Seventh International Conference on Computational Fluid Dynamics (ICCFD7), Big Island, Hawaii, July 9-13, 2012

ICCFD7-3802

Output-Adaptive Solution Strategies for Unsteady Aerodynamics on Deformable Domains

Steven M. Kast, Marco A. Ceze, and Krzysztof J. Fidkowski
Corresponding author: kfid@umich.edu

University of Michigan, USA

Abstract: We present an output-based adaptation strategy for high-order simulations of the compressible Navier-Stokes equations on deformable domains. The equations are solved on a mapped reference domain using an arbitrary Lagrangian-Eulerian approach. The discretization is a discontinuous Galerkin finite-element method in space and time. Discrete unsteady adjoint solutions, derived for both the state and the geometric conservation law, provide scalar output error estimates and drive adaptive refinement of the space-time mesh. Spatial adaptation is performed using dynamic order refinement on a fixed tessellation of the domain, while in the temporal domain a combination of coarsening and refinement is used to determine the most efficient time slab distribution. Results from compressible Navier-Stokes simulations demonstrate the accuracy of the error estimates and efficiency of the proposed output-based adaptation approach.

Keywords: Unsteady Adjoint, Output Error Estimation, Deformable Domains, Space-Time Mesh Adaptation, Geometric Conservation Law.

1 Introduction

As fluid dynamics simulations become more complex, estimates of discretization error are of increasing interest for improving robustness and accuracy of the computations. Adjoint-based methods are especially suited for this task as they provide reliable estimates for errors in outputs of interest. Furthermore, these estimates can be localized for adapting the mesh to improve output accuracy and computational efficiency.

The development of output-based methods has focused primarily on steady-state problems [1–7], although unsteady problems have recently begun to receive increased attention. Topics addressed thus far include temporal error estimation and adaptation [8,9], static spatial mesh adaptation [10], combined temporal and static spatial adaptation [11], and combined temporal and spatial adaptation with dynamic h and p refinement [12–14].

This work considers combined temporal refinement and dynamic spatial order adaptation for aerodynamics simulations on deformable domains. Such simulations have far-reaching applications, from bio-inspired flight to aircraft maneuver and flutter analysis. The runs are generally computationally intensive and the resulting solutions are often rich in features. We show that for these problems, output error estimation and adaptation can have a significant impact on solution accuracy and robustness.

One of the main differences between static and deformable domains is the requirement that a geometric conservation law (GCL) be satisfied in the latter cases in order to preserve a free-stream

solution. Satisfying the GCL in our case involves solving for an auxiliary variable that is co-evolved with the state. In the current work, we use the discontinuous Galerkin method for both spatial and temporal discretizations of the state and GCL.

In addition to its effect on the primal solution, the GCL also has an impact on error estimation. Since outputs in general depend on both the state and GCL variable, a separate adjoint equation must be solved for the GCL as well. In this work, discrete adjoint equations are derived for both the state and the GCL, and an adjoint-weighted residual is presented for estimating the output error.

The present research is a continuation of previous work in unsteady output-based adaptation on static meshes [11,14,15]. The discretization and error estimation extend naturally to the geometric conservation law, and the required modifications are discussed here. The outline for this paper is as follows: in Section 2 we introduce the mapping for deformable domains and discuss the GCL; in Section 3 we give the primal discretization for the state and GCL; in Section 4 we present the adjoint discretization; Section 5 contains the output error estimation; and Section 6 concerns mesh adaptation. Finally, results for the compressible Navier-Stokes equations are presented in Section 7.

2 Arbitrary Lagrangian-Eulerian Mapping

Our system of conservation laws on the physical domain is

$$\frac{\partial \mathbf{u}}{\partial t} + \nabla \cdot \vec{\mathbf{F}}(\mathbf{u}, \nabla \mathbf{u}) = \mathbf{0}, \qquad \vec{\mathbf{F}} = \vec{\mathbf{F}}^i(\mathbf{u}) - \vec{\mathbf{F}}^v(\mathbf{u}, \nabla \mathbf{u}), \tag{1}$$

where $\mathbf{u}(\vec{x}, t) \in \mathbb{R}^s$ is the state vector, $\vec{x} \in \mathbb{R}^d$ is the spatial coordinate, and $t \in \mathbb{R}$ is time. $\vec{\mathbf{F}}^i$ and $\vec{\mathbf{F}}^v$ are the inviscid and viscous fluxes, respectively. Rather than solving this system on the physical domain, we instead map it to a static reference domain using an arbitrary Lagrangian-Eulerian (ALE) approach. Specifically we use the approach presented by Persson *et al* [16].

2.1 ALE mapping

The mapping between reference and physical domains is summarized graphically in Figure 1, and definitions of relevant variables are given in Table 1.

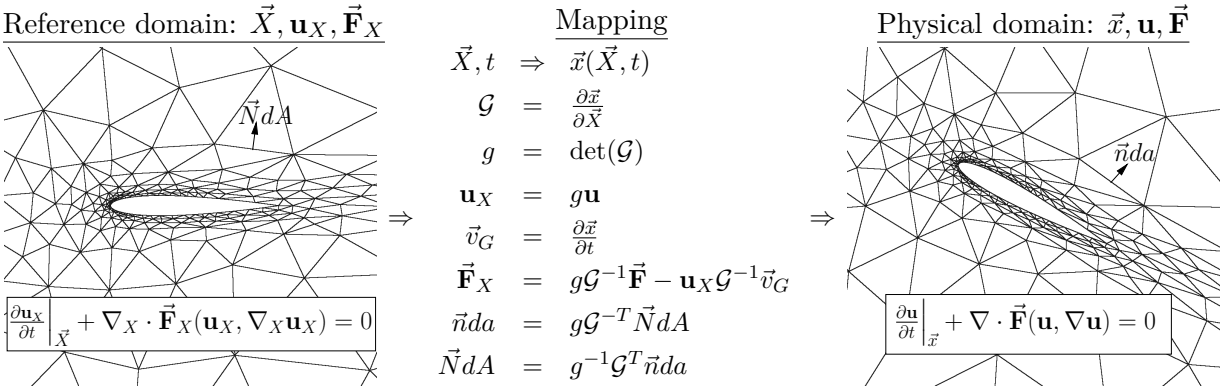


Figure 1: Summary of the mapping between reference and physical spaces.

The expression for the transformation of the normals is obtained using $dv = gdV$ for infinitesimal volumes and $d\vec{l} = \mathcal{G}d\vec{L}$ for infinitesimal vectors, as derived in [16]. We break up the transformed

Table 1: Definitions of variables used in the ALE mapping. Boldface indicates a state vector and an arrow indicates a spatial vector.

$\vec{X}$	=	reference-domain coordinates	da	=	differential area on physical domain
$\vec{x}$	=	physical-domain coordinates	dA	=	differential area on reference domain
$\mathcal{G}$	=	mapping Jacobian matrix	$\vec{v}_G$	=	grid velocity, $\partial\vec{x}/\partial t$
g	=	determinant of Jacobian matrix	$\mathbf{u}$	=	physical state
$\vec{n}$	=	normal vector on physical domain	$\mathbf{u}_X$	=	state on reference domain
$\vec{N}$	=	normal vector on reference domain	$\vec{\mathbf{F}}$	=	flux vector on physical domain
$v(t)$	=	physical domain (dynamic)	$\vec{\mathbf{F}}_X$	=	flux vector on reference domain
V	=	reference domain (static)			

flux, $\vec{\mathbf{F}}_X$, into inviscid and viscous fluxes and lump the grid-velocity term into the inviscid flux:

$$\vec{\mathbf{F}}_X = \vec{\mathbf{F}}^i_X - \vec{\mathbf{F}}^v_X, \qquad \vec{\mathbf{F}}^i_X = g\mathcal{G}^{-1}\left(\vec{\mathbf{F}}^i(\mathbf{u}) - \mathbf{u}\vec{v}_G\right), \qquad \vec{\mathbf{F}}^v_X = g\mathcal{G}^{-1}\vec{\mathbf{F}}^v(\mathbf{u}, \nabla\mathbf{u}).$$

The gradient of the state required for the viscous flux transforms via the chain and product rules. Using implied summation,

$$\begin{aligned}\nabla\mathbf{u} = \frac{\partial\mathbf{u}}{\partial x_j} = \frac{\partial(g^{-1}\mathbf{u}_X)}{\partial X_d}\frac{\partial X_d}{\partial x_j} &= \left(g^{-1}\frac{\partial\mathbf{u}_X}{\partial X_d} - g^{-2}\frac{\partial g}{\partial X_d}\mathbf{u}_X\right)\mathcal{G}^{-1}_{dj} \\ &= g^{-1}\left(\frac{\partial\mathbf{u}_X}{\partial X_d} - g^{-1}\frac{\partial g}{\partial X_d}\mathbf{u}_X\right)\mathcal{G}^{-1}_{dj}, \end{aligned} \tag{2}$$

where d and j index the reference and physical coordinates, respectively. We also have

$$\mathcal{G} = \mathcal{G}_{jd} = \frac{\partial x_j}{\partial X_d}, \qquad \delta_{ji} = \frac{\partial x_j}{\partial x_i} = \frac{\partial x_j}{\partial X_d}\frac{\partial X_d}{\partial x_i} = \mathcal{G}_{jd}\mathcal{G}^{-1}_{di}, \qquad \Rightarrow \quad \mathcal{G}^{-1} = \mathcal{G}^{-1}_{di} = \frac{\partial X_d}{\partial x_i}.$$

Once the fluxes and states have been transformed in the above manner, the new equations to be satisfied on the reference domain are simply

$$\frac{\partial\mathbf{u}_X}{\partial t}\Big|_X + \nabla_X \cdot \vec{\mathbf{F}}_X(\mathbf{u}_X, \nabla_X\mathbf{u}_X) = \mathbf{0}, \tag{3}$$

and the solution procedure is the same as if the mesh were not deforming at all.

2.2 The Geometric Conservation Law

A desirable property of numerical schemes is the ability to preserve a free-stream state. However, for arbitrary mappings in the method described above, a constant state in the physical domain ($\mathbf{u} = \bar{\mathbf{u}} =$ constant) will not be an exact solution to Eqn. 3 in the reference domain. This means that in the presence of mesh motion, a free-stream will not be preserved.

The first thing to note is that for general mappings, the quantity $g\bar{\mathbf{u}}$ will be non-polynomial, which means it cannot be represented exactly with the reference-domain bases. This will introduce spatial errors into an initially free-stream state. In addition, conservation errors in time can also arise if the change in element size is not exactly integrated by the numerical scheme.

Persson *et al* [16] describe one technique, a geometric conservation law, for addressing these issues. This technique relies on using $\mathbf{u}_{\bar{X}} = \bar{g}\mathbf{u} = \bar{g}g^{-1}\mathbf{u}_X$ as the reference-domain state instead of

$\mathbf{u}_X$, where $\bar{g}$ is a separate variable satisfying the following equation:

$$\frac{\partial \bar{g}}{\partial t} - \nabla_X \cdot (g\mathcal{G}^{-1}\vec{v}_G) = 0. \tag{4}$$

This equation comes from inserting a constant state $\bar{\mathbf{u}}$ into Eqn. 3 and observing what remains on the left-hand side. If $\bar{g}$ is approximated using the same spatial basis as the state and is marched according to Eqn. 4 using the same unsteady solver, the reference state $\mathbf{u}_{\bar{X}} = \bar{g}\bar{\mathbf{u}}$ is then a representable solution in the discrete approximation space.

