Oral presentation | Higher order methods

Higher order methods-II

Tue. Jul 16, 2024 10:45 AM - 12:45 PM Room C

[4-C-03] A Third-Order Hyperbolic Navier-Stokes Solver for Unsteady Simulations on Adaptive Space-Time Tetrahedral Grids

*Hiroaki Nishikawa[1] （1. National Institute of Aerospace）

Keywords: Third-order, Hyperbolic Navier-Stokes, Adaptive grids

Twelfth International Conference on Computational Fluid Dynamics (ICCFD12), Kobe, Japan, July 14-19, 2024

A Third-Order Hyperbolic Navier-Stokes Solver for Unsteady Simulations on Adaptive Space-Time Tetrahedral Grids

Hiroaki Nishikawa
Corresponding author: hiroaki.nishikawa@nianet.org
National Institute of Aerospace, Hampton, VA, USA.

Abstract: In this paper, we present a third-order hyperbolic Navier-Stokes solver for two-dimensional unsteady simulations on adaptive space-time tetrahedral grids. The developed third-order hyperbolic Navier-Stokes solver is highly efficient: (1) no second derivatives are required, (2) it only requires a single numerical flux per edge, (3) third-order accuracy is achieved in both solution variables and their gradients on irregular grids, (4) third-order accuracy is achieved on linear tetrahedral grids (high-order curved elements are not required), (5) convergence acceleration is achieved by eliminating second-order operators despite the increase in the number of variables and equations. These advantages are demonstrated for space-time computations with tetrahedral grids generated by anisotropic grid adaptation.

Keywords: High-Order Methods, Edge-Based Solver, Space Time, Hyperbolic Navier-Stokes.

1 Introduction

Towards efficient, accurate, and automated computational fluid dynamics (CFD) simulations, we have recently been focusing on the development of space-time solvers [1, 2, 3], where unsteady governing equations are solved in a larger space with the physical time treated as an additional coordinate [4, 5, 6, 7]. A larger discrete problem arising from the space-time formulation allows us to exploit an ever-increasing parallelism, breaking the latency barrier, which limits the degree of parallelism for conventional solvers wth an explicit/implicit time-stepping scheme. It can be especially efficient when combined with anisotropic grid adaptation with tetrahedral grids as demonstrated in Ref. [6], for example, and in our previous papers [1, 2, 3]. However, it is well known that accuracy of gradients is severely degraded on such highly-skewed irregular tetrahedral grids. To address the issue of inaccurate gradients on adaptive tetrahedral grids, and to further improve efficiency, we developed a second-order accurate space-time hyperbolic Navier-Stokes (HNS) solver [3] and demonstrated superior gradient accuracy and faster convergence for two-dimensional unsteady viscous flows. In this paper, we extend the space-time HNS solver to third-order accuracy.

To achieve third-order accuracy, we employ the third-order edge-based method [8], which is a highly efficient discretization method in that no second derivatives are required and only a single numerical flux is required per edge. Moreover, it can achieve third-order accuracy without a high-order curved grid even for problems involving curved boundaries [9, 10]. By applying the third-order edge-based method to the space-time HNS formulation [3], we achieve third-order accuracy not only in the solution variables but also in their gradients on arbitrary tetrahedral grids. Moreover, iterative convergence acceleration can be expected due to the elimination of numerical stiffness associated with the discretization of second-derivative operators [11]. These features are highly desirable especially in computations on adaptive tetrahedral grids, where many existing CFD methods fail to produce accurate gradient quantities such as viscous stresses and heat fluxes [12].

2 Unsteady Compressible Navier-Stokes Equations

Consider the unsteady compressible Navier-Stokes (NS) equations:

$$\begin{aligned}
\partial_t \rho + \mathrm{div}(\rho \mathbf{v}) &= 0, &(1)\\
\partial_t (\rho \mathbf{v}) + \mathrm{div}(\rho \mathbf{v} \otimes \mathbf{v}) &= -\mathrm{grad}\, p + \mathrm{div}\, \boldsymbol{\tau}, &(2)\\
\partial_t (\rho E) + \mathrm{div}(\rho \mathbf{v} H) &= \mathrm{div}\,(\boldsymbol{\tau} \mathbf{v}) - \mathrm{div}\, \mathbf{q}, &(3)
\end{aligned}$$

where t is the physical time, $\otimes$ denotes the dyadic product, ρ is the density, $\mathbf{v}$ is the velocity vector, p is the pressure, E is the specific total energy, and $H = E + p/\rho$ is the specific total enthalpy. The viscous stress tensor $\boldsymbol{\tau}$ and the heat flux $\mathbf{q}$ are given, under Stokes' hypothesis, by

$$\boldsymbol{\tau} = -\frac{2}{3}\mu(\mathrm{div}\, \mathbf{v})\mathbf{I} + \mu \left(\mathrm{grad}\, \mathbf{v} + (\mathrm{grad}\, \mathbf{v})^t\right), \quad \mathbf{q} = -\frac{\mu}{Pr(\gamma - 1)} \mathrm{grad}\, T, \tag{4}$$

where $\mathbf{I}$ is the identity matrix, T is the temperature, γ is the ratio of specific heats ($\gamma = 1.4$ for air), Pr is the Prandtl number, μ is the viscosity defined by Sutherland's law, and the superscript t denotes the transpose. All the quantities are assumed to have been nondimensionalized by their free-stream values except that the velocity and the pressure are scaled by the free-stream speed of sound and the free-stream dynamic pressure, respectively (see Ref.[13]). Thus, the viscosity is given by the following form of Sutherland's law:

$$\mu = \frac{M_\infty}{Re_\infty} \frac{1 + C/\tilde{T}_\infty}{T + C/\tilde{T}_\infty} T^{\frac{3}{2}}, \tag{5}$$

where $\tilde{T}_\infty$ is the dimensional free stream temperature, and $C = 110.5$ [K] is the Sutherland constant. The ratio of the free stream Mach number, M_∞, to the free stream Reynolds number per grid unit, Re_∞, arises from the nondimensionalization. The system is closed by the nondimensionalized equation of state for ideal gases:

$$\gamma p = \rho T. \tag{6}$$

As in the previous work [3], we will first reformulate the viscous terms as a hyperbolic system and then discretize it by an upwind discretization method. In the next section, we briefly describe the hyperbolic formulation.

3 A Hyperbolic Navier-Stokes System

There exist five possible hyperbolic formulations for the viscous terms: HNS8 [12], HNS14 [14], HNS17 [11], HNS20 [11], and HNS20G [15]. HNS8 involves only the temperature derivatives as extra unknowns and is an economical choice if one is interested only in getting accurate heat fluxes [12]. In this work, as in the previous paper, we employ HNS20G, which was first introduced in Ref.[15] to simplify the development of high-order reconstructed discontinuous Galerkin methods. In HNS20G, we reformulate an HNS system by adding extra variables $\mathbf{r}$, $\mathbf{g}$, $\mathbf{h}$ corresponding to the gradients of the primitive variables, which are called the gradient variables:

$$\begin{aligned}
\partial_\tau \rho + \partial_t \rho + \mathrm{div}(\rho \mathbf{v}) &= \mathrm{div}\ (\nu_r \mathbf{r})\,, &(7)\\
\partial_\tau (\rho \mathbf{v}) + \partial_t (\rho \mathbf{v}) + \mathrm{div}(\rho \mathbf{v} \otimes \mathbf{v}) &= -\mathrm{grad}\, p + \mathrm{div}\ (\mu_v \widetilde{\boldsymbol{\tau}})\,, &(8)\\
\partial_\tau (\rho E) + \partial_t (\rho E) + \mathrm{div}(\rho \mathbf{v} H) &= \mathrm{div}\,(\mu_v \widetilde{\boldsymbol{\tau}} \mathbf{v}) \ + \ \mathrm{div}\left(\frac{\mu_h}{\gamma(\gamma - 1)} \mathbf{h}\right), &(9)\\
T_v\, \partial_\tau \mathbf{g} &= \mathrm{grad}\, \mathbf{v} - \mathbf{g}, &(10)\\
T_h\, \partial_\tau \mathbf{h} &= \mathrm{grad}\, T - \mathbf{h}, , &(11)\\
T_r\, \partial_\tau \mathbf{r} &= \mathrm{grad}\, \rho - \mathbf{r}, &(12)
\end{aligned}$$

where τ is a pseudotime variable, and $\widetilde{\boldsymbol{\tau}} = -\frac{1}{2} tr(\mathbf{g})\mathbf{I} + \frac{3}{4}\left(\mathbf{g} + \mathbf{g}^t\right)$,

$$T_r = \frac{L_r^2}{\nu_r}, \quad T_v = \frac{\rho L_v^2}{\mu_v}, \quad T_h = \frac{\rho L_h^2}{\mu_h}, \quad \mu_v = \frac{4\mu}{3}, \quad \mu_h = \frac{\gamma \mu}{Pr}, \quad \nu_r = 10^{-12}, \tag{13}$$

and the length scales may be defined as $L_r = L_v = L_h = 1/(2\pi)$ or, as in Refs.[15, 16],

$$\begin{aligned} L_r &= L_d = \frac{1}{2\pi}, \\ L_v &= \frac{L_d}{\sqrt{Re_{L_d}^{v\infty}}}, \quad Re_{L_d}^{v\infty} = \frac{U_\infty L_d}{\nu_{v\infty}}, \\ L_h &= \frac{L_d}{\sqrt{Re_{L_d}^{h\infty}}}, \quad Re_{L_d}^{h\infty} = \frac{U_\infty L_d}{\nu_{h\infty}}, \end{aligned} \tag{14}$$

where U_∞ is the free-stream flow speed and the subscript ∞ refers to the free-stream state. The latter is more suitable for high-Reynolds-number flows.

The HNS20G system is equivalent to the original NS system as soon as the pseudotime derivatives are dropped, and we will do so after the discretization. Then, the gradient variables $\mathbf{r}$, $\mathbf{g}$ and $\mathbf{h}$ correspond to the density, temperature, and velocity gradients:

$$\mathbf{r} = \text{grad}\, \rho, \quad \mathbf{g} = \text{grad}\, \mathbf{v}, \quad \mathbf{h} = \text{grad}\, T. \tag{15}$$

The viscous part of HNS20G is hyperbolic in the pseudotime, and therefore can be discretized by an upwind method similarly to the inviscid part. Once the viscous part is discretized, we will drop the pseudotime terms and arrive at a consistent discretization of the original NS system in a first-order form. The HNS method is a way of deriving a superior discretization for the original NS system via a hyperbolic formulation.

