solution that satisfies the steady flow equations. Therefore, the output-based indicator is only computed when the flow solver

satisfies a strict convergence criterion,

$$\mathbf{R}_H(\mathbf{U}_H) = 0. \tag{25}$$

We denote the converged discrete solution on a "coarse" mesh by $\mathbf{U}_H$ and the residual operator in the coarse space by $\mathbf{R}_H$. The error for an scalar output J is defined as the difference:

$$\delta J = J(\mathbf{U}_H) - J(\mathbf{U}_h), \tag{26}$$

where $\mathbf{U}_h$ denotes a fine-space solution that is a surrogate for the exact discrete solution.

The coarse-space solution will generally not satisfy the fine-space residual operator:

$$\mathbf{R}_h(\mathbf{U}_h^H) \neq 0, \tag{27}$$

where $\mathbf{U}_h^H$ denotes the projection of the coarse-space solution onto a finer space $\mathcal{V}_h \supset \mathcal{V}_H$. On defining an adjoint solution $\boldsymbol{\psi}$ as the sensitivity of an output J with respect to a infinitesimal source of residual we have:

$$\delta J \equiv \psi^T \delta \mathbf{R} = -\boldsymbol{\psi}^T \mathbf{R}(\mathbf{U} + \delta \mathbf{U}). \tag{28}$$

We can obtain the adjoint equation by linearizing the output about the state and using the definition in Eqn. 28:

$$\left(\frac{\partial \mathbf{R}}{\partial \mathbf{U}}\right)^T \boldsymbol{\psi} + \frac{\partial J}{\partial \mathbf{U}} = 0. \tag{29}$$

Note that the above equation is linear and the cost to solve it is equivalent to one step of the nonlinear solver previously described. Since the difference between the fine-space and coarse-space solution is not necessarily infinitesimal the equality sign in Eqn. 28 becomes an approximation and the error estimate is computed as:

$$\delta J \approx -\boldsymbol{\psi}_h^T \mathbf{R}_h(\mathbf{U}_h^H), \tag{30}$$

where $\boldsymbol{\psi}_h^T$ can be approximated or computed based on the projected state $\mathbf{U}_h^H$. The fine-space can be generated by mesh subdivision or interpolation order increment.

Many adjoint-weighted methods for error estimation are available[29] and they differ mostly in the technique for obtaining the fine-space approximations and consequently the estimate Eqn. 30. The adaptive indicator is defined as the contribution to δJ of the fine-space degrees of freedom pertaining to each element of the mesh,

$$\eta_e = \Big| \sum_{j \in e} \boldsymbol{\psi}_{H,j}^T \mathbf{R}_{h,j} \Big|. \tag{31}$$

C. Adaptation Mechanics

The elemental adaptive indicators described above drive a fixed-fraction, hanging-node adaptation strategy where each cell can be refined directionally or isotropically. In this strategy, a fraction f^{adapt} of the elements with the largest adaptive indicators are adapted with a maximum difference of one level of refinement between adjacent elements.

The anisotropic adaptation framework used in this work was proposed in Ref. 14 and it solves local optimization problems to estimate the most efficient direction of refinement when only a discrete set of options is available. Since that framework requires a surrogate for the adaptive indicator to estimate the gain of each refinement option, we only use anisotropic refinement with the output-error indicator.

When the mesh adaptation algorithm is in robustness mode, the cells with the largest $\theta_e > 0$ are marked for refinement. Therefore, the fraction of elements that are refined can be smaller then f^{adapt}.

V. Implementation

The residual optimization technique was implemented in a finite element code with a DG discretization of the compressible Navier-Stokes equations. We used Roe's approximate Riemann solver[30] for the inviscid fluxes and the second form of Bassi and Rebay,[31] BR2, for the viscous discretization.

Both the PTC and ROT methods march to the steady-state solution via a backward Euler solver with a local time stepping. The linear solves are performed using Newton-GMRES with line-Jacobi preconditioning and a convergence criterion based on the current nonlinear residual norm. The CFL evolution strategy is the EXPur method with ω^k calculated such that the changes in pressure and density are limited to $\eta_{\max} = 10\%$ of their local values at iteration k. If $\omega^k < \omega_{\min}$, the CFL is reduced by a factor $\kappa = 0.1$, up to a maximum of $N_{\kappa,\max}$ CFL reductions.

The same Newton-GMRES linear solver is used for computing both adaptive indicators. Since the Jacobian matrix is stored, the additional complexity of the transpose solve for the adjoint problem is minimal. Finally, the fine approximation space, $\mathcal{V}_h$, required for the output-error indicator $\boldsymbol{\psi}_h$ is obtained by increasing the approximation order from p to $p+1$ on the same mesh.

VI. Results

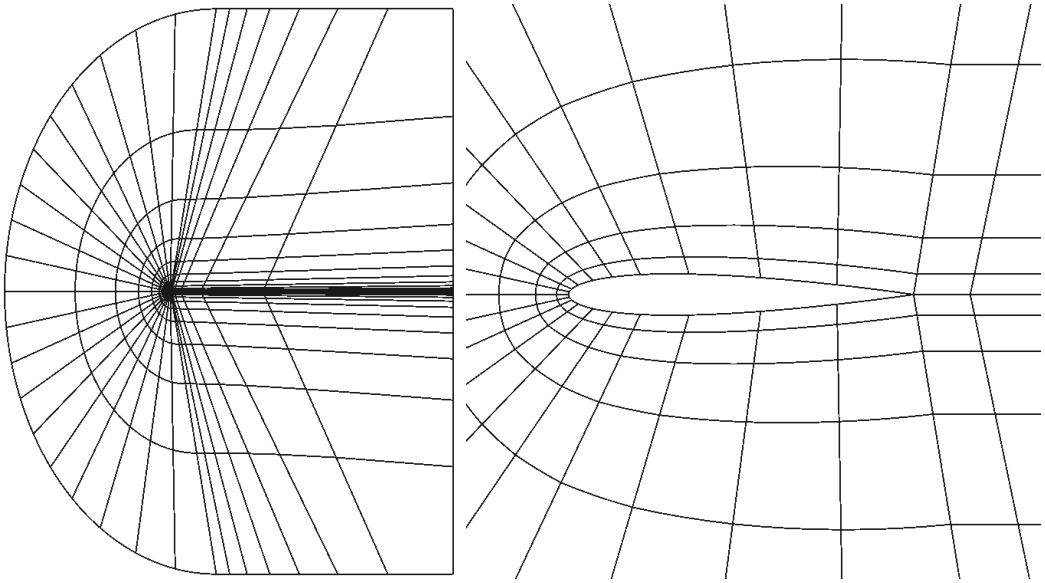
We present a comparison between PTC and ROT for laminar and RANS flows. Both methods use the EXPur CFL strategy for which $\omega_{\min} = 0.01$ and $\beta = 1.2$. The number of CFL reductions is limited to $N_{\kappa,\max}$. After that, the mesh is adapted using the robustness indicator and the solution proceeds from the last full update ($\omega^k = 1.0$). This restart strategy avoids non-physical states when injecting the under-converged solution into the newly refined mesh. The penalty factor is initialized such that $\left\langle 1 + \mathbb{P}_e(\mathbf{U}^0, \mu^0) \right\rangle = 10^{0.25}$.

The mesh is adapted using the robustness indicator until the solver converges. Once the residual norm is less than $\varepsilon_{\text{res}} = 10^{-9}$ the mesh is anisotropically adapted using the output-error indicator and the process re-continues until the next residual convergence. For all the cases we present, the fraction f^{adapt} of elements selected for refinement is 20% when in robustness mode and 10% when in output-error mode.

A. Laminar NACA 0012, $M = 0.8, \alpha = 0.0^\circ, Re = 5,000$

The first example is laminar flow over the NACA 0012 airfoil. The outer boundary is located approximately 200 chord-lengths from the airfoil and the initial and adapted meshes consist of cubic ($q = 3$) quadrilaterals. We consider linear ($p = 1$) and quadratic ($p = 2$) interpolation

orders with Lagrange basis functions on the purposely coarse initial mesh shown in Figure 1. We allow $N_{\kappa,\max} = 3$ CFL reductions and the maximum number of robustness adaptation steps is $N_{\theta,\max} = 10$. At each nonlinear iteration, the linear solve is computed such that the norm of the GMRES residual is less than $|\mathbf{R}(\mathbf{U}^k)|_{L_1} \times 10^{-5}$.



(a) Initial mesh (b) Initial mesh - zoom.

Figure 1. NACA 0012, $M = 0.8, \alpha = 0.0^\circ, Re = 5,000$: Initial mesh - 476 $q = 3$ elements.

