51st AIAA Aerospace Sciences Meeting including the New Horizons Forum and Aerospace Exposition
07 - 10 January 2013, Grapevine (Dallas/Ft. Worth Region), Texas

AIAA 2013-0520

Downloaded by Dimitri Mavriplis on January 15, 2013 | http://arc.aiaa.org | DOI: 10.2514/6.2013-520

Efficient Adaptation using Discrete Adjoint Error Estimates in Unsteady Flow Problems

Bryan T. Flynt[*] and Dimitri J. Mavriplis[†]

Department of Mechanical Engineering, University of Wyoming, Laramie WY, 82071, USA

This paper presents a framework for the reduction of the spatial, temporal and algebraic error by use of the discrete adjoint solution in a time integrated functional (i.e. Lift, Drag, etc.) for problems of aerodynamic interest. The three types of error are separated into distinct contributions that are used to selectively adapt the discretization to reduce the error in the functional in the most efficient manner. A simple unsteady problem of a vortex convecting through a time integrated box is used to demonstrate that the frequency of refinement has a large impact on the overall computational efficiency of the solution. Analysis of the simplified problem shows that adaption of one parameter in isolation results in a limited improvement in accuracy and a combination of adapting on the three types of error is necessary to get any significant improvement. A method is developed to address this restriction and tested. The improved efficiency of the new method is demonstrated on the simplified test problem and an additional test case of a vortex convecting past an airfoil.

I. Introduction

Achieving satisfactory results with computational fluid dynamics has traditionally been a balancing act between available computational resources and the desired accuracy the practitioner hopes to achieve. This is especially true for time accurate simulations where the number of time steps, step size, grid resolution and convergence criteria all must be chosen to accurately capture the flow physics of interest without resulting in excessive computational cost. An experienced practitioner uses past results to chose these parameters expecting similar results but has no way to directly evaluate the error resulting from their choices. Recently, adjoint based adaptive methods have been gaining popularity to provide answers to this problem for both steady and unsteady problems.

An adjoint variable based approach permits the estimation of error relevant to a functional, and the corresponding distribution of this error in the time and spatial domains. The method relies on applying discrete adjoint equations on a Taylor series expansion of a functional, where a linear approximation of the functional between two different resolutions (space or time) is estimated by solving the flow problem and the adjoint problem at the coarse resolution and then projecting it to the finer resolution. The method only predicts the error in the flow solution that is relevant to the functional of interest between two different resolutions and not against the exact analytical value of the functional. Adjoint based approaches also permit distinct identification of the contributions to this error arising from the three primary sources, namely the spatial/temporal resolution and the effect of partial convergence of the equations. This is different from common adaption or control schemes which are based on algebraic estimates of the local error within the solution, in that the global error in the functional can be estimated and adapted on.

This method is most mature for steady problems where researchers have used the error estimates to adaptively refine the grid[1–3] and put error bounds on solution accuracy.[4–6] Unsteady problems have received considerably less attention and the primary focus has been on performing adaption in time,[7–9] space[9,10] and convergence tolerances[7,8] individually. In our previous paper,[11] we combined all three types of error estimates into the same framework and presented results showing each of the three components can be separated into

[*] Ph.D. Candidate, AIAA Student Member, email: bflynt@uwyo.edu
[†] Professor, AIAA Associate Fellow, email: mavripl@uwyo.edu

Copyright © 2013 by Bryan T. Flynt. Published by the American Institute of Aeronautics and Astronautics, Inc., with permission.

individual contributions. Furthermore, we demonstrated the three individual error estimates could be used to adapt the discretization and improve the efficiency of the time accurate solution.

The present study builds on our previous work[11] by eliminating the requirement that the convergence tolerance and spatial discretization remain constant for all time steps of the time accurate simulation. Using this added capability we explore the effect of the refinement interval and amount of error to refine on any one interval has on the efficiency of the error reduction. In addition, we are able to demonstrate that adaption of only one error component in isolation will result in a limited improvement in accuracy and a combination of adapting on the three types of error is necessary to get a significant improvement. The methodology is demonstrated using an implicit cell-centered finite volume solver with gradient reconstruction and a Roe flux splitting scheme[12] to solve the Euler equations on an unstructured mesh up to 2nd order accuracy in both space and time. The following sections contain the formulations required for calculation of each of the error estimates, implementation details and test case results showing the benefits of the proposed adaption strategy using the adjoint based error estimates.

II. Governing Equations

II.A. Euler Equations

Downloaded by Dimitri Mavriplis on January 15, 2013 | http://arc.aiaa.org | DOI: 10.2514/6.2013-520

The Euler equations are the governing equations for inviscid flow where the dissipative phenomena of viscosity and thermal conductivity are ignored. The three-dimensional equations in Cartesian coordinates for a compressible flow minus any source terms are represented by a system of five non-linear partial differential equations.

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \vec{V}) = 0 \tag{1}$$

$$\frac{\partial (\rho u)}{\partial t} + \nabla \cdot (\rho u \vec{V}) = -\frac{\partial P}{\partial x} \tag{2}$$

$$\frac{\partial (\rho v)}{\partial t} + \nabla \cdot (\rho v \vec{V}) = -\frac{\partial P}{\partial y} \tag{3}$$

$$\frac{\partial (\rho w)}{\partial t} + \nabla \cdot (\rho w \vec{V}) = -\frac{\partial P}{\partial z} \tag{4}$$

$$\frac{\partial}{\partial t}\left[\rho\left(e + \frac{V^2}{2}\right)\right] + \nabla \cdot \left[\rho\left(e + \frac{V^2}{2}\right)\vec{V}\right] = -\frac{\partial uP}{\partial x} - \frac{\partial vP}{\partial y} - \frac{\partial wP}{\partial z} + \rho \vec{f} \cdot \vec{V} \tag{5}$$

By examining the equations we see there are five equations but six unknowns (ρ, u, v, w, e, P). Therefore, in this paper an ideal gas is assumed and the equation of state relates the pressure to total energy by:

$$P = (\gamma - 1)\left[E - \frac{1}{2}\rho\left(u^2 + v^2 + w^2\right)\right] \tag{6}$$

To facilitate solution by computer the equations are manipulated into vector form by re-writing the system as:

$$\frac{\partial \vec{U}}{\partial t} + \frac{\partial \vec{F}_x}{\partial x} + \frac{\partial \vec{F}_y}{\partial y} + \frac{\partial \vec{F}_z}{\partial z} = 0 \tag{7}$$

where the vector of conserved quantities becomes:

$$\vec{U} = \begin{pmatrix} \rho \\ \rho u \\ \rho v \\ \rho w \\ \rho E \end{pmatrix} \tag{8}$$

and the Cartesian inviscid flux vectors reduced to:

$$\vec{F_x} = \begin{pmatrix} \rho u \\ \rho u^2 + P \\ \rho uv \\ \rho uw \\ \rho u \left(E + \frac{P}{\rho}\right) \end{pmatrix} \quad \vec{F_y} = \begin{pmatrix} \rho v \\ \rho vu \\ \rho v^2 + P \\ \rho vw \\ \rho v \left(E + \frac{P}{\rho}\right) \end{pmatrix} \quad \vec{F_z} = \begin{pmatrix} \rho w \\ \rho wu \\ \rho wv \\ \rho w^2 + P \\ \rho w \left(E + \frac{P}{\rho}\right) \end{pmatrix} \tag{9}$$

The Euler equations (Eqn. 7) are solved implicitly in time using the second order backwards difference formula (BDF2) with second order accurate spatial derivatives on an unstructured grid. This is done for each time step by introducing an implicit residual as

$$\vec{R^n}(\vec{U}^n, \vec{U}^{n-1}, \vec{U}^{n-2}, \vec{x}) = 0 \tag{10}$$

where $\vec{U}^n$ is the solution at the current time step, $\vec{U}^{n-1}$ and $\vec{U}^{n-2}$ are the solutions at the two previous time steps and $\vec{x}$ is the vector of grid coordinates. This system is linearized with respect to the unknown solution $\vec{U}^n$ and solved using Newton's method.

$$\begin{gathered} \left[\tfrac{\partial \vec{R}(\vec{U}^k, \vec{x})}{\partial \vec{U}^k}\right] \delta\vec{U}^k = -\vec{R}(\vec{U}^k, \vec{x}) \\ \vec{U}^{k+1} = \vec{U}^k + \delta\vec{U}^k \\ \vec{U}^n = \vec{U}^{k+1}, \delta\vec{U}^k \to 0 \end{gathered} \tag{11}$$

II.B. Error due to Spatial Resolution

Consider an objective function L that is a quantity dependent on the flow solution U for all time steps of the simulation. For this case, the spatial error will be derived by using a Taylor series expansion of the true objective calculated on a fine grid $L_s(\vec{U}_s)$.

$$L_s(\vec{U}_s) = L_s(\tilde{U}_s) + \frac{\partial L_s(\tilde{U}_s)}{\partial \tilde{U}_s}\left(\vec{U}_s - \tilde{U}_s\right) + \cdots \tag{12}$$

where $L_s(\tilde{U}_s)$ is the fine grid objective computed using an approximate solution $\tilde{U}_s$. In general, L_s may be a time integrated objective and will depend on the entire time history of the solution $\vec{U}_s$. Therefore no time level superscripts exist with the understanding that unsuperscripted variables refer to the entire time history of the flow field.

To avoid the need for computing the true solution $\vec{U}_s$ on the fine mesh in the proceeding equation we will use a Taylor series expansion about the fine level residual equation

$$R_s(\vec{U}_s) = 0 = R_s(\tilde{U}_s) + \frac{\partial R_s(\tilde{U}_s)}{\partial \tilde{U}_s}\left(\vec{U}_s - \tilde{U}_s\right) + \cdots \tag{13}$$

where the unsuperscripted residual denotes the residuals over all time steps and all mesh cells of the simulation. Using this expression to substitute for the term $\left(\vec{U}_s - \tilde{U}_s\right)$ we can rewrite the objective as:

$$L_s(\vec{U}_s) \cong L_s(\tilde{U}_s) - \left[\frac{\partial L_s(\tilde{U}_s)}{\partial \tilde{U}_s}\right]\left[\frac{\partial R_s(\tilde{U}_s)}{\partial \tilde{U}_s}\right]^{-1} R_s(\tilde{U}_s) \tag{14}$$

Denoting the matrix product as the adjoint variable ${\Lambda_s}^T$

$${\Lambda_s}^T = -\left[\frac{\partial L_s(\tilde{U}_s)}{\partial \tilde{U}_s}\right]\left[\frac{\partial R_s(\tilde{U}_s)}{\partial \tilde{U}_s}\right]^{-1} \tag{15}$$

Downloaded by Dimitri Mavriplis on January 15, 2013 | http://arc.aiaa.org | DOI: 10.2514/6.2013-520

Downloaded by Dimitri Mavriplis on January 15, 2013 | http://arc.aiaa.org | DOI: 10.2514/6.2013-520

leads to the following expression for the linear approximation to the fine grid objective.

$$L_s(\vec{U}_s) \cong L_s(\tilde{U}_s) + {\Lambda_s}^T R_s(\tilde{U}_s) \tag{16}$$

The inner product given by the last term is taken over all mesh cells and all time steps of the simulation. The values in this expansion are obtained by constructing the approximate solution $\tilde{U}_s$ by interpolating a coarse grid solution onto the fine grid at each time step and evaluating $L_s(\tilde{U}_s)$ and $R_s(\tilde{U}_s)$. The difficulty lies in obtaining the adjoint value Λ_s^T. A direct calculation of this quantity costs as much as obtaining the true solution $\vec{U}_s$, in which case, we might as well evaluate $L_s(\vec{U}_s)$ directly. To alleviate this expense we approximate the fine grid adjoint value by calculating the coarse grid adjoint denoted as Λ and interpolating onto the fine grid using a projection matrix (I_S^s) to arrive at the final linear approximation to the fine grid objective function.

$$\begin{gathered}\tilde{\Lambda}_s^T = I_S^s \Lambda^T \\ \\ L_s(\vec{U}_s) \cong L_s(\tilde{U}_s) + \tilde{\Lambda}_s^T R_s(\tilde{U}_s)\end{gathered} \tag{17}$$

$$\underbrace{L_s(\vec{U}_s) - L_s(\tilde{U}_s)}_{\varepsilon_s} \cong I_S^s \Lambda^T R_s(\tilde{U}_s) \tag{18}$$

From this derivation we see that the term $\tilde{\Lambda}_s^T R_s(\tilde{U}_s)$ is a direct estimation of the linear error in our calculation from the lack of spatial resolution. Therefore, the elements with the largest value can be flagged for future refinement.

