Eleventh International Conference on
Computational Fluid Dynamics (ICCFD11),
Maui, HI, USA, July 11-15, 2022 **ICCFD11-2022-3001**

An Adaptive Space-Time Hyperbolic Navier-Stokes Solver for Two-Dimensional Unsteady Viscous Flows

E. Padway and H. Nishikawa
Corresponding author: emmett.padway@nianet.org

National Institute of Aerospace, Hampton, VA, USA.

Abstract: In this paper, we present an adaptive-grid space-time solver based on a hyperbolic Navier-Stokes formulation for two-dimensional unsteady viscous flows, where the two-dimensional Navier-Stokes equations are discretized and solved as a steady system in a three-dimensional space-time domain with the coordinates (x, y, t), where t denotes time, using adaptive tetrahedral grids. The hyperbolic viscous formulation drastically simplifies the viscous discretization and greatly improves the accuracy and quality of derivatives on adaptive grids with convergence acceleration. These advantages are demonstrated for space-time computations for two-dimensional unsteady viscous problems.

Keywords: Hyperbolic Navier-Stokes, Space-Time, Adaptive Grids.

1 Introduction

Laminar-flow control is a key technology for achieving economical access to space and efficient flights with extended range by reducing aerodynamic drag and heating. In particular, physics-based boundary-layer transition control (BLT) techniques have been widely studied as an effective means to perform laminar-flow control, including hypersonic-flow applications. Yet, these techniques need to be integrated into a practical unstructured-grid computational-fluid-dynamics solver, in order to perform high- and multi-fidelity design optimizations for realistic geometries of interest. Towards the grand challenge of optimizations with BLT control strategies, this work explores the development of an efficient unsteady Navier-Stokes solver for adaptive unstructured grids based on a space-time method.

Space-time methods, where unsteady governing equations are solved in a larger space with the physical time treated as an additional coordinate, are promising approaches for exploiting an ever-increasing parallelism. The methods are especially efficient when combined with anisotropic grid adaptation as we demonstrated in our previous papers [1, 2]. However, solution derivatives obtained on adaptive grids are often contaminated with numerical noise, which is a serious issue for viscous simulations, where target quantities depend on solution derivatives (e.g., viscous stresses and heat fluxes). To address this issue, we develop a new space-time solver based on a hyperbolic Navier-Stokes (HNS) formulation [3], which has been demonstrated to achieve superior accuracy and quality in the solution derivatives on irregular tetrahedral grids [4, 5, 6]. In particular, we consider the HNS formulation called HNS20G [3] in two dimensions:

$$\frac{\partial \mathbf{u}}{\partial t} + \frac{\partial \mathbf{f}_x}{\partial x} + \frac{\partial \mathbf{f}_y}{\partial y} = 0, \tag{1}$$

where $\mathbf{u}$ is a vector of conservative variables and the gradients of the primitive variables (12 variables in total), $\mathbf{f}_x$ and $\mathbf{f}_y$ are fluxes in the x- and y-directions. This system consists of hyperbolic systems (time, inviscid, viscous), therefore can be discretized by an upwind method, and then solved as a steady system in a three-dimensional space with the coordinates $(x, y, z = t)$. Accurate derivatives are obtained directly in the extra variables rather than computed from the numerical solution, for example, by a least-squares method. Moreover, this approach has the advantage of not requiring any special fix to avoid zero division at an edge aligned with the time axis as encountered in a conventional viscous discretization [2].

In this paper, we focus on the simplest algorithm of the space-time HNS method and demonstrate its potential for space-time computations. Further developments towards more robust, accurate, and efficient space-time HNS solvers will be reported in the future. For example, a more accurate HNS solver can be constructed by upgrading least-squares gradients with gradients of gradient variables incorporated as second derivatives. Also, the method can be extended to achieve third-order accuracy in both inviscid and viscous terms, and at the same time, third-order accuracy will be achieved in the gradients as well. The equal order of accuracy in the solution variables and their gradients is another advantage of the HNS method. It is demonstrated here for a second-order discretization method.

2 Unsteady Compressible Navier-Stokes Equations

Our target equations are the unsteady compressible NS equations:

$$\partial_t \rho + \operatorname{div}(\rho \mathbf{v}) = 0, \tag{2}$$

$$\partial_t(\rho \mathbf{v}) + \operatorname{div}(\rho \mathbf{v} \otimes \mathbf{v}) = -\operatorname{grad} p + \operatorname{div} \boldsymbol{\tau}, \tag{3}$$

$$\partial_t(\rho E) + \operatorname{div}(\rho \mathbf{v} H) = \operatorname{div}(\boldsymbol{\tau} \mathbf{v}) - \operatorname{div} \mathbf{q}, \tag{4}$$

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

$$\boldsymbol{\tau} = -\frac{2}{3}\mu(\operatorname{div} \mathbf{v})\mathbf{I} + \mu \left(\operatorname{grad} \mathbf{v} + (\operatorname{grad} \mathbf{v})^t\right), \quad \mathbf{q} = -\frac{\mu}{Pr(\gamma - 1)} \operatorname{grad} T, \tag{5}$$

where $\mathbf{I}$ is the identity matrix, T is the temperature, γ is the ratio of specific heats, 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.[7]). Thus, the viscosity is given by the following form of Sutherland's law:

$$\mu = \frac{M_\infty}{Re_\infty} \frac{1 + C/\tilde{T}_\infty}{T + C/\tilde{T}_\infty} T^{\frac{3}{2}}, \tag{6}$$

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:

$$\gamma p = \rho T. \tag{7}$$

In the previous paper [2], we discretized the Navier-Stokes system directly and developed a space-time viscous solver. In this work, we will first reformulate the viscous terms as a hyperbolic system and then discretize it by an upwind discretization just like the inviscid terms.

3 Hyperbolic Navier-Stokes System: HNS20G

There exist four possible hyperbolic formulations for the viscous terms: HNS14 [4], HNS17 [5], HNS20 [5], and HNS20G [3]. Here, we employ HNS20G, which was first introduced in Ref.[3] to simplify the development of high-order reconstructed discontinuous Galerkin methods. In HNS20G, we extend the Navier-Stoke 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:

$$\begin{aligned}
\partial_\tau \rho + \partial_t \rho + \text{div}(\rho \mathbf{v}) &= \text{div}\,(\nu_r \mathbf{r})\,, && (8)\\
\partial_\tau(\rho\mathbf{v}) + \partial_t(\rho\mathbf{v}) + \text{div}(\rho\mathbf{v}\otimes\mathbf{v}) &= -\text{grad}\,p + \text{div}\,(\mu_v \widetilde{\boldsymbol{\tau}})\,, && (9)\\
\partial_\tau(\rho E) + \partial_t(\rho E) + \text{div}(\rho\mathbf{v}H) &= \text{div}\,(\mu_v \widetilde{\boldsymbol{\tau}}\mathbf{v}) \;+\; \text{div}\left(\frac{\mu_h}{\gamma(\gamma-1)}\mathbf{h}\right), && (10)\\
T_v\,\partial_\tau \mathbf{g} &= \text{grad}\,\mathbf{v} - \mathbf{g}, && (11)\\
T_h\,\partial_\tau \mathbf{h} &= \text{grad}\,T - \mathbf{h},\,, && (12)\\
T_r\,\partial_\tau \mathbf{r} &= \text{grad}\,\rho - \mathbf{r}, && (13)
\end{aligned}$$

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

$$T_r = \frac{L_r^2}{\nu_r}, \quad T_v = \frac{L_v^2}{\nu_v}, \quad T_h = \frac{L_h^2}{\nu_h}, \quad \nu_v = \frac{4\mu}{3\rho}, \quad \nu_h = \frac{\gamma\mu}{\rho Pr}, \quad \nu_r = \min(10^{-12}, V_{min}^{4/3})M_\infty, \tag{14}$$

and the length scale may be taken as $L_r = L_v = L_h = 1/(2\pi)$ or, as in Refs.[3, 8],

$$\begin{aligned}
&L_r = L_d = \frac{1}{2\pi},\\
&L_v = \frac{L_d}{\sqrt{Re_{L_d}^{v\infty}}}, \quad Re_{L_d}^{v\infty} = \frac{U_\infty L_d}{\nu_{v\infty}},\\
&L_h = \frac{L_d}{\sqrt{Re_{L_d}^{h\infty}}}, \quad Re_{L_d}^{h\infty} = \frac{U_\infty L_d}{\nu_{h\infty}},
\end{aligned} \tag{15}$$

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. Note that L_r is different from that in Refs.[3, 8]. For a particular discretization employed in this work (called Scheme-I [9, 6]), it has been observed that the order of accuracy of the density gradient is reduced by one order if L_r is too small, and therefore we keep $L_r = L_d = \frac{1}{2\pi}$.

Note that the HNS20G system is equivalent to the original Navier-Stokes system in a pseudo steady state or as soon as the pseudotime derivatives are dropped. Then, the gradient variables $\mathbf{h}$ and $\mathbf{g}$ correspond to the temperature and velocity gradients:

$$\mathbf{h} = \text{grad}\,T, \quad \mathbf{g} = \text{grad}\,\mathbf{v}. \tag{16}$$

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 Navier-Stokes system in a first-order form.

In two dimensions, we write the HNS20G system as

$$\mathbf{P}^{-1}\frac{\partial \mathbf{u}}{\partial \tau} + \mathbf{M}\frac{\partial \mathbf{u}}{\partial t} + \frac{\partial \mathbf{f}_x}{\partial x} + \frac{\partial \mathbf{f}_y}{\partial y} = \mathbf{s}, \tag{17}$$

where

$$\mathbf{P}^{-1} = \text{diag}(1,1,1,1,T_r,T_r,T_v,T_v,T_v,T_v,T_h,T_h), \tag{18}$$

$$\mathbf{M} = \text{diag}(1,1,1,1,0,0,0,0,0,0,0,0), \tag{19}$$

$$\mathbf{u}=\begin{bmatrix}\rho\\ \rho u\\ \rho v\\ \rho E\\ g_{ux}\\ g_{uy}\\ g_{vx}\\ g_{vy}\\ h_x\\ h_y\\ r_x\\ r_y\end{bmatrix},\quad \mathbf{f}_x=\mathbf{f}_x^{inv}+\mathbf{f}_x^{vis}=\begin{bmatrix}\rho u\\ \rho u^2+p\\ \rho uv\\ \rho uH\\ 0\\ 0\\ 0\\ 0\\ 0\\ 0\\ 0\\ 0\end{bmatrix}+\begin{bmatrix}0\\ -\mu_v\widetilde{\tau}_{xx}\\ -\mu_v\widetilde{\tau}_{xy}\\ -\mu_v\widetilde{\tau}_{xx}u-\mu_v\widetilde{\tau}_{xy}v+\frac{\mu_h}{\gamma(\gamma-1)}h_x\\ -u\\ 0\\ -v\\ 0\\ -T\\ 0\\ -\rho\\ 0\end{bmatrix}, \tag{20}$$

$$\mathbf{f}_y=\mathbf{f}_y^{inv}+\mathbf{f}_y^{vis}=\begin{bmatrix}\rho v\\ \rho uv\\ \rho v^2+p\\ \rho vH\\ 0\\ 0\\ 0\\ 0\\ 0\\ 0\\ 0\\ 0\end{bmatrix}+\begin{bmatrix}0\\ -\mu_v\widetilde{\tau}_{yx}\\ -\mu_v\widetilde{\tau}_{yy}\\ -\mu_v\widetilde{\tau}_{yx}u-\mu_v\widetilde{\tau}_{yy}v+\frac{\mu_h}{\gamma(\gamma-1)}h_y\\ 0\\ -u\\ 0\\ -v\\ 0\\ -T\\ 0\\ -\rho\end{bmatrix},\quad \mathbf{s}=\begin{bmatrix}s_1(x,y,t)\\ s_2(x,y,t)\\ s_3(x,y,t)\\ s_4(x,y,t)\\ -g_{ux}\\ -g_{uy}\\ -g_{vx}\\ -g_{vy}\\ -h_x\\ -h_y\\ -r_x\\ -r_y\end{bmatrix}, \tag{21}$$

where s_1, s_2, s_3, s_4 are forcing terms. As derived in Ref.[3], the dissipation vector of an upwind flux for the viscous part is given by

