Journal of Computational Physics 229 (2010) 415–440

Contents lists available at ScienceDirect

Journal of Computational Physics

journal homepage: www.elsevier.com/locate/jcp

Error estimation and adaptation for functional outputs in time-dependent flow problems

Karthik Mani *, Dimitri J. Mavriplis

Department of Mechanical Engineering, University of Wyoming, Laramie, WY 82071-3295, United States

ARTICLE INFO

Article history:
Received 15 January 2009
Received in revised form 15 September 2009
Accepted 24 September 2009
Available online 9 October 2009

Keywords:
Unsteady adjoint method
A posteriori error estimation
Temporal adaptation

ABSTRACT

The paper presents an adjoint-based approach for determining global error in the time domain that is relevant to functional outputs from unsteady flow simulations. The algorithm is derived for the unsteady Euler equations that are discretized for second-order accuracy in both space and time and takes into account the effect of dynamic meshes. In addition to error due to temporal resolution, the formulation also takes into account algebraic error arising from partial convergence of the governing equations at each implicit time-step. The resulting error distributions are then used to drive adaptation of the temporal resolution and the convergence tolerances for the governing equations at each time-step. The method is demonstrated in the context of both time-integrated and instantaneous functionals and the results are compared against traditional adaptation methods.

1. Introduction

The time-integration process in unsteady flow problems is typically carried out using a uniform time-step which is applied throughout the time domain of interest. The limitation on the size of the time-step is mainly governed by the temporal resolution required to capture essential unsteady flow features and deliver acceptably low temporal error. The traditional approach is to use a uniform time-step of a size that by engineering knowledge is known to be sufficient to capture essential flow features. However, the inherent non-linearity of the flow equations makes it difficult to determine the temporal scales of the flow features that are relevant to the functional output of interest from the unsteady simulation. Additionally, the disadvantage of using a uniform time-step is that it becomes expensive due to high resolution in regions of the time domain where such resolution may not be necessary. While the overall cost of an unsteady simulation may be correlated against the resolution of the time domain, the primary cost of obtaining the solution of the flow equations at each time level arises from the non-linear iterative solution required at that time-step in the context of an implicit time discretization. Typically the equations are never solved to machine precision unless the spatial resolution of the problem under consideration is relatively coarse. Convergence to small tolerance levels can become time consuming for stiff problems particularly in the presence of anisotropic mesh effects. A sensible guideline for determining suitable implicit system convergence levels is that the algebraic error resulting from incomplete system convergence at each time-step be reduced to levels below the local temporal discretization error at each time-step [1,2]. However, this requires a good measure of the local temporal error as well as some estimate of the algebraic error due to incomplete system convergence, and is seldom enforced. The end result is that, as in the case of temporal resolution, some predetermined constant convergence limit is set based purely on engineering judgment such that the resulting solution has acceptably low overall error at practical computational expense. It is quite possible that the equations converged to limits set by such ad-hoc methods could result in unnecessarily restrictive convergence in

* Corresponding author.
E-mail addresses: kmani@uwyo.edu (K. Mani), mavripl@uwyo.edu (D.J. Mavriplis).

0021-9991/$ - see front matter © 2009 Elsevier Inc. All rights reserved.
doi:10.1016/j.jcp.2009.09.034

some regions of the time domain and insufficient convergence in others. Under most circumstances the effect or error due to insufficient convergence becomes difficult to quantify and even harder to remedy.

An adjoint-based approach permits the estimation of error relevant to a functional, and the corresponding distribution of this error in the time domain. The method relies on applying time-dependent discrete adjoint equations on a Taylor expansion of the functional. A linear approximation of the error in the functional between temporal meshes of two different resolutions can be estimated by solving the flow problem and the adjoint problem on the coarser temporal mesh, and then projecting the flow and adjoint solutions to the finer temporal mesh. It should be noted that in this particular formulation, only the change in the functional between two different temporal resolutions is predicted by the adjoint equations, and not the total error defined as the difference between the current and exact analytic value of the functional.

Adjoint-based approaches also permit distinct identification of the contributions to this error as arising from the two primary sources, namely the temporal resolution and the effect of partial convergence of the equations at each time-step. Additional separation of the error into flow equation and mesh equation components provides the criteria necessary to target specifically the set of governing equations that influences the functional the most. This is uniquely different from common adaptation or time-step control schemes which are based on estimates of the local temporal error at each time-step, in that the global temporal error in the functional can be estimated and adapted on.

The total error incurred in time-dependent simulations can be broken down into spatial error, temporal error, and algebraic error. For steady-state problems, adjoint methods have been used successfully to drive adaptive mesh refinement schemes for reducing the spatial discretization error for output functionals of interest [3–11]. Spatial mesh adaptation in time-dependent problems can be approached in one of two ways. The first and more straightforward approach is to use a single adapted spatial mesh at all time levels, where the adaptation is based on the spatial error distribution over all time levels. The application of adjoint equations to spatial mesh adaptation in time-dependent problems in this manner has been investigated in the past [12,13]. The second approach is to adapt the baseline spatial mesh uniquely at each time level on the basis of the spatial error at that time level. However, this creates discontinuities in time as new grid points are added and deleted between time-steps leading to the loss of one-to-one correspondence between spatial elements at different time levels. Such adaptation of the spatial mesh has been attempted in time-dependent problems using estimates of the local spatial error but not using adjoint equations to estimate global error in functionals of interest [14,15]. The targeting of temporal discretization error and the adaptation of the time-step size in addition to spatial mesh adaptation adds another layer of complexity to the problem. While the full error reduction and control problem for time-dependent problems must involve simultaneous adaptation in time and in space, the focus of the current work is on adjoint-based adaptation in the time domain, in order to reduce temporal error, with no attempt to reduce spatial discretization error. However, algebraic error due to incomplete convergence of the governing equations at each time-step is targeted in this work.

Much work has been done on temporal adaptation and error control in the absence of spatial adaptation, particularly for ordinary differential equations [16,17]. The most common approach consists of using local temporal error estimation to drive time-step size selection, where the temporal error at a given time-step is estimated by discretizing the time derivative with different orders-of-accuracy and comparing the corresponding results [18]. Also, such methods implicitly assume that any solutions obtained are numerically exact, or at least converged to levels below the local temporal discretization error, and do not consider the effect of partially converged solutions. While local error estimation has been used successfully for temporal refinement, a more appealing approach would be one that targets the global temporal error for a functional of interest directly, since for any particular functional, there is no guarantee that a gradient based or any other local error-based adaptation of the time domain would necessarily lead to an improved estimate of the functional. As in the case of spatial error estimation, this arises from the fact that there may be little or no correlation between the flow solution gradients in the time domain and the functional of interest. If direct information such as the sensitivity of the functional to the solution at each time interval, which in turn is a function of the time-step size and convergence criterion at that interval were available, a more targeted and efficient time adaptation scheme can be developed. While adjoint methods have been used in the spatial domain for goal oriented error estimation and control, their use in the time domain has mostly been confined to simpler problems such as those involving ordinary differential equations [19,20] although their use for partial differential equations has recently been demonstrated [21]. This work possibly represents the first attempt at extending and applying unsteady adjoint algorithms [22–25] for the purpose of estimating global temporal discretization error and adapting the time domain in fluid flow problems with dynamic meshes.

2. Analysis problem formulation

2.1. Governing equations of the flow problem in ALE form

The conservative form of the Euler equations is used in solving the flow problem. The paper is limited to inviscid flow problems since the primary focus is the development and verification of an unsteady adjoint method for the estimation of functional relevant temporal error. 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:

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

Eq. (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 values 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 at each time-step. Here A refers to the area or volume of the element, $\dot{\mathbf{x}}$ is the vector of 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.

The method presented in the paper for the estimation of global temporal error involves projections of the flow and adjoint solutions from a given temporal mesh onto a temporal mesh of higher resolution. In order to remove the dependence of the order-of-magnitude of the computed adjoint variables on the temporal mesh resolution we utilize the time-integrated form of Eq. (7) as

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

or in a discrete sense when considering a temporal element of size Δt as

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

2.2. Temporal discretization

The time derivative term is discretized using a second-order accurate BDF2 (Backward Difference Formula 2) scheme. Since the goal of the paper is to adapt the temporal mesh, the base scheme requires modification to handle non-uniform temporal mesh spacing. The modified BDF2 scheme for non-uniform temporal meshes is shown below.

$$\begin{aligned} &\frac{dA\mathbf{U}}{dt} = c_0 A^n U^n + c_1 A^{n-1} U^{n-1} + c_2 A^{n-2} U^{n-2} \\ &c_0 = \frac{2\Delta t_1 \Delta t_2 + \Delta t_2^2}{\Delta t_1 \Delta t_2 (\Delta t_1 + \Delta t_2)} \\ &c_1 = \frac{-(\Delta t_1 + \Delta t_2)^2}{\Delta t_1 \Delta t_2 (\Delta t_1 + \Delta t_2)} \\ &c_2 = \frac{\Delta t_1^2}{\Delta t_1 \Delta t_2 (\Delta t_1 + \Delta t_2)} \\ &\Delta t_1 = T^n - T^{n-1} \\ &\Delta t_2 = T^{n-1} - T^{n-2} \end{aligned} \tag{10}$$

Although not a sufficient test, substituting ($\Delta t = \Delta t_1 = \Delta t_2$) should verify that a standard second-order accurate backward difference can be recovered. The BDF2 discretization on uniformly spaced temporal meshes with mesh spacing Δt is shown below:

$$\frac{dA\mathbf{U}}{dt} = \frac{\frac{3}{2}A^n U^n - 2A^{n-1}U^{n-1} + \frac{1}{2}A^{n-2}U^{n-2}}{\Delta t} \tag{11}$$

2.3. Spatial discretization

The flow solver uses a cell-centered finite-volume formulation where the inviscid flux integral S around a closed control volume is discretized as

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

where B_e is the edge length, V_e is the normal edge velocity, $\mathbf{n}_e$ is the unit normal of the edge, and $F_e^{\perp}$ is the normal flux across the edge. The normal flux across the edge is computed using the second-order accurate matrix dissipation scheme [26] as the sum of a central difference and an artificial dissipation term as shown below,

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

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 element i as

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

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 unit edge normal vector components.

2.4. Mesh deformation strategy

Deformation of the mesh is achieved through the linear tension spring analogy [27,28] which approximates the mesh as a network of inter-connected springs. The spring coefficient is assumed to be inversely proportional to the edge length. Two independent force balance equations are formulated for each node based on displacements of neighbors. 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

$$[\mathbf{K}]\delta\mathbf{x}_{int} = \delta\mathbf{x}_{surf} \tag{16}$$

where $[\mathbf{K}]$ is the stiffness matrix assembled using the spring coefficients of each of the edges in the computational mesh. As in the case of the governing flow equations, the mesh motion and the mesh adjoint solutions have to be projected between temporal meshes of different resolutions, and in order to avoid projection related scaling issues we redefine the mesh motion equations in the form of a discrete time-integrated mesh residual as

$$\mathbf{G}(\mathbf{x}) = \Delta t\{[\mathbf{K}]\delta\mathbf{x}_{int} - \delta\mathbf{x}_{surf}\} = 0 \tag{17}$$

2.5. The discrete geometric conservation law (GCL)

The discrete geometric conservation law (GCL) requires that a uniform flow field be preserved when Eq. (7) is integrated in time. In other words the deformation of the computational mesh should not introduce conservation errors in the solution

of the flow problem. This translates into **U** = *constant* being an exact solution of Eq. (7). For a conservative scheme, the integral of the inviscid fluxes around a closed contour goes to zero when **U** = *constant*. Applying these conditions to Eq. (7) results in the mathematical description of the GCL as stated below.

$$\frac{\partial A}{\partial t} - \int_{dB(t)} \dot{\mathbf{x}} \cdot \mathbf{n}\, dB = 0 \tag{18}$$

For the second-order BDF2 time-integration scheme utilized in this work, this can be discretely represented using Eqs. (10) and (12) as:

$$(c_0 A^n + c_1 A^{n-1} + c_2 A^{n-2}) - \sum_{i=1}^{n_{edge}} V_{e_i} B_i = 0 \tag{19}$$

The edge velocities V_{e_i} for each of the edges encompassing the element must therefore be chosen such that Eq. (19) is satisfied. While various methods for computing the edge velocities satisfying the GCL have been developed for different temporal discretizations [29,30], a unifying approach applicable to first, second and third-order backwards differencing schemes (BDF) as well as higher-order accurate implicit Runge–Kutta (IRK) schemes has been developed in Ref. [31]. Following the procedure outlined in Ref. [31], the edge velocity $V^n_{e_i}$ multiplied by the edge length B_i at time-step n can be written as a linear combination of the area swept by the edge between three temporal locations n, $n-1$ and $n-2$ as:

$$V^n_{e_i} B_i = k_1(\Delta\Omega^n) + k_2(\Delta\Omega^{n-1}) \tag{20}$$

Here Ω^n refers to the area swept by the edge between time-steps n and $n-1$, while Ω^{n-1} is the area swept between $n-1$ and $n-2$. The coefficients k_1 and k_2 are functions of the time-step sizes Δt_1 and Δt_2 and are given by:

$$k_1 = \frac{2\Delta t_1 \Delta t_2 + \Delta t_2^2}{\Delta t_1 \Delta t_2 (\Delta t_1 + \Delta t_2)} \tag{21}$$

$$k_2 = \frac{-\Delta t_1^2}{\Delta t_1 \Delta t_2 (\Delta t_1 + \Delta t_2)} \tag{22}$$