Implementation of the GCL requires evolving this additional scalar $\bar{g}$, which is now used to convert the stored reference state to the physical state. The final form of the reference-domain equation is then

$$\frac{\partial \mathbf{u}_{\bar{X}}}{\partial t}\Big|_X + \nabla_X \cdot \vec{\mathbf{F}}_{\bar{X}}(\mathbf{u}_{\bar{X}}, \nabla_X \mathbf{u}_{\bar{X}}) \quad = \quad 0, \tag{5}$$

where $\vec{\mathbf{F}}_{\bar{X}}$ is just $\vec{\mathbf{F}}_X$ but with $\mathbf{u}_{\bar{X}}$ as input rather than $\mathbf{u}_X$. Note that the expression for the physical gradient in Eqn. 2 changes when the stored reference state is $\mathbf{u}_{\bar{X}}$ instead of $\mathbf{u}_X$:

$$\nabla \mathbf{u} = \frac{\partial \mathbf{u}}{\partial x_j} = \frac{\partial(\bar{g}^{-1}\mathbf{u}_{\bar{X}})}{\partial X_d}\frac{\partial X_d}{\partial x_j} \quad = \quad \bar{g}^{-1}\left(\frac{\partial \mathbf{u}_{\bar{X}}}{\partial X_d} - \bar{g}^{-1}\frac{\partial \bar{g}}{\partial X_d}\mathbf{u}_{\bar{X}}\right) G^{-1}_{dj}. \tag{6}$$

Figure 2 shows the effect of the GCL on a free-stream preservation test. In this case, which is similar to the one in [16], the Navier-Stokes equations are solved on a rectangular domain with an analytical sinusoidal mapping defined in the domain interior. The temporal and spatial discretization are both discontinuous Galerkin, with order $r = 1$ and p, respectively.

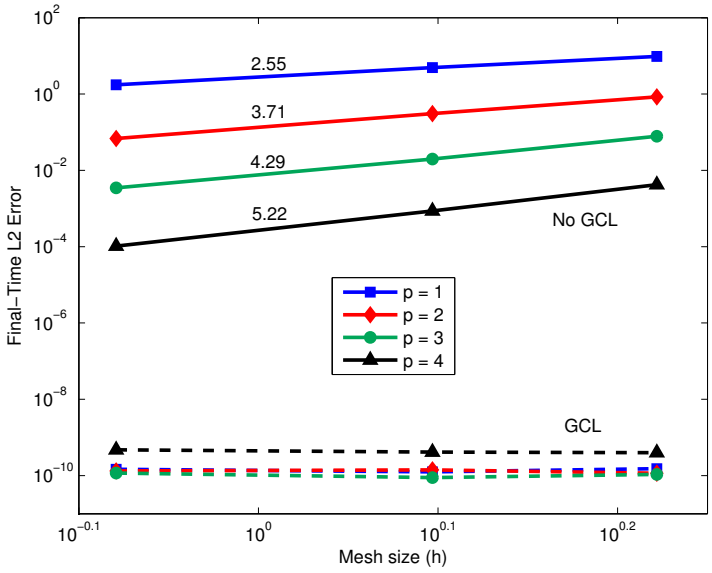


Figure 2: Free-stream errors with and without the GCL. A large number of time steps is used so that the spatial error dominates the temporal error in each case.

Without the GCL, the free-stream solution is not preserved, though the error (shown in an

L_2 state norm) converges with both h and p refinement of the spatial mesh. With the GCL, however, the free-stream is preserved to residual tolerance, which was approximately ten orders of magnitude for all runs. We note that to achieve this level of accuracy with the GCL, very high-order quadrature rules are required. To demonstrate that the GCL is working properly, we used rules of order $6p$ for this case. In practical cases we use more modest rules, namely $2p + 5$ for the Navier-Stokes equations, since in such cases discretization errors tend to dominate, making extremely high quadrature rules unnecessary.

2.3 Analytical Mesh Motion

To simulate problems of physical interest, we employ analytical mesh motions obtained by blending superimposed rigid-body motions in circular disks around the objects of interest [16]. For example, an airfoil situated inside of a rigid-body disk can be made to pitch and plunge in an analytically-prescribed manner (see Figure 3).

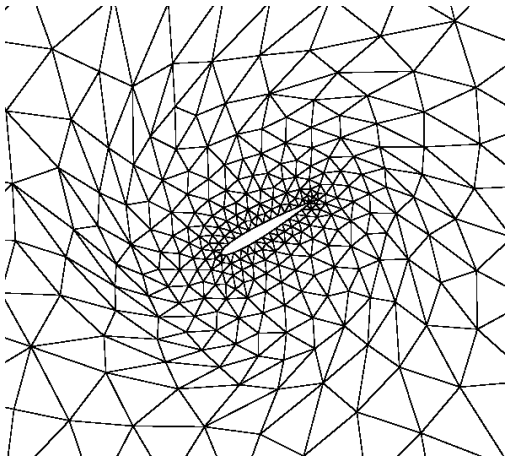


Figure 3: Airfoil undergoing a blended rigid-body pitch/plunge.

A polynomial blending function is used to transition smoothly from the rigid body motion into the static mesh further out. For the cases in this work, we use quintic polynomials.

Since the entire motion is described analytically, g, $\mathcal{G}$, and $\vec{v}_G$ can be evaluated at any point in space, $\vec{x}$, and time, t. This greatly simplifies the calculation of the GCL residual (Eqn. 4), since the GCL flux is an analytical function that is independent of $\bar{g}$.

3 Primal Discretization

We use a discontinuous Galerkin finite element method in both space and time to discretize Eqn. 5 and, when using the GCL, Eqn. 4. The temporal discretization is restricted to slabs on which the temporal variation is approximated with polynomials of order r. This means that all elements advance at the same time step, Δt, which can vary in time. However, the spatial approximation order need not be the same for all spatial elements in a given time slab, nor for all time slabs corresponding to a given spatial element.

3.1 Approximation

Each space-time element is identified by two indices, (e, k), where e is the spatial element index and k is the time slab index. $p_{e,k}$ denotes the order of approximation on space-time element (e, k).

On each space-time element, the state and GCL variable are approximated as

$$\mathbf{u}_{\bar{X},H}(\vec{X},t)\Big|_{e,k} = \underbrace{\mathbf{u}^{k,n}_{\bar{X},H,e,j}}_{\in\mathbb{R}^s}\underbrace{\phi^k_{H,e,j}(\vec{X})}_{\text{order } p_{e,k}}\underbrace{\varphi^n_H(t)}_{\text{order r}}, \tag{7}$$

$$\bar{g}_H(\vec{X},t)\Big|_{e,k} = \underbrace{\bar{g}^{k,n}_{H,e,j}}_{\in\mathbb{R}^1}\underbrace{\phi^k_{H,e,j}(\vec{X})}_{\text{order } p_{e,k}}\underbrace{\varphi^n_H(t)}_{\text{order r}}, \tag{8}$$

where $1 \le j \le \text{dof}(p_{e,k})$ is the spatial degree-of-freedom index on space-time element (e,k) and $1 \le n \le r{+}1$ is the temporal degree of freedom index on time slab k. The number of spatial degrees of freedom depends on the approximation basis and the dimension; e.g. for full-order approximation in two dimensions, $\text{dof}(p_{e,k}) = (p_{e,k}{+}1)(p_{e,k}{+}2)/2$. Note that the spatial basis functions, $\phi^k_{H,e,j}(\vec{X})$, are specific to an element and time slab, while the temporal basis functions, $\varphi^n_H(t)$, are the same for each time slab. In this work, we use Lagrange bases in both space and time.

For compactness of notation, we lump all spatial degrees of freedom associated with time node n on slab k into one vector, for both the state and the GCL variable,

$$\mathbf{U}^{k,n}_H = \left\{\mathbf{u}^{k,n}_{\bar{X},H,e,j}\right\}_{\forall e,j} \in \mathbb{R}^{sN^k_H}, \tag{9}$$

$$\mathbf{G}^{k,n}_H = \left\{\bar{g}^{k,n}_{H,e,j}\right\}_{\forall e,j} \in \mathbb{R}^{N^k_H}, \tag{10}$$

where $N^k_H = \sum_e \text{dof}(p_{e,k})$ is the total number of spatial degrees of freedom on time slab k. As a shorthand, we will denote by $\mathbf{U}^k_H$ and $\mathbf{G}^k_H$ the sets of unknowns over an entire time slab, k.

3.2 Residuals

A nonlinear system of equations on each time slab is obtained by substituting the approximations from Eqns. 7 and 8 into Eqns. 5 and 4, respectively, multiplying by test functions in the same space as the approximation functions, and integrating by parts to incorporate discontinuities at time slab and spatial element interfaces. These equations are expressed in terms of $r+1$ residual vectors on time slab k,

State residual:

$$\overline{\mathbf{R}}^{k,m}_{\mathbf{U},H} \equiv a^{m,n}\mathbf{M}^{k,k}_H\mathbf{U}^{k,n}_H - \varphi^m_H(t_{k-1})\mathbf{M}^{k,k-1}_H\mathbf{U}^{k-1,r+1}_H + \int_{t_{k-1}}^{t_k}\varphi^m_H(t)\mathbf{R}_{\mathbf{U},H}(\mathbf{U}^k_H(t),\mathbf{G}^k_H(t))\,dt = \mathbf{0},$$

GCL residual:

$$\overline{\mathbf{R}}^{k,m}_{\mathbf{G},H} \equiv a^{m,n}\mathbf{M}^{k,k}_H\mathbf{G}^{k,n}_H - \varphi^m_H(t_{k-1})\mathbf{M}^{k,k-1}_H\mathbf{G}^{k-1,r+1}_H + \int_{t_{k-1}}^{t_k}\varphi^m_H(t)\mathbf{R}_{\mathbf{G},H}(t)\,dt = \mathbf{0}, \tag{11}$$

where $1 \le m \le r{+}1$, and $a^{m,n}$ are time-slab-independent coefficients that are defined by

$$a^{m,n} = -\int_{t_{k-1}}^{t_k}\varphi^n_H\frac{d\varphi^m_H}{dt}\,dt + \varphi^n_H(t_k)\varphi^m_H(t_k). \tag{12}$$

The temporal approximations of the state and GCL variable on time slab k are given by $\mathbf{U}^k_H(t) = \sum_n \mathbf{U}^{k,n}_H\varphi^n_H(t)$ and $\mathbf{G}^k_H(t) = \sum_n \mathbf{G}^{k,n}_H\varphi^n_H(t)$, respectively, and the spatial state and GCL residuals are given by $\mathbf{R}_{\mathbf{U},H} \in \mathbb{R}^{sN_H}$ and $\mathbf{R}_{\mathbf{G},H} \in \mathbb{R}^{N_H}$. We do not focus on the details of the spatial discretization and only mention that it is a discontinuous Galerkin method employing the Roe

inviscid flux [17] and the second form of Bassi and Rebay for the viscous flux [18]. Note that the spatial state residual depends on both the state and the GCL, while the spatial GCL residual depends only on time and analytical terms.

Given two time slabs k and l, $\mathbf{M}_H^{k,l}$ is the mass matrix formed from the spatial basis functions on each slab, which need not be the same due to the possibility of dynamic order refinement. We slightly abuse matrix-vector product notation in Eqn. 11 since $\mathbf{U}_H^{k,n}$ and $\mathbf{G}_H^{k,n}$ are of different size – for discretized systems the mass matrix is understood to act on each equation separately.

3.3 Implementation

Since the state residual depends on the GCL variable, the GCL is advanced in time before advancing the state. This adds minimal cost due to the simplicity of the GCL residual and the fact that $\bar{g}_H$ is a scalar field. Because of the non-linear nature of the ALE mapping, an increment ΔQ is added to the default numerical quadrature order used to evaluate spatial integrals. For the Navier-Stokes results in this paper, we augment the original $2p+1$ quadrature order by an amount $\Delta Q = 4$, unless otherwise noted.

