10.1115/OMAE2015-41012

Proceedings of the ASME 2015 34th International Conference on Ocean, Offshore and Arctic Engineering
OMAE2015
May 31-June 5, 2015, St. John's, Newfoundland, Canada

OMAE2015-41012

A SIMPLE, EFFICIENT, AND HIGH-ORDER ACCURATE SLIDING-MESH INTERFACE APPROACH TO SPECTRAL DIFFERENCE METHOD ON COUPLED ROTATING AND DEFORMING/STATIONARY DOMAINS

Bin Zhang, Chunlei Liang, Jingjing Yang
Computational Aero and Hydrodynamics Laboratory
Department of Mechanical and Aerospace Engineering
The George Washington University
Washington, DC 20052

ABSTRACT

In this paper, we present a novel sliding-mesh interface approach to the spectral difference (SD) method on coupled rotating and deforming/stationary domains. This approach is an extension of the previously reported method for coupled rotating and stationary domains for the flux reconstruction (FR) method [1]. The use of sliding-mesh interfaces not only allows us to study flows around freely rotating bodies but can also dramatically reduce grid twists on deforming domains with large-angle rotations. We test the accuracy of the solver on an inviscid flow, and it is found that the solver is high-order accurate on coupled dynamic grids with sliding-mesh interface. Simulation of viscous flow over a plunging and pitching airfoil is also carried out to verify the solver, it is seen that the solver is accurate and is very efficient in terms of computational cost. Finally, we apply the solver to study a two-dimensional vertical axis wind turbine at different Reynolds numbers. It is found that the turbine is efficient and extracts energy from the flows at high Reynolds numbers. This solver can also be applied to other problems, such as the aerodynamics of rotorcrafts, oscillating wing wind power generators.

INTRODUCTION

During the last two decades, high-order (third and above) numerical methods have received wider range of interests in research communities, especially in the computational fluid dynamics (CFD) community. High-order methods are capable of producing more accurate solutions on relatively coarse grids at lower computational cost than low-order methods [2]. High-order methods are also being applied to deal with very challenging fluid flow problems with complex geometries. Summaries of recent development and applications of high-order methods can be found in several books [3,4,5,6] and review papers [2,7,8].

Among the numerous high-order methods, the discontinuous Galerkin (DG) method gained enormous popularity in the past decade for solving conservation laws on unstructured grid. The idea of the DG method was first introduced by Reed and Hill [9] to solve neutron transport equations. Cockburn, Shu, Bassi, Rebay and others [6, 10, 11, 12, 13] developed the DG method extensively and applied it to fluid dynamics problems. However, the computational complexity of the DG method increases rapidly as the order of accuracy increases [14].

Instead of solving the equations in integral form as in the DG method, Kopriva [15] proposed a staggered-grid Chebyshev multidomain method, which solves the differential form of the conservation laws. Liu et al. [16] and Wang et al. [17] extended this method to unstructured triangular and quadrilateral grids and named the more general method as spectral difference (SD) method. For SD method on quadrilateral grids, solutions and fluxes in each coordinate direction are represented by $1D$ Lagrange polynomials, whereas the multi-dimensional solutions within each cell are reconstructed by tensor products of the Lagrange polynomials. The $1D$ representation reduces the computational cost dramatically, and the tensor-product reconstruction makes the SD method high-order accurate.

We have seen more and more applications of the SD method to realistic flow simulations. For example, for large eddy simulation on fixed grids [18, 19, 20, 21]. The SD method is also particularly suitable for simulating vortex-dominated flows on moving and deforming grids [22, 23]. Liang, et al. [24] extended the SD method for simulating 2D unsteady flow around plunging or pitching airfoils. However, when the grid undergoes very large rotational motion, the deforming-mesh approach fails since the grid is severely twisted resulting in cells with negative volumes.

In this paper, we present a newly developed high-order accurate flow solver for coupled rotating and deforming/stationary domains using spectral difference (SD) method and a novel sliding-mesh interface approach. This sliding-mesh interface approach is an extension of the previously reported method for coupled rotating and stationary domains for the flux reconstruction (FR) method [1]. The use of sliding-mesh interfaces allows us to study flows around freely rotating bodies. Furthermore, it can dramatically reduce twists on the grids for deforming domain with large-angle rotations.

This paper is organized as follows. We first give the governing equations and the transformed equations in the computational space. Subsequently, a brief review of the SD method and a description of the sliding-mesh interface approach are presented. After that, accuracy and verification tests of the solver are carried out, and a study of a vertical axis wind turbine is performed. Finally, we conclude the paper in the last section.

GOVERNING EQUATIONS

In this section, we give the governing equations in both the physical space and the computational space.

Compressible Navier-Stokes Equations

We consider the 2D unsteady compressible Navier-Stokes equations in conservative form,

$$\frac{\partial \mathbf{Q}}{\partial t} + \frac{\partial \mathbf{F}}{\partial x} + \frac{\partial \mathbf{G}}{\partial y} = 0, \quad (1)$$

where $\mathbf{Q}$ is the vector of conservative variables, and $\mathbf{F}$ and $\mathbf{G}$ are the x and y fluxes with the following expressions,

$$\mathbf{Q} = [\rho \; \rho u \; \rho v \; E]^T, \quad (2)$$

$$\mathbf{F} = \mathbf{F}_{inv}(Q) + \mathbf{F}_{vis}(Q, \nabla Q), \quad (3)$$

$$\mathbf{G} = \mathbf{G}_{inv}(Q) + \mathbf{G}_{vis}(Q, \nabla Q), \quad (4)$$

where ρ is fluid density, u and v are x and y velocities, E is the total energy per volume defined as $E = p/(\gamma - 1) + \frac{1}{2}\rho(u^2 + v^2)$, p is pressure, γ is the ratio of specific heats and is set to 1.4.

As shown in Equations (3) and (4), the fluxes have been divided into inviscid and viscous parts. The inviscid fluxes are only functions of conservative variables,

$$\mathbf{F}_{inv} = \begin{bmatrix} \rho u \\ \rho u^2 + p \\ \rho uv \\ (E+p)u \end{bmatrix}, \; \mathbf{G}_{inv} = \begin{bmatrix} \rho v \\ \rho uv \\ \rho v^2 + p \\ (E+p)v \end{bmatrix}. \quad (5)$$

The viscous fluxes are functions of the conservative variables and their gradients. They have the following expressions,

$$\mathbf{F}_{vis} = -\begin{bmatrix} 0 \\ \tau_{xx} \\ \tau_{yx} \\ u\tau_{xx} + v\tau_{yx} + \mathrm{k}T_x \end{bmatrix}, \quad (6)$$

$$\mathbf{G}_{vis} = -\begin{bmatrix} 0 \\ \tau_{xy} \\ \tau_{yy} \\ u\tau_{xy} + v\tau_{yy} + \mathrm{k}T_y \end{bmatrix}, \quad (7)$$

where τ_{ij} is the shear stress tensor which is related to velocity gradients as $\tau_{ij} = \mu(u_{i,j} + u_{j,i}) + \lambda \delta_{ij} u_{k,k}$, and μ is the dynamic viscosity, $\lambda = -2/3\mu$ based on Stokes' hypothesis, δ_{ij} is the Kronecker delta, k is thermal conductivity, T is temperature that is related to density and pressure through the ideal gas law $p = \rho R T$, where R is the gas constant.

