AIAA SciTech Forum
9 - 13 January 2017, Grapevine, Texas
55th AIAA Aerospace Sciences Meeting

Third-Order Inviscid and Second-Order Hyperbolic Navier-Stokes Solvers for Three-Dimensional Unsteady Inviscid and Viscous Flows

Yi Liu*and Hiroaki Nishikawa†
National Institute of Aerospace, Hampton, VA 23666

This paper presents third-order-inviscid implicit edge-based solvers for unsteady inviscid and viscous flows on unstructured tetrahedral grids. Steady third-order-inviscid solvers recently developed in NASA's FUN3D code are extended to unsteady computations with implicit time-stepping schemes. The physical time derivative is discretized by a backward-difference formula, and incorporated into the third-order edge-base scheme as a source term. In the third-order edge-based scheme, the source term needs to be discretized in space by a special formula to preserve third-order accuracy. A very economical source discretization formula is derived, and the resulting unsteady third-order unstructured-grid scheme is made completely free from second derivative computations. Developed unsteady schemes are investigated and compared for some representative test cases for unsteady inviscid and viscous flows.

I. Introduction

This paper reports extensions of third-order-inviscid edge-based schemes presented in Ref.[1] to unsteady flow problems. In the previous paper, two types of third-order spatial discretizations were considered. One is an inviscid third-order edge-based scheme with the P_1 continuous Galerkin viscous discretization. The other is a second-order hyperbolic Navier-Stokes (HNS) scheme, which achieves third-order accuracy in the inviscid terms by gradient variables used to form a hyperbolic viscous system. These schemes have been implemented in NASA's FUN3D code, and demonstrated for steady inviscid and viscous flow problems in Ref.[1]. They are here extended to unsteady computations by implicit time-stepping with the backward-difference formulas (BDF), where the physical time derivatives are incorporated as source terms and discretized in space. The resulting unsteady residual equations are solved over each physical time step by the steady solver developed in the previous work.

To extend the third-order edge-based scheme to unsteady problem, the physical time derivatives are discretized in time by BDF formulas and treated as source terms. These source terms need to be carefully discretized in space in order to preserve third-order accuracy [2,3]: first- and second-order truncation errors must vanish on irregular and regular grids, respectively. Two approaches exist for source term discretizations: an extended Galerkin formula [2], and a divergence formulation of source terms [3]. The former, however, has been found to provide third-order accuracy only in two dimensions. Also, the divergence formulation requires second derivatives of source terms and introduces complications at boundary nodes. In order to generate a truly efficient third-order scheme for unstructured tetrahedral grids, we have developed a special source term discretization formula that does not require second derivatives at all, and also can be easily applied at boundary nodes. As a result, the developed edge-based schemes achieve third-order accuracy on arbitrary tetrahedral grids without computing nor storing second derivatives for both steady and unsteady computations. Third-order accuracy will be verified for an inviscid test problem, and the third-order inviscid schemes will be tested for an unsteady viscous flow over a cylinder.

The paper is organized as follows. Section II describes the governing equations. Sections III and IV describe the temporal and spatial discretizations, respectively. Section V discusses a compatible source discretization, and the derivation of new formulas. Section VI gives a brief description of the nonlinear solver used in the numerical experiments. Section VII presents numerical results. Finally, Section VIII concludes the paper.

*Senior Research Scientist (yi.liu@nianet.org), National Institute of Aerospace, 100 Exploration Way, Hampton, VA 23666 USA, Senior Member AIAA
†Associate Research Fellow (hiro@nianet.org), National Institute of Aerospace, 100 Exploration Way, Hampton, VA 23666 USA, Associate Fellow AIAA

II. Governing Equations

We consider governing equations of the form:

$$\partial_t \mathbf{U} + \mathrm{div}\mathbf{F} = \mathbf{S}, \tag{II.1}$$

where t is the physical time, $\mathbf{U}$ is a solution vector, $\mathbf{F}$ is a flux tensor, and $\mathbf{S}$ is a source term. For inviscid flow problems, the governing equations are the Euler equations, $\mathbf{U}$ is a vector of conservative variables, $\mathbf{F}$ is the inviscid flux tensor, and $\mathbf{S} = 0$. For viscous flow problems, the governing equations are the Navier-Stokes equations, where the flux tensor consists of the inviscid and viscous parts: $\mathbf{F} = \mathbf{F}^i + \mathbf{F}^v$. In the hyperbolic Navier-Stokes method, the governing system is the HNS20 system [4, 5], where $\mathbf{U}$ has extra variables (the gradient variables) proportional to the gradients of the primitive variables but their physical time derivatives are dropped from $\partial_t \mathbf{U}$, and $\mathbf{S}$ contains source terms associated with the hyperbolic formulation. For the sake of convenience, the governing system is written as

$$\mathbf{P}^{-1}\partial_\tau \mathbf{U} + \mathrm{div}\mathbf{F} = \mathbf{S} - \partial_t \mathbf{U}, \tag{II.2}$$

where τ is a pseudo time. For the Euler equations, $\mathbf{P}$ is the identity matrix, and for the HNS20 system, $\mathbf{P}$ is a diagonal matrix that scales the equations for the extra variables [4, 5]. In implicit time-stepping schemes, the flux balance, the source term, and the physical time derivatives are fully coupled in the discrete level. The resulting global system of discrete equations is then solved for the pseudo steady state to advance the solution over each physical time step.

It is emphasized again that $\partial_t \mathbf{U}$ contains only the physical time derivatives of the conservative variables; components corresponding to the gradient variables are all zero. Hence, the HNS20 system is equivalent to the Navier-Stokes equations at any instant of time t, provided $\partial_\tau \mathbf{U} = 0$ or equivalently $\mathrm{div}\mathbf{F} - \mathbf{S} + \partial_t \mathbf{U} = 0$.

III. Temporal Discretization: Backward Difference Scheme

The physical time derivative is discretized in time by the BDF method:

$$\partial_t \mathbf{U} = \alpha \mathbf{U} + \sum_{k=1}^{m} \alpha_{n-(k-1)} \mathbf{U}^{n-(k-1)}, \tag{III.1}$$

where n is a time level, $\mathbf{U}$ denotes the solution at the next time level $n+1$, and the coefficients α and $\{\alpha_m\}$ can be easily derived by fitting a polynomial of degree m over $m+1$ consecutive time steps [6]. In this work, we consider $m = 1, 2, 3$, i.e., the first-, second-, and third-order schemes, denoted respectively by BDF1, BDF2, and BDF3 schemes. The BDF1 and BDF2 schemes are known to be unconditionally stable (A-stable), but the BDF3 scheme is conditionally stable.

The BDF term is treated as a source term, and incorporated into the governing system as

$$\mathbf{P}^{-1}\partial_\tau \mathbf{U} + \mathrm{div}\mathbf{F} = \mathbf{S} - \mathbf{G}, \tag{III.2}$$

where

$$\mathbf{G} = \alpha \mathbf{U} + \sum_{k=1}^{m} \alpha_{n-(k-1)} \mathbf{U}^{n-(k-1)}. \tag{III.3}$$

Note, again, that the physical time derivatives are present only in the continuity, momentum, and energy equations, and thus the components of $\mathbf{G}$ corresponding to the gradient variables are all zero in the HNS20 system.

IV. Spatial Discretization: Node-Centered Edge-Based Scheme

The governing system is discretized on a tetrahedral grid by the node-centered edge-based method:

$$V_j \frac{d\mathbf{U}_j}{d\tau} = -\mathbf{P}_j \left(\sum_{k \in \{k_j\}} \mathbf{\Phi}_{jk} A_{jk} - \mathcal{L}_j(\mathbf{S} - \mathbf{G}) \right), \tag{IV.1}$$

Scheme	Discretization						Jacobian	
	Inviscid			Viscous			Inviscid	Viscous
	Flux		LSQ $(\rho, \mathbf{v}, p)$	Flux		LSQ $(\mathbf{r}, \mathbf{g}, \mathbf{q})$		
FUN3D	Roe(2nd)	: 2	Linear	Galerkin(2nd)	: 1	None	Van Leer	Exact
FUN3D-i3rd	Roe(3rd)	: 3	Quadratic	Galerkin(2nd)	: 1	None	Van Leer	Exact
HNS20	Roe(3rd)	: 2	C-quadratic	Upwind(2nd)	: 2	Linear	Van Leer	Upwind

Table 1: Summary of discretizations and Jacobians. The number on the right side of each colon indicates the level of neighbors contributing to the residual at a node: 1 = up to the neighbors, 2 = up to neighbors of the neighbors, 3 = up to neighbors of the neighbors of the neighbors. Order of accuracy is indicated by 2nd and 3rd in the parentheses.

where $\mathbf{P}_j$ is the diagonal scaling matrix evaluated at a node j, $\mathcal{L}_j$ denotes a source discretization operator, V_j is the measure of the dual control volume around the node j, $\{k_j\}$ is a set of neighbors of j, $\mathbf{\Phi}_{jk}$ is a numerical flux, and A_{jk} is the magnitude of the directed area vector, which is a sum of the dual-triangular faces associated with all tetrahedral elements sharing the edge (see Figure 1). The numerical flux, which is defined as a sum of the inviscid and viscous fluxes, is computed for a pair of solution values linearly reconstructed at the edge midpoint. The linear reconstruction is performed by using solution gradients computed by a linear/quadratic LSQ fit for second/third-order accuracy. The edge-based discretization is applicable to arbitrary three-dimensional grids, but can be formally second- and third-order accurate only on tetrahedral grids [7, 8]. In this work, we consider only tetrahedral grids.