4 Discrete Adjoint

Typically, outputs of interest such as lift or drag are functions of the physical state, $\mathbf{u}$. However, from the transformations defined in Section 2, the physical state is a function of both the reference state $\mathbf{u}_{\bar{X}}$ and the GCL variable $\bar{g}$. Therefore, outputs will be sensitive to perturbations in both of these variables, and to correctly account for this sensitivity, adjoint equations must be derived for the GCL variable in addition to the state.

The coupling between the state and GCL residuals in Eqn. 11 will be reflected in the adjoint system. To derive the discrete adjoint for the state and GCL, we linearize the discrete residuals in Eqn. 11. The sparsity patterns of the complete Jacobian matrix and its transpose are shown in Figure 4.

Jacobian matrix

	$\mathbf{U}^1$	$\mathbf{G}^1$	$\mathbf{U}^2$	$\mathbf{G}^2$	$\mathbf{U}^3$	$\mathbf{G}^3$
$\mathbf{R}_\mathbf{U}^1$	•	•				
$\mathbf{R}_\mathbf{G}^1$		•				
$\mathbf{R}_\mathbf{U}^2$	•		•	•		
$\mathbf{R}_\mathbf{G}^2$		•		•		
$\mathbf{R}_\mathbf{U}^3$			•		•	•
$\mathbf{R}_\mathbf{G}^3$				•		•

Jacobian matrix transpose

	$\mathbf{R}_\mathbf{U}^1$	$\mathbf{R}_\mathbf{G}^1$	$\mathbf{R}_\mathbf{U}^2$	$\mathbf{R}_\mathbf{G}^2$	$\mathbf{R}_\mathbf{U}^3$	$\mathbf{R}_\mathbf{G}^3$
$\mathbf{U}^1$	•		•			
$\mathbf{G}^1$	•	•		•		
$\mathbf{U}^2$			•		•	
$\mathbf{G}^2$			•	•		•
$\mathbf{U}^3$					•	
$\mathbf{G}^3$					•	•

Figure 4: Sparsity patterns for the coupled state/GCL system in a DG-in-time discretization, shown for the first three time slabs. We drop subscripts H for clarity.

We denote by $\mathbf{\Psi}_{\mathbf{U},H}^{l,m}$ and $\mathbf{\Psi}_{\mathbf{G},H}^{l,m}$ the discrete adjoint vectors corresponding to the state and GCL, respectively. These adjoints represent sensitivities of a chosen output $J_H(\mathbf{U}_H^{k,n}, \mathbf{G}_H^{k,n})$ to residual source perturbations added to Eqn. 11. As mentioned, outputs of engineering interest are generally functions of the physical state, which means that dependence on both $\mathbf{U}_H^{k,n}$ and $\mathbf{G}_H^{k,n}$ must be taken into account when linearizing J_H.

4.1 Adjoint Residuals

The coupled discrete adjoint system is obtained by multiplying the transposed Jacobian matrix by the adjoint state and adding the output linearization. Considering the sparsity pattern in

Figure 4, the result for our DG-in-time discretization is

$$\left(\frac{\partial \overline{\mathbf{R}}^k_{\mathbf{U},H}}{\partial \mathbf{U}^k_H}\right)^T \mathbf{\Psi}^k_{\mathbf{U},H} + \left(\frac{\partial \overline{\mathbf{R}}^{k+1}_{\mathbf{U},H}}{\partial \mathbf{U}^k_H}\right)^T \mathbf{\Psi}^{k+1}_{\mathbf{U},H} + \left(\frac{\partial J_H}{\partial \mathbf{U}^{k,n}_H}\right)^T = 0, \quad (13)$$

$$\left(\frac{\partial \overline{\mathbf{R}}^k_{\mathbf{U},H}}{\partial \mathbf{G}^k_H}\right)^T \mathbf{\Psi}^k_{\mathbf{U},H} + \left(\frac{\partial \overline{\mathbf{R}}^k_{\mathbf{G},H}}{\partial \mathbf{G}^k_H}\right)^T \mathbf{\Psi}^k_{\mathbf{G},H} + \left(\frac{\partial \overline{\mathbf{R}}^{k+1}_{\mathbf{G},H}}{\partial \mathbf{G}^k_H}\right)^T \mathbf{\Psi}^{k+1}_{\mathbf{G},H} + \left(\frac{\partial J_H}{\partial \mathbf{G}^{k,n}_H}\right)^T = 0. \quad (14)$$

We see that the state adjoint governed by Eqn. 13 is oblivious to the GCL. On the other hand, the GCL adjoint governed by Eqn. 14 depends on the state adjoint. The result is that the state adjoint on a given time slab must be solved first for use in the GCL adjoint equation.

4.2 Implementation

In solving the GCL adjoint equation, Eqn. 14, the derivatives of the unsteady GCL residual with respect to the GCL variable are straightforward to obtain, as the result is just a mass matrix – see Eqn. 11. Obtaining the derivative of the output with respect to $\mathbf{G}^{k,n}_H$ is done by simply applying the chain rule on derivatives with respect to the physical state, which are already used in the state adjoint equation. The only new term is then the derivative of the state residual with respect to the GCL variable, $\frac{\partial \overline{\mathbf{R}}^k_{\mathbf{U},H}}{\partial \mathbf{G}^k_H}$. In the present work, as a preliminary measure, we use finite differences to evaluate this term. Finally, for adjoint consistency, we replace $\frac{\partial \bar{g}}{\partial X_d}$ in the state gradient (Eqn. 6) with the analytical $\frac{\partial g}{\partial X_d}$.

5 Output Error Estimation

Since the approximation space for the forward solution is finite-dimensional, the calculated output $J_H(\mathbf{U}_H, \mathbf{G}_H)$ will generally include discretization error. We estimate this error by comparing the output to one computed on a finer space, denoted by the subscript h, which consists of order enrichment in both space and time: i.e. using a spatial order of $p_{e,k}+1$ and a temporal order of $r+1$ for each space-time element e,k.

5.1 The Adjoint-Weighted Residual

We use the adjoint-weighted residual [7] to approximate the output error, which is the difference between the output computed with the coarse solution and that computed with the fine solution,

$$\begin{aligned}
\delta J &\approx J_H(\mathbf{U}_H, \mathbf{G}_H) - J_h(\mathbf{U}_h, \mathbf{G}_h) \\
&= \frac{\partial J_h}{\partial \mathbf{U}_h}\delta\mathbf{U}_h + \frac{\partial J_h}{\partial \mathbf{G}_h}\delta\mathbf{G}_h + \mathcal{R}^2 \\
&= \frac{\partial J_h}{\partial \mathbf{U}_h}\underbrace{\left[\frac{\partial \overline{\mathbf{R}}_{\mathbf{U},h}}{\partial \mathbf{U}_h}\right]^{-1}\left(-\delta\overline{\mathbf{R}}_{\mathbf{U},h} + \left[\frac{\partial \overline{\mathbf{R}}_{\mathbf{U},h}}{\partial \mathbf{G}_h}\right]\left[\frac{\partial \overline{\mathbf{R}}_{\mathbf{G},h}}{\partial \mathbf{G}_h}\right]^{-1}\delta\overline{\mathbf{R}}_{\mathbf{G},h}\right)}_{\delta\mathbf{U}_h} + \frac{\partial J_h}{\partial \mathbf{G}_h}\underbrace{\left[-\frac{\partial \overline{\mathbf{R}}_{\mathbf{G},h}}{\partial \mathbf{G}_h}\right]^{-1}\delta\overline{\mathbf{R}}_{\mathbf{G},h}}_{\delta\mathbf{G}_h} + \mathcal{R}^2 \\
&= \underbrace{\left(-\mathbf{\Psi}^{k,m}_{\mathbf{U},h}\right)^T \overline{\mathbf{R}}^{k,m}_{\mathbf{U},h}(\mathbf{U}^H_h, \mathbf{G}^H_h)}_{\text{state contribution}} + \underbrace{\left(-\mathbf{\Psi}^{k,m}_{\mathbf{G},h}\right)^T \overline{\mathbf{R}}^{k,m}_{\mathbf{G},h}(\mathbf{G}^H_h)}_{\text{GCL contribution}} + \mathcal{R}^2
\end{aligned} \quad (15)$$

where $\mathbf{U}^H_h$, $\mathbf{G}^H_h$ are injections of the coarse state and GCL variables into the fine space, $\delta\mathbf{U}_h = \mathbf{U}^H_h - \mathbf{U}_h$, $\delta\mathbf{G}_h = \mathbf{G}^H_h - \mathbf{G}_h$, and $\mathcal{R}^2$ is a neglected remainder term that is second-order in the

errors in the state, GCL, and their respective adjoints. Summation is implied on the index m, which indexes the fine-space temporal degrees of freedom, $1 \leq m \leq r+2$. We will refer to Eqn. 15 without the remainder term as δJ_{est}.

When Galerkin orthogonality holds, coarse-space approximations of the adjoints can be subtracted from the fine space adjoints appearing in Eqn. 15. Theoretically this has no effect on δJ, but in practice it minimizes errors due to converging residuals only to a finite tolerance. We note however that care must be applied when using the BR2 viscous discretization, which does not exhibit Galerkin orthogonality for coarse-space solutions injected into an order-enriched fine space. This is due to an order-dependence of the stabilization terms (the δ's) in BR2. We employ a simple remedy [19], which consists of using the coarse-space orders to approximate the stabilization terms when evaluating the fine-space residuals in Eqn. 15. In addition, as quadrature effects tend to be more pronounced for simulations on deformable domains, we use the coarse-space quadrature rules in these fine-space residuals.

The error estimate in Eqn. 15 requires an evaluation of the fine-space unsteady residual associated with the coarse solution, and the fine-space adjoint solution $\mathbf{\Psi}_h^{k,m}$. In this work we solve the fine-space adjoint equation to machine precision to minimize additional sources of error in our estimates. However, in practice, the fine-space adjoint can be computed by smoothing or reconstructing the coarse-space adjoint in order to minimize computational cost [20].

5.2 Error Localization

Since our aim is to adapt the mesh to reduce the output error, we need to first localize the error contributions to individual space-time elements in the mesh. This is done by noting that the output error estimate in Eqn. 15 can be written as a sum over all space-time elements,

$$\delta J_{\text{est}} = \sum_k \sum_e \varepsilon_{e,k}, \tag{16}$$

where the error contribution of a given space-time element (e,k) is

$$\varepsilon_{e,k} = \left(-\mathbf{Z}_e \mathbf{\Psi}_{\mathbf{U},h}^{k,m}\right)^T \mathbf{Z}_e \overline{\mathbf{R}}_{\mathbf{U},h}^{k,m}\left(\mathbf{U}_h^H\right) + \left(-\mathbf{Z}_e \mathbf{\Psi}_{\mathbf{G},h}^{k,m}\right)^T \mathbf{Z}_e \overline{\mathbf{R}}_{\mathbf{G},h}^{k,m}(\mathbf{G}_h^H), \tag{17}$$

and where summation is implied on the fine-space temporal degrees of freedom index m. $\mathbf{Z}_e$ is a mask matrix that simply restricts the adjoint-residual product to a specific element e. The error indicator is taken as the absolute value of the elemental contribution to the output error,

$$\text{error indicator} = \epsilon_{e,k} = |\varepsilon_{e,k}|.$$

This indicator identifies space-time elements that contribute most to the output error. However, it does not contain information about the source of the error, i.e. spatial or temporal. This information is obtained from a space-time anisotropy measure, which in this work is calculated in the same manner as presented in [14, 20]. Specifically, we calculate the anisotropy using separate projections of the fine-space adjoint onto semi-coarsened spatial and temporal spaces. The spatial and temporal error estimates for space-time element (e,k) are obtained by using these projected adjoints in Eqn. 17, resulting in separate $\varepsilon_{e,k}^{\text{space}}$ and $\varepsilon_{e,k}^{\text{time}}$ estimates. We use the ratio of these values to calculate the spatial/temporal error fractions on element e,k,

$$\beta_{e,k}^{\text{space}} = \frac{|\varepsilon_{e,k}^{\text{space}}|}{|\varepsilon_{e,k}^{\text{space}}| + |\varepsilon_{e,k}^{\text{time}}|}, \qquad \beta_{e,k}^{\text{time}} = 1 - \beta_{e,k}^{\text{space}}. \tag{18}$$

6 Spatial and Temporal Adaptation

The output error estimate drives an adaptive process in which the unsteady problem is solved on successively refined space-time meshes. At each adaptive iteration, the process requires forward and adjoint solutions, which become more expensive on the finer meshes. The temporal refinement preserves the slab-structure of the discretization in order to allow a solution strategy based on an approximate factorization [20]. Adaptive indicators identifying the amount of temporal error associated with each time slab and the amount of spatial error associated with each space-time element are given by

$$\text{spatial indicator on space-time element } e,k = \quad \epsilon_{e,k}^{\text{space}} \quad = \epsilon_{e,k}\beta_{e,k}^{\text{space}}, \tag{19}$$

$$\text{temporal indicator on time slab } k = \quad \epsilon_{k}^{\text{time}} \quad = \sum_{e} \epsilon_{e,k}\beta_{e,k}^{\text{time}}, \tag{20}$$

where the sum indexed by e is taken over all spatial elements.