3. Error formulation

3.1. Error due to temporal resolution

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, **U** is the unsteady flow solution set and **x** is the set of unsteady mesh solutions. The functional evaluated on two different temporal 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 temporal mesh and h is a fine resolution temporal mesh constructed by nested subdivision of the time-steps 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 $L_h\left(\mathbf{U}_h^H,\mathbf{x}_h^H\right)$, 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 time domain flow solution $\mathbf{U}_H$ and the coarse 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 time domain functional can be written as:

$$L_h(\mathbf{U}_h,\mathbf{x}_h) = L_h\left(\mathbf{U}_h^H,\mathbf{x}_h^H\right) + \left[\frac{\partial L}{\partial \mathbf{U}}\right]_{\mathbf{U}_h^H,\mathbf{x}_h^H}\left(\mathbf{U}_h - \mathbf{U}_h^H\right) + \left[\frac{\partial L}{\partial \mathbf{x}}\right]_{\mathbf{x}_h^H,\mathbf{U}_h^H}\left(\mathbf{x}_h - \mathbf{x}_h^H\right) + \cdots \tag{23}$$

If n_{steps} is defined as the number of time-steps in the temporal mesh h, and if n_{cells} and n_{nodes} are the number of elements and nodes in the spatial mesh, then $\mathbf{U}_h$, $\mathbf{U}_h^H$, $\mathbf{x}_h$ and $\mathbf{x}_h^H$ are vectors of size $[n_{steps} \times 1]$ where each element in the vectors are by themselves a vector of size $[n_{cells} \times 1]$ for the flow variables and a vector of size $[n_{nodes} \times 1]$ for the mesh variables. This indicates that $[\partial L/\partial \mathbf{U}]_{\mathbf{U}_h^H,\mathbf{x}_h^H}$ is a vector of size $[1 \times n_{steps}]$, where each element in the vector is again a vector of size $[1 \times n_{cells}]$. The same argument applies for the sensitivity of the functional to the mesh coordinates.

The procedure for computing an unsteady solution involves obtaining the solution to the non-linear residual operator $\mathbf{R}(\mathbf{U},\mathbf{x})$ at each time interval. The non-linear operator $\mathbf{R}(\mathbf{U},\mathbf{x})$ is constructed as described in earlier sections. When the time derivative term in the Euler equations is based on the second-order BDF2 discretization, the non-linear residual is defined as $\mathbf{R}^n = \mathbf{R}^n(\mathbf{U}^n,\mathbf{U}^{n-1},\mathbf{U}^{n-2},\mathbf{x}^n,\mathbf{x}^{n-1},\mathbf{x}^{n-2}) = 0$, where n refers to the time index. The solution of the non-linear system can then be obtained at each time interval using Newton iterations of the form:

$$\begin{aligned} &\left[\frac{\partial \mathbf{R}(\mathbf{U}^k,\mathbf{x})}{\partial \mathbf{U}^k}\right]\delta\mathbf{U}^k = -\mathbf{R}(\mathbf{U}^k,\mathbf{x}) \\ &\mathbf{U}^{k+1} = \mathbf{U}^k + \delta\mathbf{U}^k \\ &\delta\mathbf{U}^k \to 0, \quad \mathbf{U}^{k+1} = \mathbf{U}^n \end{aligned} \tag{24}$$

Expanding a fine time domain residual set about the coarse time domain set of residuals yields:

$$\mathbf{R}_h(\mathbf{U}_h,\mathbf{x}_h) = \mathbf{R}_h\left(\mathbf{U}_h^H,\mathbf{x}_h^H\right) + \left[\frac{\partial \mathbf{R}}{\partial \mathbf{U}}\right]_{\mathbf{U}_h^H,\mathbf{x}_h^H}\left(\mathbf{U}_h - \mathbf{U}_h^H\right) + \left[\frac{\partial \mathbf{R}}{\partial \mathbf{x}}\right]_{\mathbf{x}_h^H,\mathbf{U}_h^H}\left(\mathbf{x}_h - \mathbf{x}_h^H\right) + \cdots = 0 \tag{25}$$

and since the non-linear residual operator $\mathbf{R}$ must vanish for a converged solution at each time interval, an estimate for the error vector $\left(\mathbf{U}_h - \mathbf{U}_h^H\right)$ at each time interval in Eq. (23) can obtained by rearranging Eq. (25) as:

$$\left(\mathbf{U}_h - \mathbf{U}_h^H\right) \approx -\left[\frac{\partial \mathbf{R}}{\partial \mathbf{U}}\right]^{-1}_{\mathbf{U}_h^H,\mathbf{x}_h^H}\left\{\mathbf{R}_h\left(\mathbf{U}_h^H,\mathbf{x}_h^H\right) + \left[\frac{\partial \mathbf{R}}{\partial \mathbf{x}}\right]_{\mathbf{x}_h^H,\mathbf{U}_h^H}\left(\mathbf{x}_h - \mathbf{x}_h^H\right)\right\} \tag{26}$$

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 Eq. (26) into Eq. (23) results in:

$$L_h(\mathbf{U}_h,\mathbf{x}_h) \approx L_h\left(\mathbf{U}_h^H,\mathbf{x}_h^H\right) \underbrace{-\left[\frac{\partial L}{\partial \mathbf{U}}\right]_{\mathbf{U}_h^H,\mathbf{x}_h^H}\left[\frac{\partial \mathbf{R}}{\partial \mathbf{U}}\right]^{-1}_{\mathbf{U}_h^H,\mathbf{x}_h^H}\left\{\mathbf{R}_h\left(\mathbf{U}_h^H,\mathbf{x}_h^H\right) + \left[\frac{\partial \mathbf{R}}{\partial \mathbf{x}}\right]_{\mathbf{x}_h^H,\mathbf{U}_h^H}\left(\mathbf{x}_h - \mathbf{x}_h^H\right)\right\} + \left[\frac{\partial L}{\partial \mathbf{x}}\right]_{\mathbf{x}_h^H,\mathbf{U}_h^H}\left(\mathbf{x}_h - \mathbf{x}_h^H\right)}_{\varepsilon_{cc}} \tag{27}$$

The right-hand-side in Eq. (27) can be described as an estimate for the exact functional evaluated directly on the fine time domain. Based on the derivation it is approximately equal to the sum of the functional evaluated on the fine time domain using a projected solution and projected mesh coordinates from the coarse domain and a computable correction term ε_{cc}.

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:

$$\Lambda_{\mathbf{U}_h^T}\Big|_{\mathbf{U}_h^H,\mathbf{x}_h^H} = -\left[\frac{\partial L}{\partial \mathbf{U}}\right]_{\mathbf{U}_h^H,\mathbf{x}_h^H}\left[\frac{\partial \mathbf{R}}{\partial \mathbf{U}}\right]^{-1}_{\mathbf{U}_h^H,\mathbf{x}_h^H} \tag{28}$$

Transposing and rearranging Eq. (28) yields

$$\left[\frac{\partial \mathbf{R}_h}{\partial \mathbf{U}_h}\right]^T_{\mathbf{U}_h^H,\mathbf{x}_h^H}\Lambda_{\mathbf{U}_h}\Big|_{\mathbf{U}_h^H,\mathbf{x}_h^H} = -\left[\frac{\partial L}{\partial \mathbf{U}}\right]^T_{\mathbf{U}_h^H,\mathbf{x}_h^H} \tag{29}$$

The solution of Eq. (29) involves projecting the coarse time domain solution and mesh coordinates onto the fine time domain, reconstructing the flow Jacobian matrices on the fine time domain 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 time domain, an approximation is used where the adjoint is evaluated on the coarse time domain and then projected onto the fine domain. This circumvents the expensive evaluations on the fine time domain. Eq. (29) recast on the coarse time domain becomes:

$$\left[\frac{\partial \mathbf{R}}{\partial \mathbf{U}}\right]^T_{\mathbf{U}_H,\mathbf{x}_H}\Lambda_{\mathbf{U}_H} = -\left[\frac{\partial L}{\partial \mathbf{U}}\right]^T_{\mathbf{U}_H,\mathbf{x}_H} \tag{30}$$

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

$$\Lambda_{\mathbf{U}_h} = I_h^H \Lambda_{\mathbf{U}_H} \tag{31}$$

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 time domain and then evaluating a vector inner product as follows:

$$\varepsilon_{cc_1} = \Lambda_{\mathbf{U}_h^{H^T}} R_h\left(\mathbf{U}_h^H,\mathbf{x}_h^H\right) \tag{32}$$

ε_{cc_1} represents the contribution from the flow equations to the error arising due to insufficient temporal mesh resolution. The residual $\mathbf{R}_h\left(\mathbf{U}_h^H,\mathbf{x}_h^H\right)$ is non-zero since it is computed using the projection of the coarse time domain flow solution onto the fine time domain. Closer examination reveals that Eq. (32) is actually a summation of the vector inner products of the flow adjoint and the non-zero residual at each time-step on fine time domain. The remaining contributions to the total correction term may be written in combined form as:

$$\varepsilon_{cc_2} = \left\{\Lambda_{\mathbf{U}_h^T}\left[\frac{\partial \mathbf{R}_h}{\partial \mathbf{x}_h}\right]_{\mathbf{U}_h^H,\mathbf{x}_h^H} + \left[\frac{\partial L_h}{\partial \mathbf{x}_h}\right]_{\mathbf{x}_h^H}\right\}\left(\mathbf{x}_h - \mathbf{x}_h^H\right) = \lambda_{\mathbf{x}}\left(\mathbf{x}_h - \mathbf{x}_h^H\right) \tag{33}$$

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 time domain about its value constructed using projected mesh coordinates from the coarse to the fine domain as:

$$\mathbf{G}(\mathbf{x}_h) = \mathbf{G}(\mathbf{x}_h^H) + \left[\frac{\partial \mathbf{G}}{\partial \mathbf{x}}\right]_{\mathbf{x}_h^H} (\mathbf{x}_h - \mathbf{x}_h^H) + \cdots = 0 \tag{34}$$

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

$$(\mathbf{x}_h - \mathbf{x}_h^H) = -\left[\frac{\partial \mathbf{G}}{\partial \mathbf{x}}\right]^{-1}_{\mathbf{x}_h^H} \mathbf{G}(\mathbf{x}_h^H) \tag{35}$$

However, referring to Eq. (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 $[\mathbf{K}]$ multiplied by the local temporal element size Δt. Therefore,

$$(\mathbf{x}_h - \mathbf{x}_h^H) = -\left(\frac{1}{\Delta t}\right)[\mathbf{K}]^{-1}\mathbf{G}(\mathbf{x}_h^H) \tag{36}$$

Substituting this back into Eq. (33) gives us an expression for the second and final contribution to the computable correction term as:

$$\varepsilon_{cc_2} = -\lambda_{\mathbf{x}_h}\left(\frac{1}{\Delta t}\right)[\mathbf{K}]^{-1}\mathbf{G}(\mathbf{x}_h^H) \tag{37}$$

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

$$[\mathbf{K}]^T \Lambda_{\mathbf{x}_h} = -\left(\frac{1}{\Delta t}\right)\lambda_{\mathbf{x}_h^T} \tag{38}$$

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

$$[\mathbf{K}]^T \Lambda_{\mathbf{x}_H} = -\left(\frac{1}{\Delta t}\right)\lambda_{\mathbf{x}_h^T} \tag{39}$$

$$\Lambda_{\mathbf{x}_h} = I_h^H \Lambda_{\mathbf{x}_H} \tag{40}$$

Both Eqs. (31) and (40) represent linear systems and span the complete space and time domains on which the solution to the analysis problem was obtained. These equations can be solved using back substitution beginning at the final time-step and sweeping backward in time while iteratively solving for a flow and mesh adjoint variable at each time-step. Details on the exact solution process involving the backward sweep in time can be found in Refs. [32,24]. The final form for the second contribution to the correction term can now be expressed as:

$$\varepsilon_{cc_2} = \Lambda^{H^T}_{\mathbf{x}_h}\mathbf{G}(\mathbf{x}_h^H) \tag{41}$$

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, Eq. (41) represents a summation of vector products of the mesh adjoint and the mesh residual at each time interval. 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 examples are non-linear in time. This causes the mesh residual to be non-zero when computed on the fine time domain using projected mesh coordinates from the coarse time domain.

The contribution to the total correction from each individual time-step can be interpreted as the representation of the error in the functional arising from that time interval. From the derivation it is clear that there are two distinct sources of error, specifically the flow and the mesh that contribute to the total temporal resolution error. The resulting distribution in time of the total error ε_{cc} can thus be conveniently used as the criteria for identifying regions that require higher temporal resolution.

