Available online at www.sciencedirect.com

SCIENCE DIRECT®

ELSEVIER

Journal of Computational Physics 213 (2006) 557–573

JOURNAL OF COMPUTATIONAL PHYSICS

www.elsevier.com/locate/jcp

Construction of the discrete geometric conservation law for high-order time-accurate simulations on dynamic meshes

Dimitri J. Mavriplis *, Zhi Yang

Department of Mechanical Engineering, University of Wyoming, Dept. 3295, 1000 E. University Avenue, Laramie, WY 82071, USA

Received 3 February 2005; received in revised form 16 August 2005; accepted 22 August 2005
Available online 4 October 2005

Abstract

The formulation and implementation of higher-order accurate temporal schemes for dynamic unstructured mesh problems which satisfy the discrete conservation law are presented. The general approach consists of writing the spatially-discretized equations for an arbitrary-Lagrange–Eulerian system (ALE) as a non-homogeneous coupled set of ODE's where the dependent variables consist of the product of the flow variables with the control volume. Standard application of backwards difference (BDF) and implicit Runge–Kutta (IRK) schemes to these ODE's, when grid coordinates and velocities are known smooth functions of time, results in the design temporal accuracy of these schemes. However, in general, these schemes do not satisfy the GCL and are therefore not conservative. Using a suitable approximation of the grid velocities evaluated at the locations in time prescribed by the specific ODE time integrator, a GCL compliant scheme can be constructed which retains the design temporal accuracy of the underlying ODE time integrator. This constitutes a practical approach, since the grid velocities are seldom known as continuous functions in time. Numerical examples demonstrating design accuracy and conservation are given for one, two, and three-dimensional inviscid flow problems.

Keywords: Dynamic mesh; Geometric conservation law; High-order; Time accuracy

1. Introduction

High fidelity time-accurate fluid flow simulations are becoming increasingly important as practitioners seek to model more complex physical problems, and computational costs continue to decrease. However, time-accurate computational fluid dynamic (CFD) problems remain several orders of magnitude more expensive than equivalent steady-state problems, thus prompting the search for more efficient and accurate algorithms.

For problems with appreciable separation in time and spatial scales, such as unsteady aerodynamics, aeroelastics, and other moving body problems, where the time scale of the body motion is far removed from the characteristic fluid time scales, implicit time-integration schemes are required for practical solution methods. In many cases, this is achieved using operator splitting algorithms or backwards-difference schemes

[*] Corresponding author. Tel.: +1 307 7662868; fax: +1 307 7662695.
E-mail address: mavripl@uwyo.edu (D.J. Mavriplis).

0021-9991/$ - see front matter © 2005 Elsevier Inc. All rights reserved.
doi:10.1016/j.jcp.2005.08.018

[8,15,21,16]. While the vast majority of time-accurate implementations rely on schemes which are second-order accurate in time, recent work has shown the benefits of using higher-order time-integration strategies, such as third-order backwards difference schemes or up to fourth-order implicit Runge–Kutta schemes [10,2,9]. Higher-order time-integration schemes achieve equivalent accuracy, while using larger time steps, which results in fewer function evaluations (i.e., residual evaluations and implicit system solutions) for a given level of accuracy. While the benefits of higher-order time-integration schemes increase for higher accuracy tolerances, these methods have been shown to be effective at reducing overall solution time even for relatively low error tolerances typical of engineering type calculations [2,9].

However, the majority of the work on high-order time accuracy has been performed for cases involving static grids. There exists a large class of problems involving relative body motion, such as aeroelastics, which require the use of dynamically deforming computational meshes. The objective of this paper is to extend the formulation of high-order time-accurate methods devised for static mesh applications to dynamic mesh problems, using unstructured ALE type meshes for geometric flexibility. Previous work has shown the importance of respecting the discrete conservation law (GCL) for structured and unstructured dynamic mesh problems. Briefly stated, the GCL corresponds to the statement that no disturbances should be introduced by any arbitrary mesh motion for a uniform flow.

The original statement of the GCL dates back to [20] for structured meshes. The GCL has been discussed in the context of aeroelastic problems for structured grids [6], and unstructured grids [24,18,21,14,12,7,5], and has been formulated for higher-order accurate spatial schemes on structured meshes as well [22]. For aeroelastic problems, severe degradation in overall simulation accuracy has been shown when the GCL is not respected, which may result in erroneous flutter boundary predictions [12]. Thus, the correlation of the GCL with time accuracy has been implied in most previous studies, although until recently [5], a quantitative study of the GCL on time accuracy had not been performed. For example, increased solution accuracy has been demonstrated by respecting the GCL on unstructured meshes in [12], and for high-order spatial schemes on structured meshes in [22], but in neither case is the design accuracy of the temporal scheme demonstrated either analytically, or through a time-step refinement study.

A notable contribution was the proof that respecting the GCL is a sufficient condition for achieving first-order temporal accuracy [7]. More recently, a precise derivation of the time accuracy of various GCL-compliant and non-compliant schemes has been performed in [5]. This study has shown, perhaps surprisingly, that respecting the GCL is not a *necessary* condition for achieving the design temporal accuracy of the underlying time-integration scheme. Furthermore, while respecting the GCL remains a sufficient condition for achieving first-order temporal accuracy, it is not a sufficient condition for obtaining the design accuracy of the underlying time-integration scheme (when the latter is higher than one) [5]. On the other hand, it has also been shown that respecting the GCL is a necessary and sufficient condition for preserving the non-linear stability of the underlying time-integration scheme [4]. This provides a strong incentive for constructing time-integration schemes which respect the GCL.

In order to devise higher-order time-accurate schemes for dynamic meshes, the interplay between the GCL and the time accuracy of the resulting scheme must be better understood. In this paper, we will argue that the role of the GCL is not related to time accuracy. Rather, the GCL is a conservation statement, and just as one can construct spatially conservative and non-conservative schemes of any order of accuracy, we will demonstrate both high-order accurate temporal schemes which violate and respect the GCL. In the interest of reducing overall sources of errors and preserving non-linear stability, we then concentrate on the formulation of high-order time-accurate (up to fourth-order) schemes for dynamic meshes which respect the GCL.

2. Governing equations in arbitrary-Lagrangian–Eulerian (ALE) form

The Navier–Stokes equations in conservative form can be written as:

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

where $\mathbf{U}$ represents the vector of conserved quantities (mass, momentum, and energy), $\mathbf{F}(U)$ represents the convective fluxes and $\mathbf{G}(U)$ represents the viscous fluxes. Integrating over a (moving) control volume $\Omega(t)$, we obtain:

$$\int_{\Omega(t)} \frac{\partial \mathbf{U}}{\partial t}\,\mathrm{d}V + \int_{\partial\Omega(t)} (\mathbf{F}(\mathbf{U}) \cdot \vec{\mathbf{n}})\,\mathbf{dS} + \int_{\partial\Omega(t)} (\mathbf{G}(\mathbf{U}) \cdot \vec{\mathbf{n}})\,\mathbf{dS} = \mathbf{0}. \tag{2}$$

Using the differential identity

$$\frac{\partial}{\partial t}\int_{\Omega(t)} \mathbf{U}\,\mathrm{d}V = \int_{\Omega(t)} \frac{\partial \mathbf{U}}{\partial t}\,\mathrm{d}V + \int_{\partial\Omega(t)} \mathbf{U}(\dot{\mathbf{x}} \cdot \vec{\mathbf{n}})\,\mathbf{dS}, \tag{3}$$

where $\dot{\mathbf{x}}$ and $\vec{\mathbf{n}}$ are the velocity and normal of the interface $\partial\Omega(t)$, respectively, Eq. (2) becomes:

$$\frac{\partial}{\partial t}\int_{\Omega(t)} \mathbf{U}\,\mathrm{d}V + \int_{\partial\Omega(t)} (\mathbf{F}(\mathbf{U}) - \dot{\mathbf{x}}\mathbf{U}) \cdot \vec{\mathbf{n}}\mathbf{dS} + \int_{\partial\mathbf{\Omega}(\mathbf{t})} \mathbf{G}(\mathbf{U}) \cdot \vec{\mathbf{n}}\mathbf{dS} = \mathbf{0}. \tag{4}$$