One term of this derivation, that cannot be directly evaluated and has yet to be solved for, is the adjoint variable on the coarse grid Λ^T. It is most easily solved by rearranging the matrix product expression in equation (15) to be:

$$\left[\frac{\partial R(\vec{U})}{\partial \vec{U}}\right]^T \vec{\Lambda} = -\left[\frac{\partial L(\vec{U})}{\partial \vec{U}}\right]^T \tag{19}$$

Notice the approximate solutions $\tilde{U}$ have been replaced with converged solutions $\vec{U}$ since this equation is solved using the coarse grid solution. The matrices in equation (19) represent each equation of every element on every time step therefore we can expand the equations into an expression where each row represents the solution to a single time step (shown here for BDF2).

$$\left[\begin{array}{cccccc} \left[\frac{\partial R(\vec{U})}{\partial \vec{U}}\right] & & & & & \\ \left[\frac{\partial R(\vec{U})}{\partial \vec{U}}\right] & \left[\frac{\partial R(\tilde{U})}{\partial \tilde{U}}\right] & & 0 & & \\ & \left[\frac{\partial R(\vec{U})}{\partial \vec{U}}\right] & \left[\frac{\partial R(\vec{U})}{\partial \vec{U}}\right] & & & \\ & & \ddots & \ddots & & \\ 0 & & & \left[\frac{\partial R(\vec{U})}{\partial \vec{U}}\right] & \left[\frac{\partial R(\vec{U})}{\partial \vec{U}}\right] & \\ & & & & \left[\frac{\partial R(\vec{U})}{\partial \vec{U}}\right] & \left[\frac{\partial R(\vec{U})}{\partial \vec{U}}\right] \end{array}\right]^T \left[\begin{array}{c} \Lambda^1 \\ \Lambda^2 \\ \Lambda^3 \\ \vdots \\ \Lambda^{n-1}\Lambda^n \end{array}\right] = -\left[\begin{array}{c} \left[\frac{\partial L(\vec{U})}{\partial \vec{U}}\right] \\ \left[\frac{\partial L(\vec{U})}{\partial \vec{U}}\right] \\ \left[\frac{\partial L(\vec{U})}{\partial \vec{U}}\right] \\ \vdots \\ \left[\frac{\partial L(\vec{U})}{\partial \vec{U}}\right] \\ \left[\frac{\partial L(\vec{U})}{\partial \vec{U}}\right] \end{array}\right]^T \tag{20}$$

By carrying out the transpose operator the system of equations that must be solved to acquire the adjoint solution becomes evident.

$$\left[\begin{array}{cccccc} \left[\frac{\partial R(\vec{U})}{\partial \vec{U}}\right]^T & \left[\frac{\partial R(\vec{U})}{\partial \vec{U}}\right]^T & & & & \\ & \left[\frac{\partial R(\tilde{U})}{\partial \tilde{U}}\right]^T & \left[\frac{\partial R(\vec{U})}{\partial \vec{U}}\right]^T & & 0 & \\ & & \ddots & \ddots & & \\ & & & \left[\frac{\partial R(\vec{U})}{\partial \vec{U}}\right]^T & \left[\frac{\partial R(\vec{U})}{\partial \vec{U}}\right]^T & \\ & 0 & & & \left[\frac{\partial R(\vec{U})}{\partial \vec{U}}\right]^T & \left[\frac{\partial R(\vec{U})}{\partial \vec{U}}\right]^T \\ & & & & & \left[\frac{\partial R(\vec{U})}{\partial \vec{U}}\right]^T \end{array}\right] \left[\begin{array}{c} \Lambda^1 \\ \Lambda^2 \\ \Lambda^3 \\ \vdots \\ \Lambda^{n-1} \\ \Lambda^n \end{array}\right] = -\left[\begin{array}{c} \left[\frac{\partial L(\vec{U})}{\partial \vec{U}}\right] \\ \left[\frac{\partial L(\vec{U})}{\partial \vec{U}}\right] \\ \left[\frac{\partial L(\vec{U})}{\partial \vec{U}}\right] \\ \vdots \\ \left[\frac{\partial L(\vec{U})}{\partial \vec{U}}\right] \\ \left[\frac{\partial L(\vec{U})}{\partial \vec{U}}\right] \end{array}\right]^T \tag{21}$$

Downloaded by Dimitri Mavriplis on January 15, 2013 | http://arc.aiaa.org | DOI: 10.2514/6.2013-520

From this transposed system of equation it is evident that the unsteady adjoint variables can be solved by sweeping backwards through all time steps of the solution solving for each adjoint solution using the flow and adjoint solutions from one step in the future of the time series.

II.C. Error due to Temporal Resolution

The use of the adjoint variable to identify errors in temporal resolution follows a similar path to that of spatial error (Section II.B). We start by using a Taylor series expansion of the true time integrated objective calculated using a small time step $L_t(\vec{U}_t)$

$$L_t(\vec{U}_t) = L_t(\tilde{U}_t) + \frac{\partial L_t(\tilde{U}_t)}{\partial \tilde{U}_t}\left(\vec{U}_t - \tilde{U}_t\right) + \cdots \tag{22}$$

where $L_t(\tilde{U}_t)$ is the small time step objective calculated with an approximate solution. The need to calculate the true small time-step solution $\vec{U}_t$ is eliminated by using a Taylor series expansion of the small time step residual equation:

$$R_t(\vec{U}_t) = 0 = R_t(\tilde{U}_t) + \frac{\partial R_t(\tilde{U}_t)}{\partial \tilde{U}_t}\left(\vec{U}_t - \tilde{U}_t\right) + \cdots \tag{23}$$

Following the steps previously presented we can arrive at the expression for the small time step objective using an approximate small time step solution $\tilde{U}_t$ obtained by projecting a coarse time step solution to the finer time step space:

$$L_t(\vec{U}_t) \cong L_t(\tilde{U}_t) + {\Lambda_t}^T R_t(\tilde{U}_t) \tag{24}$$

The requirement for the small time step adjoint solution (Λ_t) is relaxed by approximating it using the adjoint solution from a coarse time step solution. This results in an equation that is a linear approximation to the small time step objective using only solutions from the coarse time step simulation.

$$L_t(\vec{U}_t) \cong L_t(\tilde{U}_t) + \tilde{\Lambda}_t^T R_t(\tilde{U}_t) \tag{25}$$

Again we can re-arrange the terms to arrive at a direct estimation of the linear error in our calculation, this time from a lack of resolution in the time domain.

$$\underbrace{L_t(\vec{U}_t) - L_t(\tilde{U}_t)}_{\varepsilon_t} \cong \tilde{\Lambda}_t^T R_t(\tilde{U}_t) \tag{26}$$

II.D. Error due to Partial Convergence

In the former sections we assumed the solution on the coarse space and time domain was obtained by full convergence of the flow equations at each implicit time step resulting in a coarse level residual equation that evaluates to zero. If the flow equations were only partially converged at each time step, the fully converged objective can be linearly approximated using the the Taylor series expansion:

$$L(\vec{U}) = L(\tilde{U}_c) + \frac{\partial L(\tilde{U}_c)}{\partial \tilde{U}_c}\left(\vec{U} - \tilde{U}_c\right) + \cdots \tag{27}$$

where the objective $L(\tilde{U}_c)$ is obtained through partially converging the flow equations to an approximate solution $\tilde{U}_c$. Again, we use the expansion of the residual equation to eliminate the need for the fully converged solution $\vec{U}$ and introduce an approximate adjoint variable into the equation.

$$L(\vec{U}) \cong L(\tilde{U}_c) + \tilde{\Lambda}_c^T R(\tilde{U}_c) \tag{28}$$

The approximate adjoint $\tilde{\Lambda}_c$ is obtained by partially converging the adjoint equations using the same mesh and time step distribution as for the analysis problem. The contribution to the integrated linear error (ε_c) for the partially converged solution can be obtained by re-arranging the equation.

$$\underbrace{L(\vec{U}) - L(\tilde{U}_c)}_{\varepsilon_c} \cong \tilde{\Lambda}_c^T R(\tilde{U}_c) \tag{29}$$

It is of interest to note the residual $R(\tilde{U}_c)$ is non-zero only because the flow equations are not fully converged. If the system is fully converged, the residual is zero and the approximate objective is no longer approximate and becomes the true objective.

II.E. Combined Error

Our goal is to estimate the total error in a time dependent simulation objective and to determine the respective contributions to the total error from spatial discretization, temporal discretization, and incomplete convergence in order that adaptive refinement methods may be used to efficiently reduce the error. Using a Taylor series expansion, we can estimate the exact objective L_{st} computed on a fine time step and fine mesh with full convergence using an approximate solution $\tilde{U}_{cst}$ computed on a coarser mesh with a larger time step with partial convergence as:

$$L_{st}(\vec{U}_{cst}) = L_{st}(\tilde{U}_{cst}) + \frac{\partial L_{st}(\tilde{U}_{cst})}{\partial \tilde{U}_{cst}} \left(\vec{U}_{cst} - \tilde{U}_{cst}\right) + ... \tag{30}$$

In the above expression, $\tilde{U}_{cst}$ is obtained from the partially converged approximate solution $\tilde{U}_c$ computed on a coarser grid with a larger time step, projected onto the finer mesh and time step space. Although the above equation could be used to obtain an expression for the total error, it does not provide a mechanism for separating out the various error components. However, since the adjoint error estimation procedure relies on a linearization, we can assume these different error components are additive. Therefore, the error in the solution may be written as:

$$\vec{U}_{cst} - \tilde{U}_{cst} = \vec{U}_s - \tilde{U}_s + \vec{U}_t - \tilde{U}_t + \vec{U} - \tilde{U}_c \tag{31}$$

Here $\vec{U}_s - \tilde{U}_s$ corresponds to the difference between the solution computed on the fine grid and the approximate coarse level solution $\tilde{U}_c$ projected onto the fine grid with a fixed time step value. Similarly, $\vec{U}_t - \tilde{U}_t$ corresponds to the change in the solution when the time step is refined, with all other simulation parameters (mesh size, convergence tolerance) remaining fixed, and $\vec{U} - \tilde{U}_c$ corresponds to the change in the coarse mesh and time step solution that would be observed if full convergence was enforced at each time step. Clearly, interactions between these various sources of error will exist in actual simulations, but these will be higher order non-linear error interactions that cannot be accounted for in an adjoint formulation. Furthermore, as the errors are reduced, the additive assumption will become asymptotically more exact. Inserting the above expression into equation (33), and making use of an adjoint solution computed on the coarse space and time discretization level at the partially converged state leads to the following expression for the objective error:

$$L_{st}(\vec{U}_{cst}) \cong L_{st}(\tilde{U}_{cst}) + \tilde{\Lambda}_s^T R_s(\tilde{U}_s) + \tilde{\Lambda_t}^T R_t(\tilde{U}_t) + \tilde{\Lambda}_c^T R(\tilde{U}_c) \tag{32}$$

where the adjoint variables $\tilde{\Lambda}_s$, and $\tilde{\Lambda}_t$ correspond to the approximate coarse level adjoint $\tilde{\Lambda}_c$ projected onto the fine mesh and fine time step domain, respectively. The various residual operators R, R_s and R_t correspond to the space-time residuals evaluated on the coarse mesh and time step domain, the fine mesh domain (with coarse time step), and the fine time step domain (with coarse mesh), respectively. The one remaining term that is not readily available is the fine time and space approximate objective value $L_{st}(\tilde{U}_{cst})$. Instead of separately constructing this term it is formed using a linear combination of the terms already used for each individual error correction. To do this we let the variables $(\delta s, \delta t)$ represent the difference between the individual fine domain objective values calculated using approximate solutions and the coarse approximate objective value $L(U_c)$ already calculated.

$$\begin{aligned} \delta s &= L(U_c) - L_s(\tilde{U}_s) \\ \delta t &= L(U_c) - L_t(\tilde{U}_t) \end{aligned} \tag{33}$$

Using these two values we can linearly predict what the approximation would be for the combined approximation $L_{st}(\tilde{U}_{cst})$:

$$L_{st}(\tilde{U}_{cst}) = L(U_c) - \delta s - \delta t \tag{34}$$

Inserting this approximation into the combined linear approximation from earlier (Eqn. 32) we get the final form of the combined approximation.

$$L_{st}(\vec{U}_{cst}) \cong L_s(\tilde{U}_s) + L_t(\tilde{U}_t) - L(U_c) + \tilde{\Lambda}_s^T R_s(\tilde{U}_s) + \tilde{\Lambda}_t^T R_t(\tilde{U}_t) + \tilde{\Lambda}_c^T R(\tilde{U}_c) \tag{35}$$

By re-arranging we can isolate the terms to again produce an estimate of integrated linear error

$$\underbrace{L_{st}(\vec{U}_{cst}) - L_s(\tilde{U}_s) - L_t(\tilde{U}_t) + L(U_c)}_{\varepsilon_{cst}} \cong +\tilde{\Lambda}_s^T R_s(\tilde{U}_s) + \tilde{\Lambda}_t^T R_t(\tilde{U}_t) + \tilde{\Lambda}_c^T R(\tilde{U}_c) \tag{36}$$

Downloaded by Dimitri Mavriplis on January 15, 2013 | http://arc.aiaa.org | DOI: 10.2514/6.2013-520

III. Implementation Details

III.A. Mesh Refinement

In the current work the conformal nature of a mixed element grid is maintained by using element subdivision for refinement and un-refinement.[13] This is accomplished using a set of allowable subdivision types for each element as shown in Figure 1. An element that is flagged for refinement is isotropically split into 4 elements. The non-flagged neighboring element is partially subdivided using an allowable pattern to keep the grid conformal. On subsequent steps, if one of the elements resulting from a partial subdivision is flagged for refinement, the partial element is collapsed back into the parent element which is then isotropically split into 4 smaller elements. This method of refinement limits the formation of sliver elements within the grid and lends itself to a simple tree data structure which can be unrolled to un-refine elements within the grid. Furthermore, no smoothing of the adaptive grid is performed so that un-refinement of elements will result in the starting grid.