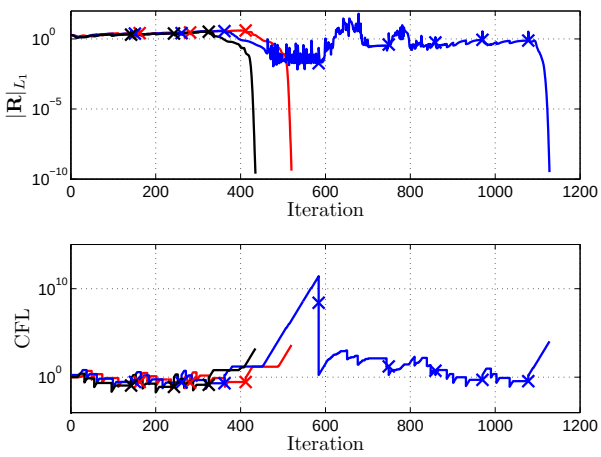
Figure 2(a) shows the residual norm history of PTC and ROT for $p = 1$ solution interpolation. The initial CFL is 1.0 for both methods. However, in order to account for the initial CFL amplification in ROT a additional case is shown for PTC with $\text{CFL}^0 = \left\langle 1 + \mathbb{P}_e(\mathbf{U}^0, \mu^0) \right\rangle = 10^{0.25}$ (blue line in Fig. 2(a)). The cross-marks indicate when the CFL was reduced $N_\kappa > N_{\kappa,\max}$ times. At that point, $f^{\text{adapt}} = 20\%$ of the elements were adapted in robustness mode and the solution continued as previously described. Note that the initial CFL correction caused the PTC solver to encounter a unsteady behavior in the residual history. Despite that behavior, residual convergence was achieved after 8 robustness adaptation cycles. With $\text{CFL}^0 = 1.0$, ROT and PTC used 3 robustness adaptation cycles to converge the residual.

Figure 2(b) shows the average penalization and the penalty factor histories. Note that these quantities are reset after each adaptation cycle. As mentioned above, this strategy avoids ill-conditioned initial linear systems and lets the time continuation term help in propagating waves out of the domain.

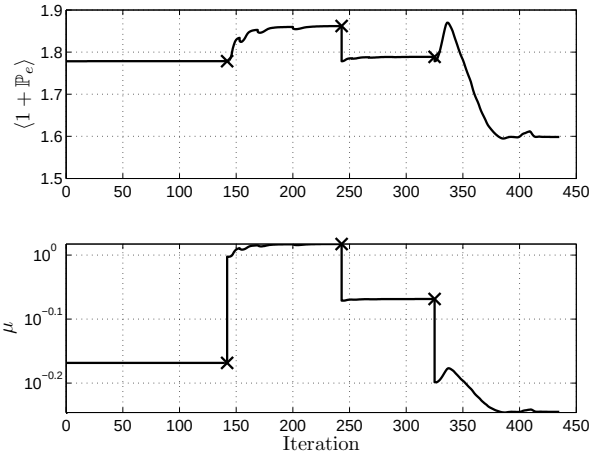
Figure 3 compares PTC and ROT using $p = 2$ solution interpolation. Despite the enhanced spatial resolution provided by the quadratic interpolation, the number of robustness adaptation cycles taken for convergence is larger than for $p = 1$ under the same conditions.

Table 1 summarizes the runs with the starting mesh shown in Figure 1. We used 32 CPUs for the $p = 1$ runs and 48 CPUs for $p = 2$. For linear interpolation, the shortest CPU time and the mesh with fewest elements is obtained with ROT.

With hanging-node refinement, meshes with the same number of refinement cycles can have



(a) Residual L_1-norm history; red: PTC; blue: PTC with corrected initial CFL; black: ROT.



(b) Penalization terms history.

Figure 2. NACA 0012, $M = 0.8, \alpha = 0.0^\circ, Re = 5,000$: comparison between PTC and ROT methods with $p = 1$ solution interpolation order; the crosses mark when the mesh was adapted using the robustness indicator; red: PTC with $\mathbf{CFL}^0 = 1.0$; blue: PTC with $\mathbf{CFL}^0 = \langle 1 + \mathbb{P}_e(\mathbf{U}^0, \mu^0) \rangle$; black: ROT with $\mathbf{CFL}^0 = 1.0$.

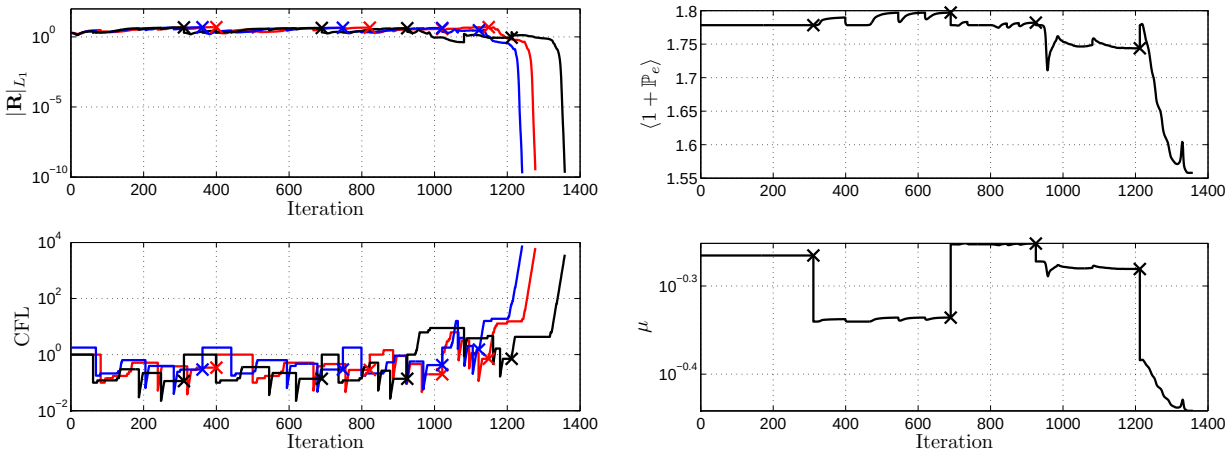
Table 1. Summary of runs for NACA 0012, $M = 0.8, \alpha = 0.0^\circ, Re = 5,000$ with the initial mesh shown in Figure 1.

Method	$\mathbf{CFL}^0$	#iter.	#adapt.	#elem.	$\boldsymbol{p}$-order	CPU time (sec)	$\lvert\mathbf{R}\rvert_{\boldsymbol{L_1}} \leq \boldsymbol{\varepsilon}_{\text{res}}$
PTC	1.0	520	3	1995	1	2505.9	✓
PTC	1.78	1128	8	13739	1	83586	✓
ROT	1.0	435	3	1977	1	2023.7	✓
PTC	1.0	1277	4	3164	2	21195	✓
PTC	1.78	1241	4	3166	2	28229	✓
ROT	1.0	1358	4	2576	2	28451	✓

small differences in cell-count due to implementation aspects of the adaptation mechanics. However the difference in element-count observed between ROT and PTC meshes with the same number of adaptation cycles is larger and not only caused by the mesh mechanics. The additional cause is because the ROT solver accounts for the feasibility constraints and when the meshes are adapted in robustness mode, the number of elements with positive constraint projection (Eqn. 24) tends to be smaller with ROT than with PTC despite f^{adapt} being the same for both cases.

Figures 4 and 5 show the first *zero-residual* meshes and Mach contours for both methods. Note that the $p = 1$ runs focused the refinement on the fore region around the airfoil where the pressure field presents more variation, while the $p = 2$ runs refined more on the aft part of the airfoil and in the wake region.

In engineering practice, the goal of CFD calculations is to compute an output of interest within a



(a) Residual L_1-norm history; red: PTC; blue: PTC with corrected initial CFL; black: ROT.

(b) Penalization terms history.

Figure 3. NACA 0012, $M = 0.8, \alpha = 0.0^\circ, Re = 5,000$: comparison between PTC and ROT methods with $p = 2$ solution interpolation order; the crosses mark when the mesh was adapted using the robustness indicator; red: PTC with $\text{CFL}^0 = 1.0$; blue: PTC with $\text{CFL}^0 = \langle 1 + \mathbb{P}_e(\mathbf{U}^0, \mu^0) \rangle$; black: ROT with $\text{CFL}^0 = 1.0$.

certain tolerance. We address this requirement with output-error estimation and mesh adaptation. Figure 6 shows the convergence of the drag coefficient for linear and quadratic solution interpolations using ROT. We used the initial solutions described above.