4 Hyperbolicity and Upwind Fluxes

Consider the vector form of the HNS20G system in two dimensions:

$$\mathbf{P}^{-1}\frac{\partial \mathbf{u}}{\partial \tau} + \frac{\partial \mathbf{f}_t^{time}}{\partial t} + \frac{\partial \mathbf{f}_x^{inv}}{\partial x} + \frac{\partial \mathbf{f}_y^{inv}}{\partial y} + \frac{\partial \mathbf{f}_x^{vis}}{\partial x} + \frac{\partial \mathbf{f}_y^{vis}}{\partial y} + \frac{\partial \mathbf{f}_x^{ahd}}{\partial x} + \frac{\partial \mathbf{f}_y^{ahd}}{\partial y} = \mathbf{s}, \tag{16}$$

where τ is a pseudotime variable, $\mathbf{P}^{-1} = \text{diag}(1, 1, 1, 1, T_r, T_r, T_v, T_v, T_v, T_v, T_h, T_h)$,

$$\mathbf{u} = \begin{bmatrix} \rho \\ \rho u \\ \rho v \\ \rho E \\ g_{ux} \\ g_{uy} \\ g_{vx} \\ g_{vy} \\ h_x \\ h_y \\ r_x \\ r_y \end{bmatrix}, \quad \mathbf{f}_t^{time} = \begin{bmatrix} \rho \\ \rho u \\ \rho v \\ \rho E \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \quad \mathbf{f}_x^{inv} = \begin{bmatrix} \rho u \\ \rho u^2 + p \\ \rho u v \\ \rho u H \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \quad \mathbf{f}_y^{inv} = \begin{bmatrix} \rho v \\ \rho u v \\ \rho v^2 + p \\ \rho v H \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \tag{17}$$

$$\mathbf{f}_x^{vis} = \begin{bmatrix} 0 \\ -\mu_v \widetilde{\tau}_{xx} \\ -\mu_v \widetilde{\tau}_{xy} \\ -\mu_v \widetilde{\tau}_{xx} u - \mu_v \widetilde{\tau}_{xy} v - \frac{\mu_h}{\gamma(\gamma-1)} h_x \\ -u \\ 0 \\ -v \\ 0 \\ -T \\ 0 \\ -\rho \\ 0 \end{bmatrix}, \quad \mathbf{f}_y^{vis} = \begin{bmatrix} 0 \\ -\mu_v \widetilde{\tau}_{yx} \\ -\mu_v \widetilde{\tau}_{yy} \\ -\mu_v \widetilde{\tau}_{yx} u - \mu_v \widetilde{\tau}_{yy} v - \frac{\mu_h}{\gamma(\gamma-1)} h_y \\ 0 \\ -u \\ 0 \\ -v \\ 0 \\ -T \\ 0 \\ -\rho \end{bmatrix}, \tag{18}$$

$$\mathbf{f}_x^{ahd} = \begin{bmatrix} -\nu_r r_x \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ -\rho \\ 0 \end{bmatrix}, \quad \mathbf{f}_y^{ahd} = \begin{bmatrix} -\nu_r r_y \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ -\rho \end{bmatrix}, \quad \mathbf{s} = \begin{bmatrix} s_1(x,y,t) \\ s_2(x,y,t) \\ s_3(x,y,t) \\ s_4(x,y,t) \\ -g_{ux} \\ -g_{uy} \\ -g_{vx} \\ -g_{vy} \\ -h_x \\ -h_y \\ -r_x \\ -r_y \end{bmatrix}, \tag{19}$$

where s_1, s_2, s_3, s_4 are forcing terms used here only for verification studies, and $\mathbf{f}_x^{ahd}$ and $\mathbf{f}_y^{ahd}$ are the fluxes for the artificial hyperbolic diffusion system introduced to obtain accurate density gradients [11].

If we focus on the inviscid terms,

$$\mathbf{P}^{-1} \frac{\partial \mathbf{u}}{\partial \tau} + \frac{\partial \mathbf{f}_x^{inv}}{\partial x} + \frac{\partial \mathbf{f}_y^{inv}}{\partial y} = \mathbf{0}, \tag{20}$$

this system is known to be hyperbolic in τ, and an upwind flux (or a Riemann solver) in an arbitrary direction $\hat{\hat{\mathbf{n}}}_{xy}$ in the (x,y)-space can be constructed as

$$\mathbf{\Phi}^{inv}(\hat{\hat{\mathbf{n}}}_{xy}) = \frac{1}{2}\left[\mathbf{f}^{inv}(\mathbf{w}_L) + \mathbf{f}^{inv}(\mathbf{w}_R)\right] - \frac{1}{2}\mathbf{P}^{-1}\left|\mathbf{P}\frac{\partial \mathbf{f}^{inv}}{\partial \mathbf{u}}\right|(\mathbf{u}_R - \mathbf{u}_L), \tag{21}$$

where $\mathbf{f}^{inv} = (\mathbf{f}_x^{inv}, \mathbf{f}_y^{inv}) \cdot \hat{\hat{\mathbf{n}}}_{xy}$, $\hat{\hat{\mathbf{n}}}_{xy}$ is a unit vector, $\mathbf{u}_L$ and $\mathbf{u}_R$ are left and right states in the conservative variables, $\mathbf{w}_L$ and $\mathbf{w}_R$ are left right states in the primitive variables,

$$\mathbf{w} = (\rho, u, v, p, g_{ux}, g_{uy}, g_{vx}, g_{vy}, h_x, h_y, r_x, r_y)^t, \tag{22}$$

and the dissipation matrix is evaluated at the Roe averages [17]. Note that the local preconditioning matrix $\mathbf{P}$ has no impact on the inviscid terms, and therefore, we actually have

$$\mathbf{\Phi}^{inv} = \frac{1}{2}\left[\mathbf{f}^{inv}(\mathbf{w}_L) + \mathbf{f}^{inv}(\mathbf{w}_R)\right] - \frac{1}{2}\left|\frac{\partial \mathbf{f}^{inv}}{\partial \mathbf{u}}\right|(\mathbf{u}_R - \mathbf{u}_L). \tag{23}$$

The Roe flux is just one example; any other numerical flux can be employed.

Similarly, if we focus on the physical time derivative term,

$$\mathbf{P}^{-1}\frac{\partial \mathbf{u}}{\partial \tau} + \frac{\partial \mathbf{f}_t^{time}}{\partial t} = \mathbf{0}, \tag{24}$$

we find that this system is also hyperbolic in τ and $\mathbf{P}$ has no impact on this system. Therefore, we can construct an upwind flux in an arbitrary direction $\hat{\hat{n}}_t$ in time, simply as

$$\boldsymbol{\Phi}^{time}(\hat{\hat{n}}_t) = \frac{1}{2}\left[\tilde{\mathbf{u}}_L + \tilde{\mathbf{u}}_R\right]\hat{\hat{n}}_t - \frac{1}{2}|\hat{\hat{n}}_t|\left(\tilde{\mathbf{u}}_R - \tilde{\mathbf{u}}_L\right), \tag{25}$$

where $\hat{\hat{n}}_t$ is -1 or 1, and $\tilde{\mathbf{u}} = (\rho, \rho u, \rho v, \rho E, 0, 0, 0, 0, 0, 0, 0, 0)$.

Next, we focus on the viscous terms,

$$\mathbf{P}^{-1}\frac{\partial \mathbf{u}}{\partial \tau} + \frac{\partial \mathbf{f}_x^{vis}}{\partial x} + \frac{\partial \mathbf{f}_y^{vis}}{\partial y} = \mathbf{0}, \tag{26}$$

which is also hyperbolic in τ, and a upwind flux in an arbitrary direction $\hat{\hat{\mathbf{n}}}_{xy}$ in the (x, y)-space can be constructed as

$$\boldsymbol{\Phi}_{jk}^{vis}(\hat{\hat{\mathbf{n}}}_{xy}) = \frac{1}{2}\left[\mathbf{f}^{vis}(\mathbf{w}_L) + \mathbf{f}^{vis}(\mathbf{w}_R)\right] - \frac{1}{2}\mathbf{P}^{-1}\left\{\left|\mathbf{P}\frac{\partial \mathbf{f}^{vis}}{\partial \mathbf{u}}\right|(\mathbf{u}_R - \mathbf{u}_L) + \Delta\boldsymbol{\Phi}^{AHD}\right\}, \tag{27}$$

where $\mathbf{f}^{vis} = (\mathbf{f}_x^{vis}, \mathbf{f}_y^{vis}) \cdot \hat{\hat{\mathbf{n}}}_{xy}$, and $\Delta\boldsymbol{\Phi}^{AHD}$ is the artificial hyperbolic dissipation to introduce strong coupling among velocities and their gradients [3, 11]. The analytical expression for the dissipation term can be found in Ref. [3].

Finally, we focus on the artificial hyperbolic diffusion part [11]:

$$\mathbf{P}^{-1}\frac{\partial \mathbf{u}}{\partial \tau} + \frac{\partial \mathbf{f}_x^{ahd}}{\partial x} + \frac{\partial \mathbf{f}_y^{ahd}}{\partial y} = \mathbf{0}, \tag{28}$$

which is also hyperbolic in τ, and an upwind flux can be constructed as

$$\boldsymbol{\Phi}^{ahd}(\hat{\hat{\mathbf{n}}}_{xy}) = \frac{1}{2}\left[\mathbf{f}^{ahd}(\mathbf{w}_L) + \mathbf{f}^{ahd}(\mathbf{w}_R)\right] - \frac{1}{2}\mathbf{P}^{-1}\left|\mathbf{P}\frac{\partial \mathbf{f}^{ahd}}{\partial \mathbf{u}}\right|(\mathbf{u}_R - \mathbf{u}_L), \tag{29}$$

where $\mathbf{f}^{ahd} = (\mathbf{f}_x^{ahd}, \mathbf{f}_y^{ahd}) \cdot \hat{\hat{\mathbf{n}}}_{xy}$, and

$$\left|\mathbf{P}\frac{\partial \mathbf{f}^{ahd}}{\partial \mathbf{u}}\right|(\mathbf{u}_R - \mathbf{u}_L) = \left(\frac{\nu_r}{L_r}\Delta\rho, 0, 0, 0, 0, 0, 0, 0, 0, 0, \frac{\nu_r}{L_r}\Delta r_n \hat{n}_x, \frac{\nu_r}{L_r}\Delta r_n \hat{n}_y\right)^t, \tag{30}$$

$\Delta\rho = \rho_R - \rho_L$, and $\Delta r_n = (r_x, r_y) \cdot \hat{\hat{\mathbf{n}}}_{xy}$.

Therefore, all the terms in the HNS20G system are hyperbolic in τ and upwind fluxes can be constructed for all of them. We are now ready to discuss the third-order discretization.

