Spatially Non-Uniform Time-Step Adaptation for Functional Outputs in Unsteady Flow Problems

Karthik Mani * and Dimitri J. Mavriplis †

Department of Mechanical Engineering, University of Wyoming, Laramie, Wyoming 82071-3295

This paper presents a space-time finite-volume formulation for the Euler equations, which allows for the use of spatially non-uniform time-steps. The formulation also inherently accounts for the effect of dynamically deforming computational meshes. The space and time dimensions are treated in a unified manner so as to permit the variation of control volume sizes in both dimensions. The primary goal is to maintain solution accuracy while reducing the number of unknowns in the overall solution process and potentially lower computational expense. While the formulation presented here is capable of simultaneously handling non conformal meshes in both space and time, the scope of the paper is limited to conformal spatial meshes with non conformal temporal meshes. At any slice in time, the number of spatial elements remains the same, but across any slice in space, the number of time-steps is allowed to vary. In traditional terms, this translates to non-uniform temporal advancement of spatial elements in an unsteady problem. Two unsteady problems, one where an isentropic vortex is convected through a rectangular domain and one of a pitching NACA64A010 airfoil in transonic conditions are presented to demonstrate the algorithm. In the vortex convection problem, a local temporal error indicator is used to identify space-time elements which require higher resolution in the time dimension, thus marking them for temporal refinement. In the case of the transonic pitching airfoil, the adjoint-weighted residual method targeting the lift is used as the error indicator. The results indicate that significant reduction in the overall degrees-of-freedom required to solve an unsteady problem can be achieved using the proposed algorithm. Modest improvements in computational expense for specific problems are also observed.

I. Introduction

The fundamental goal of numerical simulations is to obtain accurate solutions to nonlinear governing equations at acceptable computational expense. In the early days of the field, unsteady problems were deemed unreasonably expensive and the focus remained on accurate solutions to steady-state problems. Several algorithmic advances in terms of numerically solving nonlinear equations have been made over the years to reduce the cost associated with obtaining accurate solutions.[1,2] Such advances coupled with improved hardware capabilities have since enabled solutions to unsteady problems, firstly using explicit methods and then progressing to faster and more efficient implicit time-integration schemes. While simple unsteady problems remain tractable, problems exhibiting strong unsteadiness currently require simulation times running into days or even weeks. Several methods have been been investigated to address this issue. Higher-order implicit time-integration schemes[3–7] operate directly in the time domain and maintain solution accuracy while using fewer time-steps. For problems exhibiting strong periodic behavior, frequency domain methods[8,9] transform the unsteady problem into time spectral space and restrict the number of modes required for the solution. Solution techniques such as multigrid[10–12] have also been developed to reduce the cost associated with solving the non-linear system of equations at each each time-step. Although frequency-domain based techniques are quite powerful, their performance is inefficient when applied to problems with no dominant periodicity. This necessitates the development of improved algorithms operating in the time domain so as to be applicable to all unsteady problems in a general sense.

In the context of implicit time-integration schemes, which impose no restrictions on the time-step size, additional savings can be realized by varying the time-step size at each implicit time-step provided an estimate of the local temporal error at each time-step is available. This approach is directly analogous to the universally common practice of utilizing non-uniform spatial meshes when solving problems. It is unnecessary to use high spatial resolution in regions of very low gradients such as the far field in aerodynamic simulations. Using lower resolution in such regions

*Postdoctoral Research Associate; email: kmani@uwyo.edu.

†Professor; email: mavripl@uwyo.edu.

immediately frees up essential resources to focus on high gradient regions such as shock waves, boundary layers contact discontinuities, and wakes that exist closer to the aerodynamic body of interest. The same argument applies to the time domain and computational savings can be achieved by focusing temporal resolution on regions of the time domain where such resolution is necessary. While the locations of high gradient spatial features are to a certain extent known a priori, the locations of similar interesting features in the time domain are generally not known ahead of time and an error indicator becomes necessary.

Several methods currently exist for time-step size control based on temporal discretization error. The purpose of such methods is to adaptively change the time-step size in various regions of the time domain resulting in overall reduction in computational expense with minimal loss in solution accuracy. Local error-based methods[4,13,14] use temporal discretizations of different orders of accuracy for the time derivative term in the governing equations in order estimate the local temporal discretization error. Higher temporal resolution (i.e. smaller and more time-steps) is used in regions of the time domain where the local temporal discretization error is high and lower resolution is used in regions of low local error. Goal-based or global error-based methods[15,16] determine the distribution of the temporal discretization error that is relevant to functional quantities of interest and similarly adapt the time-step size to improve functional accuracy.

While non-uniform spacing in the time domain is essential to reducing overall simulation cost, a fundamental point that is overlooked here is that resolution in the spatial and temporal domains can be treated independently of each other. Just as spatial flow gradients are small in the far field, it should be noted that there may large regions of the spatial domain where the temporal gradients also remain small. Much like the spatial flow gradients near the aerodynamic object of interest, it is quite likely that high temporal flow gradients exist only near features that exhibit strong unsteadiness such as moving shock waves or shedding vortices. The interpretation is that spatial elements with small temporal gradients do not have to be advanced in time at the same pace as spatial elements with high temporal gradients. This realization changes the traditional view on how unsteady problems are solved and opens a new avenue to explore for the reduction of computational cost. Since existing error indicators (both local- and goal-based) construct the temporal discretization error at each time-step from a spatial distribution of the same at each time-step, the means already exist to identify individual spatial elements which require refinement in the time dimension.

The main hurdle is the development of efficient discretizations for the governing equations that permit non-uniform advancement in time. This paper presents a method to overcome this problem by treating the governing equations in integral form over space-time elements which have a constant cell averaged state over the entire element in space and time. While there have been a few attempts to solve the governing equations in space-time integral form,[17–19] advancing spatial elements non-uniformly in time has mostly been limited to explicit time-integration schemes.

It should be noted that in the work presented in this paper no attempt is made to adapt the spatial domain. Hence there is direct correspondence between spatial elements at different times and there are no spatial discontinuities going from one point in time to another within the same space-time element. However, there exists discontinuities in time going from one point in space to another.

Both local error-based and adjoint-based error indicators are investigated for use with the proposed method to idenfify regions in space-time that require higher temporal resolution.

II. Analysis Problem Formulation

A. Governing equations of the flow Problem in ALE form

We begin with the traditional conservative form of the Euler equations. In vectorial form the conservative form of the Euler equations may be written as:

$$\frac{\partial \mathbf{U}(\mathbf{x},t)}{\partial t} + \nabla \cdot \mathbf{F}(\mathbf{U}) = 0 \tag{1}$$

where the state vector $\mathbf{U}$ of conserved variables and the cartesian inviscid flux vector $\mathbf{F} = (\mathbf{F}^x, \mathbf{F}^y)$ are:

$$\mathbf{U} = \begin{pmatrix} \rho \\ \rho u \\ \rho v \\ E_t \end{pmatrix}, \quad \mathbf{F}^x = \begin{pmatrix} \rho u \\ \rho u^2 + p \\ \rho uv \\ u(E_t + p) \end{pmatrix}, \quad \mathbf{F}^y = \begin{pmatrix} \rho v \\ \rho uv \\ \rho v^2 + p \\ v(E_t + p) \end{pmatrix}, \tag{2}$$

Here ρ is the fluid density, (u,v) are the cartesian fluid velocity components, p is the pressure and E_t is the total energy. For an ideal gas, the equation of state relates the pressure to total energy by:

$$p = (\gamma - 1)\left[E_t - \frac{1}{2}\rho(u^2+v^2)\right] \tag{3}$$

where $\gamma = 1.4$ is the ratio of specific heats. Applying the divergence theorem and integrating over a moving control volume $A(t)$ that is bounded by the control surface $B(t)$ yields:

$$\int_{A(t)} \frac{\partial \mathbf{U}}{\partial t} dA + \int_{B(t)} \mathbf{F}(\mathbf{U}) \cdot \mathbf{n} dB = 0 \tag{4}$$

Using the differential identity:

$$\frac{\partial}{\partial t}\int_{A(t)} \mathbf{U} dA = \int_{A(t)} \frac{\partial \mathbf{U}}{\partial t} dA + \int_{B(t)} (\dot{\mathbf{x}} \cdot \mathbf{n}) dB \tag{5}$$

equation (4) is rewritten as:

$$\frac{\partial}{\partial t}\int_{A(t)} \mathbf{U} dA + \int_{B(t)} [\mathbf{F}(\mathbf{U}) - \dot{\mathbf{x}}\mathbf{U}] \cdot \mathbf{n} dB = 0 \tag{6}$$

or when considering cell-averaged constant values within each spatial element for the state $\mathbf{U}$, as:

$$\frac{dA\mathbf{U}}{dt} + \int_{B(t)} [\mathbf{F}(\mathbf{U}) - \dot{\mathbf{x}}\mathbf{U}] \cdot \mathbf{n} dB = 0 \tag{7}$$

This is the Arbitrary-Lagrangian-Eulerian (ALE) finite-volume form of the Euler equations. The equations are required in ALE form since the problem involves deforming meshes where mesh elements change in shape and size in time. Here A refers to the area or volume of the spatial element, $\dot{\mathbf{x}}$ is the vector of spatial mesh face or edge velocities, $\mathbf{n}$ is the unit normal of the face or edge, and B refers to the area or length of the bounding surface or edge.

Since the goal is to adapt the time-step size locally for each spatial element in the mesh, the ALE finite-volume form shown in equation (7) is further integrated in time to obtain the finite-volume formulation in space-time as:

$$\int_T \left\{ \frac{dA\mathbf{U}}{dt} + \int_{B(t)} [\mathbf{F}(\mathbf{U}) - \dot{\mathbf{x}}\mathbf{U}] \cdot \mathbf{n} dB \right\} dt = 0 \tag{8}$$

where separating the time integral yields:

$$\int_T \frac{dA\mathbf{U}}{dt} dt + \int_T \int_{B(t)} [\mathbf{F}(\mathbf{U}) - \dot{\mathbf{x}}\mathbf{U}] \cdot \mathbf{n} dB dt = 0 \tag{9}$$

B. The space-time element

Figure (1) illustrates a space-time element constructed using an arbitrary spatial element ABC at two different time indices n and $n-1$. This can also be interpreted as a space-time element bounded by two temporal faces which are defined by the spatial elements ABC at the two time indices. The spatial components of the normal vector for the temporal faces are zero by definition of the problem. The space-time element is also bounded on the sides by space-time faces. These faces are inclined due to spatial mesh deformation in time and usually have nonzero spatial and temporal normal vector components. Temporal faces are bounded by spatial edges and space-time faces are bounded by both temporal and spatial edges. Temporal edges connect the tails or heads of spatial edges at different times as shown in the figure. Figure (2) illustrates an example of a space-time face. There may be multiple temporal edges between two spatial edges in time because of neighboring space-time elements which take multiple time-steps in relation to the space-time element under consideration. Every edge (temporal or spatial) connects to a space-time node. It should also be noted that the top and bottom defining spatial edges of a space-time face are always oriented in the same direction. Temporal edges are always oriented such that they point in the direction of advancing time.