The Transformed Equations

As will be discussed in the next section, we map each quadrilateral cell from the physical domain to a standard square element in the computational domain. This mapping facilitates the construction of solution and flux polynomials. As a result, we only need to solve a set of transformed equations within each standard element. Let us assume that the physical time and coordinates (t, x, y) are mapped to the computational ones (τ, ξ, η) through a transformation: $t = \tau$, $x = x(\tau, \xi, \eta)$, $y = y(\tau, \xi, \eta)$. It can be shown that Equation (1) will take the following conservative form after coordinates transformation,

$$\frac{\partial \widetilde{\mathbf{Q}}}{\partial \tau} + \frac{\partial \widetilde{\mathbf{F}}}{\partial \xi} + \frac{\partial \widetilde{\mathbf{G}}}{\partial \eta} = 0. \quad (8)$$

The computational variable and fluxes are related to the physical ones as

$$\begin{pmatrix} \widetilde{\mathbf{Q}} \\ \widetilde{\mathbf{F}} \\ \widetilde{\mathbf{G}} \end{pmatrix} = |\mathscr{J}| \mathscr{J}^{-1} \begin{pmatrix} \mathbf{Q} \\ \mathbf{F} \\ \mathbf{G} \end{pmatrix}, \quad (9)$$

where $|\mathscr{J}|$ is the determinant of the transformation Jacobian matrix, and $\mathscr{J}^{-1}$ is the inverse transformation Jacobian matrix. They have the following definitions,

$$\mathscr{J} = \frac{\partial(t,x,y)}{\partial(\tau,\xi,\eta)} = \begin{bmatrix} t_\tau & t_\xi & t_\eta \\ x_\tau & x_\xi & x_\eta \\ y_\tau & y_\xi & y_\eta \end{bmatrix}, \tag{10}$$

$$\mathscr{J}^{-1} = \frac{\partial(\tau,\xi,\eta)}{\partial(t,x,y)} = \begin{bmatrix} \tau_t & \tau_x & \tau_y \\ \xi_t & \xi_x & \xi_y \\ \eta_t & \eta_x & \eta_y \end{bmatrix}. \tag{11}$$

Considering the fact that $\mathscr{J}\mathscr{J}^{-1} = \mathbf{I}$, $\mathscr{J}^{-1}$ can be expressed as the adjugate matrix of $\mathscr{J}$ over the determinant,

$$\mathscr{J}^{-1} = \frac{1}{|\mathscr{J}|}\mathscr{S}, \tag{12}$$

where $\mathscr{S}$ is the adjugate matrix of $\mathscr{J}$. Since the temporal transformation is independent of spatial coordinates, it results in $t_\tau = 1$, $t_\xi = 0$, $t_\eta = 0$. Substitute these values into $\mathscr{J}$ and after some algebra the following expression can be obtained for $\mathscr{S}$,

$$\mathscr{S} = \begin{bmatrix} |\mathscr{J}| & 0 & 0 \\ A & y_\eta & -x_\eta \\ B & -y_\xi & x_\xi \end{bmatrix}, \tag{13}$$

with $|\mathscr{J}| = x_\xi y_\eta - y_\xi x_\eta$, $A = -x_\tau y_\eta + y_\tau x_\eta$ and $B = x_\tau y_\xi - y_\tau x_\xi$. In the above $\mathscr{S}$ matrix, x_τ and y_τ can be interpreted as the grid velocities. For all the rotating- and deforming-mesh simulations in this study, the grid coordinates (x,y) as well as the velocities (x_τ, y_τ) are prescribed analytically.

For deforming grids, besides the above equations, the following Geometric Conservation Law (GCL) [25] has to be considered to ensure global conservation,

$$\begin{cases} \dfrac{\partial}{\partial\xi}(|\mathscr{J}|\xi_x) + \dfrac{\partial}{\partial\eta}(|\mathscr{J}|\eta_x) = 0 \\ \dfrac{\partial}{\partial\xi}(|\mathscr{J}|\xi_y) + \dfrac{\partial}{\partial\eta}(|\mathscr{J}|\eta_y) = 0 \\ \dfrac{\partial|\mathscr{J}|}{\partial\tau} + \dfrac{\partial}{\partial\xi}(|\mathscr{J}|\xi_t) + \dfrac{\partial}{\partial\eta}(|\mathscr{J}|\eta_t) = 0. \end{cases} \tag{14}$$

Since grid motions are given analytically in this study, the first two GCL equations are satisfied automatically. From the third equation, $\partial|\mathscr{J}|/\partial\tau$ is computed and added to Equation (8) as a source term.

NUMERICAL METHODS

In this section, we first give a brief review of the SD method. Subsequently, we describe a newly formulated sliding-mesh interface technique that is built on the SD formulation. For temporal discretization, an explicit strong stability preserving Runge-Kutta method [26] is used for all computations throughout this paper.

The SD Method

For SD method on quadrilateral grids, we first transform each cell from the physical domain to a standard square element $(0 \le \xi \le 1, 0 \le \eta \le 1)$ in the computational domain. The transformation can be done through iso-parametric mapping. As was reported in [27,28], using linear cell which is defined by 4 nodes is not good enough for problems involving curved boundaries. High order cubic cell with 12 nodes are used along the curved boundaries to ensure stability and accuracy in the present study.

After the mapping, solution points (SPs) and flux points (FPs) are defined on each standard element as shown in Figure 1 for a third order scheme. For an *N-th* order SD method, N SPs are required along each coordinate direction to construct degree $(N-1)$ solution polynomials. Due to the existence of first order spatial derivatives on fluxes in Equation (8), $(N+1)$ FPs must be employed in each direction to construct degree N polynomials. In the current implementation, the SPs: X_s, where $s = 1,2,...,N$, are chosen as N Chebyshev-Gauss points. The FPs: $X_{s+1/2}$, where $s = 0,1,2,...N$, are chosen as $(N-1)$ Legendre-Gauss points plus two end points to align in a staggered fashion with respect to the SPs (see [27,28] for more details).

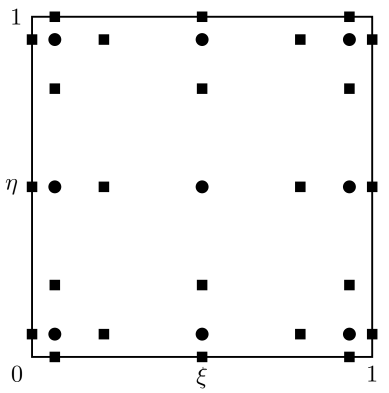


FIGURE 1. Schematic of distribution of solution points (circles) and flux points (squares) for a third order SD scheme.