5 A Third-Order Space-Time Hyperbolic Navier-Stokes Solver

5.1 Third-Order Accurate Discretization

We consider a tetrahedral grid in the (x, y, t)-space, and discretize the HNS20G system by the third-order edge-based discretization method [8]:

$$\mathbf{P}_j^{-1}\frac{d\mathbf{u}_j}{d\tau} + \mathbf{Res}_j = \mathbf{0}, \tag{31}$$

$$\mathbf{Res}_j = \sum_{k \in \{k_j\}}\left[\boldsymbol{\Phi}_{jk}|\mathbf{n}_{jk}| - \frac{1}{2}\left(a_L \mathbf{s}_j + b_L \nabla \mathbf{s}_j \cdot (\mathbf{x}_k - \mathbf{x}_j) + a_R \mathbf{s}_k\right) V_{jk}\right], \tag{32}$$

where $\{k_j\}$ is a set of neighbor nodes of j, $\mathbf{x}_k - \mathbf{x}_j$ is the edge vector, $|\mathbf{n}_{jk}|$ is the magnitude of the directed area vector $\mathbf{n}_{jk} = (n_x, n_y, n_t)$, which is the sum of the directed-areas corresponding to the dual-triangular faces associated with all tetrahedral elements sharing the edge $[j, k]$ (see Figure 1), $V_{jk} = (\mathbf{x}_k - \mathbf{x}_j) \cdot \mathbf{n}_{jk}/6$, $a_L = 13/5$, $b_L = 3/5$, and $a_R = -3/5$, which correspond to an economical formula derived for third-order accuracy in Ref. [8], and $\mathbf{\Phi}_{jk}$ is the sum of the upwind fluxes constructed for the inviscid, viscous, and time fluxes:

$$\mathbf{\Phi}_{jk} \quad = \quad \left[\mathbf{\Phi}_{jk}^{inv}(\hat{\hat{\mathbf{n}}}_{xy}) + \mathbf{\Phi}_{jk}^{vis}(\hat{\hat{\mathbf{n}}}_{xy}) \right] \sqrt{\hat{n}_x^2 + \hat{n}_y^2} + \mathbf{\Phi}_{jk}^{time}(\hat{\hat{n}}_t)|\hat{n}_t|, \tag{33}$$

where $\hat{\mathbf{n}}_{jk} = \mathbf{n}_{jk}/|\mathbf{n}_{jk}| = (\hat{n}_x, \hat{n}_y, \hat{n}_t)^t$, and

$$\hat{\hat{\mathbf{n}}}_{xy} = \frac{(\hat{n}_x, \hat{n}_y)^t}{\sqrt{\hat{n}_x^2 + \hat{n}_y^2}}, \quad \hat{\hat{n}}_t = \frac{\hat{n}_t}{|\hat{n}_t|}. \tag{34}$$

For third-order accuracy, we must perform the solution reconstruction,

$$\mathbf{w}_L \quad = \quad \kappa \frac{\mathbf{w}_j + \mathbf{w}_k}{2} + (1 - \kappa) \left[\mathbf{w}_j + \frac{1}{2} \nabla \mathbf{w}_j^{LSQ} \cdot (\mathbf{x}_k - \mathbf{x}_j) \right], \tag{35}$$

$$\mathbf{w}_R \quad = \quad \kappa \frac{\mathbf{w}_j + \mathbf{w}_k}{2} + (1 - \kappa) \left[\mathbf{w}_k - \frac{1}{2} \nabla \mathbf{w}_k^{LSQ} \cdot (\mathbf{x}_k - \mathbf{x}_j) \right], \tag{36}$$

with $\kappa = 0$ (third-order accuracy is lost with $\kappa \neq 0$ as demonstrated in Ref. [11]), and we must compute the solution gradients $\nabla \mathbf{w}_j^{LSQ}$ and $\nabla \mathbf{w}_k^{LSQ}$ by a quadratic least-squares method. In this study, we employed the two-step method proposed in Ref. [18], which involves neighbors of the edge-connected neighbors but can be implemented without accessing them directly. Once the solution reconstruction is done, we compute the conservative variables $\mathbf{u}_L$ and $\mathbf{u}_R$ algebraically from the primitive variables: $\mathbf{u}_L(\mathbf{w}_L)$ and $\mathbf{u}_R(\mathbf{w}_R)$. For third-order accuracy, we must also perform the flux extrapolation:

$$\mathbf{\Phi}_{jk}^{inv}(\hat{\hat{\mathbf{n}}}_{xy}) = \frac{1}{2} \left[\mathbf{f}_L^{inv} + \mathbf{f}_R^{inv} \right] - \frac{1}{2} \left| \frac{\partial \mathbf{f}_n^{inv}}{\partial \mathbf{u}} \right| (\mathbf{u}_R - \mathbf{u}_L), \tag{37}$$

$$\mathbf{\Phi}_{jk}^{vis}(\hat{\hat{\mathbf{n}}}_{xy}) = \frac{1}{2} \left[\mathbf{f}_L^{vis} + \mathbf{f}_R^{vis} \right] - \frac{1}{2} \mathbf{P}^{-1} \left\{ \left| \mathbf{P} \frac{\partial \mathbf{f}_n^{vis}}{\partial \mathbf{u}} \right| (\mathbf{u}_R - \mathbf{u}_L) + \Delta \Phi^{AHD} \right\}, \tag{38}$$

where

$$\mathbf{f}_L^{inv} \quad = \quad \mathbf{f}_j^{inv} + \frac{1}{2} \left(\frac{\partial \mathbf{f}^{inv}}{\partial \mathbf{w}} \right)_j \nabla \mathbf{w}_j^{LSQ} \cdot (\mathbf{x}_k - \mathbf{x}_j), \tag{39}$$

$$\mathbf{f}_R^{inv} \quad = \quad \mathbf{f}_k^{inv} - \frac{1}{2} \left(\frac{\partial \mathbf{f}^{inv}}{\partial \mathbf{w}} \right)_k \nabla \mathbf{w}_k^{LSQ} \cdot (\mathbf{x}_k - \mathbf{x}_j), \tag{40}$$

$$\mathbf{f}_L^{vis} \quad = \quad \mathbf{f}_j^{vis} + \frac{1}{2} \left(\frac{\partial \mathbf{f}^{vis}}{\partial \mathbf{w}} \right)_j \nabla \mathbf{w}_j^{LSQ} \cdot (\mathbf{x}_k - \mathbf{x}_j), \tag{41}$$

$$\mathbf{f}_R^{vis} \quad = \quad \mathbf{f}_k^{vis} - \frac{1}{2} \left(\frac{\partial \mathbf{f}^{vis}}{\partial \mathbf{w}} \right)_k \nabla \mathbf{w}_k^{LSQ} \cdot (\mathbf{x}_k - \mathbf{x}_j), \tag{42}$$

where the chain rule has been used to avoid computing the flux gradients. Note that the time flux $\mathbf{\Phi}_{jk}^{time}$ and the artificial hyperbolic diffusion flux are linear in the solution variables, and therefore, we employ the fluxes given in Equations (25) and (29), which are equivalent to the flux extrapolation.

The residual is closed at boundaries by using an accuracy-preserving boundary flux quadrature formula derived in Ref. [9] for third-order accuracy. It is emphasized that third-order accuracy is achieved over a curved boundary without using a high-order curved grid and the grid must be purely tetrahedral with triangulated boundaries in order to achieve third-order accuracy. See Refs. [9, 10]. Boundary conditions are implemented in a weak manner through the right state in a numerical flux at boundary nodes.

5.2 Nonlinear Solver

To obtain a space-time solution, we drop the peudotime terms and solve the global system of the residual equations:

$$\mathbf{Res}(\mathbf{U}) = \mathbf{0}, \tag{43}$$

where $\mathbf{U}$ denotes a global vector of unknowns and $\mathbf{Res}$ is a global vector of residuals, whose j-th system is given by Equation (32), by a Jacobian-Free Newton-Krylov (JFNK) solver:

$$\mathbf{U}^{m+1} = \mathbf{U}^{m} + \Delta\mathbf{U}, \tag{44}$$

where m is the iteration counter and $\Delta\mathbf{U}$ is a correction computed by the generalized conjugate residual (GCR) method (a Krylov solver). Following the algorithm presented in Ref.[11, 19], we set $\mathbf{x}_0 = \mathbf{0}$, $\mathbf{r}_0 = -\mathbf{Res}(\mathbf{U}^m)$, $\mathbf{p}_0 = \tilde{\mathbf{A}}^{-1}\mathbf{r}_0$, and perform for $i = 0, 1, 2, \cdots, i_{max}^{GCR}$

$$\alpha_i = \frac{(\mathbf{A}\mathbf{p}_i)^T \mathbf{r}_i}{(\mathbf{A}\mathbf{p}_i)^T (\mathbf{A}\mathbf{p}_i)}, \quad \mathbf{x}_{i+1} = \mathbf{x}_i + \alpha_i \mathbf{p}_i, \quad \mathbf{r}_{i+1} = \mathbf{r}_i - \alpha_i \mathbf{A}\mathbf{p}_i, , \tag{45}$$

where i_{max}^{GCR} is a user-defined integer, and $\mathbf{A} = \partial\mathbf{Res}/\partial\mathbf{U}$ is the true Jacobian, but the computation and storage are avoided by computing the matrix-vector product, e.g., $\mathbf{A}\mathbf{p}_i$, by the Fréchet derivative approximation:

$$\mathbf{A}\mathbf{p}_i = \frac{\mathbf{Res}(\mathbf{U}^m + \epsilon \mathbf{p}_i) - \mathbf{Res}(\mathbf{U}^m)}{\epsilon}, \tag{46}$$

where

$$\epsilon = \max(1, |\mathbf{U}^m|)\sqrt{10^{-16}}. \tag{47}$$

If the residual $\mathbf{r}_{i+1}$ is sufficiently small, i.e., $\sqrt{\mathbf{r}_{i+1}^T \mathbf{r}_{i+1}}/\sqrt{\mathbf{r}_0^T \mathbf{r}_0} < \mu^{GCR}$, where μ^{GCR} is a user-defined tolerance (e.g., $\mu^{GCR} = 0.1$), then stop and obtain the correction as $\Delta\mathbf{U} = \mathbf{x}_{i+1}$. If the residual is not sufficiently small, compute $\mathbf{p}_{i+1} = \tilde{\mathbf{A}}^{-1}\mathbf{r}_{i+1}$, perform the orthogonalization for $k = 0, 1, 2, \cdots, i$:

$$\beta_i = \frac{(\mathbf{A}\mathbf{p}_i)^T (\mathbf{A}\mathbf{p}_k)}{(\mathbf{A}\mathbf{p}_k)^T (\mathbf{A}\mathbf{p}_k)}, \quad \mathbf{p}_{i+1} = \mathbf{p}_{i+1} + \beta_k \mathbf{p}_k, \quad \mathbf{A}\mathbf{p}_{i+1} = \mathbf{A}\mathbf{p}_{i+1} - \beta_k \mathbf{A}\mathbf{p}_k, \tag{48}$$

and go back to the step (45).

The symbol $\tilde{\mathbf{A}}^{-1}$ denotes a variable-preconditioner based on a multi-color Gauss-Seidel relaxation, which approximately inverts an approximate Jacobian $\tilde{\mathbf{A}} = \frac{\partial \overline{\mathbf{Res}}}{\partial \mathbf{U}}$, where $\overline{\mathbf{Res}}$ is a lower-order residual with zero LSQ gradients. Note that it corresponds to the exact derivative of a first-order scheme for the HNS schemes but of a zeroth-order scheme for conventional viscous schemes [20, 21, 22]. Therefore, superior convergence is expected especially for highly-skewed grids. The linear relaxation is performed with the tolerance μ^{GS} of 50% residual reduction (i.e., $\mu^{GS} = 0.5$), or for a specified maximum number i_{max}^{GS} of relaxations (e.g., $i_{max}^{GS} = 10$). In this study, we do not add any psuedotime term to any of the Jacobians; in a typical nonlinear solver, this corresponds to setting the CFL number to infinity.