Considering $\mathbf{U}$ as cell averaged quantities, these equations are discretized in space as:

$$\frac{\partial}{\partial t}(V\mathbf{U}) + \mathbf{R}(\mathbf{U}, \mathbf{x}(\mathbf{t}), \dot{\mathbf{x}}(\mathbf{t}), \vec{\mathbf{n}}(\mathbf{t})) + \mathbf{S}(\mathbf{U}, \mathbf{x}(\mathbf{t}), \vec{\mathbf{n}}(\mathbf{t})) = \mathbf{0}, \tag{5}$$

where $\mathbf{R}(\mathbf{U}, \mathbf{x}(\mathbf{t}), \dot{\mathbf{x}}(\mathbf{t}), \vec{\mathbf{n}}(\mathbf{t})) = \int_{\partial\mathbf{\Omega}(\mathbf{t})}(\mathbf{F}(\mathbf{U}) - \dot{\mathbf{x}}\mathbf{U}) \cdot \vec{\mathbf{n}}\mathbf{dS}$ represents the discrete convective fluxes in ALE form, $\mathbf{S}(\mathbf{U}, \mathbf{x}(\mathbf{t}), \vec{\mathbf{n}}(\mathbf{t}))$ represents the discrete viscous fluxes, and V denotes the control volume. In the discrete form, $\dot{\mathbf{x}}(\mathbf{t})$ and $\vec{\mathbf{n}}(\mathbf{t})$ now represent the time varying velocities and surface normals of the control volume boundary faces, while $\mathbf{x}(\mathbf{t})$ denotes the mesh point positions. Note that the normals are computed from the mesh point positions, (i.e., $\vec{\mathbf{n}}(\mathbf{t}) = \vec{\mathbf{n}}(\mathbf{x}(\mathbf{t}))$), so that we may drop the $\vec{\mathbf{n}}(\mathbf{t})$ argument in the operators in the above equation.

3. The discrete geometric conservation law (GCL)

The discrete geometric conservation law requires that the state $\mathbf{U}$ = constant be an exact solution of Eq. (5). In this case, we have $S(\mathbf{U}, \mathbf{x}) = \mathbf{0}$, since the viscous fluxes are based on gradients of $\mathbf{U}$. Additionally, we have:

$$\int_{\partial\Omega(t)} (\mathbf{F}(\mathbf{U}) - \dot{\mathbf{x}}\mathbf{U}) \cdot \vec{\mathbf{n}}\mathbf{dS} = \mathbf{R}(\mathbf{U}, \mathbf{x}, \dot{\mathbf{x}}) = -\mathbf{U}\bar{\mathbf{R}}(\mathbf{x}, \dot{\mathbf{x}}), \tag{6}$$

since the integral of the convective fluxes $\mathbf{F}(U)$ around a closed control volume must be zero for constant $\mathbf{U}$, for any spatially conservative scheme, with $\mathbf{U}\bar{\mathbf{R}}$ referring to the discretization of the second term in the above boundary integral. Dividing through by $\mathbf{U}$, the GCL can be stated, in semi-discrete form as:

$$\frac{\partial V}{\partial t} - \bar{\mathbf{R}}(\mathbf{x}(\mathbf{t}), \dot{\mathbf{x}}(\mathbf{t})) = \mathbf{0}. \tag{7}$$

There are various ways in which the GCL may be satisfied. One approach [6] consists of adding the difference $\frac{\partial V}{\partial t} - \bar{\mathbf{R}}(\mathbf{x}(\mathbf{t}), \dot{\mathbf{x}}(\mathbf{t}))$ as a source term to the right-hand side of Eq. (5), thus ensuring that $\mathbf{U}$ = constant is a solution of Eq. (5), even in cases where Eq. (7) is not satisfied. Another approach consists of rewriting Eq. (5) as:

$$V\frac{\partial \mathbf{U}}{\partial t} + \mathbf{U}\frac{\partial V}{\partial t} + \mathbf{R}(\mathbf{U}, \mathbf{x}(\mathbf{t}), \dot{\mathbf{x}}(\mathbf{t})) + \mathbf{S}(\mathbf{U}, \mathbf{x}(\mathbf{t})) = \mathbf{0} \tag{8}$$

and then evaluating the volume derivative as

$$\frac{\partial V}{\partial t} = \bar{\mathbf{R}}(\mathbf{x}(\mathbf{t}), \dot{\mathbf{x}}(\mathbf{t})), \tag{9}$$

which automatically satisfies the GCL. However, as discussed in [22], this approach is not strictly conservative. While these approaches have been used in various structured grid calculations, the temporal accuracy of these schemes has not been addressed.

For dynamic unstructured meshes, fully conservative GCL compliant schemes are generally constructed after the integration of Eq. (5) in time. For example, for inviscid flows, using a second-order backwards difference scheme (BDF2), Eq. (5) is written as [14]:

$$\frac{3}{2}V\mathbf{U}^{n+1} - 2V\mathbf{U}^{n} + \frac{1}{2}V\mathbf{U}^{n-1} = -\Delta t\sum_{k=1}^{K_f}\alpha_k\mathbf{R}(\mathbf{U}^{n+1}, \mathbf{x}^{k}, \dot{\mathbf{x}}^{k}). \tag{10}$$

The evaluation of $\mathbf{U}^{n+1} = \mathbf{U}(t^{n+1})$ on the right hand side of Eq. (10) along with the condition $\sum_{k=1}^{K_f}\alpha_k = 1$ ensures this scheme recovers the standard backwards difference scheme in the absence of mesh motion. However, the apparent ambiguity of where to evaluate the grid coordinates and velocities in time is used as a degree of freedom in order to satisfy the GCL condition:

$$\frac{3}{2}V^{n+1} - 2V^n + \frac{1}{2}V^{n-1} = \Delta t\sum_{k=1}^{K_f}\alpha_k\bar{\mathbf{R}}(\mathbf{x}^k, \dot{\mathbf{x}}^k). \tag{11}$$

Because the left-hand side of Eq. (11) can be calculated exactly, since the grid point positions (and thus $V(t)$) are known exactly as a function of time, this equation can be used to determine the values $\mathbf{x}^k$, and $\dot{\mathbf{x}}^k$ which result in the BDF scheme of Eq. (10) satisfying the GCL. A related approach, which is computationally less expensive, consists of using a single residual evaluation with averaged grid metrics and velocities as [12]:

$$\frac{3}{2}V\mathbf{U}^{n+1} - 2V\mathbf{U}^n + \frac{1}{2}V\mathbf{U}^{n-1} = -\Delta t\mathbf{R}(\mathbf{U}^{n+1}, \bar{\mathbf{n}}, \bar{\dot{\mathbf{n}}}) \tag{12}$$

with

$$\bar{\mathbf{n}} = \sum_{k=1}^{K_f}\alpha_k\mathbf{n}(\mathbf{x}^\mathbf{k}), \bar{\dot{\mathbf{n}}} = \sum_{k=1}^{K_f}\alpha_k\dot{\mathbf{n}}(\mathbf{x}^\mathbf{k}, \dot{\mathbf{x}}^k), \tag{13}$$

using Eq. (11) to determine the values of $\mathbf{n}(\mathbf{x}^\mathbf{k})$ and $\dot{\mathbf{n}}(\mathbf{x}^\mathbf{k}, \dot{\mathbf{x}}^\mathbf{k})$, which represent the face normals and face normal integrated velocities, which appear explicitly in the ALE residual formulations, and which are functions of the mesh point positions and velocities. It is easily verified that both schemes obey the GCL condition, and both schemes reduce to the standard BDF2 scheme for static meshes. However, to determine the temporal accuracy of these schemes in the presence of dynamic meshes, a full Taylor series expansion in time is required, as described in [5]. The simultaneous requirement of second-order accuracy, and satisfying the GCL, results in the determination of the coefficients α_k and the k locations in time ($t_k = t^n + \theta_k\Delta t$, $0 < \theta_k < 1$) for evaluating the mesh positions and velocities. However, the extension of this procedure to high-order temporal discretizations represents a difficult task [23].