3.2. Error due to partial convergence

Consider the solution of the problem on the coarse time domain. Previously we assumed that the solutions on the coarse time domain were obtained by full convergence of the flow and mesh motion equations. This translates into the flow residual and the mesh residual equations evaluating to zero on the coarse time domain. However, we now consider the functional L evaluated on the coarse time domain using partially converged solutions. If the flow and mesh equations were partially converged on the coarse time domain, the error resulting in the functional as a consequence may be linearly approximated as:

$$L_H(\mathbf{U}_H, \mathbf{x}_H) - L_H(\overline{\mathbf{U}}_H, \bar{\mathbf{x}}_H) \approx \left[\frac{\partial L}{\partial \mathbf{U}}\right]_{\overline{\mathbf{U}}_H, \bar{\mathbf{x}}_H} (\mathbf{U}_H - \overline{\mathbf{U}}_H) + \left[\frac{\partial L}{\partial \mathbf{x}}\right]_{\bar{\mathbf{x}}_H, \overline{\mathbf{U}}_H} (\mathbf{x}_H - \bar{\mathbf{x}}_H) \tag{42}$$

where $\overline{\mathbf{U}}_H$ and $\bar{\mathbf{x}}_H$ refer to the approximate values of the flow variables and mesh coordinates obtained through partial convergence of the respective equations. Evaluating the flow and mesh residual equations using partially converged solutions would result in non-zero residuals. The fully converged zero residuals can then be written using a Taylor expansion about the partially converged values as:

$$\mathbf{R}(\mathbf{U}_H, \mathbf{x}_H) = \mathbf{R}(\overline{\mathbf{U}}_H, \bar{\mathbf{x}}_H) + \left[\frac{\partial \mathbf{R}}{\partial \overline{\mathbf{U}}}\right]_{\overline{\mathbf{U}}_H, \bar{\mathbf{x}}_H} (\mathbf{U}_H - \overline{\mathbf{U}}_H) + \left[\frac{\partial \mathbf{R}}{\partial \mathbf{x}}\right]_{\bar{\mathbf{x}}_H, \overline{\mathbf{U}}_H} (\mathbf{x}_H - \bar{\mathbf{x}}_H) + \cdots = 0 \tag{43}$$

$$\mathbf{G}(\mathbf{x}_H) = \mathbf{G}(\bar{\mathbf{x}}_H) + \left[\frac{\partial \mathbf{G}}{\partial \mathbf{x}}\right]_{\bar{\mathbf{x}}_H} (\mathbf{x}_H - \bar{\mathbf{x}}_H) + \cdots = 0 \tag{44}$$

At this point the similarity with the derivation of the temporal resolution error becomes clear. Following the same procedure as the temporal resolution error derivation, we can obtain two contributions that add up to the total error due to partial convergence as:

$$\varepsilon_{cc_{1p}} = \Lambda_{\mathbf{U}_H^T} \mathbf{R}(\overline{\mathbf{U}}_H, \bar{\mathbf{x}}_H) \tag{45}$$

$$\varepsilon_{cc_{2p}} = \Lambda_{\mathbf{x}_H^T} \mathbf{G}(\bar{\mathbf{x}}_H) \tag{46}$$

where the adjoint variables are solved for as:

$$\left[\frac{\partial \mathbf{R}}{\partial \overline{\mathbf{U}}}\right]^T_{\overline{\mathbf{U}}_H, \bar{\mathbf{x}}_H} \Lambda_{\mathbf{U}_H} = -\left[\frac{\partial L}{\partial \overline{\mathbf{U}}}\right]^T_{\overline{\mathbf{U}}_H, \bar{\mathbf{x}}_H} \tag{47}$$

$$[\mathbf{K}]^T \Lambda_{\mathbf{x}_H} = -\left(\frac{1}{\Delta t}\right) \lambda_{\mathbf{x}_H^T} \tag{48}$$

$$\lambda_{\mathbf{x}_H} = \left\{ \Lambda_{\mathbf{U}_H^T} \left[\frac{\partial \mathbf{R}}{\partial \mathbf{x}}\right]_{\overline{\mathbf{U}}_H, \bar{\mathbf{x}}_H} + \left[\frac{\partial L}{\partial \mathbf{x}}\right]_{\bar{\mathbf{x}}_H, \overline{\mathbf{U}}_H} \right\} \tag{49}$$

Closer examination reveals that these systems are nearly identical to Eqs. (31) and (40) with the only difference being that all quantities are evaluated using the partially converged flow and mesh solutions on the coarse time domain, rather than the fully converged values. In the case of the temporal resolution error, the adjoint variables were solved for on the coarse time domain using the fully converged coarse time domain flow solution and mesh coordinates. These were then projected onto the fine time domain and multiplied with the non-zero residuals constructed on the fine time domain. The residuals on the fine time domain were non-zero due to the projection of the flow solution and mesh coordinates from the coarse to the fine time domain. In the case of partial convergence on the coarse time domain, there are no projections involved and the error due to partial convergence on the coarse time domain can be estimated directly by multiplying the non-zero partially converged coarse residuals with the coarse adjoint variables.

Relaxing the assumption of fully converged coarse time domain solutions for the temporal resolution error does not affect the error estimated by that procedure. The important realization is that the error computed as temporal resolution error by relaxing this assumption includes the error due to the partial convergence of the flow and mesh equations on the coarse time domain. The temporal resolution error in reality is blind to what solution is projected from the coarse time domain, since our only enforcement in the derivation is that $\mathbf{R}_h\left(\mathbf{U}_h^H, \mathbf{x}_h^H\right) \neq 0$ which can also be enforced as $\mathbf{R}_h(\overline{\mathbf{U}}_h^H, \bar{\mathbf{x}}_h^H) \neq 0$. The error due to temporal resolution can be separated from the total error by subtraction of the error due to partial convergence which is computed on the coarse time domain.

The distributions in time of $\varepsilon_{cc_{1p}}$ and $\varepsilon_{cc_{2p}}$ arise from the error due to partial convergence of the flow and mesh equations, respectively, and can be used to drive adaptation of the convergence tolerances of the corresponding disciplines.

4. General implementation details

4.1. Solver details

The adaptive solver consists of two parts namely the forward flow solver and backward adjoint solver. The projection, error computation and adaptation routines are built into the adjoint solver. The flow solver performs the forward integration in time and writes out the flow solution, mesh solution and the partially converged flow and mesh residual values to the hard drive in a piecewise manner (i.e. at each time level). The adjoint solver reads in the flow and mesh solutions and performs the backward sweep in time to compute the flow and adjoint variables at each time level. All of the data exchange between the adjoint and flow solvers occurs through file I/O operations since the adjoint solver performs a backward sweep in time. It would be impractical to hold the unsteady solution set obtained by the flow solver in memory for use by the adjoint solver particularly for problems with very high temporal mesh resolutions. In our work, the cost of performing the backward sweep in the adjoint solver is generally similar to the cost of performing the forward integration in time to obtain the analysis solution. However, other researchers have noted that for large-scale unsteady viscous problems with turbulence models, the adjoint solution can in fact cost more than the analysis solution [33,34].

While the backward sweep occurs in the adjoint solver and the adjoint variable distribution in time becomes available, linear interpolation of the flow solution, mesh coordinates and the adjoint variables on the coarse time domain is used to determine unknown values on the fine time domain in order to compute the total error at each time-step due to projection between levels and partial convergence. The adjoint solver also reads in the non-zero partially converged flow and mesh residuals during the backward sweep and multiplies them with the corresponding adjoint variables to determine the algebraic partial convergence error at each time-step, which is then subtracted from the total error to determine the temporal resolution error. We use a refinement ratio of 2-to-1, where each temporal element in the coarse level temporal mesh is divided into two to construct the fine level temporal mesh.

Once the distributions of the error from various sources have been computed, their sum provides the functional corrections term. The adaptation routine is then invoked to analyze the error distribution and adapt the temporal resolution and the convergence tolerances. Details of the adaptation strategy are presented toward the end of this section. The adapted time-step distribution and adapted convergence tolerances are then written to disk for use by the flow solver. In our work we terminate the adaptation process once the correction to the functional reaches some prescribed error tolerance.

Computational cost is measured using the *systemclock* intrinsic function in FORTRAN95. As a convention, the cost shown in the comparative plots at any particular adaptation cycle regardless of the adaptation method employed includes the cost of all preceding adaptation cycles.

4.2. Effect of initial condition and change of order during startup

Multistep time-integration schemes such as BDF2 typically must be initiated with a lower-order scheme. In the case of second-order temporal accuracy, the first time-step is normally treated using a first-order accurate scheme. This is due to the unavailability of a stencil large enough to accommodate the BDF2 scheme at the first time-step. Since the adjoint computation is based simply on the linearization of the governing flow equations, this change in order during startup gets carried over to the adjoint solution process. In most circumstances, adjoint variables vary smoothly throughout the entire time domain irrespective of the temporal resolution. The exception to this is when there is a change in the order-of-accuracy of the temporal discretization between consecutive temporal elements. Since this does happen at the very first time-step for the BDF2 scheme, there is a discontinuity in the adjoint variable distribution at the first time-step. The method described in this paper relies on computing the adjoint variable on a coarse level temporal mesh and then projecting it onto a finer temporal mesh rather than directly computing the adjoint variable on the finer level mesh. Although the Taylor expansion requires that the adjoint variables be computed directly on the fine level temporal mesh, we assume that computing and projecting the adjoint variables from the coarser level will suffice for the purpose of error estimation. However, the discontinuity in the time variation of the adjoint variables due to change in discretization order occurs at different temporal locations in the coarse and fine meshes. In the case of 2-to-1 refinement between fine and coarse levels, the fine level discontinuity occurs at the midpoint of the first temporal element on the coarse level. Fig. 1 clearly shows the difference in the location of the discontinuity when comparing adjoint variables computed on the fine and coarse levels. The implication is that using a coarse level adjoint projected onto the fine level leads to significant inaccuracies in the predicted error. It is virtually impossible for a linear or any higher-order projection operator to shift the location of this spike when going from the coarse to the fine levels. As per theory the correct location for the spike is that captured when the adjoint variables are computed directly on the fine level.

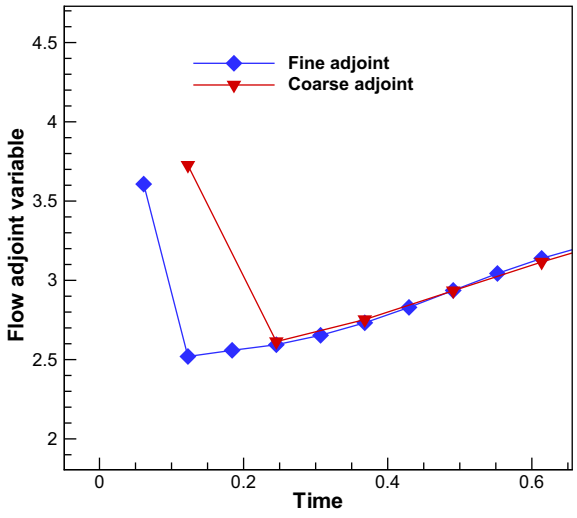


Fig. 1. Spike due to change in temporal order-of-accuracy (arbitrary spatial location).

Two different methods, namely the application of a second-order accurate initial condition and the direct solution of the fine level adjoint at only the start up location were investigated to remedy this problem. The second-order initial condition is analogous to a second-order boundary condition in space. A ghost temporal point is introduced with a spacing equal to that of the first time-step on the coarse level and the explicit first-order backward Euler formula is used to determine the state at this point. The purpose of this is to avoid a change in the order of the temporal discretization within the time domain of interest thus avoiding the spike altogether. Table 1 shows the inaccuracy in the predicted error between two levels due to this issue and the gains in accuracy using the two different methods proposed. The details of this particular test problem or not important, since the spike due to change in temporal discretization order occurs regardless of problem description. It is also evident from the table that both proposed methods perform reasonably well. For our work we compute the adjoint variable directly on the fine level at the first time-step. Although this is a solution on the fine level, the cost incurred in doing so corresponds exactly to that of two additional time-steps in the coarse level temporal mesh. As the temporal resolution increases, the cost of this additional computation becomes negligible.

4.3. Other shortcomings of projecting the adjoint variable

In the case of a first-order BDF1 time-integration scheme, the difference between coarse and fine level adjoint solutions is minimal allowing the interpolation of coarse level variables to capture the details of the fine level [32]. For higher-order accurate temporal discretizations, differences in the adjoint variable distribution between the two levels may exist which cannot be captured effectively by any interpolating operator. Fig. 2 shows the difference in the time variation of the fine level adjoint and the coarse level projected (third-order spline interpolant) adjoint for an arbitrary spatial point between temporal resolutions of 32 and 16. It should be noted that this problem has a tendency to surface mostly when dealing with instantaneous functionals. It is likely that this is due to the primary source term for the adjoint variables occurring only at the end of the time-integration process. The backward sweep in time appears to wash out the details in the adjoint distribution as it progresses farther from the location of the main source term (i.e. $n = n_{steps}$). This is not so apparent for time-integrated functionals where a functional linearization source term appears at each time level. Another factor contributing to this problem is the hyperbolic nature of time. It is not possible for elliptic interpolating operators such as splines to account for differences in coarse and fine level solution histories arising due to the hyperbolicity in time. While higher-order interpolation operators

Table 1
Treatment of error due to change in order of accuracy at startup.