6 Results

6.1 Accuracy Verification

To verify third-order accuracy, we solve the residual system for a series of four irregular tetrahedral grids (sections of the coarsest one is shown in Figure 2 in the space-time domain) by the second-order HNS scheme, HNS(2nd), developed in Ref. [3], the third-order HNS scheme, HNS(3rd), developed in this work, and a conventional NS scheme, Alpha4/3, directly discretizing the original NS system [2].

Figures 3a and 3b show error convergence results for the density and the x-derivative of the density. Results are similar for other variables, and therefore not shown. For the density, as can be seen, Alpha4/3 and HNS(2nd) achieve second-order accuracy while HNS(3rd) achieves third-order accuracy as expected. On the other hand, for the x-derivative of the density, HNS(2nd) and HNS(3rd) achieve second- and third-order accuracy, respectively, as expected, whereas Alpha4/3 gives only first-order accuracy in the gradients, which is one-order-lower than the solution accuracy as typical in conventional schemes. Improved gradient accuracy can be seen clearly in the contour plots shown for the coarsest grid in Figure 4. Clearly, the HNS schemes produce more accurate gradients and HNS(3rd) produces the most accurate gradients.

Figure 5 shows that the solver converged faster not only in the number of iterations but also in CPU time for HNS(2nd) and HNS(3rd) than for Alpha4/3. For this test case, we set the JFNK solver parameters as $i^{GCR}_{max} = 2$, $\mu^{GCR} = 0.5$, $i^{GS}_{max} = 10$, and $\mu^{GS} = 0.5$. The solver is taken to be converged when the nonlinear residuals are reduced by five orders of magnitude. Here, we observe that the improved condition number (by the elimination of second-derivative operators) leads to acceleration in terms of the number of iterations. In this case, $i^{GS}_{max} = 10$ is not sufficient to meet the tolerance $\mu^{GS} = 0.5$ for Alpha4/3 while it is sufficient for the HNS schemes; and the GCR linear residual is only reduced by 15% (i.e., $\mu^{GCR} = 0.5$ is not met with $i^{GCR}_{max} = 2$) whereas the tolerance $\mu^{GCR} = 0.5$ is always met for the HNS schemes. As a result of fewer numbers of iterations, the solver converged faster in CPU time for the HNS schemes despite the extra computational cost (due to 8 additional equations) required in each iteration. This is a known advantage of the HNS method as demonstrated in previsou studies [11, 14].

6.2 A Viscous Flow over Circular Cylinder at $Re_\infty = 150$

Next, we consider a viscous flow over a circular cylinder centered at the origin at $Re_\infty = 150$ and $M_\infty = 0.15$. The outer boundary is defined by the semi-circle of radius 10 centered at the origin, straight farfield boundaries located at the distance 10 from the x-axis, and a straight outflow boundary at $x = 15$; and the two-dimensional domain is extended in the z direction from $z = 0$ to $z = 150$ for computing an unsteady viscous flow from time 0 to time 150. The grid is an adaptive tetrahedral grid with 304,370 nodes and 1,703,679 tetrahedra generated in the previous study [3] by the *refine* package (https://github.com/nasa/refine) developed at NASA Langley Research Center. The grid and a section at $y = 0$ are shown in Figures 6a and 6b, respectively. As can be seen, the grid is anisotropically adapted to capture vorticies generated from the cylinder and convected towards the outflow boundary. This grid is far more efficient than a time-slab space-time grid generated by extruding a two-dimensional grid, such as the top boundary in Figure 6a, in the time direction. Observe, in particular, the coarse-grid region ahead of the cylinder, where the flow is nearly uniform and high resolution is not necessary, in contrast to a time-slab space-time grid, where the grid spacing in time will be uniformly fine including such a constant-flow region. For this test case, we set the JFNK solver parameters as $i^{GCR}_{max} = 25$, $\mu^{GCR} = 0.5$, $i^{GS}_{max} = 150$, and $\mu^{GS} = 0.1$. The solver is taken to be converged when the nonlinear residuals are reduced by eight orders of magnitude. As before, we compare three schemes: Alpha4/3 [2], HNS(2nd) [3], and HNS(3rd) developed in this work.

Figures 7, 8, 9, and 10 show grids and Mach contours for four sections taken from $z = 10$, 50, 130, and 149, respectively. As can be seen, a typical vortex shedding is observed towards the final time $z = 149$, and the grid is adapted for each solution. Note that these grids are sections of the three-dimensional grid adapted for solutions in the space-time domain. Overall, the numerical solutions look similar for the three schemes. The vortices look stronger in the third-order accurate solution; see Figure 10, for example.

Figures 11a, 11b, 11c, and 11d show a grid at $z = 149$ and vorticity contours computed by the LSQ method from the numerical solution for Alpha4/3, and those obtained by the gradient variables for the HNS schemes, respectively. Note that the vorticity in the HNS cases is obtained directly from the gradient variables: $g_{uy} - g_{vx}$. As can be seen, the HNS vorticity contours are smoother than the LSQ vorticity contours as expected, and HNS(3rd) produced smother contours than HNS(2nd). Improvements are more significant in regions of high anisotropy, for example, near the surface, as can be observed in the zoomed-in plots as shown in Figure 12. In Figure 13, the vorticity contours are compared over the circular cylinder surface. It can be clearly seen that the HNS vorticity contours are significantly smoother than the LSQ vorticity contours, and HNS(3rd) gives smother contours than HNS(2nd).

Figure 14 shows iterative convergence results. First, we see from Figure 14a that the solver converged for nearly the same number of iterations for Alpha4/3, HNS(2nd), and HNS(3rd). However, in terms of CPU time, the solver converged faster for HNS(2nd) and HNS(3rd) than for Alpha4/3. As well known, a linearized equation of the HNS residuals has a reduced condition number by a factor of $O(h)$, where h denotes a representative mesh spacing, compared with a conventional residual. This leads to a speed-up in the linear relaxation scheme in the preconditioner, and consequently to a reduction in the GCR projections (Krylov vectors). These are shown in Figures 14c and 14d. Especially, near the end of the iteration, Alpha4/3 required significantly more linear relaxations and GCR projections. As a result, the solver converged faster in CPU time for the HNS schemes despite the additional cost (associated with 8 extra variables and equations) per iteration. Therefore, the HNS solvers can produce significantly more accurate and smoother gradients on adaptive grids faster in CPU time than a conventional solver.

7 Concluding Remarks

In this paper, we developed a third-order hyperbolic Navier-Stokes solver for two-dimensional unsteady simulations on adaptive space-time tetrahedral grids. Numerical results are presented to demonstrate that the third-order HNS solver is more efficient than a conventional Navier-Stokes solver and is capable of producing highly accurate gradients on adaptive tetrahedral grids. In particular, we have shown that the third-order HNS solver can produce more accurate solutions and gradients that the second-order HNS solver at little additional cost.

In this work, we focused on the HNS formulation with spatial gradients only. It is possible and may be interesting to include the temporal derivatives of the solutions as additional unknowns. It will allow us to construct more efficient variants of the HNS schemes, Scheme II and Scheme IQ [10]. In the future work, we plan to apply the third-order HNS solver to compute an unsteady flow over a pitching airfoil as an example of simulating an unsteady flow with a moving body by a single nonlinear solve in a static space-time grid (nothing is moving in the space-time domain).

Acknowledgments

This work has been funded by the U.S. Army Research Office under the contract/grant number W911NF-19-1-0429.

References

[1] H. Nishikawa and E. Padway. An adaptive space-time edge-based solver for two-dimensional unsteady inviscid flows. In *AIAA Aviation 2020 Forum*, AIAA Paper 2020-3024, 2020.

[2] E. Padway and H. Nishikawa. An adaptive space-time edge-based solver for two-dimensional unsteady viscous flows. In *SciTech 2022 Forum*, AIAA Paper 2022-2201, San Diego, CA, 2022.

[3] E. Padway and H. Nishikawa. An adaptive space-time hyperbolic Navier-Stokes solver for two-dimensional unsteady viscous flows. In *Eleventh International Conference on Computational Fluid Dynamics*, Maui, Hawaii, 2022.

[4] L. Wang and P.-O. Persson. A high-order discontinuous Galerkin method with unstructured space-time meshes for two-dimensional compressible flows on domains with large deformations. *Comput. Fluids*, 118:53–68, 2015.

[5] R. D. Falgout, S. Friedhoff, Tz. V. Kolev, S. P. MacLachlan, J. B. Schroder, and S. Vandewalle. Multigrid methods with space-time concurrency. *Computing and Visualization in Science*, 18:123–143, 2017.

[6] S. Jayasinghe, D. L. Darmofal, N. K. Burgess, M. C. Galbraith, and S. R. Allmaras. A space-time adaptive method for reservoir flows: Formulation and one-dimensional application. *Comput. Geosci.*, 2:107–123, 2018.

[7] I. Flamarique Ederra, T. C. S. Rendall, A. L. Gaitonde, D. Jones, and C. B. Allen. A spacetime formulation for unsteady aerodynamics with geometry and topology changes. *Aeronautical Journal*, 126:1771–1800, 2022.

[8] H. Nishikawa and Y. Liu. Accuracy-preserving source term quadrature for third-order edge-based discretization. *J. Comput. Phys.*, 344:595–622, 2017.

[9] H. Nishikawa. Accuracy-preserving boundary flux quadrature for finite-volume discretization on unstructured grids. *J. Comput. Phys.*, 281:518–555, 2015.

[10] Y. Liu and H. Nishikawa. Third-order inviscid and second-order hyperbolic Navier-Stokes solvers for three-dimensional inviscid and viscous flows. In *46th AIAA Fluid Dynamics Conference*, AIAA Paper 2016-3969, Washington, D.C., 2016.

[11] H. Nishikawa. Alternative formulations for first-, second-, and third-order hyperbolic Navier-Stokes schemes. In *22nd AIAA Computational Fluid Dynamics Conference*, AIAA Paper 2015-2451, Dallas, TX, 2015.

[12] K. B. Thompson, H. Nishikawa, and E. Padway. Economical third-order methods for accurate surface heating predictions on simplex element meshes. In *AIAA SciTech 2023 Forum*, AIAA Paper 2023-2629, National Harbor, MD & Online, 2023.

[13] K. Masatsuka. I do like CFD, VOL.1, Second Edition. http://www.cfdbooks.com, 2013.

[14] H. Nishikawa. First, second, and third order finite-volume schemes for Navier-Stokes equations. In *7th AIAA Theoretical Fluid Mechanics Conference, AIAA Aviation and Aeronautics Forum and Exposition 2014*, AIAA Paper 2014-2091, Atlanta, GA, 2014.