C. Temporal discretization

The solution state $\mathbf{U}$ is assumed to be constant within each space-time element and hence at any interface between space-time elements in the time dimension, the solution is double valued. Based on these assumptions, the time derivative term may be discretely represented as:

$$\int_T \frac{dA\mathbf{U}}{dt} dt = \frac{dA\mathbf{U}}{dt} \Delta t \tag{10}$$

where Δt refers to the height of the space-time element in the time dimension. The time derivative term is discretized using a first-order accurate BDF1 (Backward Difference Formula 1) scheme as:

$$\frac{dA\mathbf{U}}{dt} \Delta t = A^n \mathbf{U}^n - A^{n-1} \mathbf{U}^{n-1} \tag{11}$$

where $\mathbf{U}^n$ refers to the solution state within the space-time element extending from t^{n-1} and t^n, $\mathbf{U}^{n-1}$ refers to the solution state in the lower space-time element, and A^{n-1}, A^n refer to the temporal face areas. This discretization essentially amounts to a fully upwinded flux balance in the time dimension for the space-time element under consideration, accurate up to first-order, where the unit time normal for the upper face is 1 and the unit time normal for the lower face is -1. All spatial components of the normal vector for the upper and lower temporal faces are zero by definition of the problem.

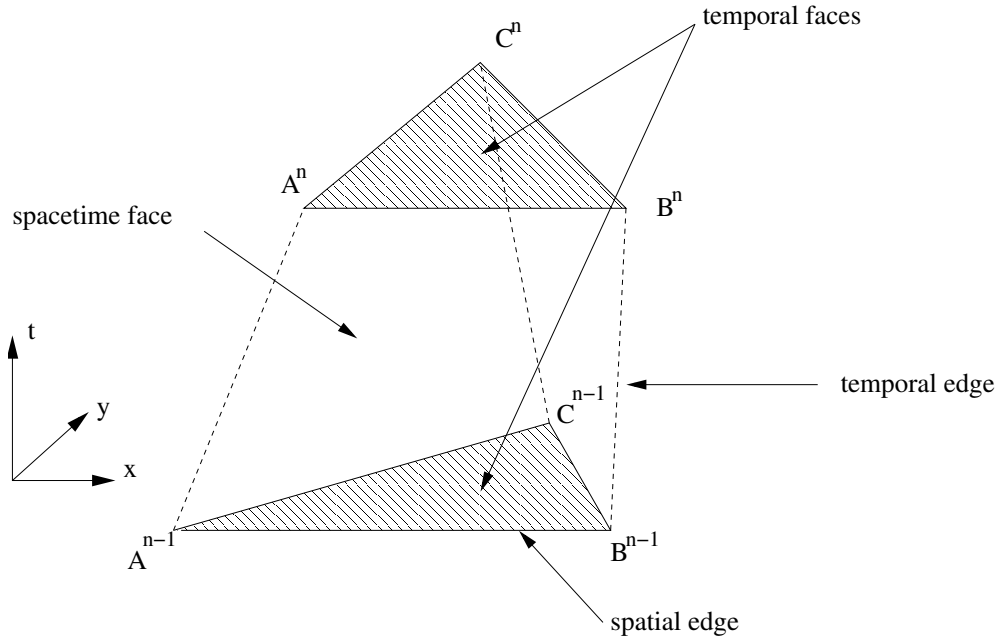


Figure 1. Example of a space-time element constructed from a spatial element at two time locations.

D. Spatial discretization

The flow solver uses a cell-centered finite-volume formulation where the inviscid flux integral S over the inclined space-time faces (i.e. with non-zero spatial components for the normal vector) bounding a space-time element is discretized as:

$$S = \int_T \int_{dB(t)} [\mathbf{F}(\mathbf{U}) - \dot{\mathbf{x}}\mathbf{U}] \cdot \mathbf{n} dB dt = \sum_{i=1}^{n_{edge}} \mathbf{F}_{e_i}^{\perp}(V_{e_i}, \mathbf{U}, \mathbf{n}_{ei}) B_{e_i} \Delta t_{e_i} \tag{12}$$

where B_e represents the L_2 norm of the spatial components of the dimensional normal vector of the face, V_e is the normal face velocity, $\mathbf{n}_e$ is the unit spatial normal vector of the face and is obtained by normalizing the dimensional spatial normal components of the face by B_e, $F_e^{\perp}$ is the normal convective flux across the face, and Δt_e is the height

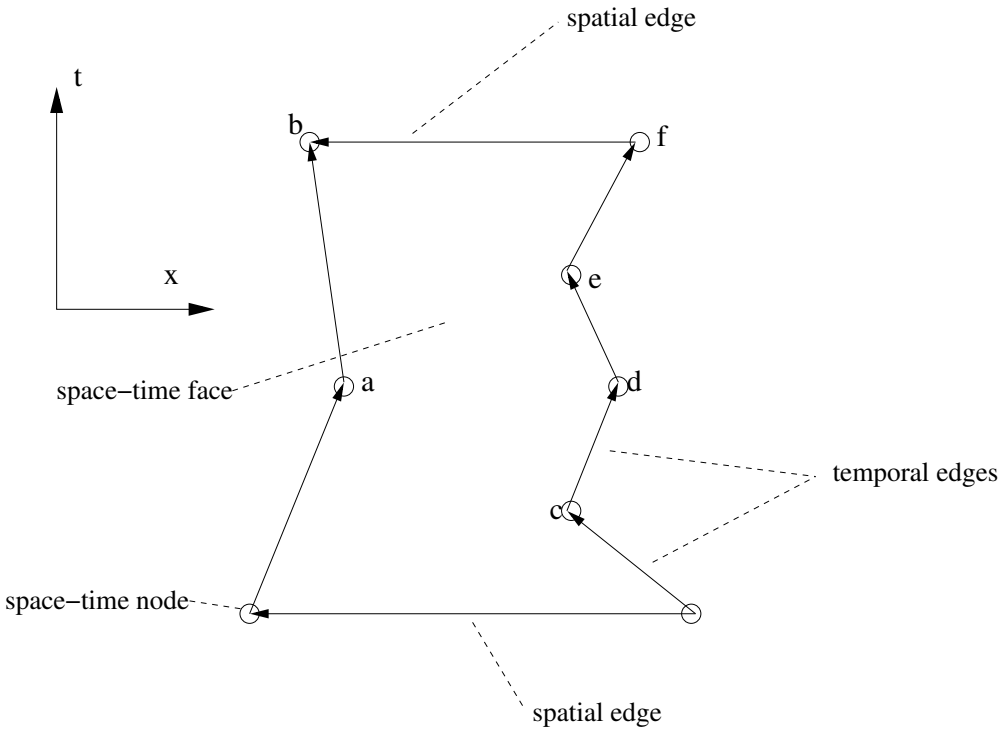


Figure 2. Example of a space-time face bounding a space-time element.

of the face in the time dimension. The normal convective flux across the face is computed using the second-order accurate matrix dissipation scheme[20] as the sum of a central difference and an artificial dissipation term as shown below,

$$\mathbf{F}_e^{\perp} = \frac{1}{2}\left\{\mathbf{F}_L^{\perp}(\mathbf{U}_L, V_e, \mathbf{n}_e) + \mathbf{F}_R^{\perp}(\mathbf{U}_R, V_e, \mathbf{n}_e) + \kappa^{(4)}[T]|[\lambda]|[T]^{-1}\left\{(\nabla^2\mathbf{U})_L - (\nabla^2\mathbf{U})_R\right\}\right\} \tag{13}$$

where $\mathbf{U}_L$, $\mathbf{U}_R$ are the left and right state vectors and $(\nabla^2\mathbf{U})_L$, $(\nabla^2\mathbf{U})_R$ are the left and right undivided Laplacians computed for any space-time element i as

$$(\nabla^2\mathbf{U})_i = \sum_{k=1}^{\text{spatial neighbors}} (\mathbf{U}_k - \mathbf{U}_i) \tag{14}$$

The matrix $[\lambda]$ is diagonal and consists of the eigenvalues (adjusted by normal edge velocity V_e) of the flux Jacobian matrix $\frac{\partial \mathbf{F}^{\perp}}{\partial \mathbf{U}}$, and the matrix $[T]$ consists of the corresponding eigenvectors. The scalar parameter $\kappa^{(4)}$ is empirically determined and controls the amount of artificial dissipation added to the centrally differenced flux. For transonic problems this is usually taken as 0.1. The advantage of using the difference of the undivided Laplacians in the construction of the convective flux is that it offers second-order spatial accuracy without the need for state reconstruction techniques. The normal native flux vector is computed as

$$\mathbf{F}^{\perp} = \begin{pmatrix} \rho(V^{\perp} - V_e) \\ \rho(V^{\perp} - V_e)u + \hat{n}_x p \\ \rho(V^{\perp} - V_e)v + \hat{n}_y p \\ E_t(V^{\perp} - V_e) + pV^{\perp} \end{pmatrix} \tag{15}$$

where the fluid velocity normal to the edge $V^{\perp}$ is defined as $u\hat{n}_x + v\hat{n}_y$, where $\hat{n}_x$ and $\hat{n}_y$ are the spatial components of the unit face normal vector.