To construct solution and flux polynomials, the following

Lagrange bases at the SPs and FPs are used,

$$h_i(X) = \prod_{s=1,s\neq i}^{N} \left(\frac{X - X_s}{X_i - X_s}\right), \tag{15}$$

$$l_{i+1/2}(X) = \prod_{s=0,s\neq i}^{N} \left(\frac{X - X_{s+1/2}}{X_{i+1/2} - X_{s+1/2}}\right). \tag{16}$$

The solution and fluxes within each element are simply tensor products of the Lagrange bases:

$$\mathbf{Q}(\xi,\eta) = \sum_{j=1}^{N}\sum_{i=1}^{N} \frac{\widetilde{\mathbf{Q}}_{i,j}}{|\mathscr{J}_{i,j}|} h_i(\xi)\cdot h_j(\eta), \tag{17}$$

$$\widetilde{\mathbf{F}}(\xi,\eta) = \sum_{j=0}^{N}\sum_{i=0}^{N} \widetilde{\mathbf{F}}_{i+1/2,j} l_{i+1/2}(\xi)\cdot h_j(\eta), \tag{18}$$

$$\widetilde{\mathbf{G}}(\xi,\eta) = \sum_{j=0}^{N}\sum_{i=0}^{N} \widetilde{\mathbf{G}}_{i,j+1/2} h_i(\xi)\cdot l_{j+1/2}(\eta). \tag{19}$$

The above reconstructed solution and fluxes are only element-wise continuous, but discontinuous across cell interfaces. For inviscid fluxes, a Riemann solver is employed to compute common flux at cell interfaces to ensure conservation and stability. In the current implementation, the Rusanov solver [29] has been used. For viscous fluxes on cell interfaces, we first compute the common values for both conservative variables and their gradients by averaging values from the left and right sides of each cell face, and common viscous fluxes are then computed from the common variables and gradients, details can be found in a previous paper by Liang, et al. [27].

The Sliding-mesh Approach

Sliding interfaces are formed between two domains with different motions. The simplest situation involves only one rotating mesh and one stationary mesh as shown in Figure 2. The inner mesh can rotate while the outer is fixed, or vice versa. Communication between the stationary and rotating meshes are realized through “mortars”. To make the explanation intuitive, we have scaled the inner mesh in order to place mortars in between two coupled meshes.

The mortars are arranged in a counterclockwise order. We refer the inner mesh as left (L) and the outer mesh as right (R) with respect to a mortar. To facilitate code implementation and reduce computational cost, cell faces on both sides of the sliding interface have been meshed uniformly. A closer look at Figure 2 reveals how mortars and cell faces on the sliding interface are connected: at each time instant, a cell face is connected to two mortars, and each mortar is associated with one left cell face and one right cell face. This face and mortar connectivity needs to be updated at every stage of the Runge-Kutta time stepping method.

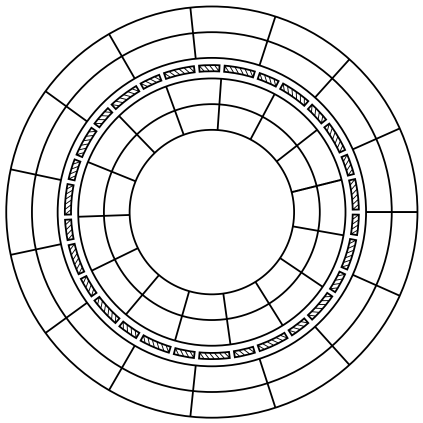


FIGURE 2. Schematic of distribution of mortars (hatched) between a rotating mesh and a stationary mesh.

Figure 3 shows a cell face Ω and its two mortars Ξ_1 and Ξ_2. Each curved mortar is mapped to a straight edge $0 \le z \le 1$ through $1D$ iso-parametric mapping. Face Ω is mapped to a straight edge $0 \le \xi \le 1$ when the associated cell is mapped to a standard square element, thus no extra mapping is required. ξ and z are related by

$$\xi = o(t) + s(t)z, \tag{20}$$

where $o(t)$ is the offset of the mortar relative to the bottom node of Ω at time t, and $s(t)$ is the relative scaling. For the example shown in Figure 3, we have $o_1 = 0$ and $s_1 = L^{\Xi_1}/L^{\Omega}$ for Ξ_1, $o_2 = L^{\Xi_1}/L^{\Omega}$ and $s_2 = L^{\Xi_2}/L^{\Omega}$ for Ξ_2, where L means physical length of face and mortars.

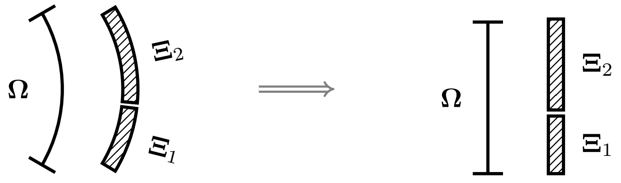


FIGURE 3. Mapping of curved cell face and mortars to straight ones: left, curved face and mortars in physical domain; right, straight face and mortars in computational domain.

According to Equation (17), solutions on Ω can be represented as,

$$\mathbf{Q}^{\Omega} = \sum_{i=1}^{N} \mathbf{Q}_i^{\Omega} h_i(\xi), \tag{21}$$

where $\mathbf{Q}_i^{\Omega}$ represents solution at the i-th SP on Ω, h_i is the Lagrange basis defined in Equation (15). If we define the same set of SPs on $0 \le z \le 1$ for each mortar, then solutions on each mortar can be reconstructed similarly as

$$\mathbf{Q}^{\Xi} = \sum_{i=1}^{N} \mathbf{Q}_i^{\Xi} h_i(z), \tag{22}$$

where $\mathbf{Q}_i^{\Xi}$ is the solution at the i-th SP on a mortar Ξ.

The procedure for computing $\mathbf{Q}_i^{\Xi}$ is demonstrated in Figure 4(a). For simplicity, we only show the process on the left side of mortar Ξ. To get the solutions, we require that,

$$\int_0^1 (\mathbf{Q}^{\Xi,L}(z) - \mathbf{Q}^{\Omega}(\xi)) h_j(z) dz = 0, \;\; j = 1,2,...,N. \tag{23}$$

It was shown in [30] that the above requirement is equivalent to an unweighted L^2 projection. Substitute Equations (20)-(22) into the above equation and evaluate it at each SP on Ξ will give a system of linear equations. The solution of this system when written in matrix form is,

$$\mathbf{Q}^{\Xi,L} = \mathbf{P}^{\Omega\to\Xi}\mathbf{Q}^{\Omega} = \mathbf{M}^{-1}\mathbf{S}^{\Omega\to\Xi}\mathbf{Q}^{\Omega}, \tag{24}$$

where $\mathbf{P}^{\Omega\to\Xi}$ is the projection matrix from Ω to Ξ, the matrices $\mathbf{M}$ and $\mathbf{S}^{\Omega\to\Xi}$ have the following elements,

$$M_{i,j} = \int_0^1 h_i(z) h_j(z) dz, \;\; i,j = 1,2,...,N, \tag{25}$$

$$S_{i,j}^{\Omega\to\Xi} = \int_0^1 h_i(o+sz) h_j(z) dz, \;\; i,j = 1,2,...,N, \tag{26}$$

where o and s are the offset and scaling of Ξ with respect to Ω. It is important to note that o and s are time-dependent for the sliding-mesh interface method.