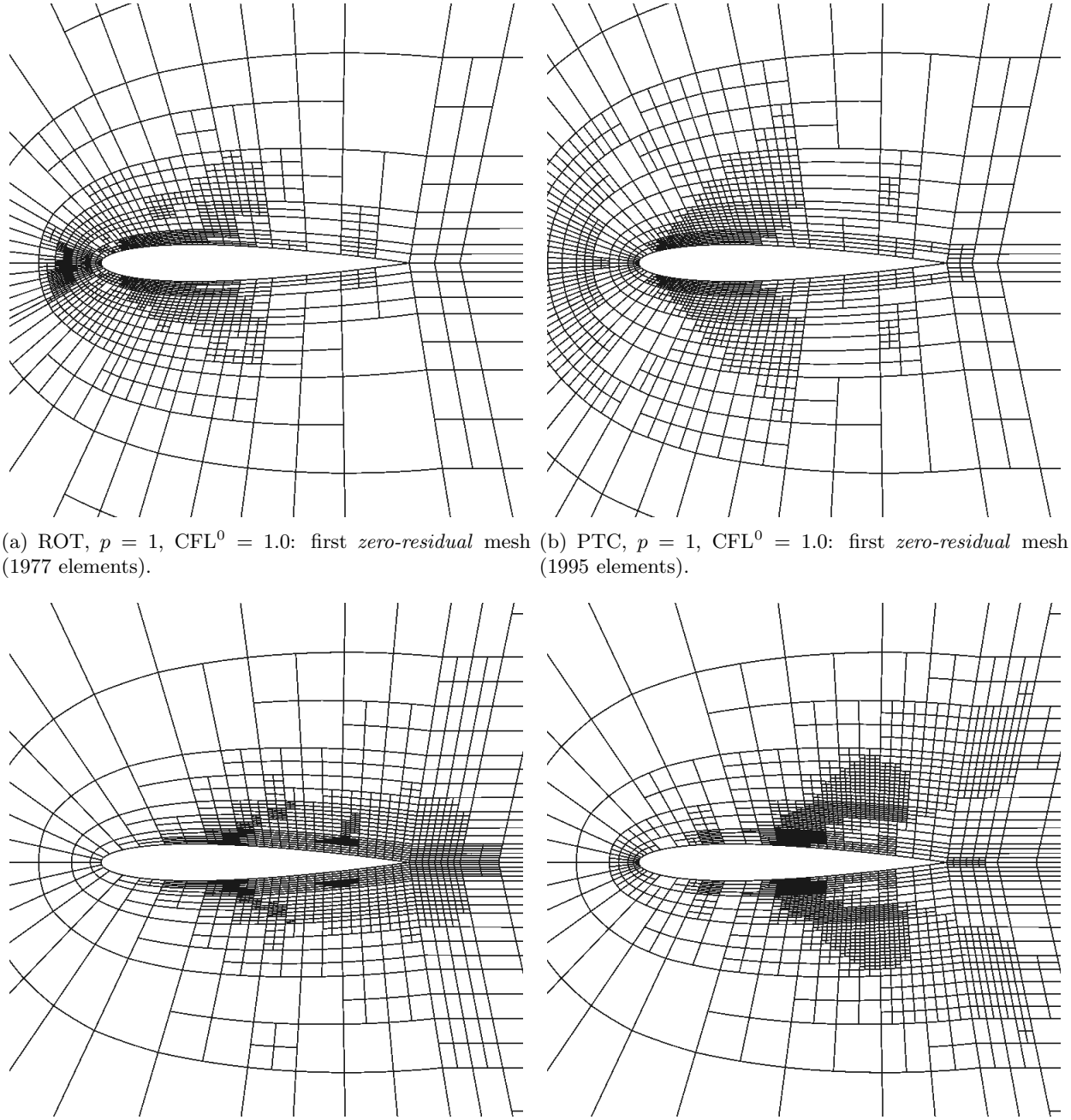
The error bars correspond to the sum of error indicators, η_e. They were computed with adjoint-based error estimation and the reference drag value was computed by uniformly refining the finest mesh and raising the interpolation order to $p = 3$. Note in Figure 6(a) that between the second and third output adaptation cycles, the solver requested one robustness adaptation in order to converge the residual and continue the output-based cycle. Conversely, once the first *zero-residual* solution was obtained with $p = 2$ no robustness adaptations were requested and the output-based cycle ran without interruptions.

Figure 6(c) shows the history of the error measures. As expected, $p = 2$ presents a faster drag convergence rate than $p = 1$, however its initial setup cost is higher in terms of degrees of freedom. From the engineering perspective, a adequate metric for cost is CPU time and a safe metric for benefit is the sum of error indicators since it does not allow cancellation due to different signs. For this assessment, we show in Figure 6(d) the absolute value of the drag error estimate for each output-adapted mesh and its corresponding computational time. Note that the choice between $p = 1$ and $p = 2$ depends on the level of error required for the calculation.

Figure 7 shows the final drag-adapted meshes and Mach contours. Note the anisotropic refinement on the edge of the separated flow region and the added spatial resolution on the leading edge of the airfoil.

B. RANS NACA 0012, $M = 0.8, \alpha = 1.25^\circ, Re = 200,000$

The second case we present is turbulent flow over the NACA 0012 airfoil. The outer boundary is located approximately 200 chord-lengths from the airfoil and the initial and adapted meshes consist



(a) ROT, $p = 1$, $\mathrm{CFL}^0 = 1.0$: first *zero-residual* mesh (1977 elements).

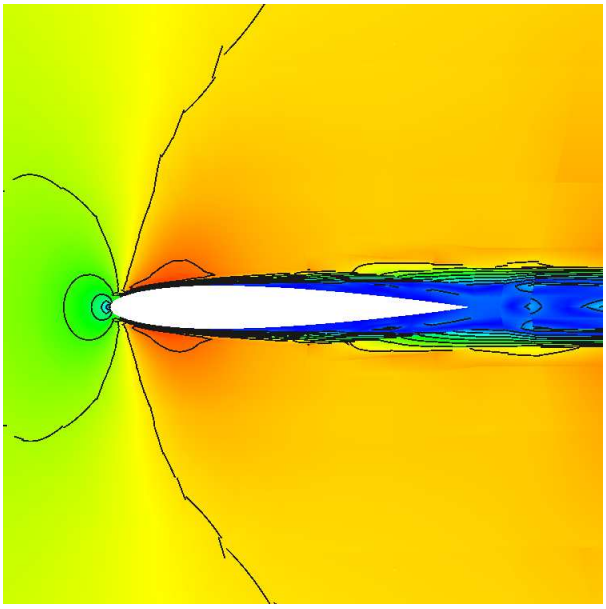
(b) PTC, $p = 1$, $\mathrm{CFL}^0 = 1.0$: first *zero-residual* mesh (1995 elements).

(c) ROT, $p = 2$, $\mathrm{CFL}^0 = 1.0$: first *zero-residual* mesh (2576 elements).

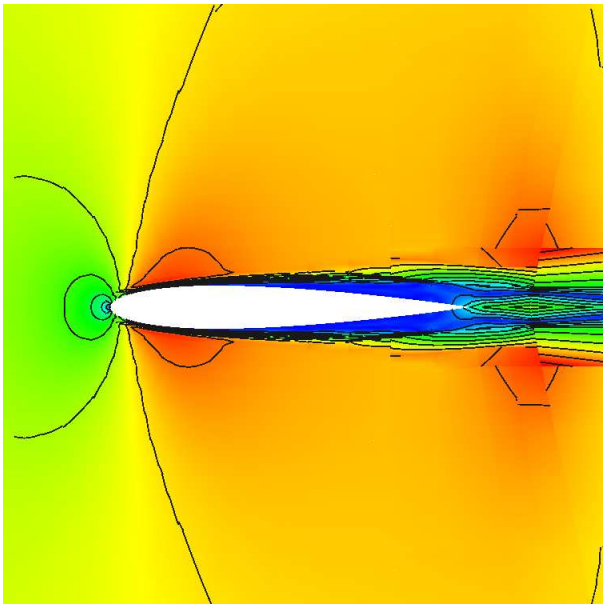
(d) PTC, $p = 2$, $\mathrm{CFL}^0 = 1.0$: first *zero-residual* mesh (3164 elements).

Figure 4. NACA 0012, $M = 0.8, \alpha = 0.0^\circ, Re = 5,000$**: first *zero-residual* meshes.**

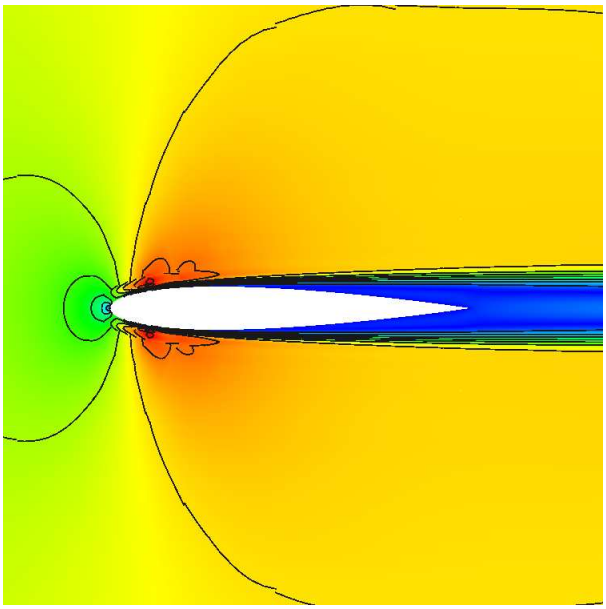
of cubic ($q = 3$) quadrilaterals. We consider $p = 2$ Lagrange basis interpolation on the mesh shown in Figure 8.



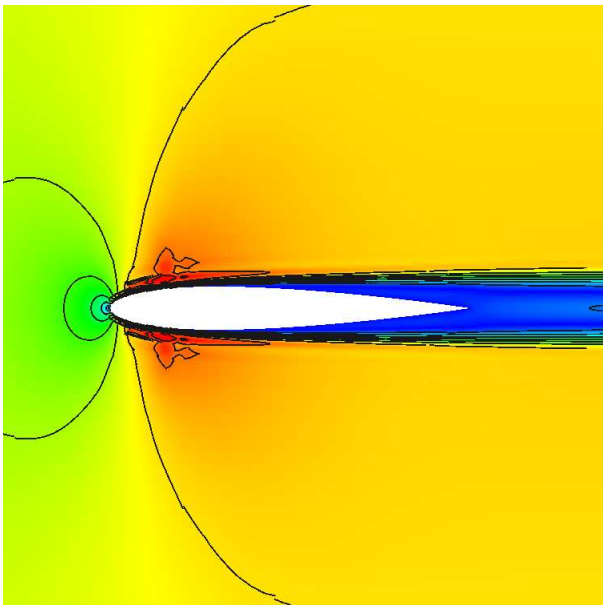
(a) ROT, $p = 1$, $\text{CFL}^0 = 1.0$: first *zero-residual* Mach contours.



(b) PTC, $p = 1$, $\text{CFL}^0 = 1.0$: first *zero-residual* Mach contours.