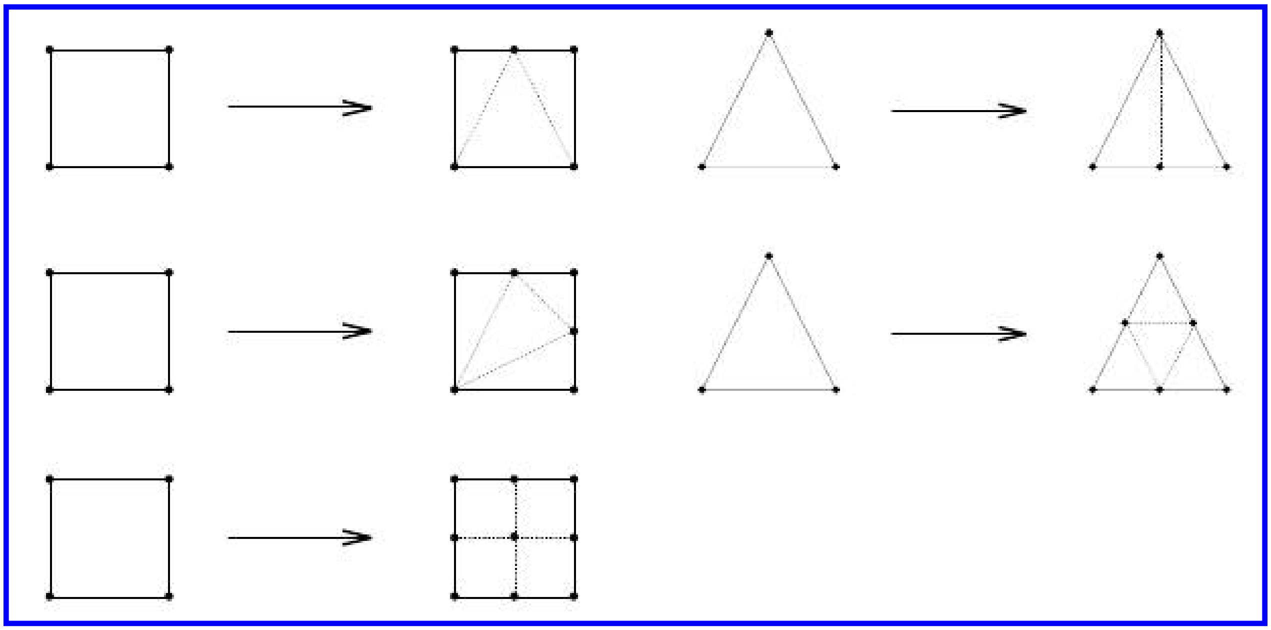

Figure 1. Allowable subdivision patterns for triangle and quadrilateral cells

Downloaded by Dimitri Mavriplis on January 15, 2013 | http://arc.aiaa.org | DOI: 10.2514/6.2013-520

III.B. Spatial Projection

To project the solutions to grids of different spatial refinement levels a general interpolation scheme is used based on compact radial basis functions.[14] Specifically, the compact functions of Wendland[15] are used with an additional polynomial constraint to ensure 2nd order accuracy for all cases. The approximations have the general form

$$s(\vec{x}) = \sum_{j=1}^{N} \alpha_j \phi\left(\vec{x} - \vec{x_j}\right) + p(\vec{x}) \tag{37}$$

where α_j is the weight of the known value at element j, $\phi\left(\vec{x} - \vec{x_j}\right)$ is the radial function evaluated between the two centers and $p(\vec{x})$ is a polynomial of the required order of accuracy. The required number of nearest neighbors is found using an alternating digital tree[16] followed by a linear search over the remaining candidates.

III.C. Temporal Projection

For the temporal refinement a 2-to-1 refinement pattern is enforced where a single time interval is split into 2 equal intervals. In addition, any adjacent time intervals that vary more than 2 times the size of the current time interval will be flagged for splitting to ensure no large jumps in the time step size exist.

To perform the temporal projections from coarse to fine time steps linear interpolation is used to arrive at the intermediate solution.

$$\begin{aligned} \tilde{\Lambda}_t^{n-\frac{1}{2}} &= \tfrac{1}{2}\left(\Lambda_T^n + \Lambda_T^{n-1}\right) \\ \tilde{\vec{U}}_t^{n-\frac{1}{2}} &= \tfrac{1}{2}\left(\vec{U}_T^n + \vec{U}_T^{n-1}\right) \end{aligned} \tag{38}$$

III.D. Solver Details

The adaptive solver consists of three parts namely the forward flow solver, backward adjoint solver and a time/grid adaption module. The projection operations are done within the adaption module while the error computations are built into the adjoint solver. The simulation starts by using the adaption module to write out all grids for all time steps to file for the current (first) sweep. The flow solver then performs the forward integration in time by reading in the grid for the new time step and projecting the previous solution to it for the initial condition and time history information. Once the flow solution has been solved on all time steps, the adjoint solver reads in the flow solution and associated grid and performs the backwards sweep in time to compute the adjoint variables at each time step and saves them to file. Next the adaption module will create an isotropically refined grid for each of the fine domain time steps and perform the necessary interpolation onto each. These are saved to disk so that the flow solver followed by the adjoint solver can read them in and calculate the fine grid approximate objective solution, evaluate the residual and form the contribution to the error for each grid element of each time step and save them to file. In addition, smoothing of the approximate flow or adjoint solution can be done by their respective solver during this part of the process. For this research only the spatially projected approximate adjoint was smoothed by one order of magnitude in the L_2 norm for each time step. Finally, the individual error contributions are read from disk one time step at a time and the error is used to flag time intervals and grid elements for refinement based on the desired refinement criteria selected. If another more refined simulation is desired the adaption module will again generate and save out a newly refined grid for each adapted time step and the process will repeat again.

Downloaded by Dimitri Mavriplis on January 15, 2013 | http://arc.aiaa.org | DOI: 10.2514/6.2013-520

IV. Adaption Strategy and Threshold Selection

IV.A. Model Test Problem

The test case used to develop the algorithms in this paper involves the time integrated density inside a region of the flow field as a 2-D inviscid isentropic vortex[17,18] convects through uniform flow. In this case the mean flow density (ρ_∞), velocity (u_∞,v_∞), pressure (P_∞), and temperature (T_∞) are taken to be free stream values. We set these flow parameters to be (ρ_∞,u_∞,u_∞,P_∞,T_∞) = (1.0,0.5,0.0,1.0,1.0) with characteristic boundary conditions[19] on all free stream boundaries. At the initial time the flow is perturbed by an isentropic vortex $(\delta u, \delta v, \delta T)$ centered at $(x_o, y_o) = (-15.0, 0.0)$ with the form:

$$\begin{aligned} \delta u &= -\tfrac{V_m}{R_c}\left(y - y_o\right) e^{\left[\frac{1}{2} - \left(\frac{r}{2R_c}\right)^2\right]} \\ \delta v &= \tfrac{V_m}{R_c}\left(x - x_o\right) e^{\left[\frac{1}{2} - \left(\frac{r}{2R_c}\right)^2\right]} \\ \delta T &= -\tfrac{V_m^2(\gamma-1)}{2\gamma R_c^2} e^{\left[1 - \left(\frac{r}{R_c}\right)^2\right]} \end{aligned} \tag{39}$$

where V_m is the maximum perturbed velocity, R_c is the distance r that this maximum velocity occurs at from the vortex center and $\gamma = 1.4$ is the ratio of specific heats of the fluid. From the relationship for an ideal gas and assuming isentropic flow the density is found for every point in the domain as:

$$\rho = T^{1/(\gamma-1)} = \left(T_\infty + \delta T\right)^{1/(\gamma-1)} \tag{40}$$

In this case the strength of the vortex and size were specified by assigning the values of 0.2 to the maximum velocity ($V_m = 0.2$) at a core radius of 0.5 ($R_c = 0.5$) grid units.

To properly validate the results a single quantity is needed as the objective function and for this we define the objective (Eqn. 41) to be the time and space integrated density inside a square box centered about the origin $(x, y) = (0, 0)$ with equal length sides of 2 grid units. The vortex is convected for 60 non-dimensional time units allowing it to travel from the starting position of $(x_o, y_o) = (-15.0, 0.0)$ to the final position of $(x, y) = (15.0, 0.0)$ and being centered in the integrated region exactly half way through the computation. A portion of the coarsest grid with the prescribed initial condition and highlighted integration region is shown in Figure 2. The exact analytic solution to this problem was found using a common computer math program to be 239.52558800471 which we will compare against as the true analytic solution in subsequent analysis.

$$L(U) = \int_0^{60} \int_{-1}^{1} \int_{-1}^{1} \rho \; \mathrm{d}x\mathrm{d}y\mathrm{d}t \tag{41}$$

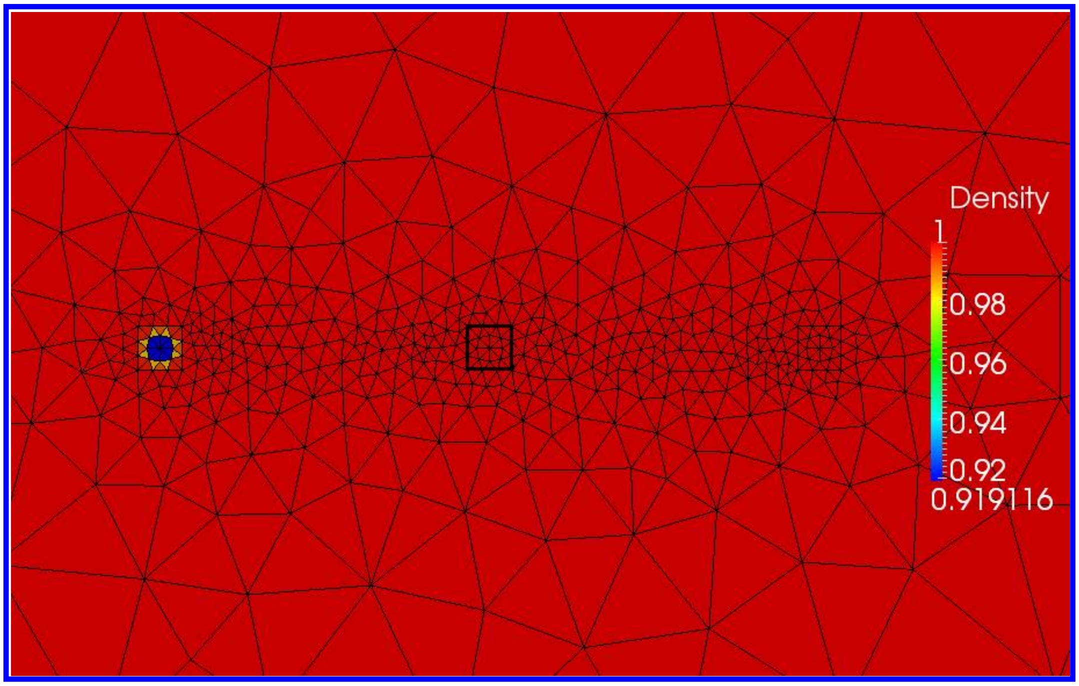

Figure 2. Initial density distribution with highlighted integrated region for grid of 1176 elements

Downloaded by Dimitri Mavriplis on January 15, 2013 | http://arc.aiaa.org | DOI: 10.2514/6.2013-520

IV.B. Global Adaption Strategy

In our earlier work[11] we examined the prediction and convergence properties of using the adjoint to approximate the spatial, temporal and convergence errors. In doing this we had to carefully select the resolution of the discretization not being refined upon to avoid influencing the properties being measured. For example, measuring the convergence properties of mesh refinement while using a time step size or convergence tolerance that produced an error greater than the spatial error results in a smaller than theoretical convergence rate for the spatial refinement. This can be seen in Figure 3 where the error is plotted as the mesh is isotropically refined 4 times while holding the time steps constant at 16, 32, 64, 128 and 256 steps to march through the 60 seconds of the full simulation. For the larger time step size (smaller number of steps) it is evident that refining the mesh past a certain point does not adequately reduce the total error with respect to the true solution because the temporal error is the dominant error. A similar test was carried out with fixed grid resolution and isotropically refined time steps and plotted in Figure 4. Again, the limitations of refining in only one dimension (time) are evident. When a grid of only 1176 elements is used for each time step there is almost no improvement in solution accuracy by using more than 64 time steps to complete the time accurate simulation unless the mesh resolution is increased to 4704 elements.