$$\left|\mathbf{P}\frac{\partial \mathbf{f}_n^{vis}}{\partial \mathbf{u}}\right|\Delta\mathbf{u}=\left(\begin{array}{c} 0 \\ \\ \rho[(a_{nv}-a_{mv})\Delta u_n\mathbf{n}+a_{mv}\Delta\mathbf{v}] \\ \\ \rho\mathbf{v}\cdot\{(a_{nv}-a_{mv})\Delta u_n\mathbf{n}+a_{mv}\Delta\mathbf{v}\}+\dfrac{\rho a_h\Delta T}{\gamma(\gamma-1)}+\dfrac{\mu_v^2}{\rho a_h}(\mathcal{L}_\tau\cdot\Delta\widetilde{\boldsymbol{\tau}}_n) \\ \\ \left[\left(a_{nv}-\dfrac{4}{3}a_{mv}\right)\Delta\widetilde{\tau}_{nn}n_x+\dfrac{4}{3}a_{mv}\Delta\widetilde{\tau}_{nx}\right]\mathbf{n} \\ \\ \left[\left(a_{nv}-\dfrac{4}{3}a_{mv}\right)\Delta\widetilde{\tau}_{nn}n_y+\dfrac{4}{3}a_{mv}\Delta\widetilde{\tau}_{ny}\right]\mathbf{n} \\ \\ a_h\left[\Delta h_n+(\mathcal{L}_\tau\cdot\Delta\mathbf{v})\right]\mathbf{n} \\ \\ \mathbf{0} \end{array}\right), \tag{22}$$

where $\mathbf{f}_n^{vis}=(\mathbf{f}_x^{vis},\mathbf{f}_y^{vis})\cdot\mathbf{n}$, Δ denotes a jump across two states, e.g., $\Delta\mathbf{v}=\mathbf{v}_R-\mathbf{v}_L$, $\mathbf{v}=(u,v)^t$, $\mathbf{n}=(n_x,n_y)^t$ is a unit vector, and

$$u_n=\mathbf{v}\cdot\mathbf{n},\quad h_n=\mathbf{h}\cdot\mathbf{n},\quad \widetilde{\boldsymbol{\tau}}_n=\widetilde{\boldsymbol{\tau}}\mathbf{n},\quad \mathcal{L}_\tau=\frac{\widetilde{\tau}_{nn}\mathbf{n}}{Pr_n+1}+\frac{\widetilde{\boldsymbol{\tau}}_n-\widetilde{\tau}_{nn}\mathbf{n}}{Pr_m+1}, \tag{23}$$

$$\widetilde{\tau}_{nm}=\widetilde{\tau}_{nx}m_x+\widetilde{\tau}_{ny}m_y,\quad \widetilde{\tau}_{nn}=\widetilde{\tau}_{nx}n_x+\widetilde{\tau}_{ny}n_y, \tag{24}$$

$$\widetilde{\tau}_{nx}=\widetilde{\tau}_{xx}n_x+\widetilde{\tau}_{xy}n_y,\quad \widetilde{\tau}_{ny}=\widetilde{\tau}_{xy}n_x+\widetilde{\tau}_{yy}n_y, \tag{25}$$

$$Pr_n=\frac{a_{nv}}{a_h},\quad Pr_m=\frac{a_{mv}}{a_h},\quad Pr_l=\frac{a_{lv}}{a_h}=\frac{a_{mv}}{a_h}=Pr_m, \tag{26}$$

$$a_{nv}=\sqrt{\frac{\nu_v}{T_v}},\quad a_{mv}=\sqrt{\frac{3\nu_v}{4T_v}},\quad a_h=\sqrt{\frac{\nu_h}{T_h}}. \tag{27}$$

All quantities in the dissipation matrix $\left|\mathbf{P}\frac{\partial \mathbf{f}_n^{vis}}{\partial \mathbf{u}}\right|$ are evaluated with arithmetic averages of the two states, e.g., $\mathbf{v}=(\mathbf{v}_L+\mathbf{v}_R)/2$.

4 Second-Order Edge-Based Discretization

We drop the pseudotime term and discretize the HNS20G system by the second-order edge-based method over a tetrahedral grid. The residual at a node j is defined by

$$\mathbf{Res}_j\equiv\sum_{k\in\{k_j\}}\mathbf{\Phi}_{jk}A_{jk}-\mathbf{s}_jV_j, \tag{28}$$

where V_j is the dual control volume, $\{k_j\}$ is a set of neighbor nodes of j, A_{jk} is the magnitude of the directed area vector $\mathbf{n}_{jk}=(n_x,n_y,n_t)$, which is a sum of the directed-areas corresponding to the dualtriangular faces associated with all tetrahedral elements sharing the edge $[j,k]$ (see Figure 1), and $\mathbf{\Phi}_{jk}$ is a numerical flux. The numerical flux is computed with the left and right states in the primitive variables $\mathbf{w}_L$ and $\mathbf{w}_R$, respectively, where

$$\mathbf{w} = (\rho, u, v, p, g_{ux}, g_{uy}, g_{vx}, g_{vy}, h_x, h_y, r_x, r_y)^t, \tag{29}$$

by using the kappa reconstruction scheme [10, 11]:

$$\mathbf{w}_L \quad = \quad \kappa \frac{\mathbf{w}_j + \mathbf{w}_k}{2} + (1-\kappa)\left[\mathbf{w}_j + \frac{1}{2}\nabla \mathbf{w}_j^{LSQ} \cdot (\mathbf{x}_k - \mathbf{x}_j)\right], \tag{30}$$

$$\mathbf{w}_R \quad = \quad \kappa \frac{\mathbf{w}_j + \mathbf{w}_k}{2} + (1-\kappa)\left[\mathbf{w}_k - \frac{1}{2}\nabla \mathbf{w}_k^{LSQ} \cdot (\mathbf{x}_k - \mathbf{x}_j)\right], \tag{31}$$

where κ is a real-valued parameter. The corresponding conservative variables $\mathbf{u}_L$ and $\mathbf{u}_R$ are algebraically computed from the primitive variables $\mathbf{w}_L$ and $\mathbf{w}_R$. The nodal gradients $\nabla \mathbf{w}_j^{LSQ}$ and $\nabla \mathbf{w}_k^{LSQ}$ are computed by an unweighted least-squares (LSQ) method with neighbor nodes. For the results shown in this paper, we used $\kappa = 1/2$.

Note that the LSQ gradients of the gradient variables correspond to second derivatives of the primitive variables. They can be incorporated in the LSQ method to upgrade the linear LSQ method to a quadratic one. The resulting scheme is called Scheme-IQ [6], in contrast to the scheme used in this paper, which is called Scheme-I [9, 6]. However, for the space-time solver, gradient variables corresponding to time derivatives are missing, and therefore Scheme-IQ will not be complete. In the future, we will explore Scheme-IQ by introducing additional equations for the time derivative variables and investigate the impact of not having the time derivative variables.

As in our previous papers [1], we construct the numerical flux as a sum of the 2D spatial inviscid and viscous fluxes and the temporal flux:

$$\mathbf{\Phi}_{jk} \quad = \quad \mathbf{\Phi}_{jk}^{inv}|(\hat{n}_x, \hat{n}_y)| + \mathbf{\Phi}_{jk}^{vis}|(\hat{n}_x, \hat{n}_y)| + \mathbf{\Phi}_{jk}^{time}|\hat{n}_t|, \tag{32}$$

where $\hat{\mathbf{n}}_{jk} = \mathbf{n}_{jk}/|\mathbf{n}_{jk}| = (\hat{n}_x, \hat{n}_y, \hat{n}_t)^t$, $\mathbf{\Phi}_{jk}^{inv}$ and $\mathbf{\Phi}_{jk}^{vis}$ are the spatial inviscid and viscous fluxes respectively and $\mathbf{\Phi}_{jk}^{time}$ is the temporal flux. For the spatial fluxes, we first set $\mathbf{n}_{xy} = (\hat{n}_x, \hat{n}_y)$, and define the unit directed-area vector $\hat{\mathbf{n}}_{xy}$ in space:

$$\hat{\hat{\mathbf{n}}}_{xy} = \frac{\hat{\mathbf{n}}_{xy}}{|\hat{\mathbf{n}}_{xy}|} = \frac{(\hat{n}_x, \hat{n}_y)^t}{|(\hat{n}_x, \hat{n}_y)|} = \frac{(\hat{n}_x, \hat{n}_y)^t}{\sqrt{\hat{n}_x^2 + \hat{n}_y^2}}. \tag{33}$$

Then, the inviscid flux $\mathbf{\Phi}_{jk}^{inv}$ is computed by the 2D Roe flux in the direction $\hat{\hat{\mathbf{n}}}_{xy}$. The time flux is computed by the upwind flux with $\hat{\hat{n}}_t = \hat{n}_t/|\hat{n}_t|$. See Ref.[2] for details. For the viscous flux $\mathbf{\Phi}_{jk}^{vis}$, since the viscous terms are reformulated here as a hyperbolic system, we use the following upwind flux:

$$\mathbf{\Phi}_{jk}^{vis} = \frac{1}{2}\left[\mathbf{f}_n^{vis}(\mathbf{w}_L) + \mathbf{f}_n^{vis}(\mathbf{w}_R)\right] - \frac{1}{2}\mathbf{P}^{-1}\left\{\left|\mathbf{P}\frac{\partial \mathbf{f}_n^{vis}}{\partial \mathbf{u}}\right|(\mathbf{u}_R - \mathbf{u}_L) + \Delta\Phi^{AHD}\right\}, \tag{34}$$

where the dissipation term is given by Equation (22) with $\mathbf{n} = \hat{\hat{\mathbf{n}}}_{xy}$, and the extra term $\Delta\Phi^{AHD}$ is the artificial hyperbolic dissipation vector:

$$\Delta\Phi^{AHD} = \begin{bmatrix} 0 \\ a_{nv}\Delta(\rho\mathbf{v}) \\ 0 \\ a_{nv}\Delta\left\{(g_{ux}, g_{uy}) \cdot \hat{\hat{\mathbf{n}}}_{xy}\right\}\hat{\hat{\mathbf{n}}}_{xy} \\ a_{nv}\Delta\left\{(g_{vx}, g_{vy}) \cdot \hat{\hat{\mathbf{n}}}_{xy}\right\}\hat{\hat{\mathbf{n}}}_{xy} \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \tag{35}$$

which is required to achieve the same order of accuracy in the velocity components and the corresponding gradient variables [5]. Note that $\Delta\Phi^{AHD}$ takes the same form for both HNS20 and HNS20G.

The resulting global system of residual equations are solved by the implicit defect-correction solver as described in the previous paper [2]. Pseudotime terms are included in the residual Jacobian, but not in the gradient equations as described in Ref.[12]. At every nonlinear iteration, a specified number of linear relaxations are performed, which is set to be 15 for all cases in this paper.

5 Results

We implemented the space-time HNS solver based on the node-centered edge-based method in NASA's FUN3D code. Below, we will compare it with the previously developed conventional space-time solver based on the alpha-damping viscous scheme [2]. As in the previous work [2], grid adaptation was performed in a similar manner as described in Ref.[13] with the *refine* package (https://github.com/nasa/refine) developed by Mike Park at NASA Langley Research Center for all problems.

5.1 Accuracy Verification

We first perform an accuracy verification test in a cube domain using the method of manufactured solutions (MMS) in order to verify the implementation. Using a series of consistently-refined irregular tetrahedral grids with n^3 nodes, where n=16, 32, 64, 128, (see Figure 2(a) for the coarsest grid with 16^3 nodes) in a unit cube domain, we solved the system of residual equations and verified the order of discretization error convergence. The exact solution is defined by a smooth exponential function as described in the previous paper [2]. For the flow parameters, we set the freestream Mach number $M_\infty = 0.2$, the freestream Reynolds number $Re_\infty = 1$, and the dimensional freestream temperature 460 Rankine, $P_r = 0.72$.

Error convergence results are shown in Figure 2(b) for the x-velocity u, g_{ux}, which is an extra variable corresponding to its x-derivative (results are similar to other variables), and the derivative computed by a least-squares (LSQ) method applied to u denoted by $LSQ(u_x)$. As can be seen, both u and g_{ux} are asymptotically second-order accurate while $LSQ(u_x)$ is first-order accurate and has larger errors. Also as can be shown in Figures 2(c) and 2(d), contours of g_{ux} is much smoother than those of the LSQ gradient, $LSQ(u_x)$ (obtained on the third grid with 64^3 nodes). These results are encouraging and demonstrate the potential of the HNS20G solver for accurate viscous space-time simulations with adaptive grids.