		Ratio against exact error
Exact functional using 32 time-steps (BDF2)	0.0182487713384828	–
Exact functional using 16 time-steps (BDF2)	0.0182499683163204	–
Exact error between 32 and 16 time-steps	−1.1969778376e−06	–
Prediction using coarse level adjoint	−5.218147366e−07	0.43594
Prediction using coarse level adjoint and 2nd-order initial condition	−1.1694744767e−06	0.97702
Prediction using coarse level adjoint + fine level adjoint for first coarse time-step only	−1.212083697e−06	1.01262
Prediction using fine level adjoint	−1.1949466580e−06	0.99830

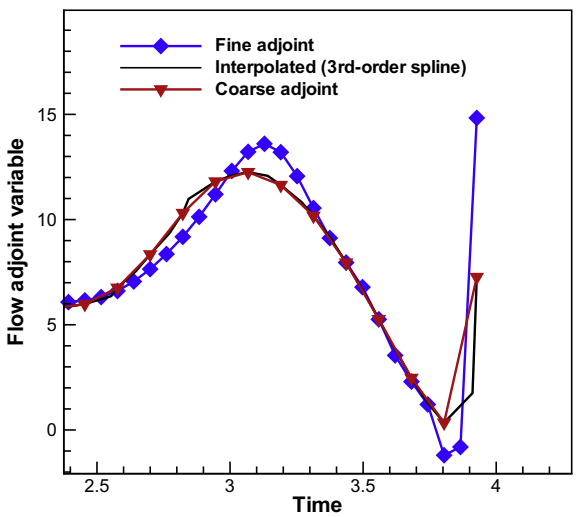


Fig. 2. Difference in the time variation of the fine level, coarse level, and projected adjoint variables (arbitrary spatial location).

have been shown to provide improved error estimates in spatial error problems, they do not offer any advantage over linear interpolation when used in the time domain. Two different methods were investigated to improve the accuracy of the projected adjoint variable as described below.

4.3.1. Utilizing fine level information in coarse adjoint solution

From a cost standpoint it is important that fine level solutions of the adjoint equations be avoided, but solving only the coarse level adjoint equations and projecting the adjoint variable to the fine level may lead to significant inaccuracies in the predicted error for several cases as described above. Accuracy gains while keeping costs similar to that of solving the coarse level adjoint can be achieved by attempting a solution of the fine level adjoint equations but only at points on the fine level temporal mesh that coincide with the coarse level mesh. The main difficulty is the construction of source terms for the adjoint equations from fine level temporal mesh points that do not coincide with those on the coarse level since no adjoint equations are solved for at these locations and corresponding adjoint variables are unavailable. This problem can be addressed by combining the projection operation with the adjoint solution process. The unknown adjoint variables on the fine level temporal mesh are written as functions of known adjoint variables located at points that coincide with the coarse level mesh and solved simultaneously. Fig. 3 illustrates solving the adjoint equations on the coarse level, on the fine level, and on the fine level but only at points coincident with the coarse level along with simultaneous interpolation. Both linear and third-order operators chosen to represent the unknown adjoint variables indicate improvements with the third-order spline operator fairing better. However, implementation of this method involves coding complexities that result in increased computational cost thus rendering it only marginally cheaper than solving the adjoint equations directly at all fine level temporal mesh points. This method does significantly reduce the difference between a coarse level adjoint projected to the fine level and the true fine level adjoint solution since it does not completely ignore fine level information during the solution process. Fig. 4 indicates significant improvement at least in the qualitative sense.

4.3.2. Direct smoothing of error in adjoint variable on fine level

While fine level solutions are to be avoided, the projection of the coarse level adjoint variable onto the fine level temporal mesh provides a good initial guess for the fine level adjoint equations. If the error between the coarse and fine level adjoint solutions are in the high frequency range in the context of the fine level temporal mesh, solution of the fine level adjoint equations starting with the coarse level estimates should in theory converge very rapidly using conventional smoothing techniques such as Gauss–Seidel iterations. Although this is an increase in the overall cost of the process and is actually a partial solution directly on the fine level temporal mesh, the gain in accuracy somewhat justifies the increase in cost. Typically we have noticed anywhere between 20% and 30% increase in cost compared to solving only the coarse level equations.

In our work we employ direct fine level smoothing after solving for the adjoint variable on the coarse level and linearly interpolating to estimate an initial guess for the fine level equations. Also, fine level smoothing is performed regardless of whether the functional is time-integrated or instantaneous.

4.4. Validation of method

The validation of the method is done in several steps where each component of the computed total error is individually verified. The process can be broken down as follows:

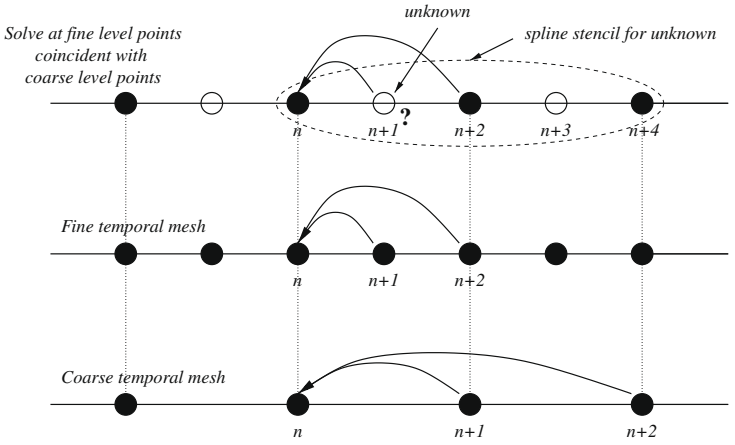


Fig. 3. Illustration of solving the adjoint equations on the coarse level, the fine level, and on the fine level but only at points coincident with coarse level points. The time index where the adjoint equations are being solved for is n and arrows indicate required source terms.

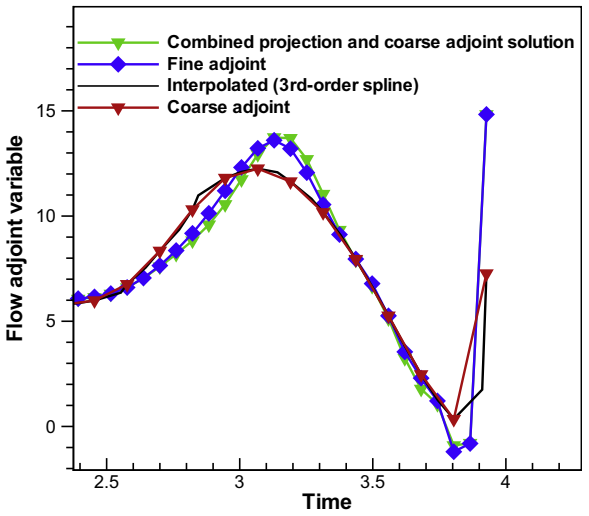


Fig. 4. Simultaneous projection and solution of coarse level adjoint variable (arbitrary spatial location).

1. Validate temporal resolution error without the effect of partially converged solutions.
2. Validate error due to partially converged flow equations without the effect of partially converged mesh equations or temporal resolution.
3. Similarly validate error due to partially converged mesh equations.
4. Validate total combined error which includes temporal resolution and partial convergence of flow and mesh equations.

4.4.1. Temporal resolution error

Since the method computes a linear approximation of the error between temporal meshes successively refined using a ratio of 2-to-1, it would be wise to choose a test problem that exhibits minimal non-linear behavior with respect to increasing temporal resolution for the purpose of validation. Our approach is to consider a sinusoidally pitching NACA64A010 airfoil and zoom into a small temporal window in the pitch cycle to use as our time domain of interest. The purpose of this is to reduce the effect of non-linear behavior as the temporal resolution is increased. The conditions chosen for the problem include a freestream Mach number of 0.8, a zero mean angle-of-attack, an amplitude of pitch of 5°, a reduced frequency of 0.1 and a pitch center located two chord lengths ahead of the leading edge of the airfoil. The chosen conditions produce oscillating shockwaves on the upper and lower surfaces of the airfoil. The time domain we consider is the first 16th of the first pitch period starting from a steady-state solution at the mean angle-of-attack of the airfoil. The functional of interest is the time-integrated lift coefficient given as

$$L = \sum_{n=1}^{n_{steps}} \Delta t_n C_{Ln} \tag{50}$$

The coarsest temporal mesh consists of 8 temporal elements of uniform size and fine meshes are constructed by successive nested subdivision of these 8 elements. The finest mesh consists of 1024 elements. In order to remove the effect of partially converged solutions and to validate only the error due to temporal resolution, all of the equations (governing and corresponding adjoint) are converged to machine precision. At each temporal resolution we compare against the functional computed directly on the next temporal mesh that has twice the temporal resolution. As an example, consider a coarse mesh resolution H of 8 elements and the corresponding fine mesh resolution h of 16 elements. The solution to the analysis problem is first obtained on H and then projected onto h using linear interpolation. The functional $L_h\left(\mathbf{U}_h^H, \mathbf{x}_h^H\right)$ is then evaluated on the fine mesh h using the projected solution. The exact error in the functional as a consequence of evaluating it on h using a projected solution from H is defined as the difference with respect to the functional evaluated using a solution obtained directly h. The goal is to use the adjoint to estimate as closely as possible this exact error. The adjoint equations are solved on the coarse mesh H, then projected onto h, after which a few Gauss–Seidel iterations on the fine mesh are employed to improve the accuracy of computed adjoint variable. Finally, the inner product between the adjoint variable and the non-zero analysis residual on the fine level (arising due to projection) forms the predicted error. We define the error in the error as the difference between the exact error as defined above and the adjoint-based error prediction. It is expected for this quantity to asymptotically approach zero as the temporal resolution increases since a linear approximation (adjoint-based) to the exact error performs better in the asymptotic range. It should be noted that the rate at which the error in the error reduces will not be higher than first-order even for higher-order spatial or temporal discretizations of the governing equations. There is no relationship between the discretization accuracy and the measure of error defined here, which is more an estimate of interpolation accuracy. It may be intuitive to expect higher rate of reduction of the error in the error for higher-order interpola-

tion operators, but as has been discussed earlier, this has not been observed to be the case in the time domain. Table 2 shows the results from this study and indicates the expected trends. The relative error in the error shown in the table is with respect to the exact error. Fig. 5 is plot of the data in the table.

4.4.2. Flow partial convergence error

In order to validate the error due only to partial convergence of the flow, we fully converge the mesh motion equations and partially converge the flow equations over a range of convergence tolerances while performing all computations on a single temporal mesh of some arbitrarily chosen resolution. This is done in order to remove the effect of mesh partial convergence and temporal resolution. The chosen temporal mesh resolution consists of 16 uniform temporal elements. The convergence tolerances for the flow equations range from 1e−4 to 1e−13 in decrements of one order-of-magnitude. The comparison is made against the functional computed on the same temporal mesh by fully converging both the flow and mesh equations. Full convergence refers to a tolerance of 1e−14 for the flow equations and 1e−15 for the mesh equations. Table 3 shows the results from the study while Fig. 6 is a plot of the data in the table. For very loose convergence tolerances the predicted error shows significant inaccuracies due to non-linearities in the analysis problem. The predicted error improves greatly as the convergence tolerance is tightened, but begins to stall beyond 1e−10 due to machine precision related issues. There is a band of convergence tolerances within which the adjoint-based error prediction shows excellent agreement with the exact error. It should be noted that the residual for the flow equations are not normalized and refers to the L_2 norm of the non-linear residual $\mathbf{R}(\mathbf{U},\mathbf{x})$ defined earlier. The flow convergence tolerance at which the non-linear Newton solver is terminated refers to this measure of the non-linear residual. There are no projections between temporal meshes of different resolutions when validating algebraic error, and the error reduction rate should closely follow the accuracy of the discretization. This trend is observed in the presented data.

4.4.3. Mesh partial convergence error

This method is nearly identical to that of validating the flow partial convergence error, except now the flow equations are fully converged (1e−14) while the mesh equations are partially converged over a range of tolerances from 1e−3 to 1e−14. Comparisons are made against the functional evaluated using a fully converged solution for the flow and mesh equations

Table 2
Validation of temporal resolution error prediction.