In the HNS method, the edge-based discretization is applied to both the inviscid and viscous terms. In this work, we consider the improved version of HNS20-I [1], which is denoted by HNS20 in this paper. The HNS20 scheme employs compact quadratic LSQ gradients (c-quadratic) in the reconstruction for the primitive variables, and thereby achieves third-order accuracy in the inviscid terms [1]. For conventional schemes, we consider the default FUN3D inviscid scheme and the third-order inviscid scheme, which are designated as FUN3D and FUN3D-i3rd, respectively. In both FUN3D and FUN3D-i3rd, the Galerkin discretization of the viscous terms, which can be implemented as a loop over edges for tetrahedral grids, is added for viscous flow problems. These discretizations are summarized in Table 1.

V. Source Term Discretization for Third-Order Accuracy

V.A. Compatibility Condition

For second-order accuracy, it suffices to discretize the source terms by the point quadrature:

$$\mathcal{L}_j(\mathbf{S} - \mathbf{G}) = (\mathbf{S}_j - \mathbf{G}_j)V_j. \tag{V.1}$$

However, the point quadrature is not compatible with the third-order edge-based scheme [3]. As discussed in Refs.[3, 9], the edge-based scheme achieves third-order accuracy by a second-order truncation error containing derivatives of the target equation. For a regular tetrahedral grid as shown in Figure 2, which is constructed by dividing a hexahedral element into six tetrahedra, the truncation error of the inviscid terms is given by

$$\mathcal{T}_j = \mathrm{div}\mathbf{F}^i - \frac{h^2}{12}\left[\partial_{xx}\left(\mathrm{div}\mathbf{F}^i\right) + \partial_{yy}\left(\mathrm{div}\mathbf{F}^i\right) + \partial_{zz}\left(\mathrm{div}\mathbf{F}^i\right) - \partial_{xy}\left(\mathrm{div}\mathbf{F}^i\right) - \partial_{yz}\left(\mathrm{div}\mathbf{F}^i\right) + \partial_{zx}\left(\mathrm{div}\mathbf{F}^i\right)\right] + O(h^3), \tag{V.2}$$

where h denotes the uniform spacing of the hexahedral grid from which the tetrahedral grid is derived. The second-order error term vanishes because the exact solution satisfies $\mathrm{div}\mathbf{F}^i = 0$, and it leads to $\mathcal{T}_j = O(h^3)$. This mechanism, often called the residual property and shared by other methods such as the residual-compact method [10] and the residual-distribution method [11], implies that a careful discretization is necessary if other terms such as source terms are present. If source terms are present in the target equation, then a special quadrature formula is required for the source term to preserve third-order accuracy. The source terms $\mathbf{S}$ and $\mathbf{G}$

must be discretized in such a way that it yields the following second-order truncation error:

$$\mathcal{T}_j = \mathrm{div}\mathbf{F}^i - \mathbf{S}_j + \mathbf{G}_j - \frac{h^2}{12}\left[\partial_{xx}\mathbf{r}_j + \partial_{yy}\mathbf{r}_j + \partial_{zz}\mathbf{r}_j - \partial_{xy}\mathbf{r}_j - \partial_{yz}\mathbf{r}_j + \partial_{zx}\mathbf{r}_j\right] + O(h^3), \tag{V.3}$$

where

$$\mathbf{r}_j = \mathrm{div}\mathbf{F}^i - \mathbf{S} + \mathbf{G}. \tag{V.4}$$

so that the second-order error vanishes for exact solutions satisfying $\mathrm{div}\mathbf{F}^i - \mathbf{S} + \mathbf{G} = 0$. On irregular grids, the truncation error will have a first-order error term. To achieve third-order accuracy on unstructured grids, this first-order error must be eliminated. Therefore, source term discretization formulas need to satisfy two conditions: the compatibility condition on regular grids and the elimination of first-order truncation errors.

In two dimensions, two techniques are available. One is an extended Galerkin formula [2], and the other is a divergence formulation method for source terms [3]. These techniques, however, require computations and storage of second derivatives of the source term. Although a particular divergence formulation requires fewer second derivatives, it still takes substantial resources to compute and store second derivatives, especially for three-dimensional unsteady applications. Moreover, in our study, the extended Galerkin formula [2] has been found to work only in two dimensions, and does not provide third-order accuracy in three dimensions. The divergence formulation approach can be easily extended to three dimensions, but it requires a careful discretization at boundary nodes. In order to reduce the cost of unsteady computations and also simplify the discretization at boundary nodes, we have developed a family of new source discretization formulas for three dimensional grids. In particular, we have discovered a source discretization formula that does not require second derivatives at all. This is a remarkable discovery since it makes the third-order edge-based scheme completely free from expensive computations and storage of second derivatives. In the next section, we describe the new approach and various economical formulas; a comprehensive study, including two dimensional formulas and accuracy at boundary nodes, is presented in Ref.[12].

V.B. Accuracy-Preserving Source Term Quadrature Formulas

In this study, we only consider third-order accuracy in the inviscid terms, and therefore $\mathbf{S}$ from HNS method is discretized by the point quadrature formula (V.1) and we focus on the discretization of $\mathbf{G}$. We seek a general accuracy-preserving source term quadrature in the form:

$$\int_{V_j} \mathbf{G}\, dV = \sum_{k\in\{k_j\}} \frac{1}{2}(\mathbf{G}_L + \mathbf{G}_R)V_{jk}, \tag{V.5}$$

where

$$\partial_{jk} = \Delta\mathbf{x}_{jk}\cdot(\partial_x,\partial_y,\partial_z) = (x_k - x_j, y_k - y_j, z_k - z_j)\cdot(\partial_x,\partial_y,\partial_z), \tag{V.6}$$

$$V_{jk} = \frac{1}{6}(\Delta\mathbf{x}_{jk}\cdot\mathbf{n}_{jk}), \quad V_j = \sum_{k\in\{k_j\}} V_{jk}, \tag{V.7}$$

$$\mathbf{G}_L = a_L\mathbf{G}_j + b_L\partial_{jk}\mathbf{G}_j + c_L\partial_{jk}^2\mathbf{G}_j, \tag{V.8}$$

$$\mathbf{G}_R = a_R\mathbf{G}_k + b_R\partial_{jk}\mathbf{G}_k + c_R\partial_{jk}^2\mathbf{G}_k. \tag{V.9}$$

Expanding $\mathbf{G}_k$, $\partial_{jk}\mathbf{G}_k$, and $\partial_{jk}^2\mathbf{G}_k$ around the node j, and neglecting high-order terms, we obtain

$$\sum_{k\in\{k_j\}} \frac{\mathbf{G}_L+\mathbf{G}_R}{2}V_{jk} = \sum_{k\in\{k_j\}}\left(\frac{a_L+a_R}{2}\mathbf{G}_j + \frac{a_R+b_L+b_R}{2}\partial_{jk}\mathbf{G}_j + \frac{2(b_R+c_R+c_L)+a_R}{4}\partial_{jk}^2\mathbf{G}_j\right)V_{jk}. \tag{V.10}$$

Two conditions are obtained by consistency and elimination of the first-order error term:

$$a_L + a_R = 2, \quad a_R + b_L + b_R = 0. \tag{V.11}$$

	Grid type	a_L	b_L	c_L	a_R	b_R	c_R
Regular	Regular Tetrahedra	13/5	0	0	$-3/5$	0	0
Compact	Arbitrary Tetrahedra	13/5	3/5	0	$-3/5$	0	0
Economical(1)	Arbitrary Tetrahedra	1	$-1/5$	0	1	$-4/5$	0
One-sided	Arbitrary Tetrahedra	2	0	-3/10	0	0	0
Symmetric	Arbitrary Tetrahedra	1	$-1/2$	$-3/20$	1	$-1/2$	$-3/20$

Table 2: Summary of accuracy-preserving source term quadrature formulas in three dimensions. Top one is for the regular grid, the next two do not require second derivatives of the source term; the bottom two require the second derivatives. The value in the parenthesis indicates the chosen value of a_L to generate the formula from the one-parameter family of formulas satisfying $c_L = c_R = 0$.

Another condition comes from requiring that the quadratic error term yields compatible second-order error terms on a regular tetrahedral grid. Expanding the source quadrature (V.5) on the regular grid, we find

$$\frac{1}{V_j}\sum_{k\in\{k_j\}}\frac{\mathbf{G}_L+\mathbf{G}_R}{2}V_{jk}=\mathbf{G}_j+\frac{5h^2}{36}(2c_L+2b_R+2c_R+a_R)(\partial_{xx}\mathbf{G}_j+\partial_{yy}\mathbf{G}_j+\partial_{zz}\mathbf{G}_j-\partial_{xy}\mathbf{G}_j-\partial_{yz}\mathbf{G}_j+\partial_{zx}\mathbf{G}_j). \tag{V.12}$$

The compatibility condition (V.3) requires

$$\frac{5}{36}\left(2c_L+2b_R+2c_R+a_R\right)=-\frac{1}{12}, \tag{V.13}$$

or

$$a_R=-2(b_R+c_R+c_L)-\frac{3}{5}. \tag{V.14}$$

Therefore, the quadrature formula (V.5) achieves third-order accuracy on arbitrary tetrahedral grids if the coefficients satisfy the three conditions:

$$a_L+a_R=2,\quad a_R+b_L+b_R=0,\quad a_R=-2(b_R+c_R+c_L)-\frac{3}{5}. \tag{V.15}$$

These conditions form an underdetermined system for the unknowns: a_L, b_L, c_L, a_R, b_R, and c_R, thus defining a three-parameter family of accuracy preserving source discretization formulas.