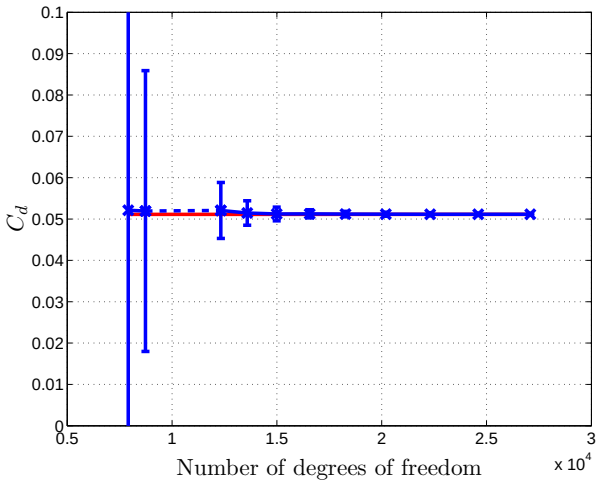
(c) ROT, $p = 2$, $\text{CFL}^0 = 1.0$: first *zero-residual* Mach contours.



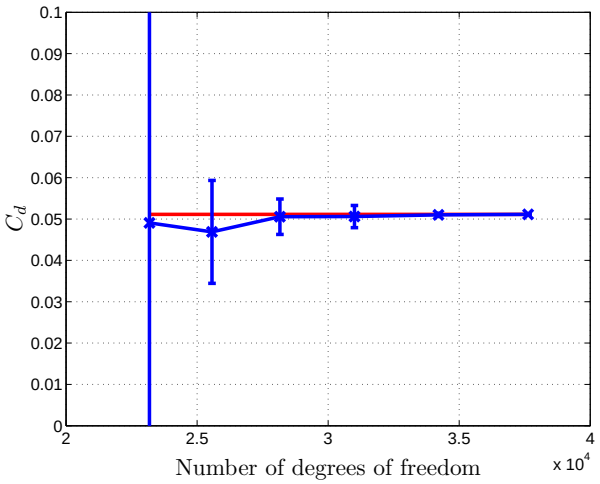
(d) PTC, $p = 2$, $\text{CFL}^0 = 1.0$: first *zero-residual* Mach contours.

Figure 5. NACA 0012, $M = 0.8, \alpha = 0.0^\circ, Re = 5,000$**: first *zero-residual* Mach contours.**

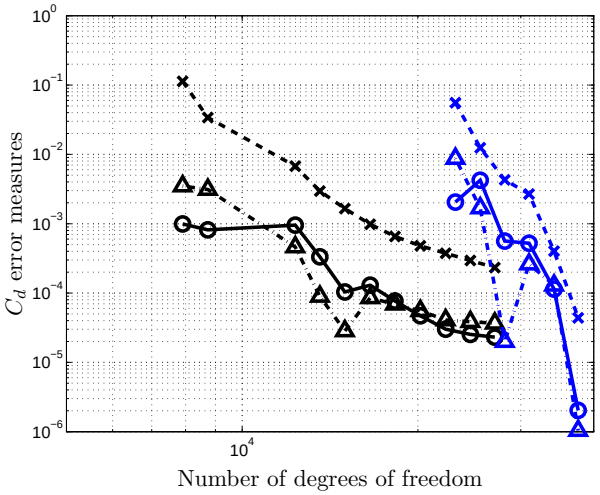
We compare ROT and PTC using two strategies to evolve the solution from the free-stream initial conditions. First we consider order-sequencing where we seek a $p = 1$ solution and use it as a initial condition for $p = 2$. The second strategy uses $p = 2$ directly from the free-stream condition.



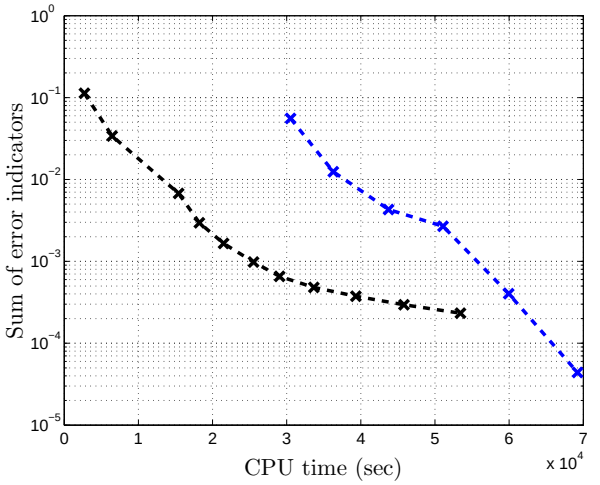
(a) Drag coefficient convergence for $p = 1$ solution interpolation; the initial mesh was obtained after 3 adaptation cycles in robustness mode; dashed line indicates that one robustness adaptation was performed to achieve residual convergence.



(b) Drag coefficient convergence for $p = 2$ solution interpolation; the initial mesh was obtained after 4 adaptation cycles in robustness mode.



(c) Drag error measures; black: $p = 1$; blue: $p = 2$. Circles: $C_d - C_{d_{\text{ref}}}$; crosses: sum of error indicators; triangles: error estimate.

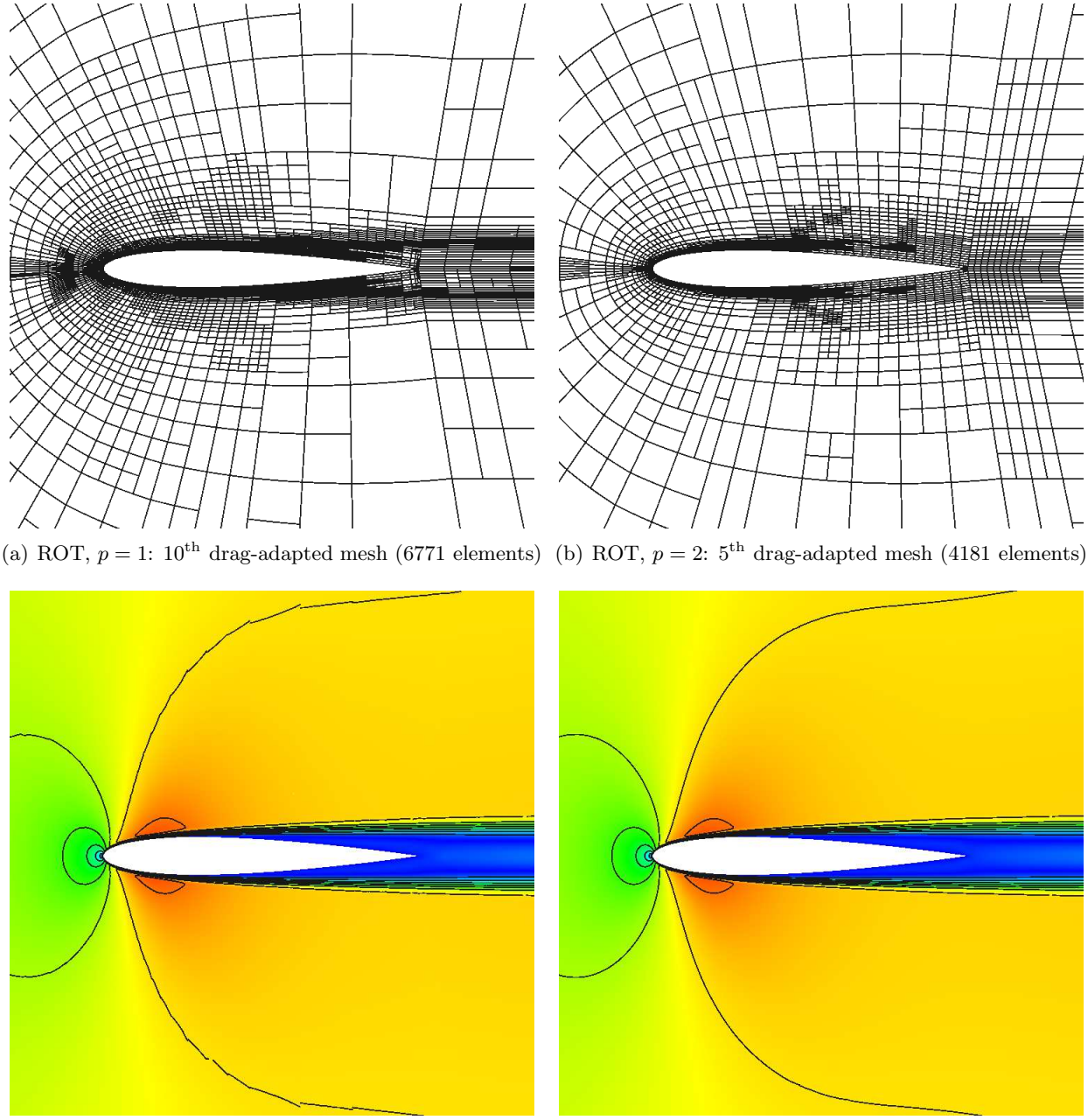


(d) Sum of drag error indicators versus computational time; black triangles: $p = 1$; blue crosses: $p = 2$.

Figure 6. NACA 0012, $M = 0.8, \alpha = 0.0°, Re = 5,000$: drag coefficient convergence for output-based adaptation.

The starting CFL for both methods and strategies is $\text{CFL}^0 = 1.0$ and the residual convergence criterion is $\varepsilon_{\text{res}} = 10^{-9}$. Similarly to the previous case, the linear solves at each nonlinear iteration are computed such that the norm of the GMRES residual is less than $|\mathbf{R}(\mathbf{U}^k)|_{L_1} \times 10^{-5}$. We allow $N_{\kappa,\max} = 2$ CFL reductions and the maximum number of robustness adaptation steps is $N_{\theta,\max} = 10$.