H/h	$L_h(\mathbf{U}_h,\mathbf{x}_h)$	$L_h\left(\mathbf{U}_h^H,\mathbf{x}_h^H\right)$	Exact error $L_h(\mathbf{U}_h,\mathbf{x}_h)-L_h\left(\mathbf{U}_h^H,\mathbf{x}_h^H\right)$	Predicted error $\Lambda_h^{H^T}\mathbf{R}_h\left(\mathbf{U}_h^H,\mathbf{x}_h^H\right)$	Error in error	Relative (%)
8/16	0.569396322575267	0.560169315190123	9.227007385144e−3	9.178713487305005e−3	4.8293897839032e−5	0.52
16/32	0.562025471736337	0.556876350433573	5.149121302764e−3	5.130824481983147e−3	1.8296820781312e−5	0.36
32/64	0.558558547353816	0.555768236598189	2.790310755627e−3	2.783289991186009e−3	7.020764441403e−6	0.25
64/128	0.556916008012822	0.555430655044149	1.485352968673e−3	1.482292641578744e−3	3.060327093949e−6	0.21
128/256	0.556126469068276	0.555352069994796	7.7439907348e−4	7.729714607539632e−4	1.4276127261289e−6	0.18
256/512	0.555710827580821	0.555344499890371	3.6632769045e−4	3.657852929566094e−4	5.423974938093e−7	0.15
512/1024	0.555465579355258	0.555319845746303	1.45733608955e−4	1.456522968010164e−4	8.13121535183e−8	0.06

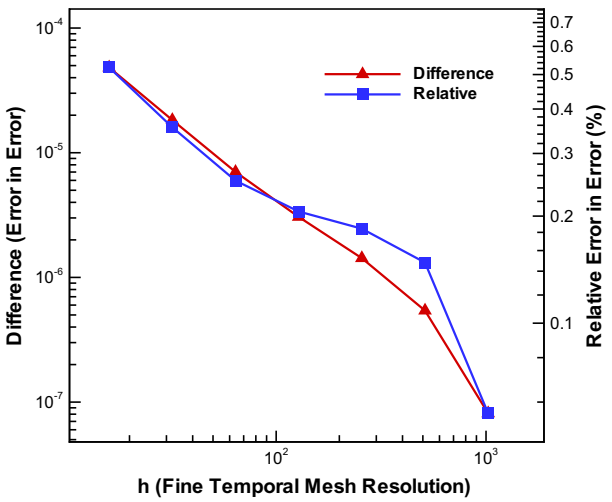


Fig. 5. Error convergence in the validation of predicted temporal resolution error.

Table 3
Validation of flow partial convergence error prediction. The functional based on full convergence against which comparisons are made is 0.569396322575267.

Flow Tol.	$L_H(\overline{\mathbf{U}}_\mathbf{H}, \mathbf{x}_\mathbf{H})$	Exact error $L_H(\mathbf{U}_H, \mathbf{x}_H) - L_H(\overline{\mathbf{U}}_\mathbf{H}, \mathbf{x}_\mathbf{H})$	Predicted error $\Lambda_H^T \mathbf{R}_H(\overline{\mathbf{U}}_\mathbf{H}, \mathbf{x}_\mathbf{H})$	Error in error	Relative (%)
1e−4	2.9124800193341 27e−3	0.566483842555933	0.611018567593253	−4.453472503731970e−2	7.8616
1e−5	0.601413576272080	−3.201725369681307e−2	−3.196448033169741e−2	−5.277336511566588e−5	0.1648
1e−6	0.569978715518064	−5.823929427970498e−4	−5.832683469617397e−4	8.754041646898960e−7	0.1503
1e−7	0.569466741506641	−7.041893137404998e−5	−7.037842432530698e−5	−4.050704874299191e−8	0.0575
1e−8	0.569397608253941	−1.285678674078916e−6	−1.286266403071332e−6	5.877289924155624e−10	0.0457
1e−9	0.569395345108406	9.774668610074144e−7	9.774649176494041e−7	1.943358010287938e−12	0.0002
1e−10	0.569396315623312	6.951954900635826e−9	6.943693310523341e−9	8.261590112484963e−12	0.1188
1e−11	0.569396334871903	−1.229663604274123e−8	−1.230491020053879e−8	8.274157797561656e−12	0.0673
1e−12	0.569396323198692	−6.234250893299986e−10	−6.317086900752890e−10	8.283600745290359e−12	1.3287
1e−13	0.569396322522541	5.272593472938070e−11	4.444156015344455e−11	8.284374575936148e−12	15.7121

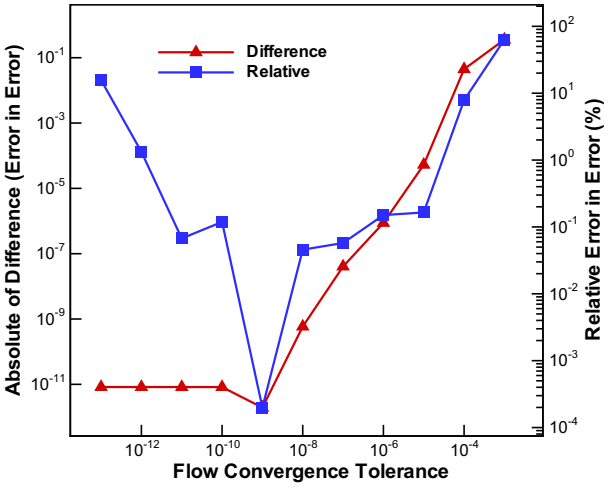


Fig. 6. Error convergence in the validation of predicted flow partial convergence error.

with 1e−15 used as the tolerance for the mesh. Table 4 shows the results from the study and Fig. 7 is the corresponding plot of the data. Results are similar to the flow partial convergence validation study and indicates a small range of tolerances where error estimates are very accurate. Although it may seem that error estimates should be accurate in the loose tolerance range due to the linear nature of the mesh motion equations, this is not true since the non-linear flow equations which are dependent on the mesh solution also have to be solved in the process.

4.4.4. Combined total error due to resolution and partial convergence

In this study we successively refine the temporal resolution and tighten the convergence tolerances from loosely set values in order to validate the total predicted error. The coarsest temporal mesh consists of 8 equally sized temporal elements while the finest mesh consists of 1024 elements. The convergence tolerances for the flow equations and mesh equations be-

Table 4
Validation of mesh partial convergence error prediction. The functional based on full convergence against which comparisons are made is 0.569396322575267.

Mesh Tol.	$L_H(\mathbf{U}_H, \bar{\mathbf{x}}_H)$	Exact error $L_H(\mathbf{U}_H, \mathbf{x}_H) - L_H(\mathbf{U}_H, \bar{\mathbf{x}}_H)$	Predicted error $\Lambda_H^T \mathbf{R}_H(\mathbf{U}_H, \bar{\mathbf{x}}_H)$	Error in error	Relative (%)
1e−3	0.569379509842224	1.681273304299236e−5	−2.286832036745351e−6	1.909956507973771e−5	86.3982
1e−4	0.569392611900076	3.710675191004320e−6	2.944203825426592e−6	7.664713655777281e−7	20.6558
1e−5	0.569395930641548	3.919337189239869e−7	3.806557237014826e−7	1.127799522250424e−8	2.8775
1e−6	0.569396283431473	3.914379398395340e−8	3.898434192762393e−8	1.594520563294712e−10	0.4073
1e−7	0.569396318629529	3.945737958588325e−9	3.944071544193236e−9	1.666414395088515e−12	0.0422
1e−8	0.569396322177531	3.977359552820303e−10	3.977202472758628e−10	1.570800616755659e−14	0.0039
1e−9	0.569396322535569	3.969791162461433e−11	3.969800092969266e−11	−8.930507832981892e−17	0.0002
1e−10	0.569396322571281	3.985922703009237e−12	3.984606300974339e−12	1.316402034898012e−15	0.0330
1e−11	0.569396322574870	3.969047313034935e−13	3.965971027703913e−13	3.076285331021767e−16	0.0775
1e−12	0.569396322575227	3.996802888650564e−14	3.975520234663100e−14	2.128265398746332e−16	0.5325
1e−13	0.569396322575263	3.996802888650564e−15	3.952727164980954e−15	4.407572366960945e−17	1.1028
1e−14	0.569396322575266	9.992007221626409e−16	3.962733380202208e−16	6.029273841424201e−16	60.3410

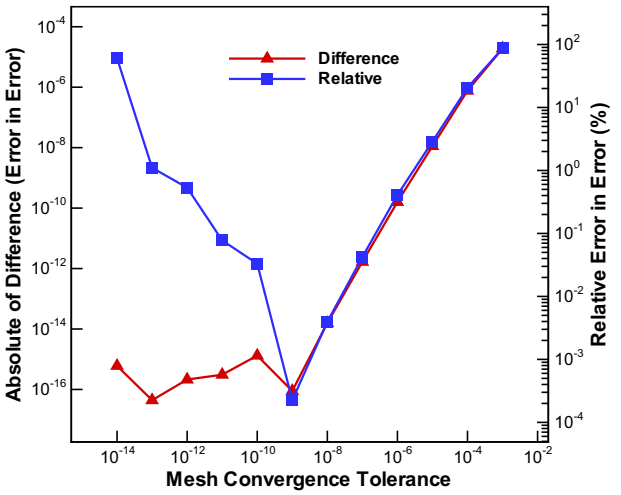


Fig. 7. Error convergence in the validation of predicted mesh partial convergence error.

gin at 1e−5 and 1e−4, respectively, on the coarsest temporal mesh and are tightened by an order-of-magnitude during each refinement of the temporal resolution. These starting limits were chosen such that the algebraic and resolution errors are roughly the same order-of-magnitude and do not mask any discrepancies from either. The comparison at each temporal resolution is made against the functional computed using full convergence of the flow (1e−14) and mesh (1e−15) equations on the next temporal mesh of double the resolution. It is important that the method be validated for prediction of total combined error since it is to be applied in this form to adaptation problems in general. Table 5 shows the results from this study and indicates the expected trend of more accurate predictions as the resolution is increased while simultaneously convergence tolerances are tightened. Fig. 8 is a plot of the data presented in the table.

4.5. Adaptation strategy

The adjoint-based adaptation method involves the forward integration in time to determine the solution to the analysis problem and a backward sweep in time to compute the time-dependent adjoint variables and the corresponding necessary error distributions (resolution and algebraic). The sum of the individual error distributions provides an estimate of the correction from each type of error. The temporal elements are first sorted in decreasing order by their contribution to the total error of each type. Then parsing down each of the lists, elements are flagged for refinement or tolerance tightening until 99% of the total error of the type under consideration is reached. The method assures that only the most offending elements are targeted in the case of an extremely non-uniform error distribution. The method also assures that near uniform refinement or tolerance tightening is requested once the error in the temporal domain has been equidistributed. Elements flagged for refinement are subdivided into two, while elements flagged for algebraic error have their convergence tolerances tightened by a factor of three. This factor for the tightening of the convergence tolerances was chosen to permit at least 7 or 8 adaptation cycles without the tolerances becoming smaller than 1e−10 given the starting tolerances in the example problems.

4.6. Comparative methods

We present two example problems, one where the functional of interest is an instantaneous quantity evaluated at the end of the time-integration process and the other where the functional is a time-integrated quantity. The purpose of the exam-

Table 5
Validation of combined temporal resolution and partial convergence error prediction.

Flow/Mesh Tol.:H/h	$L_h(\mathbf{U}_h,\mathbf{x}_h)$	$L_h(\bar{\mathbf{U}}^{\mathbf{H}}_{\mathbf{h}},\bar{\mathbf{x}}^{\mathbf{H}}_{\mathbf{h}})$	Exact error $L_h(\mathbf{U}_h,\mathbf{x}_h)-L_h(\bar{\mathbf{U}}^{\mathbf{H}}_{\mathbf{h}},\bar{\mathbf{x}}^{\mathbf{H}}_{\mathbf{h}})$	Predicted error $\Lambda_h^{HT}\mathbf{R}_h(\bar{\mathbf{U}}^H_h,\bar{\mathbf{x}}_H)$	Error in error	Relative (%)
1e−5/1e−4:8/16	0.569396322575267	0.569051773422440	3.44549152827e−4	2.856116483680940e−4	5.8937504458906e−5	17.12
1e−6/1e−5:16/32	0.562025471736337	0.557461052659754	4.56441907658e−3	4.545668447896880e−3	1.875062868612e−5	0.41
1e−7/1e−6:32/64	0.558558547353816	0.555927382132259	2.63116522156e−3	2.624206889915153e−3	6.958331641847e−6	0.26
1e−8/1e−7:64/128	0.556916008012822	0.555479279653642	1.43672835918e−3	1.433676569201321e−3	3.051789978679e−6	0.21
1e−9/1e−8:128/256	0.556126469068276	0.555365326064910	7.61143003366e−4	7.597161855665300e−4	1.42681779947e−6	0.19
1e−10/1e−9:256/512	0.555710827580821	0.555348173956032	3.62653624789e−4	3.621113447427561e−4	5.422800462439e−7	0.15
1e−11/1e−10:512/1024	0.555465579355258	0.555320771985109	1.44807370149e−4	1.447260123838227e−4	8.13577651773e−8	0.06

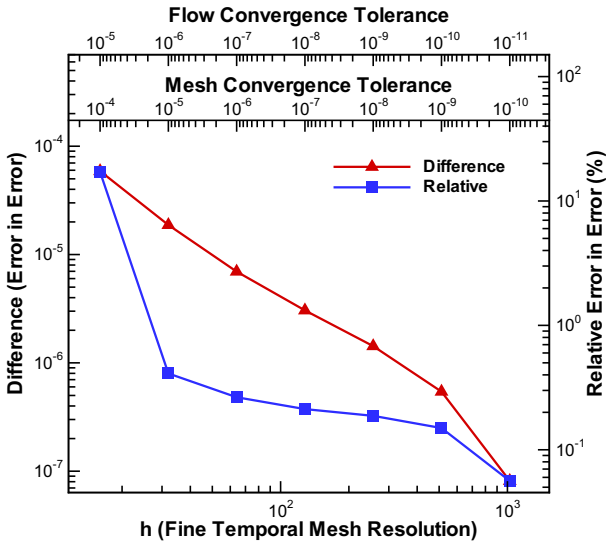


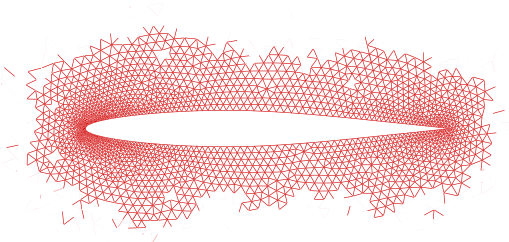
Fig. 8. Error convergence in the validation of predicted total combined error.

ples presented is to demonstrate the advantage of using goal/adjoint-based adaptation of the temporal domain over traditional adaptation methods particularly when there is little correlation between local error and functionals of interest. To this end we compare our results against the more commonly used but basic method of uniform temporal refinement, and the more sophisticated method of utilizing temporal discretizations of different orders-of-accuracy for estimating local temporal error.