5.2 Manufactured Unsteady Boundary-Layer Solution

Next, we consider another case with a manufactured solution resembling an oscillating boundary layer in a unit cube domain, using an adaptive tetrahedral grid as in Figure 7(a), taken from the previous paper [2]. As can be seen from the x-velocity contours in Figure 3(b), there is a boundary layer at the wall $y = 0$ with its thickness varying in time. The grid is an anisotropically adapted grid with 69,076 nodes and 378,209 tetrahedra, generated in the previous work [2] with the software *refine*. In this problem, our target quantity is the wall-normal gradient u_y, which is a major contribution to the viscous stress. Accurate prediction of u_y is critical to the success of anisotropic grid adaptation for viscous problems, but is generally considered as a difficult task for fully irregular grids, especially where first-off-the-wall edges are irregularly oriented rather than aligned with a wall-normal direction. See Figure 3(c). Many gradient methods are susceptible to numerical noise on such a grid while the HNS methods are known to produce highly accurate and smooth gradients [4, 5, 6, 12].

To demonstrate the advantages of the HNS solver, we compare the results with those obtained by the conventional solver developed previously based on the alpha-damping viscous scheme [2], which is referred to as Alpha in the results. In both solvers, we set $M_\infty = 0.2$, $Re_\infty = 10^4$, $T_\infty = 460$ [R], CFL$= 10^{15}$, $\kappa = 1/2$, and perform 15 linear relaxations per iteration. The initial values are given by the free stream state, and the solver is taken to be converged when the residual norm reaches 10^{-12} for all equations. Here, the focus is on the quality of the gradient $\partial u/\partial y$, corresponding to the skin friction and the iterative convergence.

Normalized energy equation residual histories are shown in Figure 4(a): the HNS solver converged in 73 iterations while the conventional solver converged in 1,165 iterations to reach the level of 10^{-12} for all the equations. Note that these solvers stopped at different levels because of differences in the initial residual norms used to normalize the data. Here, only the energy residual norm is shown for simplicity. Faster convergence of HNS has been observed in previous studies and are considered as due to the reduced condition number in the linearized system, $O(1/h)$ rather than $O(1/h^2)$, where h denotes a typical mesh spacing, and the first-order accurate Jacobian, not an inconsistent Jacobian as typical in conventional viscous schemes. Figure 4(b) shows the residual convergence in terms of wall time. As can be seen, the HNS solver converged faster than the conventional solver despite the larger cost per iteration. This type of superior iterative convergence has been observed for HNS solvers in Ref.[5].

Figures 5(a), 5(b), and 5(c), and 5(d) show, respectively, the grid at $y = 0$, contours of the gradient $\partial_y u$, which is relevant to the viscous stress force at a boundary. As expected, the contours of the gradient variable g_{uy} obtained with the HNS solver are significantly smoother than those of the LSQ gradient computed from the numerical solution for the conventional solver. Compare Figures5(b) and 5(c). Furthermore, the gradient variable g_{uy} is so accurate that the contours are very close to those of the exact solution shown in Figure 5(d). The superior gradient accuracy can be seen also in Figure 6, where the gradients are compared with the exact gradient u_y along the line $x \in [0, 1]$ at $(y, z) = (0, 0.619)$. The symbols are the data sampled at a finite number of points along the line. Clearly, the gradient variable g_{uy} is smooth and much closer to the exact gradient (solid curve) whereas the LSQ gradient is not as smooth and close to the exact gradient. It is emphasized that the accurate gradient has been obtained faster in the wall time nearly by a factor of four.

Finally, Figure 7 shows the grid and contours of u_y over a section taken at $z = 0.619$. As can be seen, the gradient variable g_{uy} is significantly smoother and more accurate than the LSQ gradient and very close to the exact gradient even away from the wall.

5.3 Viscous Flow over Circular Cylinder at $Re_\infty = 200$

Next, we consider a viscous flow over a circular cylinder of unit diameter at the free stream Mach number $M_\infty = 0.2$ and the free stream Reynolds number per diameter $Re_\infty = 200$. The circular cylinder is centered at the origin, and the outer boundary is a square whose side is 100. This 2D domain is then extended from $z = 0$ to $z = 300$. The grid with 15,720,940 nodes and 91,499,780 tetrahedra was generated after performing 13 grid adaptation cycles using numerical solutions produced by the HNS solver. See Figures 8(a) and 8(b). To demonstrate advantages of the HNS solver, we obtained a result, on the same grid, with the conventional solver developed previously based on the alpha-damping viscous scheme [2], which is referred to as AD in the results. For both AD and HNS, we perform 10 linear relaxations per iteration using 400 processors and begin the nonlinear solver iteration starting from freestream values with smoothing applied at $z = 0$ around the cylinder to reduce the strength of an initial transient wave. The CFL number is ramped from 1 to 10,000 over some initial iterations. CFL is set to be 10^8 for the gradient equations as described in Ref.[12].

Residual convergence results are shown in Figures 13. In both cases, the residuals are reduced at least by two orders of magnitude. It is well known that a stronger solver such as a Jacobian-Free Newton-Krylov (JFNK) solver will help achieve deeper convergence on adaptive tetrahedral grids. The implementation of a JFNK solver in the FUN3D space-time solver is left as important future work.

Nevertheless, numerical solutions show expected flow features as shown in Figure 10. Mach contours show vortex shedding for both AD and HNS solutions. It can be seen that Mach contours are very similar for AD and HNS, and therefore the grid turns out to be well adapted for the conventional scheme as well. In general, such is not guaranteed, and the vortex shedding could start at any other time and the solution could be out of phase.

Figure 11 shows the comparison of vorticity contours and the velocity gradient $\partial_n u$ in the direction normal to the surface, which is relevant to viscous force. As can be seen, HNS results (directly evaluated by the gradient variables) are smoother than those obtained by an LSQ method from the AD solution. Moreover, significantly smoother contours are observed for the density gradients as can be seen in Figure 12.

5.4 Viscous Flow over Circular Cylinder at $Re_\infty = 400$

Finally, we consider again a viscous flow over a circular cylinder centered at the origin but at $Re_\infty = 400$ and in a slightly different domain: 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 the 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 by *refine* after two cycles of grid adaptation. The grid and a section at $y = 0$ are shown in Figures 13(a) and 13(b), respectively. As can be seen, the grid is anisotropically adapted to capture vorticies generated from the cylinder and convected towards the outflow boundary.

For this problem, we used a Jacobian-Free Newton-Krylov solver based on a variable-GCR-preconditioner implemented in a serial code (not FUN3D) [5, 14]. The solver converged with CFL=10^{15} and the residuals have been reduced by eight orders of magnitude for all equations. Figures 14 and 15 show grids and Mach contours are shown for eight sections taken from $z = 10$ to $z = 149$. As can be seen, typical vortex shedding is observed towards the final time $z = 150$, and the grid is adapted for each solution (which is a result of the grid adaptation performed in the three-dimensional space). The adaptive grid was generated by using the solutions obtained with the HNS solver, and then we obtained results with the conventional method with the alpha-damping scheme, which is referred to as AD here, for comparison. As befoew, it can be seen that Mach contours are very similar for AD and HNS, and therefore the grid turns out to be well adapted for the conventional scheme as well.

Figures 16(a), 16(b), and 16(c) show a grid at $z = 149$ and vorticity contours computed by the LSQ method from the numerical solution obtained with the alpha-damping scheme, and those obtained by the gradient variables with the HNS scheme, respectively. Note that the vorticity in the HNS case 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. Improvements are more significant in regions of high anisotropy, for example, near the surface, as can be observed in the zoomed-in plots as given by Figures 16(d), 16(e), and 16(f). In Figure 17, 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.

Figures 18(b) and 18(a) show the residual convergence versus iteration. The solver converged in 44 iterations for the alpha-damping scheme and 34 iterations for the HNS scheme, but there are significant differences in the number of GCR projections and the total number of preconditioner linear relaxations per iteration. As well known, a linearized equation for an HNS discretization has a reduced condition number by a factor of $O(h)$, where h denotes a representative mesh spacing, compared with a conventional viscous discretization. This leads to a speed-up in linear relaxation schemes as we can see from Figures 18(b) and 18(a). Here, we performed each preconditioner relaxation until the linear residual is reduced by one order of magnitude or it reaches the maximum of 50 relaxations for each GCR projection. Moreover, we can see that the solver required more GCR projections with the alpha-damping scheme, often performing the maximum of 25 GCR projections (GCR tolerance is set to be 0.5, requiring to reduce the GCR linear residual by half) with 50 preconditioner relaxations per projection (total of 1,250 relaxations). As a result, the solver converged faster in CPU time for HNS, nearly twice as fast, as can be seen in Figures 18(d) and 18(c). Therefore, the HNS solver can produce significantly more accurate and smoother gradients on adaptive grids faster than a conventional solver.

6 Concluding Remarks

We have developed a space-time hyperbolic Navier-Stokes solver and demonstrated its capabilities with adaptive tetrahedral grids for viscous problems: accurate and smooth gradients on irregular adaptive tetrahedral grids, and superior iterative convergence by an implicit defect-correction solver.

Future work includes the development of more accurate hyperbolic Navier-Stokes methods. For example, we may add time derivatives as extra unknown variables and construct Scheme-IQ, which achieves third-order accuracy in the inviscid terms without introducing quadratic LSQ methods. Also, we will consider a third-order hyperbolic Navier-Stokes scheme with a quadratic implicit gradient method, in particular, because implicit gradient methods are known to stabilize implicit solvers. Finally, for a practical reason, we plan to develop a time-slab approach, where a space-time domain is split into smaller pieces in the time direction, in order to be able to perform space-time computations more efficiently with limited computer resources available.

Acknowledgments

This work has been funded by AFOSR under the grant number FA9550-20-0023. The second author gratefully acknowledges support 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 snd H. Nishikawa. An adaptive space-time edge-based solver for two-dimensional unsteady inviscid flows. In *SciTech 2022 Forum*, AIAA Paper 2022-2201, San Diego, CA, 2022.

[3] 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.

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

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

[6] 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.

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

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

[9] H. Nishikawa. First-, second-, and third-order finite-volume schemes for diffusion. *J. Comput. Phys.*, 256:791–805, 2014.

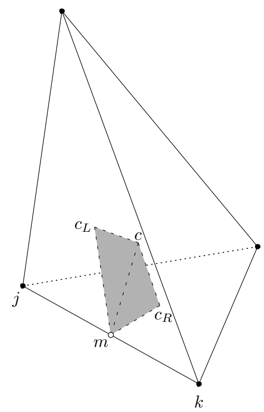
[10] B. van Leer. Upwind-difference methods for aerodynamic problems governed by the Euler equations. In *Fluid Mechanics, Lectures in Applied Mathematics*, volume 22, pages 327–335, 1985.

[11] C. O. E. Burg. Higher order variable extrapolation forunstructured finite volume RANS flow solvers. In *17th AIAA Computational Fluid Dynaamics Conference*, AIAA Paper 2005-4999, Orlando, FL, 2005.

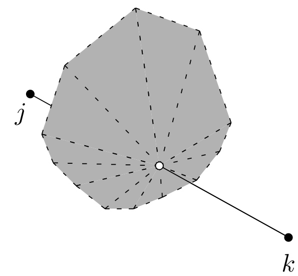
[12] Y. Nakashima, N. Watanabe, and H. Nishikawa. Hyperbolic Navier-Stokes solver for three-dimensional flows. In *54th AIAA Aerospace Sciences Meeting*, AIAA Paper 2016-1101, San Diego, CA, 2016.

[13] W. L. Kleb, M. A. Park, W. A. Wood, K. L. Bibb, K. B. Thompson, and R. J. Gomez. Sketch-to-solution: An exploration of viscous CFD with automatic grids. In *24th AIAA Computational Fluid Dynamics Conference*, AIAA Paper 2019-2948, Dallas, TX, 2019.

[14] 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.

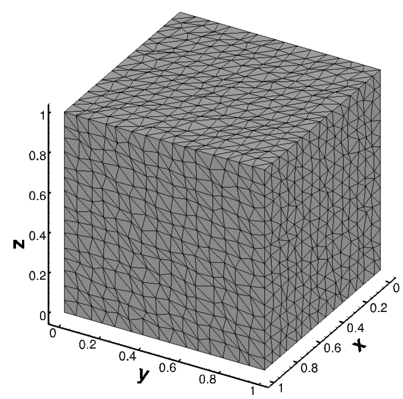


(a) Dual face contributions from a tetrahedron.