Figure 9(a) shows the residual norm and the CFL histories for the order continuation strategy



(a) ROT, $p = 1$: 10^{th} drag-adapted mesh (6771 elements)

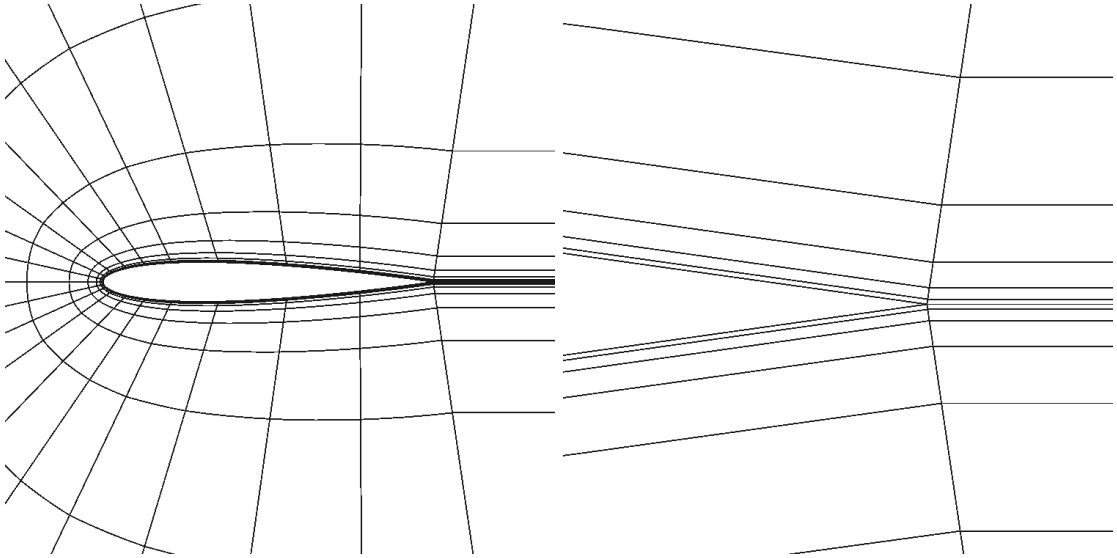
(b) ROT, $p = 2$: 5^{th} drag-adapted mesh (4181 elements)

(c) ROT, $p = 1$: 10^{th} drag-adapted solution - Mach contours

(d) ROT, $p = 2$: 5^{th} drag-adapted solution - Mach contours

Figure 7. NACA 0012, $M = 0.8, \alpha = 0.0^\circ, Re = 5,000$**: drag-adapted meshes and corresponding Mach contours.**

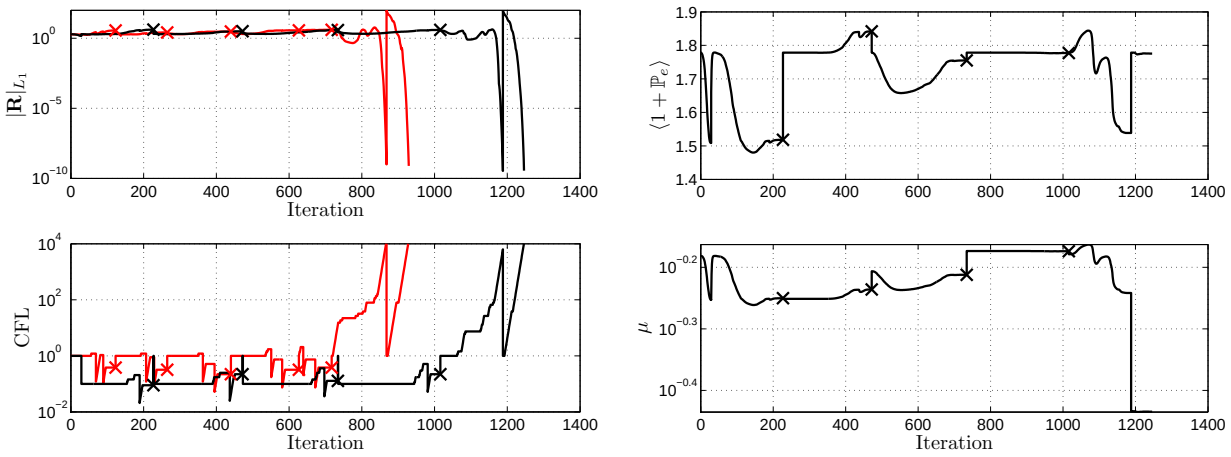
with PTC and ROT. Note that the double residual convergence for both methods corresponds to the convergence of $p = 1$ and $p = 2$ solution interpolations, respectively. PTC used 5 robustness adaptation steps to converge the $p = 1$ solution while ROT used 4 steps. Once the $p = 1$ solution



(a) Initial mesh (b) Initial mesh - trailing edge zoom.

Figure 8. NACA 0012, $M = 0.8, \alpha = 1.25^\circ, Re = 200,000$: Initial mesh - 392 $q = 3$ elements.

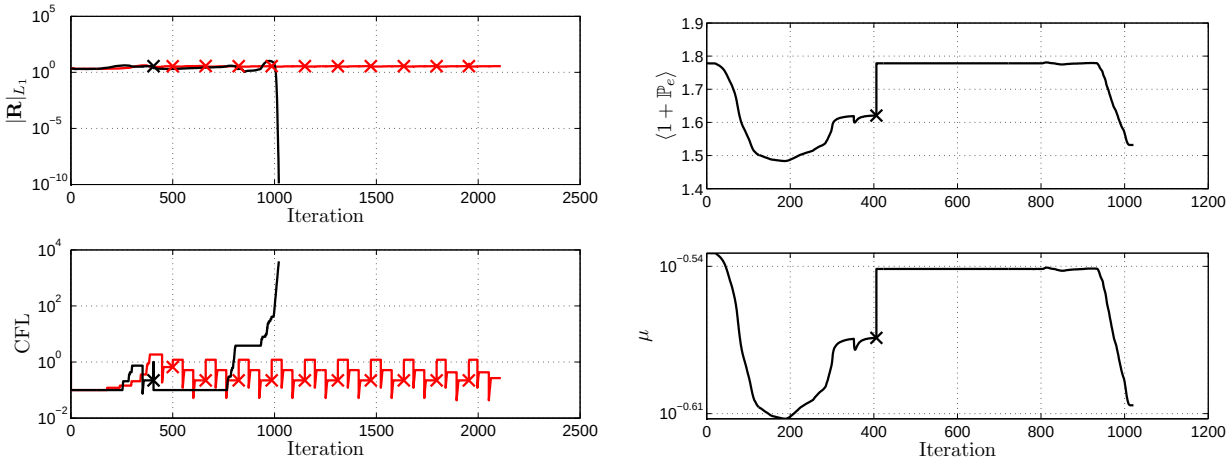
was obtained, both methods converged the $p = 2$ solution in a small number of iterations. Figure 10(a) shows the CFL and residual histories for both methods starting directly with $p = 2$. Note that the PTC method could not converge the residual after 10 robustness adaptation steps whereas ROT used one adaptation step to achieve convergence.



(a) Residual L_1-norm history; red: PTC; black: ROT. (b) Penalization terms history.

Figure 9. NACA 0012, $M = 0.8, \alpha = 1.25^\circ, Re = 200,000$: Comparison between PTC and ROT methods for interpolation order continuation from $p = 1$ to $p = 2$; the crosses mark when the mesh was adapted using the robustness indicator; red: PTC with $\mathbf{CFL}^0 = 1.0$; black: ROT with $\mathbf{CFL}^0 = 1.0$.

Table 2 summarizes the runs using the initial mesh shown in Figure 8. Note that ROT with direct $p = 2$ took considerably less computational time and elements to achieve convergence than



(a) Residual L_1-norm history; red: PTC; black: ROT.

(b) Penalization terms history.

Figure 10. NACA 0012, $M = 0.8, \alpha = 1.25^\circ, Re = 200,000$: Comparison between PTC and ROT methods for $p = 2$ solution interpolation order; the crosses mark when the mesh was adapted using the robustness indicator; red: PTC with $\text{CFL}^0 = 1.0$; black: ROT with $\text{CFL}^0 = 1.0$.

