Development of a High-Order Space-Time Matrix-Free Adjoint Solver

Marco A. Ceze*, Laslo T. Diosady†, and Scott M. Murman‡

NASA Ames Research Center, Moffett Field, CA, USA

I. Introduction

The growth in computational power and algorithm development in the past few decades has granted the science and engineering community the ability to simulate flows over complex geometries, thus making Computational Fluid Dynamics (CFD) tools indispensable in analysis and design. Currently, one of the pacing items limiting the utility of CFD for general problems is the prediction of unsteady turbulent flows.[1–3] Reynolds-averaged Navier-Stokes (RANS) methods, which predict a time-invariant mean flowfield, struggle to provide consistent predictions when encountering even mild separation, such as the side-of-body separation at a wing-body junction. NASA's Transformative Tools and Technologies project is developing both numerical methods and physical modeling approaches to improve the prediction of separated flows. A major focus of this effort is efficient methods for resolving the unsteady fluctuations occurring in these flows to provide valuable engineering data of the time-accurate flow field for buffet analysis, vortex shedding, etc. This approach encompasses unsteady RANS (URANS), large-eddy simulations (LES), and hybrid LES-RANS approaches such as Detached Eddy Simulations (DES). These unsteady approaches are inherently more expensive than traditional engineering RANS approaches, hence every effort to mitigate this cost must be leveraged. Arguably, the most cost-effective approach to improve the efficiency of unsteady methods is the optimal placement of the spatial and temporal degrees of freedom (DOF) using solution-adaptive methods.

Under the steady-flow assumption, solution-adaptive methods have been demonstrated to be efficient in reducing discretization error by spatially distributing degrees of freedom according to a scalar field – the adaptive indicator – that assesses the importance of different regions of the domain. A robust method for identifying all important regions in the domain is the adjoint-weighted residual which estimates the discretization error in an output of interest (*e.g.:* lift and drag) and the elemental contribution to this error yields an adaptive indicator.[4] Degrees of freedom are added/removed by locally refining/coarsening (h-adaptation) the mesh or by increasing/decreasing (p-adaptation) the scheme's approximation order. The combination and the choice between these options are not trivial tasks and they have been the subjects of many research works.[5–11] Other approaches that remesh[12–14] the domain based on a metric field derived from adjoint-based error estimates and anisotropy measures have also been demonstrated to be efficient in controlling discretization error.

A far less explored research topic is solution adaptation for accurate and efficient simulation of unsteady flows.[14–17] The limited number of research works in space-time adaptivity is partly due to algorithmic challenges to affordably simulate large-scale unsteady flows and to the difficulty of designing and implementing a robust space-time mesh adaptation strategy. Nevertheless, these challenges are worth taking as the two-dimensional results in Ref. 17 show that adjoint-based space-time adaptation can reduce the number of degrees of freedom by one to three orders of magnitude in comparison to uniform refinement, especially when a high level of accuracy is required. More importantly, the authors show that certain heuristic indicators cause the adaptive procedure to converge to the wrong output value.

In steady problems, the cost of an adjoint solution is comparable to the cost of the primal solution with small variations depending on the implementation. The time dependence in unsteady problems requires the adjoint to be solved backwards in time and, at each time step, a residual linearization about the current

*Postdoctoral Fellow, Oak Ridge Associated Universities, marco.a.ceze@nasa.gov
†Science and Technology Corporation,laslo.diosady@nasa.gov
‡scott.m.murman@nasa.gov

primal state makes the cost balance between primal and dual solutions depend highly on the implementation. One of the main contributors to the computational cost is forming and storing the residual Jacobian. This cost is twofold, as storing the Jacobian not only requires a large amount of memory but also hampers the computational efficiency due to memory bandwidth usage.

In this work, we avoid these issues by implementing the transpose-Jacobian action on a vector to use the existing matrix-free Krylov infrastructure[18] as the adjoint solver. This work extends our current solver capability that has been demonstrated to achieve high orders of accuracy (larger than 8^{th}-order) on turbulent flows.[19,20] We will use this work as a building block for our goal of output-based space-time *hp*-adaptation which will allow us to perform LES at significantly lower computational cost than currently. In this paper, we will concentrate on the adjoint solver and the adaptation mechanics will be covered in future papers. This abstract is organized as follows. In Section II, we describe the discretization followed by the primal solution strategy presented in Section III. Section IV presents the derivation of the adjoint equation and Section V presents preliminary results. We present the plan for the final manuscript in Section VI.