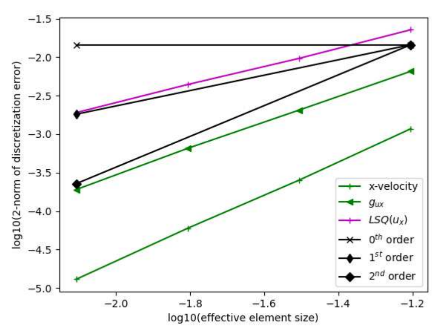


(b) Total dual faces.

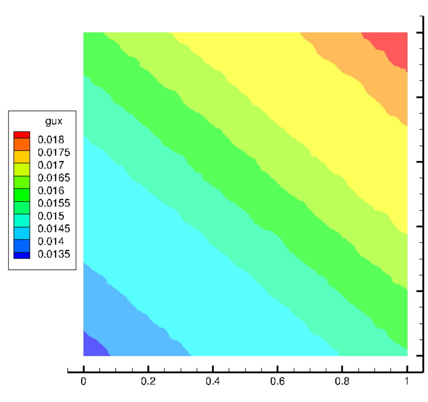
Figure 1: Dual face contributions at the edge $[j, k]$. A numerical flux is evaluated at the midpoint of the edge $[j, k]$. In (a), the centroid of the tetrahedral element is denoted by c, and the centroids of the two adjacent triangles are denoted by c_L and c_R.



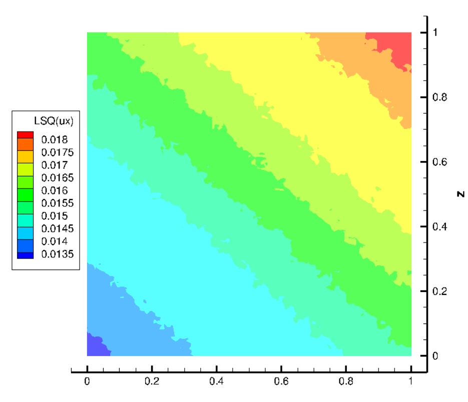
(a) Coarsest grid.



(b) Error convergence.

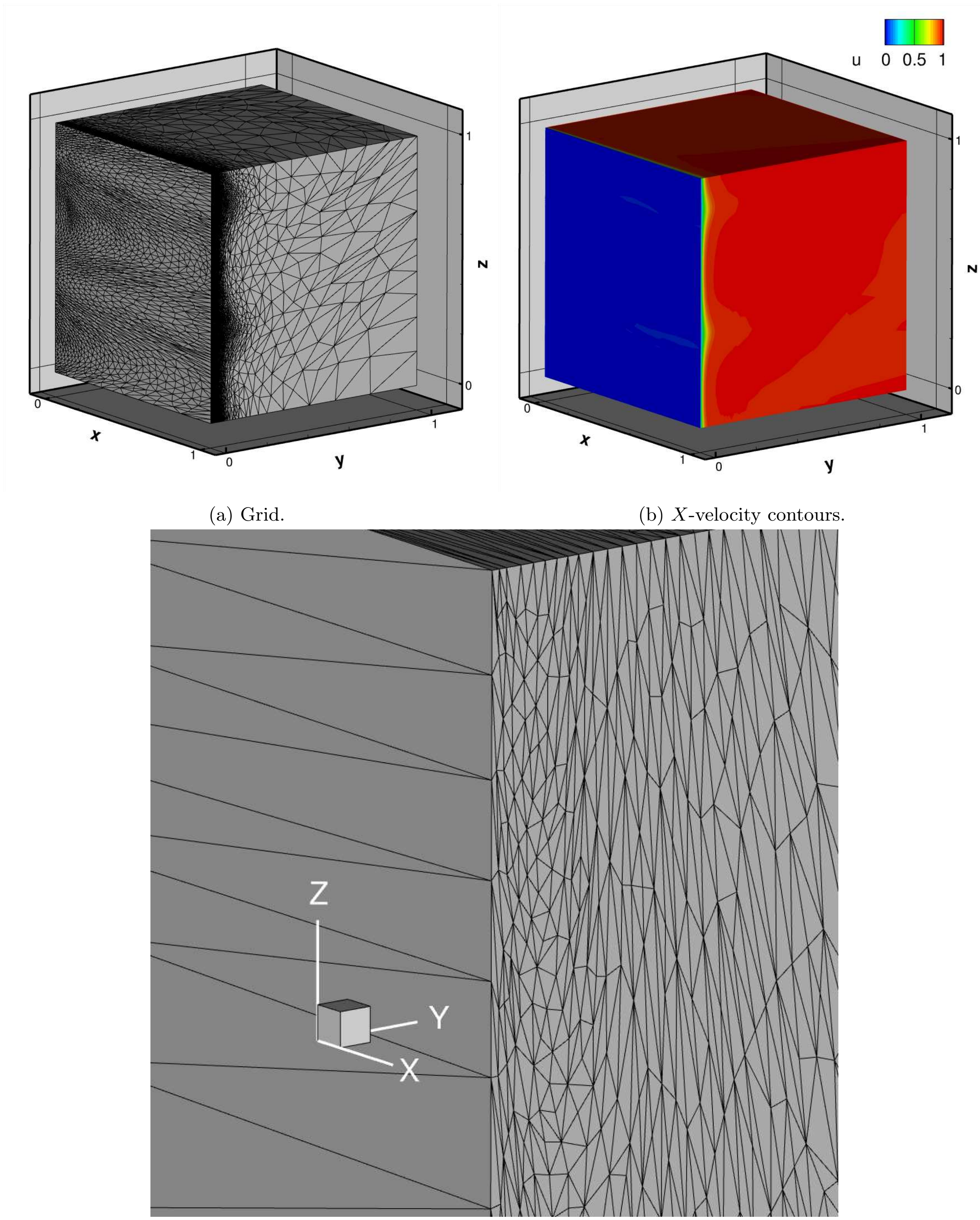


(c) $g_{ux}(= u_x)$ on a 32^3-node grid.



(d) $LSQ(u_x)$ on a 32^3-node grid.

Figure 2: Accuracy verification results. Contours plots are sampled over the section at $x = 0.5$.

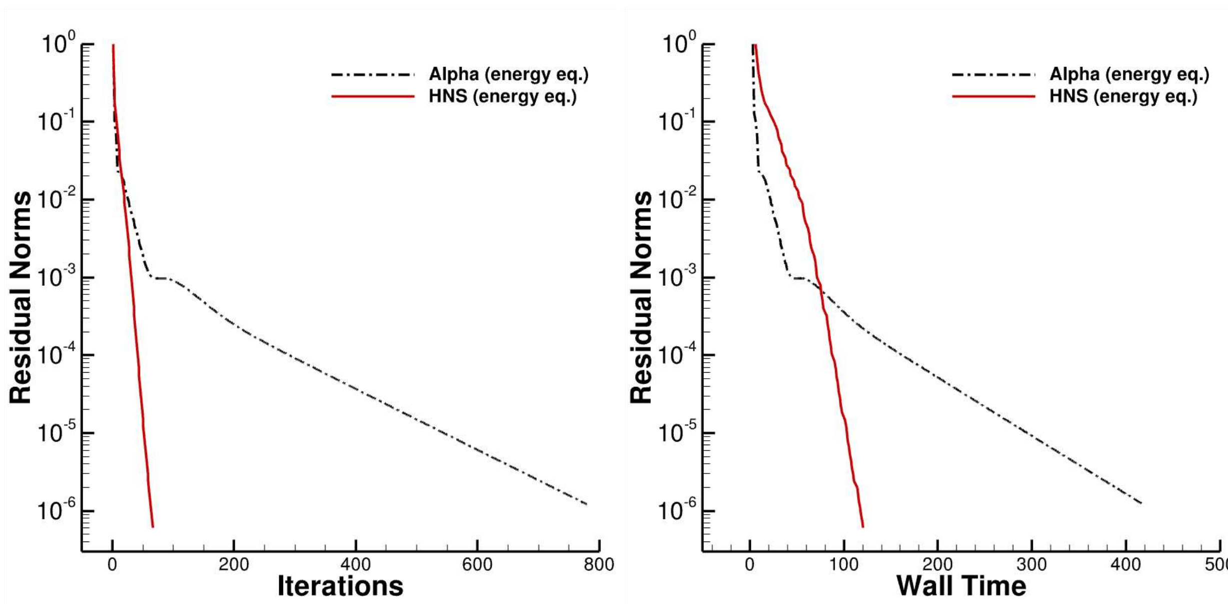


(a) Grid.

(b) X-velocity contours.

(c) Zoomed-in view around the corner $(x, y, z) = (1, 0, 1)$

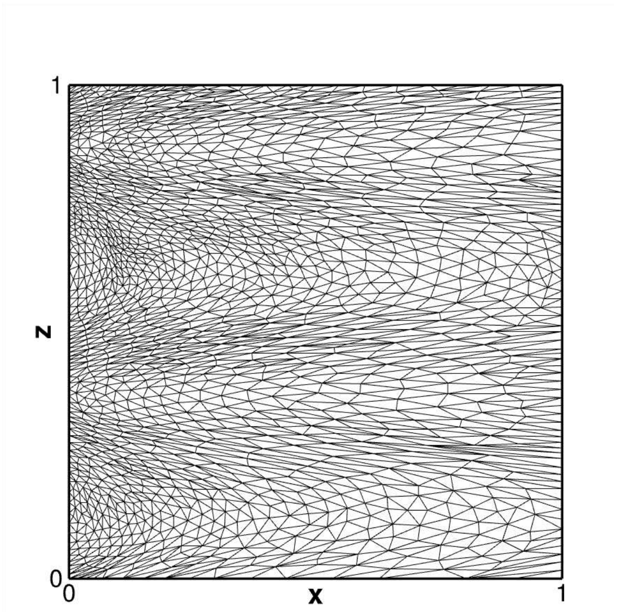
Figure 3: Grid and solution for MMS boundary layer test case.



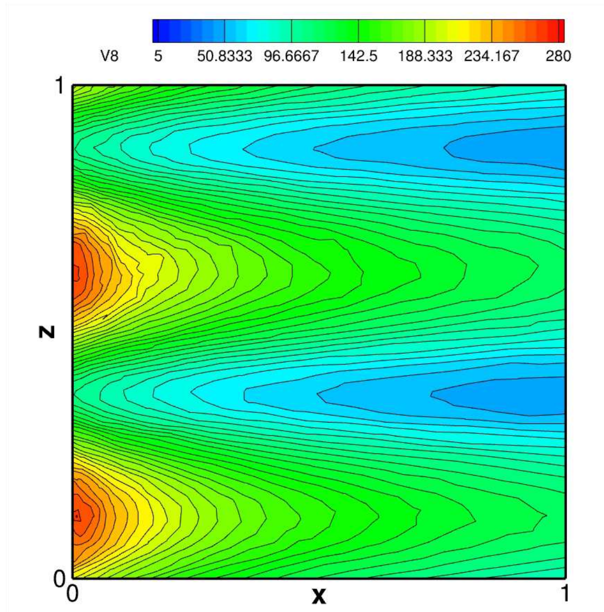
(a) Residual versus iteration.

(b) Residual versus wall time.

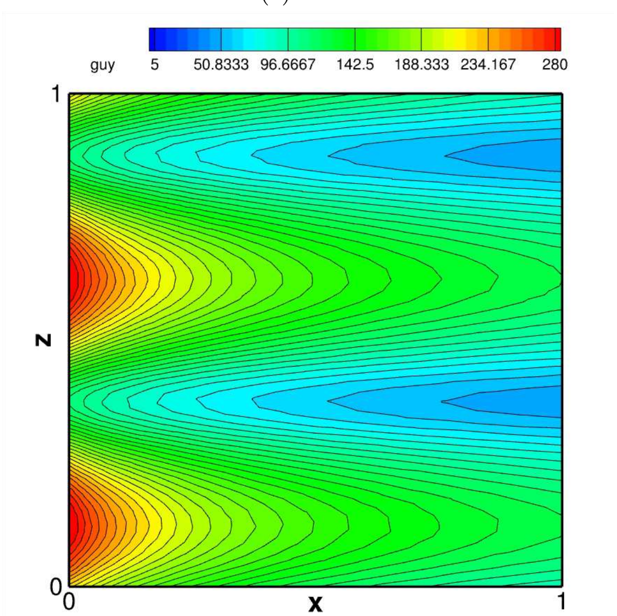
Figure 4: Residual convergence results for MMS boundary layer test case. Only the energy equation residual norms are shown.