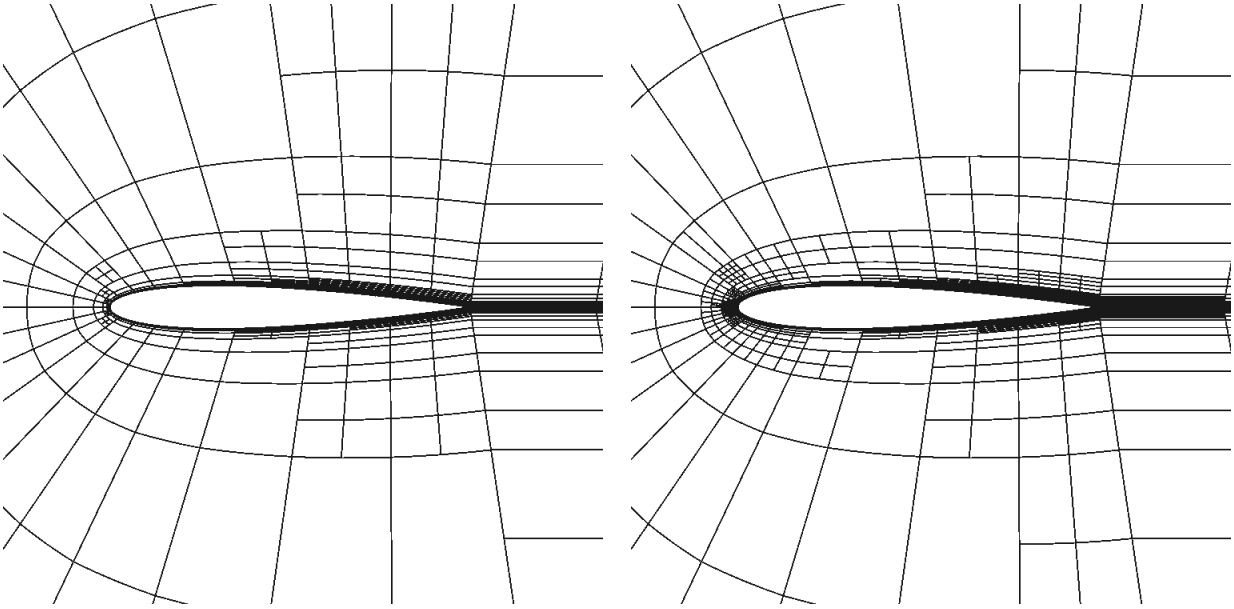
the other runs. Additionally, even though ROT took more nonlinear iterations than PTC in the order continuation runs, the overall computational time is shorter because the mesh has fewer elements.

Table 2. Summary of runs for NACA 0012, $M = 0.8, \alpha = 1.25^\circ, Re = 200,000$ with the initial mesh shown in Figure 8.

Method	CFL^0	#iter.	#adapt.	#elem.	p-order	CPU time (sec)	$\|\mathbf{R}\|_{L_1} \leq \varepsilon_{\text{res}}$
PTC	1.0	929	5	4310	$1 \rightarrow 2$	50422	✓
ROT	1.0	1246	4	2663	$1 \rightarrow 2$	48996	✓
PTC	1.0	2111	10	59913	2	262950	
ROT	1.0	1021	1	626	2	9963.8	✓

Figures 11 and 12 show the final meshes and solution states for the cases in Table 2. Note that the under-converged PTC case concentrated the refinement on the leading edge area. This indicates that the PTC method had difficulty propagating the solid boundary information in the upstream direction. The meshes for ROT present some refinement on the region near the leading edge, however, most of the refinement is in the aft part of the airfoil and in the wake region.

The first $p = 2$ *zero-residual* meshes and solutions obtained with ROT were used in a output-based adaptation cycle. Figures 13(a) and 13(b) show the drag coefficient convergence using the initial meshes described above. Similarly to the laminar case, the reference solution was obtained by uniformly refining the finest mesh and raising the interpolation order to $p = 3$. The difference between the computed drag with each output-adapted mesh and the reference drag is shown in Figure 13(c) along with other error measures. Note that the initial mesh obtained directly with $p =$



(a) ROT, $p = 1 \rightarrow 2$, $\text{CFL}^0 = 1.0$: first *zero-residual* mesh (2663 elements).

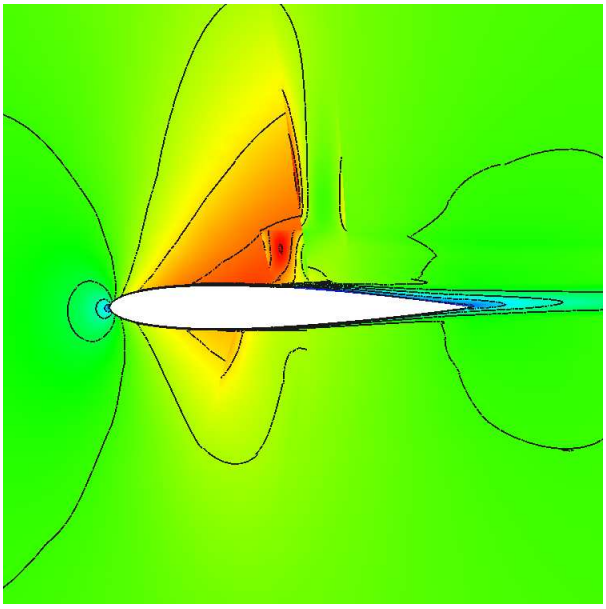
(b) PTC, $p = 1 \rightarrow 2$, $\text{CFL}^0 = 1.0$: first *zero-residual* mesh (4310 elements).

(c) ROT, $p = 2$, $\text{CFL}^0 = 1.0$: first *zero-residual* mesh (626 elements).

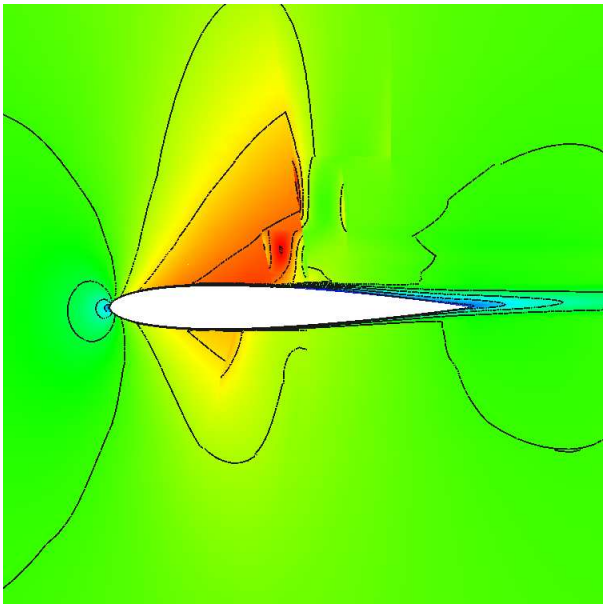
(d) PTC, $p = 2$, $\text{CFL}^0 = 1.0$: final mesh adapted in robustness mode (59913 elements).

Figure 11. NACA 0012, $M = 0.8, \alpha = 1.25^\circ, Re = 200,000$**: final meshes for the runs in Table 2.**

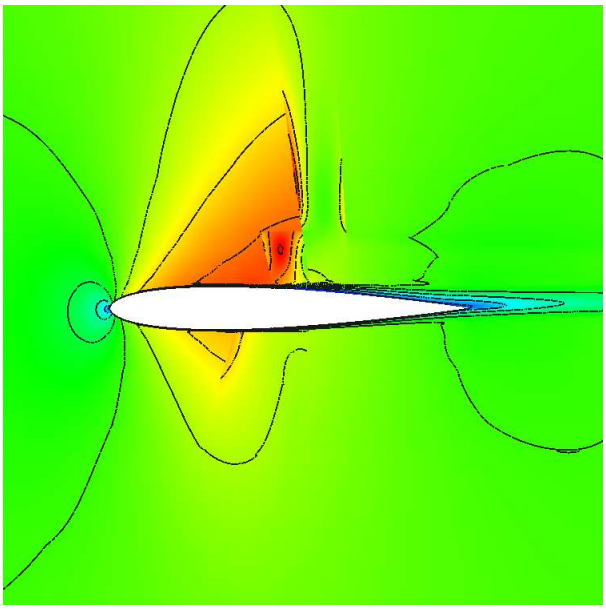
2 has roughly 4 times fewer elements than the mesh obtained with order continuation. Despite this difference, the drag value and its error estimate are similar for both initial meshes. Moreover, the



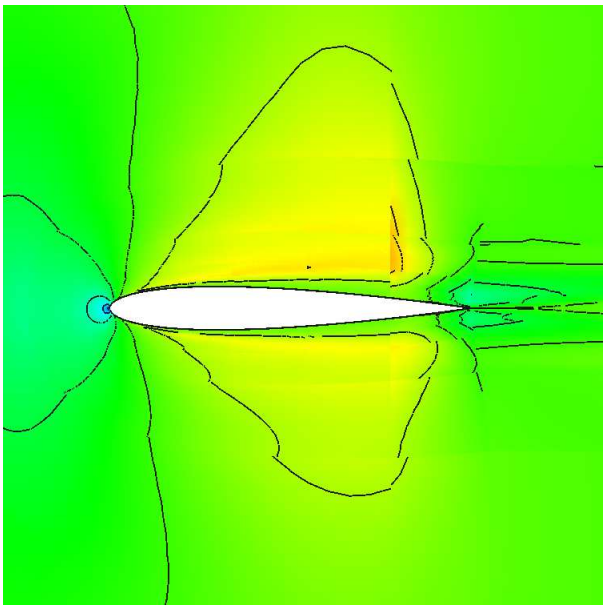
(a) ROT, $p = 1 \rightarrow 2$, $\text{CFL}^0 = 1.0$: first *zero-residual* solution - Mach contours.



(b) PTC, $p = 1 \rightarrow 2$, $\text{CFL}^0 = 1.0$: first *zero-residual* solution - Mach contours.



(c) ROT, $p = 2$, $\text{CFL}^0 = 1.0$: first *zero-residual* solution - Mach contours.

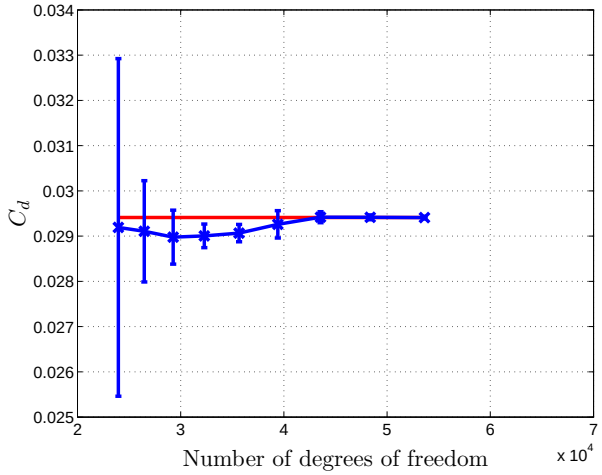


(d) PTC, $p = 2$, $\text{CFL}^0 = 1.0$: final solution state (not converged) - Mach contours.