4. GCL and time accuracy

In this work, we take an alternative point of view for deriving high-order time-accurate GCL compliant schemes. Of interest is to better understand the role of the GCL in determining the time accuracy of a scheme, since in the above approaches, both GCL and time-accuracy requirements are considered concurrently. We begin by examining the issue of time accuracy alone, returning subsequently to the incorporation of a GCL condition. Our starting point is the semi-discretized equation (5), which was obtained by discretizing the governing equations in space. This represents a system of coupled ordinary differential equations which must be integrated in time. The time accuracy of ODE integrators is well understood, and has been used to devise accurate methods for time-dependent flows on static meshes in previous work. It is worth noting at this stage, that the standard form for a system of ODEs is often written as:

$$\frac{\mathrm{d}\mathbf{y}}{\mathrm{d}t} = \mathbf{f}(\mathbf{y}, t), \tag{14}$$

where $\mathbf{y}(t)$ represents the solution vector. For static mesh problems, Eq. (5) reduces to

$$V\frac{\partial}{\partial t}(\mathbf{U}) + R(\mathbf{U}) + S(\mathbf{U}) = 0, \tag{15}$$

which dividing through by V, which is now a constant, can be seen to correspond to a system of ODEs of the homogeneous form:

$$\frac{\mathrm{d}\mathbf{y}}{\mathrm{d}t} = \mathbf{f}(\mathbf{y}), \tag{16}$$

resulting in straightforward application of ODE integrators to the semi-discrete equations. However, for dynamic mesh problems, Eq. (5) must be re-written in a form analogous to (14) prior to the application of ODE integration schemes. This can be achieved by first noticing that the dependent variable in (5) is no longer $\mathbf{U}$, but $V\mathbf{U}$. Consequently, Eq. (5) is re-written as:

$$\frac{\partial}{\partial t}(V\mathbf{U}) = -\mathbf{R}(V\mathbf{U}, V, \mathbf{x}(t), \dot{\mathbf{x}}(t), \vec{\mathbf{n}}(t)) - \mathbf{S}(V\mathbf{U}, V, \mathbf{x}(t), \vec{\mathbf{n}}(t)). \tag{17}$$

Furthermore, since the volume V is computed using the grid point coordinates, we have $V = V(\mathbf{x}(t))$, and the V arguments in the right-hand-side operators may be omitted, since their dependence is reflected in the argument $\mathbf{x}(t)$ of these operators. Additionally, the grid velocities appear only as integrals along face normals, so that we may define the integrated face velocities as $\dot{\mathbf{n}}(t) = \dot{\mathbf{x}} \cdot \vec{n}$, finally leading to the form:

$$\frac{\partial}{\partial t}(V\mathbf{U}) = -\mathbf{R}(V\mathbf{U}, \mathbf{x}(t), \dot{\mathbf{n}}(t)) - \mathbf{S}(V\mathbf{U}, \mathbf{x}(t)). \tag{18}$$

Thus, if the grid point positions and velocities are known functions of time, the above may be written as

$$\frac{\partial}{\partial t}(V\mathbf{U}) = -\mathbf{R}(V\mathbf{U}, t) - \mathbf{S}(V\mathbf{U}, t) = \mathbf{Q}(V\mathbf{U}, t), \tag{19}$$

which is of the form of Eq. (14). Applying any ODE time-integration scheme to Eq. (19), the design temporal accuracy of the scheme must be recovered, provided the right-hand side of Eq. (19) is a sufficiently smooth function of time (and of $V\mathbf{U}$), the degree of which depends on the particular ODE integration scheme. Furthermore, there is no ambiguity as to where the grid coordinates and velocities should be evaluated. These are dictated by the chosen ODE integration scheme, and if these are evaluated elsewhere, design accuracy cannot be guaranteed. For example, the general form for a k-step backwards difference scheme applied to Eq. (19) can be written as:

$$(V\mathbf{U})^{n+1} - \sum_{i=0}^{k-1} \alpha_i (V\mathbf{U})^{n-i} = \Delta t \beta_k \mathbf{Q}((V\mathbf{U})^{n+1}, t^{n+1}). \tag{20}$$

Thus, for a backwards-difference (BDF) scheme, the grid coordinates and velocities in the residual should only be evaluated at the new time step location $n + 1$. The second-order accurate (BDF2) scheme, which will be used in the subsequent numerical examples, is obtained by setting $k = 2$ and using the coefficients $\alpha_0 = -\frac{4}{3}$, $\alpha_1 = \frac{1}{3}$, $\beta_2 = \frac{2}{3}$.

The general form for an s-stage implicit Runge–Kutta scheme applied to Eq. (19) is given as:

$$(V\mathbf{U})^k = (V\mathbf{U})^n + \Delta t \sum_{j=1}^{s} a_{kj} \mathbf{Q}((V\mathbf{U})^j, t_n + c_j \Delta t), \quad k = 1, \ldots, s, \tag{21}$$

$$(V\mathbf{U})^{n+1} = (V\mathbf{U})^n + \Delta t \sum_{j=1}^{s} b_j \mathbf{Q}((V\mathbf{U})^j, t_n + c_j \Delta t), \tag{22}$$

where the n and $n + 1$ superscripts on the dependent variable refer to the values at the beginning and end of the current time step, and the k and j superscripts refer to stage values computed by the Runge–Kutta scheme within the current time-step. Once again, it is clear from Eq. (21) that the grid coordinates and velocities are to be evaluated at the locations in time defined precisely by the c_j coefficients of the Runge–Kutta scheme itself. In the following examples, we will concentrate on the Explicit first-stage, Single diagonal coefficient, Diagonally Implicit class of RK schemes (ESDIRK). The coefficients for the considered schemes can be represented by the Butcher tableau [3], where the numerical values of the coefficients for the scheme used in this work are given in Appendix. These schemes are characterized by a lower triangular form of the coefficient table, thus resulting in a single implicit solve at each individual stage. The $c_k = \sum_{j=1}^{k} a_{kj}$, $k = 1, \ldots, s$ coefficients denote the corresponding point in time for each individual stage, which fixes the evaluation of grid coordinates and velocities. The first stage is explicit ($a_{1j} = 0$), and the last stage coefficients are such that $a_{kj} = b_j$, $j = 1, \ldots, s$, which omits the requirement of evaluating Eq. (22), since the value of $(V\mathbf{U})^{n+1}$ at the new time step is given by the last stage value $(V\mathbf{U})^{k=s}$. These properties ensure that the first and last stage values are identical to the time-integrated solution values in the RK scheme (i.e., avoiding the requirement of making any distinction between the estimated stage value and the final solution value at these points in time) which is consistent with the fact that the function $V(t)$ is single valued in time, thus simplifying our subsequent consideration of the GCL (see Table 1).

As an illustrative example, we consider a one-dimensional inviscid compressible flow problem, namely Sod's shock-tube problem with the conditions:

$$\begin{aligned} &\rho = 1, \quad p = 10^5, \quad u = 0, \quad -L \leqslant x < 0, \\ &\rho = 0.125, \quad p = 10^4, \quad u = 0, \quad 0 \leqslant x \leqslant L, \end{aligned} \tag{23}$$

where ρ, p, u represent fluid density, pressure and velocity, respectively, and $L = 10$. The computational domain contains 1001 grid points. A dynamically deforming mesh is obtained by forcing the grid points in the region $-L/3 < x < L/3$ to oscillate about their mean (equispaced) position according to:

$$\Delta x(t) = A(x)\sin(ft), \tag{24}$$

where $\Delta x(t)$ represents the deviation from the initial equispaced grid point locations. The frequency is taken as $f = 900\pi$ and the amplitude $A(x)$ is prescribed as a sin function which vanishes at $-L/3$ and at $L/3$. This smooth variation in the amplitude, and the vanishing motion near the boundaries is used to avoid possible boundary effects on the subsequent accuracy study. The shock-tube problem is solved on this dynamically deforming mesh using a second-order backwards differencing scheme BDF2, and a six stage, fourth-order accurate implicit ESDIRK Runge–Kutta scheme, IRK64, which has been previously employed for static mesh simulations [2,9] (see Appendix for complete description of scheme coefficients). A typical solution, in terms of the density profile at $t = 0.01$, is shown in Fig. 1. In both cases, the exact analytical values for the mesh point positions and velocities are used in the time-integration procedure. We denote this approach as Scheme A, for both the BDF2 and IRK64 cases. Fig. 2 depicts the error as a function of time-step size, for the solution integrated from $t = 0$ to $t = 0.01$. The error is defined as the L2 norm of the difference of all the conservative flow variables at all grid points between the current solution and a reference solution, which is obtained using the IRK64 scheme with a very fine time step of 6.25×10^{-7} (i.e., 10 times smaller than the smallest time step used in the tested cases). The slope of the error curves in log–log format is 1.9 for the BDF2 scheme, and 3.5 for the IRK64 scheme, which is close to design accuracy for both schemes.

It is easily verified, either analytically or numerically, that the GCL is not satisfied by Scheme A, which employs exact analytical values of the grid coordinates and velocities, using either the BDF2 or IRK64 time integrators. This example demonstrates the fact that the GCL is not a necessary condition for achieving design accuracy of a particular time-integration scheme, as was previously shown in [5].