In the previous two examples we converged the solution to a sufficiently small residual to eliminate the error resulting from partial convergence from the results. If instead, we vary the convergence tolerance while holding the number of grid elements at each step constant for all equal spaced time steps we can again see how the largest error term will dominate the solution. In Figure 5 we have plotted the relative error with respect to the true solution for simulations isotropically refined in both space and time (i.e 16 steps of 1176, 32 steps of 4704, etc.) for varying level of convergence tolerance from 10^{-1} to 10^{-14}. From this we can see that at each space and time resolution we eventually reach a point where converging the solution further does not get us closer to the true solution (i.e. we are converging to the wrong solution). The only way to get closer to the solution is to increase the spatial and temporal resolution such as going from 150624 total degrees of freedom to 1204224 and even more.

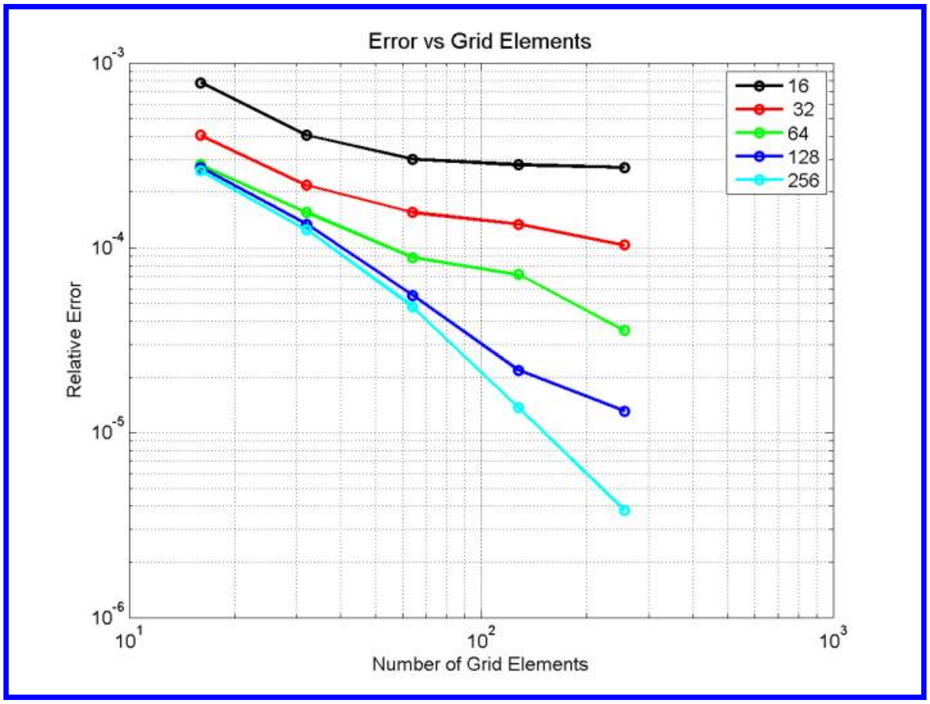

Figure 3. Effect of spatial refinement on accuracy when temporal resolution is held constant at 16, 32, 64, 128 and 256 equal spaced time steps.

Downloaded by Dimitri Mavriplis on January 15, 2013 | http://arc.aiaa.org | DOI: 10.2514/6.2013-520

Taking into consideration the above discussion we have developed a process which uses the linear estimate of the error in time, space and convergence tolerance to adapt the discretization in the most efficient manner. The flow chart for this process is shown in Figure 6 where the simulation starts at the top flowing down to the bottom and can be repeated until the desired level of accuracy has been achieved.

The process allows each component of the error to be reduced if it is above some defined tolerance. To choose the individual temporal, spatial, and algebraic error tolerance (Tol_T, Tol_S and Tol_C respectively) in the refinement decision steps of the flow chart (Fig. 6) we equidistribute the user defined global error tolerance Tol_{Global} over the three types of error refinement possibilities as shown in Equation 42. This ensures no single error type can dominate the others and all three errors will be roughly equal in magnitude when the global error tolerance has been achieved.

$$Tol_T = \frac{Tol_{Global}}{3} \qquad Tol_S = \frac{Tol_{Global}}{3} \qquad Tol_C = \frac{Tol_{Global}}{3} \tag{42}$$

IV.C. Local Refinement Criteria

The above discussion concerned how to choose which dimension of the problem (time,space or convergence tolerance) to refine but still to be discussed is what within each discretization to refine, how much to refine and how frequently to do it. To examine these aspects we will discuss the spatial, temporal and convergence error contributions separate.

IV.C.1. Space Refinement

To examine the effects of frequency of spatial refinement and how many elements to refine at each refinement we conducted a sweep of runs to examine the benefits and consequences of each. Shown in Figure 7 we have

Downloaded by Dimitri Mavriplis on January 15, 2013 | http://arc.aiaa.org | DOI: 10.2514/6.2013-520

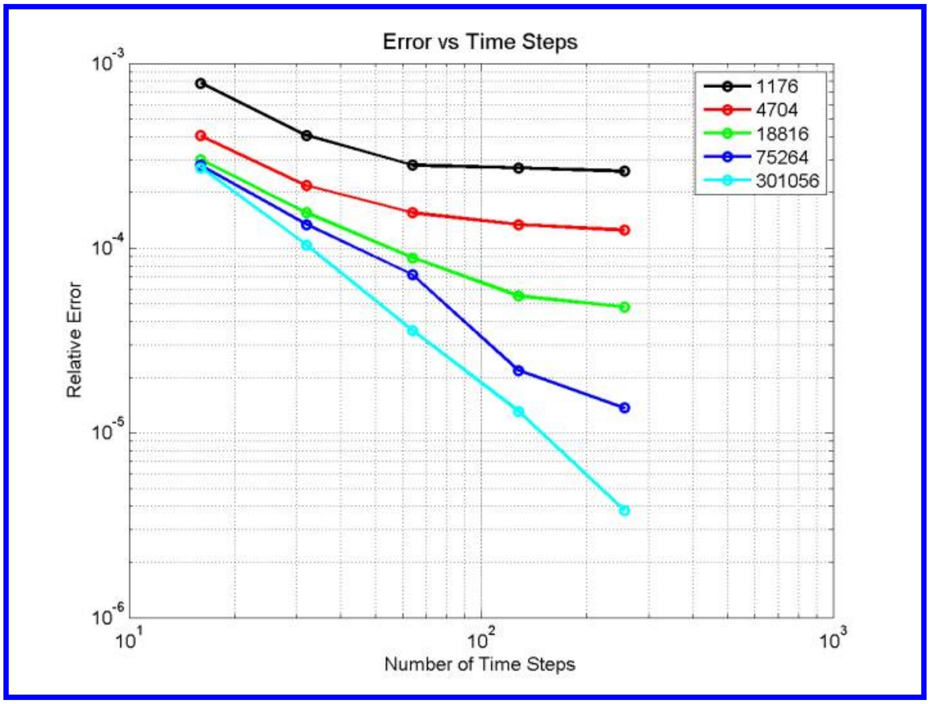

Figure 4. Effect of temporal refinement on accuracy when spatial resolution is held constant at 1176, 4704, 18816, 75264 and 301056 mesh elements per time step.

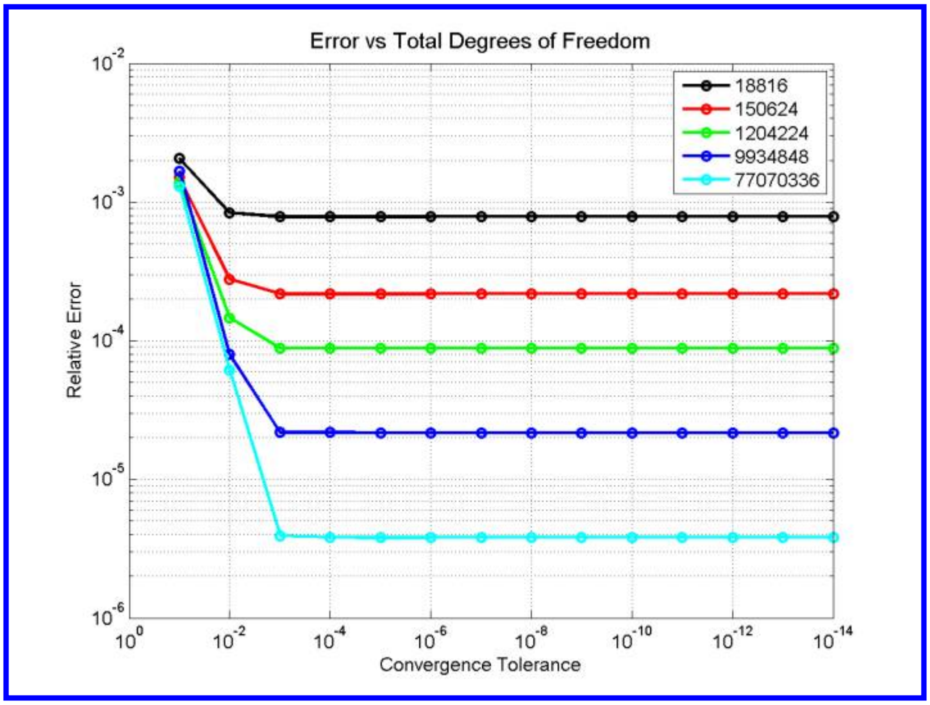

Figure 5. Effect of convergence tolerance on accuracy for 5 iso-tropically refined time and space domains starting with 16 equal time steps of 1176 mesh elements producing 18816 total elements.

Downloaded by Dimitri Mavriplis on January 15, 2013 | http://arc.aiaa.org | DOI: 10.2514/6.2013-520

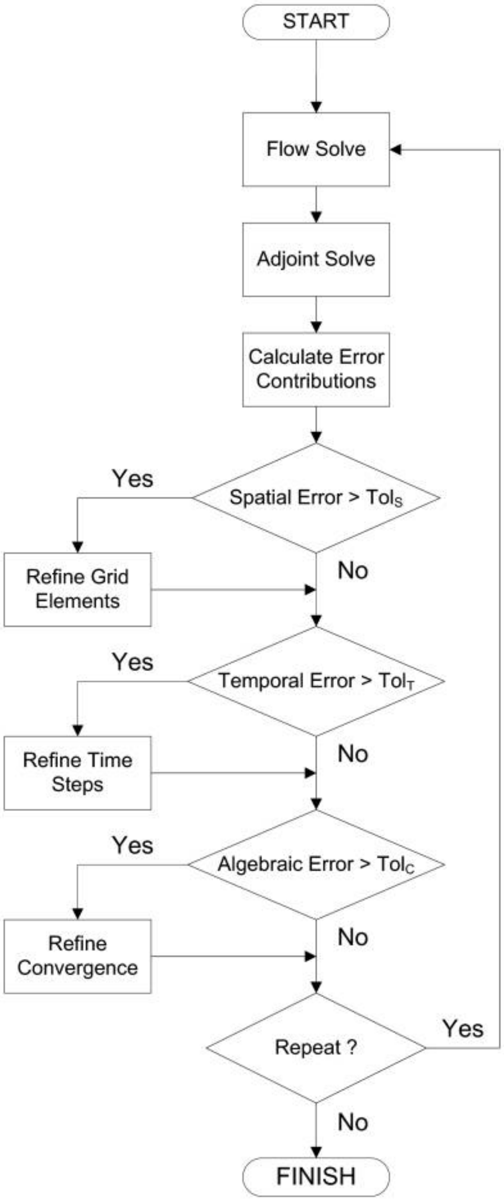
START
Flow Solve
Adjoint Solve
Calculate Error Contributions
Spatial Error > Tol_S
Yes
Refine Grid Elements
No
Temporal Error > Tol_T
Yes
Refine Time Steps
No
Algebraic Error > Tol_C
Yes
Refine Convergence
No
Repeat ?
Yes
No
FINISH

Figure 6. Global solution and refinement process

Downloaded by Dimitri Mavriplis on January 15, 2013 | http://arc.aiaa.org | DOI: 10.2514/6.2013-520

plotted a series of three full refinement sweeps while we held the frequency of refining and the amount of total error we refined at each refinement constant. Cases were run where the frequency of refinement was varied from every time step of the full time accurate simulation to every 2, 4, 8, 16, 32, 64, 128 and 256 (i.e. once per simulation) time steps. The percentage of the error refined within any one refinement interval was varied from 10%, 25%, 50%, 85%, 95%, 99% and 100% of the error.