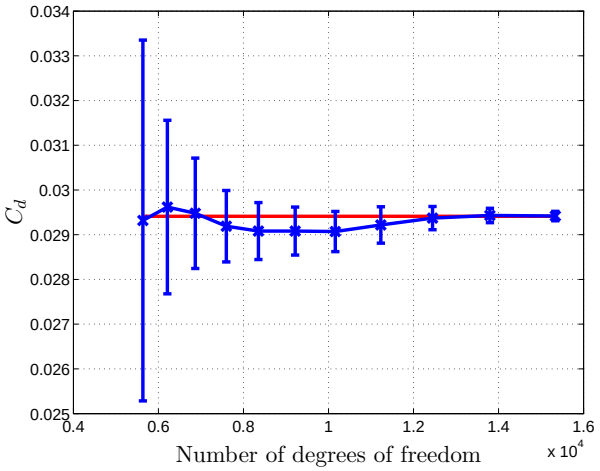
Figure 12. NACA 0012, $M = 0.8, \alpha = 1.25°, Re = 200,000$**: final solution state for the runs in Table 2.**

savings in terms of computational time to achieve an error level of 1 drag count is also approximately a factor of 4 when computing the initial solution directly with $p = 2$ approximation order.

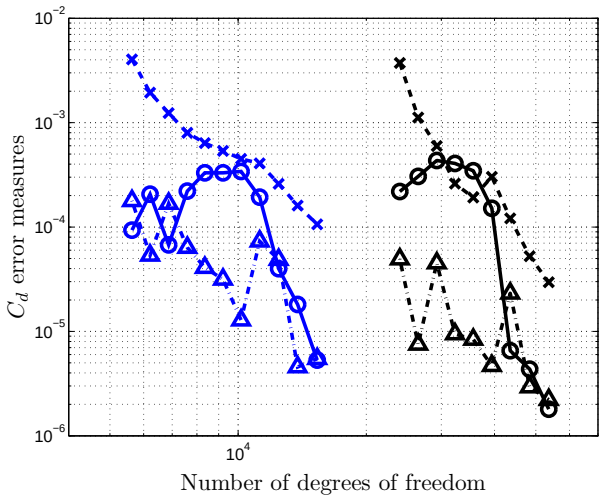
Figure 14 shows the final output-adapted meshes and their corresponding Mach-number con-



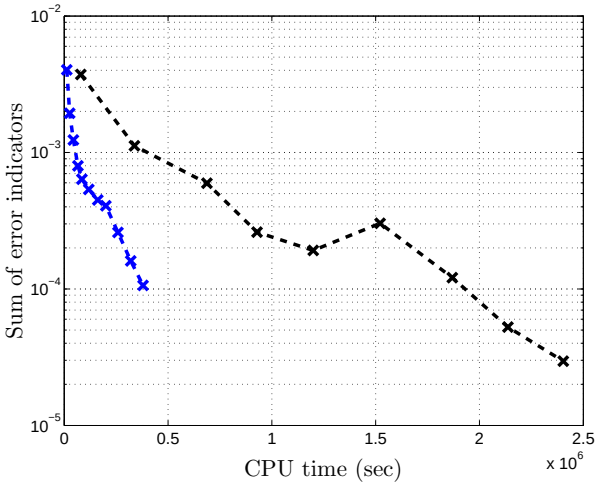
(a) Drag coefficient convergence for $p = 2$ solution interpolation; the initial mesh was obtained with ROT using order continuation and robustness adaptation.



(b) Drag coefficient convergence for $p = 2$ solution interpolation; the initial mesh was obtained with ROT directly for $p = 2$ using robustness adaptation.



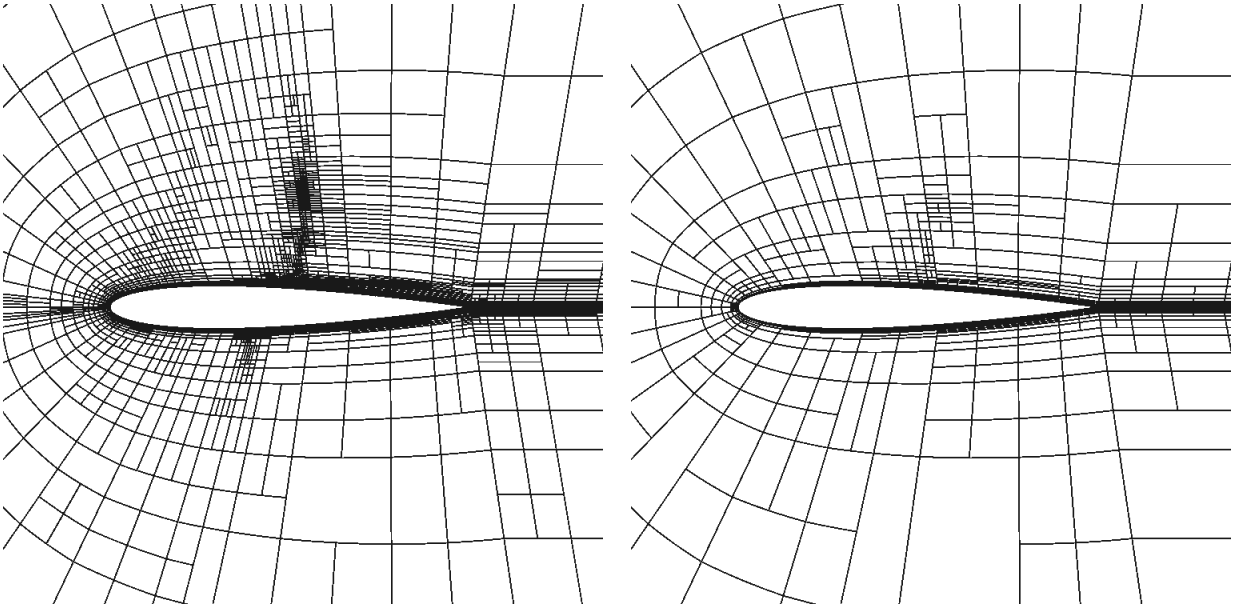
(c) Drag error measues; black triangles: order continuation for initial mesh and solution; blue crosses: direct $p = 2$ initial solution and mesh. Circles: $C_d - C_{d_{\text{ref}}}$; crosses: sum of error indicators; triangles: error estimate.



(d) Sum of drag error indicators versus computational time; black: order continuation for initial for mesh and solution; blue: direct $p = 2$ initial mesh and solution.

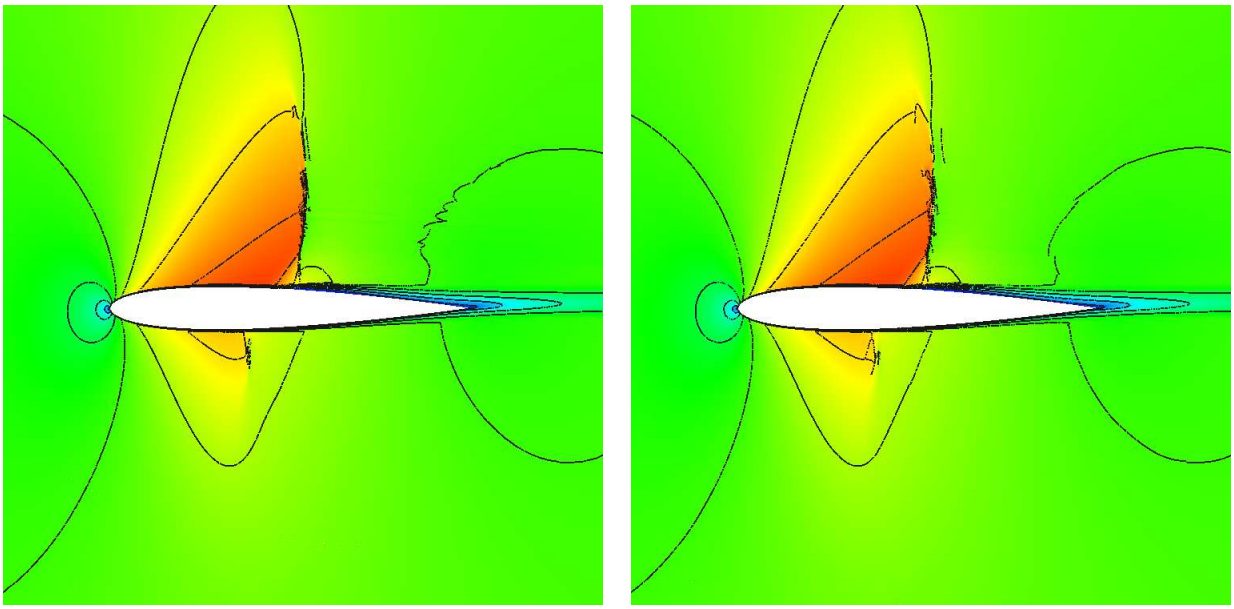
Figure 13. NACA 0012, $M = 0.8, \alpha = 1.25°, Re = 200,000$: drag coefficient convergence for output-based adaptation.

tours. Note the refinement on the edges of the supersonic regions and on the boundary layer after the shocks for both strategies.