E. Mesh deformation strategy

Deformation of the mesh is achieved through the linear tension spring analogy[21,22] which approximates the complete space-time mesh as a network of inter-connected springs. Every spatial and temporal edge defines a spring connected to a space-time node. The spring coefficient is assumed to be inversely proportional to the edge length. Three independent force balance equations for x, y and t are formulated for each node based on displacements of neighboring nodes. This results in a nearest neighbor stencil for the final linear system to be solved. The linear system that relates the interior node displacements in the mesh to known displacements on the boundaries is given as

$$\mathbf{G}(\mathbf{x}) = [\mathbf{K}]\delta\mathbf{x}_{int} - \delta\mathbf{x}_{bound} = 0 \tag{16}$$

where $[\mathbf{K}]$ is the stiffness matrix assembled using the spring coefficients of each of the edges in the computational mesh. Since the equations are decoupled between dimensions, prescribing zero displacements in the time dimension at the boundaries assures that temporal faces have zero spatial normal vector components. The linear mesh motion system is solved using Gauss-Seidel iterations and a comparison of the typical convergence of the mesh motion equations for spatial meshes and space-time meshes is shown in Figure (3).

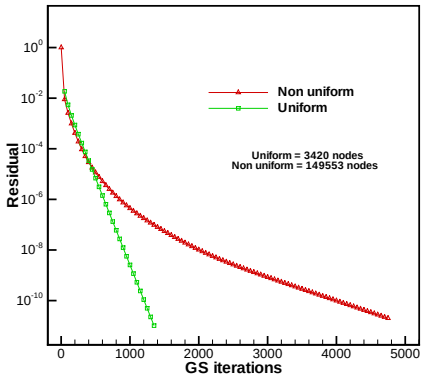


Figure 3. Typical convergence of the mesh motion linear system at a single time-step for spatially uniform time-stepping and a single slab for the case of spatially non-uniform time-stepping.

F. The discrete geometric conservation law (GCL) and space-time face normals

The discrete geometric conservation law (GCL) requires that a uniform flow field be preserved exactly when equation (9) is integrated. In other words the deformation of the computational mesh should not introduce conservation errors in the solution of the flow problem. This translates into $\mathbf{U} = constant$ being an exact solution of equation (9). The space-time face velocities and normal vectors have to be chosen such that this condition is satisfied.

The normal space-time face velocity including the spatiotemporal area of the face can be computed as follows. Collapsing the two spatial edges defining the top and bottom of a space-time face to the same time level, the face can be viewed as an area swept in x, y by a spatial edge deforming from its orientation at time index $n-1$ to its new orientation at time index n. The path followed by the head and tail nodes of the spatial edge between the two orientations is given by the spatial coordinates of the head nodes belonging to the intermediate temporal edges. The area swept by the spatial edge is computed using the Green-Gauss contour integral and used as the face velocity which includes the space-time face area.

Only the dimensional spatial components of the space-time face normal vector are required and these are computed using the time-averaged coordinates of the head and tail nodes belonging to the sweeping spatial edge between the two orientations. Choosing the space-time face normal velocities and face normal vectors in this manner assures that for a constant flow field, the governing equations discretely integrate in space-time to zero.

III. Space-Time Implicit Solution Procedure

Consider the normal case of uniformly advancing spatial elements in time. For the first-order BDF1 time-integration scheme, the flow equations are solved implicitly at each time-step by introducing an implicit residual $\mathbf{R}$ as:

$$\mathbf{R}^n(\mathbf{U}^n, \mathbf{U}^{n-1}, \mathbf{x}^n, \mathbf{x}^{n-1}) = 0 \tag{17}$$

which is linearized with respect to the unknown state $\mathbf{U}^n$ and solved using Newton's method as:

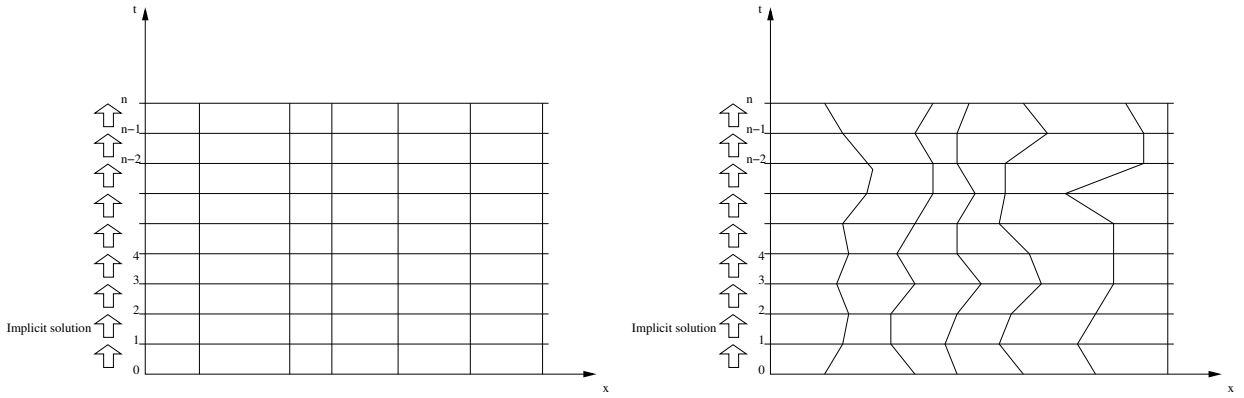
$$\begin{aligned} \left[\frac{\partial \mathbf{R}(\mathbf{U}^k, \mathbf{x})}{\partial \mathbf{U}^k}\right] \delta\mathbf{U}^k &= -\mathbf{R}(\mathbf{U}^k, \mathbf{x}) \\ \mathbf{U}^{k+1} &= \mathbf{U}^k + \delta\mathbf{U}^k \\ \delta\mathbf{U}^k \to 0, \mathbf{U}^{k+1} &= \mathbf{U}^n \end{aligned} \tag{18}$$

In the case of a uniform time-step size for all spatial elements, the state vector $\mathbf{U}$ for each spatial element is unknown at time-index n but is known at the previous time index $n-1$. The solution procedure involves constructing the nonlinear residual $\mathbf{R}$ for each spatial element at the time index n and solving for the state $\mathbf{U}^n$ using Newton's method, where the intermediate linear systems arising in the non-linear solution are solved approximately using Gauss-Seidel iterations. Figures (4(a)) and (4(b)) illustrate the uniform advancement in time with and without the presence of spatial mesh deformation in time for a spatially one-dimensional problem.

In the case of spatially nonuniform time stepping, neighboring spatial elements are advanced in time at different rates. The solution procedure for such a problem involves first the division of the time domain into slabs of some predetermined temporal spacing and then subdividing certain space-time elements appropriately in the time dimension when additional temporal resolution is required. The temporal slab thickness is chosen such that essential nonlinearities in the time dimension are captured. The advancement in time is now carried out from slab to slab, where the solution state $\mathbf{U}$ for all space-time elements in a particular slab are solved for simultaneously in an implicit manner before moving onto the next slab in time. The residual operator for a slab can be written as:

$$\mathbf{R}^n(\mathbf{U}^n, \mathbf{U}^b, \mathbf{x}^n, \mathbf{x}^b) = 0 \tag{19}$$

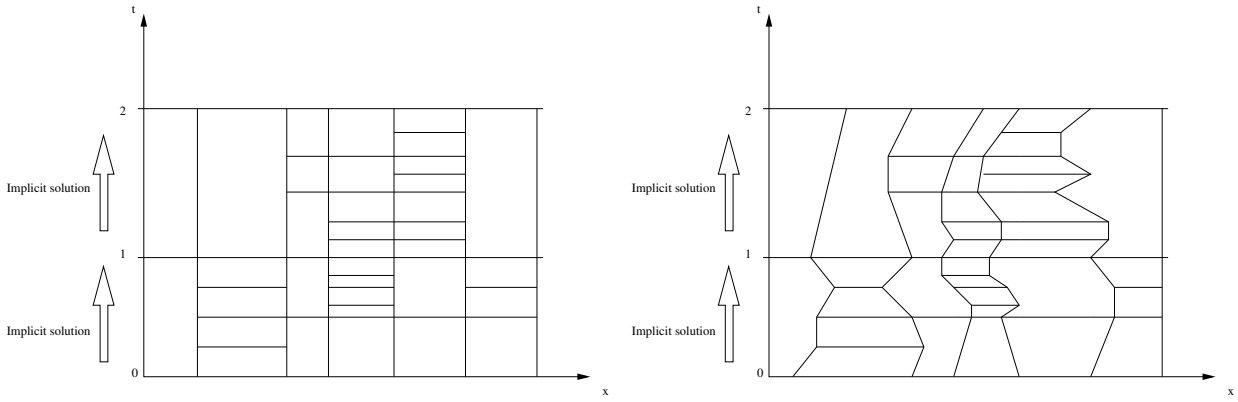
where the index n now refers to a temporal slab and all of the space-time elements contained within it. The index b refers to boundary values for the slab in both space and in time. The solution process involves applying Newton's method on the nonlinear residual operator $\mathbf{R}$ defined over the entire temporal slab. It is important to note that the state vectors of each space-time element in the slab under consideration are unknown and have to be obtained simultaneously irrespective of their location in time within the slab. Figures (5(a)) and (5(b)) illustrate the time advancement procedure for spatially nonuniform time stepping with and without the presence of mesh deformation. A comparison of the typical non-linear convergence of the flow equations at a single time-step for the case of uniform time-stepping and a single slab for the case of non-uniform time-stepping is shown in Figure (6).



(a) Spatially uniform time stepping without mesh deformation

(b) Spatially uniform time stepping with spatial mesh deformation

Figure 4. Illustration of spatially uniform time stepping for a spatially one-dimensional problem.



(a) Spatially nonuniform time stepping without mesh deformation (b) Spatially nonuniform time stepping with spatial mesh deformation

Figure 5. Illustration of spatially nonuniform time stepping for a spatially one-dimensional problem.

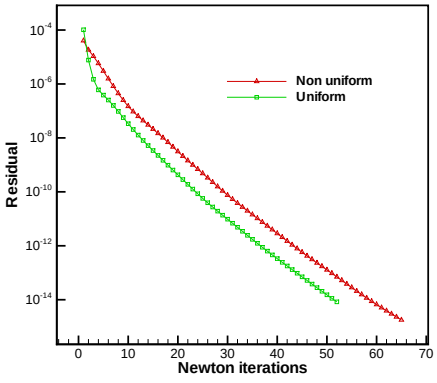


Figure 6. Typical nonlinear convergence of the flow equations at a single time-step for spatially uniform time-stepping and a single slab for the case of spatially non-uniform time-stepping. The non-uniform time-stepping solution was performed within a slab consisting of 211232 space-time elements, while the uniform time-stepping solution consisted of 6601 spatial elements. 20 Gauss-Seidel iterations were employed for the linear systems in both cases.