For example, if we seek a symmetric formula with

$$a_L=a_R,\quad b_L=b_R,\quad c_L=c_R, \tag{V.16}$$

then

$$a_L=a_R=1,\quad b_L=b_R=-\frac{1}{2},\quad c_L=c_R=-\frac{3}{20}. \tag{V.17}$$

A one-sided formula is obtained by imposing

$$a_R=b_R=c_R=0, \tag{V.18}$$

resulting in

$$a_L=2,\quad b_L=0,\quad c_L=-\frac{3}{10},\quad a_R=b_R=c_R=0. \tag{V.19}$$

There exists a one-parameter family of economical formulas that do not require second derivatives, defined by

$$c_L=c_R=0. \tag{V.20}$$

For example, if we choose $a_L=1$, then

$$a_L=a_R=1,\quad b_L=-\frac{1}{5},\quad b_R=-\frac{4}{5},\quad c_L=c_R=0. \tag{V.21}$$

These economical formulas bring a substantial saving in computational cost in three dimensions since there is no need to compute and store nine (or six if symmetry is enforced) second derivatives in three dimensions. Furthermore, a compact formula can be found by requiring no second derivatives at both nodes and no gradients at the neighbor:

$$b_R = c_L = c_R = 0, \tag{V.22}$$

which leads to

$$a_L = \frac{13}{5}, \quad a_R = -\frac{3}{5}, \quad b_L = \frac{3}{5}, \quad b_R = c_L = c_R = 0. \tag{V.23}$$

This formula is closely related to a formula for regular grids. On regular grids, the second condition in Equation (V.15) is redundant because the first-order error term identically vanishes on regular grids, and therefore we can seek a formula with no derivatives at all by imposing

$$b_L = b_R = c_L = c_R = 0, \tag{V.24}$$

which leads to

$$a_L = \frac{13}{5}, \quad a_R = -\frac{3}{5}, \quad b_L = b_R = c_L = c_R = 0. \tag{V.25}$$

Therefore, the compact formula (V.23) can be considered as a simple extension of this regular formula to irregular grids. The derived formulas are summarized in Table 2.

The most efficient formula for unsteady computations would be the compact formula since it does not require second derivatives nor the source gradients at the neighbor nodes, leading to a compact source discretization depending only on the neighbors. To implement this, the gradient of $\mathbf{G}$,

$$\nabla \mathbf{G} \quad = \quad \alpha \nabla \mathbf{U} + \sum_{k=1}^{m} \alpha_{n-(k-1)} \nabla \mathbf{U}^{n-(k-1)}, \tag{V.26}$$

needs to be computed and stored at each node. No extra computations are required since the solution gradients are already available. However, FUN3D-i3rd needs to store the solution gradients at previous time steps. On the other hand, recall that the vector $\mathbf{G}$ is nonzero only in the first five components in HNS20, and the gradient variables are equivalent to the gradients of the primitive variables. Therefore, the gradient $\nabla \mathbf{U}$ can be simply replaced by the gradients variables, and thus HNS20 does not require additional work nor storage to evaluate $\nabla \mathbf{G}$. As a result, FUN3D-i3rd and HNS20 have the same storage requirement in the back planes.

Finally, the residual is written as

$$V_j \frac{d\mathbf{U}_j}{d\tau} = -\mathbf{P}_j \left(\sum_{k \in \{k_j\}} \mathbf{\Phi}_{jk} A_{jk} + \frac{1}{2}(\mathbf{G}_L + \mathbf{G}_R) V_{jk} - \mathbf{S}_j V_j \right), \tag{V.27}$$

which guarantees the compatibility for the inviscid terms and the physical time derivative terms. In future, we will apply a similar discretization to the source term $\mathbf{S}$ in order to achieve third-order accuracy for both the inviscid and viscous terms.

VI. Nonlinear Solver

The discretization yields a global system of residual equations, which is solved by an implicit defect-correction solver similar to the one described specifically for the HNS schemes in Ref.[5]. The linear relaxation is performed by a multi-color Gauss-Seidel scheme available in FUN3D. The Jacobian matrix is constructed exactly for the Galerkin viscous discretization, and approximately for others by the exact linearization of first-order-accurate residuals. The inviscid Jacobian is constructed by the derivative of Van Leer's flux vector splitting scheme for the inviscid terms. The Jacobian consists of 5×5 blocks for FUN3D and FUN3D-i3rd, and 20×20 blocks for HNS20. The Jacobian is updated based on an efficient algorithm available in FUN3D, which automatically adjusts the Jacobian-update-frequency based on residual reduction. The pseudo time step is defined locally with CFL= 200 for all cases unless otherwise stated. The sub-iteration over each physical time step is terminated by the error control module in FUN3D [6], with a specified maximum number of iterations, which is set to be 50. The tolerance is set as one order of magnitude reduction in an estimated error.

VII. Numerical Results

VII.A. Inviscid Moving Vortex

Third-order inviscid accuracy has been verified with the inviscid vortex problem, which is widely used for accuracy verification studies (see, e.g., Ref.[13]). The solution is two-dimensional, but we consider a 3D rectangular domain $(x, y, z) \in [-20, 20] \times [0, y_{max}] \times [-10, 10]$ while the v-velocity, i.e., the velocity component in the y-direction, is set to be zero.

First, a series of regular tetrahedral meshes have been generated with three layers in y-direction: 241×3×121 nodes ($y_{max} = 4.0$), 321×3×161 nodes ($y_{max} = 3.0$), 641×3×321 nodes ($y_{max} = 1.5$), 961×3×481 nodes ($y_{max} = 1.0$). The coarsest grid is shown in Figure 3. Boundary conditions are the inflow at $x = -20.0$, the back-pressure condition at $x = 20.0$, a far-field characteristic condition at $z = -10$ and $z = 10$, and a periodic condition for the planes at $y = 0$ and $y = y_{max}$. This is a pure inviscid problem, but we use Navier-Stokes solvers with $M_\infty = 0.5$ and $Re_\infty = 10^6$, where Re_∞ is based on the unit grid-length. The solution is advanced in time by BDF3 with $\Delta t = 0.02$; BDF1 and BDF2 are used at the first and second time levels, respectively.

The accuracy of the algorithm is examined with the error convergence results at $t = 10$ as shown in Figure 4. As expected, FUN3D gives second-order accuracy for all variables, especially when the grid is refined enough. FUN3D-i3rd and HNS20 successfully yield third-order accuracy, verifying the new source term quadrature formula for regular tetrahedral meshes. We have found that the viscosity for the artificial hyperbolic diffusion in the continuity equation as defined by $\nu_\rho = V_{min}$ [1], where V_{min} is the minimum control volume in a given grid, is not small enough for the grids used here, so that the errors in the density can be quite large although the order of error convergence is third-order as designed (see Figure 5). To avoid larger errors, we modify ν_ρ as follows:

$$\nu_\rho = \min\left(10^{-6}, V_{min}\right), \tag{VII.1}$$

so that ν_ρ will never be larger than 10^{-6}. This formula has been found to work well for this test problem as shown in Figure 4(b). However, it is not optimal nor general in any sense, and remains a subject for future study.

In order to check the error propagation for second-order and third-order methods, we take the regular mesh of 641x321x3 with the inflow and outflow boundaries changed to periodic boundaries so that the vortex can travel at a long distance. The vortex is started at $x = -10$ moving along $z = 0$ line. With the vortex moving velocity of 0.5, it takes a time of 80 to finish one cycle. Three time-step snapshots have been taken for comparison, which are $t = 10$, 400(= 5 cycles) and 800(=10 cycles). The pressure contours at each time step, as well as the pressure and z-velocity distribution data at a slice through the center of vortex ($z = 0$ line), have been compared between FUN3D and FUN3D-i3rd. At $t = 10$, as shown in Figures 6 and 7, the vortex just moved a distance of 5, and there is no much difference between FUN3D and FUN3D-i3rd. However at $t = 400$ after 5 cycles, the pressure contours and the sliced data have shown some difference between these two methods as shown in Figures 8 and 9. The vortex has moved away from the center line along the mesh bias direction for FUN3D as shown in 8(a), whereas the FUN3D-i3rd simulation still keeps the vortex moving accurately along the center line. Also, the vortex location predicted by FUN3D is behind the exact location of $x = -10$ at $t = 400$ as shown in Figure 9(a), while the vortex location predicted by FUN3D-i3rd is still accurate. Moreover, the dispersive error at the high velocity gradient area is also much larger for FUN3D than FUN3D-i3rd as shown in Figure 9(b). A similar trend can be seen at $t = 800$ after 10 cycles, as shown in Figures 10 and 11. The dispersive error in FUN3D gets larger while FUN3D-i3rd seems free from such dispersive errors. These results are expected from the fact that leading truncation errors are dispersive for second-order schemes, and dissipative for third-order schemes. The dissipative error of the predicted pressure and w-velocity is only slightly improved by FUN3D-i3rd apparently because the leading dissipative-truncation error is fourth order for both FUN3D and FUN3D-i3rd.

Next a series of perturbed irregular meshes has been generated. These meshes have the same size and dimension as the regular mesh, but each boundary plane is randomly triangulated. The coarsest grid is shown in Figure 12. The error convergence results for the irregular meshes at t=10 are shown in Figure 13. As expected, FUN3D gives second-order accuracy for all variables. FUN3D-i3rd and HNS20 successfully yield third-order accuracy, verifying that the new source term quadrature formula works for irregular mesh too.