The uniform refinement method we employ involves the doubling of the temporal mesh resolution at each adaptation cycle using a 2-to-1 nested subdivision of temporal elements, while simultaneously tightening the convergence tolerances by a factor of three for all temporal elements. As for the estimation of local temporal error, the method proposed in Ref. [35] is used. The solution to the analysis problem obtained using the second-order accurate BDF2 scheme is used to evaluate the temporal derivative term discretized based on a third-order accurate BDF3 time-integration scheme. The estimate of the local error at each time-step is then computed as:

$$e_{local} = \left| \left[\frac{dA\mathbf{U}}{dt} \right]_{BDF3} - \left[\frac{dA\mathbf{U}}{dt} \right]_{BDF2} \right| \tag{51}$$

The adaptation for this method is based on refining temporal elements that have local error higher than the computed mean of the local error distribution in the time domain. The convergence tolerances at each time-step are set to be equal to that of

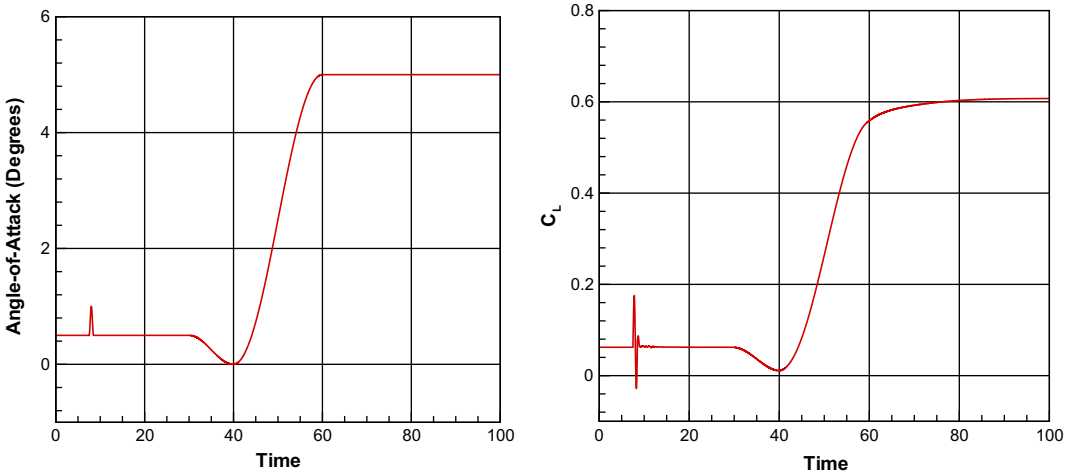


the local temporal error in order to ensure that the algebraic error is less than or equal to the temporal discretization error. All adaptation methods are compared in reference to numerically exact results computed using a uniform time domain of resolution at least two orders-of-magnitude higher than those achieved through adaptation.

5. Results

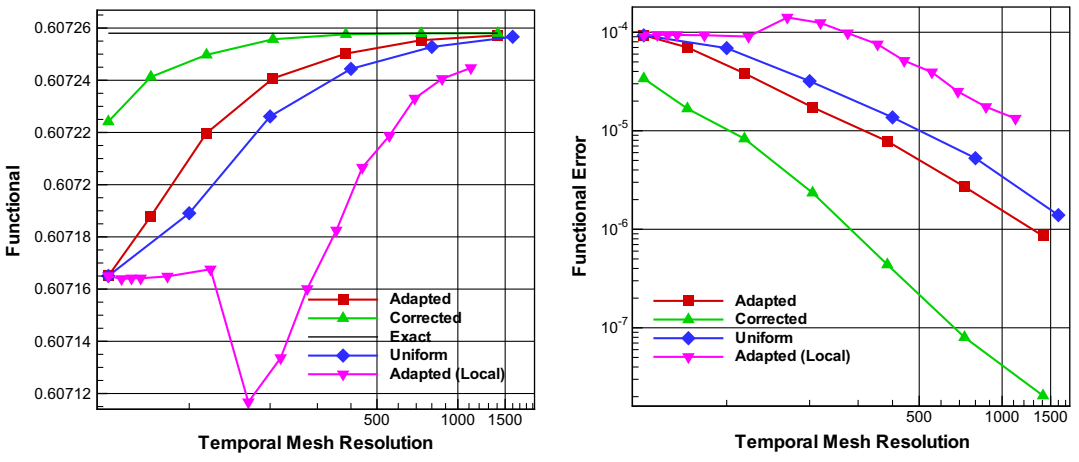
5.1. Case (A): Example of an instantaneous functional

The first example consists of a pitching NACA64A010 airfoil where the angle-of-attack as a function of time is prescribed. The functional of interest is the lift coefficient of the airfoil evaluated at the end of the time-integration process. The computational mesh for this case consists of approximately 6600 elements and is shown in Fig. 9. The airfoil operates at a Mach number of 0.3 and Fig. 10(a) shows the prescribed displacement in time. The corresponding time variation of the lift coefficient of the airfoil is shown in Fig. 10(b). The prescribed motion begins at an angle-of-attack of 0.5° and remains undisturbed for a short period before a spike (albeit continuous) of 0.5° is introduced. The angle-of-attack then returns to the steady-state value of 0.5° where it remains for an extended period before a shallow pitch down by 0.5° is followed by a rapid pitch up to 5°. Once the angle-of-attack reaches the maximum of 5°, the motion of the airfoil is stopped and only transient effects remain in the flow. The error tolerance used as the termination criteria for adaptation was set as 1e−6. While this is a remarkably tight tolerance for lift coefficient, it becomes necessary due to the dominant spatial discretization errors arising



(a) Prescribed variation of airfoil angle-of-attack (b) Time varying lift coefficient for prescribed displacement

Fig. 10. Prescribed displacement and corresponding variation of lift coefficient in time of the airfoil for Case (A).

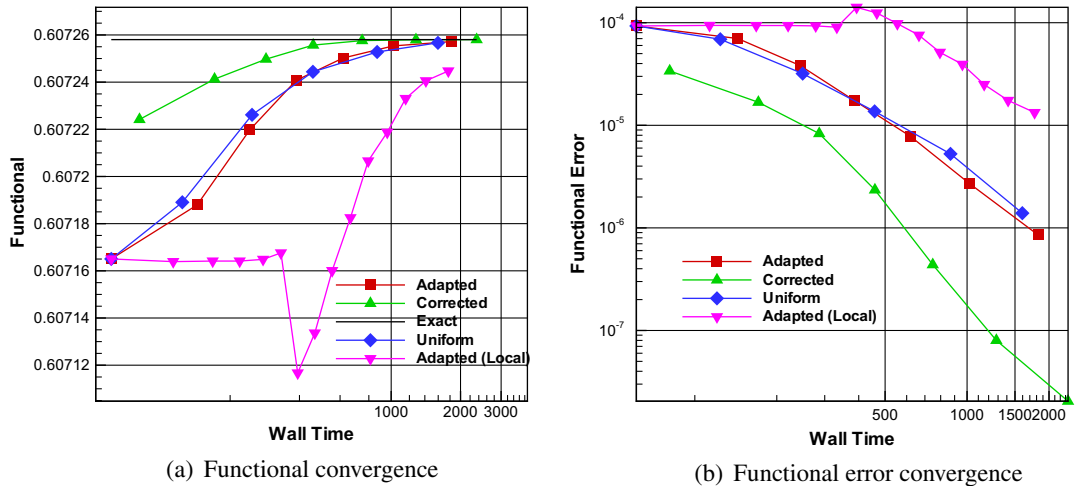


(a) Functional convergence (b) Functional error convergence

Fig. 11. Convergence of functional and error with respect to temporal mesh resolution for Case (A).

from computational mesh coarseness. The problem is carefully chosen to demonstrate the advantage in using goal-based adaptation over local error-based adaptation when there is no correlation between local error and the output functional of interest. Fig. 11(a) and (b) show the convergence of the functional and error in the functional using the three different adaptation methods. The plots indicate that adjoint based adaptation is able to achieve similar error levels to that of unguided uniform refinement with fewer time-steps. On average the method appears to provide a factor of saving anywhere between 1.6 and 2 when considering only adaptation. The method however also provides a correction term as shown in Eq. (27) that can be used to improve the value of the functional computed using the adapted domain. When including the correction term in the functional, the method far outperforms uniform refinement at least in terms of the temporal resolution required for the same error levels. Fig. 12(a) and (b) show the convergence with respect to computational cost. When considering the adjoint-based adapted curve in the plots, the costs appear similar to that of uniform refinement for the same error levels. However, the method is still competitive when the correction term is included in the functional. It should be noted that the sum of the adjoint-based error distributions provides the correction term and no additional work is necessary in order to compute it. Therefore, a fair comparison is that between the corrected functional convergence and uniform refinement.

It is interesting to note from the plots that local error-based adaptation performs very poorly for this particular problem. This is expected since the problem was setup specifically to demonstrate the weakness of local error-based adaptation in such situations. Theoretically it is known that the spike occurring close to the start of the time-integration process has no effect on the lift coefficient at the end of the time-integration since there is sufficient time after the spike for the airfoil to recover its steady-state lift. Local error-based adaptation methods however are blind to this effect and will target the spike



(a) Functional convergence (b) Functional error convergence

Fig. 12. Convergence of functional and error with respect cost in terms of wall time for Case (A).

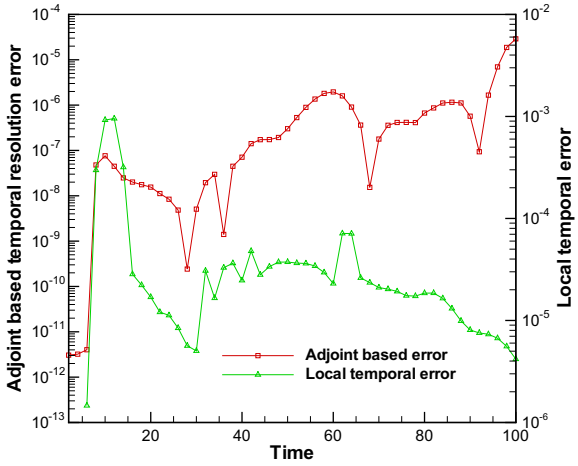


Fig. 13. Comparison of adjoint-based error distribution and local temporal error distribution for Case (A) on the coarsest temporal mesh.

for adaptation as that would be the location of maximum temporal discretization error. This is quite obvious from Fig. 13 where the distribution of local error shows significant differences compared to the adjoint-based error, both evaluated at the first adaptation cycle. Nearly all of the refinement during the first six adaptation cycles based on local temporal error occurs at the location of the spike. It is only after the local temporal error at this location has been brought down to the same levels as those closer to the end of the time-integration does the method actually start attacking the correct regions relevant to the functional. As a consequence of this, local error-based adaptation performs less efficiently than even uniform refinement of the time domain. Fig. 14 compares the distribution of the time-step sizes between adjoint adaptation and local error-based adaptation after the final adaptation cycle. It is clear from this plot that local error-based adaptation has refined the temporal mesh significantly in the vicinity of the spike, while the adjoint-based adaptation has almost completely ignored the spike.

The other aspect of the problem that is tackled by the adaptation procedure is the algebraic error at each time-step due to partial convergence. Figs. 15 and 16 show the distribution of the error due to partial convergence of the flow and mesh equations at start and termination of adaptation. The algebraic error is dominant when the airfoil is moving and also close to when the functional is evaluated. As in the case of temporal resolution, the temporal location where the spike in the angle-of-attack occurs is irrelevant to the functional. Fig. 17 shows the final distribution of the convergence tolerances for the flow and mesh equations after termination of the adaptation process. The tolerances progressively become tighter for both sets of equations as the time-integration approaches the point of the functional evaluation and this trend matches closely with intuition.