[15] L. Li, J. Lou., H. Nishikawa, and H. Luo. Reconstructed discontinuous Galerkin methods for compressible flows based on a new hyperbolic Navier-Stokes system. *J. Comput. Phys.*, 427:110058, 2021.

[16] H. Nishikawa and Y. Liu. Hyperbolic advection-diffusion schemes for high-Reynolds-number boundary-layer problems. *J. Comput. Phys.*, 352:23–51, 2018.

[17] P. L. Roe. Approximate Riemann solvers, parameter vectors, and difference schemes. *J. Comput. Phys.*, 43:357–372, 1981.

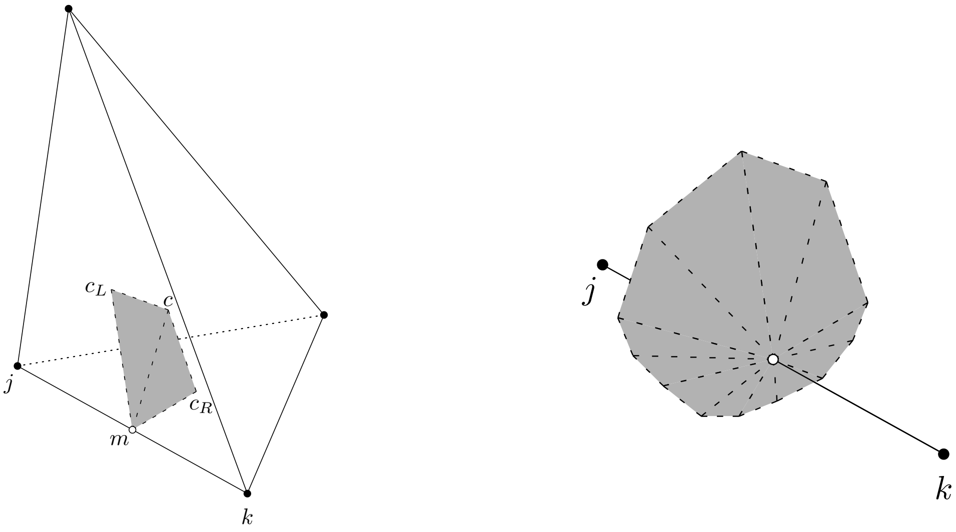
[18] H. Nishikawa. First, second, and third order finite-volume schemes for advection-diffusion. *J. Comput. Phys.*, 273:287–309, 2014.

[19] M. J. Pandya, B. Diskin, J. L. Thomas, and N. T. Frink. Improved convergence and robustness of USM3D solutions on mixed element grids. *AIAA J.*, 54(9):2589–2610, September 2016.

[20] H. Nishikawa, B. Diskin, and J. L. Thomas. Critical study of agglomerated multigrid methods for diffusion. *AIAA J.*, 48(4):839–847, April 2010.

[21] J. L. Thomas, B. Diskin, and H. Nishikawa. A critical study of agglomerated multigrid methods for diffusion on highly-stretched grids. *Comput. Fluids*, 41(1):82–93, February 2011.

[22] H. Nishikawa, Y. Nakashima, and N. Watanabe. Effects of high-frequency damping on iterative convergence of implicit viscous solver. *J. Comput. Phys.*, 348:66–81, 2017.



(a) Dual face contributions to the directed-area vector from an adjacent tetrahedral element to the edge.

(b) Total dual face at the edge contributing to the directed-area vector.

Figure 1: Dual face contributions at the edge $[j, k]$. A numerical flux is evaluated at the midpoint of the edge $[j, k]$. In (a), 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.

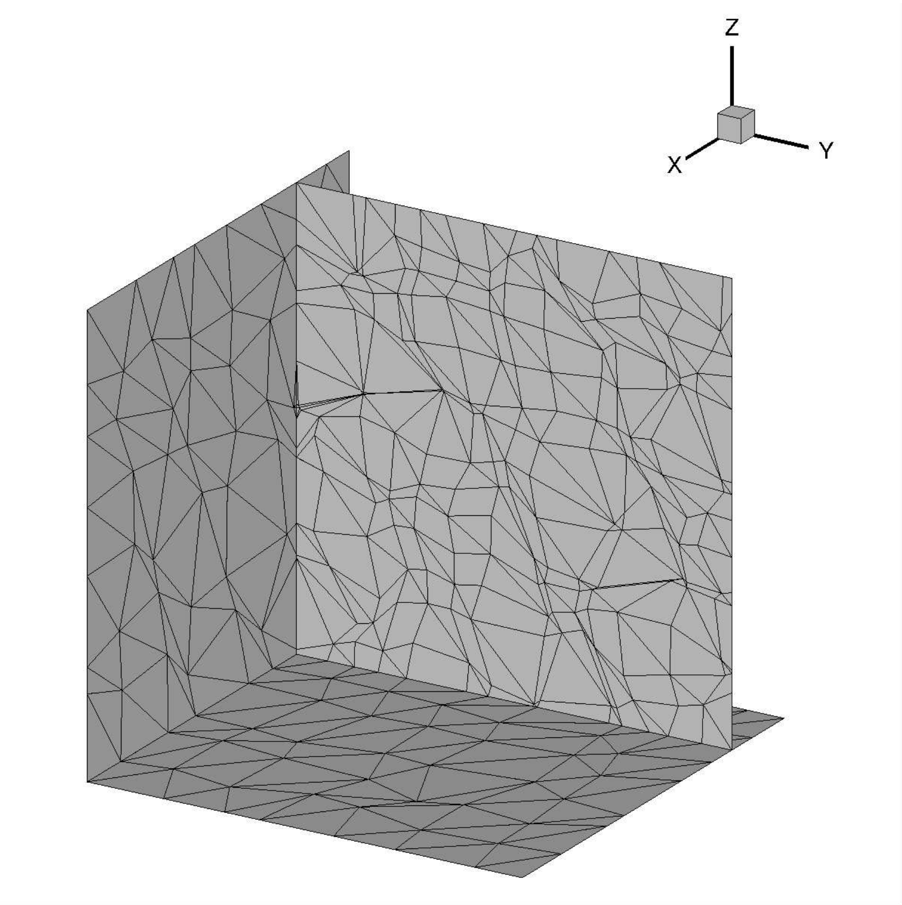
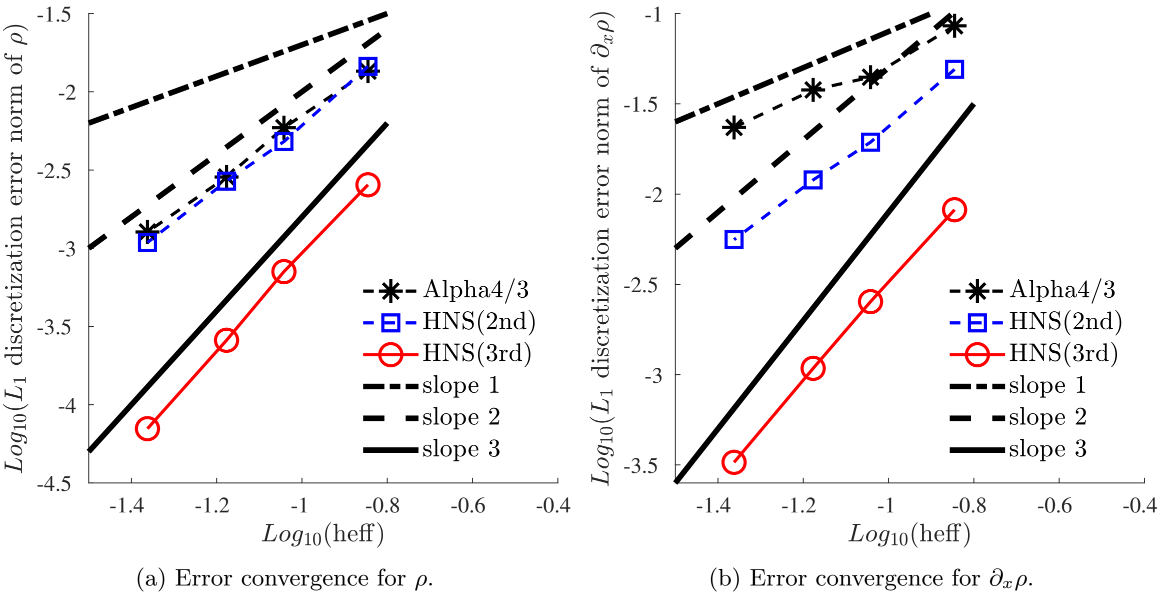


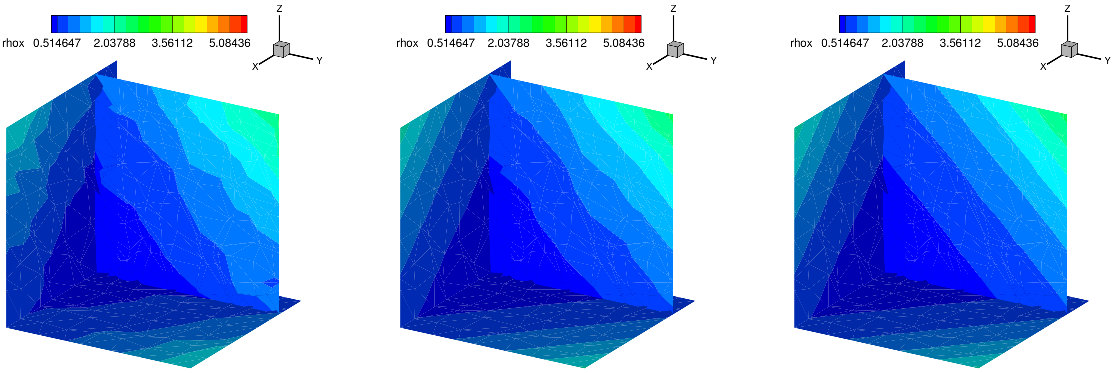
Figure 2: Coarsest grid in accuracy verification.



(a) Error convergence for ρ.

(b) Error convergence for $\partial_x\rho$.

Figure 3: Accuracy verification results: error convergence.