From these plots it is evident that refining more frequently is more advantageous in that it both lowers the error for any given refinement sweep but also reduces the total degrees of freedom making the sweep less computationally expensive to perform. A somewhat interesting result is the error appears lowest when the refinement interval is set to refine every 2 or 4 time steps. It is believed this is because at those refinement intervals the refinement pattern reproduces the effect of refining a layer or two of neighboring cells (i.e. buffer cells) around the cells flagged for refinement. This is a common practice when tracking moving flow features and appears to reduce the error in this case as well.

To develop the rationale for how much spatial error to refine at every time step we can look at previous work[20] for the steady state problem where the goal was to equidistribute the spatial error among all the cells. For the unsteady problem we will extend this to treat the time accurate problem as one single system of equations much like we did to develop the adjoint equations (Eqn. 21). To determine which elements to refine we define a maximum allowable error level for each cell s on every time step n by equidistributing the spatial error tolerance Tol_S developed in Section IV.B over all the cells of the N time step simulation.

$$s = \frac{Tol_S}{\sum_{n=1}^{N} Elements_n} \tag{43}$$

Again, extending the steady state case, a spatial refinement parameter r_s^i for each element i is defined as the ratio of the cell-wise error ε_s^i to the maximum allowed error s as given by:

$$r_s^i = \frac{\varepsilon_s^i}{s} \tag{44}$$

Using this definition we can flag an element at any time step for refinement when its refinement parameter exceeds a threshold λ_s that is pre-defined. This has the desirable effect of allowing spatial refinement every time step, if elements within the time step exceed the error threshold, but it does not force refinement to occur every time step if no elements are above the threshold.

To choose how many elements to refine at any refinement step the simplest choice would be all elements that exceed a threshold of one ($\lambda_s = 1$) where the allowable element spatial error s is determined by the global error as shown before in Equation 43. Doing this at every refinement interval would insure that eventually the error would become equidistributed and result in an acceptable global spatial error. If we follow this scenario and use our results shown in Figure 7 we can see that refining a large percentage of the error at any step will result in the total degrees of freedom in the simulation quickly getting large. A smarter choice and one that has already been explored for steady state problems by Nemec, Aftosmis and Wintzer[20] is to use a decreasing threshold value. In this scenario the threshold is initially set high where only a small fraction of the highest error cells are refined then on subsequent steps the threshold is lowered until eventually a threshold of 1 is reached. This has the effect of equidistributing the error while not increasing the computational cost significantly, and then in the last couple refinement sweeps heavily refining to meet the desired error goals.

IV.C.2. Time Refinement

The time refinement will follow a similar path to the spatial refinement although it will be modified by the requirement that all elements within a time step are equally stepped through time. In previous work we have investigated techniques for enabling spatially varying time steps using adjoint error estimators.[21] However, such approaches are best suited for use with space-time formulations, which is beyond the scope of the current investigation. To begin, we will again seek to equidistribute the temporal error evenly among all elements of all time steps by developing an allowable error t for each element i by taking the allowable time error tolerance Tol_T, developed in Section IV.B and dividing it by the total number of elements in the

Downloaded by Dimitri Mavriplis on January 15, 2013 | http://arc.aiaa.org | DOI: 10.2514/6.2013-520

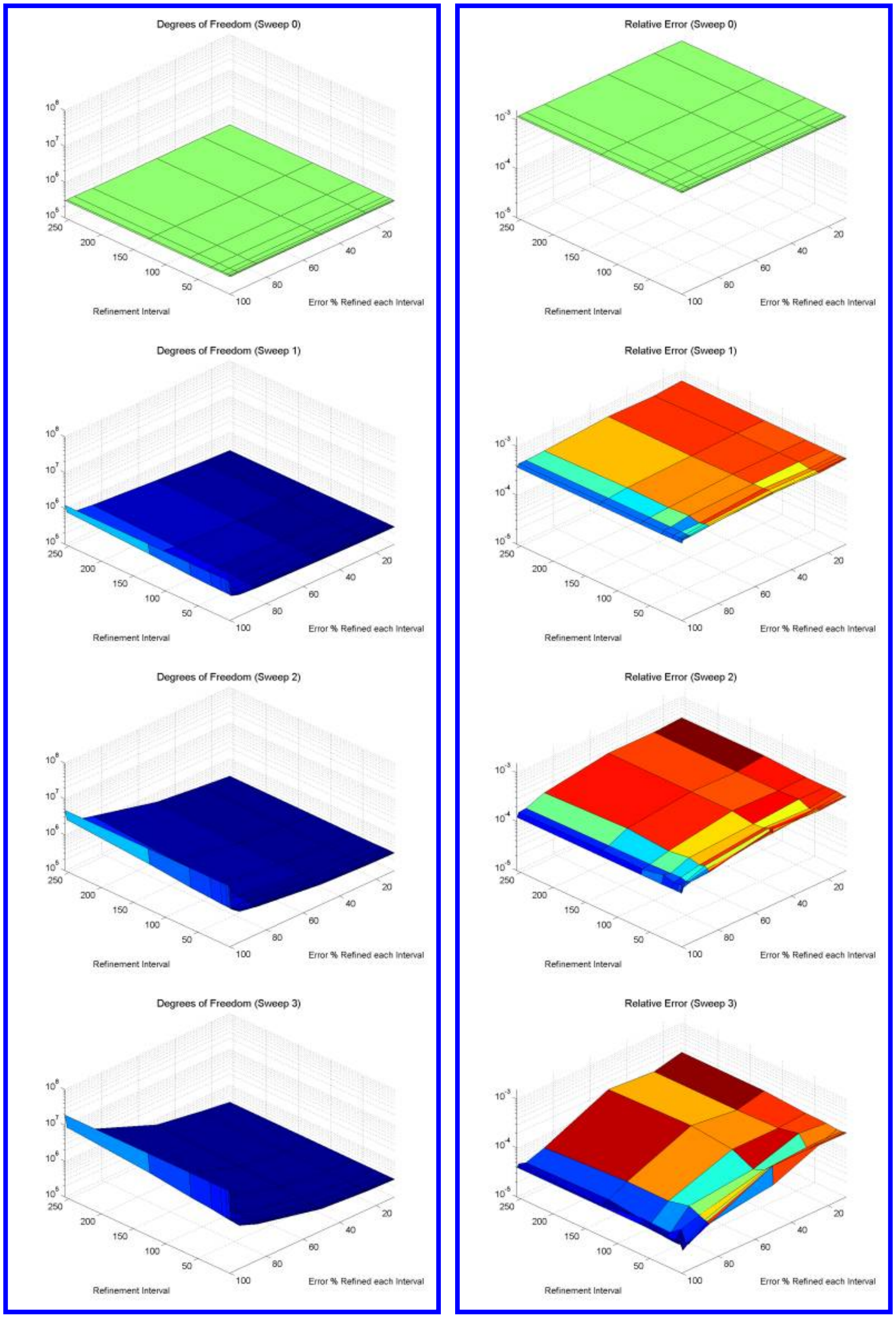

Figure 7. Total degrees of freedom and spatial error for 3 refinement passes when the frequency and percentage of error refined vary.

N time step simulation.

$$t = \frac{Tol_T}{\sum\limits_{n=1}^{N} Elements_n} \tag{45}$$

The temporal refinement parameter r_t^n is then implemented to sum all the cell-wise error contributions ε_t^i within a single time step and dividing it by the allowable error for a step with the same number of elements.

$$r_t^n = \frac{\sum\limits_{i=1}^{Elem^n} \varepsilon_t^i}{t \times Elements_n} \tag{46}$$

By summing over all the elements within the time step we have reduced the spatially varying cell-wise temporal error down to a single value for each time step to indicate which time steps contain the most temporal error. To control the number of time steps that are refined at the completion of any one complete simulation sweep we will use a threshold for the time λ_t which can be varied in a similar descending manner as was decided upon for the spatial refinement (Sec.IV.C.1).

IV.C.3. Convergence Refinement

To perform the refinement of the convergence tolerance we could follow the spatial refinement procedure if our flow solver allowed us to selectively remove equations or elements from the newton iterations as their individual convergence tolerances are met. Since our solver does not have this capability we will follow the temporal refinement steps and will only summarize the steps below.

Each element i has an allowable error c calculated using the pre-determined convergence tolerance Tol_C based on how many total elements within the time accurate simulation as shown.

$$c = \frac{Tol_C}{\sum\limits_{n=1}^{N} Elements_n} \tag{47}$$

The refinement parameter r_c^n will again be formed by summing all the cell-wise error contributions ε_c^i within a single time step and dividing it by the allowable error for a step with the same number of elements.

$$r_c^n = \frac{\sum\limits_{i=1}^{Elem^n} \varepsilon_c^i}{c \times Elements_n} \tag{48}$$

Another threshold variable λ_c will be used to control the amount of time steps the convergence tolerance is reduced at the completion of any one sweep of the simulation.

V. Results

V.A. Convecting Vortex

For the first test of the new algorithm we use the model test problem of section IV.A with a starting discretization of 1176 grid elements over 16 equal spaced time steps converged 1 order in magnitude with the final relative error in the objective set to $Tol_{Global} <= 10^{-6}$. The simulation is allowed to progress through the refinement algorithm presented in Figure 6 for 7 cycles of the loop with the threshold value decreasing in each successive loop from 32, 16, 8, 4, 2 and then fixed at 1 until the error tolerance is achieved. The plots of the adapted convergence tolerance, time step size and number of mesh elements at each time step are displayed in Figures 8, 9 and 10 for each cycle of the algorithm.

The plot of the convergence tolerance shows the requested reduction of 10^{-1} is sufficient for the first 3 refinement cycles but after using the same convergence tolerance on the 4th cycle the decreasing error threshold of 4 is exceeded. Once the tolerance has been exceeded, the algorithm branches and identifies the time steps at the start of the simulation as those exceeding the error threshold for the number of elements at

Downloaded by Dimitri Mavriplis on January 15, 2013 | http://arc.aiaa.org | DOI: 10.2514/6.2013-520

Downloaded by Dimitri Mavriplis on January 15, 2013 | http://arc.aiaa.org | DOI: 10.2514/6.2013-520

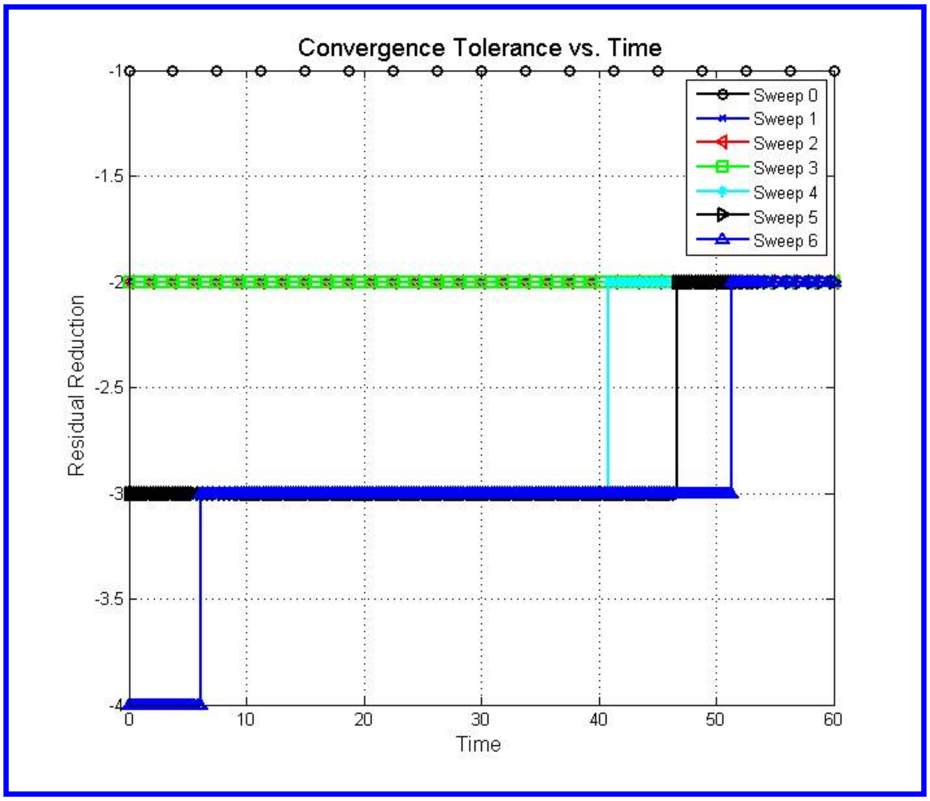

Figure 8. Convergence tolerance for each sweep of the time accurate simulation.