On the other hand, it is a relatively simple task to construct a BDF scheme or IRK scheme which obeys the GCL. Consider the following scheme, denoted as Scheme B, and derived following the procedure described in [19], for IRK integrators. For each Runge–Kutta stage (or quadrature point) k, we define the average interface velocity and normal as:

$$\dot{\mathbf{x}}^{\frac{n+k}{2}} = \frac{\mathbf{x^k} - \mathbf{x^n}}{c_k \Delta t}, \quad \vec{\mathbf{n}}^{\frac{n+k}{2}} = \frac{1}{2}\left(\vec{\mathbf{n}}^n + \vec{\mathbf{n}}^k\right) \tag{25}$$

and perform stage updates as:

Table 1
Butcher tableau for the ESDIRK class of RK schemes with number of stages, $s = 6$

$c_1 = 0$	0	0	0	0	0	0
c_2	a_{21}	a_{66}	0	0	0	0
c_3	a_{31}	a_{32}	a_{66}	0	0	0
c_4	a_{41}	a_{42}	a_{43}	a_{66}	0	0
c_5	a_{51}	a_{52}	a_{53}	a_{54}	a_{66}	0
$c_6 = 1$	b_1	b_2	b_3	b_4	b_5	a_{66}
$\mathbf{w}^{n+1}$	b_1	b_2	b_3	b_4	b_5	a_{66}

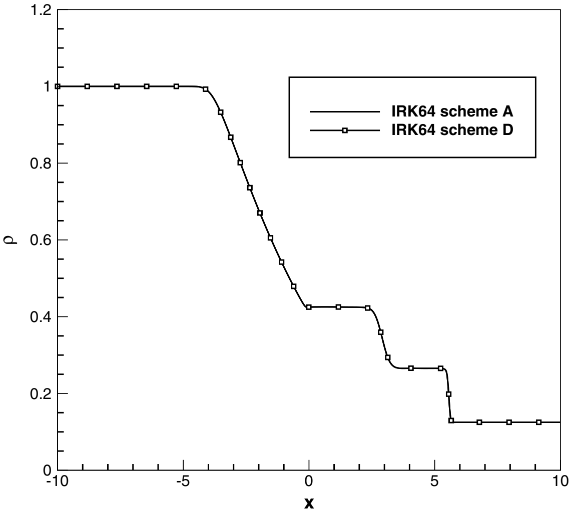

Fig. 1. Density distribution at $t = 0.01$. Shock tube – Reimann problem.

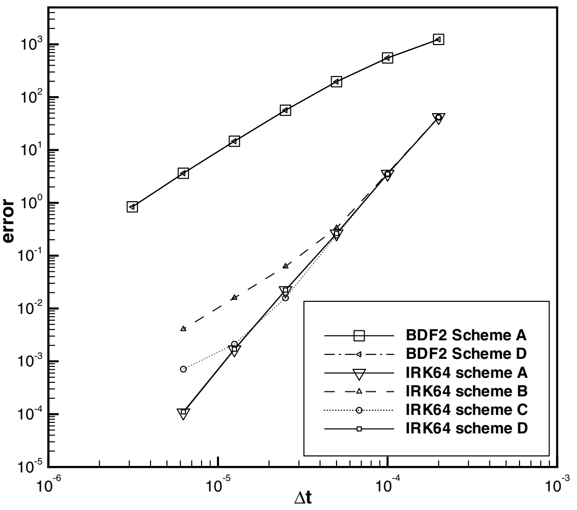

Fig. 2. Comparison of various BDF2 and IRK64 schemes for one-dimensional shock tube – Reimann problem.

$$(V\mathbf{U})^k = (V\mathbf{U})^n - \Delta t \sum_{j=1}^{k} \alpha_{kj}\mathbf{R}(\mathbf{U}^k, V^k, \dot{\mathbf{x}}^{\frac{n+k}{2}}, \vec{\mathbf{n}}^{\frac{n+k}{2}}), \quad k = 1, \ldots, s, \tag{26}$$

where the viscous flux operator $\mathbf{S}(\mathbf{U}, \mathbf{x}(t))$ has been dropped for this inviscid flow test case. This scheme satisfies the GCL at each Runge–Kutta stage. However, the error plot for this scheme applied to the dynamic mesh shock-tube problem results in curves with a final slope of 1.9, as shown in Fig. 2, which is far removed from the fourth-order design accuracy of the underlying IRK scheme.

An alternate Runge–Kutta-based GCL satisfying scheme, denoted as Scheme C is constructed by using the single values

$$\dot{\mathbf{x}} = \frac{\mathbf{x}^{n+1} - \mathbf{x}^n}{\Delta t}, \quad \vec{\mathbf{n}} = \frac{1}{2}(\vec{\mathbf{n}}^n + \vec{\mathbf{n}}^{n+1}) \tag{27}$$

for grid face velocities and normals, which are used at each Runge–Kutta stage, while redefining the volume at each stage as:

$$V^k = V^n + c_k(V^{n+1} - V^n). \tag{28}$$

The achieved temporal accuracy of this scheme for the shock-tube problem is again seen to fall below design accuracy, from the error curves in Fig. 2. Because both of these schemes are designed to satisfy the GCL, and both schemes revert to the standard IRK64 scheme for static meshes, these examples illustrate that the GCL is not a sufficient condition for recovering the design accuracy of the underlying static grid time-integration scheme, when applied to dynamic mesh problems, as was also shown in [5].

There are various reasons to explain why these schemes do not recover design accuracy on dynamic meshes. For scheme B, the fact that the grid coordinates (normal vectors) are not evaluated precisely at the quadrature points of the RK scheme is sufficient to corrupt the temporal accuracy of the scheme. However, the evaluation of the grid velocities as averages between initial and stage values is also questionable, as the function $\dot{\mathbf{x}}(t)$, and thus the entire right-hand side of Eq. (20) is no longer a differentiable function of time, as is required in the Runge–Kutta theory [3,13].

The question thus becomes how to construct a Runge–Kutta scheme which evaluates the terms $\mathbf{x}(t)$ and $\dot{\mathbf{x}}(t)$ at the specified quadrature points, and which at the same time obeys the GCL. Our initial example, where the grid coordinates and velocities are prescribed functions of time, is somewhat unrealistic, in that analytic expressions for these quantities are seldom available. However, the grid-point coordinates are usually available as a function of time, particularly if the grid deformation is governed by the solution of a set of partial differential equations (Poisson equation, linear elasticity) as is often the case [1,17,23]. On the other hand, the grid velocities are seldom available as a function of time, and are most often obtained by finite differencing the grid point coordinate function. The approximation of the grid velocities, evaluated at the RK quadrature points, provides the added degree of freedom necessary to devise an RK scheme which also obeys the GCL. In the following scheme, denoted as Scheme D, we assume that the grid point coordinates are known functions of time, and that all quantities are evaluated at the quadrature points. Each stage of the Runge–Kutta scheme is given as:

$$(V\mathbf{U})^k = (V\mathbf{U})^n - \Delta t \sum_{j=1}^{k} \alpha_{kj} \mathbf{R}(\mathbf{U}^j, \mathbf{x}^j, \dot{\mathbf{n}}^j) \quad k = 1, \ldots, s \tag{29}$$

with the corresponding values: $\mathbf{x}^j = \mathbf{x}(t + c_j\Delta t)$, and $\dot{\mathbf{n}}^{\mathbf{j}} = \dot{\mathbf{x}}(t + c_j\Delta t) \cdot \vec{\mathbf{n}}(t + c_j\Delta t)$, with the values of the c_j being given by the RK scheme. While the grid point coordinates may be evaluated directly at these locations in time, the face integrated velocities $\dot{\mathbf{n}}$ must be estimated. These estimates are constructed such that the GCL is satisfied. Setting $\mathbf{U} = \text{constant}$ in Eq. (29), we obtain the GCL condition at each RK stage as:

$$V^k - V^n = \Delta t \sum_{j=1}^{k} \alpha_{kj} \bar{\mathbf{R}}(\dot{\mathbf{n}}^j) \quad k = 1, \ldots, s. \tag{30}$$

The $\bar{\mathbf{R}}$ operator corresponds to a discrete surface integral, which is obtained as a summation over individual control volume faces:

$$\bar{\mathbf{R}}(\dot{\mathbf{n}}^j) = \sum_{E=1}^{\text{Faces}} \dot{\mathbf{n}}_E^j. \tag{31}$$