II. Discretization

For completeness, we briefly present the discretization of the flow equations. More details are found in Ref. 18. The compressible Navier-Stokes equations are written in conservative form as:

$$\boldsymbol{q}_{,t} + \nabla \cdot (\boldsymbol{f}^I - \boldsymbol{f}^V) = \boldsymbol{g}, \tag{1}$$

where $(\cdot)_{,t}$ denotes partial differentiation with respect to time. The conservative state vector is

$$\boldsymbol{q} = \begin{bmatrix} \rho \\ \rho \boldsymbol{u} \\ \rho E \end{bmatrix}, \tag{2}$$

where ρ is the density, $\boldsymbol{u}$ is the velocity vector, and E the total energy. The inviscid and viscous fluxes are given, respectively, by:

$$\boldsymbol{f}^I = \begin{bmatrix} \rho \boldsymbol{u} \\ \rho \boldsymbol{u}\boldsymbol{u}^T + p\boldsymbol{I} \\ \rho \boldsymbol{u} H \end{bmatrix}, \qquad \text{and} \qquad \boldsymbol{f}^V = \begin{bmatrix} 0 \\ \boldsymbol{\tau} \\ \boldsymbol{\tau} \cdot \boldsymbol{u} - \kappa_T \nabla T \end{bmatrix}, \tag{3}$$

where p is the static pressure, $H = E + \frac{p}{\rho}$ is the total enthalpy, $\boldsymbol{\tau}$ the viscous stress tensor, κ_T is the thermal conductivity, $T = p/\rho R$ is the temperature, and R is the gas constant. The pressure is given by:

$$p = (\gamma - 1)\left(\rho E - \tfrac{1}{2}\rho \boldsymbol{u}^2\right), \tag{4}$$

where γ is the specific heat ratio. The viscous stress tensor, $\boldsymbol{\tau}$, is given by:

$$\tau = \mu\left(\nabla \boldsymbol{u} + \nabla \boldsymbol{u}^T\right) - \lambda(\nabla \cdot \boldsymbol{u})\boldsymbol{I}, \tag{5}$$

where μ is the viscosity, and $\lambda = \frac{2}{3}\mu$ is the bulk viscosity.

Applying a change of variables $\boldsymbol{q} = \boldsymbol{q}(\mathbf{v})$, where $\mathbf{v}$ are the entropy variables:

$$\mathbf{v} = \begin{bmatrix} -\frac{s}{\gamma-1} + \frac{\gamma+1}{\gamma-1} - \frac{\rho E}{p} \\ \frac{\rho \boldsymbol{u}}{p} \\ -\frac{\rho}{p} \end{bmatrix}, \tag{6}$$

we rewrite the Navier-Stokes equations as:

$$\boldsymbol{A}_0 \mathbf{v}_{,t} + \bar{\boldsymbol{A}} \nabla \mathbf{v} - \nabla \cdot (\bar{\bar{\boldsymbol{K}}} \nabla \mathbf{v}) \quad = \quad \boldsymbol{g}, \tag{7}$$

with symmetric $\boldsymbol{A}_0 = \boldsymbol{q}_{,\mathbf{v}}$, $\bar{\boldsymbol{A}} = \boldsymbol{f}^I_{,\boldsymbol{q}} \boldsymbol{A}_0 = \boldsymbol{f}^I_{,\mathbf{v}}$ and $\bar{\bar{\boldsymbol{K}}} = \boldsymbol{f}^V_{,\nabla \boldsymbol{q}} \boldsymbol{A}_0 = \boldsymbol{f}^V_{,\nabla \mathbf{v}}$.[21]

We proceed to discretize (7) as follows. The domain, Ω, is partitioned into non-overlapping elements, κ, while the time is partitioned into intervals (time-slabs), $I^n = [t^n, t^{n+1}]$. Define $\mathcal{V}_h = \left\{\boldsymbol{w}, \boldsymbol{w}|_\kappa \in [\mathcal{P}(\kappa \times I)]^5\right\}$,

the space-time finite-element space consisting of piece-wise polynomial functions in both space and time on each element. We seek a solution $\mathbf{v} \in \mathcal{V}_h$ such that the weak form:

$$\begin{aligned} \boldsymbol{r}(\boldsymbol{w}, \mathbf{v}) \quad = \quad & \sum_{\kappa} \Bigg\{ \int_{I^n} \int_{\kappa} - \Big(\boldsymbol{w}_{,t} \cdot \boldsymbol{q} + \nabla \boldsymbol{w} \cdot (\boldsymbol{f}^I - \boldsymbol{f}^V) + \boldsymbol{w} \boldsymbol{g} \Big) \\ & + \int_{I^n} \int_{\partial \kappa} \boldsymbol{w} \cdot (\widehat{\boldsymbol{f}^I \cdot \boldsymbol{n}} - \widehat{\boldsymbol{f}^V \cdot \boldsymbol{n}}) \\ & + \int_{\kappa} \boldsymbol{w}(t_-^{n+1}) \cdot \boldsymbol{q}(t_-^{n+1}) - \boldsymbol{w}(t_+^n) \cdot \boldsymbol{q}(t_-^n) \Bigg\} = 0, \end{aligned} \tag{8}$$

is satisfied for all $\boldsymbol{w} \in \mathcal{V}_h$. Here $\widehat{\boldsymbol{f}^I \cdot \boldsymbol{n}}$ and $\widehat{\boldsymbol{f}^V \cdot \boldsymbol{n}}$ denote numerical flux functions approximating the inviscid and viscous fluxes, respectively, while $\boldsymbol{n}$ is the outward pointing normal vector. In this work, the inviscid flux is discretized using the method of Ismail and Roe,[22] while the viscous flux is discretized using the second method of Bassi and Rebay.[23]

The space $\mathcal{V}_h$ is represented using a tensor-product basis such that on each element $\mathbf{v}$ is given by the product of Lagrange polynomials:

$$\mathbf{v}(\mathbf{x}(\xi), t(\tau)) = \mathbf{V}_{ijkl} \Phi_{ijkl} \qquad \text{with} \qquad \Phi_{ijkl} = \phi_i(\xi_1)\phi_j(\xi_2)\phi_k(\xi_3)\phi_l(\tau), \tag{9}$$

where $\mathbf{x}(\xi)$ defines a mapping from the reference cube, $\xi \subset [-1,1]^3$, to physical space, while $t(\tau)$ is the mapping from the reference interval $[-1,1]$ to the time interval $[t^n, t^{n+1}]$. ϕ_i are one-dimensional Lagrange basis functions defined at Gauss-Legendre (GL) points, while $\mathbf{V}_{ijkl}$ are the corresponding nodal values of the entropy variables. The integrals in (8) are evaluated using numerical quadrature. For example:

$$\begin{aligned} & \frac{2}{\Delta t} \int_{I^n} \int_{\kappa} - \Big(\boldsymbol{w}_{,t} \cdot \boldsymbol{q} + \nabla \boldsymbol{w} \cdot (\boldsymbol{f}^I - \boldsymbol{f}^V)) \Big) \\ \simeq \quad & \Big\{ - \Big(\tau_{,t} \boldsymbol{w}_{,\tau} \cdot \boldsymbol{q} + \nabla_{\xi} \boldsymbol{w} \cdot (\tilde{\boldsymbol{f}}^I - \tilde{\boldsymbol{f}}^V)) \Big) |J| \Big\}_{\xi_p \xi_q \xi_r \tau_s} w_p w_q w_r w_s, \end{aligned} \tag{10}$$

where $\xi_p, \xi_q, \xi_r, \tau_s$ are one-dimensional GL quadrature points, and w_p, w_q, w_r and w_s are the associated quadrature weights. $|J|$ denotes the Jacobian of the mapping from element reference space to physical space, ∇_{ξ} denotes differentiation with respect to the reference coordinate ξ, while $\tilde{\boldsymbol{f}}^I = \xi_{,\mathbf{x}} \boldsymbol{f}^I$ and $\tilde{\boldsymbol{f}}^V = \xi_{,\mathbf{x}} \boldsymbol{f}^V$ are the fluxes mapped to the local element coordinate system. In this work we use a quadrature rule with twice as many quadrature points as nodal points in order to reduce the quadrature error (we ensure exact integration of cubic nonlinearities) thereby improving the nonlinear stability of our scheme.[24,25]

The remaining integrals appearing in (8) are evaluated in a similar manner, which may be described as a sequence of three steps:

1. Evaluate the state ($\mathbf{v}$) and gradient ($\nabla_{\xi} \mathbf{v}$) at the quadrature points.
2. Evaluate the source ($\boldsymbol{g}$) and fluxes ($\tilde{\boldsymbol{f}}^I$ and $\tilde{\boldsymbol{f}}^V$) at the quadrature points.
3. Multiply the source and fluxes with the basis functions ($\boldsymbol{w}$) or gradients ($\nabla_{\xi} \boldsymbol{w}$).