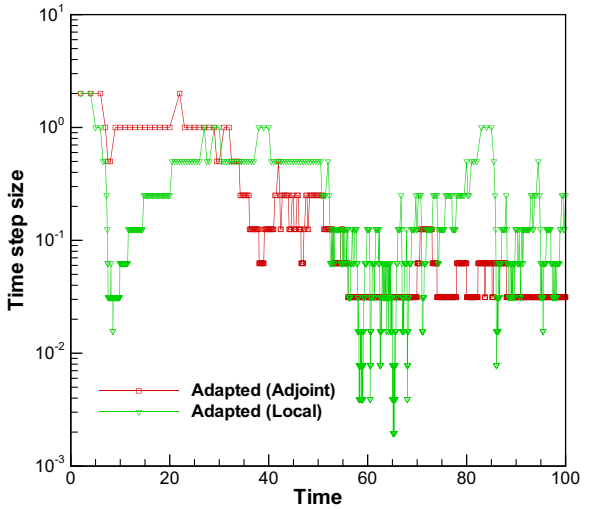


Fig. 14. Comparison of time-step size distribution for adjoint-based and local error-based adaptation after final adaptation cycle for Case (A).

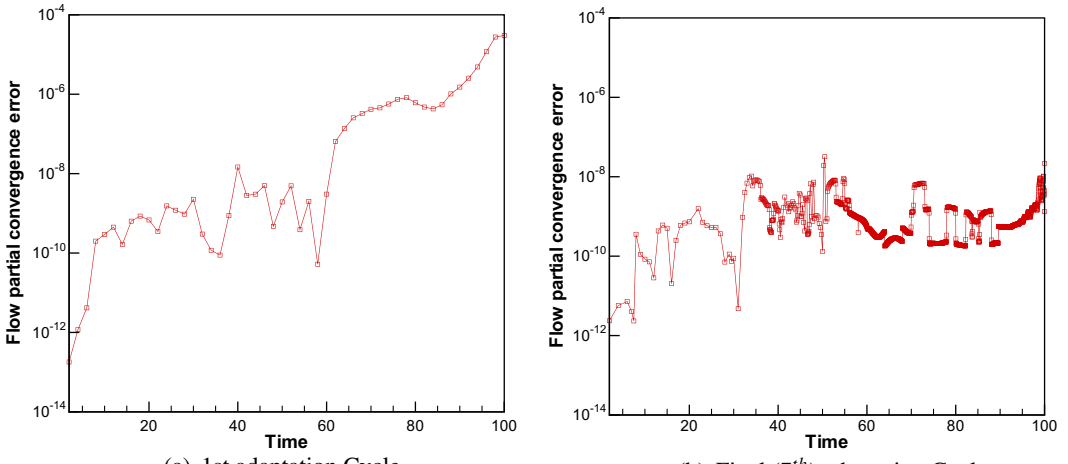
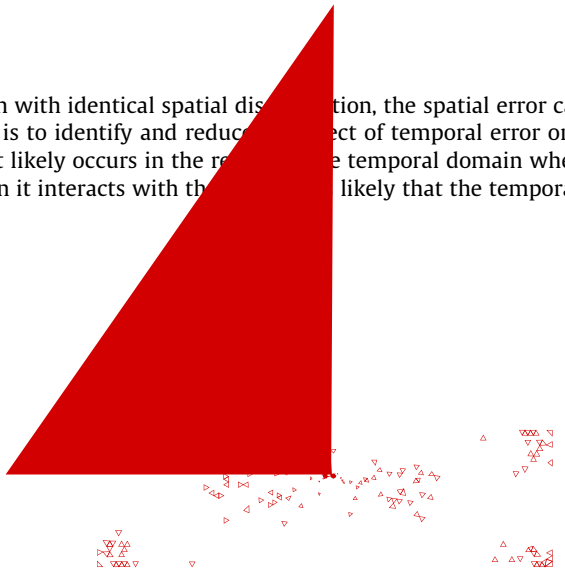
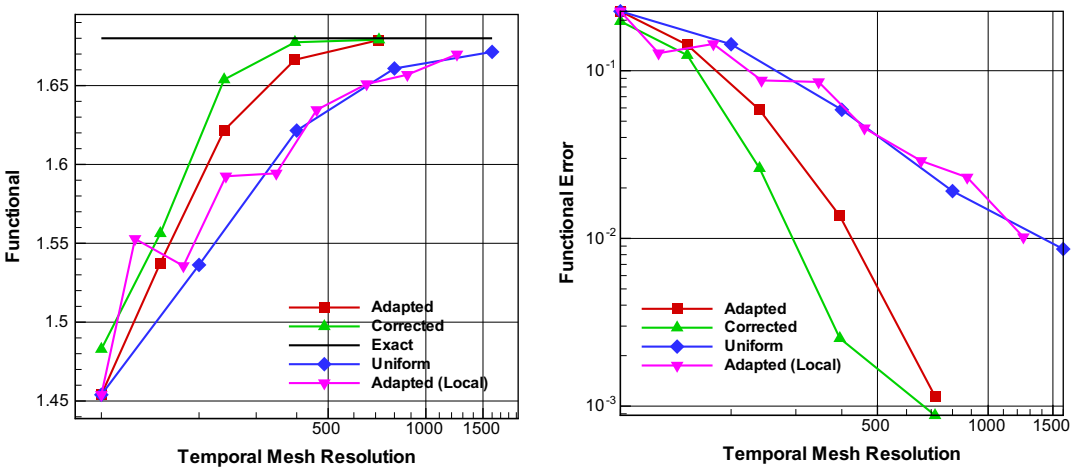


Fig. 15. Flow partial convergence error distribution for Case (A).

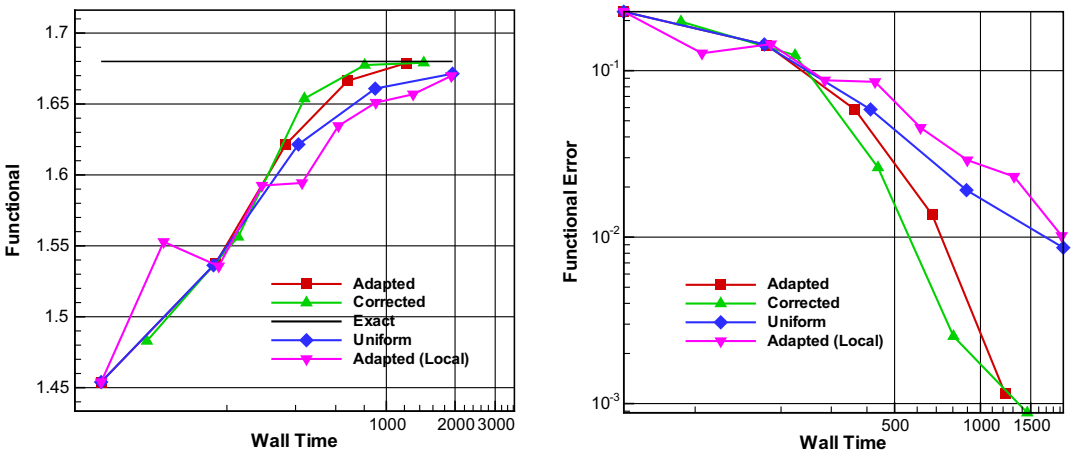
mesh with identical spatial dis tion, the spatial error can be considered to be independent of the temporal error. Our goal is to identify and reduc ect of temporal error on the functional. The temporal error relevant to the functional most likely occurs in the re e temporal domain when the vortex is still ahead of the airfoil, and also in the region when it interacts with th likely that the temporal error once the vortex passes the airfoil is not relevant to the





(a) Functional convergence (b) Functional error convergence

Fig. 21. Convergence of functional and error with respect to temporal mesh resolution for Case (B).



(a) Functional convergence (b) Functional error convergence

Fig. 22. Convergence of functional and error with respect cost in terms of wall time for Case (B).

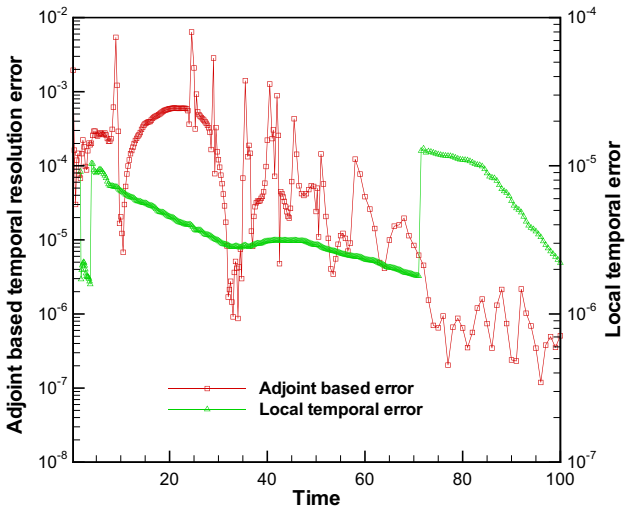


Fig. 23. Comparison of adjoint-based error distribution and local temporal error distribution for Case (B) after three adaptation cycles.

functional. The time-integration must therefore use smaller time-steps while the vortex is still ahead of the airfoil in order to ensure minimal dissipation of the vortex, and smaller time-steps during the interaction phase to ensure that all of the details in the perturbation of the airfoil lift are captured. The adjoint method being goal-based should predict the error distribution in accordance with this expectation, while local error-based adaptation will likely request refinement of the entire time domain. This happens as a consequence of the vortex continuing to convect and dissipate downstream of the airfoil, although its effect is not relevant to the load on the airfoil. Therefore local error-based adaptation is likely to perform no better than uniform refinement of the whole time domain. The error tolerance on the functional was set at 1e−3 for the termination of adaptation cycles for this case.

Fig. 21(a) and (b) show the convergence of the functional and functional error with respect to the temporal mesh resolution, while Fig. 22(a) and (b) compare the same with the computational expense. For this particular case, adjoint-based adaptation is able to achieve similar error levels as uniform refinement and local error-based adaptation with fewer than half the number of time-steps. From a cost perspective, the plots indicate that at least initially the expense is similar between all methods but after a two to three adaptation cycles, adjoint-based adaptation is able to outperform uniform refinement and local error-based adaptation. The plots also indicate the expected trend of local error-based adaptation performing similarly to uniform refinement. Fig. 23 compares the distribution of the temporal error computed using the adjoint and the local error method after three adaptation cycles. It is quite clear that the local error distribution is fairly uniform throughout the time domain in contrast to the adjoint-based error where dominance is obvious prior to the vortex passing the airfoil. Fig. 24 compares the time-step size distribution between adjoint-based and the local error-based method after termination

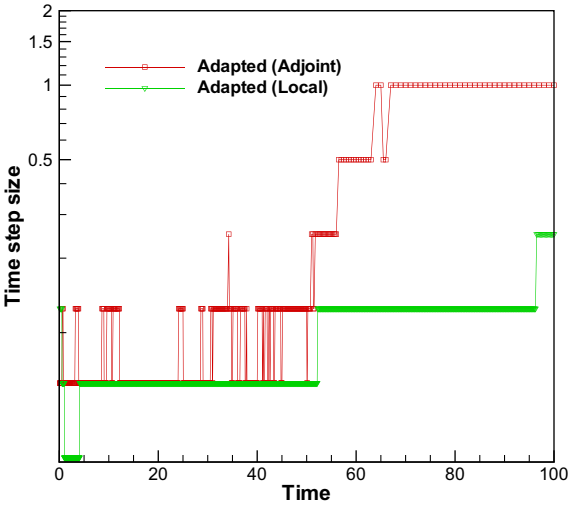
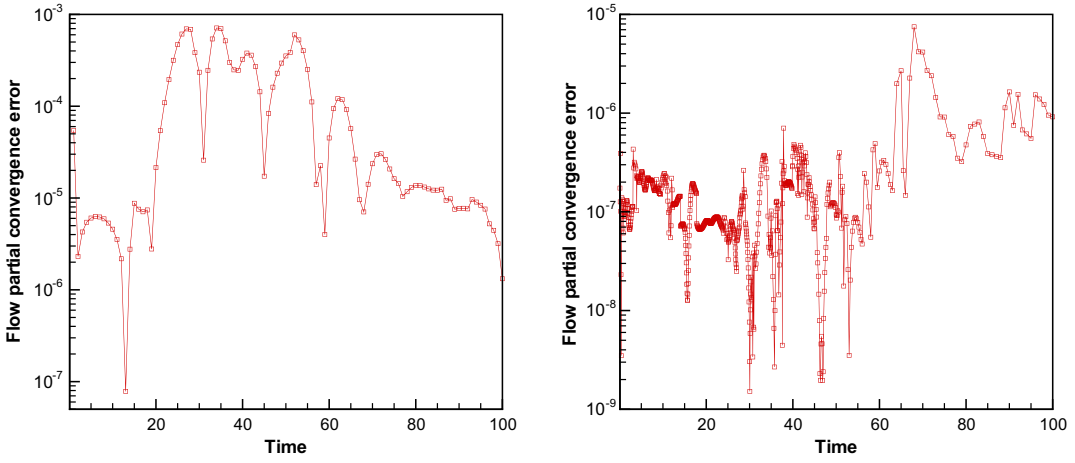
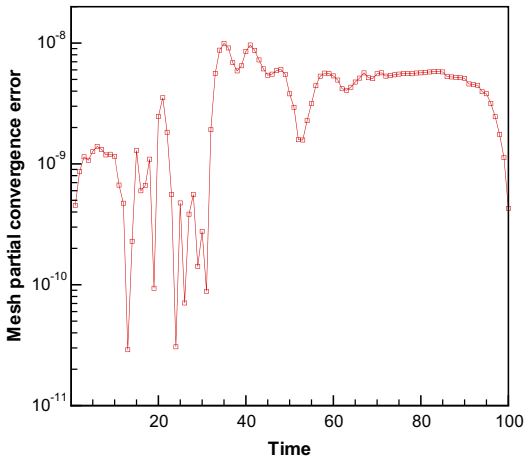


Fig. 24. Comparison of time-step size distribution for adjoint-based and local error-based adaptation after final adaptation cycle for Case (B).