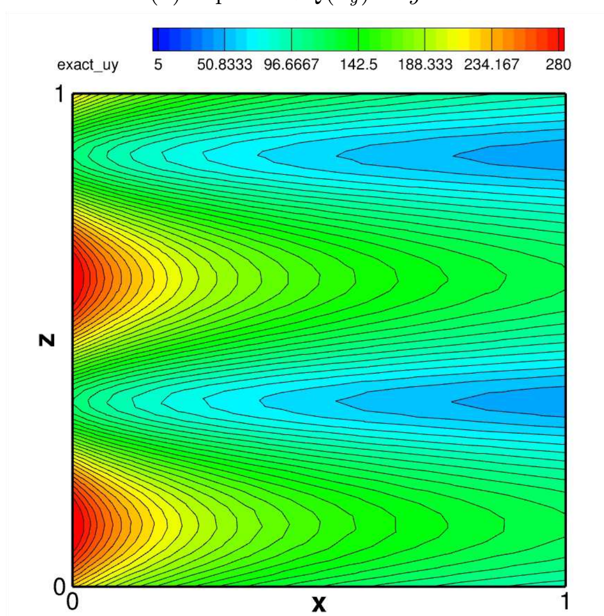
(a) Grid.



(b) Alpha: $LSQ(u_y)$ at $y = 0$.



(c) HNS: g_{uy} at $y = 0$.



(d) Exact: u_y at $y = 0$.

Figure 5: Wall normal gradient contours over the boundary at $y = 0$ for MMS boundary layer test case.

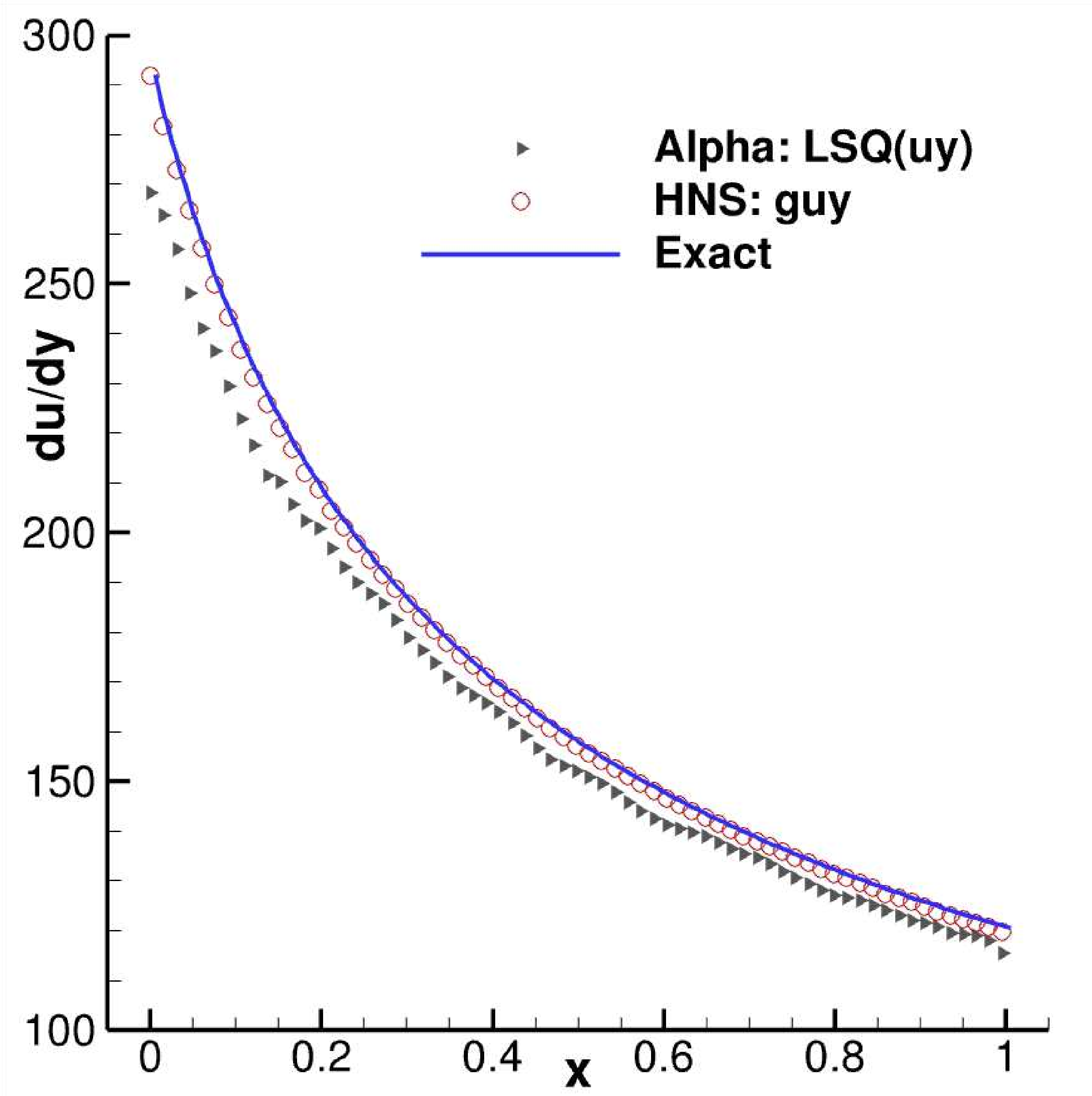
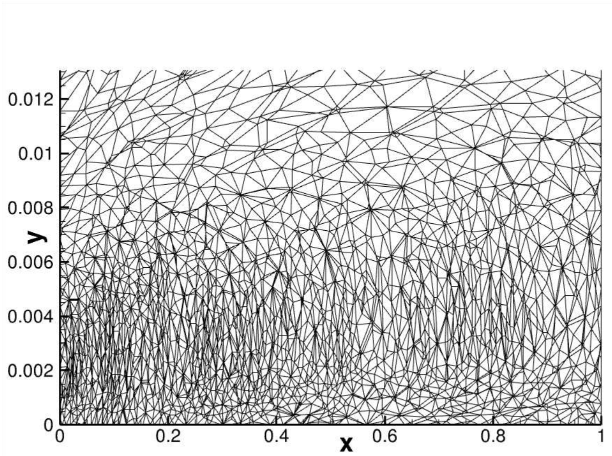
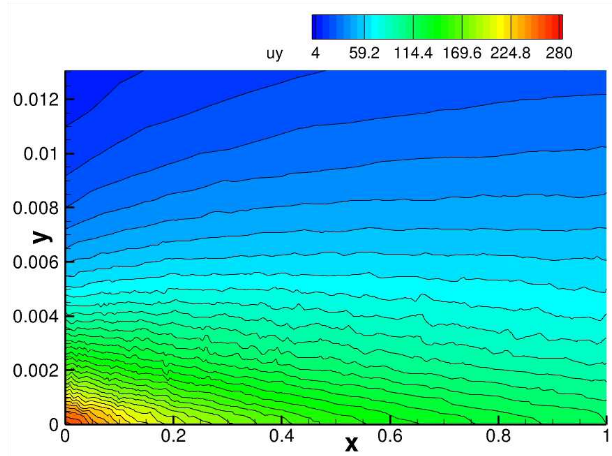


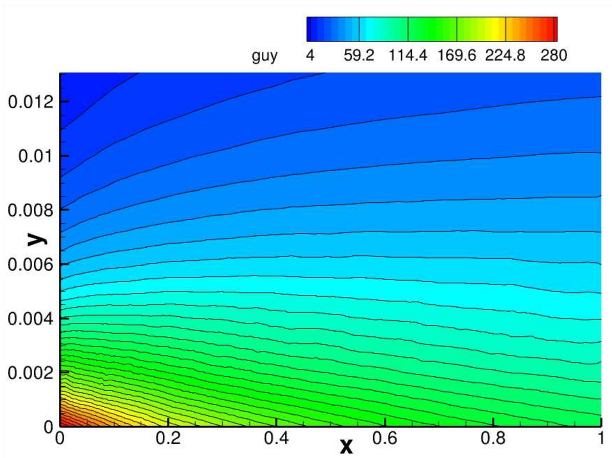
Figure 6: Section plot at $(y, z) = (0, 0.619)$ for MMS boundary layer test case.



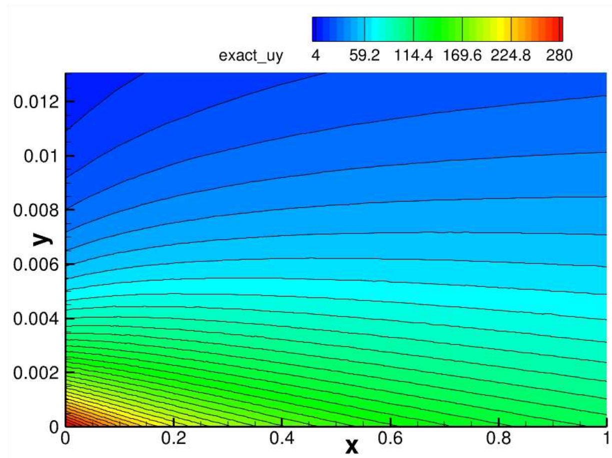
(a) Grid.



(b) Alpha: $LSQ(u_y)$ at $z = 0.619$.

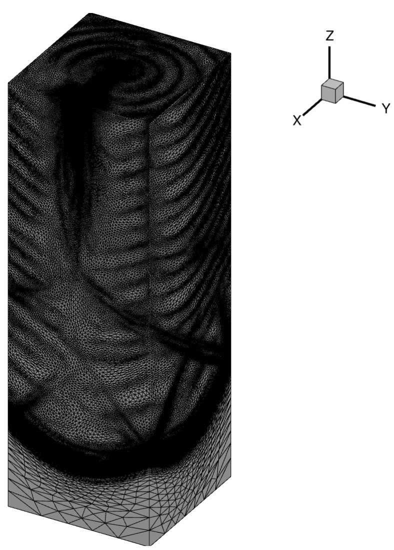


(c) HNS: g_{uy} at $z = 0.619$.

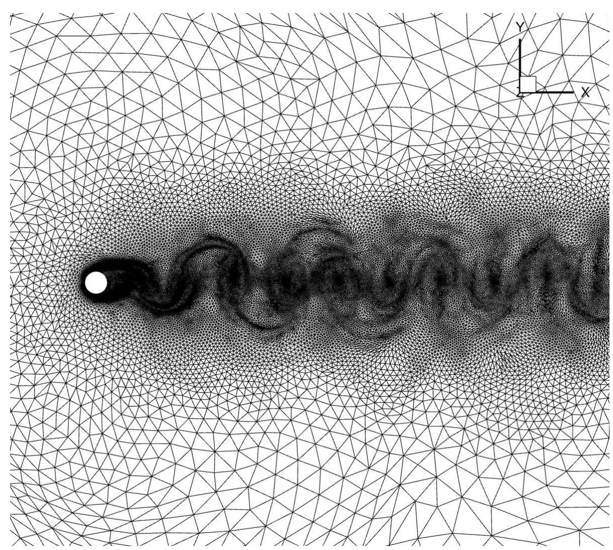


(d) Exact: u_y at $z = 0.619$.

Figure 7: Normal gradient contours at $z = 0.619$ for MMS boundary layer test case.