IV. Adjoint-Based Functional Error Formulation

Consider a functional computed based on the unsteady solutions of the flow and mesh motion equations. Mathematically this may be represented as $L = L(\mathbf{U}, \mathbf{x})$ where L is the functional, $\mathbf{U}$ is the unsteady flow solution set and $\mathbf{x}$ is the set of unsteady mesh solutions. The functional evaluated on two different space-time meshes of discrete resolutions h and H can represented by $L_h(\mathbf{U}_h, \mathbf{x}_h)$ and $L_H(\mathbf{U}_H, \mathbf{x}_H)$. For the purpose of this paper, H refers to some arbitrary coarse resolution space-time mesh and h is a fine resolution space-time mesh constructed by nested subdivision in the time dimension of each space-time element in H using a ratio of 2-to-1. An estimate of $L_h(\mathbf{U}_h, \mathbf{x}_h)$ can be obtained based on the Taylor expansion around the functional $L_h(\mathbf{U}_h^H, \mathbf{x}_h^H)$, where $\mathbf{U}_h^H$ and $\mathbf{x}_h^H$ refer to projections of the flow and mesh solutions from the coarse to the fine level. Assuming that the coarse space-time domain flow solution $\mathbf{U}_H$ and the coarse space-time domain mesh coordinates $\mathbf{x}_H$ have been obtained via full convergence of the respective equations, the Taylor series expansion of the exact fine space-time domain functional can be written as:

$$L_h(\mathbf{U}_h, \mathbf{x}_h) = L_h(\mathbf{U}_h^H, \mathbf{x}_h^H) + \left[\frac{\partial L}{\partial \mathbf{U}}\right]_{\mathbf{U}_h^H, \mathbf{x}_h^H} \left(\mathbf{U}_h - \mathbf{U}_h^H\right) + \left[\frac{\partial L}{\partial \mathbf{x}}\right]_{\mathbf{x}_h^H, \mathbf{U}_h^H} \left(\mathbf{x}_h - \mathbf{x}_h^H\right) + \cdots \tag{20}$$

The procedure for computing an unsteady solution involves obtaining the solution to the non-linear residual operator $\mathbf{R}(\mathbf{U}, \mathbf{x})$ within each temporal slab. The non-linear operator $\mathbf{R}(\mathbf{U}, \mathbf{x})$ is constructed by the previously described discretization of the governing flow equations. Expanding a fine space-time domain residual set about the coarse space-time domain set of residuals yields:

$$\mathbf{R}_h(\mathbf{U}_h, \mathbf{x}_h) = \mathbf{R}_h(\mathbf{U}_h^H, \mathbf{x}_h^H) + \left[\frac{\partial \mathbf{R}}{\partial \mathbf{U}}\right]_{\mathbf{U}_h^H, \mathbf{x}_h^H} \left(\mathbf{U}_h - \mathbf{U}_h^H\right) + \left[\frac{\partial \mathbf{R}}{\partial \mathbf{x}}\right]_{\mathbf{x}_h^H, \mathbf{U}_h^H} \left(\mathbf{x}_h - \mathbf{x}_h^H\right) + \cdots = 0 \tag{21}$$

and since the non-linear residual operator $\mathbf{R}$ must vanish for a converged solution within each time slab, an estimate for the error vector $(\mathbf{U}_h - \mathbf{U}_h^H)$ in equation (20) can obtained by rearranging equation (21) as:

$$\left(\mathbf{U}_h - \mathbf{U}_h^H\right) \approx -\left[\frac{\partial \mathbf{R}}{\partial \mathbf{U}}\right]^{-1}_{\mathbf{U}_h^H, \mathbf{x}_h^H} \left\{ \mathbf{R}_h(\mathbf{U}_h^H, \mathbf{x}_h^H) + \left[\frac{\partial \mathbf{R}}{\partial \mathbf{x}}\right]_{\mathbf{x}_h^H, \mathbf{U}_h^H} \left(\mathbf{x}_h - \mathbf{x}_h^H\right) \right\} \tag{22}$$

It should be noted that this is merely an estimate for the error between $\mathbf{U}_h$ and $\mathbf{U}_h^H$ since higher-order terms in the Taylor expansion are ignored. Substituting equation (22) into equation (20) results in:

$$L_h(\mathbf{U}_h, \mathbf{x}_h) \approx L_h(\mathbf{U}_h^H, \mathbf{x}_h^H) \underbrace{- \left[\frac{\partial L}{\partial \mathbf{U}}\right]_{\mathbf{U}_h^H, \mathbf{x}_h^H} \left[\frac{\partial \mathbf{R}}{\partial \mathbf{U}}\right]^{-1}_{\mathbf{U}_h^H, \mathbf{x}_h^H} \left\{ \mathbf{R}_h(\mathbf{U}_h^H, \mathbf{x}_h^H) + \left[\frac{\partial \mathbf{R}}{\partial \mathbf{x}}\right]_{\mathbf{x}_h^H, \mathbf{U}_h^H} \left(\mathbf{x}_h - \mathbf{x}_h^H\right) \right\} + \left[\frac{\partial L}{\partial \mathbf{x}}\right]_{\mathbf{x}_h^H, \mathbf{U}_h^H} \left(\mathbf{x}_h - \mathbf{x}_h^H\right)}_{\varepsilon_{cc}} \tag{23}$$

The right-hand-side in equation (23) can be described as an estimate for the exact functional evaluated directly on the fine space-time mesh. Based on the derivation it is approximately equal to the sum of the functional evaluated on the fine space-time mesh using a projected solution and projected mesh coordinates from the coarse space-time mesh and a computable correction term ε_{cc}.

The correction term ε_{cc} as defined above is:

$$\varepsilon_{cc} = -\left[\frac{\partial L}{\partial \mathbf{U}}\right]_{\mathbf{U}_h^H, \mathbf{x}_h^H} \left[\frac{\partial \mathbf{R}}{\partial \mathbf{U}}\right]^{-1}_{\mathbf{U}_h^H, \mathbf{x}_h^H} \left\{ \mathbf{R}_h(\mathbf{U}_h^H, \mathbf{x}_h^H) + \left[\frac{\partial \mathbf{R}}{\partial \mathbf{x}}\right]_{\mathbf{x}_h^H, \mathbf{U}_h^H} \left(\mathbf{x}_h - \mathbf{x}_h^H\right) \right\} + \left[\frac{\partial L}{\partial \mathbf{x}}\right]_{\mathbf{x}_h^H, \mathbf{U}_h^H} \left(\mathbf{x}_h - \mathbf{x}_h^H\right) \tag{24}$$

Since computing, storing and inverting the flow Jacobian matrix is expensive, a flow adjoint variable $\Lambda_{\mathbf{U}}$ is defined to aid the evaluation procedure. The flow adjoint variable is defined as:

$$\Lambda_{\mathbf{U}_h^H}^T \big|_{\mathbf{U}_h^H \mathbf{x}_h^H} = -\left[\frac{\partial L}{\partial \mathbf{U}}\right]_{\mathbf{U}_h^H, \mathbf{x}_h^H} \left[\frac{\partial \mathbf{R}}{\partial \mathbf{U}}\right]^{-1}_{\mathbf{U}_h^H \mathbf{x}_h^H} \tag{25}$$

Transposing and rearranging equation (25) yields

$$\left[\frac{\partial \mathbf{R}_h}{\partial \mathbf{U}_h}\right]^T_{\mathbf{U}_h^H \mathbf{x}_h^H} \Lambda_{\mathbf{U}h} \big|_{\mathbf{U}_h^H, \mathbf{x}_h^H} = -\left[\frac{\partial L}{\partial \mathbf{U}}\right]^T_{\mathbf{U}_h^H, \mathbf{x}_h^H} \tag{26}$$

The solution of equation (26) involves projecting the coarse space-time flow solution and mesh coordinates onto the fine space-time mesh, reconstructing the flow Jacobian matrices on the fine space-time mesh and then solving the linear system iteratively for the flow adjoint variable. Since the goal is to avoid direct solutions of any nature on the fine space-time mesh, an approximation is used where the adjoint is evaluated on the coarse space-time mesh and then projected onto the fine mesh. This circumvents the expensive evaluations on the fine mesh. Equation (26) recast on the coarse space-time mesh becomes:

$$\left[\frac{\partial \mathbf{R}}{\partial \mathbf{U}}\right]^T_{\mathbf{U}_H,\mathbf{x}_H} \Lambda_{\mathbf{U}H} = -\left[\frac{\partial L}{\partial \mathbf{U}}\right]^T_{\mathbf{U}_H,\mathbf{x}_H} \tag{27}$$

The coarse adjoint variable can then be projected onto to the fine domain as:

$$\Lambda_{\mathbf{U}h} = I_h^H \Lambda_{\mathbf{U}H} \tag{28}$$

where I_h^H is some appropriate projection operator. Once the vector of adjoint variables $\Lambda_{\mathbf{U}H}$ has been obtained the first contributions to the computable correction term may be determined by projecting the flow adjoint onto the fine space-time mesh and then evaluating a vector inner product as follows:

$$\varepsilon_{cc_1} = \Lambda_{\mathbf{U}_h^H}{}^T R_h(\mathbf{U}_h^H, \mathbf{x}_h^H) \tag{29}$$

ε_{cc_1} represents the contribution from the flow equations to the error arising due to insufficient temporal resolution within each space-time element. The residual $\mathbf{R}_h(\mathbf{U}_h^H, \mathbf{x}_h^H)$ is non-zero since it is computed using the projection of the coarse space-time flow solution onto the fine space-time mesh. Closer examination reveals that equation (29) is actually a summation of the inner products of the flow adjoint and the non-zero residual within each space-time element on the fine space-time mesh. The remaining contributions to the total correction term may be written in combined form as:

$$\varepsilon_{cc_2} = \left\{ \Lambda_{\mathbf{U}_h}^T \left[\frac{\partial \mathbf{R}_h}{\partial \mathbf{x}_h}\right]_{\mathbf{U}_h^H \mathbf{x}_h^H} + \left[\frac{\partial L_h}{\partial \mathbf{x}_h}\right]_{\mathbf{x}_h^H} \right\} \left(\mathbf{x}_h - \mathbf{x}_h^H\right) = \lambda_{\mathbf{x}} \left(\mathbf{x}_h - \mathbf{x}_h^H\right) \tag{30}$$

using the notation $\lambda_{\mathbf{x}}$ as shorthand for the large bracketed term in the middle of the above equation. We now write a Taylor expansion of the mesh residual equation $\mathbf{G}$ on the fine space-time mesh about its value constructed using projected mesh coordinates from the coarse to the fine meshes as:

$$\mathbf{G}(\mathbf{x}_h) \quad = \quad \mathbf{G}(\mathbf{x}_h^H) + \left[\frac{\partial \mathbf{G}}{\partial \mathbf{x}}\right]_{\mathbf{x}_h^H} (\mathbf{x}_h - \mathbf{x}_h^H) + \cdots = 0 \tag{31}$$