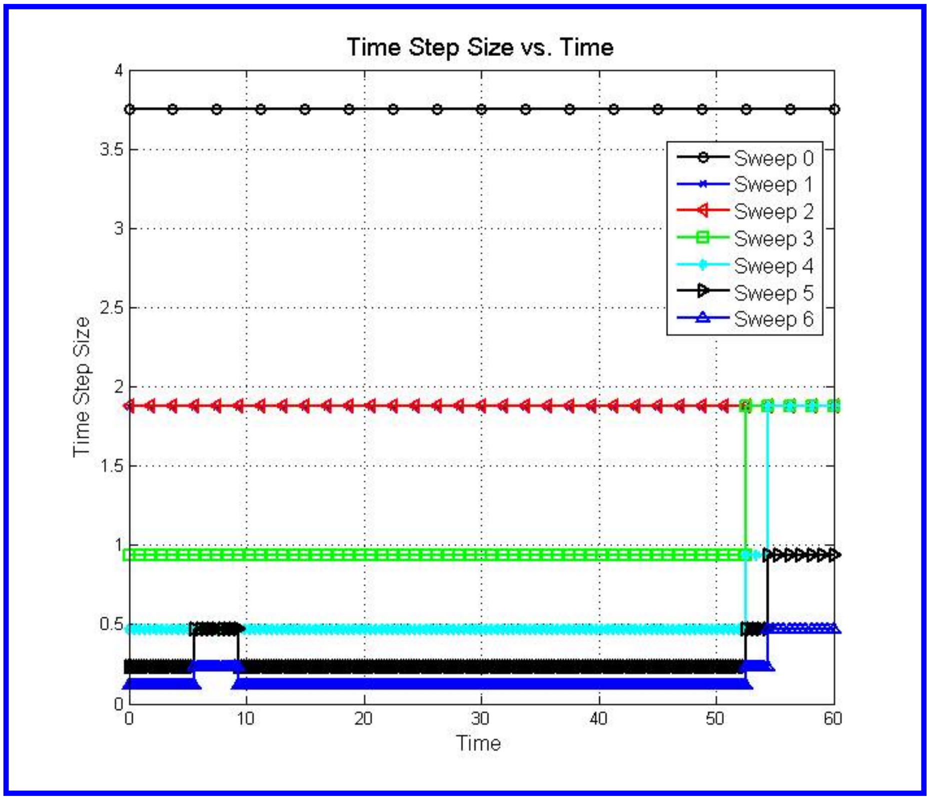

Figure 9. Time step size for each sweep of the time accurate simulation.

Downloaded by Dimitri Mavriplis on January 15, 2013 | http://arc.aiaa.org | DOI: 10.2514/6.2013-520

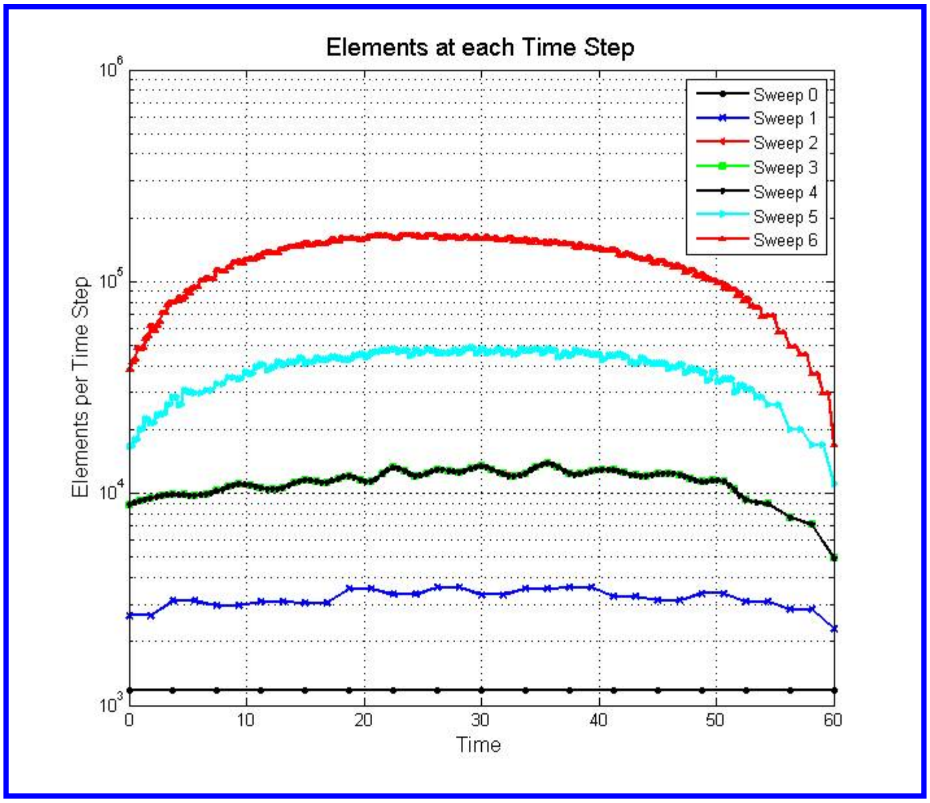

Figure 10. Number of elements for each sweep of the time accurate simulation.

each time step. These time steps are flagged for an increased convergence criteria (i.e. lower final residual) and leaves the other steps as they were before. A similar pattern is followed on the 5th and 6th cycles of the algorithm where again the initial time steps are found to need a tighter convergence tolerance to meet the error threshold.

The time step plot (Fig. 9) shows a different pattern, where the first flow simulation using only 16 time steps to model the full time accurate simulation was found to exceed the temporal error threshold. Once inside the refinement branch the algorithm determined every time step was exceeding the allowable threshold. For the next 2 cycles of the algorithm the temporal error resulting from the now 32 equally spaced time steps is below the decreasing threshold values but on the 4th, 5th and 6th cycles again the temporal error threshold is exceeded and the time steps at the start of the simulation are flagged for refinement.

The spatial refinement plots (Fig. 10) show yet a different pattern where the initial 2 cycles through the algorithm exceed the spatial error threshold. Within these cycles, when the spatial refinement branch has been taken, all elements for all time steps are evaluated against the element error threshold and in each case at least some elements in each time step are flagged for refinement. On the subsequent 2 cycles of the algorithm the spatial error is found to be below the error threshold so no spatial refinement occurs until the 5th and 6th cycles where again at least some elements on each time step are refined. Images of the refined region within the computational mesh at non-dimensional times of t = 0, 30 and 60 are shown in Figure 11 for each sweep of the algorithm. The mesh refinement patterns clearly show refinement from the location of the vortex to the region the time integrated functional is integrated over. This later feature is important and seen as a major advantage over simpler feature tracking algorithms such as gradient based refinement.

Calculating the flow and adjoint solution at each cycle of the algorithm is expensive and it is unknown whether going through multiple solution cycles to arrive at a final answer is cheaper than refining the discretization and running it once. To evaluate this we have defined our own measure of computational expense called the "Work Unit" which is the cost to converge the residual of one element one order of magnitude on one time step. For example, a 16 time step simulation on a fixed grid of 1176 elements with the residual on each step being converged 3 orders in magnitude would have a cost of $16*1176*3 = 56448$ work units. Using this definition we have plotted the cost of each adapted flow solution in Figure 12 along with the

Downloaded by Dimitri Mavriplis on January 15, 2013 | http://arc.aiaa.org | DOI: 10.2514/6.2013-520

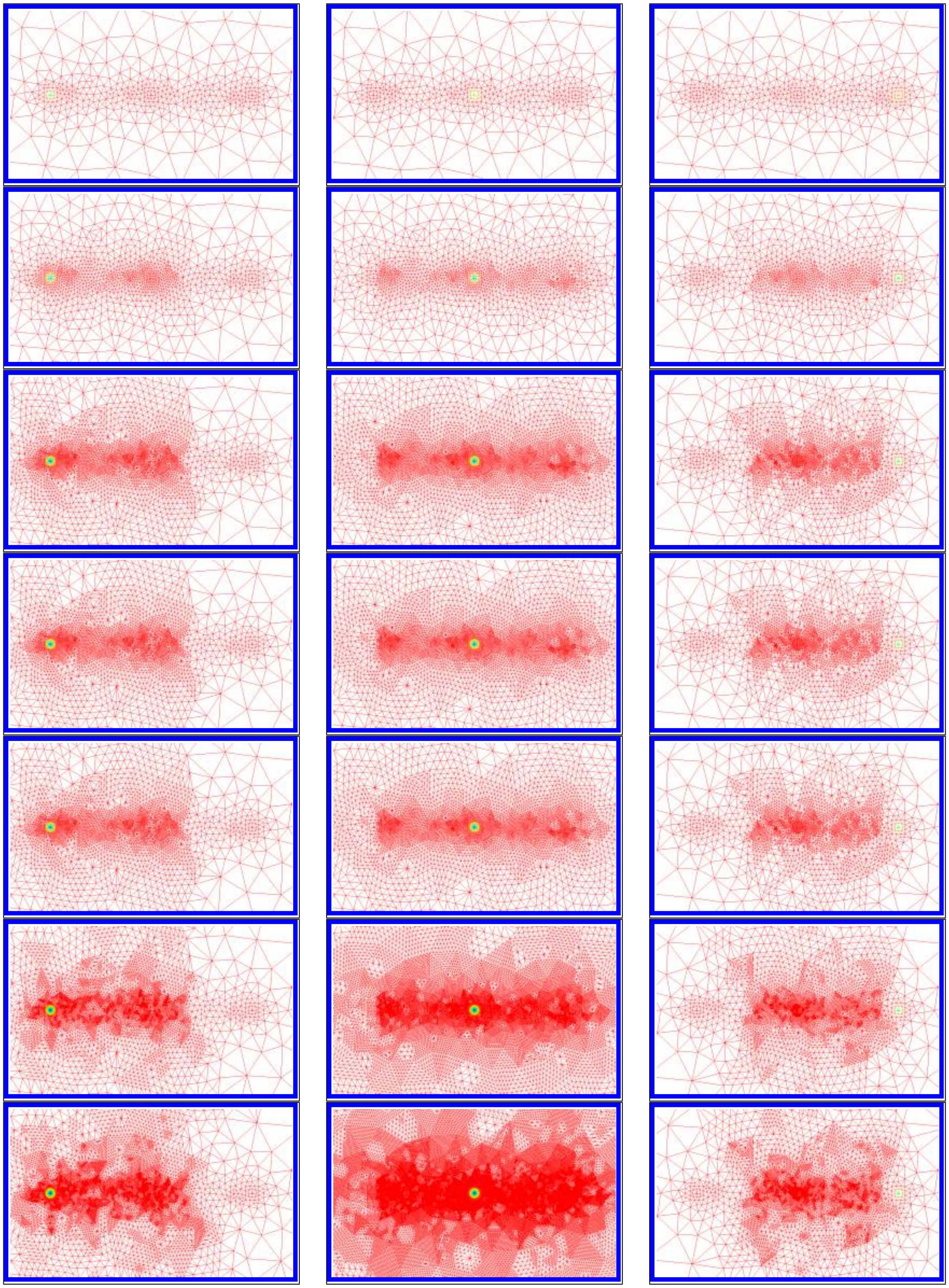

Figure 11. Convecting vortex and adapted grid at time = 0, 30, and 60 for 7 different refinement sweeps.

Downloaded by Dimitri Mavriplis on January 15, 2013 | http://arc.aiaa.org | DOI: 10.2514/6.2013-520

relative error of the objective when compared to the value of the true analytic objective. As a comparison, we've plotted a second curve resulting from uniformly refining the temporal and spatial discretization 4 times where the coarsest simulation was converged 1 order of magnitude and each subsequent refined discretization had the convergence tightened by an additional order of magnitude (i.e. 5th solution is converged 5 orders of magnitude).