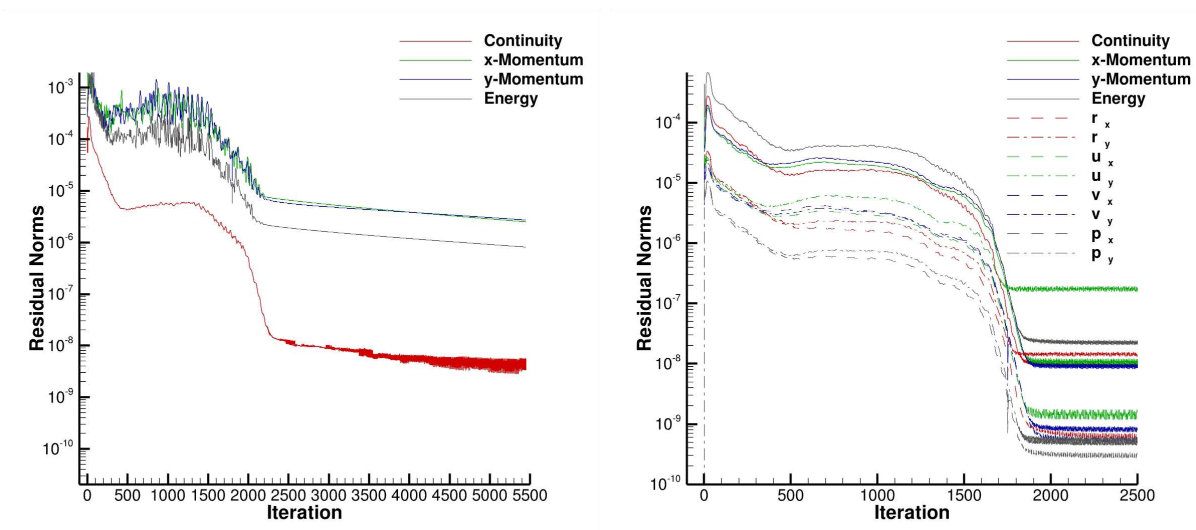


(a) Grid.



(b) Top view ($z = 300$).

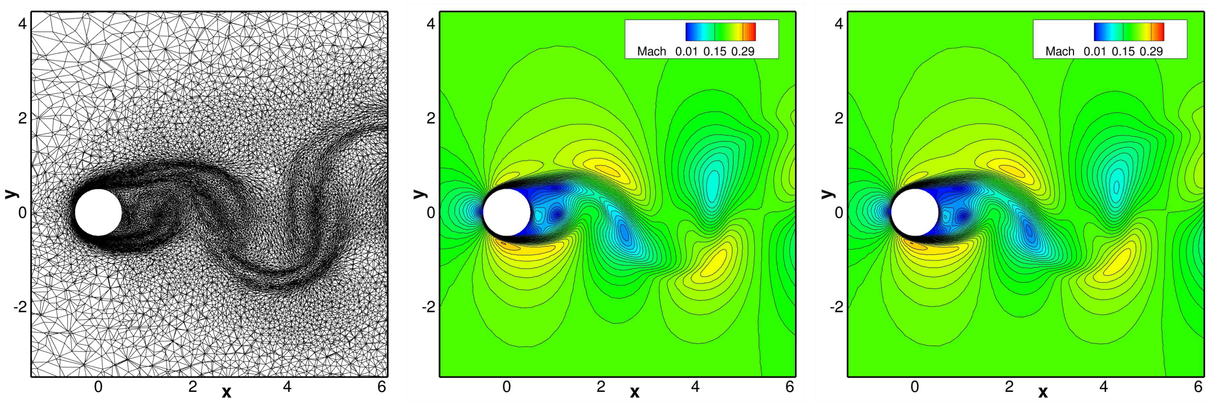
Figure 8: Grid for the cylinder test case.



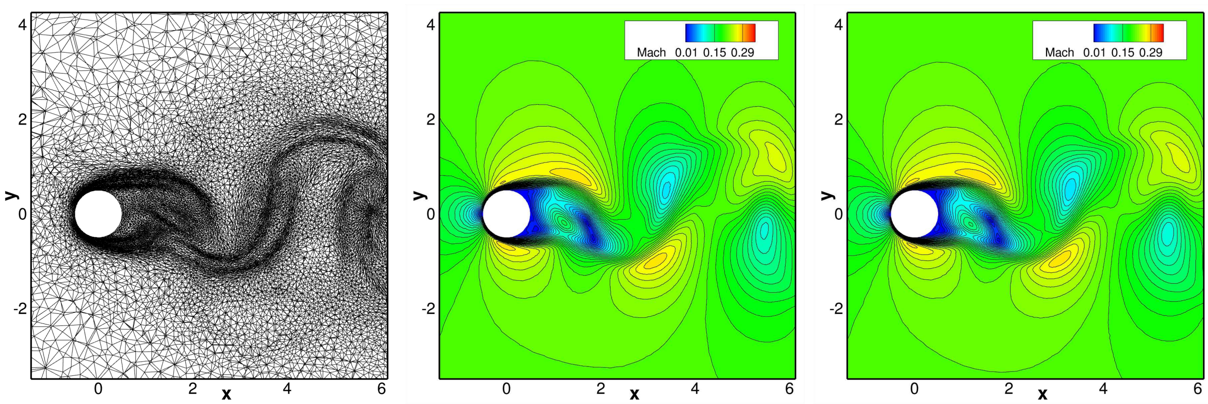
(a) Residual versus iteration (AD).

(b) Residual versus iteration (HNS).

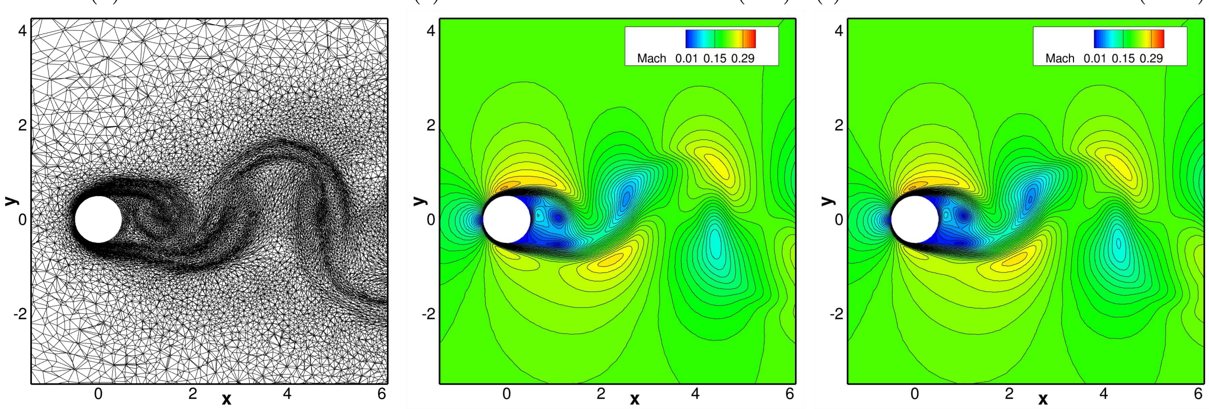
Figure 9: Residual convergence for the cylinder test case.



(a) Grid at $z = 200$. (b) Mach contours at $z = 200$ (AD). (c) Mach contours at $z = 200$ (HNS).

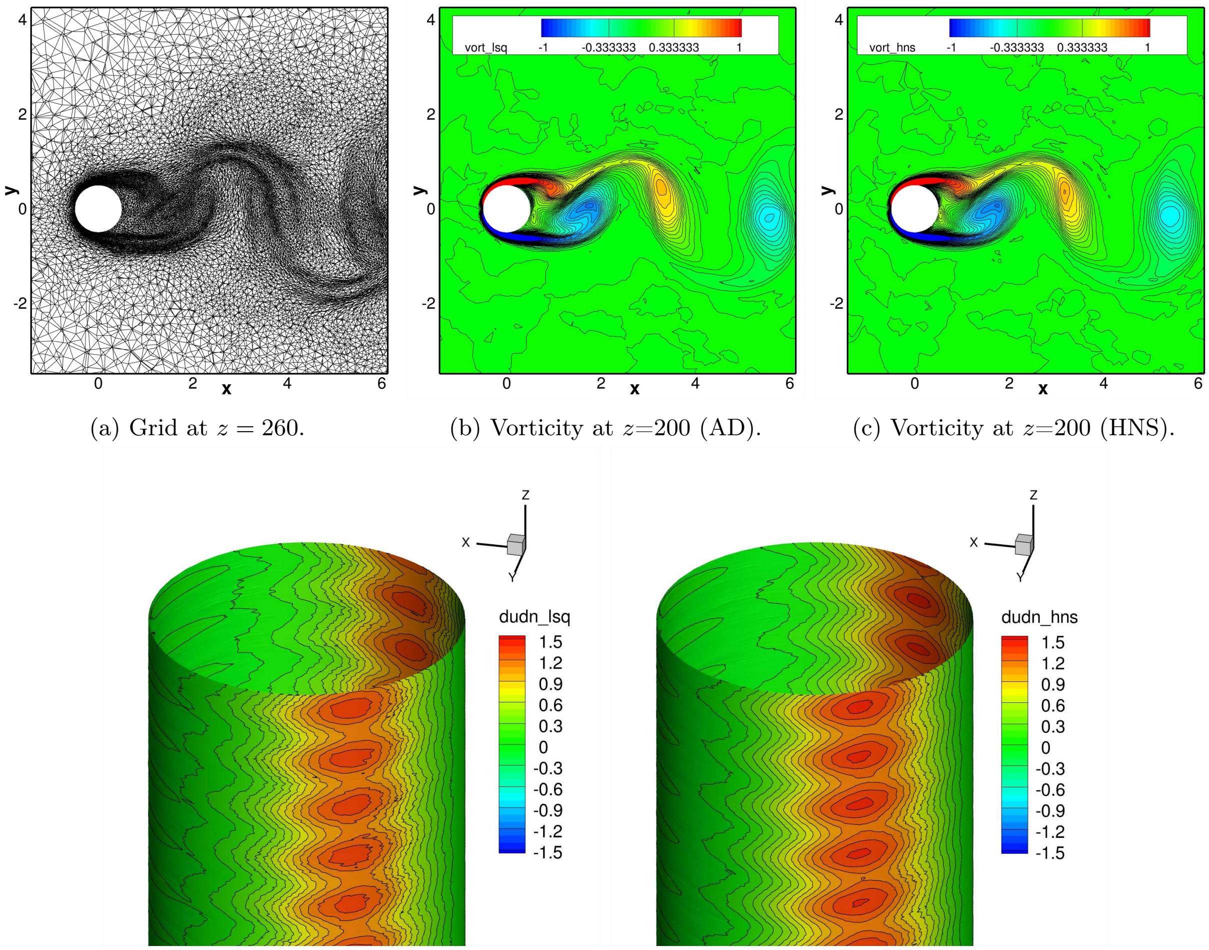


(d) Grid at $z = 220$. (e) Mach contours at $z = 220$ (AD). (f) Mach contours at $z = 220$ (HNS).



(g) Grid at $z = 240$. (h) Mach contours at $z = 240$ (AD). (i) Mach contours at $z = 240$ (HNS).

Figure 10: Section plots for $z = 200$, 220, 240 in the cylinder test case.



(a) Grid at $z = 260$.

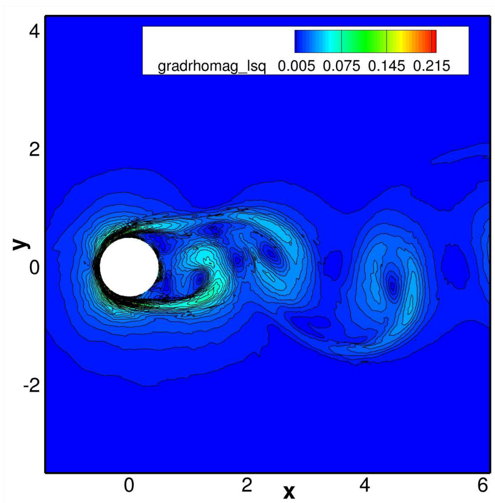
(b) Vorticity at z=200 (AD).

(c) Vorticity at z=200 (HNS).

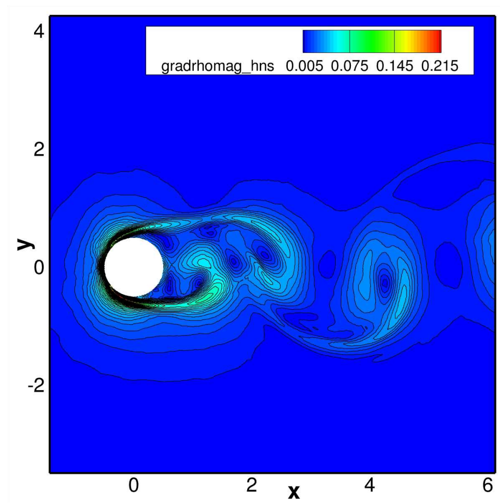
(d) $\partial_n u$ (AD).

(e) $\partial_n u$ (HNS).