VII.B. Unsteady Viscous Flow Past a Cylinder

We consider a laminar flow over a cylinder with $M_\infty = 0.2$ and $Re_\infty = 150$. The Reynolds number is defined based on the diameter of the cylinder, D. This is a widely used test case for the Navier-Stoke equations (see,

e.g., Refs.[14, 15, 16]), and it is known to generate periodic vortex shedding and forces. The domain is 3D, and the grid is composed of irregular tetrahedra with 401 nodes on the cylinder surface and 201 nodes in the radial direction to the outer boundary (see Figure 14). The first off-the-wall node is located at the distance of $0.0001D$ from the surface. The outer boundary is located at the distance of $100D$. The viscous wall condition and the free stream condition are applied to the inner and outer boundaries, respectively. The solution is advanced in time with $\Delta t = 0.02$, up to the final time $t = 400$ (i.e., 20,000 time steps). BDF2 is used as the time integration scheme since BDF3 is not stable for this case at $\Delta t = 0.02$.

Figure 15 shows the pressure contours at $t = 400$. The irregular behavior of the pressure contours produced by the FUN3D scheme is eliminated by the third-order inviscid schemes, FUN3D-i3rd and HNS20. A similar observation can be made for the vorticity magnitude contours as shown in Figure 16. These results show that the third-order-inviscid improvements reported for steady problems in Ref.[1] carry over to unsteady problems.

Figure 17 shows the sub-iteration history and the time-history of the drag coefficient. As shown in Figure 17(a), the HNS20 scheme achieves the tolerance in 6 sub-iterations on average, whereas the FUN3D and FUN3D-i3rd schemes experience a slow-down and take 30-50 sub-iterations to meet the tolerance. Figure 17(b) shows the time-history of the drag coefficient. All schemes generate a periodic behavior as expected, with slight differences in the magnitude.

VIII. Conclusions

In this paper, third-order edge-based schemes have been developed for unsteady problems. The physical time derivatives are discretized by the backward difference formulas in time, and incorporated into the third-order schemes as source terms. The time-derivative source terms are then discretized in space to preserve third-order accuracy by a new source term quadrature formula. The new formula does not require computations nor storage of second derivatives. As a result, third-order unstructured-grid schemes, which do not require second derivatives at all, have been developed. These schemes are constructed with two different approaches. One is to apply the third-order edge-bases scheme to the inviscid terms with a quadratic LSQ fit with the Galerkin viscous scheme for the viscous terms. The other is to employ the hyperbolic Navier-Stokes method, where the solution gradients are taken as additional variables, and used in the solution reconstruction to achieve third-order accuracy in the inviscid terms. In the former, the solution gradients need to be stored at previous time levels to implement the spatial discretization of the time derivatives. In the latter, additional storage for the solution gradients are not needed since they are already stored as additional variables. Numerical experiments are presented to verify third-order accuracy, and demonstrate advantages of the third-order edge-based schemes for unsteady problems. Future work includes further verification studies, efficient implementations, alternative time integration schemes, and applications to high-Reynolds-number unsteady turbulent flows.

Acknowledgments

This work was funded by the U.S. Army Research Office under the contract/grant number W911NF-12-1-0154 with Dr. Matthew Munson as the program manager, and by NASA under Contract No. NNL09AA00A with Dr. Veer N. Vatsa as the technical monitor.

References

[1]Liu, Y. and Nishikawa, H., "Third-Order Inviscid and Second-Order Hyperbolic Navier-Stokes Solvers for Three-Dimensional Inviscid and Viscous Flows," *46th AIAA Fluid Dynamics Conference*, AIAA Paper 2016-3969, Washington, D.C., 2016.

[2]Pincock, B. and Katz, A., "High-Order Flux Correction for Viscous Flows on Arbitrary Unstructured Grids," *J. Sci. Comput.*, Vol. 61, 2014, pp. 454–476.

[3]Nishikawa, H., "Divergence Formulation of Source Term," *J. Comput. Phys.*, Vol. 231, 2012, pp. 6393–6400.

[4]Nishikawa, H., "Alternative Formulations for First-, Second-, and Third-Order Hyperbolic Navier-Stokes Schemes," *Proc. of 22nd AIAA Computational Fluid Dynamics Conference*, AIAA Paper 2015-2451, Dallas, TX, 2015.

[5]Nakashima, Y., Watanabe, N., and Nishikawa, H., "Hyperbolic Navier-Stokes Solver for Three-Dimensional Flows," *54th AIAA Aerospace Sciences Meeting*, AIAA Paper 2016-1101, San Diego, CA, 2016.

[6]Vatsa, V. and Carpenter, M., "Higher-Order Temporal Schemes with Error Controllers for Unsteady Navier-Stokes Equations," *Proc. of 17th AIAA Computational Fluid Dynamics Conference*, AIAA Paper 2005-5245, Toronto, 2005.

[7]Diskin, B. and Thomas, J. L., "Accuracy Analysis for Mixed-Element Finite-Volume Discretization Schemes," *NIA Report No. 2007-08*, 2007.

[8]Nishikawa, H., "Beyond Interface Gradient: A General Principle for Constructing Diffusion Schemes," *Proc. of 40th AIAA Fluid Dynamics Conference and Exhibit*, AIAA Paper 2010-5093, Chicago, 2010.

[9]Nishikawa, H., "First, Second, and Third Order Finite-Volume Schemes for Advection-Diffusion," *J. Comput. Phys.*, Vol. 273, 2014, pp. 287–309.

[10]Corre, C., Hanss, G., and Lerat, A., "A Residual-Based Compact Schemes for the Unsteady Compressible Navier-Stokes Equations," *Comput. Fluids*, Vol. 34, 2005, pp. 561–580.

[11]Nishikawa, H. and Roe, P. L., "On High-Order Fluctuation-Splitting Schemes for Navier-Stokes Equations," *Computational Fluid Dynamics 2004*, edited by C. Groth and D. W. Zingg, Springer-Verlag, 2004, pp. 799–804.

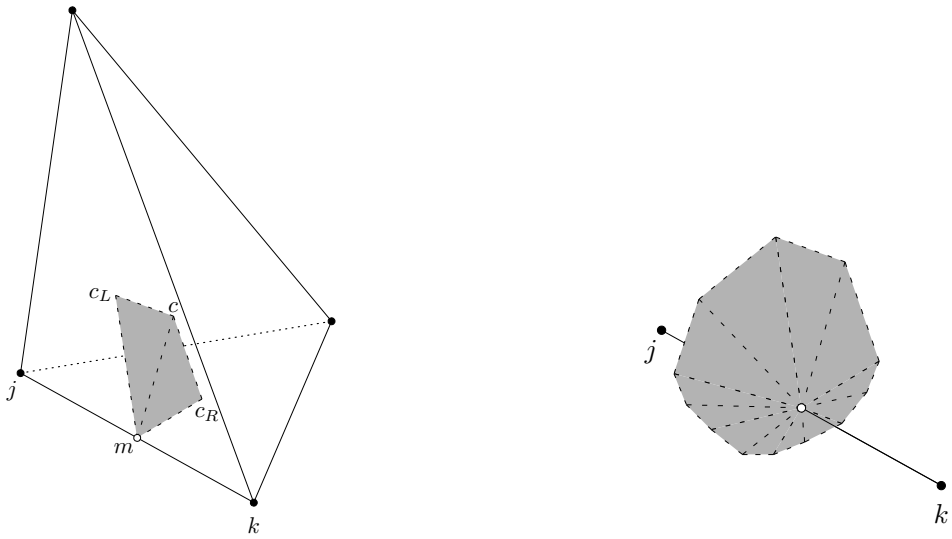
[12]Nishikawa, H. and Liu, Y., "Accuracy-Preserving Source Term Quadrature for Third-Order Edge-Based Discretization," 2016, in review.

[13]Yee, H. C., Sandham, N. D., and Djomehri, M. J., "Low-Dissipative High-Order Shock-Capturing Methods Using Characteristic-Based Filters," *J. Comput. Phys.*, Vol. 150, 1999, pp. 199–238.

[14]Visbal, M. R., "Evaluation of an Implicit Navier-Stokes Solver for Some Unsteady Separated Flows," *Proc. of AIAA/ASME 9th Fluid Mechanics, Plasma Dynamics and Laser Conference*, AIAA Paper 1986-1053, Atlanta, GA, 1986.

[15]Green, B. E., Czerwiec, R., Cureton, C., Lillian, C., Kernazhitskiy, S., Eymann, T., Torres, J., Bergeron, K., and Decker, R., "Evaluation of Flow Solver Accuracy using Five Simple Unsteady Validation Cases," *Proc. of 49th AIAA Aerospace Sciences Meeting*, AIAA Paper 2011-29, Orlando, Florida, 2011.

[16]Cox, C., Liang, C., and Plesniak, M. W., "A Flux Reconstruction Solver for Unsteady Incompressible Viscous Flow using Artificial Compressibility with Implicit Dual Time Stepping," *54th AIAA Aerospace Sciences Meeting*, AIAA Paper 2016-1827, San Diego, CA, 2016.



(a) Dual face contributions from an adjacent tetrahedral element to the edge $[j, k]$.

(b) Total dual face at the edge $[j, k]$.