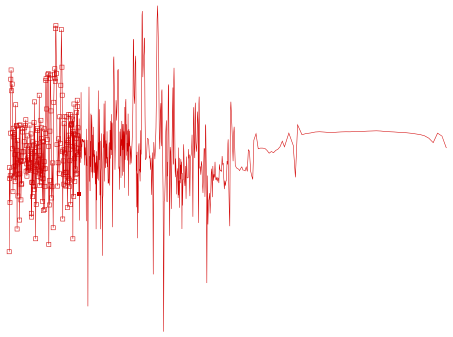


(a) 1st adaptation Cycle (b) Final (5^{th}) adaptation Cycle

Fig. 25. Flow partial convergence error distribution for Case (B).



(a) 1st adaptation Cycle



(b) Final (5^{th}) adaptation Cycle

of the adaptation. The adjoint-based method has not touched the region in the time domain when the vortex has passed the airfoil, while local error-based adaptation has requested fairly small time-step sizes in this region.

The algebraic error distribution due to partial convergence of the governing equations is particularly interesting for this problem. From Figs. 25 and 26 it can be seen that the maximum algebraic error due to the flow equations at the start of the adaptation occurs when the vortex interacts with the airfoil, while the maximum algebraic error due to the mesh equations occurs not only during the interaction of the vortex with the airfoil but even after the vortex passes the airfoil. By far the most counter-intuitive results are the algebraic error distributions for both the flow and mesh equations after the termination of adaptation. Both sets of equations continue to show higher error when the vortex is already downstream of the airfoil although adaptation of the convergence tolerances has occurred mostly in this region as shown in Fig. 27. This is a good example where engineering judgment in lieu of a mathematically rigorous procedure to determine convergence tolerances could potentially lead to poor results.

6. Conclusions

A method to determine global temporal error in unsteady flows was developed and tested. The algorithm replaces the ambiguity and guess work typically involved in intuitive temporal adaptation with reliable error estimates determined by establishing a mathematical connection between the functional of interest its relevant components of the local error. Additionally, the method also accounts for the various sources of error, such as those due to partial convergence of the governing

equations for the mesh motion and the flow. Because global adjoint error estimation is relatively expensive, due to the need to compute an unsteady adjoint solution which must be integrated backwards in time, rapid convergence of the adaptation procedure is required to make these techniques cost effective. On the other hand, the current approach provides not only an error estimate for the functional of interest, but also a breakdown of the important contributing sources of error. In principle, this technique can be extended to more complicated multi-physics simulations, where the overall error sources from various disciplines such as fluid flow, structural analysis, or conjugate heat transfer can be assessed, with implications for how best to deploy computational resources for maximum error reduction impact. Another area of interest concerns the combination of spatial and temporal adaptation for functional outputs. Considering the results demonstrated in this paper combined with existing work that has been done on spatial adaptation, the machinery now exists to adapt both the spatial and time domains independently. The obvious path from this point would be to address issues that arise from the coupling of spatial and temporal adaptation.

Acknowledgments

This work was supported by the US Air Force Office of Scientific Research under AFOSR Grant Number FA9550-07-1-0164.

Appendix A. Third-order BDF3 discretization on uniform temporal meshes

$$\left(\frac{dA\mathbf{U}}{dt}\right)^n = \left(\frac{1}{\Delta t}\right)\left[\frac{11}{6}(A\mathbf{U})^n - 3(A\mathbf{U})^{n-1} + \frac{3}{2}(A\mathbf{U})^{n-2} - \frac{1}{3}(A\mathbf{U})^{n-3}\right]$$

Appendix B. Third-order BDF3 discretization on non-uniform temporal meshes

$$\left(\frac{dA\mathbf{U}}{dt}\right)^n = -\left(\frac{C_2}{C_4}\right)(A\mathbf{U})^n - \left(\frac{C_3}{C_4}\right)(A\mathbf{U})^{n-1} + \left(\frac{1}{C_4}\right)(A\mathbf{U})^{n-2} - \left(\frac{C_1}{C_4}\right)(A\mathbf{U})^{n-3}$$

$$C_1 = \frac{k_2^2(k_1 - k_2)}{k_3^2(k_1 - k_3)}$$

$$C_2 = (1 - C_1) + \frac{C_1 k_3^3 - k_2^3}{k_1^3}$$

$$C_3 = \frac{k_2^3 - C_1 k_3^3}{k_1^3}$$

$$C_4 = \frac{k_2(k_2^2 - k_1^2) + C_1 k_3(k_1^2 - k_3^2)}{k_1^2}$$

$$k_1 = \Delta t_1$$

$$k_2 = \Delta t_1 + \Delta t_2$$

$$k_3 = \Delta t_1 + \Delta t_2 + \Delta t_3$$

$$\Delta t_1 = T^n - T^{n-1}$$

$$\Delta t_2 = T^{n-1} - T^{n-2}$$

$$\Delta t_3 = T^{n-2} - T^{n-3}$$

$$T = \text{time}$$

References

[1] M.H. Carpenter, S. Viken, E. Nielsen, The Efficiency of High-Order Temporal Schemes, AIAA Paper 2003-0086.
[2] C. Johnson, Error estimates and adaptive time step control for a class of one-step methods for stiff ordinary differential equations, SIAM Journal of Numerical Analysis 25 (1988) 908–926.
[3] D. Vendetti, D. Darmofal, Grid adaptation for functional outputs: application to two-dimensional inviscid flows, Journal of Computational Physics 176 (2002) 40–69.
[4] D. Vendetti, D. Darmofal, Anisotropic grid adaptation for functional outputs: application to two-dimensional viscous flows, Journal of Computational Physics 187 (2003) 22–46.
[5] M. Nemec, M.J. Aftosmis, M. Wintzer, Adjoint-based adaptive mesh refinement for complex geometries, in: 46th Aerospace Sciences Meeting and Exhibit, Reno, NV, 2008, AIAA Paper 2008-725.
[6] M. Wintzer, M. Nemec, M.J. Aftosmis, Adjoint-based adaptive mesh refinement for sonic boom prediction, in: 26th AIAA Applied Aerodynamics Conference, Honolulu, Hawaii, 2008, AIAA Paper 2008-6593.
[7] R. Balasubramanian, J.C. Newman III, Adjoint-based error estimation and grid adaptation for functional outputs: application to two-dimensional, inviscid, incompressible flows, Computers and Fluids 38 (2) (2009) 320–332.

[8] M. Park, Adjoint-based, three-dimensional error prediction and grid adaptation, AIAA Journal 42 (9) (2004) 1854–1862.
[9] M.B. Giles, E. Suli, Adjoint methods for PDEs: a posteriori error analysis and postprocessing by duality, Acta Numerica (2002) 145–236.
[10] P. Houston, R. Rannacher, E. Suli, A posteriori error analysis for stabilized finite-element approximations of transport problems, Computer Methods in Applied Mechanics and Engineering 190 (2000) 1483–1508.
[11] H.-J. Kim, K. Nakahashi, Output-based error estimation and adaptive mesh refinement using viscous adjoint method, in: Proceedings of the 44th Aerospace Sciences Meeting and Exhibit, Reno, NV, 2006, AIAA Paper 2006-1395.
[12] T.J. Barth, Space–time error representation and estimation in Navier–Stokes calculations, Lecture Notes in Computational Science and Engineering 56 (2007).
[13] J. Hoffman, C. Johnson, Adaptive finite element methods for incompressible fluid flow, Error Estimation and Adaptive Discretization Methods in CFD: Lecture Notes in Computational Science and Engineering, vol. 25, Springer-Verlag, 2002.
[14] J. Sitaraman, M. Floros, A. Wissink, M. Potsdam, V. Sankaran, Parallel unsteady overset mesh methodology for multi-solver paradigm with adaptive cartesian grids, in: 26th AIAA Applied Aerodynamics Conference, Honolulu, Hawaii, 2008, AIAA Paper 2008-7177.
[15] M. Potsdam, D.J. Mavriplis, Unstructured mesh CFD aerodynamic analysis of the NREL Phase VI rotor, in: 47th AIAA Aerospace Sciences Meeting and Exhibit, Orlando, FL, 2009, AIAA Paper 2009-1221.
[16] J.C. Butcher, Numerical Methods for Ordinary Differential Equations, Wiley, Chicester, UK, 2003.
[17] J.D. Lambert, Numerical Methods for Ordinary Differential Systems, Wiley, Chicester, UK, 1991.
[18] A.C. Hindmarsh, A.G. Taylor, PVODE and KINSOL: Parallel Software for Differential and Nonlinear Systems, UCRL-ID-129739, Lawrence Livermore National Laboratory.
[19] D. Estep, A posteriori error bounds and global error control for approximation of ordinary differential equations, SIAM Journal of Numerical Analysis 32 (1995) 1–48.
[20] Y. Cao, L. Petzold, A posteriori error estimation and global error control for ordinary differential equations by the adjoint method, SIAM Journal of Scientific Computing 26 (2) (2004) 359–374.
[21] S. Li, L. Petzold, Adjoint sensitivity analysis for time-dependent partial differential equations with adaptive mesh refinement, Journal of Computational Physics 198 (1) (2004) 310–325.
[22] S.K. Nadarajah, A. Jameson, Optimum shape design for unsteady flows with a time accurate continuous and discrete adjoint method, AIAA Journal (2007) 1478–1491.
[23] M.P. Rumpfkeil, D.W. Zingg, The optimal control of unsteady flows with a discrete adjoint method, Journal of Optimization and Engineering (2008). doi:10.1007/s11081-008-9035-5.
[24] K. Mani, D.J. Mavriplis, Unsteady discrete adjoint formulation for two-dimensional flow problems with deforming meshes, AIAA Journal 46 (6) (2008) 1351–1364.
[25] N. Yamaleev, B. Diskin, E. Nielsen, Discrete adjoint-based design optimization of unsteady turbulent flows on dynamic unstructured grids, in: 12th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference, Victoria, British Columbia, 2008, AIAA Paper 2008-5857.
[26] D.J. Mavriplis, Unstructured-mesh discretizations and solvers for computational aerodynamics, AIAA Journal 46 (6) (2008) 1281–1298.
[27] J.T. Batina, Unsteady Euler airfoil solutions using unstructured dynamic meshes, AIAA Journal 28 (8) (1990) 1381–1388.
[28] D.J. Mavriplis, Multigrid solution of the discrete adjoint for optimization problems on unstructured meshes, AIAA Journal 44 (1) (2006) 42–50.
[29] P. Geuzaine, C. Grandmont, C. Farhat, Design and analysis of ALE schemes with provable second-order time-accuracy for inviscid and viscous flow simulations, Journal of Computational Physics 191 (2003) 206–227.
[30] Z. Yang, D.J. Mavriplis, Higher-order time integration schemes for aeroelastic applications on unstructured meshes, AIAA Journal 45 (1) (2007) 138–150.
[31] D. Mavriplis, Z. Yang, Construction of the discrete geometric conservation law for high-order time-accurate simulations on dynamic meshes, Journal of Computational Physics 213 (2006) 557–573.
[32] K. Mani, D.J. Mavriplis, Discrete adjoint based time-step adaptation and error reduction unsteady flow problems, in: 18th AIAA Computational Fluid Dynamics Conference, Miami, FL, 2007, AIAA Paper 2007-3944.
[33] E. Nielsen, B. Diskin, N. Yamaleev, Discrete adjoint-based design optimization of unsteady turbulent flows on dynamic unstructured grids, in: 19th AIAA Computational Fluid Dynamics Conference, San Antonio, TX, 2009, AIAA Paper 2009-3802.
[34] M. Rumpfkeil, D. Zingg, A general framework for the optimal control of unsteady flows with applications, in: 45th AIAA Aerospace Sciences Meeting and Exhibit, Reno, NV, January, 2007, AIAA Paper 2007-1128.
[35] M.H. Carpenter, V. Vatsa, Higher order temporal schemes with error controllers for unsteady Navier–Stokes equations, in: 17th AIAA Computational Fluid Dynamics Conference, Toronta, Ontario, Canada, June 2005, 2005, AIAA Paper 2005-5245.