Here the residual $\mathbf{G}(\mathbf{x}_h^H)$ is evaluated using the projected values for $\mathbf{x}_{int}$ from the coarse space-time mesh to the fine mesh and the exact values of $\mathbf{x}_{bound}$ on the fine space-time mesh. The boundary coordinates $\mathbf{x}_{bound}$ are only a function of time and can be easily evaluated on the fine space-time mesh. Rearranging and simplifying the mesh residual Taylor expansion yields:

$$(\mathbf{x}_h - \mathbf{x}_h^H) = -\left[\frac{\partial \mathbf{G}}{\partial \mathbf{x}}\right]^{-1}_{\mathbf{x}_h^H} \mathbf{G}(\mathbf{x}_h^H) \tag{32}$$

However, refering to Equation (16) it is seen that the linearization of the mesh residual $\mathbf{G}$ with respect to the mesh coordinates $\mathbf{x}$ is simply the mesh stiffness matrix $[\mathbf{K}]$. Therefore,

$$(\mathbf{x}_h - \mathbf{x}_h^H) = -[\mathbf{K}]^{-1} \mathbf{G}(\mathbf{x}_h^H) \tag{33}$$

Substituting this back into equation (30) gives us an expression for the second and final contribution to the computable correction term as:

$$\varepsilon_{cc_2} = -\lambda_{\mathbf{x}_h} [\mathbf{K}]^{-1} \mathbf{G}(\mathbf{x}_h^H) \tag{34}$$

Defining a mesh adjoint variable $\Lambda_{\mathbf{x}}$ permits an iterative solution and avoids inversion of the stiffness matrix.

$$[\mathbf{K}]^T \Lambda_{\mathbf{x}h} = -\lambda_{\mathbf{x}h}{}^T \tag{35}$$

As in the case of the flow adjoint, the above system is solved on the coarse space-time mesh and the coarse mesh adjoint variable then projected onto the fine space-time mesh by some appropriate operator as:

$$[\mathbf{K}]^T \Lambda_{\mathbf{x}H} = -{\lambda_{\mathbf{x}H}}^T \tag{36}$$

$$\Lambda_{\mathbf{x}h} = I_h^H \Lambda_{\mathbf{x}H} \tag{37}$$

Both a mesh adjoint variable vector and a flow adjoint variable vector must be solved for within each time slab during the backward sweep in time. Details on the exact solution process involving the backward sweep in time can be found in Ref.[23] The final form for the second contribution to the correction term can now be expressed as:

$$\varepsilon_{cc_2} = {\Lambda_{\mathbf{x}_h^H}}^T \mathbf{G}(\mathbf{x}_h^H) \tag{38}$$

ε_{cc_2} represents the contribution from the mesh motion equations to the temporal resolution error. Just as in the case of the temporal resolution error due to the flow equations, equation (38) represents a summation of inner products of the mesh adjoint and the mesh residual within each space-time element. Although the mesh motion equations are linear, the resulting mesh coordinate variations in time are not linear, since these are driven by the prescribed surface mesh displacements, which in our following example are non-linear in time. This causes the mesh residual to be non-zero when computed on the fine space-time domain using projected mesh coordinates from the coarse space-time domain.

The contribution to the total correction from each space-time element can be interpreted as the representation of the error in the functional arising from that space-time element. The resulting distribution in space-time of the total error ε_{cc} can thus be conveniently used as the criteria for identifying regions in space-time that require higher temporal resolution.

V. General Implementation Details

A. Local error indicator for adaptation

The local error indicator proposed in Ref.[4] is used as the temporal discretization error indicator for adaptation purposes. The method involves obtaining a solution using the first-order BDF1 discretization of the time derivative term in the governing equations, and then reevaluating the time derivative on the basis of the obtained solution using the BDF2 time discretization. The local temporal error is then estimated as the L_2 norm of the difference between BDF1 and BDF2-based time derivative terms as shown below.

$$e_{local} = \left|\left| \left[\frac{d\mathbf{U}}{dt}\right]_{BDF2} - \left[\frac{d\mathbf{U}}{dt}\right]_{BDF1} \right|\right|_2 \tag{39}$$

Strictly speaking, this formulation is only valid for the case of static meshes. For the case of dynamic meshes, the residual operator constructed using a discretization of higher temporal order of accuracy should be used as an estimate of the local error. However, the space-time discretization developed in this work is limited to first-order temporal accuracy and a valid GCL compliant second-order accurate discretization is yet to be developed. For this reason, we choose to utilize the error indicator shown above, as is, and assume that small geometric variations in the mesh have an acceptably low impact on the estimated error.

B. Code details

The space-time adaptive solver consists of four separate codes, namely a mesh preprocessor, a mesh adaptation code, the flow solver and the adjoint solver. The mesh preprocessor takes in a standard unstructured spatial mesh and constructs a space-time mesh based on the chosen temporal parameters for the flow problem. The required temporal parameters are the number of temporal slabs for the unsteady problem and the thickness (in time) of each temporal slab. Slabs are normally taken to be of uniform thickness, although varying thicknesses are permitted. The initial mesh is constructed such that each spatial element in the slab consists of two uniform time-steps to begin with. This is done mainly for the reason of error estimation after the first unsteady solution, where at least a two point stencil is required to construct the BDF2-based time derivative term.

The flow solver computes a solution using this space-time mesh and invokes a local temporal error estimation routine which estimates the temporal discretization error within each element of each slab. The flow solver also writes out the solution to the hard drive in a piecewise manner (slab-by-slab) during the solution process. If an adjoint-based error estimate is required, the adjoint solver is invoked which reads in the solution from the hard drive and performs

a backward sweep in time to compute the necessary adjoint variables. The adjoint solver also constructs the residuals on the fine space-time mesh and estimates the functional error by performing the inner product between the adjoint variables and the residuals.

In the case of local error-based adaptation, elements within a slab that have temporal discretization error higher than the mean temporal error of the slab as a whole are flagged for refinement. In the case of adjoint-based adaptation, space-time elements are first ranked by their contribution to the total predicted error in the functional, and then working downward from the element with the highest contribution, elements are flagged for refinement until 99% of the total error has been accounted for.

The mesh adaptation routine takes in the old space-time mesh and the list of space-time elements that have been flagged for refinement and constructs a new space-time mesh. Space-time elements that are to be refined are subdivided into two equal elements in the time dimension. A maximum time-step size ratio of two is strictly enforced between neighboring space-time elements in the space dimension in order avoid large jumps in the time-step sizes of neighbors. However, this ratio is currently not enforced in the temporal direction. Once the new space-time mesh has been constructed, it is written to disk and picked up by the flow solver for the next unsteady solution.

Since at any single adaptation cycle, a space-time element is at most subdivided into two, comparisons are made against uniform time-stepping where the number of time-steps is increased by a factor of two and the time-step sizes are reduced by a factor of two for all spatial elements

VI. Results

A. Case (A): Convection of an isentropic vortex

The first example problem involves the convection of an isentropic vortex through a rectangular domain combined with dynamic mesh deformation. The free-stream conditions for the problem are given by the state $[\rho, u, v, p]$=[1,0.5,0,1]. The initial spatial mesh for the problem consists of 10000 elements and extends from 0 to 14 in the x-direction and from 0 to 7 in the y-direction and is shown in Figure (7). A vortex is seeded in the flow at $[x, y]$=[3.5,3.5] and allowed to convect with the free-stream. Further details on how the vortex is seeded in the flow and the respective parameters for the vortex strength and region of influence can be found in Ref.[24]

The time domain for this problem extends from 0 to 15 in non-dimensional time, indicating that the final position of the vortex is at x=11. For the case of non-uniform time-stepping, the time domain is split into 15 slabs of size 1 each. Since the starting space-time mesh consists of two time-steps of size 0.5 for all spatial elements contained within, the comparison case of uniform time-stepping begins with 30 time-steps of size 0.5 each. At each adaptation cycle, for the uniform time-stepping case, the time-step size for all spatial elements is halved and the number of time-steps is doubled. For the non-uniform case, the local error is used to adapt the time-step sizes within each slab. Both the uniform and non-uniform time-step solutions at the end of the time domain (i.e. t=15) are compared against a uniformly time-stepped reference solution obtained with 32 times the number of time-steps than the finest resolution uniform time-step case, which consists of 960 time-steps. While an exact analytic solution to the vortex convection problem is available, it is not used for comparison purposes since it does not take into account the dissipation of the vortex due to spatial discretization error. The dynamic motion of the mesh is governed by sinusoidal functions of x, y, t. Although no mesh motion solution is required for this problem, prescribed mesh motion is included in order demonstrate the GCL capabilities of the solver. The perturbation of every single node in the space-time mesh is uniquely defined by analytic mesh motion functions, details of which can be found in Ref.[25] Figure (8) shows the density contours of the solution to the unsteady problem at the final time of t=15 and also illustrates the deformation of the mesh compared to the baseline or starting configuration. This particular solution is that after six adaptation cycles of the non-uniform time-stepping solver. Both uniform and non-uniform time-stepping cases were terminated after 6 adaptation cycles.