Figure 1: Dual face contribution at the edge $[j, k]$. A numerical flux is evaluated at the midpoint of the edge m, indicated by an open circle. The centroid of the tetrahedral element is denoted by c, and the centroids of the two adjacent triangles are denoted by c_L and c_R.

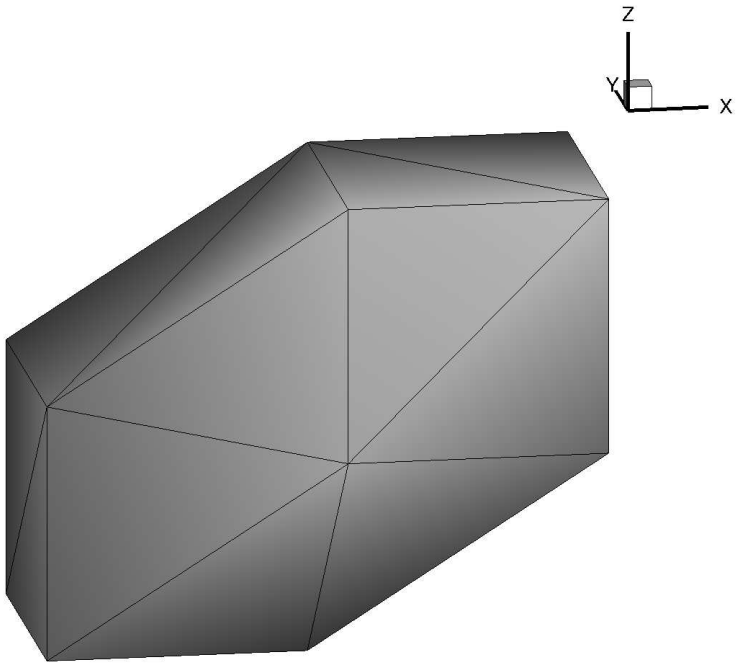
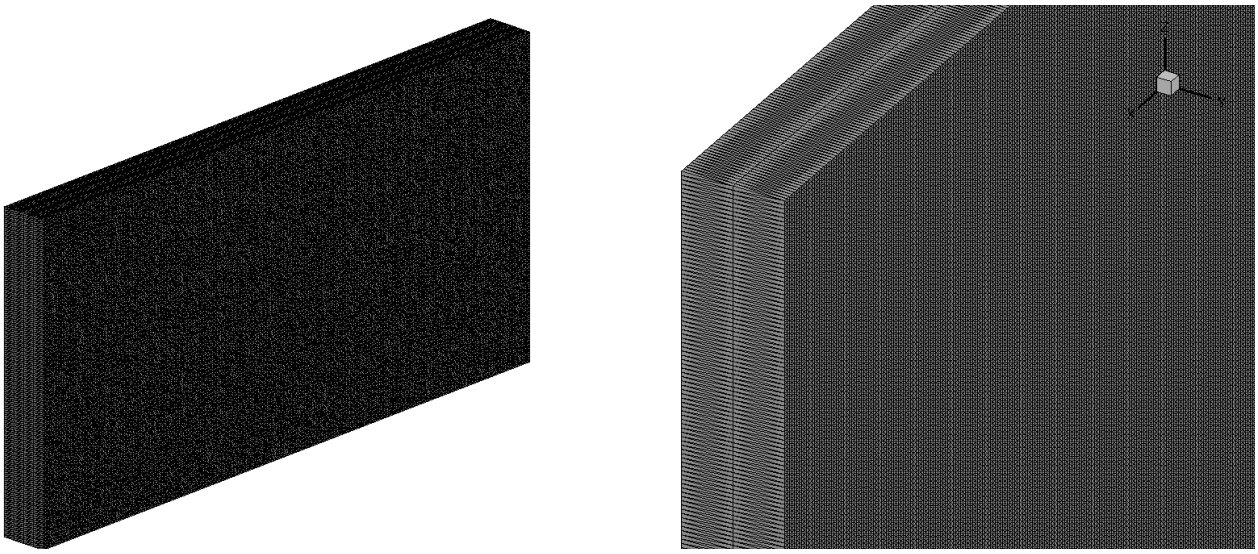
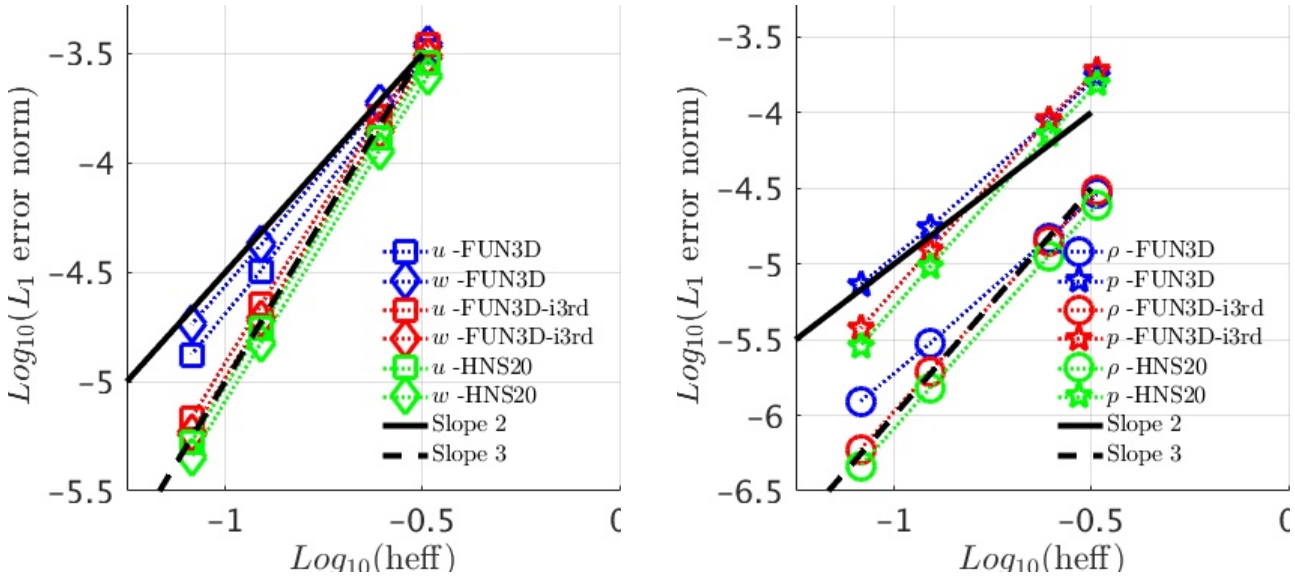


Figure 2: Regular tetrahedral grid.



(a) Tetrahedral grid. (b) Regular local tetrahedral mesh.

Figure 3: Regular tetrahedra grid for the inviscid moving vortex calculated by Navier-Stokes solvers with $Re_\infty = 10^6$.



(a) U and W Velocities. (b) Density and Pressure.

Figure 4: Accuracy verification results for the inviscid moving vortex problem for regular grid.