The right solution vector $\mathbf{Q}^{\Xi,R}$ can be computed in the same way. Having both left and right solutions on a mortar, the Rusanov solver is employed to compute common inviscid flux $\mathbf{F}_{inv}^{\Xi}$. This flux is then transformed to the computational flux as $\widetilde{\mathbf{F}}_{inv}^{\Xi}$ according to Equation (9).

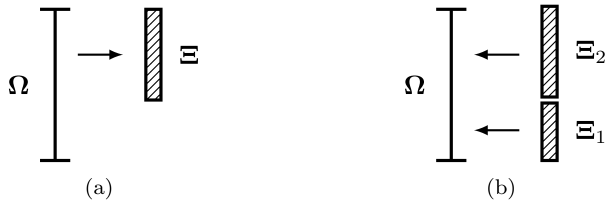


FIGURE 4. Projection between face and mortar: (a) from left face to left side of mortar, (b) from two mortars back to the associated left face.

As shown in Figure 4(b), to project the common inviscid fluxes $\widetilde{\mathbf{F}}_{inv}^{\Xi_1}$ and $\widetilde{\mathbf{F}}_{inv}^{\Xi_2}$ back to face Ω, we require that,

$$\int_0^{o_2} (\widetilde{\mathbf{F}}_{inv}^{\Omega}(\xi) - \widetilde{\mathbf{F}}_{inv}^{\Xi_1}(z)) h_j(\xi) d\xi + \int_{o_2}^1 (\widetilde{\mathbf{F}}_{inv}^{\Omega}(\xi) - \widetilde{\mathbf{F}}_{inv}^{\Xi_2}(z)) h_j(\xi) d\xi = 0, \;\; j = 1,2,...,N, \tag{27}$$

where $\widetilde{\mathbf{F}}_{inv}^{\Omega}(\xi)$ is the inviscid flux polynomial on face Ω. Solution of the above equation when written in matrix form is,

$$\begin{aligned}\widetilde{\mathbf{F}}_{inv}^{\Omega} &= \mathbf{P}^{\Xi_1\to\Omega}\widetilde{\mathbf{F}}_{inv}^{\Xi_1} + \mathbf{P}^{\Xi_2\to\Omega}\widetilde{\mathbf{F}}_{inv}^{\Xi_2} \\ &= s_1\mathbf{M}^{-1}\mathbf{S}^{\Xi_1\to\Omega}\widetilde{\mathbf{F}}_{inv}^{\Xi_1} + s_2\mathbf{M}^{-1}\mathbf{S}^{\Xi_2\to\Omega}\widetilde{\mathbf{F}}_{inv}^{\Xi_2},\end{aligned} \tag{28}$$

where the matrix $\mathbf{M}$ is identical to that in Equation (24), and matrices $\mathbf{S}^{\Xi_1\to\Omega}$ and $\mathbf{S}^{\Xi_2\to\Omega}$ are simply transposes of $\mathbf{S}^{\Omega\to\Xi_1}$ and $\mathbf{S}^{\Omega\to\Xi_2}$, respectively.

For the computation of common viscous fluxes, we first compute the common solution on each mortar as the average of the left and right solutions,

$$\mathbf{Q}^{\Xi} = \frac{1}{2}(\mathbf{Q}^{\Xi,L} + \mathbf{Q}^{\Xi,R}). \tag{29}$$

This common solution is then projected back to cell faces in the same procedure as for the inviscid flux in Equation (28). After that, solution gradients and viscous fluxes are updated on both side of the interfaces. The viscous fluxes $\widetilde{\mathbf{F}}_{vis}^{\Omega}$ on cell faces are projected to mortars in the same way as Equation (24). The common viscous flux $\widetilde{\mathbf{F}}_{vis}^{\Xi}$ on a mortar is the average of left and right viscous fluxes,

$$\widetilde{\mathbf{F}}_{vis}^{\Xi} = \frac{1}{2}(\widetilde{\mathbf{F}}_{vis}^{\Xi,L} + \widetilde{\mathbf{F}}_{vis}^{\Xi,R}). \tag{30}$$

The final step is to project $\widetilde{\mathbf{F}}_{vis}^{\Xi}$ back to faces, which is identical to the process in Equation (28).

Since uniform mesh is used for cell faces on the sliding interface, the **S** matrix only needs to be computed for the first two mortars, and can be reused by other corresponding mortars. At the same time since the **M** matrix is time independent, it can be precomputed before the actual calculation. To compute the integrals in Equations (25) and (26), one can use the Clenshaw-Curtis quadrature method as was used in [30]. In this study, the integrand is casted into a general form as product of $2(N-1)$ first degree polynomials, and we implement a recursive algorithm to compute the integrals analytically. This integration approach requires the least number of operations, and is much more efficient than the Clenshaw-Curtis quadrature method.

NUMERICAL VERIFICATIONS

In this section we test the accuracy of the solver on an inviscid flows, and then verify it on a viscous flow. A five-stage forth order Runge-Kutta method for time stepping [26] is used for all test cases. For accuracy tests, various time step sizes were used in each case to make sure that the errors are time step size independent and are dominated by spacial discretization errors.

Euler Vortex Flow on Dynamic Grid

We first test the solver on an inviscid flow. For inviscid flow test, Euler vortex problem is an ideal choice as it has been used by many researchers, one of such an example can be found in [31]. In Euler vortex problem, an isentropic vortex is superimposed to an uniform mean flow and convected by the mean flow. The flow field in an infinite domain at a time instant t can be analytically expressed as,

$$u = U_\infty \left\{ \cos\theta - \frac{\varepsilon y_r}{r_c} \exp\left(\frac{1 - x_r^2 - y_r^2}{2r_c^2} \right) \right\} \tag{31}$$

$$v = U_\infty \left\{ \sin\theta + \frac{\varepsilon x_r}{r_c} \exp\left(\frac{1 - x_r^2 - y_r^2}{2r_c^2} \right) \right\} \tag{32}$$

$$\rho = \rho_\infty \left\{ 1 - \frac{(\gamma-1)(\varepsilon M_\infty)^2}{2} \exp\left(\frac{1 - x_r^2 - y_r^2}{r_c^2} \right) \right\}^{\frac{1}{\gamma-1}} \tag{33}$$

$$p = p_\infty \left\{ 1 - \frac{(\gamma-1)(\varepsilon M_\infty)^2}{2} \exp\left(\frac{1 - x_r^2 - y_r^2}{r_c^2} \right) \right\}^{\frac{\gamma}{\gamma-1}} \tag{34}$$