The above indicators are used in a fixed-growth adaptive strategy in which some combination of time slabs and space-time elements are marked for coarsening or refinement. The change in the total degrees of freedom at every adaptive iteration is governed by a multiplicative growth factor, f^{growth}. When coarsening is requested, a factor f^{coarsen} of the current degrees of freedom, D^{current}, is first marked for coarsening. The coarsening and refinement budgets are then

$$\begin{aligned} B^{\text{coarsen}} &= f^{\text{coarsen}}\, D^{\text{current}}, \\ B^{\text{refine}} &= f^{\text{growth}}\, D^{\text{current}} + B^{\text{coarsen}}. \end{aligned}$$

The following greedy algorithm is used to decide which space-time elements or time slabs to refine or coarsen.

1. Define and sort a merit function separately for all space-time elements and all time slabs. The figure of merit of each refinement option is the amount of output error addressed, Eqns. 19 and 20, divided by the degrees of freedom added if the element/slab were refined. For temporal refinement, the latter is approximated as the degrees of freedom in the targeted slab k, $\text{dof}_k \equiv \sum_e \text{dof}(p_{e,k})$, and for spatial refinement as the number of additional degrees of freedom $\text{dof}(p_{e,k}+1) - \text{dof}(p_{e,k})$ associated with an order increase of element e,k.

2. Coarsening stage

   (a) Set coarsening degree-of-freedom tally to zero.

   (b) Choose an unmarked space-time element or time slab with the lowest merit function.

   (c) If a time slab was chosen, mark it for a factor of 2 coarsening and add $0.5\,\text{dof}_k$ to the coarsening tally.

   (d) If a space-time element was chosen, mark it for an order decrement and add $\text{dof}(p_{e,k}) - \text{dof}(p_{e,k}-1)$ to the coarsening tally.

   (e) If the tally meets or exceeds the coarsening budget, stop. Otherwise return to step 2b.

3. Refinement stage

   (a) Set refinement degree-of-freedom tally to zero.

   (b) Choose an unmarked space-time element or time slab with the highest merit function.

   (c) If a time slab was chosen, mark it for a factor of 0.5 refinement and add dof_k to the refinement tally.

(d) If a space-time element was chosen, mark it for an order increment and add $\text{dof}(p_{e,k}+1) - \text{dof}(p_{e,k})$ to the refinement tally.

(e) If the refinement budget is met or exceeded, stop. Else, return to step 3a.

We note that the factor of 2 in the temporal refinement and coarsening is somewhat arbitrary, and we have not investigated tuning or optimizing it for improved performance.

The coarsened or refined orders on the original time slabs define a spatial order layout as a function of time for the next adaptive iteration. To handle temporal coarsening, time slabs are redistributed using one-dimensional metric-based meshing, according to the following algorithm:

1. For each time slab k, define $\Delta t_k^{\text{desired}} = c\Delta t_k^{\text{current}}$ where c is 0.5 for refinement, 2 for coarsening, and 1 if the time slab is not marked.
2. Define $N^{\text{desired}} = \sum_k (\Delta t_k^{\text{current}} / \Delta t_k^{\text{desired}})$.
3. Scale $\Delta t_k^{\text{desired}}$ by $N^{\text{desired}} / \lceil N^{\text{desired}} \rceil$, and set $N^{\text{desired}} = \lceil N^{\text{desired}} \rceil$, where $\lceil\ \rceil$ is the greatest integer (ceiling) function.
4. Define $n_k^{\text{desired}} = \Delta t_k^{\text{current}} / \Delta t_k^{\text{desired}}$ = piecewise function on current time slabs.
5. The new time slab breakpoints are times t_l where $\int_0^{t_l} n_k^{\text{desired}} dt$ is an integer.

This algorithm refines the temporal grid while shuffling it slightly to make room for the new coarsened slabs.

6.1 Alternative Adaptive Methods

In the results, we compare our output-based adaptive algorithm to uniform p-refinement in space and bisection in time, as well as to a cheaper indicator based on the unweighted residual. The unweighted residual indicator is given by a form similar to the output-error (Eqn. 17), but without the adjoint and with absolute values on the individual residual components,

$$\epsilon_{e,k}^{\text{res}} = \mathbf{Z}_e |\overline{\mathbf{R}}_h^{k,m}(\mathbf{U}_h^H)|.$$

This indicator targets areas of the space-time domain where the partial differential equation is not well satisfied. Note that we do not presently include the GCL residual in this indicator to avoid complications with scaling quantities of different units. A jump-based space-time anisotropy measure [11] is used to determine $\beta_{e,k}^{\text{space}}$ and $\beta_{e,k}^{\text{time}}$ for this indicator.

7 Results

In this section, we present the results for several test cases. The first is a verification of the adjoint and error estimation procedures described above. The following cases then employ this error indicator to drive unsteady mesh adaptation for problems of engineering interest. These problems consist of airfoils pitching and plunging at low Reynolds number, and the convergence of the lift on these airfoils is compared for the three adaptive methods described above. The output convergence is shown as a function of total space-time degrees of freedom, which is representative of computational efficiency.

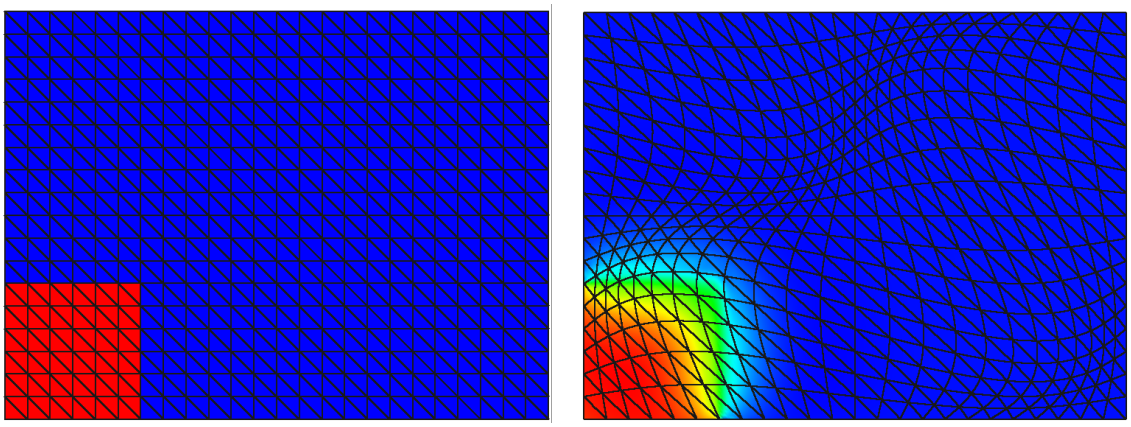
7.1 Error Estimate Verification

Here, we verify the error estimation procedures described above with a simple Navier-Stokes problem. The problem consists of a density perturbation in a stagnant fluid, situated in the corner of a rectangular basin bounded by no-slip walls (Figure 5). The density perturbation is allowed to diffuse for two time units, and the final force on the left wall is taken as our output of interest.

Now, since no boundaries are moving here, mesh motion is not required and the problem could be solved with a fixed grid. However, to test the error estimation with mesh motion, we instead choose to wave the mesh in the background, using the following formula to map the domain from reference to physical space:

$$\begin{aligned} x_1 &= X_1 + 2.0 \sin\left(\frac{2\pi X_1}{20}\right) \sin\left(\frac{\pi X_2}{7.5}\right) \sin\left(\frac{2\pi t}{3}\right), \\ x_2 &= X_2 + 1.5 \sin\left(\frac{2\pi X_1}{20}\right) \sin\left(\frac{\pi X_2}{7.5}\right) \sin\left(\frac{4\pi t}{3}\right). \end{aligned} \tag{21}$$

Since this mapping is relatively violent, with large variations in the GCL variable in both space and time, it should introduce motion-related errors and provide a good test of our error estimation procedure. Note that the movement of the mesh should in theory have no impact on the physics of the problem, and the solution should converge upon refinement to that with no mesh motion.



(a) Density, $t = 0$ (b) Density, $t = 2$

Figure 5: Initial and final meshes and densities. The initial density perturbation is 25% above the nominal value.

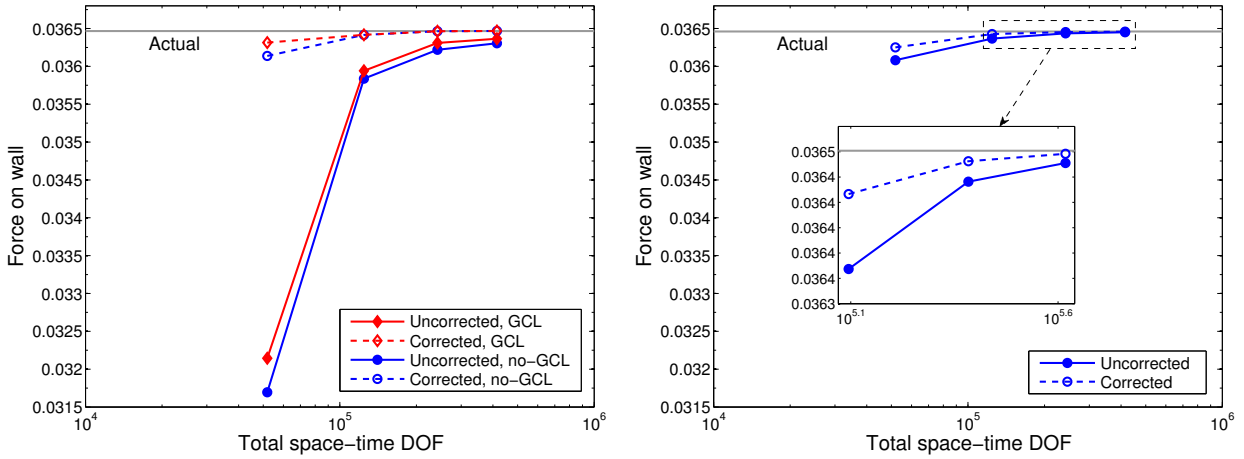
To verify the error estimation, three separate cases were run: (i) a no-motion case; (ii) a case with motion but no GCL; and (iii) a case with both motion and GCL. These cases were run at interpolation orders ranging from $p = 1$ to 4, with a DG1 time scheme and 10, 12, 14, and 16 time steps, respectively. Since we want to test both the spatial and temporal aspects of the error estimation, the number of time steps is gradually increased as p increases, in an attempt to keep the magnitude of the spatial and temporal errors comparable.