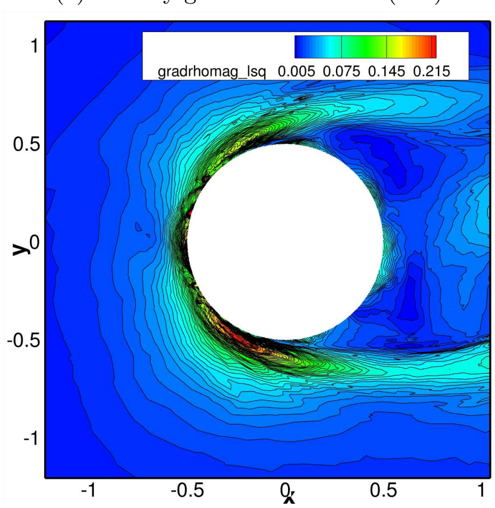
Figure 11: Vorticity contours at $z = 200$ and over the surface in the cylinder test case.



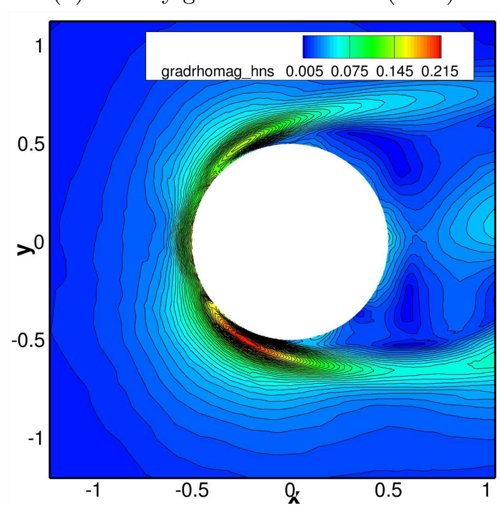
(a) Density gradient at z=200 (AD).



(b) Density gradient at z=200 (HNS).

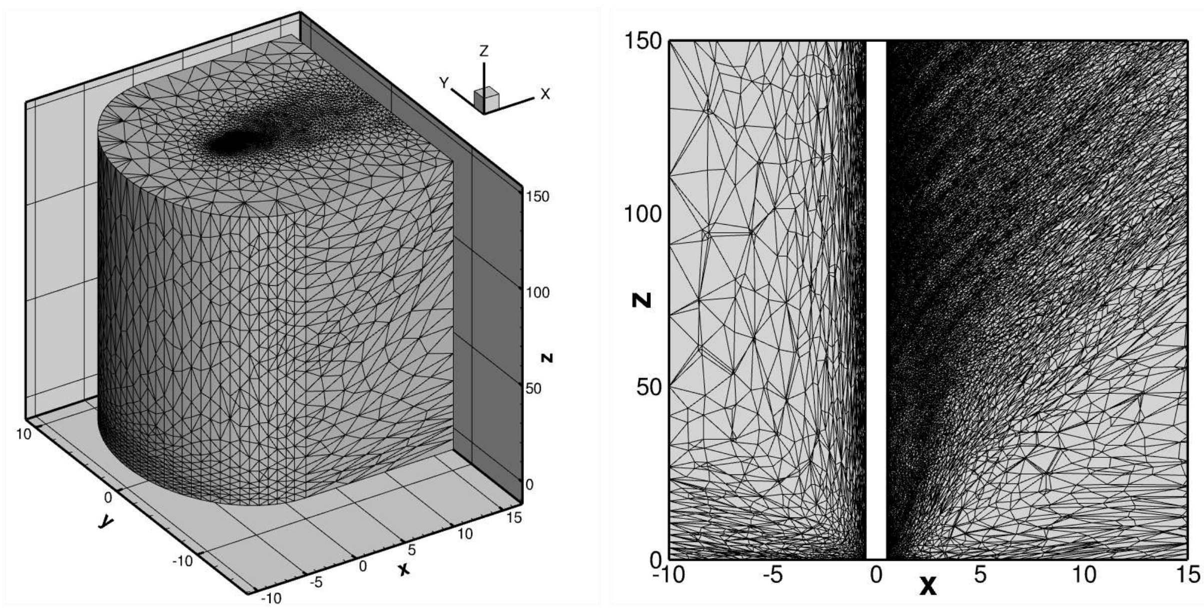


(c) Density gradient at z=200 (AD).



(d) Density gradient at z=200 (HNS).

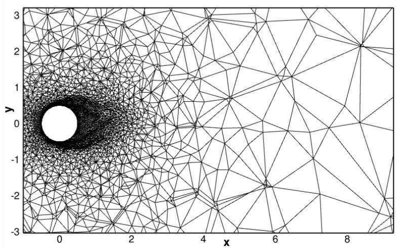
Figure 12: Density gradient contours for $z = 200$ in the cylinder test case.



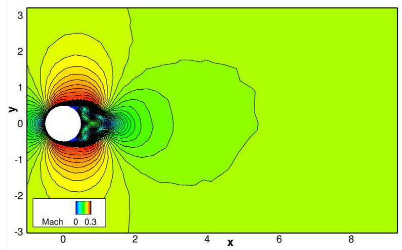
(a) Grid.

(b) Section at $y = 0$.

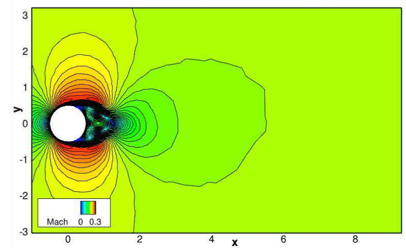
Figure 13: Grid for the second cylinder test case.



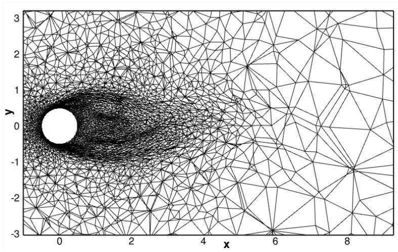
(a) Grid at $z = 10$.



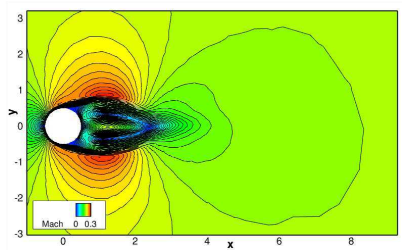
(b) Mach contours at $z = 10$ (AD).



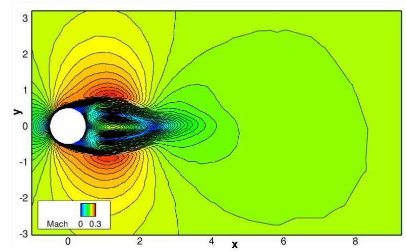
(c) Mach contours at $z = 10$ (HNS).



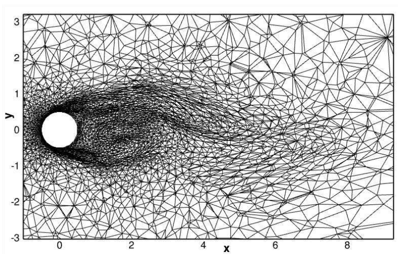
(d) Grid at $z = 30$.



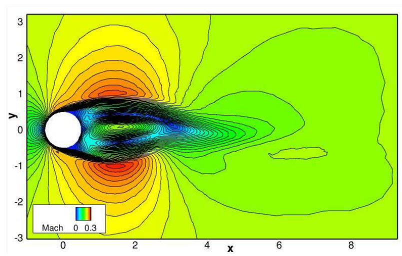
(e) Mach contours at $z = 30$ (AD).



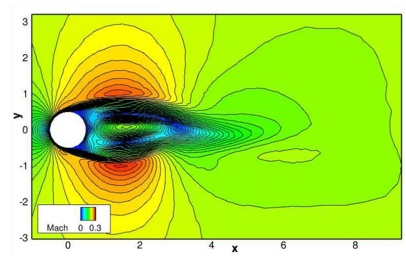
(f) Mach contours at $z = 30$ (HNS).



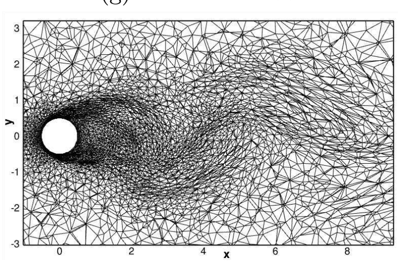
(g) Grid at $z = 50$.



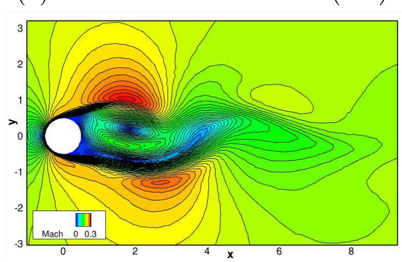
(h) Mach contours at $z = 50$ (AD).



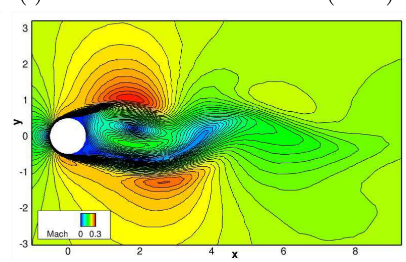
(i) Mach contours at $z = 50$ (HNS).



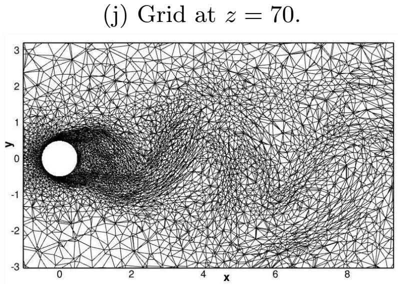
(j) Grid at $z = 70$.



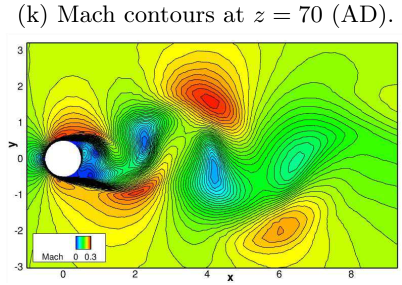
(k) Mach contours at $z = 70$ (AD).



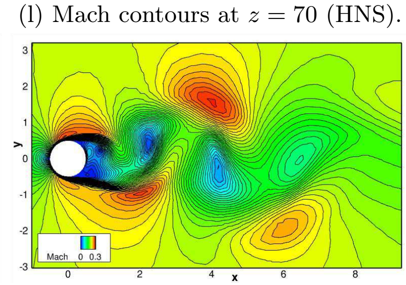
(l) Mach contours at $z = 70$ (HNS).



(m) Grid at $z = 90$.

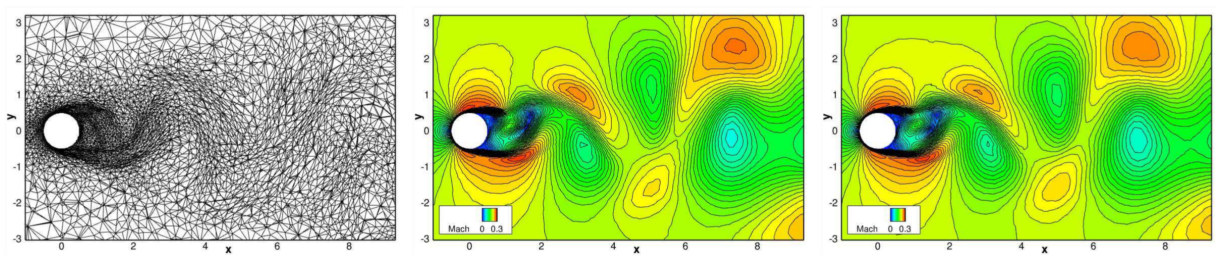


(n) Mach contours at $z = 90$ (AD).

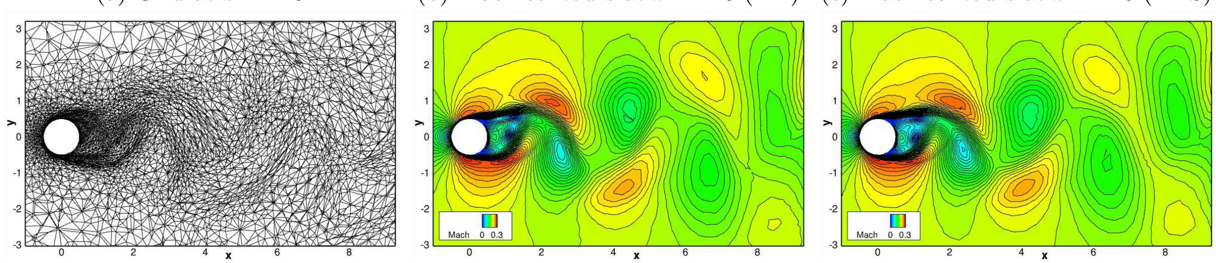


(o) Mach contours at $z = 90$ (HNS).