A key requirement for efficiency at high polynomial order is the evaluation of the first and third steps using the sum-factorization approach,[25,26] which allows the multiplication of the basis functions to be performed as a sequence of one-dimensional operations. This results in a residual evaluation cost which scales as $O(N^{d+1})$ for each space-time element where N is the solution order and d is the number of spatial-temporal dimensions (for unsteady 3D simulations $d = 4$). For a fixed number of spatial-temporal degrees of freedom (i.e. fixed $DOF = N^d$) the residual evaluation scales linearly with the solution order. However for moderate solution orders, $N = 4 - 16$, the increased operation count with solution order may be offset by the use of optimized numerical kernels[25] via SIMD vectorization. As vector length increases in future processor architectures, the aforementioned solution order range can further increase.

III. Primal Solution Strategy

Given the choice of basis functions and quadrature rule, Equation (8) represents a globally coupled system of nonlinear equations which need to be solved for each time-slab. For large three-dimensional simulations

the cost of storing the linearization for a single step of an implicit scheme may be prohibitively expensive. Using a space-time formulation, this storage cost is further scaled by the order of the basis used in the temporal direction. In general, all degrees of freedom within an element are coupled, leading to a storage cost which scales as $O(N^d)$ for a fixed number of degrees of freedom. We overcome the memory requirement limitation by using a matrix-free Newton-Krylov method.

A restarted GMRES method is used as Krylov solver, which does not require the explicit storage of the Jacobian matrix.[27] GMRES requires only the application of the linearization to each search direction, i.e. we need to compute the linearized residual in the search direction. In this work, the linearized residual is computed directly. As with the residual evaluation, the terms in the evaluation of the linearized residual in a search direction, $\boldsymbol{s}$, are computed as a sequence of three steps:

1. Evaluate the state ($\mathbf{v}$), gradient ($\nabla_\xi \mathbf{v}$), linearized state ($\boldsymbol{s}$) and linearized gradient ($\nabla_\xi \boldsymbol{s}$) at the quadrature points.

2. Evaluate the linearized source $\boldsymbol{g}^{\text{Lin}} = \frac{\partial \boldsymbol{g}}{\partial \mathbf{v}}\boldsymbol{s} + \frac{\partial \boldsymbol{g}}{\partial \nabla_\xi \mathbf{v}}\nabla_\xi \boldsymbol{s}$ and linearized fluxes ($\tilde{\boldsymbol{f}}^{\text{Lin}} = \frac{\partial \tilde{\boldsymbol{f}}}{\partial \mathbf{v}}\boldsymbol{s} + \frac{\partial \tilde{\boldsymbol{f}}}{\partial \nabla_\xi \mathbf{v}}\nabla_\xi \boldsymbol{s}$) at the quadrature points.

3. Multiply of the linearized source and fluxes with the basis functions ($\boldsymbol{w}$) or gradients ($\nabla_\xi \boldsymbol{w}$).

This approach is more expensive than the finite-difference approach often used in Jacobian-free method,[28] however it is insensitive to a step-size parameter, provides the exact linearization, and it is used to compute the solution of adjoint (dual) problems. Note that steps 1 and 3 are transpose of each other, hence, only the flux Jacobians (step 2) need to be transposed for the adjoint residual operator.

The sum factorization approach is used in the application of steps 1 and 3, such that the cost of applying the linearization scales as $O(N)$ for a fixed number of space-time degrees of freedom. We note that with increasing solution order this is more efficient than storing the linearization and computing the linearized residual as a matrix-vector product whose cost scales as $O(N^d)$ (even if there was no cost associated with forming the linearization).

Furthermore, this approach allows us to use different quadrature rules for computing the linearized and nonlinear residuals. In particular, we often use a lower order, or collocated, quadrature rule in the evaluation of the linearized residuals of our Newton-Krylov scheme. The nonlinear residual, however, is evaluated with a de-aliased quadrature. The use of a collocated quadrature for the linearized residual introduces a linearization error hence our Newton method may be viewed a inexact.