where U_∞, ρ_∞, p_∞, M_∞ are the mean flow speed, density, pressure and Mach number, respectively. θ is the direction of the mean flow (i.e. the direction along which the vortex is convected), ε and r_c can be interpreted as the vortex strength and size. The relative coordinates (x_r, y_r) are defined as,

$$x_r = x - x_0 - \bar{u}t, \tag{35}$$

$$y_r = y - y_0 - \bar{v}t, \tag{36}$$

where $\bar{u} = U_\infty \cos\theta$, $\bar{v} = U_\infty \sin\theta$ are the x and y components of the mean velocity, (x_0, y_0) is the initial position of the vortex. The exact solution of Euler vortex within a square domain ($0 \le x, y \le L$) with periodic boundary conditions can be achieved by replacing the relative coordinates with the following expressions,

$$x_r = x_r - \lfloor \frac{x_r + x_0}{L} \rfloor \cdot L, \tag{37}$$

$$y_r = y_r - \lfloor \frac{y_r + y_0}{L} \rfloor \cdot L, \tag{38}$$

where the floor operator $\lfloor x \rfloor$ gives the largest integer that is not greater than a real number x, the x_r and y_r on the right hand side are from Equation (35) and (36).

In this test, the uniform mean flow is chosen as $(U_\infty, \rho_\infty, p_\infty) = (1, 1, 1)$ with a Mach number of $M_\infty = 0.3$. The flow direction is set to $\theta = \arctan(1/2)$. A vortex with parameters: $\varepsilon = 1$, $r_c = 1$, is superimposed to the mean flow. The domain size is $0 \le x, y \le 10$ (i.e. $L = 10$), and the vortex is initially placed at the domain center $(x_0, y_0) = (5, 5)$. Periodic boundary conditions are applied in both x and y directions.

The computational domain is divided into two parts: an inner part with a radius of 2; an outer part which takes the rest of the domain. The inner domain undergoes a rotating motion around its center at a angular speed of $\omega = \pi$, at the same time it heaves vertically with the displacement prescribed as $h = 0.5\sin(\omega t)$. Due to the heaving motion, the outer domain has to be deformed to avoid overlap between meshes. The deformation is realized through a blending function approach, more details can be found in [32]. Three meshes with 180, 700 and 2731 cells have been used for accuracy tests.

Figure (5) shows the meshes and density contours at different time instants during a period. As we can see, the rotation and deformation of the mesh do not cause any change on the shape of the vortex. Qualitatively, this indicates that the sliding-mesh approach does not bring noticeable errors to the solution.

To quantitatively evaluate the accuracy of the approach, Table 1 and Table 2 give the errors and spatial accuracy of the third- and fourth-order schemes. The L_1 and L_2 errors are computed from density at $t = 2$ when vortex center is traveled right onto the sliding interface. From the two tables we see that the solver gives very reasonable order of accuracy.

To see how efficient the solver is, we compare the total computational time and the communication time on the sliding interface in Table 3 and Table 4 for third- and fourth-order schemes, respectively. Times in both tables are collected for 100 computational steps and do not include any post-processing time. It is seen that for all test cases, communication on the sliding interface takes only a few percent of the total computational time, which clearly shows that the solver is efficient. What is interesting is that the relative communication time (represented by the percentage) decreases as either number of cells or order of

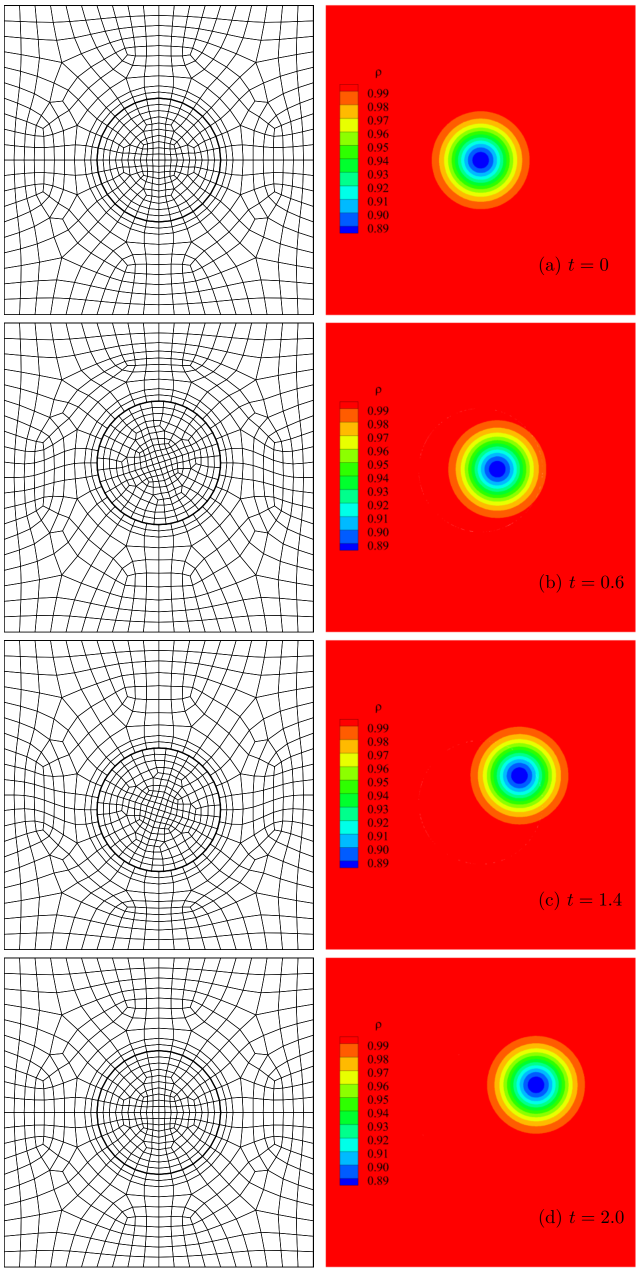


FIGURE 5. Meshes and density contours of Euler vortex flow different time instants (blue circle indicates sliding-mesh interface).

cells	L1 error	order	L2 error	order
180	3.17E-4	-	7.23E-4	-
700	4.33E-5	2.93	9.65E-5	2.96
2731	5.93E-6	2.92	1.40E-5	2.90

TABLE 1. Error and order of accuracy of the 3^{rd} order scheme on Euler vortex flow.

cells	L1 error	order	L2 error	order
180	5.89E-5	-	1.20E-4	-
700	3.72E-6	4.07	8.22E-6	3.95
2731	2.31E-7	4.07	5.47E-7	3.97

TABLE 2. Error and order of accuracy of the 4^{th} order scheme on Euler vortex flow.

cells	total time	comm. time	percentage
180	0.408551	0.016154	3.95%
700	1.361268	0.033109	2.43%
2731	5.355175	0.080721	1.51%

TABLE 3. Total computation time and interface communication time (both in seconds) for 100 computational steps using 3^{rd} oder scheme on Euler vortex flow.

cells	total time	comm. time	percentage
180	0.744496	0.020697	2.78%
700	2.299031	0.046258	2.01%
2731	9.241918	0.108961	1.18%

TABLE 4. Total computation time and interface communication time (both in seconds) for 100 computational steps using 4^{th} oder scheme on Euler vortex flow.

schemes increases. This is due to the fact that cells are one dimension higher that faces: when perform a mesh refinement, the total number of faces in the domain grows faster than on the sliding interface; when increase scheme order, the total number of degrees of freedom in the domain also grows faster than on the sliding interface.