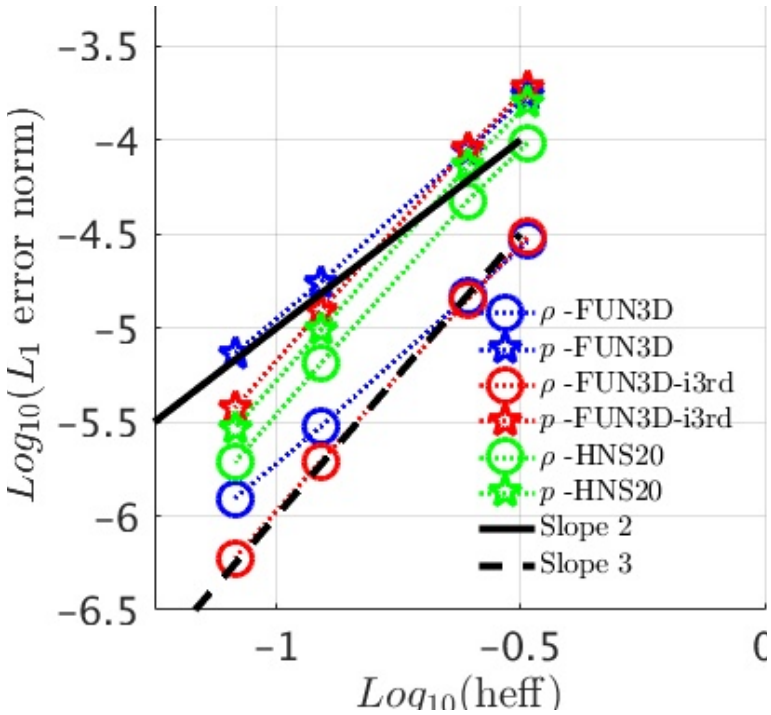
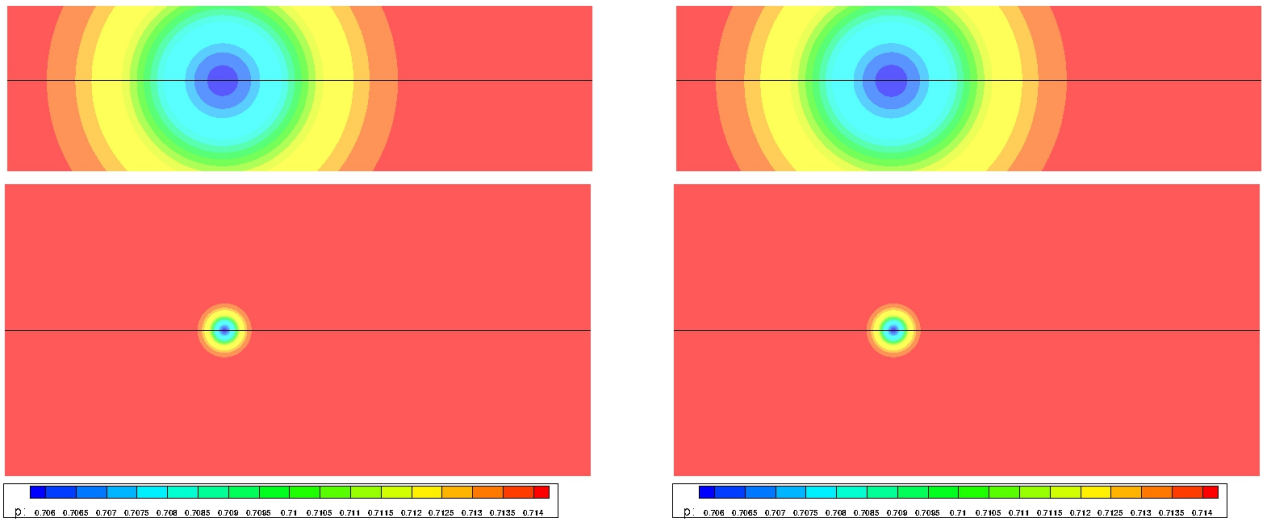
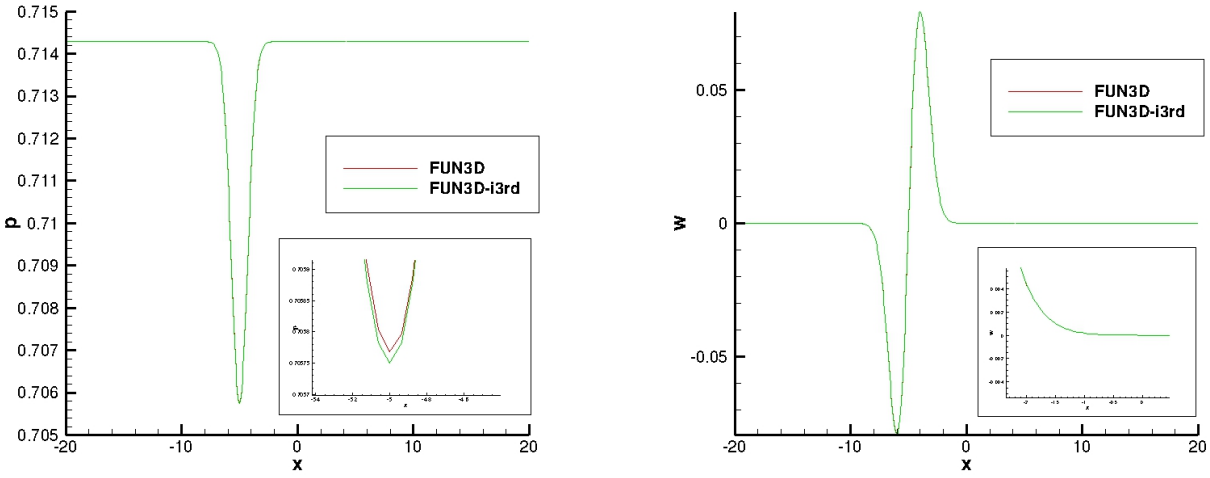


Figure 5: The errors for density and pressure for regular grid with large ν_ρ for HNS20 .



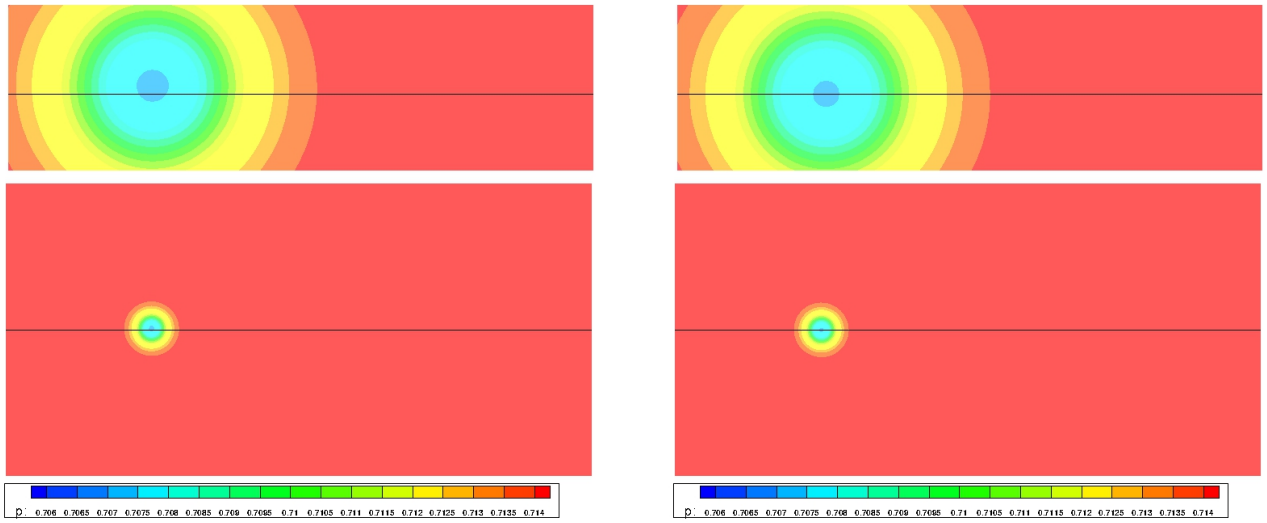
(a) FUN3D. (b) FUN3D-i3rd.

Figure 6: Pressure contours of the moving vortex at t=10.



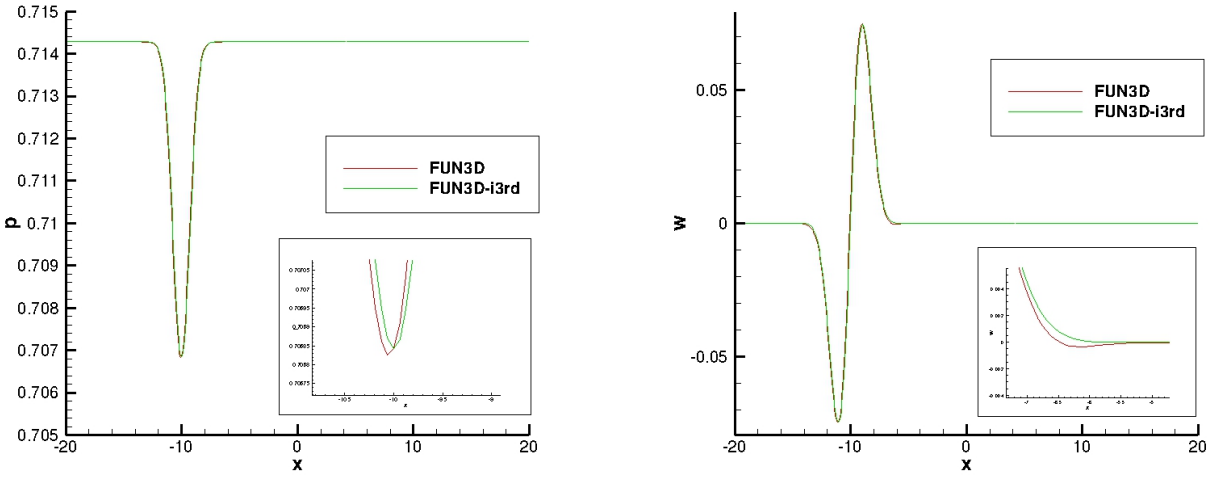
(a) Pressure. (b) W-velocity.

Figure 7: Slice data along z=0 line of the moving vortex at t=10.



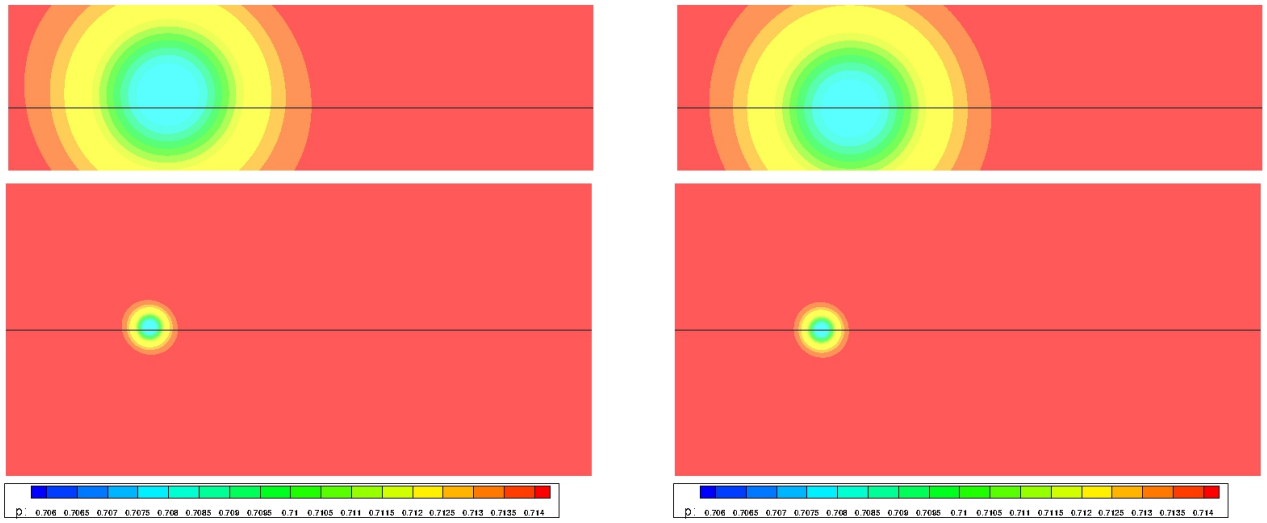
(a) FUN3D. (b) FUN3D-i3rd.