Figure (9(a)) compares the error in the solution of the uniform time-stepping case and the non-uniform time-stepping case at each adaptation cycle and indicates good agreement. Figures (9(b)) and (9(c)) compare the solution errors for the uniform and non-uniform time-stepping cases to CPU cost and to the total number of degrees-of-freedom required to solve the problem. The number of degrees-of-freedom in the case of uniform time-stepping is the product of the number of spatial elements and the number of time-steps, while for the non-uniform time-stepping case, it is the sum of space-time elements over all temporal slabs. For this particular problem, it can be seen from the plots that the spatially non-uniform time-stepping procedure is able to achieve similar error levels as uniform time-stepping with approximately 25% less computational expense and approximately one fourth of the number degrees-of-freedom. Figure (10) compares the density along the centerline of the spatial mesh at various time-levels between the uniform

and non-uniform time-stepping cases and indicates excellent agreement. Figures (11(a)) and (11(b)) indicate the distribution of the local error at the top of the first temporal slab and the top of the last (15th) temporal slab after the final (6th) adaptation cycle. The distributions shown in these figures include the local error of all the space-time elements laying underneath the spatial elements shown in the plots but within the same slab. Figures (11(c)) and (11(d)) compare the number time-steps a spatial element takes within the slab in slabs 1 and 15 after the final adaptation cycle. It can be clearly seen that most of the spatial elements take no more than 5 time-steps while spatial elements where the vortex has a strong influence take nearly 60 time-steps.

B. Case (B): Transonic pitching airfoil

The second example is that of a transonic pitching NACA64A010 airfoil. The airfoil pitches sinusoidally around its quarter-chord location in a free-stream Mach number of 0.8 with a mean angle-of-attack of zero. The amplitude of pitch is 2.5 degrees and the reduced frequency is 0.1. The time domain for this problem was chosen such that one pitch period is completed starting from a zero angle-of-attack. The unsteady solution is initiated using the steady solution around the airfoil at a zero angle-of-attack. The first quarter of the pitch cycle is modified to follow a scaled and phase shifted cosine profile rather than the default sinusoidal profile in order to assure a smooth rather than an impulsive start of the unsteady motion from steady state.

The spatially non-uniform time-stepping case consists of 18 equally sized temporal slabs of temporal thickness 1.74533 in non-dimensional time. Since the starting space-time mesh within each slab consists of each spatial element taking two time-steps, the spatially uniform time-stepping case begins with 36 time-steps, each of time-step size 0.87266. Figure (12) shows the spatial mesh for this problem and consists of 6601 spatial elements. The mesh motion equations are solved for this problem with prescribed displacements applied to the surface of the airfoil. Since different mesh motion equations are solved for the non-uniform and uniform time-stepping cases, there is no guarantee that the interior mesh points match up between these cases at the end of the time domain. However, the boundaries have prescribed displacements and they match between the cases. The adjoint-based error indicator is employed for adaptation of the space-time mesh for this example problem. The lift coefficient of the airfoil at the final time is used as the target functional. All comparisons are made against a reference functional computed using a solution obtained with 16000 uniform time-steps. Adaptation was terminated after 7 cycles for the spatially non-uniform time-stepping case and 6 cycles for the spatially and temporally uniform time-stepping case. The results are also compared against adjoint-based adaptation where all spatial elements are uniformly advanced in time but with time-step sizes that are different at different locations in time.

Figures (13(a)) and (13(b)) show the convergence of the functional and the error in the functional with respect to the total degrees-of-freedom in the problem. The plot indicates that adjoint-based adaptation with spatially non-uniform time-stepping is able to outperform both uniform time-stepping with uniform time-step sizes and adjoint-based adaptation of the time-step size in spatially uniform time-stepping. For similar error levels in the functional a reduction in the number of degrees-of-freedom by a factor anywhere between 3 and 4 is observed.

Figures (14(a)) and (14(b)) compare the functional relevant error at the middle of the pitch cycle during the first adaptation cycle. As expected, the adjoint-based error indicates that high temporal gradient features such as the moving shock waves on the upper and lower surfaces of the airfoil have to be stepped with high resolution in time. However, the zoomed out view in the farfield of the airfoil shows an unexpected result. The error distribution indicates that there is a large bubble of high error both above and below the airfoil but significantly away from it. The example problem clearly demonstrates the advantage of using adjoint-based error estimates for adaptation. More importantly, the results also demonstrate the advantage of using a space-time framework to further harness the strength of adjoint equations and adapt the time-step size in a spatially local sense. Figures (15(a)) and (15(b)) show the number of time-steps each spatial element in slab 9 takes at the final adaptation cycle. It can be seen that most of the elements in the vicinity of the airfoil take no more than 30 time-steps and there are also large regions which require only 5 time-steps. Only a small band of spatial elements which contain the shock waves take the largest number of time-steps of 60. In the farfield, it can be seen that the large bubble above and below the airfoil also require about 60 time-steps. However, the spatial sizes of the elements in these regions are large, indicating that only a few spatial elements require small time-steps in this region resulting in considerable reduction of total degrees-of-freedom. By the same argument, significant reduction in total degrees-of-freedom is also realized from regions close to the airfoil that are covered by several small spatial elements but take very few time-steps.

It should be noted that the results do not compare favorably with spatially and temporally uniform time-stepping when viewed from a computational expense point of view for this example problem. This is a direct consequence of the growth in size of the system of equations to be solved within each slab as the adaptation procedure progresses. Since no multigrid algorithm is applied and only Gauss-Seidel iterations are used in the solution of the arising linear systems,

larger systems converge significantly slower. However, this is not an issue with spatially uniform time-stepping since the size of the system to be solved at each time-step remains the same and is equal to the number of spatial elements chosen for the problem.

VII. Concluding Remarks and Future Work

A space-time finite-volume formulation was developed to treat both space and time dimensions in a unified manner. The method permits non-uniform advancement of spatial elements in time and offers the possibility reducing overall computational expense in unsteady flow solutions. The first example problem of a convecting vortex clearly shows that reduction in cost with almost no loss in solution accuracy is possible by varying the time-step size spatially. The second example problem of the transonic pitching airfoil, where an adjoint-based error indicator was utilized for adaptation, demonstrated that significant reduction in the total degrees-of-freedom is possible. However, several issues remain to be addressed particularly in relation to the solution cost. The problem of increasing mesh motion solution cost can easily be addressed by implementing multigrid to remove system size dependence on convergence. However, existing multigrid methods perform poorly for purely hyperbolic problems and cannot be used effectively to accelerate convergence in a space-time framework for the flow equations due to the directionality of information propagation in the time dimension. The development and implementation of a multigrid algorithm for the flow equations in a space-time framework is a topic future research.

Ultimately, the developed method for solving unsteady problems should find applications particularly in the context of rotorcraft and wind turbines, where large differences exist between the flow close to the hub and flow close to the tips of the blades.

VIII. Acknowledgements

This work was sponsored by the Office of Naval Research under grant number N00014-09-1-1060 and by the National Science Foundation under grant number OCI-0960490.

References

[1]Jameson, A., Schmidt, W., and Turkel, E., "Numerical Solutions of the Euler Equations by Finite Volume Methods Using Runge-Kutta Time-Stepping Schemes," *AIAA 14th Fluid and Plasma Dynamic Conference, Palo Alto, CA*, 1981, AIAA Paper 81–1259.

[2]Jameson, A., "Time Dependent Calculations Using Multigrid, with Applications to Unsteady Flows Past Airfoils and Wings," *AIAA 10th Computational Fluid Dynamics Conference, Honolulu, HI*, 1991, AIAA Paper 91–1596.

[3]Carpenter, M. H., Viken, S., and Nielsen, E., "The Efficiency of High-Order Temporal Schemes," AIAA Paper 2003-0086.

[4]Carpenter, M. H. and Vatsa, V., "Higher Order Temporal Schemes with Error Controllers for Unsteady Navier-Stokes Equations," *17th AIAA Computational Fluid Dynamics Conference, Toronta, Ontario, Canada, June 2005*, 2005, AIAA Paper 2005–5245.

[5]Bijl, H., Carpenter, M., Vatsa, V., and Kennedy, C., "Implicit Time Integration Schemes for the Unsteady Compressible Navier-Stokes Equations - Laminar Flow," *Journal of Computational Physics*, Vol. 179-1, 2002, pp. 313–329.

[6]Mavriplis, D. and Yang, Z., "Construction of the discrete geometric conservation law for high-order time-accurate simulations on dynamic meshes," *Journal of Computational Physics*, Vol. 213, 2006, pp. 557–573.

[7]Yang, Z. and Mavriplis, D. J., "Higher-Order Time Integration Schemes for Aeroelastic Applications on Unstructured Meshes," *AIAA Journal*, Vol. 45-1, January 2007, pp. 138–150.

[8]Nadarajah, S., McMullen, M., and Jameson, A., "Non-Linear Frequency Domain Based Optimum Shape Design for Unsteady Three-Dimensional Flow," *Proceedings of the 44th Aerospace Sciences Meeting and Exhibit, Reno NV*, 2006, AIAA Paper 2006–387.

[9]Nadarajah, S., McMullen, M., and Jameson, A., "Optimal Control of Unsteady Flows Using Time Accuraate and Non-Linear Frequency Domain Methods," *33rd AIAA Fluid Dynamics Conference and Exhibit, Orlando, FL, June 23-26,*, 2003, AIAA Paper 2003–3875.

[10]Jameson, A., "Acceleration of Transonic Potential Flow Calculations on Arbitrary Meshes by the Multiple Grid Method," *Fourth AIAA Computational Fluid Dynamics Conference, Williamsburg, VA*, 1979, AIAA Paper 79–1458.

[11]Mavriplis, D., "Multigrid Techniques for Unstructured Meshes," *Notes prepared for 26th Computational Fluid Dynamics Lecture Series Program of the von Karman Institute of Fluid Dynamics, Rhode St Genese, Belgium*, 1995.

[12]Melson, N. D., Sanetrik, M. D., and Atkins, H. L., "Time-accurate Navier-Stokes calculations with multigrid acceleration," *6th Copper Mountain Conf. on Multigrid Methods*, 1993, pp. 423–439, NASA Conference Publication 3224.

[13]Butcher, J. C., *Numerical methods for ordinary differential equations*, Wiley, Chicester, UK, 2003.

[14]Lambert, J. D., *Numerical methods for ordinary differential systems*, Wiley, Chicester, UK, 1991.

[15]Mani, K. and Mavriplis, D. J., "Discrete Adjoint based Time-Step Adaptation and Error Reduction Unsteady Flow Problems," *18th AIAA Computational Fluid Dynamics Conference , Miami FL*, 2007, AIAA Paper 2007–3944.