IV. Discrete Adjoint Equation

Consider an output $\mathcal{J}(\boldsymbol{\alpha}, \mathbf{V})$, where the state $\mathbf{V}$ satisfies the discrete residual equations $\mathcal{R}(\boldsymbol{\alpha}, \mathbf{V}) = 0$ for a parameter set $\boldsymbol{\alpha}$. We seek variations of the output that remain in the space of realizable solutions to the discrete residual equations. Assuming the output functional is at least C^1-continuous and the residual equations admit a single solution for a given $\boldsymbol{\alpha}$, output variations must be balanced by corresponding residual variations. We can relate these variations by forming the output Lagrangian,

$$\mathcal{L}(\boldsymbol{\alpha}, \mathbf{V}, \boldsymbol{\Psi}) = \mathcal{J}(\boldsymbol{\alpha}, \mathbf{V}) + \boldsymbol{\Psi}^T \mathcal{R}(\boldsymbol{\alpha}, \mathbf{V}) \tag{11}$$

and force its stationarity for arbitrary $\delta\mathbf{V}$ and $\delta\boldsymbol{\alpha}$:

$$\delta\mathcal{L}((\boldsymbol{\alpha}), \mathbf{V}, \boldsymbol{\psi}) = \underbrace{\left(\left.\frac{\partial \mathcal{J}}{\partial \mathbf{V}}^T\right|_{\boldsymbol{\alpha}} + \boldsymbol{\psi}^T \left.\frac{\partial \mathcal{R}}{\partial \mathbf{V}}\right|_{\boldsymbol{\alpha}} \right)}_{\text{adjoint equation}} \delta\mathbf{V} + \underbrace{\left(\left.\frac{\partial \mathcal{J}}{\partial \boldsymbol{\alpha}}^T\right|_{\mathbf{V}} + \boldsymbol{\psi}^T \left.\frac{\partial \mathcal{R}}{\partial \boldsymbol{\alpha}}\right|_{\mathbf{V}} \right)}_{\text{sensitivity equation}} \delta\boldsymbol{\alpha} = 0. \tag{12}$$

We solve the adjoint equation for $\boldsymbol{\Psi}$ and use its value to compute the output sensitivity with respect to $\boldsymbol{\alpha}$ via the sensitivity equation. A generalization of the sensitivity equation can be derived in variation form to yield the adjoint-weighted error estimate.[4] An integral part of solving the adjoint equation is the transposed action of the residual Jacobian onto a search direction. We perform this operation in a matrix-free manner to prevent the computational cost from scaling super-linearly with solution order.

As with the primal solve, we use a GMRES method which does not require the explicit storage of the Jacobian matrix; only the application of the adjoint operator to each search direction. As with the primal

solve, the adjoint residual is computed exactly. The terms in the evaluation of the adjoint residual in a search direction, $\boldsymbol{s}$, are again computed in a sequence of three steps:

1. Evaluate the state ($\mathbf{v}$), gradient ($\nabla_\xi \mathbf{v}$), linearized state ($\boldsymbol{s}$) and linearized gradient ($\nabla_\xi \boldsymbol{s}$) at the quadrature points.

2. Evaluate the adjoint source ($\boldsymbol{g}^{\text{Adj}} = \frac{\partial \boldsymbol{g}}{\partial \mathbf{v}}^T \boldsymbol{s} + \frac{\partial \tilde{\boldsymbol{f}}}{\partial \mathbf{v}}^T \boldsymbol{s}$) and fluxes ($\tilde{\boldsymbol{f}}^{\text{Adj}} = \frac{\partial \boldsymbol{g}}{\partial \nabla_\xi \mathbf{v}}^T \nabla_\xi \boldsymbol{s} + \frac{\partial \tilde{\boldsymbol{f}}}{\partial \nabla_\xi \mathbf{v}}^T \nabla_\xi \boldsymbol{s}$) at the quadrature points.

3. Multiply the adjoint sources and fluxes with the basis functions ($\boldsymbol{w}$) or gradients ($\nabla_\xi \boldsymbol{w}$).

We note that steps 1 and 3 are identical to those for the primal problem, while step 2 involves simply the transpose of the operations computed in the linearized residual.

The Fréchet derivatives of the flux at the quadrature points from Section III are not explicitly constructed in our implementation. Instead, we express them as a sequence of algebraic operations and the implementation of their transpose follows an approach similar to the reverse-mode automatic differentiation.