Flow over a Plunging and Pitching Airfoil

In this section, we verify the solver on flow over a plunging and pitching NACA0012 airfoil which has a chord length of $c = 1$. The vertical plunging motion is given as $h(t) = A\sin(2\pi\omega t)$, where h is the displacement, A is the amplitude of plunging and is set to $0.25c$. The pitching motion is prescribed as $\theta(t) = \alpha\cos(2\pi\omega t)$, where θ is the pitching angle (angle between the airfoil and freestream direction), and $\alpha = \pi/6$ is the maximum pitching angle. The pitching center is one third chord length behind the leading edge. The two motions share the same frequency which is set to $\omega = 0.4$. The freestream flow has a Mach number of $M_\infty = 0.2$. The Reynolds number based on freestream velocity and airfoil chord length is $Re = 1000$.

The overall computational domain has a size of $100c \times 100c$ with the airfoil located $30c$ downstream from the inlet. The domain is decomposed into two parts: an inner circular part with a radius of $r = 2c$, an outer part which takes the rest of the domain. The inner domain undergoes pitching and plunging motion as a rigid body, thus no grid deformation is involved. Mesh in the outer part is set to deform moderately due to the plunging motion only, thus grid cells are only squeezed or expanded but not twisted. A sliding-mesh interface is formed between these two parts due to a relative pitching motion. Doing this is particularly important for plunging and pitching airfoils with large pitching angles since it removes the otherwise very large skewness on the grid [1]. Figure 6 shows two local views of the mesh used for this test case. The inner domain is meshed into 4600 cells, and the outer 5059 cells. Mesh is refined around the leading and trailing edges as well as in the wake region. Dirichlet boundary condition is applied at the inlet, while pressure boundary condition at the outlet. The top and bottom boundaries are set to slip non-penetration walls. The airfoil surface is set as no-slip isothermal wall. The time step size for the simulation is set to $\Delta t U_\infty / c = 2.0 \times 10^{-4}$.

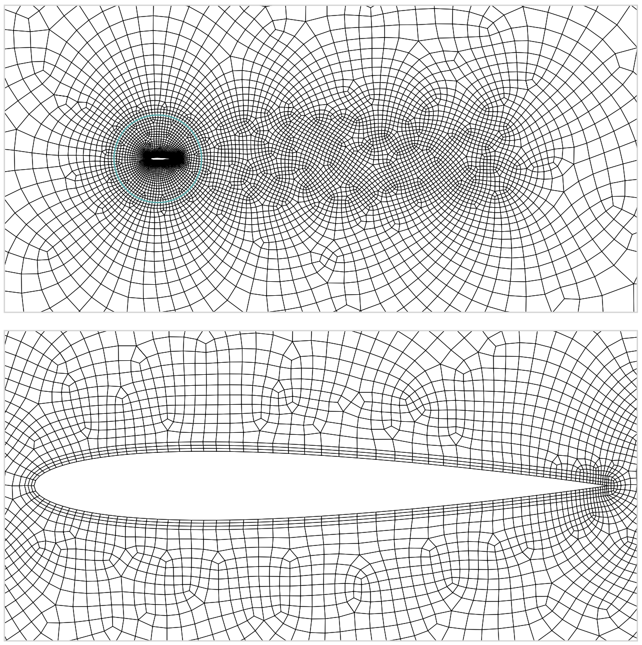


FIGURE 6. Two local views of mesh around a NACA0012 airfoil (blue circle indicates sliding interface).

Both third and fourth order schemes were used for this case, but no visible difference was seen between the results. Figure 7 shows the vorticity contours at a down stroke instant and a up stroke instant. We see that a counterclockwise and a clockwise vortices are shed off from the trailing edge as the airfoil moves, which results in a vortex street in the wake region. The wake structures are very coherent which indicates that interactions between two successive vortices are not strong. The contours also show that the vortices pass the sliding-mesh interface smoothly and the interface does not bring any noticeable effects to the flow field.

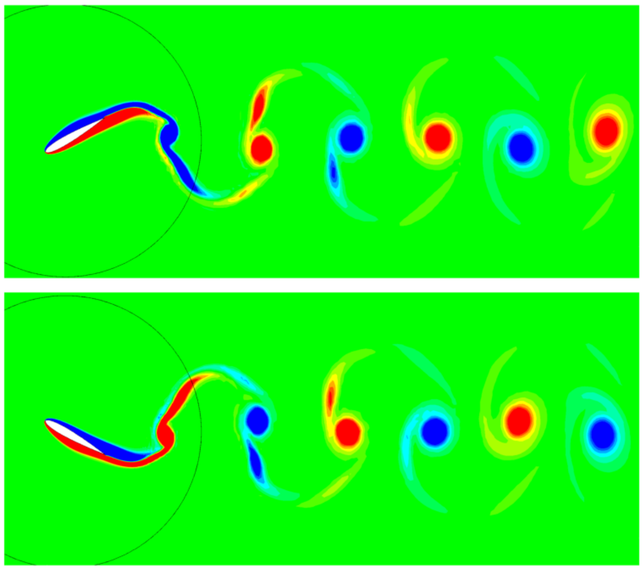
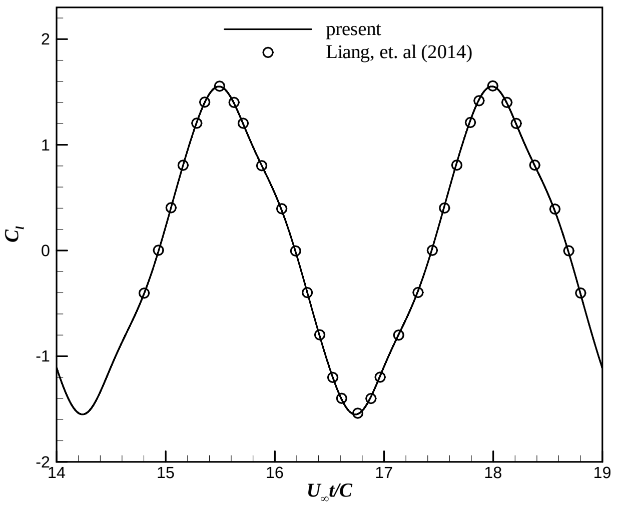


FIGURE 7. Vorticity contours (blue means negative value, red means positive) for flow over a plunging and pitching air at two time instants (circle is sliding-mesh interface).