Because the volume computation can also be formulated as a sum over control volume faces, we may require Eq. (30) to hold at each individual control volume face of the discrete integral:

$$(V^k - V^n)_E = \Delta t \sum_{j=1}^{k} \alpha_{kj} \dot{\mathbf{n}}_E^j \quad k = 1, \ldots, s, \tag{32}$$

where $(V^k - V^n)_E$ denotes the volume swept by an individual control volume boundary face, which is associated with a mesh edge for vertex-based finite-volume discretizations, and $\dot{\mathbf{n}}_E^j$ represents the unknown value of the face integrated grid velocity at the jth RK stage. Because the grid coordinates (and thus control volumes values) are known functions of time, the left hand-side of Eq. (32) can be evaluated exactly, and the $\dot{\mathbf{n}}_E^j$ unknowns can thus be obtained by solving the system (given here for a four stage RK scheme):

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ \alpha_{21} & \alpha_{22} & 0 & 0 \\ \alpha_{31} & \alpha_{32} & \alpha_{33} & 0 \\ \alpha_{41} & \alpha_{42} & \alpha_{43} & \alpha_{44} \end{pmatrix} \begin{Bmatrix} \dot{\mathbf{n}}_E^1 \\ \dot{\mathbf{n}}_E^2 \\ \dot{\mathbf{n}}_E^3 \\ \dot{\mathbf{n}}_E^4 \end{Bmatrix} = \frac{1}{\Delta t} \begin{Bmatrix} \Delta t \dot{\mathbf{n}}_E^{(1)} \\ V^2 - V^n \\ V^3 - V^n \\ V^4 - V^n \end{Bmatrix}, \tag{33}$$

where the explicit first stage $\alpha_{11} = 0$ (cf. Table 1) corresponds to equating $\dot{\mathbf{n}}_E^1$ with the value determined at the end of the previous time step. Because the IRK64 scheme is an SDIRK (single coefficient diagonally implicit RK) scheme, the matrix of a_{kj} coefficients is of lower triangular form, and Eq. (33) is easily solved by forward substitution.

This scheme is used to solve the shock-tube problem and the error convergence curve is plotted in Fig. 2, using a reference solution obtained with the same scheme with a smaller time step of 6.25×10^{-7}. The slope of the error curve is nearly identical to that obtained with the analytical values of grid coordinates and velocities, and is close to the design accuracy of the scheme. We have thus demonstrated a scheme which is both GCL compliant, and fourth-order accurate in time.

It should be noted that in all cases, the error is measured as the difference between the computed solution and a reference solution, where the reference solution is obtained using the same corresponding scheme with a much smaller time step. Because we are only considering time accuracy, and excluding spatial accuracy effects, the possibility that the various schemes may not converge to the same solution as $\Delta t \to 0$ (due for example to conservation errors), makes it important to use the corresponding scheme for the reference solution for time-accuracy studies. Fig. 3 illustrates details of the differences in the shock-tube density profile for Schemes A and D using the same time step $\Delta t = 0.0002$. While the profile is smooth for Scheme D, oscillations are present in the density profile computed by Scheme A. These errors are attributed to the lack of conservation due to the violation of the GCL in this scheme, and their behavior is consistent with the statement that the GCL is necessary and sufficient for non-linear stability [4]. These oscillations have been found to decrease with smaller time steps, however, it has not been determined whether the two schemes converge to the same solution in the limit of small time steps.

Although the numerical evidence demonstrates approximate design accuracy for Scheme D, in this particular case, a complete error analysis would be required to prove the formal order of accuracy of this scheme. A simpler approach is to examine the conditions under which the scheme is known to deliver design accuracy. Consider the case, where the volume swept by an edge $V_E(t)$ can be expressed as a fourth-order polynomial in time $p_4(t)$. Thus, we have

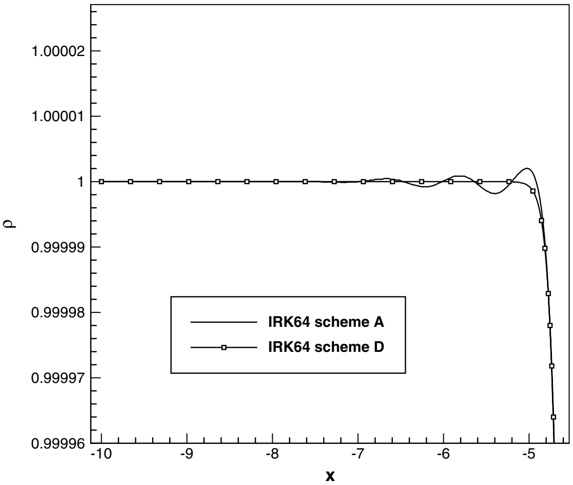

Fig. 3. Details of density distribution for non GCL compliant IRK64 Scheme A and GCL compliant IRK64 Scheme D at $t = 0.01$ for one-dimensional shock tube – Reimann problem using $\Delta t = 0.0002$.

$$\frac{\partial V_E}{\partial t} = \dot{\mathbf{n}}_{\mathbf{E}} = p_3(t), \tag{34}$$

where $p_3(t)$ denotes a third-order polynomial in time. In two dimensions, this would be the case, for example, if the grid coordinates are given as $\mathbf{x}(t) = p_2(t)$, i.e., a second-order polynomial in time. Use of the exact (analytic) values of the grid coordinates and the grid face-averaged velocities in the Runge–Kutta scheme will result in design accuracy, as demonstrated by Scheme A previously. Additionally, because the IRK64 scheme integrates a third-order polynomial exactly, Eq. (32) is automatically satisfied, and the approximate $\dot{\mathbf{n}}_E$ values derived from equation (33) coincide with the exact values. Thus, for $V(t) = p_4(t)$, Schemes A and D are equivalent, and both achieve design accuracy *and* obey the GCL simultaneously.

If the function $\mathbf{x}(t)$ is such that the resulting $V(t)$ is not four times differentiable, then the GCL will be respected, as long as Eq. (32) is satisfied, but design accuracy will be lost, since the resulting function $\mathbf{R}(\mathbf{U}, \mathbf{x}, \dot{\mathbf{n}})$ will not meet the smoothness requirements of the RK scheme. For problems where the mesh motion is governed by a set of partial differential equations, achieving high temporal accuracy requires a smooth $\mathbf{x}(t)$ variation, which implies the specification of a smooth boundary motion in time, and convergence of the mesh motion equations at each step to a suitable tolerance, which can be assumed to be of the same order as the convergence tolerance used for the implicit flow solver at each Runge–Kutta stage. The common practice of only partially converging the mesh motion equations in order to reduce computational costs, under the assumption that a valid mesh with positive cell volumes is all that is required at each time step, may thus result in the loss of higher-order temporal accuracy.

On the other hand, if the mesh motion $\mathbf{x}(t)$ is such that the resulting volume $V(t)$ is a higher-order polynomial in time (higher than order 4), or any four times differentiable non-polynomial function, the use of analytical values for $\dot{\mathbf{n}}_E$ will not satisfy the GCL, as equation (32) will not be satisfied in general. However, Eq. (32) or (33) may be viewed as a third-order polynomial interpolation for $\dot{\mathbf{n}}_E$ within a given time step, since it is exact for $\dot{\mathbf{n}}_E = p_3(t)$. Therefore, for smooth enough $\mathbf{x}(t)$ and $V(t)$, we may write:

$$\dot{\mathbf{n}}_E|_{\text{approx}}(t) = \dot{\mathbf{n}}_E(t) + \mathrm{O}(\Delta t)^4, \tag{35}$$

where $\dot{\mathbf{n}}_E|_{\text{approx}}(t)$ represents the approximations of the exact $\dot{\mathbf{n}}_E(t)$ obtained through Eq. (33), and our GCL compliant RK scheme is obtained by replacing the original ODE (cf. Eq. (18)) by the following ODE:

$$\frac{\partial}{\partial t}(V\mathbf{U}) = -\mathbf{R}(V\mathbf{U}, \mathbf{x}(t), \dot{\mathbf{n}}(t) - \mathrm{O}(\Delta t)^4) - \mathbf{S}(V\mathbf{U}, \mathbf{x}(t)), \tag{36}$$

where $\mathbf{x}(t)$ and $\dot{\mathbf{n}}(t)$ denote exact values. This equation is then integrated in time in the standard manner using the RK scheme. The remaining assumption, stated without proof, is that the differences between the solutions of the original and modified ODEs are of the same order as the temporal error of the RK scheme, thus preserving the design accuracy of the RK scheme while respecting the GCL. This also implies that the conservation errors introduced by the violation of the GCL when using the exact analytic values of $\mathbf{x}(t)$ and $\dot{\mathbf{n}}(t)$ are also of order $\mathrm{O}(\Delta t)^4$.