Suppose we compute $g(v)$ using the sequence of steps

$$\begin{aligned} g_1 &= g_1(v) && (13)\\ g_2 &= g_2(v, g_1) && (14)\\ g &= g(g_1, g_2) && (15) \end{aligned}$$

Then the linearized residual in a search direction s is computed using the chain rule

$$\begin{aligned} g_1^{lin} &= \frac{\partial g_1}{\partial v} s && (16)\\ g_2^{lin} &= \frac{\partial g_2}{\partial v} s + \frac{\partial g_2}{\partial g_1} g_1^{lin} && (17)\\ g &= \frac{\partial g}{\partial g_1} g_1^{lin} + \frac{\partial g}{\partial g_2} g_2^{lin} && (18) \end{aligned}$$

The adjoint residual in a search direction a is simply the reverse operation:

$$\begin{aligned} a_1^{adj} &= \frac{\partial g}{\partial g_1} a \qquad a_2^{adj} = \frac{\partial g}{\partial g_2} a && (19)\\ a^{adj} &= \frac{\partial g_2}{\partial v} a \qquad a_1^{adj} += \frac{\partial g_2}{\partial g_1} && (20)\\ a^{adj} &+= \frac{\partial g_1}{\partial v} && (21) \end{aligned}$$

Figure 1 shows that the transpose Jacobian application is slightly more expensive than the forward application. We intend to investigate the reason for this extra cost for the final version of this paper. Through optimized kernels, we can further improve the linear scaling provided by the sum factorization approach and achieve approximately constant cost for the linearized residual evaluation up to 16^{th}-order.[18]

The parallelization of the adjoint solver uses the same infrastructure as the primal solver. We will describe the parallelization details in the final version of the paper.

V. Preliminary Results

We present a preliminary result for the NACA0012 airfoil at $M_\infty = 0.5,\ \alpha = 1^\circ$ and $Re = 5000$ with a 4^{th}-order spatial discretization. Figure 2 shows the primal and x-force-adjoint fields for x-momentum.

We have preliminarily verified the x-force sensitivity with respect to angle of attack with a 2^{nd}-order solution by comparing the adjoint-based sensitivity with a central-difference formula. The results for $\Delta\alpha = 0.01$rad are:

$$\text{Central difference: } -0.06454802; \quad \text{Adjoint: } -0.006722043; \quad \text{difference: } 0.7\%.$$

We suspect the difference between the two sensitivities is due to the stepsizes used in both the central-difference used for the forward sensitivity and in the computation of the righthand side of the adjoint equation.

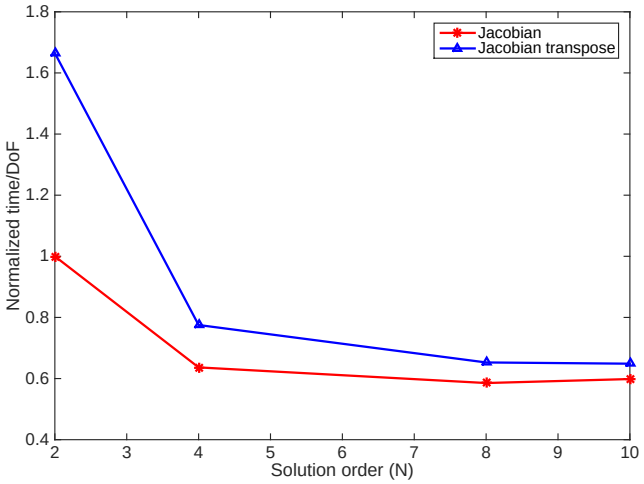


Figure 1: Computational cost per degree of freedom of the forward and transpose Fréchet derivatives.

VI. Plan for Manuscript

For the final version of this manuscript we will present two verification cases for the adjoint implementation. The first case will be steady laminar flow over the NACA0012 airfoil at $M_\infty = 0.5$, $\alpha = 1^\circ$ and $Re = 5000$. We will perform the calculations at varying solution approximation orders and we will use the sensitivity equation to verify the exact discrete sensitivity of the drag and lift outputs with respect to angle of attack. The second case will be unsteady three-dimensional turbulent flow over the MDA30p30n airfoil at high angle of attack. We will verify the exact discrete sensitivity of the integrated lift over time with respect to a variation in angle of attack.

References

[1]Leschziner, M., "Modelling turbulent separated flow in the context of aerodynamic applications," *Fluid Dynamics Research*, Vol. 38, No. 2-3, 2006, pp. 174–210.

[2]Levy, D. W., Laflin, K. R., Tinoco, E. N., Vassberg, J. C., Mani, M., Rider, B., Rumsey, C. L., Wahls, R. A., Morrison, J. H., Brodersen, O. P., Crippa, S., Mavriplis, D. J., and Murayama, M., "Summary of Data from the Fifth Computational Fluid Dynamics Drag Prediction Workshop," *Journal of Aircraft*, 2014.