The C_L and C_D curves are plotted in Figure 8 together with previous results from a flux reconstruction solver on moving and deforming grid reported by Liang, et. al. [32]. We see very good agreement between the two results, which verifies the correctness of the new solver with sliding-mesh interface for viscous flow. It is interesting to notice that the drag coefficient varies twice as fast as the lift coefficient. This is because that the drag is mainly due to the blockage effects from the airfoil since the blockage area changes twice as fast as the plunging/pitching motions. We also see that for the prescribed motion, the airfoil experiences a small mean drag and a almost negligible mean lift.

The efficiencies of the third- and fourth-order schemes on this problem is tabled in Table 5. It is seen that communications on the sliding-mesh interface introduces negligible computational cost.



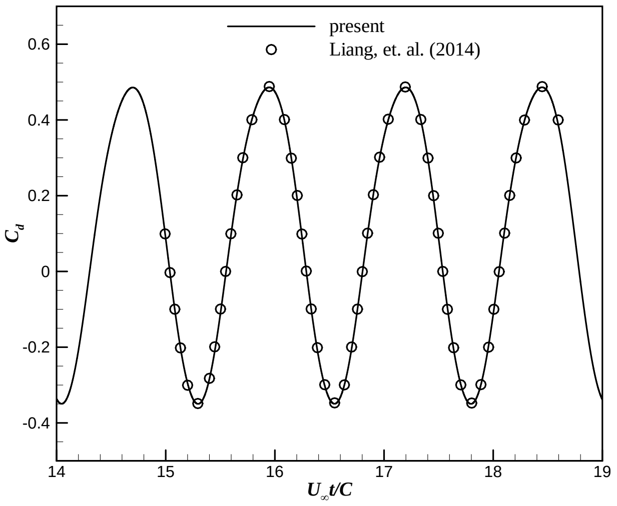


FIGURE 8. Lift and drag coefficients for flow over a plunging and pitching airfoil.

order	total time	comm. time	percentage
3	50.718697	0.235566	0.46%
4	89.858273	0.313078	0.35%

TABLE 5. Total computation time and interface communication time (both in seconds) for 100 computational steps for simulation of flow around a plunging and pitching airfoil.

STUDY OF A 2D VERTICAL AXIS WIND TURBINE

In this last test case, we apply the solver to simulate flow around a 2D vertical axis wind turbine. Figure 9 shows two local views of the mesh used for this simulation. The fixed vertical axis has a circular cross section with a radius of 0.2. NACA0015 airfoil profile is widely used for vertical axis wind turbine blades [33] and is employed in this simulation as well. The chord length of each blade is choose as $c = 1$, and the center is at $c/3$. The three blades are uniformly distributed along a circle whose radius is 1.8. The overall computational domain has a size of 100×100, and is divided into three parts: an inner part with a radius of 0.5 around the circular axis; a middle part between $r = 0.5$ and $r = 3$ that contains the turbine; an outer part which takes the rest of the domain. The circular axis is located on the centerline of the domain, and 30c downstream from the inlet. The inner domain is meshed to 480 cells, the middle 9537 cells, and the outer 5604 cells. Mesh is refined around the blades, the axis, as well as in the wake region of the turbine. The turbine rotates counterclockwise at an angular speed of $\omega = 0.25\pi$. Three test cases at different Reynolds numbers have been studied, the first case has a Reynolds number of $Re = 1000$, the second one has $Re = 10000$, where the Reynolds number is based on inflow speed and the blade chord length. The third case is an inviscid flow simulation (corresponds to Reynolds number infinity). For all cases, the inflow has a Mach number of 0.1, Dirichlet boundary condition is used at the inlet, while pressure boundary condition for the outlet, meanwhile, slip boundary conditions are adopted on the top and bottom boundaries of the domain. For the two viscous flow cases, no-slip isothermal wall boundary condition is applied on the axis surface, and adiabatic wall boundary conditions are adopted for the blades, and for the inviscid flow case no penetration condition is applied on these walls. A fourth-order scheme is used for all simulations.

Figure 10 shows the flow field around the turbine at several time instants during one rotating period for $Re = 1000$. As we can see, in the front region of the turbine, each blade interacts with only the incoming flow most of the time. In the back region, the flow becomes very complicated, each blade interacts strongly with vortices from other blades. Some vortices are convected downstream by the mean flow, and some recirculate in the near wake region, which leads to a very vortical flow field. In Figure 10, the phase angle is measured with respect to the initial position of the the turbine as shown in Figure 9.

To evaluate the efficiency of the turbine, the total force on each blade is decomposed into two components: one that is parallel to the blade; the other that is perpendicular to the blade. For each blade, we normalize its two force components by $\frac{1}{2}\rho_\infty u_\infty^2 c$ and denote the two resulted coefficients as C_τ and C_n, respectively. When C_τ is positive, the force is in the same direction as the moving direction of the blade, and the blade absorbs energy from the flow. When C_τ is negative, then force and blade are in opposite directions and the blade loses energy to the flow. Thus C_τ can be used to measure the efficiency of each blade. The total efficiency of the turbine can be measured by the summation of C_τ's from each blade, we denote it as C_T.

Figure 11 shows the time histories of C_τ for each blade and

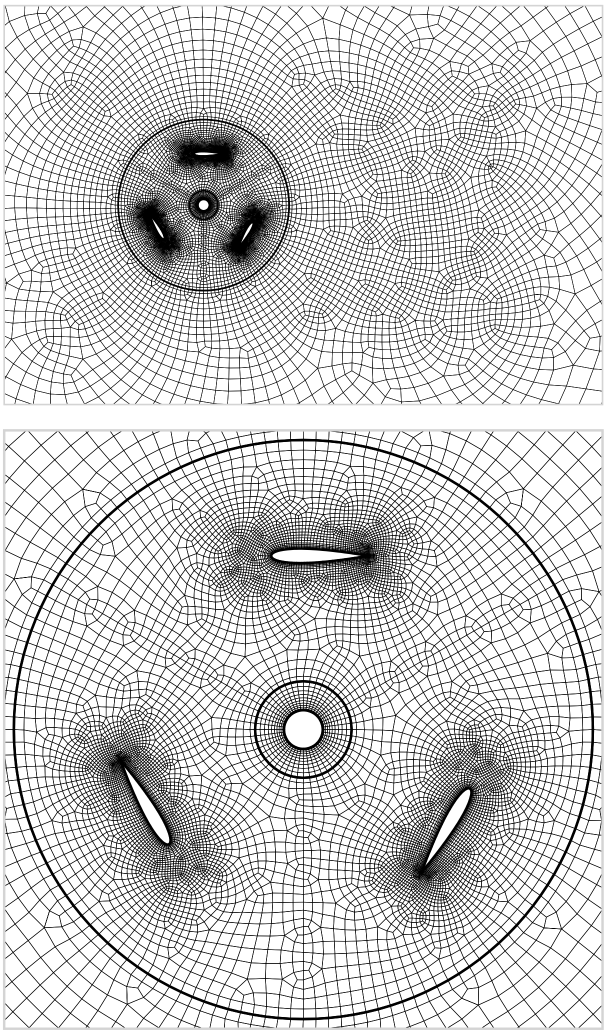


FIGURE 9. Two local views of mesh for flow around a 2D vertical axis wind turbine (two thick circles indicate sliding interfaces).