For each run, the output J_H on the current mesh is first recorded. Next, error estimation based on a $(p+1, r+1)$ fine space is performed, and the predicted change in the output relative to the fine space (δJ_{est}) is computed. Finally, the primal problem is solved directly on the $(p+1, r+1)$ space, and the output J_h is determined. The difference $\Delta J = J_H - J_h$ between coarse and fine outputs can then be obtained and compared to δJ_{est}. If the error estimation is working properly, these

values should correspond closely. Of course, since the problem is non-linear, the correspondence will generally not be exact.

Table 2 shows a comparison between the actual and predicted errors for the three cases. Here, the error in the error estimate itself is the figure of merit. From the table, we see that the error estimates with mesh motion display 93-99% accuracy relative to the actual output errors. For the GCL case specifically, this verifies that the GCL adjoint and error estimation procedures detailed above were implemented correctly.

The breakdown of errors due to the state and GCL individually are given in Table 3 (which includes an additional $p = 0$ run), and we see that the GCL-related errors constitute from 8-42% of the total error estimate. Finally, while seemingly polluted by noise at low orders, the no-motion error estimates – which are significantly lower than with mesh motion on – also converge to the actual errors upon mesh refinement.



(a) With motion

(b) Without motion

Figure 6: Output convergence for both motion and no-motion cases, shown with the same scale. Despite larger-magnitude errors, the corrected output converges rapidly to the true value for motion cases.

	GCL			**No GCL**			**No Motion**		
Run	δJ_{est}	ΔJ	% Error	δJ_{est}	ΔJ	% Error	δJ_{est}	ΔJ	% Error
$p = 1$	-4.170e-3	-4.020e-3	**3.73**	-4.443e-3	-4.489e-3	**1.03**	-1.687e-4	-2.882e-4	**41.44**
$p = 2$	-4.746e-4	-5.080e-4	**6.58**	-5.782e-4	-6.115e-4	**5.46**	-5.922e-5	-7.026e-5	**15.70**
$p = 3$	-1.551e-4	-1.543e-4	**0.49**	-2.448e-4	-2.438e-4	**0.40**	-1.610e-5	-1.555e-5	**3.51**
$p = 4$	-1.022e-4	-1.008e-4	**1.40**	-1.626e-4	-1.608e-4	**1.12**	-7.308e-6	-7.015e-6	**4.18**

Table 2: Relative accuracy of error estimates for motion and no-motion cases at different orders p. "% Error" denotes the error in δJ_{est} relative to ΔJ.

	$p = 0$	$p = 1$	$p = 2$	$p = 3$	$p = 4$
State Errors	3.593e-02	-4.607e-03	-5.930e-04	-2.526e-04	-1.675e-04
GCL Errors	2.633e-02	4.376e-04	1.184e-04	9.743e-05	6.525e-05
GCL % of Total	**42.29**	**8.67**	**16.64**	**27.84**	**28.03**

Table 3: Relative contribution of GCL and state errors to total error estimate.

7.2 Dynamic Mesh Adaptation for a Pitching Airfoil

With the performance of the error estimates confirmed, we next use them to drive mesh adaptation for a case of engineering interest – an airfoil pitching sinusoidally at low Reynolds number. The airfoil starts from an impulsive free-stream condition and undergoes three periods of pitching motion (with amplitude $30\,^\circ$ and period $T = 2.5$) at a Strouhal number of 1.0, a Reynolds number of 400, and a free-stream Mach number of 0.2. The airfoil is a NACA 0012 situated in the center of a 60 x 60 chord-length domain, and the mesh consists of 1,454 triangular elements.

Entropy contours at several phases of the motion are shown in Figure 9. A series of vortices forms behind the airfoil as it completes the motion cycle, and these vortices combine with the free-stream flow and inertial effects to generate forces on the airfoil. Our output of interest is the lift component of these forces integrated over the final 2.5% of the simulation time.

To compute this output, the three adaptive methods described in section 6.1 (output-based, residual, and uniform refinement) were considered. Several adaptations were performed starting from an initial $p = 1$, 70 time step solution. The spatial order was constrained to lie within the range $0 \leq p \leq 5$, and a DG1 time scheme was used for all runs. For the output error and residual methods, 5% of space-time elements were coarsened at each iteration, while the overall size of the space-time mesh was increased by 30%.

7.2.1 Results

The output convergence for each adaptive method as a function of total space-time degrees of freedom is shown in Figure 7. For the output error method, both the actual output J and the corrected output $J - \delta J_{est}$ are given. The output-based adaptation converges much faster than uniform refinement, requiring approximately two orders of magnitude fewer degrees of freedom to achieve the true output value. It also significantly outperforms residual adaptation, which gets distracted by acoustic waves emanating from the airfoil and fails to converge even after 14 adaptive iterations.

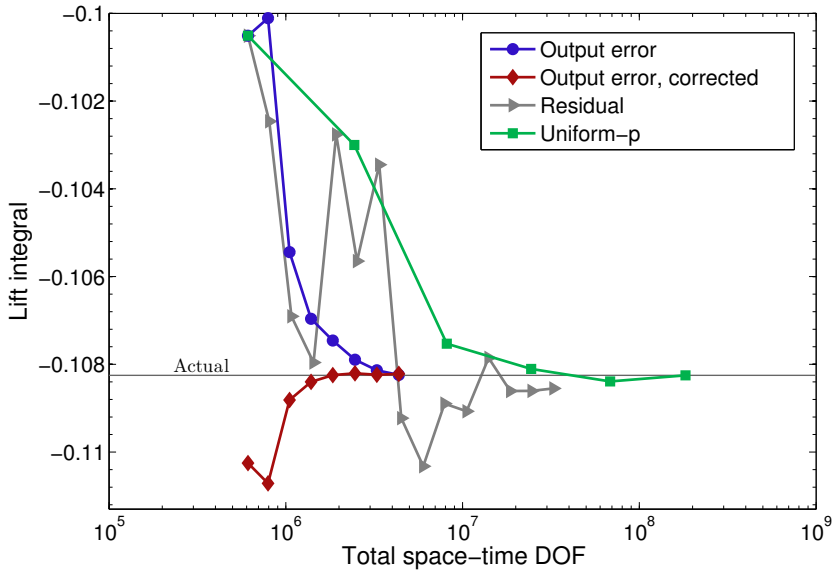


Figure 7: Pitching airfoil: Output convergence for various adaptive methods. The adjoint-based method outperforms both residual-based and uniform refinement by orders of magnitude.

The temporal and spatial grids from the final output-adapted run are shown in Figures 8 and 10, respectively. In Figure 10, we see that the regions where vortex shedding occurs are heavily targeted, while a circular region surrounding the airfoil is refined to a lesser extent. In the far field, most elements have been coarsened to $p = 0$, as expected. Temporally, the periods of strongest vortex shedding are targeted, with a preference given to those occurring in the first half of the simulation. This preference for earlier times makes sense, since the initial vortices influence the vortex shedding during later parts of the simulation, and must therefore be resolved accurately. The residual indicator also targets this vortex shedding, but does so blindly, giving roughly equal weight to each vortex shed throughout the simulation (evidenced by the equally-spaced refinement in Figure 8).

To highlight one of the factors driving the adaptation, contours of the GCL adjoint are shown alongside the entropy contours in Figure 9. The first contour shows a band of inward-moving sensitivity waves collapsing upon the trailing edge near time $t = T/3 = 0.833$. Similar behavior is seen in the other adjoint components, and this is reflected in the adaptation, which heavily targets the times following $t \approx 0.8$. The final adjoint contour in Figure 9 shows the sensitivity field converging on the airfoil near the end of the simulation. Any errors made outside of this region no longer have time to reach the airfoil and influence the output.

Finally, since the GCL adjoint and error estimates are of particular interest in this paper, a breakdown of the output error into its state and GCL components is provided in Table 4. From the table, we see that the GCL errors initially make up only a small percentage of the total error, but as the adaptations proceed, their contribution increases to over half of the total error estimate. While this does not prove that the GCL itself is essential to obtaining an accurate output, it does imply that *if* the GCL is being used, a corresponding GCL adjoint is necessary to obtain an accurate error estimate.

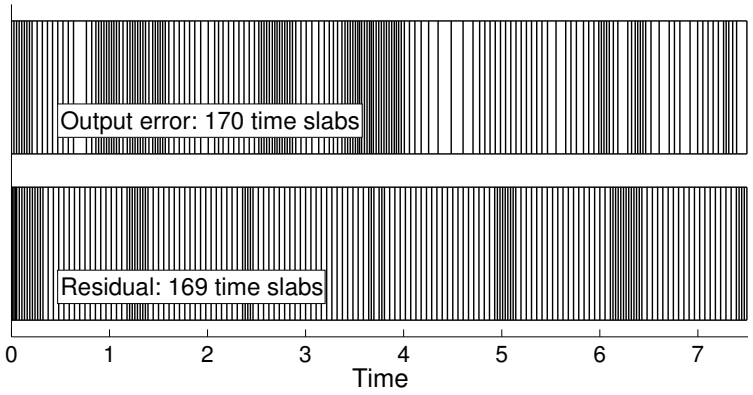
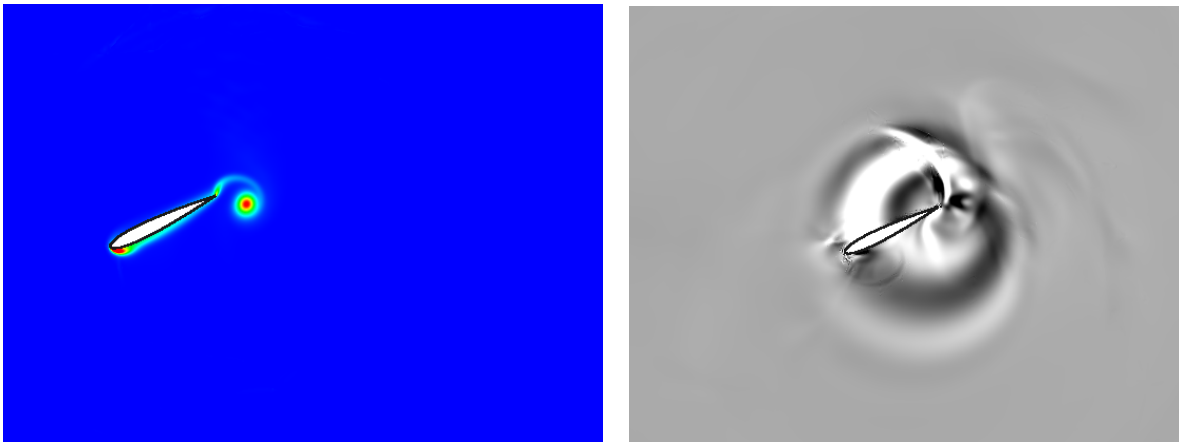


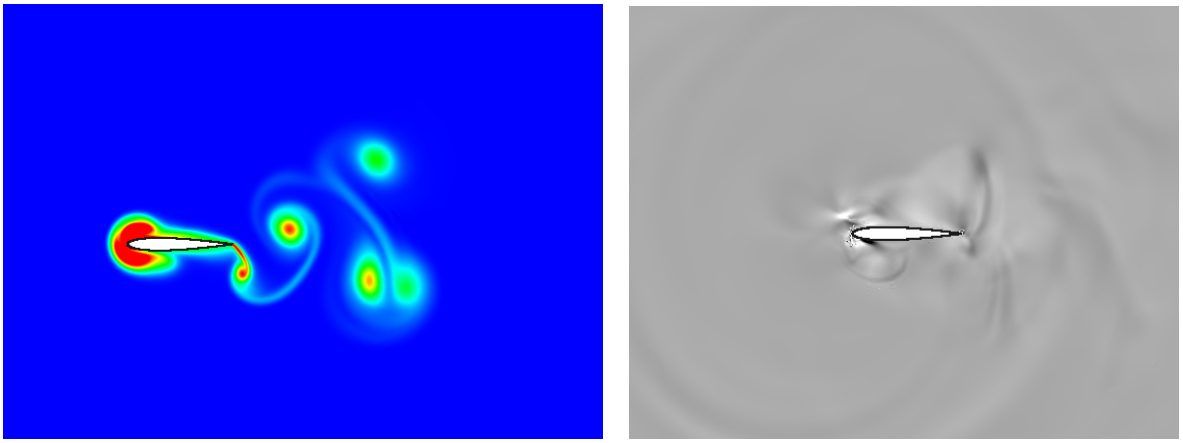
Figure 8: Pitching airfoil: Temporal grids from the seventh adaptation of both adjoint and residual runs.