Figure 8: Pressure contours of the moving vortex at t=400.



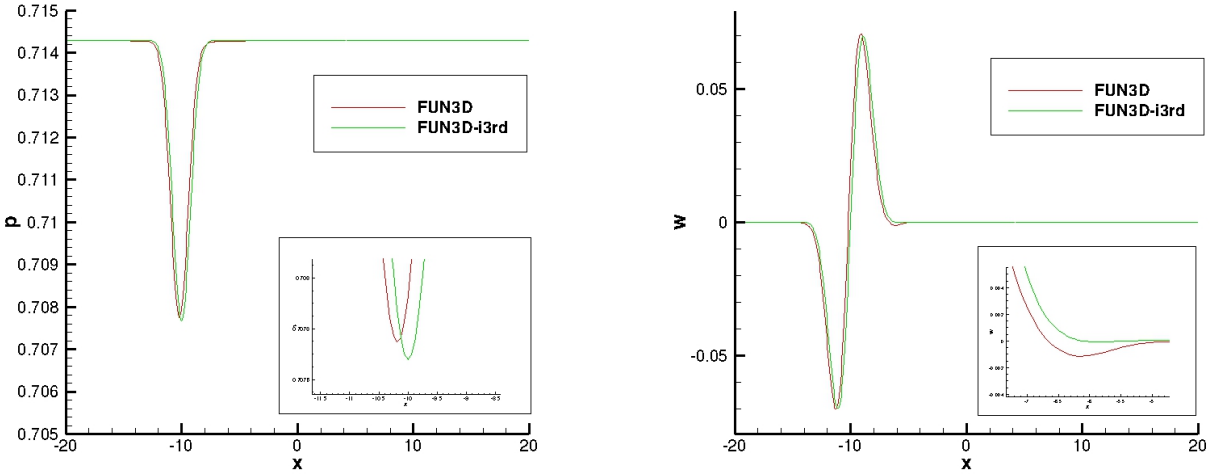
(a) Pressure. (b) W-velocity.

Figure 9: Slice data along z=0 line of the moving vortex at t=400.



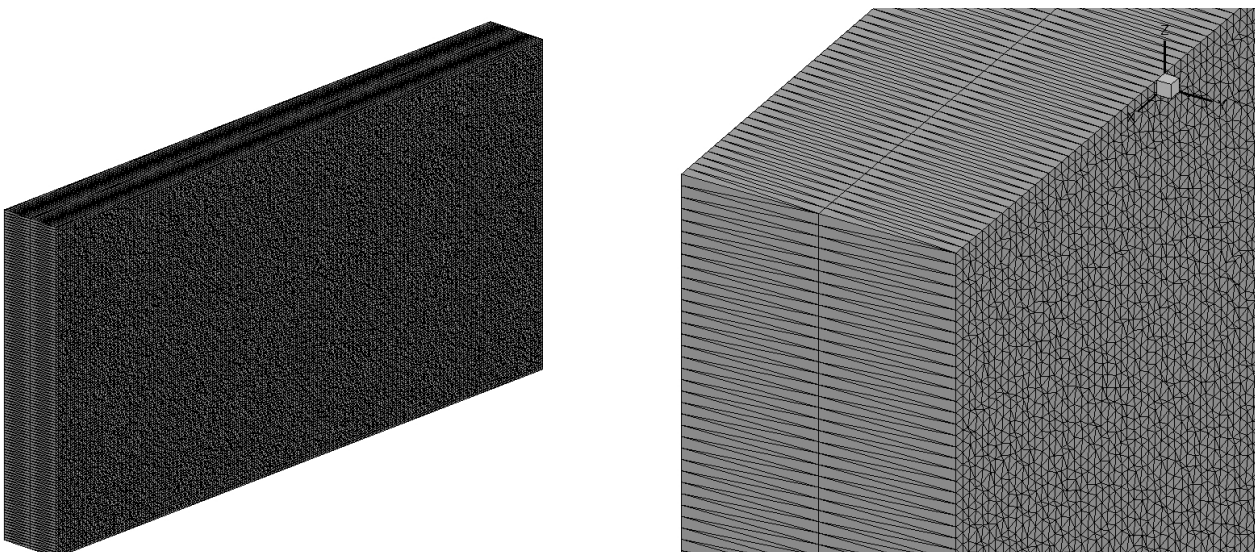
(a) FUN3D. (b) FUN3D-i3rd.

Figure 10: Pressure contours of the moving vortex at t=800.



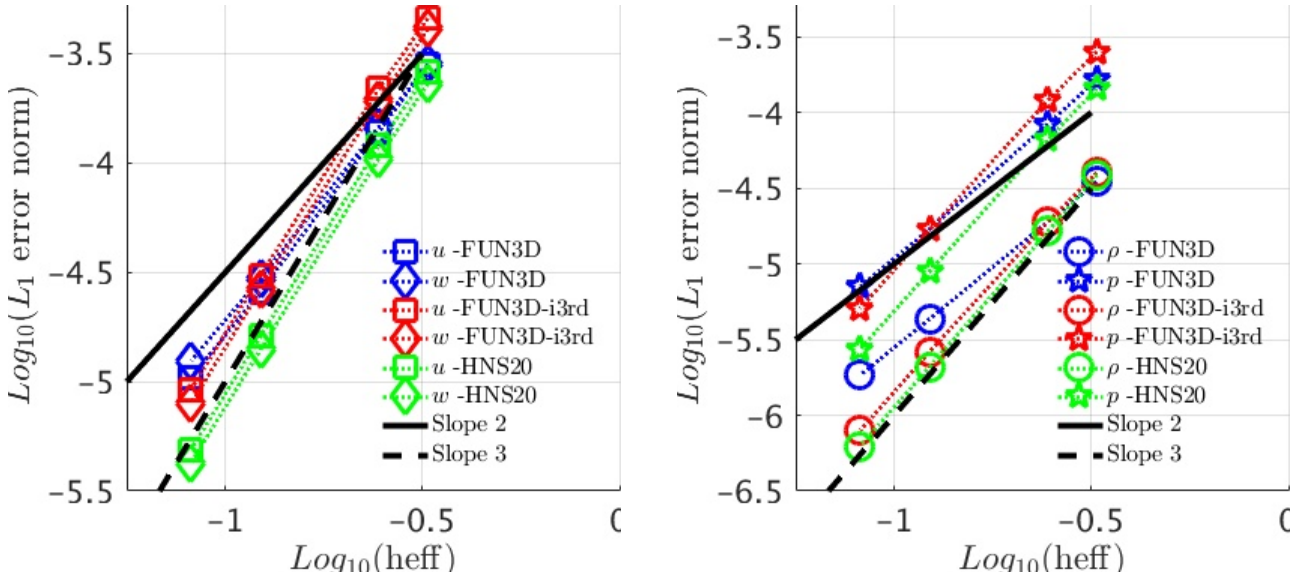
(a) Pressure. (b) W-velocity.

Figure 11: Slice data along z=0 line of the moving vortex at t=800.



(a) Tetrahedral grid. (b) Irregular local tetrahedral mesh.

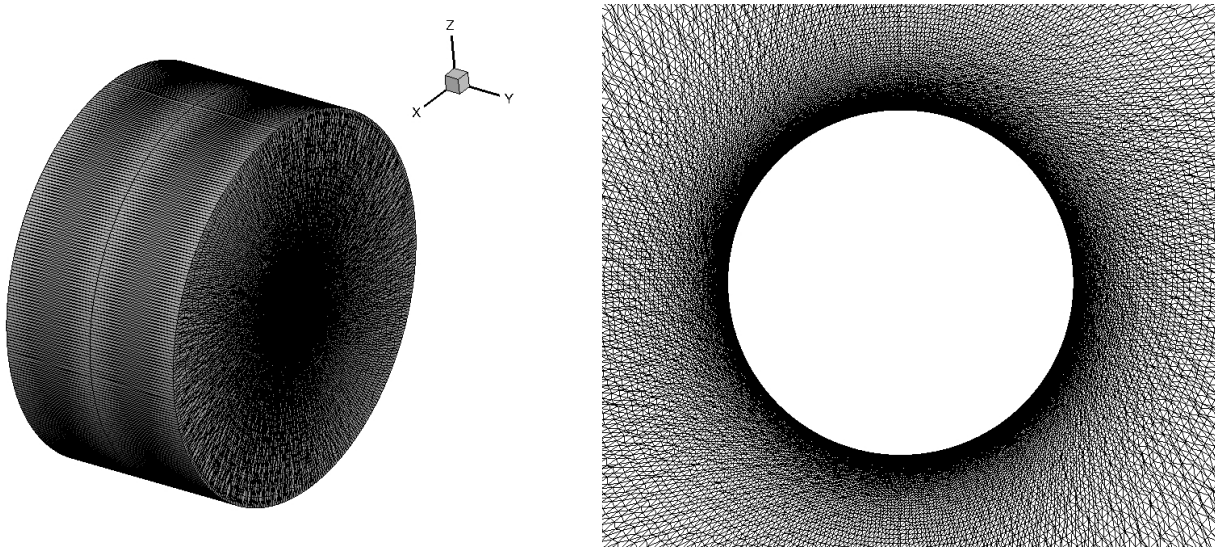
Figure 12: Irregular tetrahedra grid for the inviscid moving vortex calculated by Navier-Stokes solvers with $Re_\infty = 10^6$.