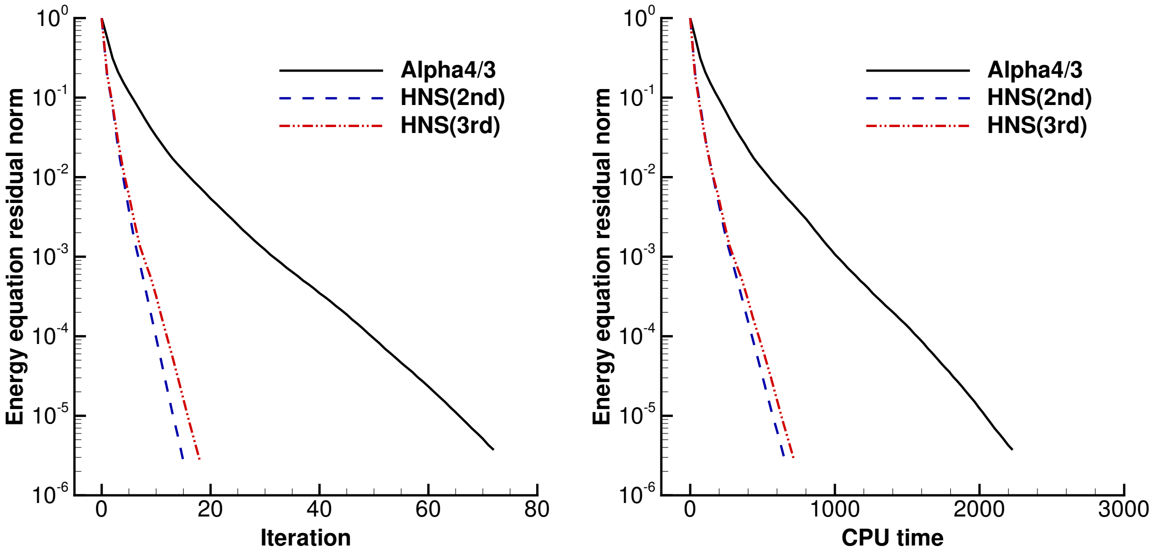

rhox 0.514647 2.03788 3.56112 5.08436
Z
X
Y
rhox 0.514647 2.03788 3.56112 5.08436
Z
X
Y
rhox 0.514647 2.03788 3.56112 5.08436
Z
X
Y

(a) Alpha4/3.

(b) HNS(2nd).

(c) HNS(3rd).

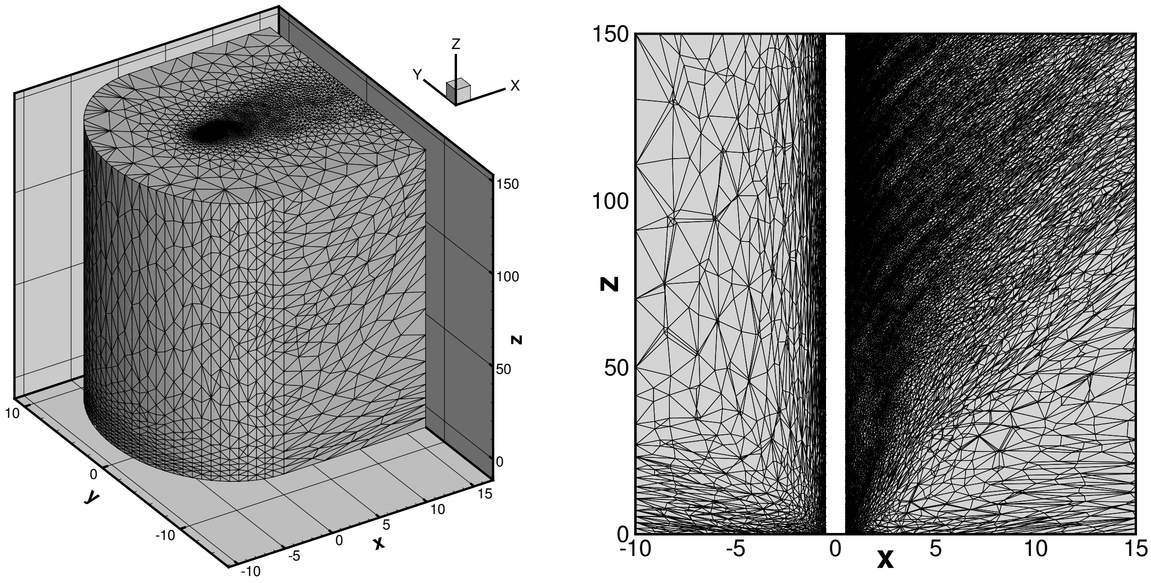
Figure 4: Accuracy verification results: gradient contours.



(a) Iterative convergence.

(b) Iterative convergence.

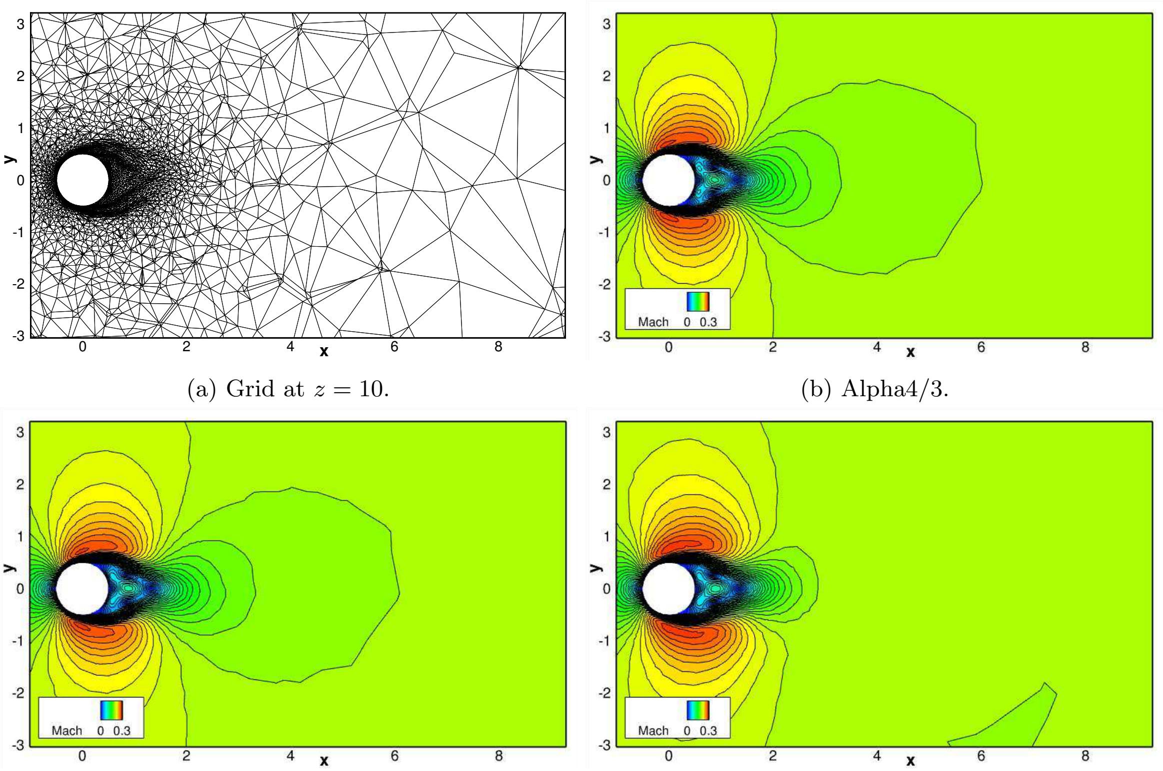
Figure 5: Accuracy verification results: iterative convergence.



(a) Grid.

(b) Section at $y = 0$.

Figure 6: Grid for the second cylinder test case.



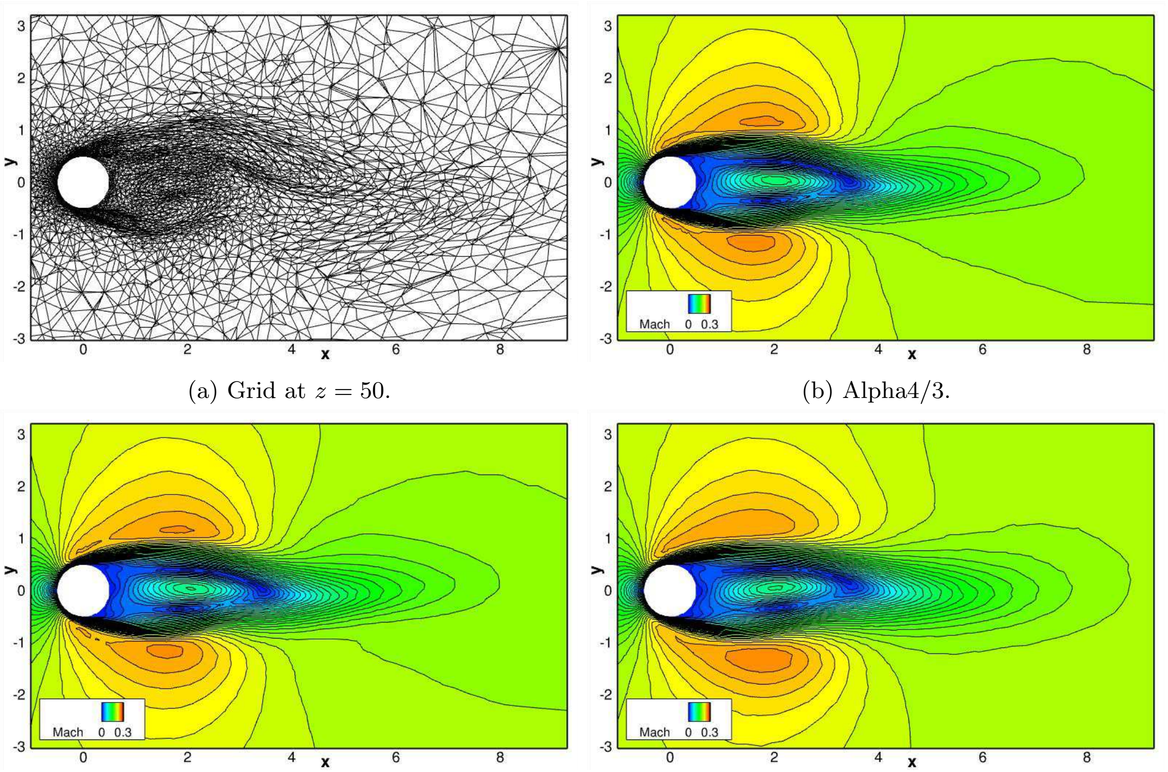
(a) Grid at $z = 10$.

(b) Alpha4/3.

(c) HNS(2nd).

(d) HNS(3rd).

Figure 7: Section plots for $z = 10$ in the cylinder test case.