	Iter. 1	Iter. 2	Iter. 3	Iter. 4	Iter. 5	Iter. 6	Iter. 7	Iter. 8
State Errors	9.87e-03	1.09e-02	3.82e-03	1.93e-03	6.62e-04	4.00e-04	4.49e-04	3.14e-04
GCL Errors	-1.26e-04	-2.75e-04	-4.54e-04	-4.87e-04	1.27e-04	-8.90e-05	-3.49e-04	-3.36e-04
GCL % of Total	**1.26**	**2.47**	**10.61**	**20.17**	**16.12**	**18.19**	**43.69**	**51.76**

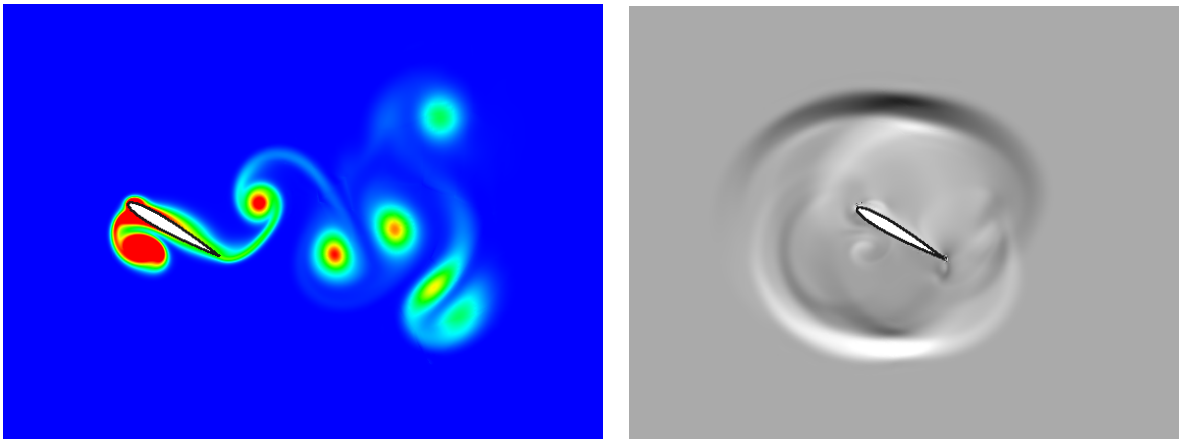
Table 4: Pitching airfoil: Relative contribution of GCL and state errors to the total error estimate for all iterations of output-based adaptation.



(a) $t = T/3$

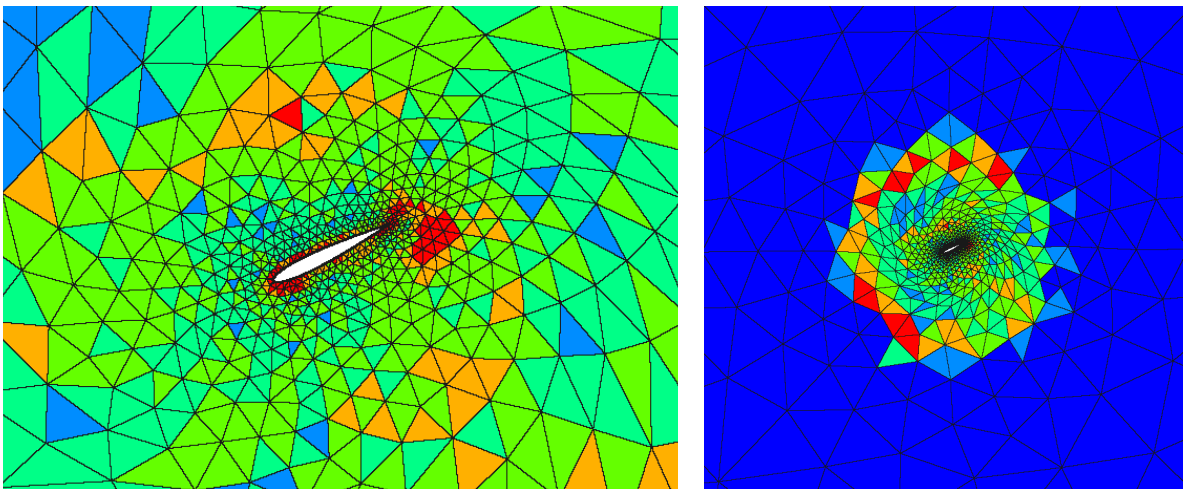


(b) $t = 2T$

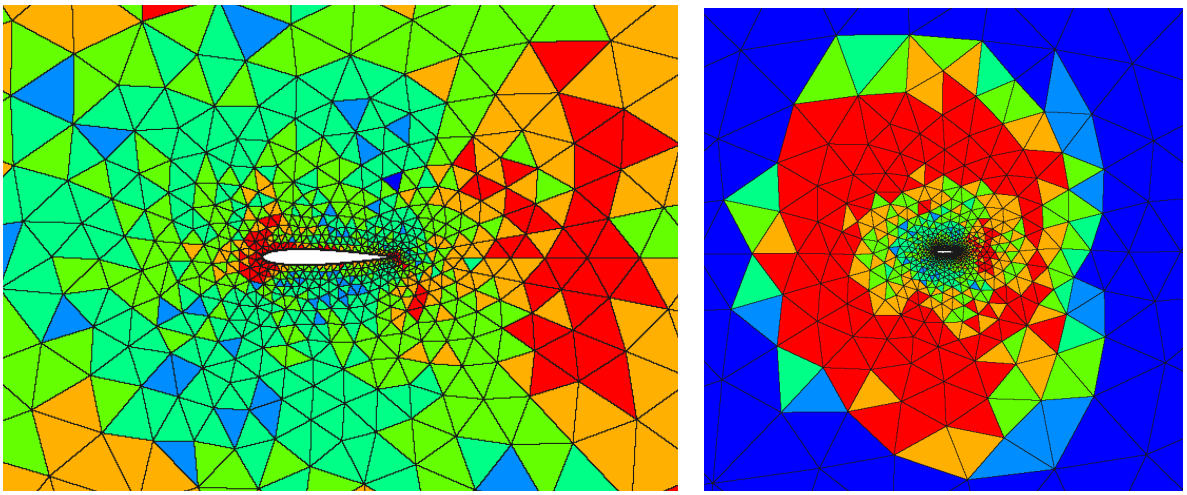


(c) $t = 11T/4$

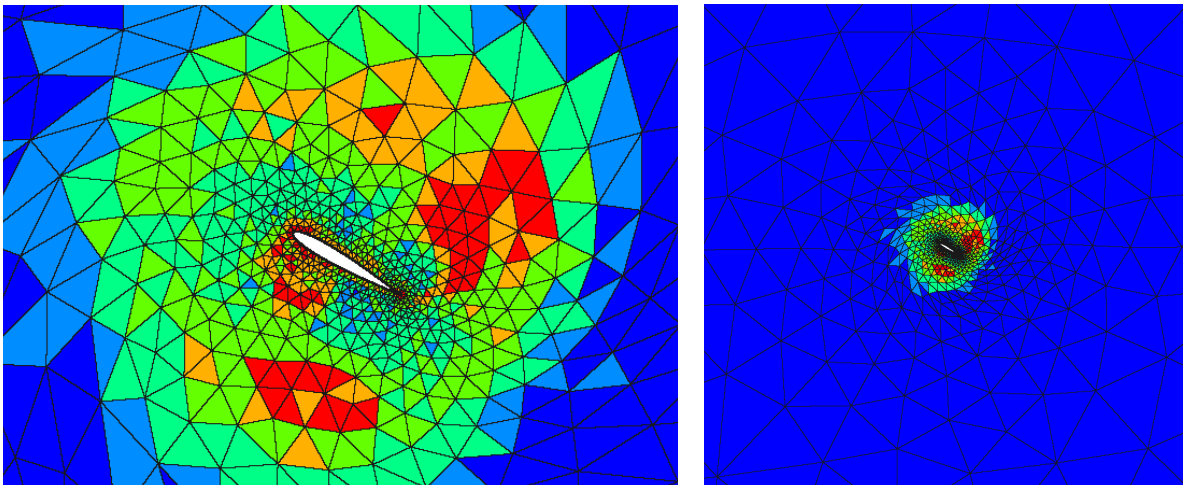
Figure 9: Pitching airfoil: Entropy (left) and GCL adjoint (right) contours at various stages of the pitch motion on a fine mesh. Note that the GCL adjoint contours have been re-scaled to more clearly show the features. (Black is -1.5, white is 0.75.)



(a) $t = T/3$



(b) $t = 2T$



(c) $t = 11T/4$

Figure 10: Pitching airfoil: Output-adapted meshes at various stages of the pitch motion. Blue is $p = 0$, red is $p = 5$.

7.3 Dynamic Mesh Adaptation for Pitching and Plunging Airfoils

Next, we try a more complicated case – two airfoils pitching and plunging in series. The airfoils start from an impulsive free-stream condition, and again undergo three periods of motion. The plunge amplitude is 0.25 chords, and the pitch amplitude is 30°. The Strouhal, Mach, and Reynolds numbers are 2/3, 0.3, and 1200, respectively. The airfoils are offset 4.5 chords horizontally and 1 chord vertically, and are situated in a 60 x 60 chord-length mesh with 3,534 triangular elements.

Entropy contours at various phases of the motion are shown in Figure 13. A reverse Kármán vortex street develops behind each airfoil, and the second airfoil interacts with the wake from the first airfoil near the end of the simulation. The output of interest is the lift on the second airfoil integrated from time $t = 7.25$ to $t = 7.5$ (the final time).

To compute this output, we again considered output error, residual, and uniform refinement strategies. Adaptations were performed starting from an initial $p = 1$, 90 time step solution, with a 35% growth factor and 5% coarsening factor used for the output- and residual-based methods. The spatial order p was again constrained to lie between 0 and 5, and a DG1 scheme was used in time.

7.3.1 Results

The output convergence for each adaptive method as a function of total space-time degrees of freedom is shown in Figure 11. Once again, the output-based adaptation performs the best. The corrected output reaches the true value 1-2 orders of magnitude before uniform refinement, while the residual adaptation is oscillatory and fails to converge.