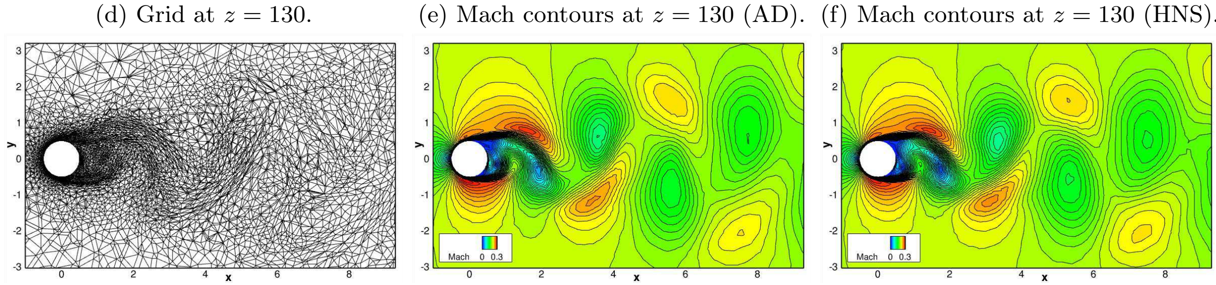
Figure 14: Section plots for $z = 10, 30, 50, 70, 90$ in the second cylinder test case.



(a) Grid at $z = 110$. (b) Mach contours at $z = 110$ (AD). (c) Mach contours at $z = 110$ (HNS).

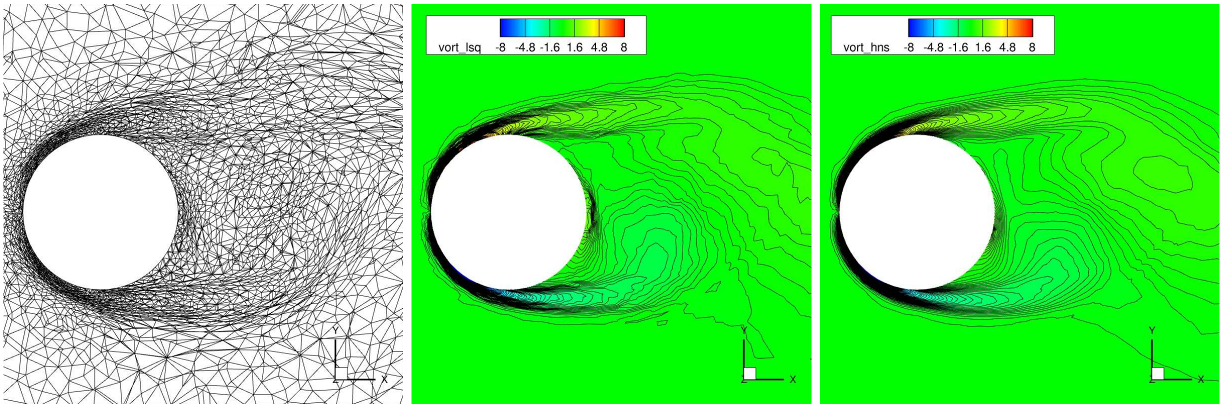


(d) Grid at $z = 130$. (e) Mach contours at $z = 130$ (AD). (f) Mach contours at $z = 130$ (HNS).

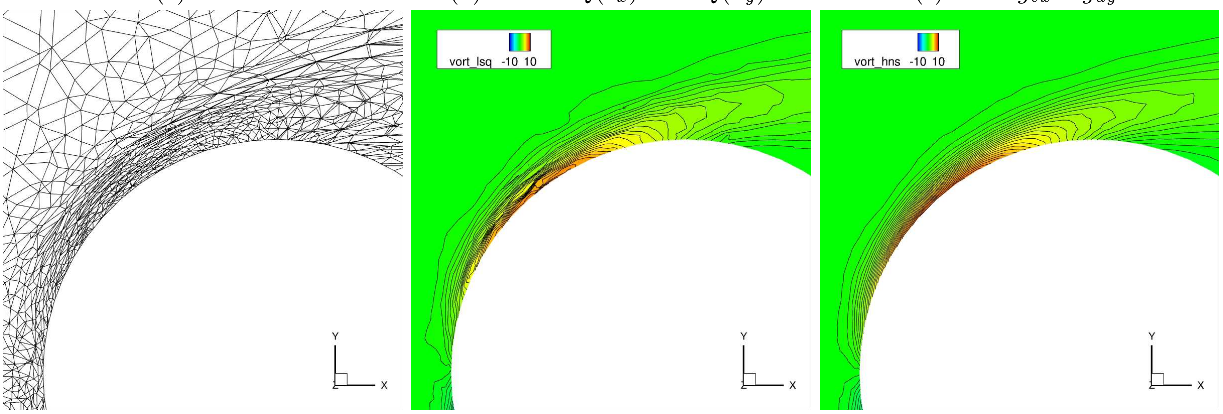


(g) Grid at $z = 149$. (h) Mach contours at $z = 149$ (AD). (i) Mach contours at $z = 149$ (HNS).

Figure 15: Section plots for $z = 90$, 110, 130, 149 in the second cylinder test case.

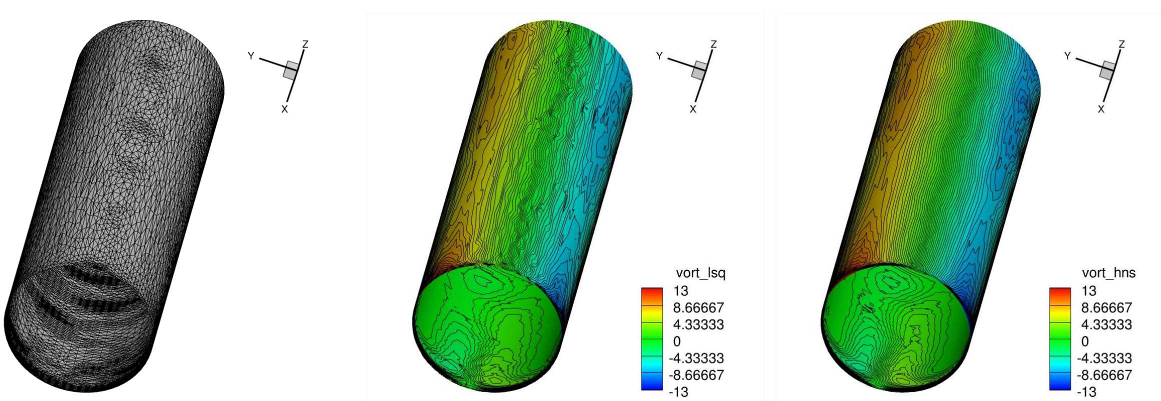


(a) Grid. (b) AD: $LSQ(v_x) - LSQ(u_y)$. (c) HNS: $g_{vx} - g_{uy}$.



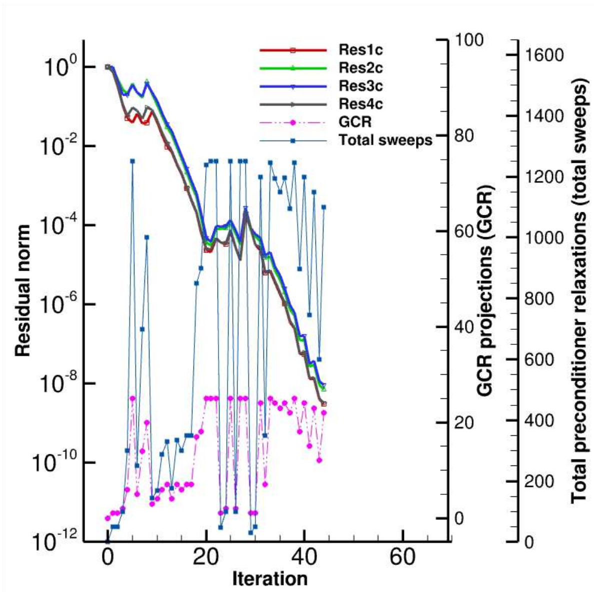
(d) Grid. (e) AD: $LSQ(v_x) - LSQ(u_y)$. (f) HNS: $g_{vx} - g_{uy}$.

Figure 16: Grids and vorticity contours at $z = 149$ for the second cylinder test case.

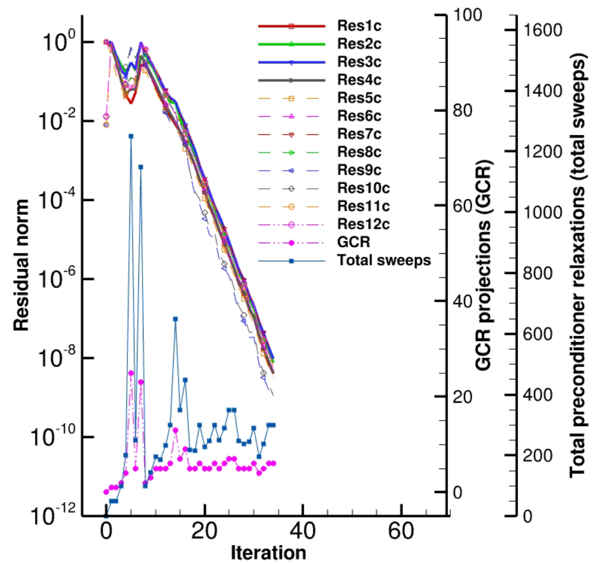


(a) Grid. (b) AD: $LSQ(v_x) - LSQ(u_y)$. (c) HNS: $g_{vx} - g_{uy}$.

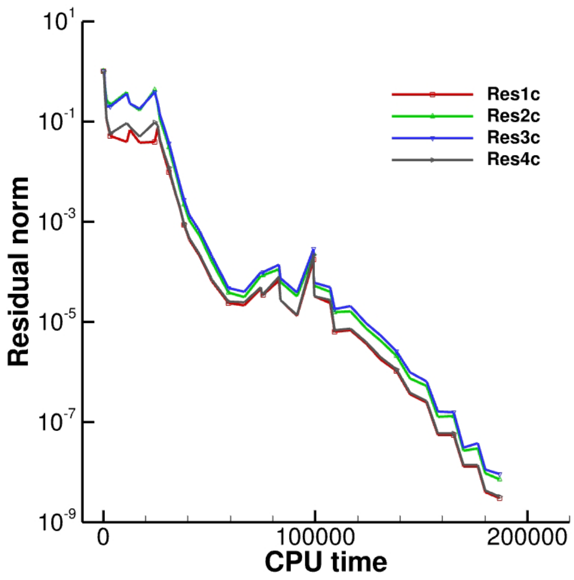
Figure 17: Grid and vorticity contours at the circular cylinder for the second cylinder test case.



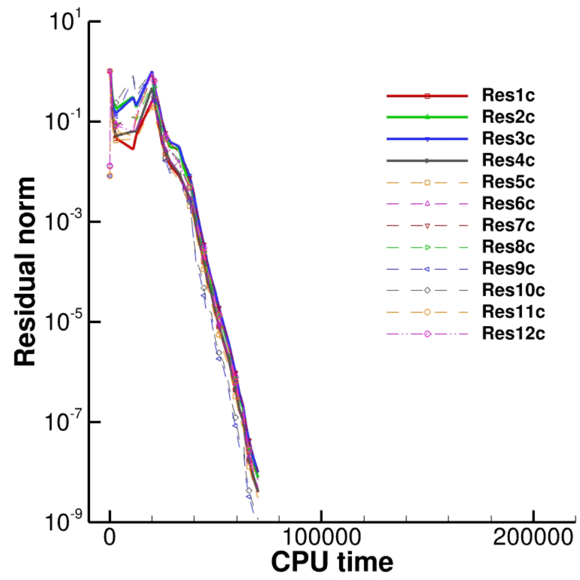
(a) Residual versus iteration (AD).



(b) Residual versus iteration (HNS).



(c) Residual versus CPU time (AD).



(d) Residual versus CPU time (HNS).

Figure 18: Residual convergence results for the second cylinder test case.