The cost of a single solution alone doesn't tell the true expense of arriving at the final adapted solution using adjoint based refinement, therefore we have plotted the relative error as a function of cumulative work units in the second plot in Figure 12 along with the same uniform curve as before. The cumulative cost is the cost to solve the flow and adjoint on all previous adaption cycles plus the cost of the current solution. From this measure of the total cost to arrive at the final adapted solution we can see that the final solution is the major expense and all previous adaption cycles do not add much to the overall cost to reach the final result. What the previous adaption cycles do though, is more accurately target the regions for refinement so that each cycle produces nearly the same relative error as uniform refinement but with far less expense.

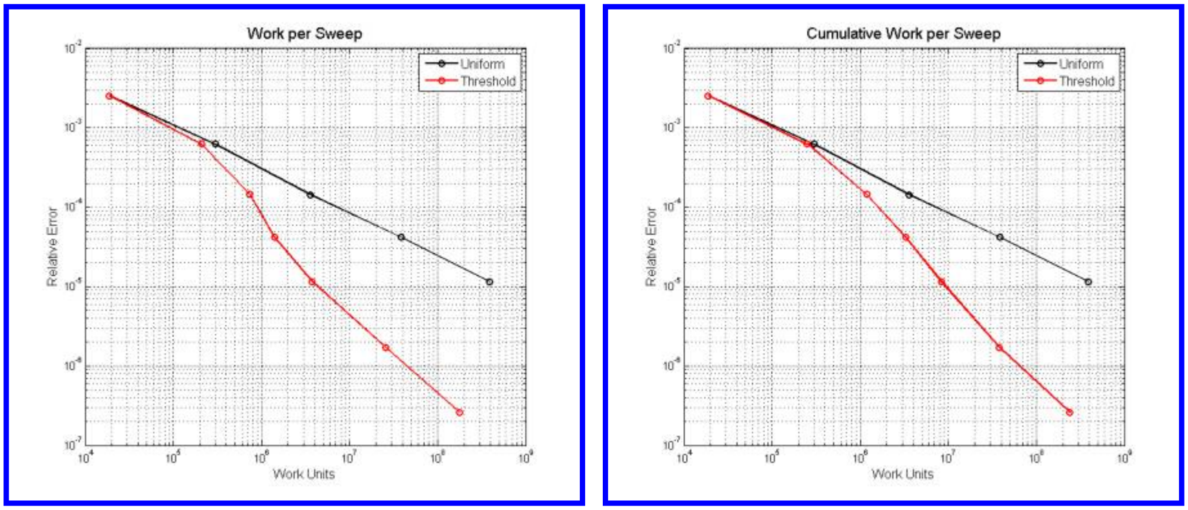

Figure 12. Work units and cumulative work units for each sweep of the time accurate simulation.

V.B. Airfoil with Vortex

The second test problem is an identical strength vortex as to that from the model problem of section IV.A convecting past a NACA 0012 airfoil placed with the leading edge at the origin. The vortex is initially placed 2 chord lengths below the airfoil and 15 chord lengths in front of the airfoil leading edge. The free stream velocity is again $M_\infty = 0.5$ and the simulation is run for 60 non-dimensional time units placing the vortex center roughly 14 chord lengths behind the airfoil trailing edge at the final time step. The initial grid contains 2401 elements and again 16 initial time steps converged 1 order of magnitude were used to march through the initial simulation. The starting condition for the time accurate steps is the fully converged flow around the airfoil with the vortex superimposed into the flow field at the starting position. This is not the exact solution at this point in time but it is assumed the vortex is far enough away from the airfoil to have little effect on the objective function of the time integrated lift coefficient on the airfoil.

Starting from the initial condition the simulation is allowed to cycle through the algorithm presented in Figure 6 until the global error estimate is less than 10^{-2} (i.e. $Tol_{Global} < 10^{-2}$). The threshold values are again set to decrease each cycle through the refinement algorithm using values of 8, 4, 2 and 1. Plots of the resulting convergence tolerance, time step size and number of mesh elements at each time step are shown in Figures 13, 14 and 15.

The plot of the convergence tolerance shows the requested reduction of 10^{-1} is sufficient for the first 2 refinement cycles but after using the same convergence tolerance on the 3rd cycle the decreasing error threshold of 2 is exceeded. Once the tolerance has been exceeded the algorithm branches and identifies the time steps less than $t = 37.5$ non-dimensional time units as needing a tighter convergence tolerance. A similar pattern is followed on the next cycle of the algorithm where the convergence tolerance is tightened on all time steps less than $t = 11.25$ non-dimensional time units.

Downloaded by Dimitri Mavriplis on January 15, 2013 | http://arc.aiaa.org | DOI: 10.2514/6.2013-520

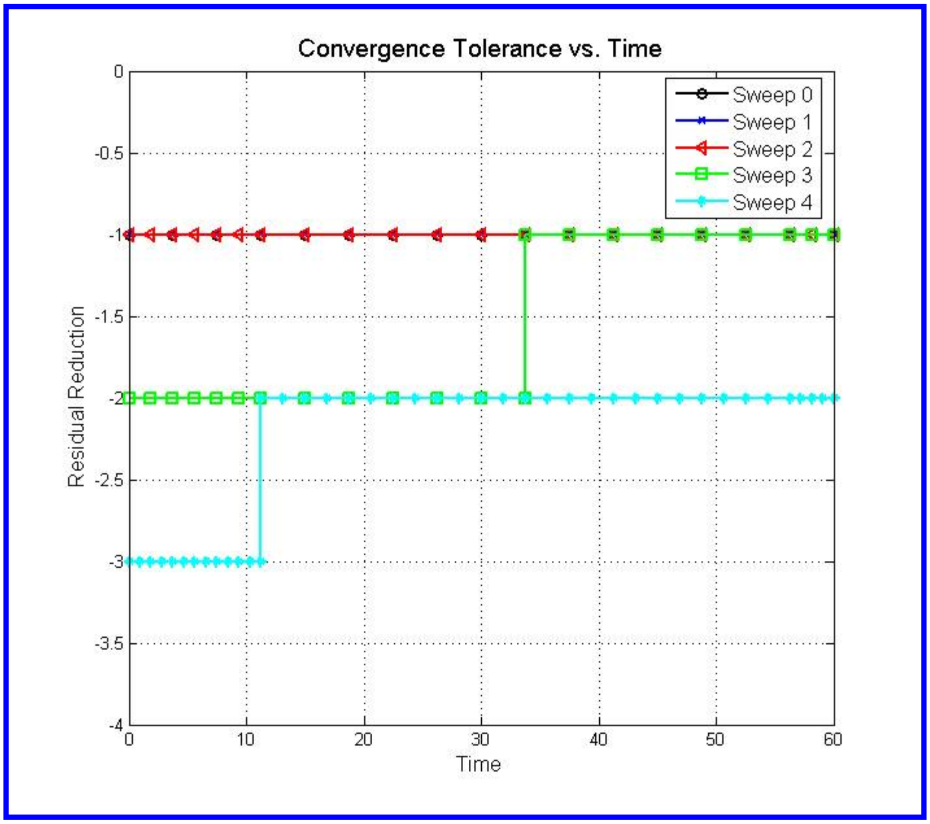

Figure 13. Convergence tolerance for each sweep of the time accurate NACA 0012 simulation.

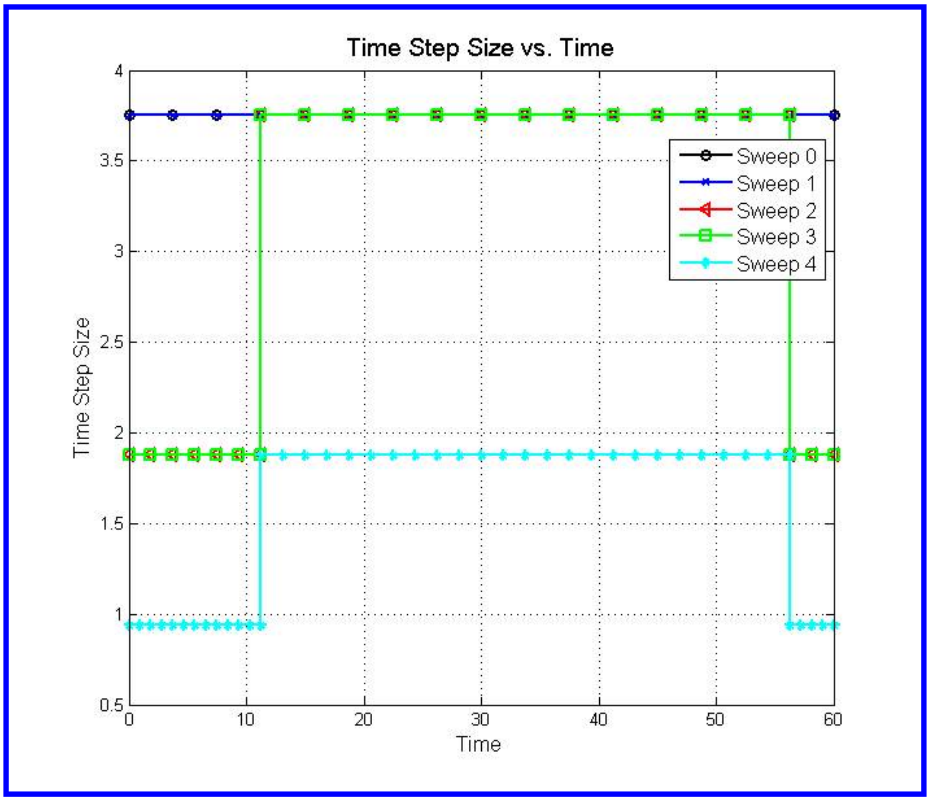

Figure 14. Time step size for each sweep of the time accurate NACA 0012 simulation.

Downloaded by Dimitri Mavriplis on January 15, 2013 | http://arc.aiaa.org | DOI: 10.2514/6.2013-520

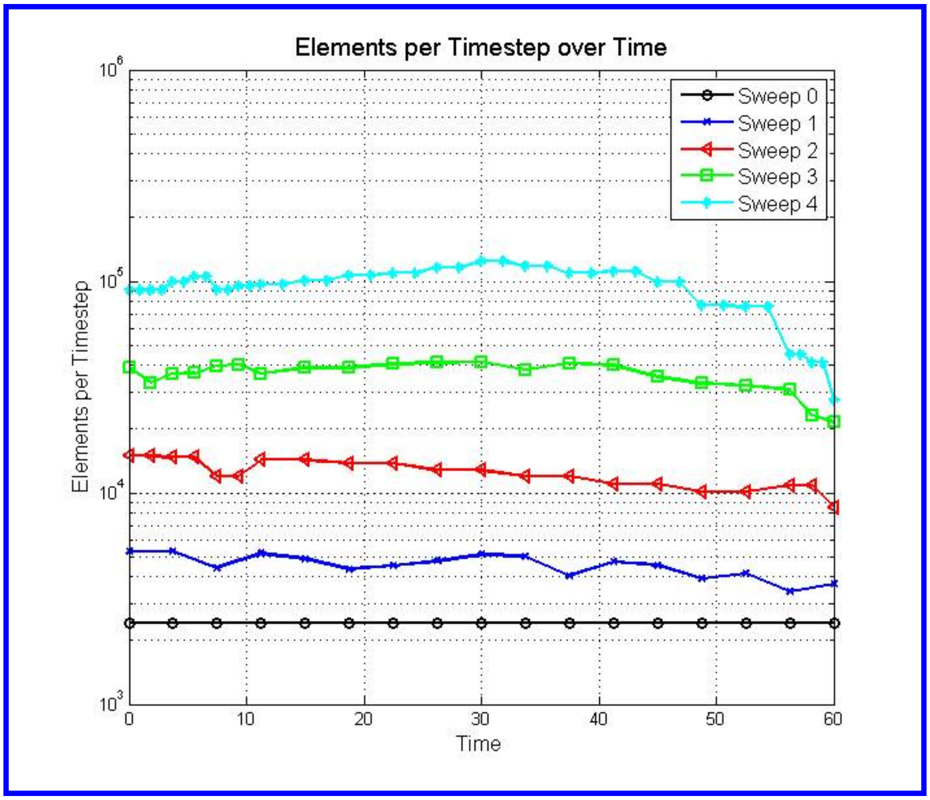

Figure 15. Number of elements for each sweep of the time accurate NACA 0012 simulation.