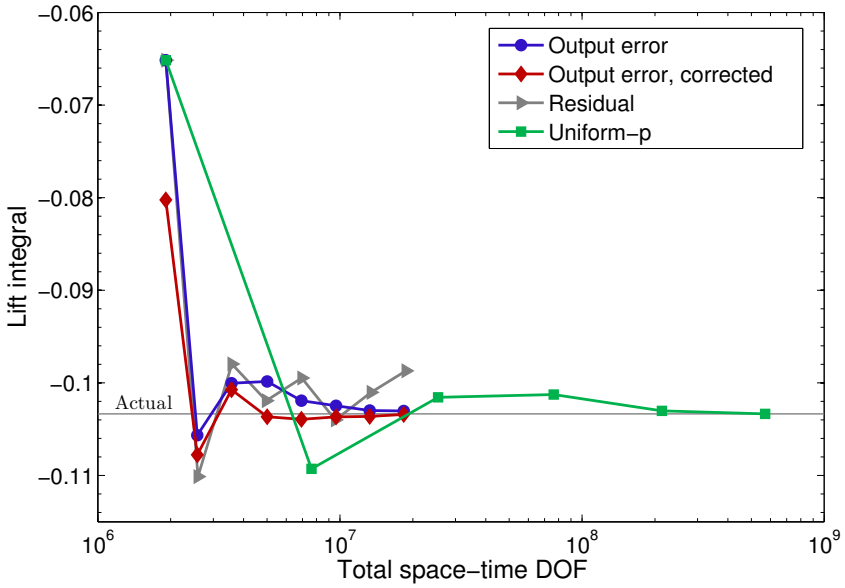


Figure 11: Two-airfoil case: Output convergence for various adaptive methods. The adjoint-based method performs the best.

The temporal and spatial meshes from the final output-based adaptation are shown in Figures 12 and 14, respectively. We see that the near-airfoil and vortex shedding regions are targeted for adaptation, as well as the group of large elements surrounding the mesh motion regions. While somewhat difficult to observe in the still-frames, the initial vortex shed from the first airfoil is

heavily targeted throughout the simulation, since this vortex later collides with the second airfoil near the final time.

The behavior of the GCL adjoint, shown in Figure 13, illustrates the output's sensitivity to this initial vortex as well. The time $t = 7T/10$ is the instant just before this vortex is shed, and the large sensitivity of the output to this event can be seen in the adjoint contours. As the simulation proceeds, the output sensitivity gradually shifts from the first airfoil to the second, before collapsing upon the second airfoil at the final time.

Some other aspects of the GCL adjoint are worth pointing out. In the first two contours, the near-circular rings represent inward-moving acoustic waves, which converge upon a particular region as the simulation proceeds. The existence of a ring implies that an important event in space-time is about to occur, and any errors made within the ring have the ability to influence this event. In this simulation, the important events tend to be instances of vortex shedding, and the rings tend to converge on the trailing edge regions. Lastly, between the two airfoils, a path can be seen tethering them together. This path appears because any errors within it ultimately reach the second airfoil via convection, and can therefore directly affect the output.

Finally, the breakdown of state versus GCL errors is shown in Table 5. Once again, the GCL component of the error estimate is relatively small during initial adaptations, but becomes significant as the adaptations proceed. For this case however, use of the GCL overall provides only a small improvement in accuracy relative to a similar no-GCL run (shown in Figure 15), particularly when considering the corrected output. This suggests that for certain cases, satisfying the GCL may not be critical to obtaining accurate outputs, especially if those outputs are corrected by an appropriate error estimate.

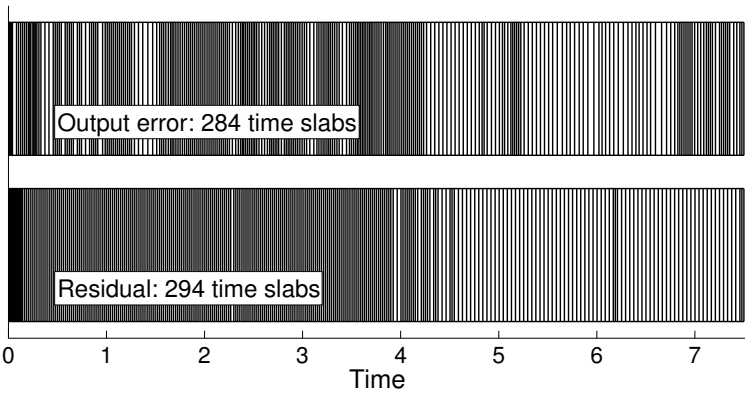
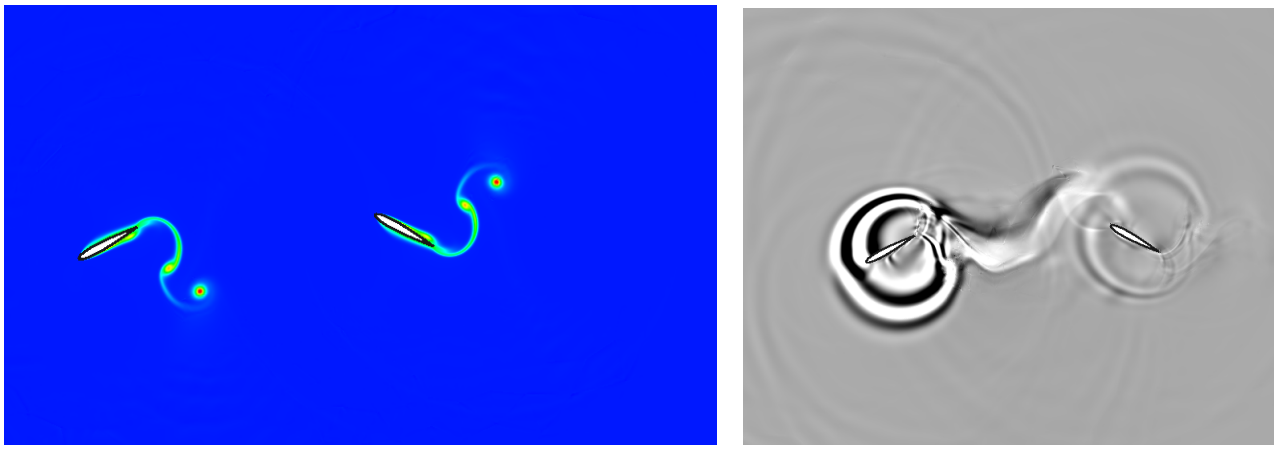


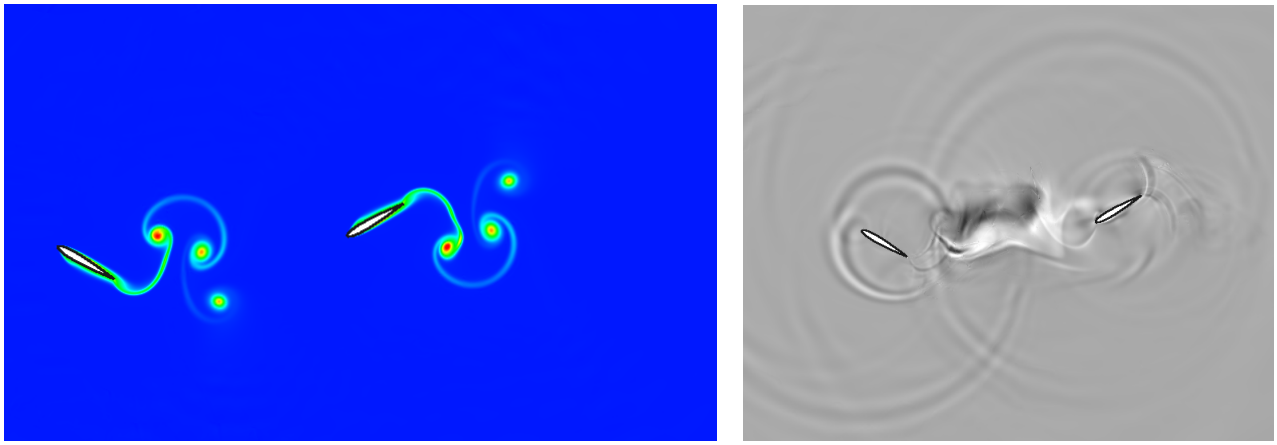
Figure 12: Two-airfoil case: Temporal grids from the seventh adaptation of both adjoint and residual runs.

	Iter. 1	Iter. 2	Iter. 3	Iter. 4	Iter. 5	Iter. 6	Iter. 7	Iter. 8
State Errors	1.52e-02	2.21e-03	7.45e-04	3.84e-03	2.37e-03	1.03e-03	3.62e-04	-5.79e-04
GCL Errors	-1.54e-04	-9.30e-05	-3.75e-05	-3.87e-05	-3.49e-04	1.68e-04	2.77e-04	9.53e-04
GCL % of Total	**1.00**	**4.04**	**4.79**	**1.00**	**12.84**	**13.99**	**43.36**	**62.19**

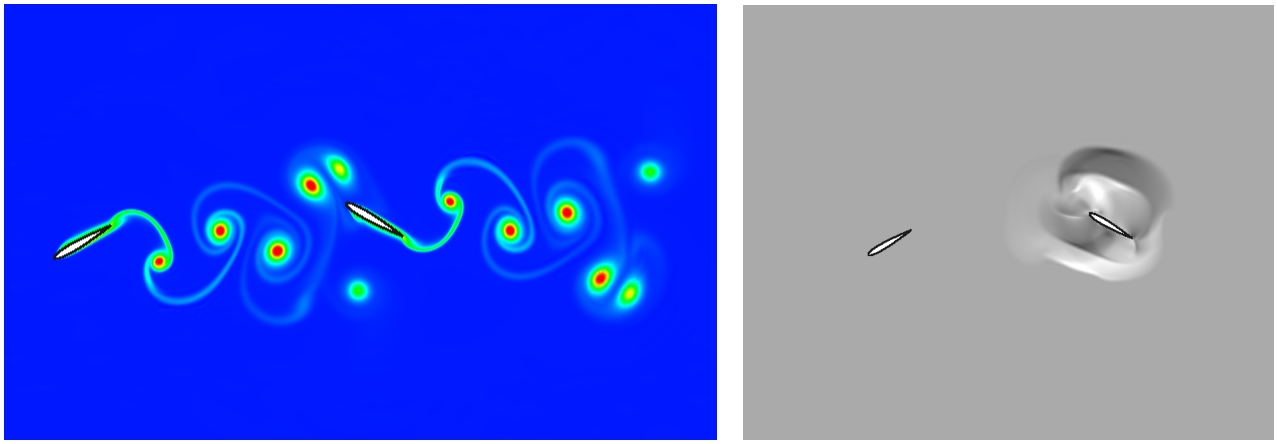
Table 5: Two-airfoil case: Relative contribution of GCL and state errors to the total error estimate for all iterations of output-based adaptation.