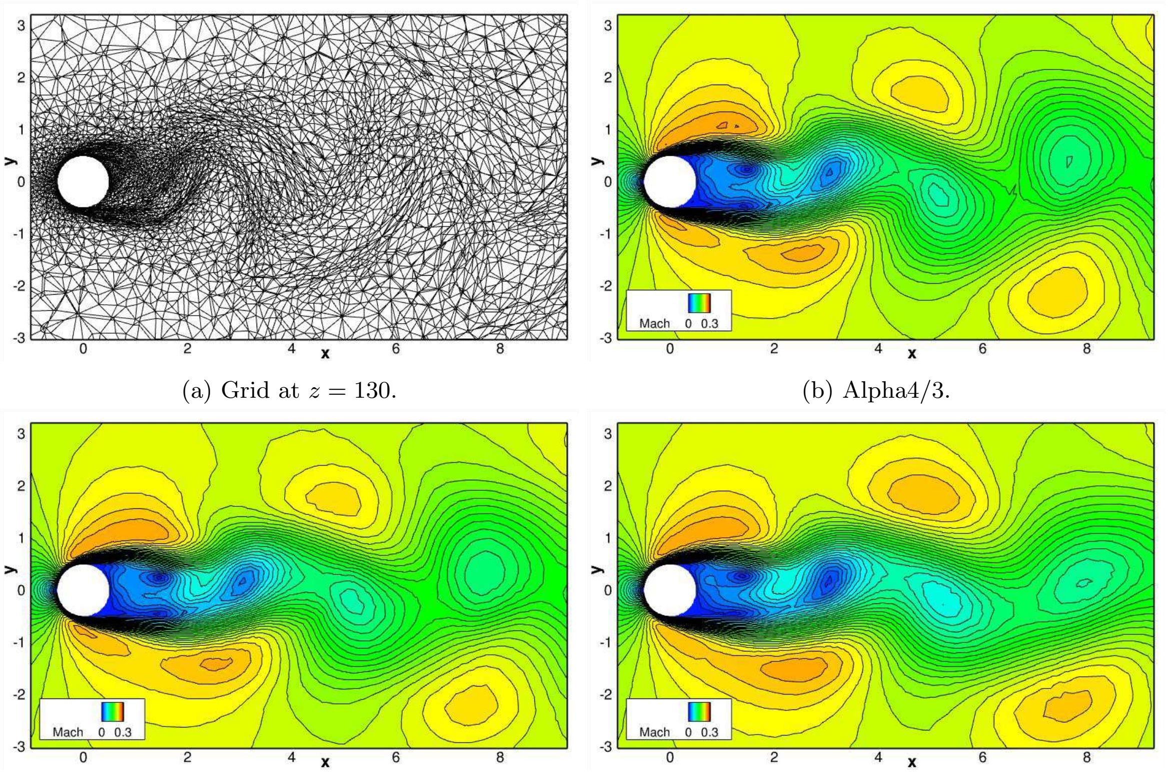
(a) Grid at $z = 50$.

(b) Alpha4/3.

(c) HNS(2nd).

(d) HNS(3rd).

Figure 8: Section plots for $z = 50$ in the cylinder test case.



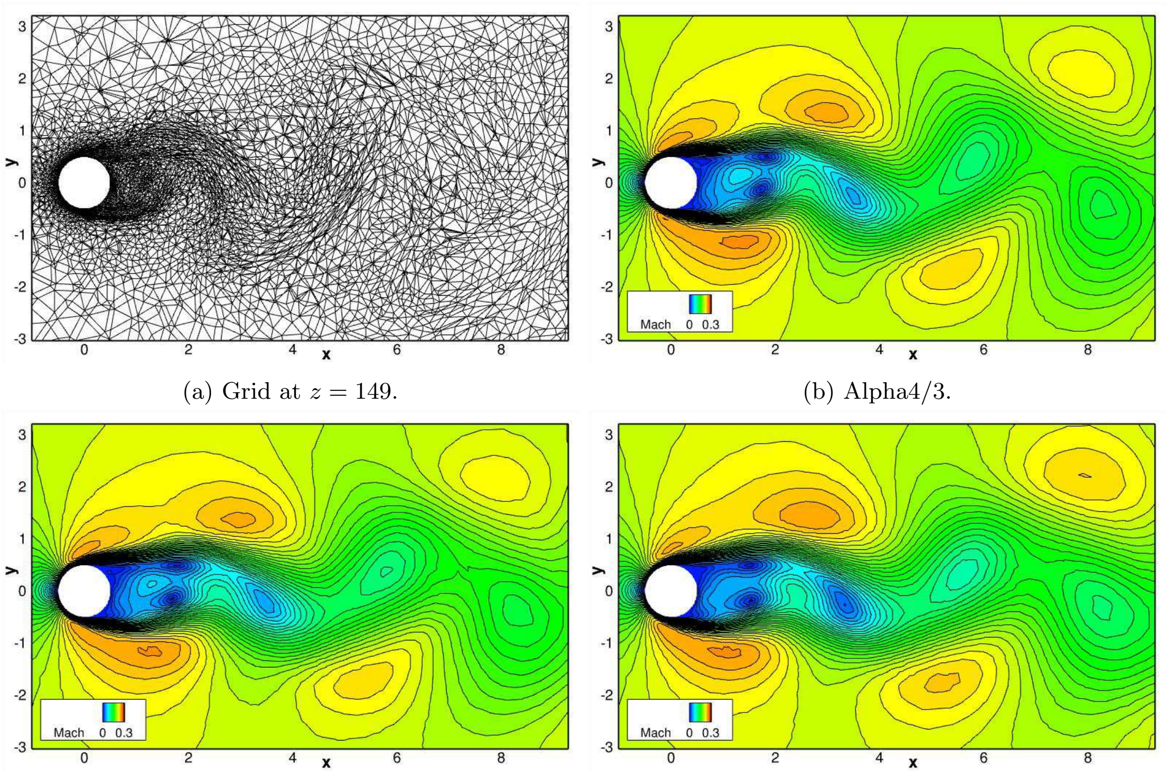
(a) Grid at $z = 130$.

(b) Alpha4/3.

(c) HNS(2nd).

(d) HNS(3rd).

Figure 9: Section plots for $z = 130$ in the cylinder test case.



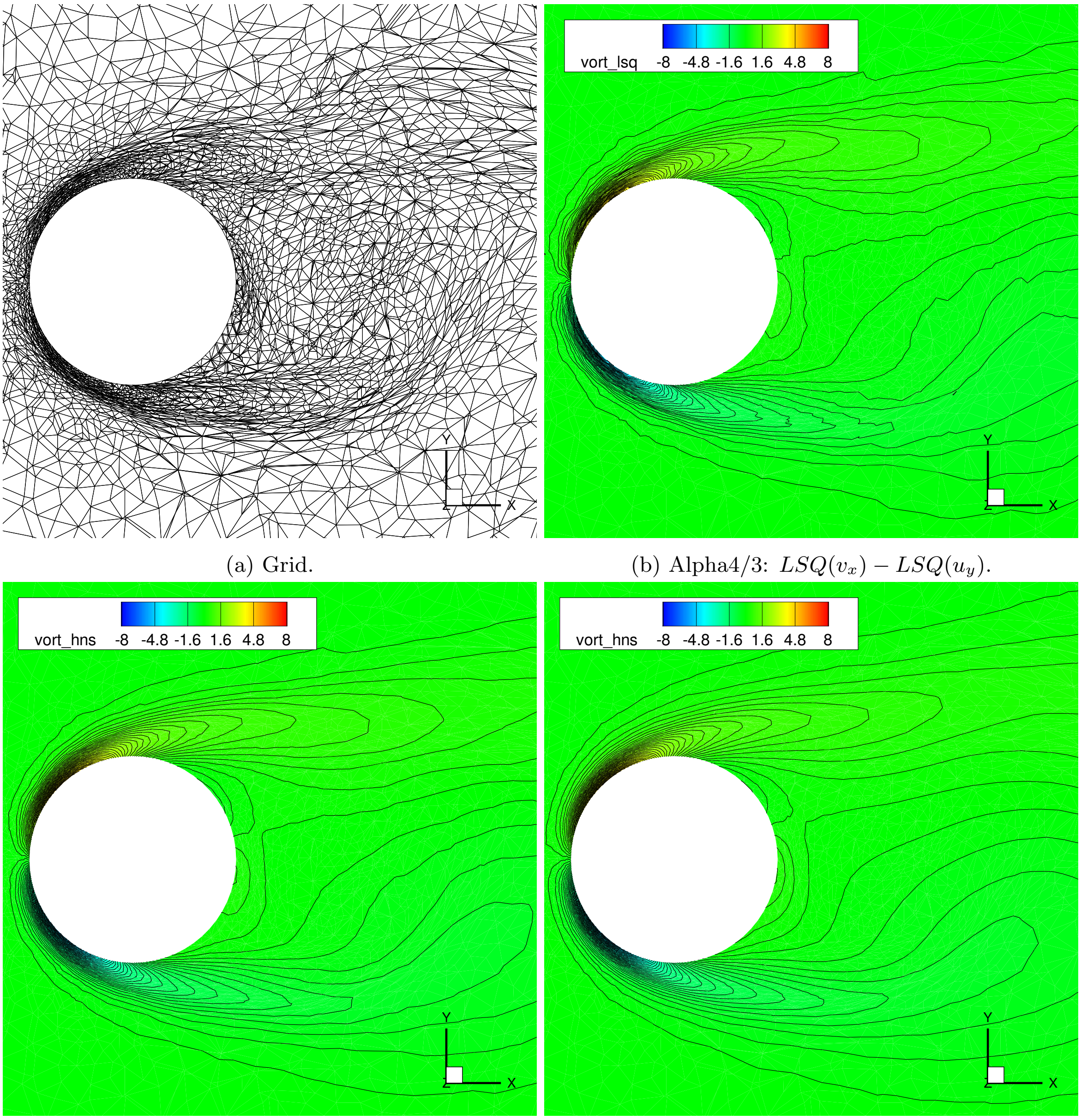
(a) Grid at $z = 149$.

(b) Alpha4/3.

(c) HNS(2nd).

(d) HNS(3rd).

Figure 10: Section plots for $z = 149$ in the cylinder test case.