(a) U and W Velocities.

(b) Density and Pressure.

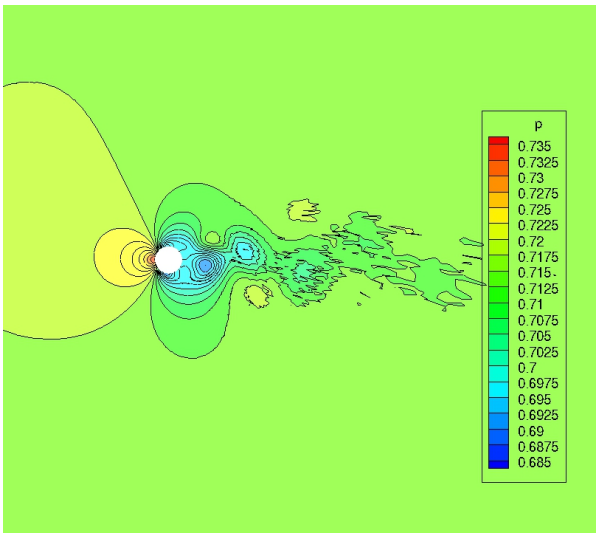
Figure 13: Accuracy verification results for the inviscid moving vortex problem for irregular grid.



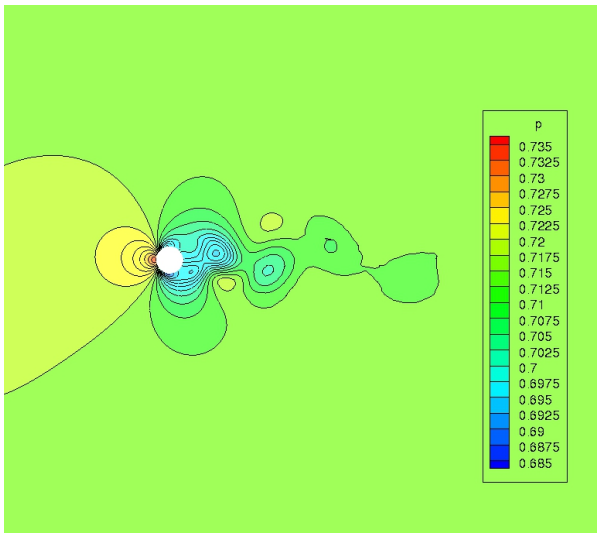
(a) 3D grid.

(b) Near-field view.

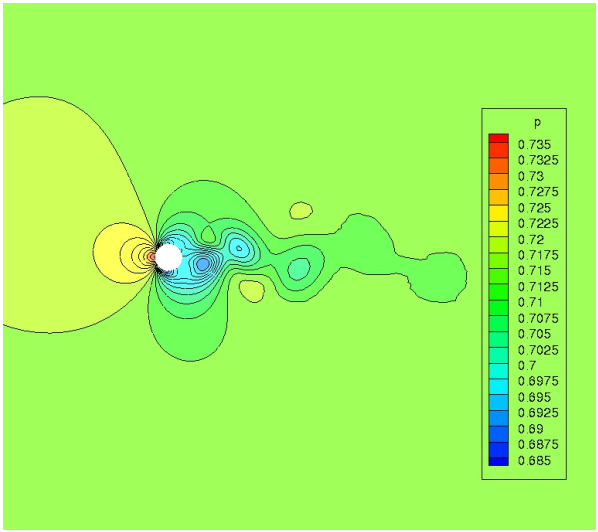
Figure 14: Irregular tetrahedral grid for laminar cylinder.



(a) FUN3D.

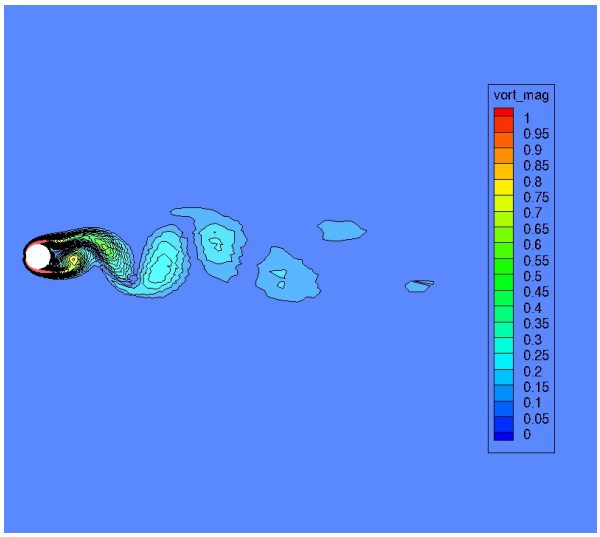


(b) FUN3D-i3rd.

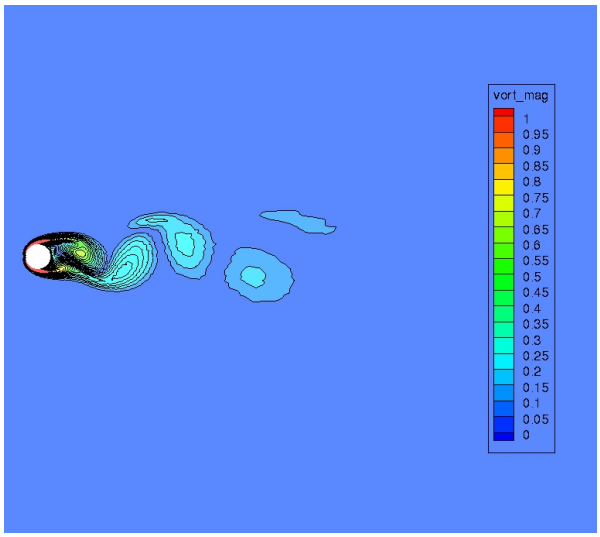


(c) HNS20.

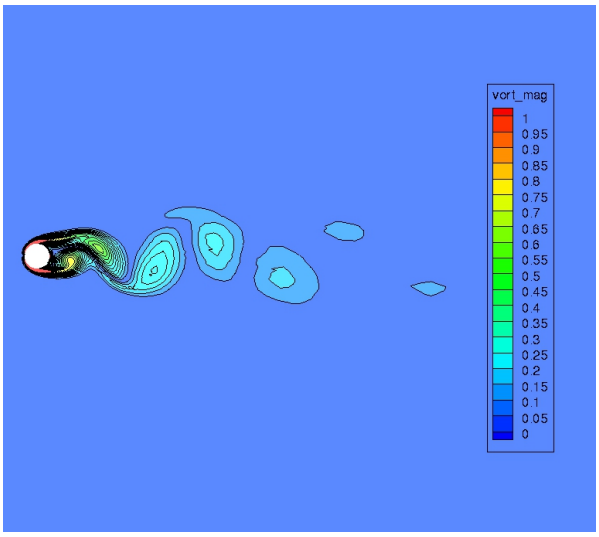
Figure 15: Pressure contours over the cylinder.



(a) FUN3D.

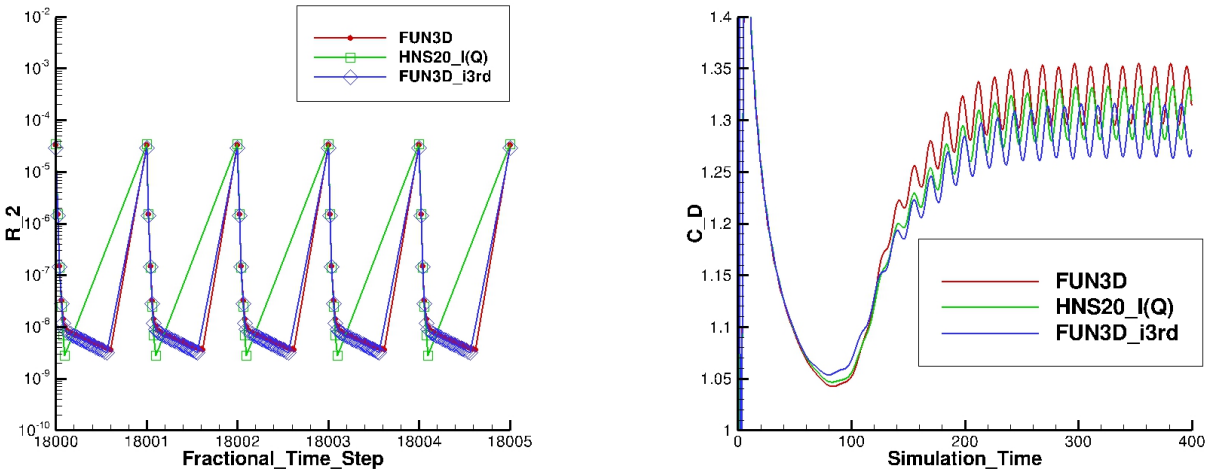


(b) FUN3D-i3rd.



(c) HNS20.

Figure 16: Vorticity magnitude contours over the cylinder.



(a) Sub-iteration histories.

(b) Drag coefficient histories.

Figure 17: Time histories of sub-iteration and drag coefficient.