[16]Mani, K. and Mavriplis, D. J., "Error Estimation and Adaptation for Functional Outputs in Time-Dependent Flow Problems," *47th AIAA Aerospace Sciences Meeting including The New Horizons Forum and Aerospace Exposition, Orlando, Florida, Jan. 5-8, 2009*, 2009, AIAA Paper 2009–1495.

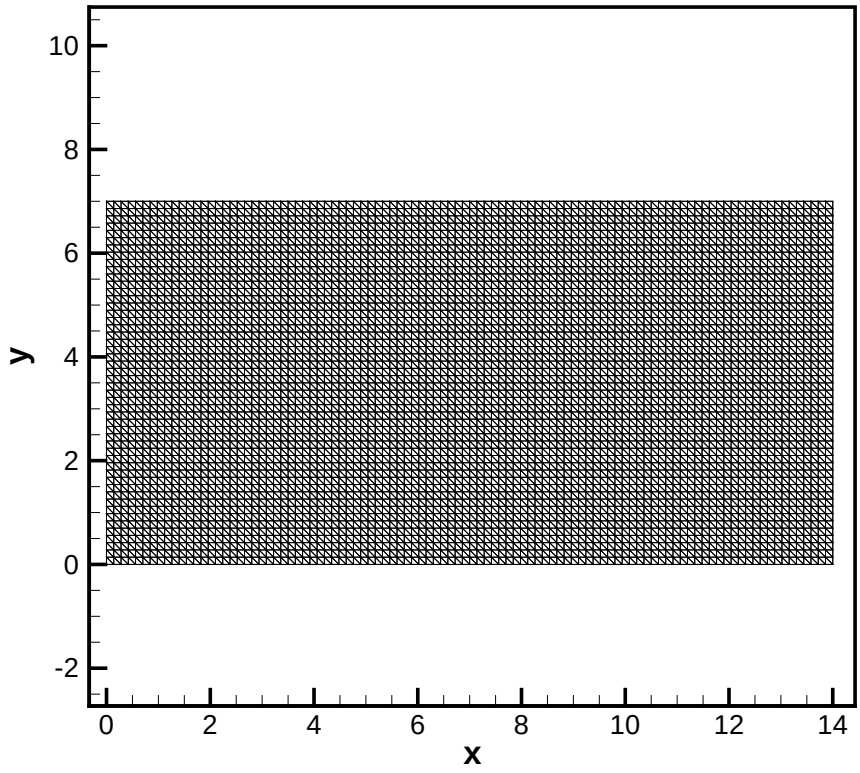


Figure 7. Initial spatial computational mesh consisting of 10000 spatial elements for the vortex convection example problem.

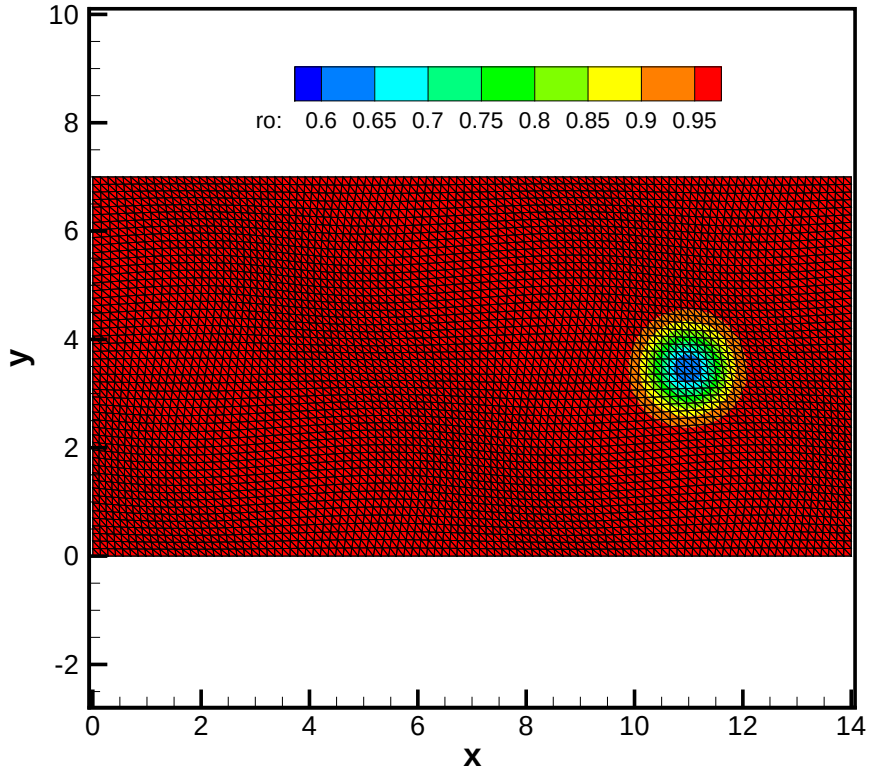
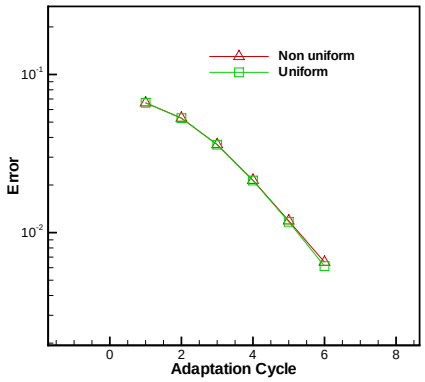
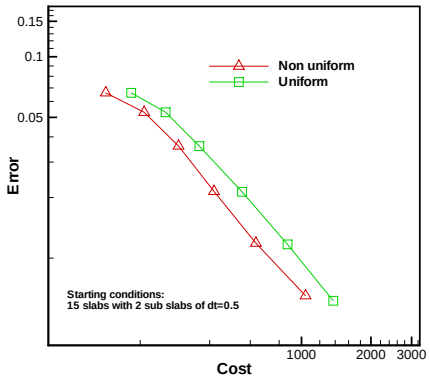


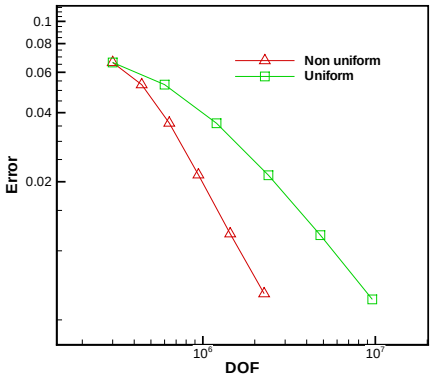
Figure 8. Density contours for the vortex convection problem at the final time of t=15. This solution is from the spatially non-uniform time-stepping case after 6 adaptation cycles.



(a) Solution error comparison at each adaptation cycle.



(b) Solution error convergence with respect to CPU cost.



(c) Solution error convergence with respect to total degrees-of-freedom.

Figure 9. Comparison of solution error between spatially uniform and non-uniform time-stepping for the convecting vortex example problem.

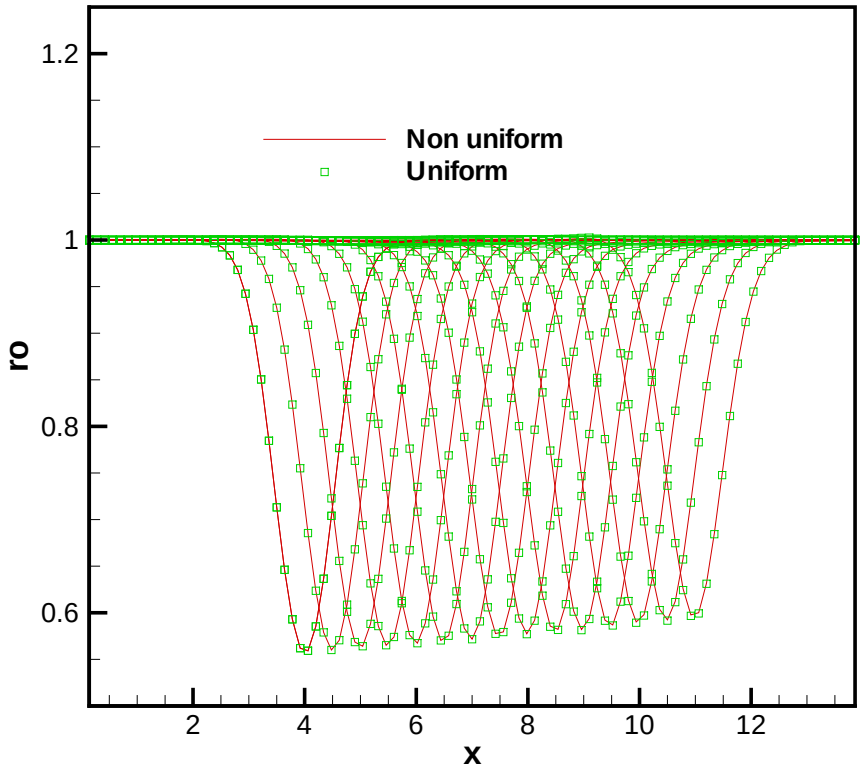
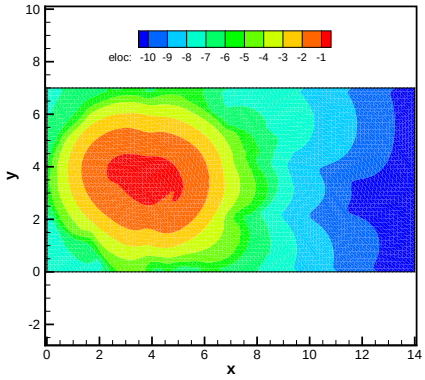
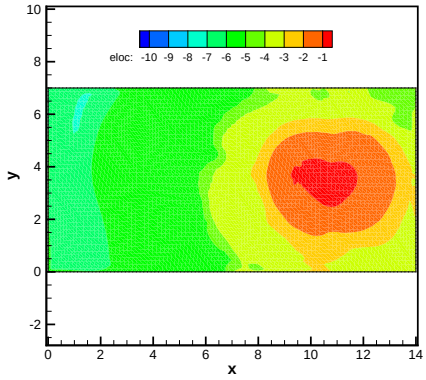


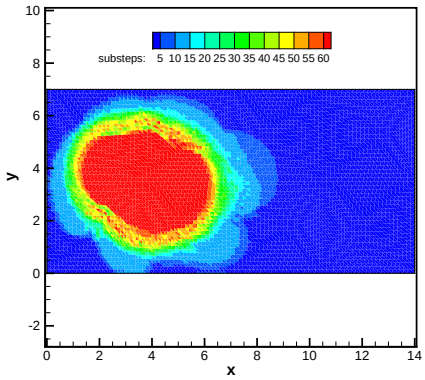
Figure 10. Comparison of mesh centerline density at different time-levels for the convecting vortex example problem.