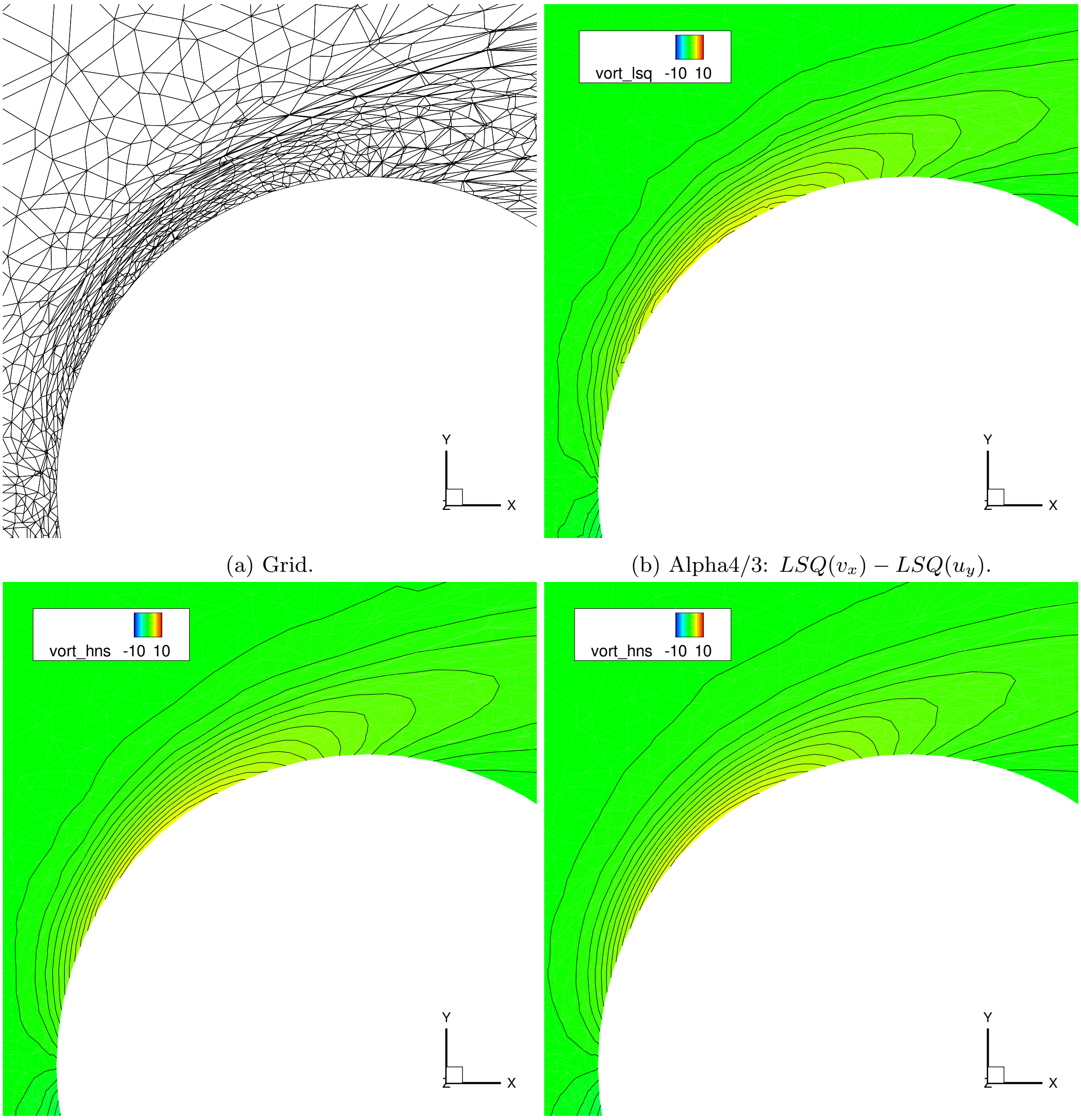
(a) Grid.

(b) Alpha4/3: $LSQ(v_x) - LSQ(u_y)$.

(c) HNS(2nd): $g_{vx} - g_{uy}$.

(d) HNS(3rd): $g_{vx} - g_{uy}$.

Figure 11: Grids and vorticity contours at $z = 149$ for the second cylinder test case.



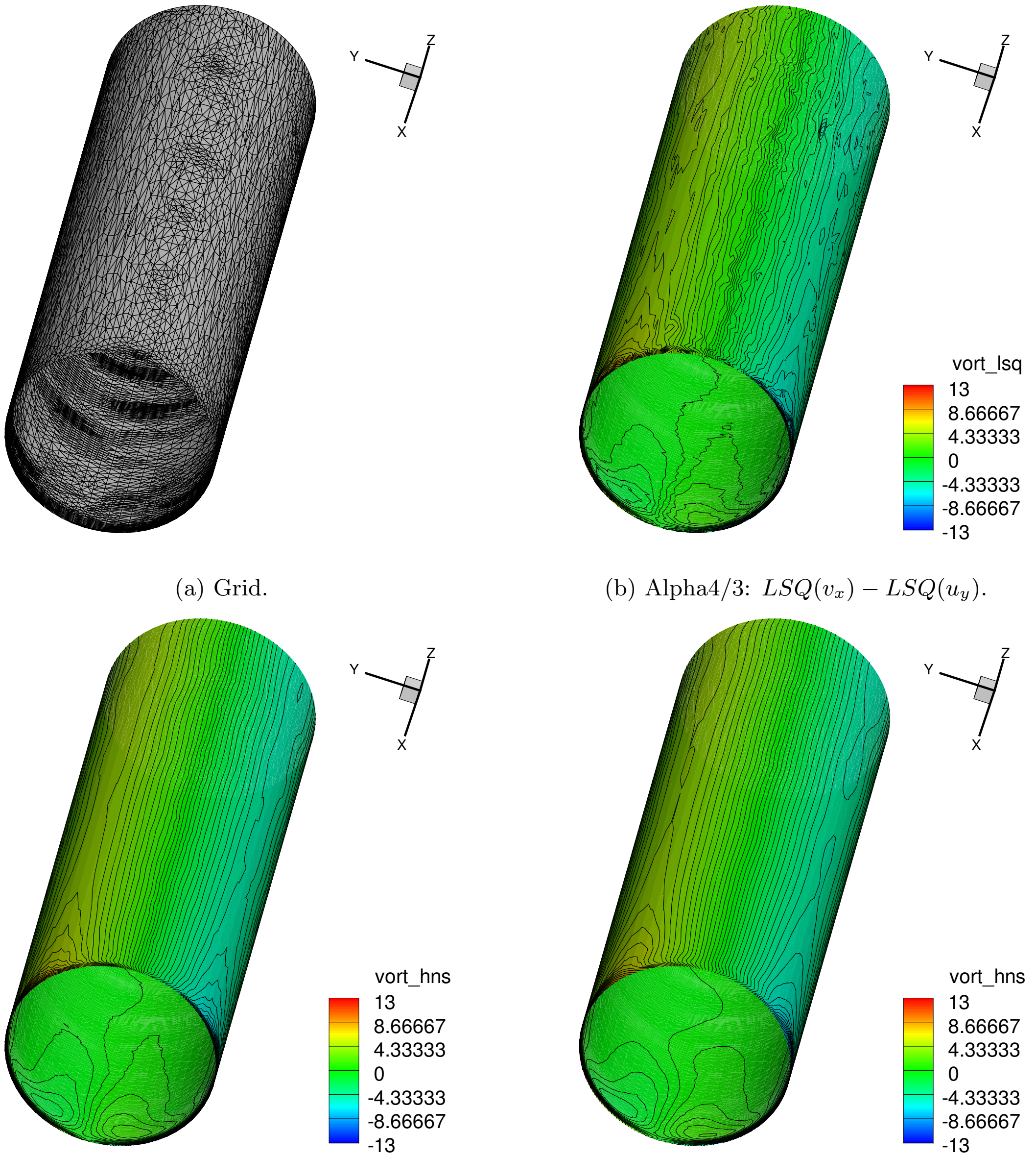
(a) Grid.

(b) Alpha4/3: $LSQ(v_x) - LSQ(u_y)$.

(c) HNS(2nd): $g_{vx} - g_{uy}$.

(d) HNS(3rd): $g_{vx} - g_{uy}$.

Figure 12: Grids and vorticity contours at $z = 149$ for the cylinder test case (zoomed-in views).



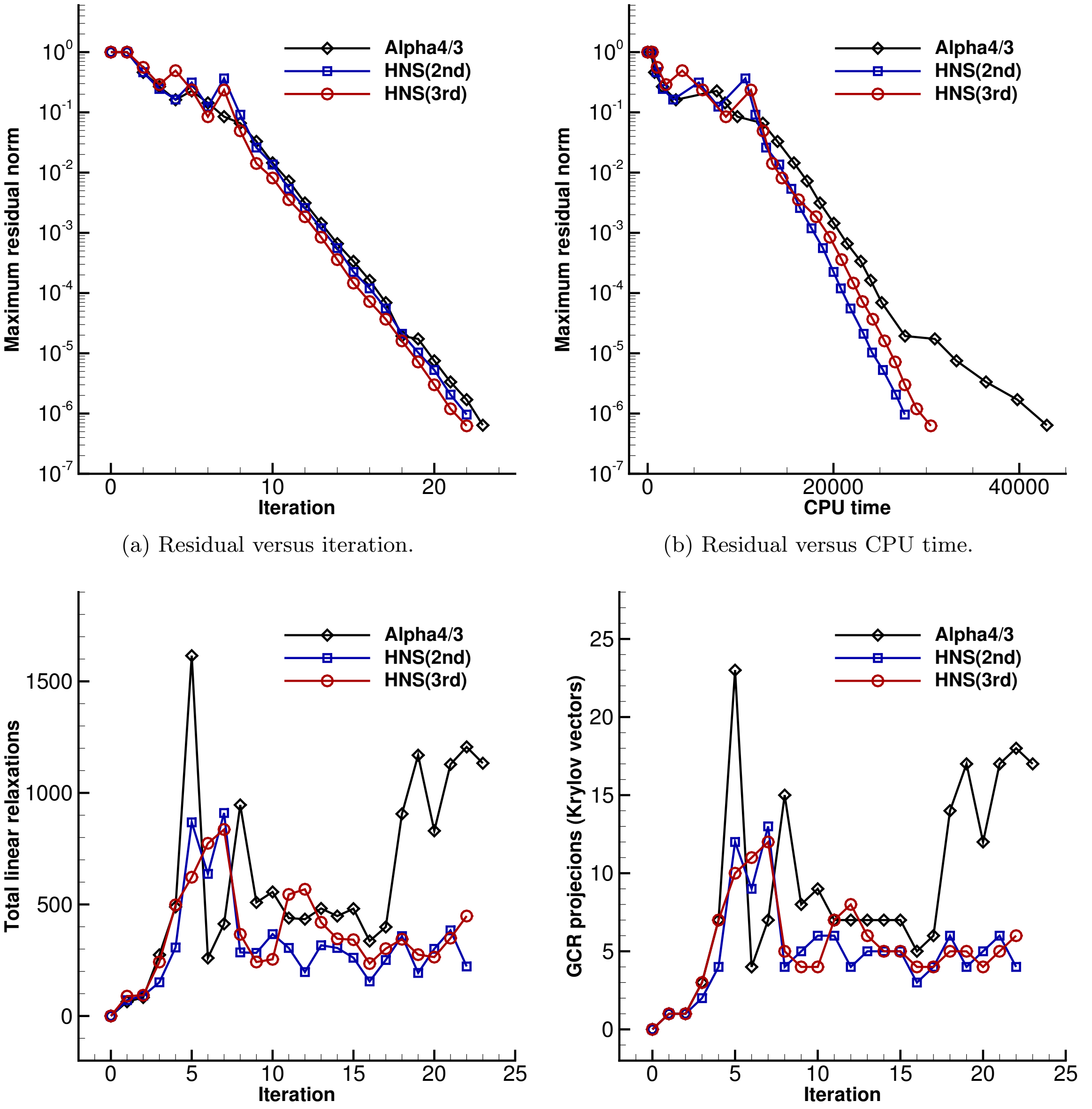
(a) Grid.

(b) Alpha4/3: $LSQ(v_x) - LSQ(u_y)$.

(c) HNS(2nd): $g_{vx} - g_{uy}$.

(d) HNS(3rd): $g_{vx} - g_{uy}$.

Figure 13: Grid and vorticity contours over the circular cylinder surface.



(a) Residual versus iteration.

(b) Residual versus CPU time.

(c) Total linear relaxations versus iteration.

(d) GCR projections (Krylov vectors) versus iteration.

Figure 14: Residual convergence results for the cylinder test case.