(a) ROT, $p = 1 \rightarrow 2$: 8^{th} drag-adapted mesh (5958 elements)

(b) ROT, $p = 2$: 10^{th} drag-adapted mesh (1704 elements)



(c) ROT, $p = 1 \rightarrow 2$: 8^{th} drag-adapted solution - Mach contours

(d) ROT, $p = 2$: 10^{th} drag-adapted solution - Mach contours

Figure 14. NACA 0012, $M = 0.8, \alpha = 1.25°, Re = 200,000$: drag-adapted meshes and corresponding Mach contours.

VII. Conclusions and Ongoing Work

This paper presents a novel method for solving the steady-state flow equations in the context of highly-under-resolved meshes where regular pseudo-transient methods may not provide a solution.

This method includes information of physical realizability constraints in the solution path with the objective of improving robustness by trying to circumvent non-physical regions of the solution space. From an optimization argument that stems from the backward Euler time continuation scheme, the constraints are incorporated in the iterative solution path by a vector penalization technique that attempts to make the prohibited regions of the solution space less attractive for the solver, and thus shifting the solution path away from non-physical states.

The results indicate a gain in robustness over the baseline pseudo-transient method when the mesh is under-resolved. Moreover, this robustness improvement combined with mesh adaptation can allow the solver to start directly with the desired approximation order, resulting in reduced computational cost for a certain level of output error. In practice, the initial meshes are likely to have better spatial resolution than the starting meshes shown in this paper. However, they attempt to simulate the condition when initial meshes are generated by *non-expert* practitioners or when adequate spatial resolution may be difficult to achieve solely at the user level.

We also introduce a mesh adaptation indicator that helps the solver achieve residual convergence. The comparison between this indicator and regular residual-based indicators is subject of ongoing work. In addition, we expect three-dimensional turbulent flows at higher Reynolds numbers to benefit more from the ability to provide physically realizable solutions on under-resolved meshes. These flows and extension to other turbulence models are future work.

Finally, the ROT method and the robustness indicator are not restricted to quadrilateral or hexahedral meshes with hanging-node refinement. Therefore, they can be applied to other types of meshes and refinement strategies.

References

[1]Buttazzo, G., Frediani, A., Allmaras, S. R., Bussoletti, J. E., Hilmes, C. L., Johnson, F. T., Melvin, R. G., Tinoco, E. N., Venkatakrishnan, V., Wigton, L. B., and Young, D. P., "Algorithm issues and challenges associated with the development of robust CFD codes," *Variational Analysis and Aerospace Engineering*, Vol. 33 of *Springer Optimization and Its Applications*, Springer New York, 2009, pp. 1–19.

[2]Levy, D. W., Zickuhr, T., Vassberg, J., Agrawal, S., Wahls, R. A., Pirzadeh, S., and Hemsch, M. J., "Data summary from the First AIAA Computational Fluid Dynamics Drag Prediction Workshop," *Journal of Aircraft*, Vol. 40, No. 5, 2003, pp. 875–882.

[3]Laflin, K. R., Vassberg, J. C., Wahls, R. A., Morrison, J. H., Brodersen, O., Rakowitz, M., Tinoco, E. N., and Godard, J.-L., "Summary of data from the second AIAA CFD drag prediction workshop," AIAA Paper 2004-0555, 2004.

[4]Frink, N. T., "Test case results from the 3rd AIAA drag prediction workshop," NASA Langley, 2007, `http://aaac.larc.nasa.gov/tsab/cfdlarc/aiaa-dpw/ Workshop3/final_results_jm.tar.gz`.

[5]Becker, R. and Rannacher, R., "An optimal control approach to a posteriori error estimation in finite element methods," *Acta Numerica*, edited by A. Iserles, Cambridge University Press, 2001, pp. 1–102.

[6]Giles, M. and Pierce, N., "Adjoint error correction for integral outputs," *Lecture Notes in Computational Science and Engineering: Error Estimation and Adaptive Discretization Methods in Computational Fluid Dynamics*, Vol. 25, Springer, Berlin, 2002.

[7]Hartmann, R. and Houston, P., "Adaptive discontinuous Galerkin finite element methods for the compressible Euler equations," *Journal of Computational Physics*, Vol. 183, No. 2, 2002, pp. 508–532.

[8]Venditti, D. A. and Darmofal, D. L., "Anisotropic grid adaptation for functional outputs: application to two-dimensional viscous flows," *Journal of Computational Physics*, Vol. 187, No. 1, 2003, pp. 22–46.

[9]Nemec, M., Aftosmis, M. J., and Wintzer, M., "Adjoint-based adaptive mesh refinement for complex geometries," AIAA Paper 2008-0725, 2008.

[10]Dwight, R. P., "Heuristic a posteriori estimation of error due to dissipation in finite volume schemes and application to mesh adaptation," *Journal of Computational Physics*, Vol. 227, 2008, pp. 2845–2863.

[11]Park, M. A., "Adjoint-based, three-dimensional error prediction and grid adaptation," AIAA Paper 2002-3286, 2002.

[12]Hartmann, R. and Houston, P., "Goal-oriented a posteriori error estimation for multiple target functionals," *Hyperbolic Problems: Theory, Numerics, Applications*, edited by T. Hou and E. Tadmor, Springer-Verlag, 2003, pp. 579–588.

[13]Fidkowski, K. J. and Darmofal, D. L., "An adaptive simplex cut-cell method for discontinuous Galerkin discretizations of the Navier-Stokes equations," AIAA Paper 2007-3941, 2007.

[14]Ceze, M. and Fidkowski, K. J., "Output-driven anisotropic mesh adaptation for viscous flows using discrete choice optimization," *48th AIAA Aerospace Sciences Meeting and Exhibit*, 2010.

[15]Cockburn, B., Lin, S.-Y., and Shu, C.-W., "TVB Runge-Kutta local projection discontinuous Galerkin finite element method for conservation laws iii: one-dimensional systems," *Journal of Computational Physics*, Vol. 84, No. 90-113, 1989.

[16]van Leer, B., "Towards the ultimate conservative difference scheme. II - monotonicity and conservation combined in a second order scheme," *Journal of Computational Physics*, , No. 14, 1974, pp. 361–370.

[17]Kuzmin, D., "A vertex-based hierarchical slope limiter for p-adaptive discontinuous Galerkin methods," *Journal of Computational and Applied Mathematics*, , No. 233, 2010.

[18]Barter, G. E., *Shock Capturing with PDE-Based Artificial Viscosity for an Adaptive Higher–Order Discontinuous Galerkin Finite Element Method*, Ph.D. thesis, Massachusetts Institute of Technology, Cambridge, Massachusetts, 2008.

[19]Venkatakrishnan, V., "Convergence to steady state solutions of the Euler equations on unstructured grid with limiters," *Journal of Computational Physics*, , No. 118, 1995, pp. 120–130.

[20]Neumann, J. V. and Richtmyer, R. D., "A method for the numerical calculation of hydrodynamic shocks," *Journal of Applied Physics*, Vol. 21, 1950, pp. 232–237.

[21]Jameson, A., Schmidt, W., and Turkel, E., "Numerical simulation of the Euler equations by finite volume methods using Runge-Kutta time stepping schemes," *AIAA 5th Computational Fluid Dynamics Conference*, 1981.

[22]Oliver, T. A., *A High–order, Adaptive, Discontinuous Galerkin Finite Elemenet Method for the Reynolds-Averaged Navier-Stokes Equations*, Ph.D. thesis, Massachusetts Institute of Technology, Cambridge, Massachusetts, 2008.

[23]Bucker, H. M., Pollul, B., and Rasch, A., "On CFL evolution strategies for implicit upwind methods in linearized Euler equations," *International Journal for Numerical Methods in Fluids*, 2009.

[24]Kelley, C. T. and Keyes, D. E., "Convergence analysis of pseudo-transient continuation," *SIAM Journal on Numerical Analysis*, 1998.

[25]Mulder, W. A. and van Leer, B., "Experiments with implicit upwind methods for the Euler equations," *Journal of Computational Physics*, 1985.

[26]Hartung, J., "A stable interior penalty method for convex extremal problems," *Numerische Mathematik*, Vol. 29, No. 2, 1978.

[27]Kelley, C. T., Liao, L.-Z., Qi, L., Chu, M. T., Reese, J., and Winton, C., "Projected pseudo-transient continuation," *SIAM Journal on Numerical Analysis*, Vol. 46, No. 6, 2008, pp. 3071–3083.

[28]Coffey, T. S., Kelley, C. T., and Keyes, D. E., "Pseudo-transient continuation and differential-algebraic equations," *Journal of Scientific Computing*, Vol. 25, No. 2, 2003.

[29]Fidkowski, K. J. and Darmofal, D. L., "Review of output-based error estimation and mesh adaptation in computational fluid dynamics," *American Institute of Aeronautics and Astronautics Journal*, Vol. 49, No. 4, 2011, pp. 673–694.

[30]Roe, P. L., "Approximate Riemann solvers, parametric vectors, and difference schemes," *Journal of Computational Physics*, Vol. 43, 1981, pp. 357–372.

[31]Bassi, F. and Rebay, S., "GMRES discontinuous Galerkin solution of the compressible Navier-Stokes equations," *Discontinuous Galerkin Methods: Theory, Computation and Applications*, edited by K. Cockburn and Shu, Springer, Berlin, 2000, pp. 197–208.