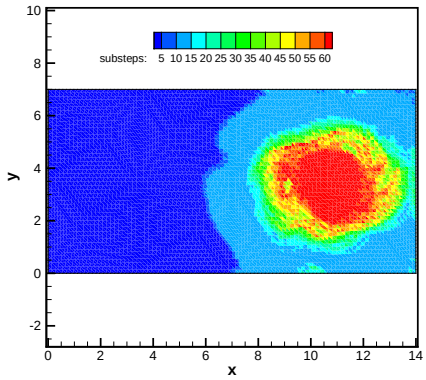
(a) Distribution of local error at the top of slab 1.



(b) Distribution of local error at the top of slab 15.



(c) Number of time-steps per spatial element at the top of slab 1.



(d) Number of time-steps per spatial element at the top of slab 15.

Figure 11. Comparison of local error distribution and number of time-steps per spatial element at the top of slabs 1 and 15 for the convecting vortex example problem. The plots are at the final adaptation cycle.

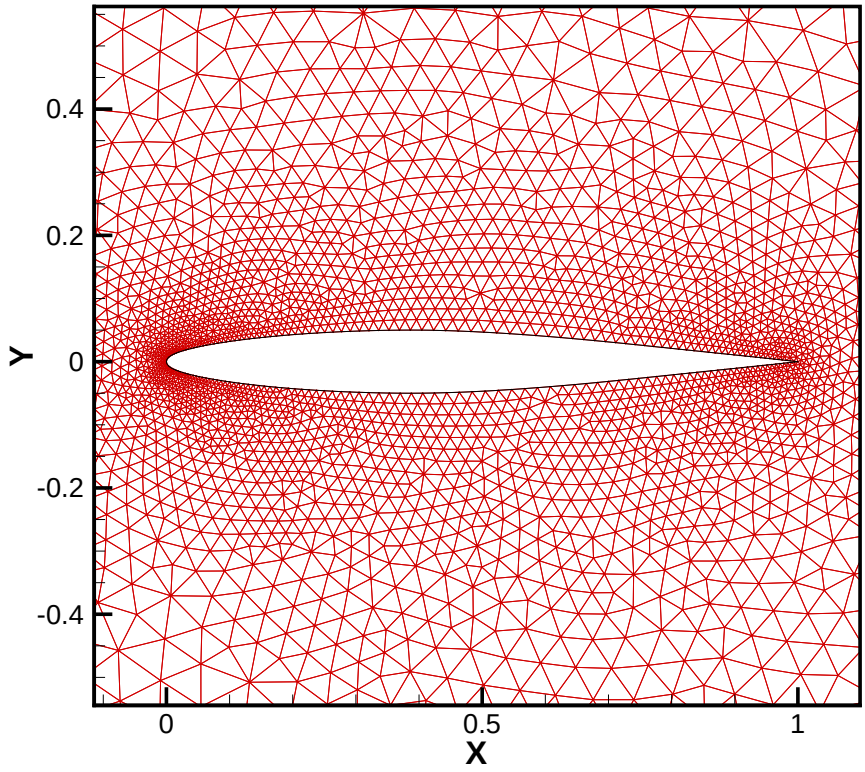
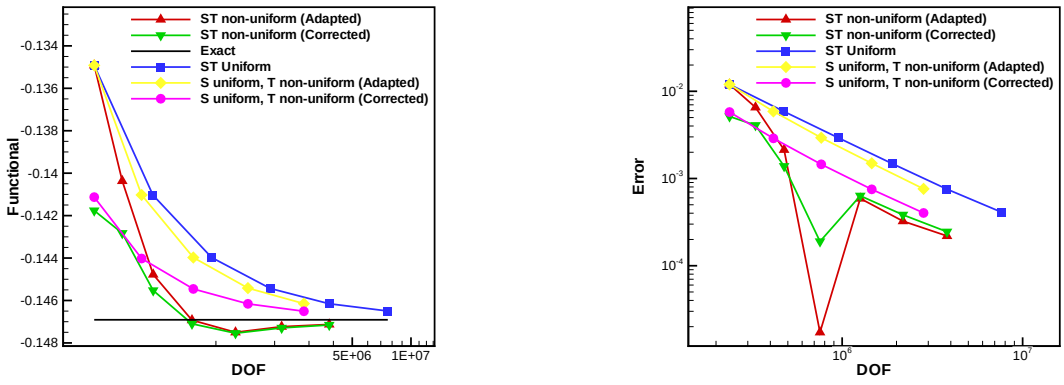


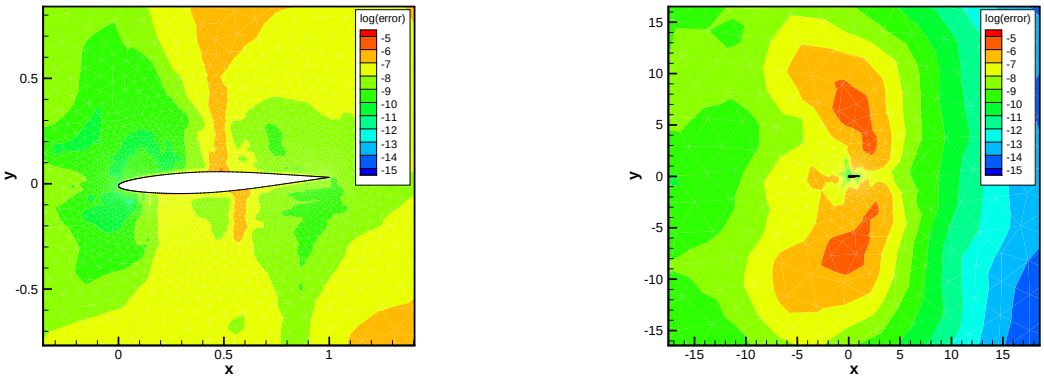
Figure 12. Computational mesh consisting of 6601 elements for the transonic pitching airfoil example.



(a) Functional convergence

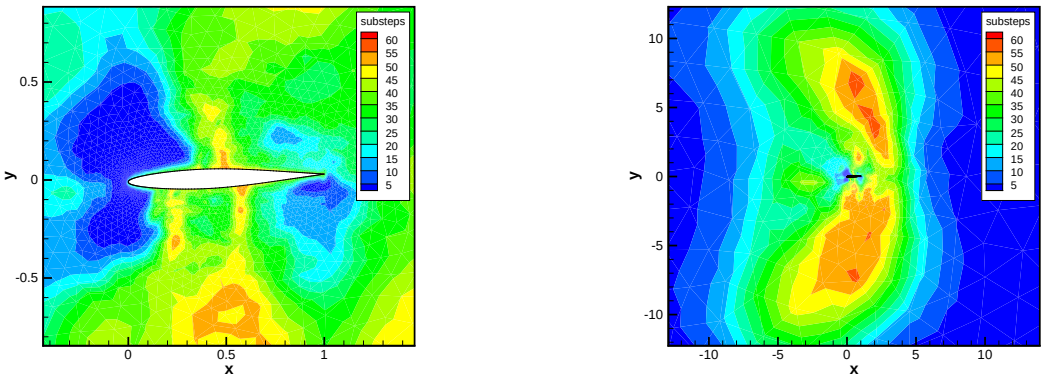
(b) Functional error convergence

Figure 13. Comparison of convergence with respect to total degrees-of-freedom using various adaptation methods for the transonic pitching airfoil example.



(a) Zoomed in view around the airfoil (b) Farfield

Figure 14. Distribution of functional relevant error at the first adaptation cycle near the center of the pitch cycle (top of slab 9) for the transonic pitching airfoil example.



(a) Zoomed in view around the airfoil (b) Farfield

Figure 15. Number of time-steps per spatial element within slab 9 (near center of pitch cycle) for the transonic pitching airfoil example.

[17]van der Vegt, J. J. W. and van der Ven, H., "Space–time discontinuous Galerkin finite element method with dynamic grid motion for inviscid compressible flows: I. general formulation," *Journal of Computational Physics*, Vol. 182-2, 2002, pp. 546–585.

[18]Abedi, R., Chung, S.-H., Erickson, J., Fan, Y., Garland, M., Guoy, D., Haber, R., Sullivan, J. M., Thite, S., and Zhou, Y., "Spacetime meshing with adaptive refinement and coarsening," *Annual Symposium on Computational Geometry, Proceedings of the twentieth annual symposium on Computational geometry Brooklyn, New York, USA*, 2004.

[19]Barth, T. J., "Space-Time Error Representation and Estimation in Navier-Stokes Calculations," *Lecture Notes in Computational Science and Engineering*, Vol. 56, 2007.

[20]Mavriplis, D. J., "Unstructured-Mesh Discretizations and Solvers for Computational Aerodynamics," *AIAA Journal*, Vol. 46-6, June 2008, pp. 1281–1298.

[21]Batina, J. T., "Unsteady Euler Airfoil Solutions Using Unstructured Dynamic Meshes," *AIAA Journal*, Vol. 28-8, August 1990, pp. 1381–1388.

[22]Mavriplis, D. J., "Multigrid Solution of the Discrete Adjoint for Optimization Problems on Unstructured Meshes," *AIAA Journal*, Vol. 44-1, January 2006, pp. 42–50.

[23]Mani, K. and Mavriplis, D. J., "Unsteady Discrete Adjoint Formulation for Two-Dimensional Flow Problems with Deforming Meshes," *AIAA Journal*, Vol. 46-6, June 2008, pp. 1351–1364.

[24]Wang, L. and Mavriplis, D. J., "Implicit solution of the unsteady Euler equations for high-order accurate discontinuous Galerkin discretizations," *Journal of Computational Physics*, Vol. 225-2, 2007, pp. 1994–2015.

[25]Nastase, C. R. and Mavriplis, D. J., "On the Geometric Conservation Law for High-Order Discontinuous Galerkin Discretizations on Dynamically Deforming Meshes," *46th Aerospace Sciences Meeting and Exhibit, Reno NV*, 2008, AIAA Paper 2008–778.