[3]Slotnik, J., Khodadoust, A., Alonso, J., Darmofal, D., Gropp, W., Lurie, E., and Mavriplis, D., "CFD Vision 2030 Study: A Path to Revolutionary Computational Sciences," Technical Report 20140003093, NASA, 2014.

[4]Fidkowski, K. J. and Darmofal, D. L., "Review of Output-Based Error Estimation and Mesh Adaptation in Computational Fluid Dynamics," *AIAA Journal*, Vol. 49, No. 4, 2011, pp. 673–694.

[5]Bey, K. S., *An hp-adaptive discontinuous Galerkin method for hyperbolic conservation laws*, PhD dissertation, University of Texas at Austin, 1994.

[6]Heuveline, V. and Rannacher, R., "Duality-based Adaptivity in the *hp*-finite element method," *Journal of Numerical Mathematics*, Vol. 11, No. 2, 2003, pp. 95–113.

[7]Rachowicz, W., Pardo, D., and Demkowicz, L., "Fully Automatic *hp*-Adaptivity in Three Dimensions," Tech. Rep. 04-22, ICES, 2004.

[8]Houston, P. and Süli, E., "A note on the design of *hp*-adaptive finite element methods for elliptic partial differential equations," *Computer Methods in Applied Mechanics and Engineering*, Vol. 194, 2005, pp. 229–243.

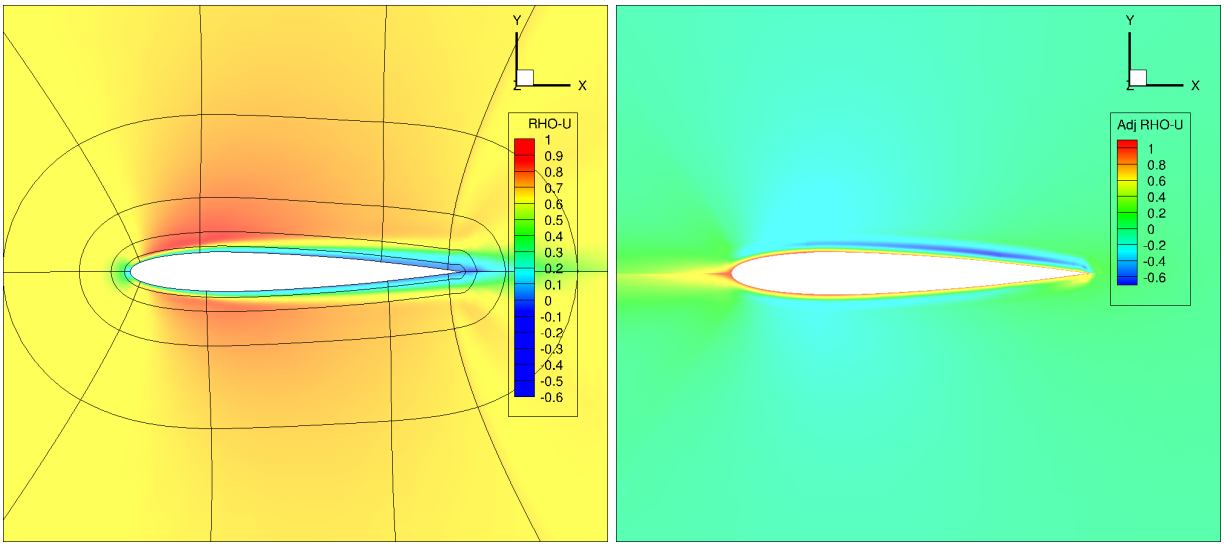
[9]Giani, S. and Houston, P., "High–Order *hp*–Adaptive Discontinuous Galerkin Finite Element Methods for Compressible Fluid Flows," *ADIGMA - A European Initiative on the Development of Adaptive Higher-Order Variational Methods for Aerospace Applications*, edited by N. Kroll, H. Bieler, H. Deconinck, V. Couaillier, H. van der Ven, and K. Sørensen, Vol. 113 of *Notes on Numerical Fluid Mechanics and Multidisciplinary Design*, Springer Berlin / Heidelberg, 2010, pp. 399–411.

[10]Burgess, N. K. and Mavriplis, D. J., "An *hp*-Adaptive Discontinuous Galerkin Solver for Aerodynamic Flows on Mixed-Element Meshes," *49th AIAA Aerospace Sciences Meeting and Exhibit*, No. AIAA 2011-490, 2011.