5. Remaining considerations

The above one-dimensional example has proven useful in studying the effect of the GCL on time accuracy and in devising a time-accurate implicit Runge–Kutta scheme which respects the GCL. In this section, we discuss the extension of these ideas to backwards-difference schemes, to multiple dimensions, and to viscous flows.

The derivation of a GCL compliant BDF scheme can be constructed in a similar fashion as discussed for IRK schemes. For a BDF scheme, all grid coordinates and face-averaged normal velocities should be evaluated at the $n+1$ time level, as dictated by Eq. (20). Setting $\mathbf{U} = \text{constant}$,we obtain the discrete GCL condition:

$$V^{n+1} - \sum_{i=0}^{k-1} \alpha_i V^{n-i} = \Delta t \beta_k \sum_{E=1}^{\text{Faces}} \dot{\mathbf{n}}_E^{n+1}. \tag{37}$$

The left-hand-side, which can be computed exactly, may also be written as a linear combination of the incremental changes in volume between the various time levels, which themselves may be decomposed into the elemental volumes swept by each control volume boundary face between time levels. The equality is then required to hold for each moving control volume boundary face, leading to:

$$\dot{\mathbf{n}}_E^{n+1} = \frac{1}{\Delta t}\sum_{i=1}^{k}\gamma_i \Delta V_E^i, \tag{38}$$

where $\Delta V_E^i = V_E^{i+1} - V_E^i$, represents the volume swept by the boundary face E between the two time levels i and $i+1$, and the γ_i coefficients are determined such that:

$$\sum_{i=1}^{k}\gamma_i \Delta V_E^i = \frac{1}{\beta_k}\left(V^{n+1} - \sum_{i=0}^{k-1}\alpha_i V^{n-i}\right). \tag{39}$$

For a BDF2 scheme (i.e., $\alpha_0 = -\frac{4}{3}$, $\alpha_1 = \frac{1}{3}$, $\beta_2 = \frac{2}{3}$) this yields the coefficients ($\gamma_1 = -\frac{1}{2}$, $\gamma_2 = \frac{3}{2}$). This method is applicable to any BDF scheme (BDF2, BDF3). In Fig. 2, the error curve for the one-dimensional dynamic mesh shock tube problem described above is shown, using a BDF2 scheme, which satisfies the GCL in the above manner, denoted as Scheme D. The figure reveals a slope for the error curve of 1.9, demonstrating design accuracy for this scheme.

A key feature of our GCL formulation, is the ability to compute the volume swept by a control volume boundary face exactly, as given by the right-hand sides of Eq. (33) for the Runge–Kutta schemes and Eq. (38) for the BDF scheme, given the function $\mathbf{x}(t)$. In one-dimension, this computation is trivial, as the control volume consists of a linear segment, and the control volume "faces" correspond to the two end-points of the segment. The volume swept by these faces is thus simply equal to the displacement of the end-points.

In two-dimensions, the control-volume boundary faces constitute straight-line segments. In the case of constant grid velocities (or at least constant velocity directions), the volume swept by these moving edges is equal to the area inside the quadrilateral formed by joining the initial and final points of the edge extremities, as shown in Fig. 4. This area can be computed exactly using a trapezoidal integration rule. While this integration is not exact for arbitrary grid velocities (time varying velocity directions), it is sufficient to yield the exact change in total control volume area when the contribution from all edges of a control volume are summed together, since the change in the control volume areas depend only on the initial and final states of the grid point coordinates. This is sufficient for the purposes of satisfying the GCL.

Likewise, in three-dimensions, the exact computation of the volume swept by a control-volume boundary face, under the assumption of constant grid velocities, is sufficient to yield the exact change in the control volume regardless of the actual variation of the grid velocities in time. For cell centered schemes on tetrahedra, control volume boundary faces consist of planar triangular elements, while on vertex-centered tetrahedral schemes, these consist of composite triangular faces associated with a mesh edge. In both cases, it is sufficient to consider the volume swept by a triangular face as its vertices move with a constant velocity between two time steps, as depicted in Fig. 5. The resulting space-time element constitutes a prismatic element, for which the quadrilateral faces are not necessarily planar, due to the different velocity vectors at the three vertices. (In fact, the volume of the prismatic element could even be negative). As pointed out in [14], a two-point integration rule (in time) is required to evaluate these volumes exactly. However, in this case, this is only used to compute the swept face volume, which is then used on the right-hand sides of Eqs. (33) and (38), rather than for directly evaluating grid coordinates and velocities as in [14]. For the BDF2 scheme described above, the face integrated velocities $\dot{\mathbf{n}}_{\mathbf{E}}$ reduce to the same expressions derived in Scheme A of [5], although the overall schemes are not identical, since as stated above, the other grid metrics (in the convective terms) are not evaluated in the same manner.

Currently, the discussion has centered on the inviscid flow equations. For viscous flows, since the viscous fluxes vanish in the case of uniform flow, the GCL will automatically be satisfied, provided the underlying inviscid flow solver satisfies the GCL. However, the time accuracy of the scheme may be compromised if the viscous flux terms are not evaluated appropriately in time [11]. Since these terms do not depend on the grid velocities, and since the grid coordinates are known functions of time, we have:

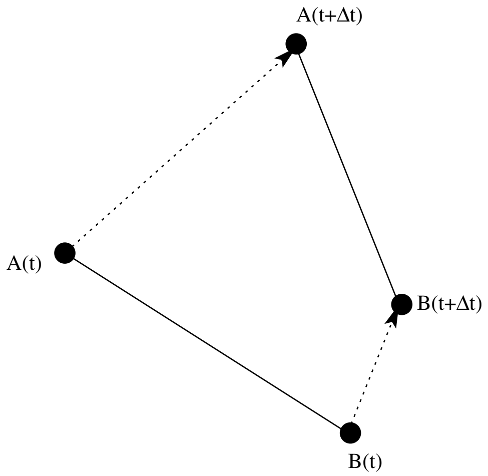

Fig. 4. Area swept by moving two dimensional control-volume boundary face.

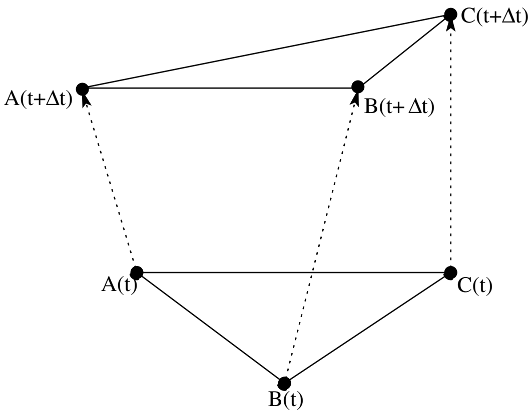

Fig. 5. Volume swept by moving three dimensional control-volume boundary face.

$$\mathbf{S}(V\mathbf{U}, \mathbf{x}(t)) = \mathbf{S}(V\mathbf{U}, t) \tag{40}$$

and the viscous fluxes must therefore be evaluated at the locations in time determined by the specific ODE integration scheme, i.e., $\mathbf{S}(V\mathbf{U}^{n+1}, t^{n+1})$ for BDF schemes, and at the stage value locations (or quadrature points) given by $t_j = t^n + c_j\Delta t$ for the Runge–Kutta schemes. This approach is consistent with the formulation for the viscous terms given in [5], and with the findings in [11], where the evaluation of viscous terms at the old time step t^n were demonstrated to result in lower time accuracy, while it is at odds with the evaluation of the grid metrics at the mid-point $t^{n+\frac{1}{2}} = \frac{1}{2}(t^n + t^{n+1})$ advocated in [11] for the viscous terms.

6. Multi-dimensional results

In the following examples, the second-order accurate backwards difference (BDF2) scheme and fourth-order six-stage ESDIRK Runge–Kutta (IRK64) scheme, which were derived above to be GCL compliant (i.e., Schemes D), are demonstrated on a two-dimensional and a three-dimensional dynamic mesh inviscid flow problem.