The time step plot (Fig. 14) shows a very similar pattern to that of the convergence refinement where the initial time step size is adequate for an initial temporal error threshold of 8. On the second cycle of the algorithm the temporal error exceeds the error threshold which is now at 4 and time steps at both the start and end of the time series are flagged for refinement. For the next cycle of the algorithm the temporal error resulting from the now non-uniformly spaced time steps is below the decreasing threshold value but on the 4th cycle again the temporal error threshold is exceeded only this time all time steps are flagged for refinement

The element refinement plots (Fig. 15) show the spatial error was found to exceed the spatial threshold every cycle of the refinement algorithm. For the initial refinement the number of elements went up by almost a factor of two, indicating roughly a quarter of the elements had spatial error contributions in excess of the threshold value of 8. On this same cycle the convergence and temporal error thresholds were met indicating that not only were many elements exceeding the limits but that spatial error was the dominant error on the initial discretization.

Images of the refined region within the computational mesh at non-dimensional times of t = 0, 30 and 60 are shown in Figure 16 for each sweep of the algorithm. Again the mesh refinement patterns show refinement not only around the convecting vortex but also along the path of its influence on the airfoil which by use of the objective function is identified as the item of interest within the problem.

To evaluate the cost of obtaining the adapted solution we have again calculated the work units required to solve a single flow solution and the cumulative work units to achieve an adapted solution in Figure 17. As a comparison the initial discretization was uniformally refined with increasing levels of convergence, as done for the isolated vortex case of Section V.A, and plotted along with the resulting objective values. As before the adapted discretization achieves comparable values of the objective function to that of uniform refinement while requiring far less cumulative work to achieve the results.

Downloaded by Dimitri Mavriplis on January 15, 2013 | http://arc.aiaa.org | DOI: 10.2514/6.2013-520

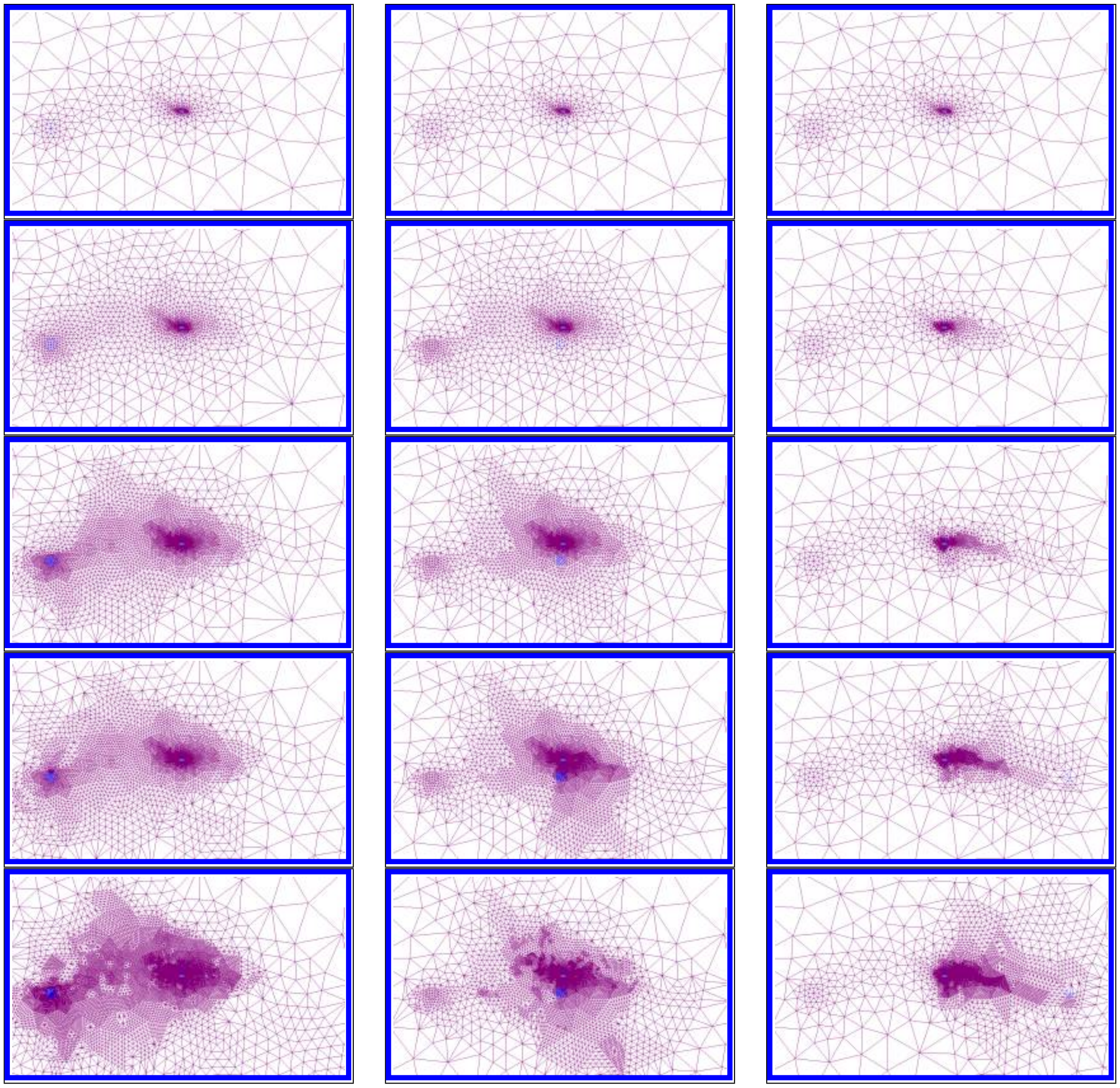

Figure 16. NACA 0012 airfoil with convecting vortex and adapted grid at time = 0, 30, and 60 for 5 different refinement sweeps.

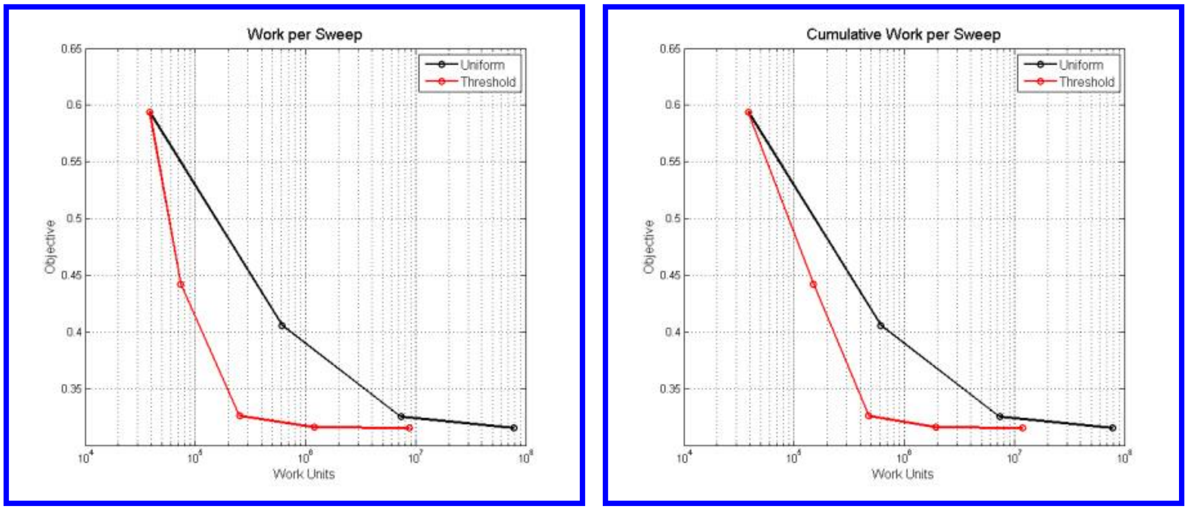

Figure 17. Work units and cumulative work units for each sweep of the time accurate simulation.

VI. Conclusions

This work presented a framework for the reduction of the spatial, temporal and algebraic error by use of the discrete adjoint solution. Using the three error estimates it was shown that refinement of only one error term without consideration of the other two error contributions would not increase the over all accuracy of the simulation. A method was developed and presented to equally weigh each of the three error contributions and selectively refine the regions with the highest error, thereby equidistributing the error over the whole domain. Two test cases were shown where significant gains in efficiency were demonstrated, even when taking into account for the solution of the flow and adjoint on all previous refinement cycles. This research has focused on 2 dimensional problems of short time duration but the real benefit would come in 3 dimensional problems were the extra dimension equates to a much larger computational expense and consequently an algorithm to more efficiently target error would result in larger cost savings. In the long term, the ideal application of this research would be within a space-time framework where all elements could be selectively refined in space, time and convergence without regard for keeping all elements at the same time or converging all elements within a time step an equal amount.

Downloaded by Dimitri Mavriplis on January 15, 2013 | http://arc.aiaa.org | DOI: 10.2514/6.2013-520

References

[1] Venditti, D. A. and Darmofal, D. L., "Grid adaptation for functional outputs: application to two-dimensional inviscid flows," *Journal of Computational Physics*, Vol. 176, No. 1, 2002, pp. 40–69.

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

[3] Nemec, M. and Aftosmis, M., "Adjoint error estimation and adaptive refinement for embedded-boundary cartesian meshes," No. 2007-4187, AIAA Paper, 2007.

[4] Houston, P., Rannacher, R., and Suli, E., "A posteriori error analysis for stabilised finite element approximations of transport problems," *Computer Methods in Applied Mechanics and Engineering*, Vol. 190, No. 11-12, 2000, pp. 1483–1508.

[5] Becker, R. and Rannacher, R., "An optimal control approach to a posteriori error estimation in finite element methods," *Acta Numerica*, , No. 10, 2001, pp. 1–102.

[6] Giles, M. B. and Suli, E., "Adjoint methods for PDEs: a posteriori error analysis and postprocessing by duality," *Acta Numerica*, , No. 11, 2002, pp. 145–236.

[7] Mani, K. and Mavriplis, D. J., "Discrete adjoint based time-step adaptation and error reduction in unsteady flow problems," No. 2007-3944, AIAA Paper, 2007.

[8] Mani, K. and Mavriplis, D. J., "Error estimation and adaptation for functional outputs in time-dependent flow problems," *Journal of Computational Physics*, Vol. 229, No. 2, 2010, pp. 415–440.

[9] Luo, Y. and Fidkowski, K. J., "Output-based space-time mesh adaption for unsteady aerodynamics," No. 2011-491, AIAA Paper, 2011.

[10] Li, S. and Petzold, L., "Adjoint sensitivity analysis for time-dependent partial differential equations with adaptive mesh refinement," *Journal of Computational Physics*, Vol. 198, No. 1, 2004, pp. 310–325.

[11] Flynt, B. T. and Mavriplis, D. J., "Discrete Adjoint Based Adaptive Error Control in Unsteady Flow Problems," No. 2012-78, AIAA Paper, 2012.

[12] Roe, P., "Approximate Riemann solvers, parameter vectors, and difference schemes," *Journal of Computational Physics*, Vol. 43, No. 1, 1981, pp. 357–372.

[13] Mavriplis, D. J., "Adaptive meshing techniques for viscous flow calculations on mixed element unstructured meshes," *International Journal for Numerical Methods in Fluids*, Vol. 34, No. 2, 2000, pp. 93–111.

[14] Buhmann, M. D., *Radial Basis Functions*, Cambridge University Press, 2004.

[15] Wendland, H., "Piecewise polynomial, positive definite and compactly supported radial functions of minimal degree," *Advances in Computational Mathematics*, Vol. 4, No. 1, 1995, pp. 389–396.

[16] Bonet, J. and Peraire, J., "An alternating digital tree (ADT) algorithm for 3D geometric searching and intersection problems," *International Journal for Numerical Methods in Engineering*, Vol. 31, No. 1, 1991, pp. 1–17.

[17] Davoudzadeh, F., Mcdonald, H., and Thompson, B., "Accuracy evaluation of unsteady cfd numerical schemes by vortex preservation," *Computer in FLuids*, Vol. 24, No. 1, 1995, pp. 1995.

[18] Yee, H., Sandham, N., and Djomehri, M., "Low dissipative high order shock-capturing methods using characteristic-based filters," *Journal of Computational Physics*, Vol. 150, No. 1, 1999, pp. 199–238.

[19] Whitfield, D. and Janus, J., "Three-Dimensional Unsteady Euler Equations Solution Using Flux Vector Splitting," No. 84-1552, AIAA Paper, 1984.

[20] Nemec, M., Aftosmis, M., and Wintzer, M., "Adjoint-Based Adaptive Mesh Refinement for Complex Geometries," No. 2008-0725, AIAA Paper, 2008.

[21] Mani, K. and Mavriplis, D. J., "Spatially Non-Uniform Time-Step Adaptation for Funtional Outputs in Unsteady Flow Problems," No. 2010-121, AIAA Paper, 2010.