[11]Ceze, M. and Fidkowski, K. J., "Anisotropic *hp*-adaptation framework for functional prediction," *AIAA Journal*, Vol. 51, No. 2, February 2013, pp. 492–509.

[12]Park, M. A., *Anisotropic Output-Based Adaptation with Tetrahedral Cut Cells for Compressible Flows*, PhD dissertation, Massachusetts Institute of Technology, Cambridge, Massachusetts, 2008.

[13]Loseille, A. and Löhner, R., "Anisotropic Adaptive Simulations in Aerodynamics," *48th AIAA Aerospace Sciences Meeting and Exhibit*, No. AIAA 2010-169, 2010.



(a) $N = 4$ primal solution.

(b) $N = 4$ x-force-adjoint x-momentum solution.

Figure 2: 4^{th}-order primal and x-force adjoint solution.

[14]Yano, M., *An Optimization Framework for Adaptive Higher-Order Discretizations of Partial Differential Equations on Anisotropic Simplex Meshes*, PhD dissertation, Massachusetts Institute of Technology, Cambridge, Massachusetts, June 2012.

[15]Barth, T., *Space-time error representation and estimation in Navier-Stokes calculations*, Vol. 26 of *Lecture Notes in Computational Science and Engineering*, Springer Berlin Heidelberg, 2007, pp. 29–48.

[16]Taube, A., Gassner, G., and Munz, C.-D., "HP-Adaptation in Space-Time within an Explicit Discontinuous Galerkin Framework," *ADIGMA - A European Initiative on the Development of Adaptive Higher-Order Variational Methods for Aerospace Applications*, edited by N. Kroll, H. Bieler, H. Deconinck, V. Couaillier, H. van der Ven, and K. Sørensen, Vol. 113, Springer Berlin Heidelberg, 2010, pp. 427–439.

[17]Fidkowski, K. J. and Luo, Y., "Output-Based Space-Time Mesh Adaptation for the Compressible Navier-Stokes Equations," *Journal of Computational Physics*, Vol. 230, 2011, pp. 5753–5773.

[18]Diosady, L. T. and Murman, S. M., "Design of a Variational Multiscale Method for Turbulent Compressible Flows," *21th AIAA Computaional Fluid Dynamics Conference*, 2013.

[19]Diosady, L. T. and Murman, S. M., "DNS of flows over periodic hills using a discontinuous Galerkin spectral element method," AIAA paper, 2014, accepted for publication.

[20]Diosady, L. T. and Murman, S. M., "Higher-Order Methods for Compressible Turbulent Flows Using Entropy Variables," *53rd AIAA Aerospace Sciences Meeting and Exhibit*, 2015.

[21]Hughes, T. J. R., Franca, L., and Mallet, M., "A new finite element formulation for computational fluid dynamics: I Symmetric forms of the compressible Euler and Navier-Stokes equations and the second law of thermodynamics," Vol. 54, 1986, pp. 223–234.

[22]Ismail, F. and Roe, P. L., "Affordable, Entropy-consistent Euler flux functions II: entropy production at shocks," *J. Comput. Phys.*, Vol. 228, No. 15, Aug. 2009, pp. 5410–5436.

[23]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.

[24]Kirby, R. M. and Karniadakis, G. E., "De-aliasing on non-uniform grids: algorithms and applications," *Journal of Computational Physics*, Vol. 191, 2003, pp. 249–264.

[25]Diosady, L. T. and Murman, S. M., "Design of a variational multiscale method for turbulent compressible flows," AIAA Paper 2013-2870, 2013.

[26]Vos, P., Sherwin, S., and Kirby, R., "From h to p Efficiently: Implementing finite and spectral/hp element discretizations to achieve optimal performance at low and high order approximations." *Journal of Computational Physics*, Vol. 229, No. 13, 2010, pp. 5161–5181.

[27]Saad, Y. and Schultz, M. H., "GMRES: A Generalized Minimal Residual Algorithm for Solving Nonsymmetric Linear Systems," *SIAM Journal on Scientific and Statistical Computing*, Vol. 7, No. 3, 1986, pp. 856–869.

[28]Knoll, D. A. and Keyes, D. E., "Jacobian-free Newton-Krylov methods: a survey of approaches and applications," *Journal of Computational Physics*, Vol. 193, No. 1, 2004, pp. 357–397.