In two dimensions, we consider the inviscid transonic flow over a pitching NACA0012 airfoil about its quarter-chord point, with the airfoil incidence prescribed as a sinusoidal function of time, with a mean incidence of 0.016, and an amplitude of 2.51°. The freestream Mach number for this case is 0.755, and the reduced frequency of the periodic airfoil motion is $\frac{fc}{U} = 0.1628$, where f is the frequency of oscillation, c is the airfoil chord, and U is the freestream flow velocity. The two-dimensional unstructured mesh used for this case is depicted in Fig. 6. This mesh contains a total of 4379 vertices. At each time step, the outer boundary of the mesh is held fixed, while the airfoil surface mesh points are constrained to follow the airfoil motion. The location of the interior mesh points is then computed at each time-step (or IRK stage) by solving a set of partial-differential equations derived using a spring analogy (whereby each mesh edge is modeled as a non-linear spring) subject to these boundary conditions [23]. Both the mesh motion equations, and the flow equations are converged to machine precision at each stage of the RK scheme, and at each time step of the BDF scheme, in order to ensure smooth variations of the mesh coordinates as a function of time, and to avoid contamination of the temporal error with algebraic error. Fig. 7 depicts the lift coefficient as a function of angle of attack for this periodic motion, calculated using 8 and 64 time-steps per period using the BDF2 and the IRK64 GCL compliant schemes. The accuracy of the IRK64 scheme using only 8 time steps per period is seen to be close to that achieved by the BDF2 scheme using 64 time steps per period, and is

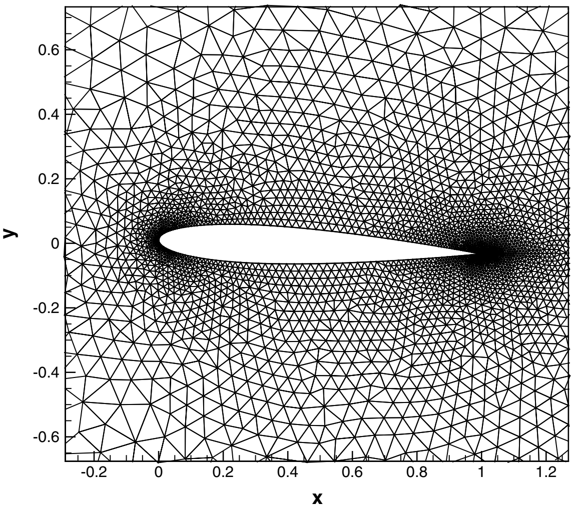

Fig. 6. Unstructured mesh employed for two-dimensional pitching NACA0012 airfoil case. Number of vertices = 4379.

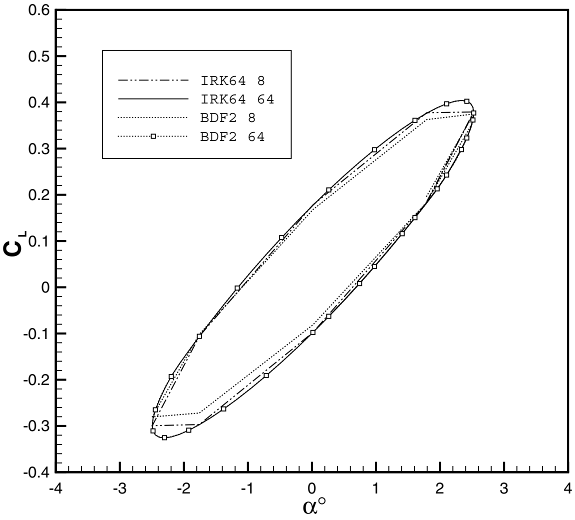

Fig. 7. Lift coefficient vs. angle of attack for pitching NACA0012 airfoil.

almost indistinguishable from the IRK64 scheme using 64 time steps. Fig. 8 shows the convergence of temporal error with time step refinement for both schemes. The error is measured as the L2 norm of the difference of all the conservative flow variables between the computed solution integrated from $t = 0$ to $t = 54$, and that of a highly resolved solution using the IRK64 scheme with a time step of 0.16875, which corresponds to 256 times steps per period. The slope of the error curves is 1.9 for the BDF2 scheme and 3.5 for the IRK64 scheme.

A three-dimensional inviscid flow test case is constructed by forced twisting of an ONERA M6 wing, at a Mach number of 0.755, and an incidence of 0.016°. A periodic pitching motion is prescribed at the quarter chord point of the wing tip, with a reduced frequency of 0.1628 and an amplitude of 2.51°. The wing root is held fixed, and a linear variation of the pitching motion is prescribed between the wing tip and root, thus

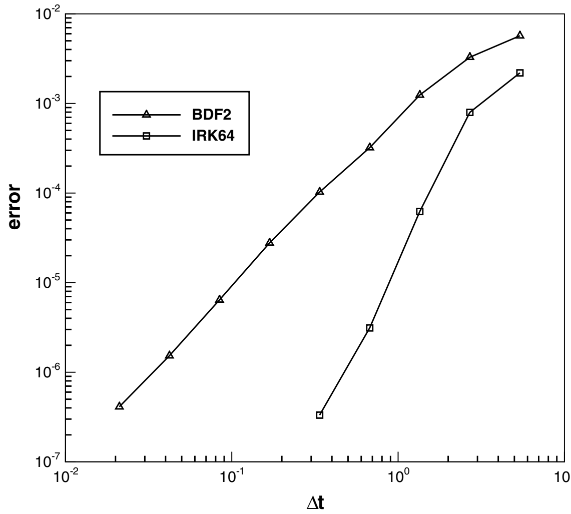

Fig. 8. Comparison of computed temporal errors for GCL compliant BDF2 and IRK64 schemes for pitching NACA0012 airfoil case at time $t = 54$.

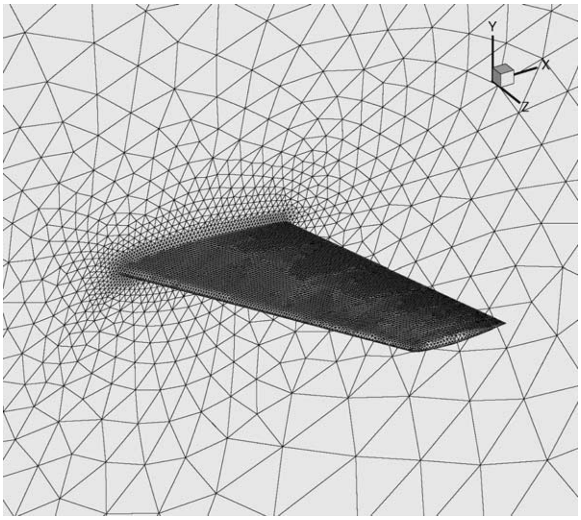

Fig. 9. Unstructured tetrahedral mesh for three-dimensional twisting ONERA M6 wing test case.

resulting in a twisting motion. A fully tetrahedral mesh, shown in Fig. 9, is used for this case, containing 53961 vertices. Mesh deformation is computed at each time-step or stage by solving the spring-analogy equations subject to a fixed outer-boundary condition. Both the mesh motion and flow equations are converged to machine precision at each stage or time step, as in the two-dimensional case. The computed lift coefficient versus time is shown in Fig. 10 for the IRK64 scheme, showing good temporal accuracy for the IRK64 scheme using as few as 8 time steps per period. Fig. 11 illustrates the computed temporal error as a function of the time-step for both schemes, at the time $t = 54$, using a highly resolved IRK64 solution with a time step of 0.3375, (corresponding to 128 time steps per period) as the reference solution. The slope of the error curve achieved for the BDF scheme is 2, while the IRK64 scheme results in an error curve slop of 3.3. As in the two-dimensional case, the IRK64 scheme generally achieves equivalent accuracy using time steps which are up to an order of magnitude larger than the BDF2 scheme.

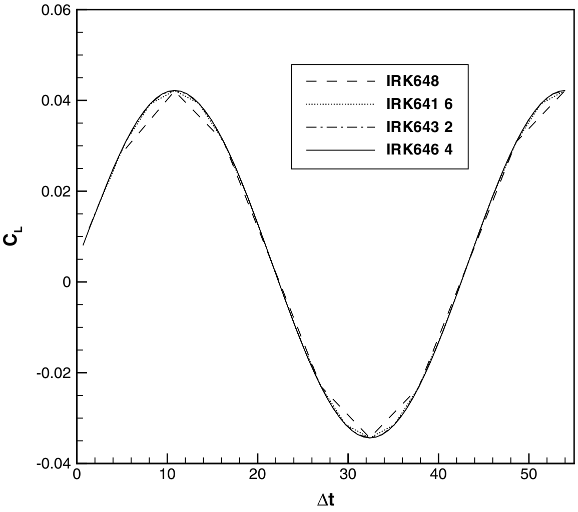

Fig. 10. Lift coefficient as a function of tip incidence for twisting ONERA M6 wing case as computed by IRK64 GCL compliant scheme.