(a) $t = 7T/10$

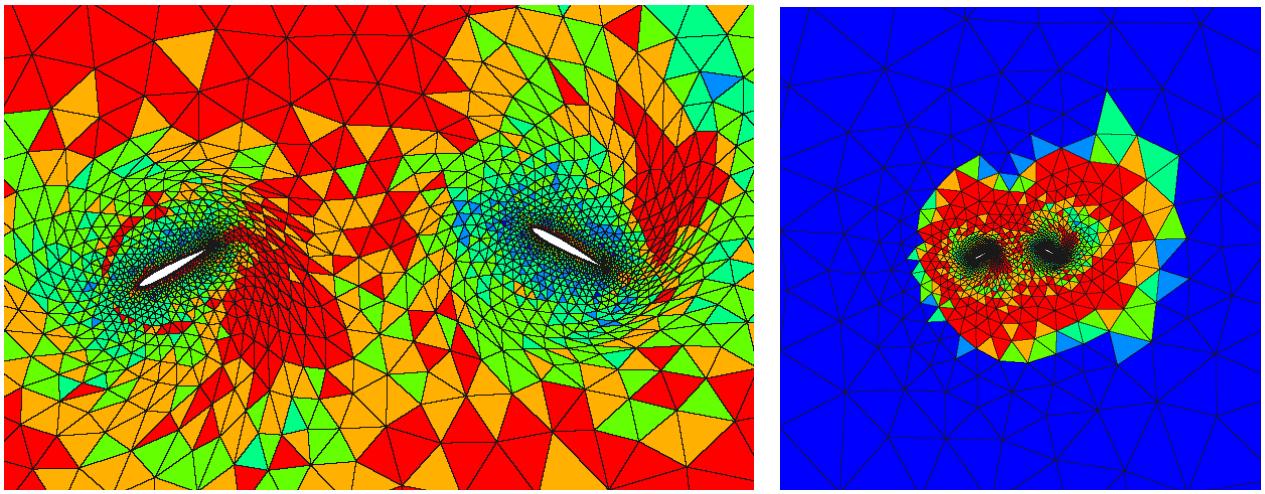


(b) $t = 5T/4$

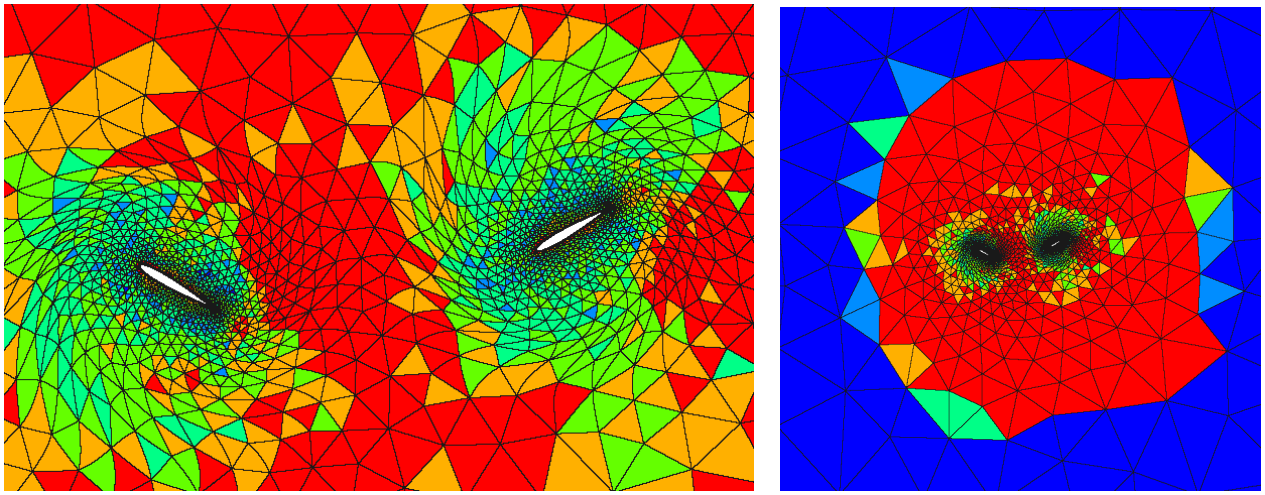


(c) $t = 11T/4$

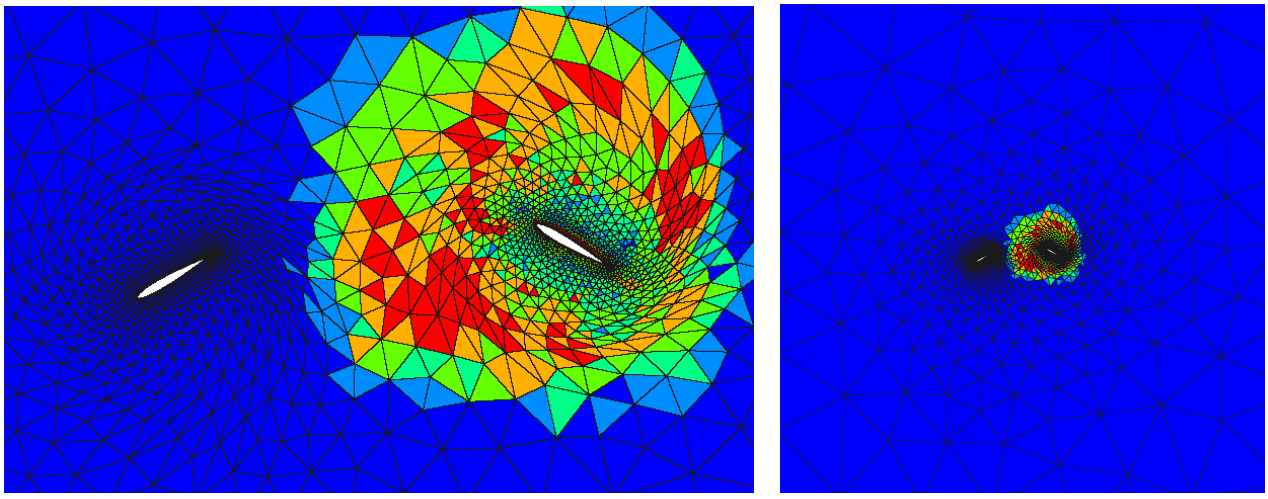
Figure 13: Two airfoil case: Entropy (left) and GCL adjoint (right) contours at various stages of the motion on a fine mesh. Note that the GCL adjoint contours have been re-scaled to more clearly show the features. (Black is -2, white is 1.)



(a) $t = 7T/10$



(b) $t = 5T/4$



(c) $t = 11T/4$

Figure 14: Two-airfoil case: Output-adapted meshes at various stages of the motion. Blue is $p = 0$, red is $p = 5$.

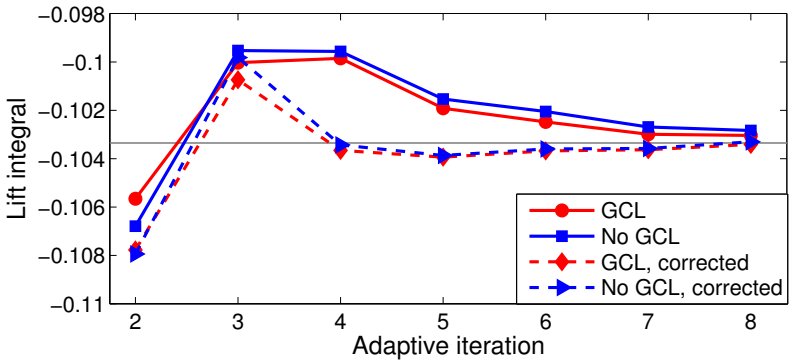


Figure 15: Two-airfoil case: Output convergence for runs with and without the geometric conservation law.

8 Conclusions

In this paper, we provided methods for unsteady error estimation and mesh adaptation for the compressible Navier-Stokes equations on deformable domains. The DG finite element discretization in both space and time combined with adjoint solves on the fine space were shown to provide rigorous error estimates for engineering outputs. Discrete adjoint equations for the geometric conservation law were derived, and the impact of the GCL on error estimation was assessed. While the output error due to the GCL is generally small relative to the state error, it can constitute a significant portion of the total error as the output nears convergence. This means that if the GCL is used, the corresponding GCL adjoint is required to obtain accurate error estimates. That said, preliminary results indicate that for certain cases of engineering interest, use of the GCL does not significantly improve output accuracy relative to runs without it.

Overall, the output-based adaptation presented here represents a marked improvement over more heuristic methods when total mesh size is used as the measure of computational efficiency. An alternative indicator of efficiency is the CPU time, which depends strongly on implementation and parallelization choices. For parallel computations, dynamic-p adaptation leads to temporally-varying processor workloads, suggesting that an algorithm for dynamically repartitioning the mesh may be beneficial. Such an algorithm is the subject of ongoing research.

Acknowledgments

The authors acknowledge support given by the University of Michigan and by the Air Force Office of Scientific Research under grant FA9550-11-1-0081. Further support was provided by the Department of Defense through the National Defense Science & Engineering Graduate Fellowship (NDSEG) Program.

References

[1] Niles A. Pierce and Michael B. Giles. Adjoint recovery of superconvergent functionals from PDE approximations. *SIAM Review*, 42(2):247–264, 2000.

[2] R. Becker and R. Rannacher. An optimal control approach to a posteriori error estimation in finite element methods. In A. Iserles, editor, *Acta Numerica*, pages 1–102. Cambridge University Press, 2001.

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

[4] D. A. Venditti and D. L. Darmofal. Anisotropic grid adaptation for functional outputs: application to two-dimensional viscous flows. *Journal of Computational Physics*, 187(1):22–46, 2003.

[5] S. Sen, K. Veroy, D.B.P. Huynh, S. Deparis, N.C. Nguyen, and A.T. Patera. "Natural norm" a posteriori error estimators for reduced basis approximations. *Journal of Computational Physics*, 217:37–62, 2006.

[6] Marian Nemec and Michael J. Aftosmis. Error estimation and adpative refinement for embedded-boundary Cartesian meshes. AIAA Paper 2007-4187, 2007.

[7] Krzysztof J. Fidkowski and David L. Darmofal. Review of output-based error estimation and mesh adaptation in computational fluid dynamics. *American Institute of Aeronautics and Astronautics Journal*, 49(4):673–694, 2011.

[8] Karthik Mani and Dimitri J. Mavriplis. Discrete adjoint based time-step adaptation and error reduction in unsteady flow problems. AIAA Paper 2007-3944, 2007.

[9] Karthik Mani and Dimitri J. Mavriplis. Error estimation and adaptation for functional outputs in time-dependent flow problems. *Journal of Computational Physics*, 229:415–440, 2010.

[10] Timothy J. Barth. Space-time error representation and estimation in Navier-Stokes calculations. In Stavros C. Kassinos, Carlos A. Langer, Gianluca Iaccarino, and Parviz Moin, editors, *Complex Effects in Large Eddy Simulations*, pages 29–48. Springer Berlin Heidelberg, Lecture Notes in Computational Science and Engineering Vol 26, 2007.

[11] Krzysztof J. Fidkowski and Yuxing Luo. Output-based space-time mesh adaptation for the compressible Navier-Stokes equations. *Journal of Computational Physics*, 230:5753–5773, 2011.

[12] D. Meidner and B. Vexler. Adaptive space-time finite element methods for parabolic optimization problems. *SIAM Journal on Control Optimization*, 46(1):116–142, 2007.

[13] Michael Schmich and Boris Vexler. Adaptivity with dynamic meshes for space-time finite element discretizations of parabolic equations. *SIAM Journal on Scientific Computing*, 30(1):369–393, 2008.

[14] Krzysztof J. Fidkowski. An output-based dynamic order refinement strategy for unsteady aerodynamics. AIAA Paper 2012-77, 2012.

[15] Y. Luo and K.J. Fidkowski. Output-based space time mesh adaptation for unsteady aerodynamics. AIAA Paper 2011-491, 2011.

[16] P.-O. Persson, J. Bonet, and J. Peraire. Discontinuous Galerkin solution of the Navier-Stokes equations on deformable domains. *Computer Methods in Applied Mechanical Engineering*, 198:1585–1595, 2009.

[17] P. L. Roe. Approximate Riemann solvers, parameter vectors, and difference schemes. *Journal of Computational Physics*, 43:357–372, 1981.

[18] F. Bassi and S. Rebay. GMRES discontinuous Galerkin solution of the compressible Navier-Stokes equations. In Karniadakis Cockburn and Shu, editors, *Discontinuous Galerkin Methods: Theory, Computation and Applications*, pages 197–208. Springer, Berlin, 2000.

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

[20] Krzysztof J. Fidkowski. Output error estimation strategies for discontinuous Galerkin discretizations of unsteady convection-dominated flows. *International Journal for Numerical Methods in Engineering*, 2011.