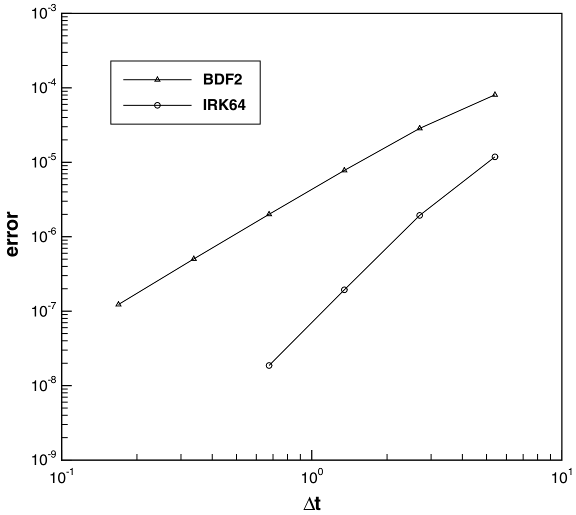

Fig. 11. Comparison of temporal errors for GCL compliant BDF2 and IRK64 schemes for three-dimensional twisting ONERA M6 wing case at time $t = 54$.

7. Conclusions

A general framework for deriving high-order temporal schemes which respect the GCL has been proposed in this work. In contrast to previous work, where the evaluation of grid coordinates and velocities is performed at various locations in time, in order to satisfy the GCL, the current approach evaluates these quantities precisely at the time locations dictated by the ODE time-integration scheme, but employs approximations to the grid face-integrated velocities at these locations in time to satisfy the GCL. It is then demonstrated that design temporal accuracy of the BDF2 and IRK64 schemes is approximately preserved, since the error involved in the grid velocity approximations is of the same order or smaller than the temporal errors of the scheme.

On the other hand, a formal accuracy proof remains to be given. This entails demonstrating that the errors between the solution of the exact system of ODEs and the modified system of ODE's using an approximate grid velocity function are of the same order or less than the temporal errors of the time-integration scheme. This is a non-trivial task for Runge–Kutta schemes, and the possibility of order-reduction at small error tolerances for the RK schemes must also be considered [3].

Nevertheless, the numerical experiments indicate that temporal accuracy close to the design order of the respective schemes is achieved over a range of accuracy levels, and the superior accuracy of the IRK64 scheme over the BDF2 scheme, which has previously been documented for static grid cases, is demonstrated for dynamic grid cases. The IRK64 scheme generally achieves equivalent accuracy using time steps which are up to an order of magnitude larger than the BDF2 scheme. On the other hand, the IRK64 scheme requires more implicit solutions per time step. In previous work, the overall cost per delivered accuracy of these schemes has been studied, indicating that the IRK schemes can be a factor of two to four times more efficient for moderate engineering-type accuracy levels [2,9] on static grids. A similar study remains to be done for dynamic mesh motion cases.

Finally, under the current approach, the extension to viscous flows appears to be straightforward, as discussed in Section 5. Future work will examine the accuracy of these schemes for viscous flow applications on dynamic meshes.

Acknowledgements

This work was partially supported by the Air Force Research Laboratory, through a subcontract administered by Ball Aerospace Corporation, by a grant from NASA Langley Research Center, and by the Wyoming NSF Epscor Program.

Appendix. Butcher coefficients for IRK64 scheme

$c_1 = 0$	0	0	0	0	0	0
$\frac{1}{2}$	$\frac{1}{4}$	$\frac{1}{4}$	0	0	0	0
$\frac{83}{250}$	$\frac{8611}{62500}$	$-\frac{1743}{31250}$	$\frac{1}{4}$	0	0	0
$\frac{31}{50}$	$\frac{5012029}{34652500}$	$-\frac{654441}{2922500}$	$\frac{174375}{388108}$	$\frac{1}{4}$	0	0
$\frac{17}{20}$	$\frac{15267082809}{155376265600}$	$-\frac{71443401}{120774400}$	$\frac{730878875}{02184768}$	$\frac{2285395}{8070912}$	$\frac{1}{4}$	0
$c_6 = 1$	$\frac{82889}{524892}$	0	$\frac{15625}{83664}$	$\frac{69875}{102672}$	$-\frac{2260}{8211}$	$\frac{1}{4}$
$c_6 = 1$	b_1	b_2	b_3	b_4	b_5	a_{66}

References

[1] T.J. Baker, Mesh movement and metamorphosis, Engineering with Computers 18 (3) (2002) 188–198.

[2] H. Bijl, M.H. Carpenter, V.N. Vatsa, C.A. Kennedy, Implicit time integration schemes for the unsteady incompressible Navier–Stokes equations: laminar flow, Journal of Computational Physics 179 (1) (2002) 313–329.

[3] J.C. Butcher, Numerical Methods for Ordinary Differential Equations, Wiley, Chichester, UK, 2003.

[4] C. Farhat, P. Geuzaine, C. Crandmont, The discrete geometric conservation law and the nonlinear stability of ALE schemes for the solution of flow problems on moving grids, Journal of Computational Physics 174 (2001) 669–694.

[5] 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.

[6] R.E. Gordnier, R.B. Melville, Transonic flutter simulations using an implicit aeroelastic solver, AIAA Journal of Aircraft 37 (5) (2000) 872–879.

[7] H. Guillard, C. Farhat, On the significance of the geometric conservation law for flow computations on moving meshes, Computer Methods in Applied Mechanics and Engineering 190 (2000) 1467–1482.

[8] C. Hirsch, Numerical Computation of Internal and External FlowsFundamentals of Numerical Discretization, vol. I, Wiley, New York, NY, 1988.

[9] G. Jothiprasad, D.J. Mavriplis, D.A. Caughey, Higher-order time integration schemes for the unsteady Navier–Stokes equations on unstructured meshes, Journal of Computational Physics 191 (2003) 542–566.

[10] C.A. Kennedy, M.H. Carpenter, Additive Runge–Kutta schemes for convection diffusion reaction equations, NASA TM-2001-211038, July 2001.

[11] B. Koobus, C. Farhat, On the implicit time integration of semi-discrete viscous fluxes on unstructured dynamic meshes, International Journal for Numerical Methods in Fluids 29 (1999) 975–996.

[12] B. Koobus, C. Farhat, Second-order time-accurate and geometrically conservative implicit schemes for flow computations on unstructured dynamic meshes, Computer Methods in Applied Mechanics and Engineering 170 (1999) 103–129.

[13] J.D. Lambert, Numerical Methods for Ordinary Differential Systems, Wiley, Chichester, UK, 1991.

[14] M. Lesoinne, C. Farhat, Geometric conservation laws for flow problems with moving boundaries and deformable meshes, and their impact on aeroelastic computations, Computer Methods in Applied Mechanics and Engineering 134 (1996) 71–90.

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

[16] V.A. Mousseau, D.A. Knoll, W.J. Rider, Physics-based preconditioning and the Newton–Krylov method for non-equilibrium radiation diffusion, Journal of Computational Physics 160 (2) (2000) 743–765.

[17] E.J. Nielsen, W.K. Anderson, Recent improvements in aerodynamic optimization on unstructured meshes, AIAA Journal 40 (6) (2002) 1155–1163.

[18] B. Nkonga, H. Guillard, Godonuv type method on non-structured meshes for three-dimensional moving boundary problems, Computational Methods in Applied Mechanical Engineering 113 (1994) 183–204.

[19] R.W. Smith, J.A. Wright, An implicit edge-based ALE method for the incompressible Navier–Stokes equations, International Journal of Numerical Methods in Fluids 43 (2003) 253–278.

[20] P.D. Thomas, C.K. Lombard, Geometric conservation law and its application to flow computations on moving grids, AIAA Journal 17 (10) (1979) 1030–1037.

[21] V. Venkatakrishnan, D.J. Mavriplis, Implicit method for the computation of unsteady flows on unstructured grids, In: Proceedings of the 12th AIAA CFD Conference, San Diego CA, June 1995. AIAA Paper 95-1705-CP.

[22] M.R. Visbal, D.V. Gaitonde, On the use of high-order finite-difference schemes on curvilinear deforming meshes, Journal of Computational Physics 181 (2002) 155–185.

[23] Z. Yang, D.J. Mavriplis, Unstructured dynamic meshes with higher-order time integration schemes for the unsteady Navier–Stokes equations, AIAA Paper 2005-1222, Jan. 2005.

[24] H. Zhang, M. Reggio, J.Y. Trepanier, R. Camarero, Discrete form of the GCL for moving meshes and its implementation in CFD schemes, Computers and Fluids 22 (1993) 9–23.