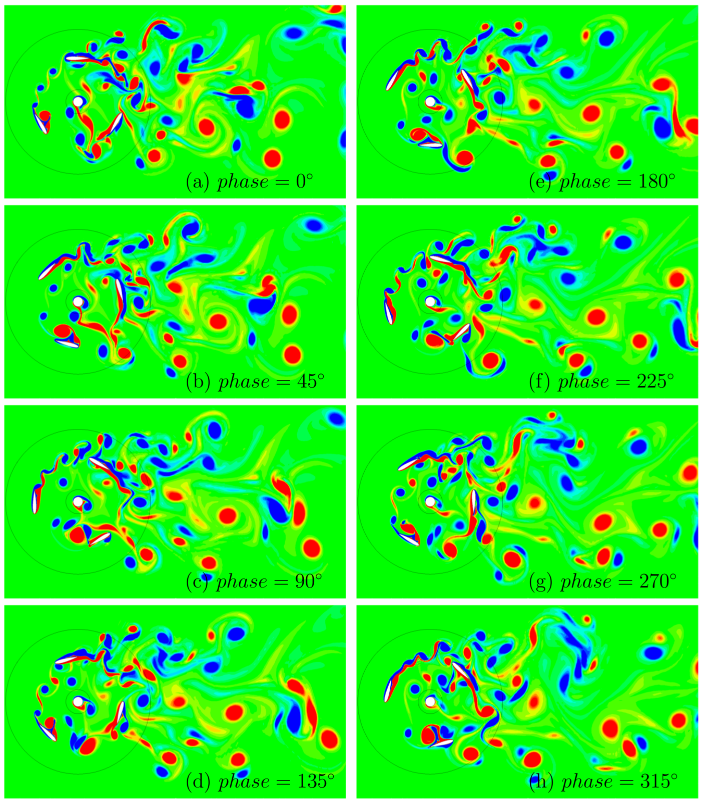


FIGURE 10. Vorticity contours of flow around a 2D vertical axis wind turbine at different phases at $Re = 1000$.

the total coefficient C_T at $Re = 1000$. The three C_τ curves share similar shapes, but have phase differences. It is seen that although each blade experiences positive C_τ sometimes, C_τ is negative most of the time. This results in negative mean values for C_τ and C_T, which means that overall the turbine loses energy to the flow. The reason for this is explained in Figure 12.

Phase averaged C_τ and C_T for $Re = 1000$ are plotted in Figure 12. The averages are done over 20 rotating periods. The rotating angle in Figure 12 is defined as the angle between positive x direction (freestream direction) and the position vector (vector from axis center to blade center) of each blade. For example, as shown in Figure 9, the top blade (blade 1) has a rotating angle of 90°, the bottom left one (blade 2) 210°, and the bottom right one (blade 3) 330°. The phase angle is measured with respect to the initial position of the the turbine as shown in Figure 9. To investigate the cause of energy loss, $\bar{C}_T$ is decomposed into two parts: pressure effect and viscous friction effect. The friction contribution is plotted together with $\bar{C}_T$ in Figure 12. It is clearly seen that friction generates a negative parallel force at every phase angle, which degrades the turbine performance dramatically. In real applications, the Reynolds number of this flow is in the order of 1.0×10^6, which indicates much smaller friction effect. In what follows, results from $Re = 10000$ and inviscid flow (Reynolds number infinity) are discussed.

For $Re = 10000$, the time histories of C_τ and C_T are plotted in Figure 13. By increasing the Reynolds number we see obvious increase in C_τ and C_T. The phase averaged values are shown in Figure 14, it is seen that the turbine absorbs energy from the flow most of the time. From the first plot in Figure 14, it is interesting to notice that each blade gains energy approximately between rotating angle of 120° and 240°, that is when it travels to the front portion of the turbine; and during the rest part of the rotation, there's no obvious gaining or losing of energy for each blade.

Results of the inviscid flow simulation are shown in Figure 15 and Figure 16. Comparing with the previous case, we see

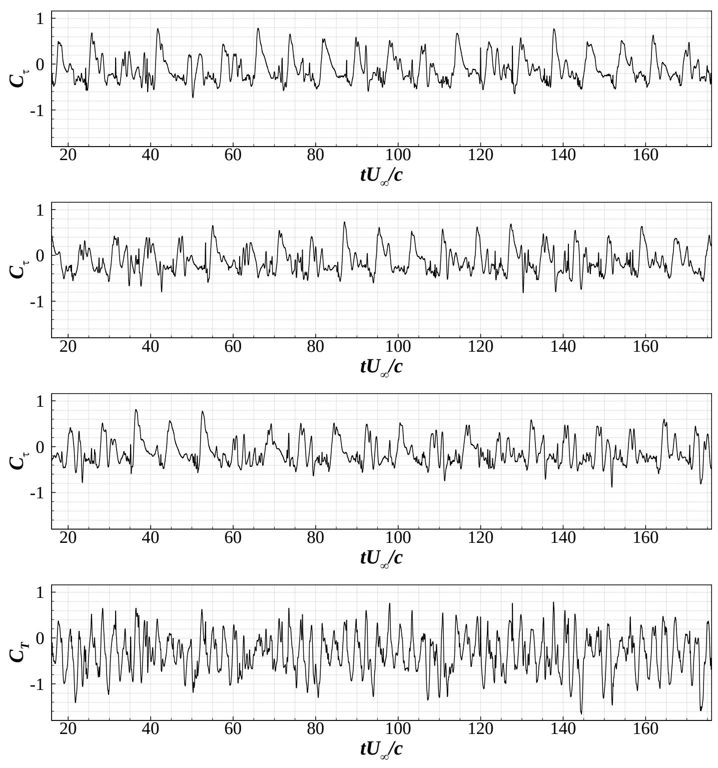


FIGURE 11. Coefficients of parallel force at $Re = 1000$ (from top to bottom: blade 1, blade 2, blade 3, total).

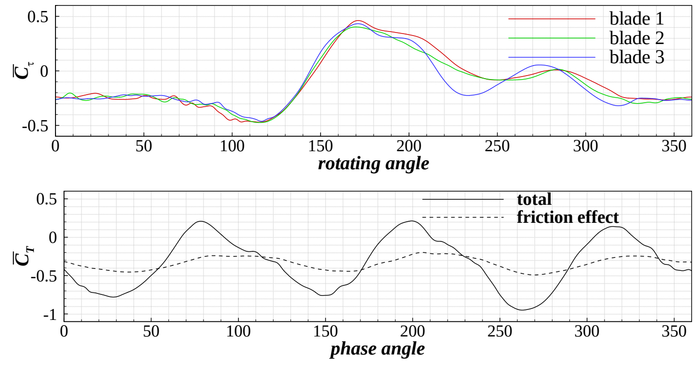


FIGURE 12. Phase averaged coefficient of parallel force at $Re = 1000$ (top, force on each blade; bottom, total force).

further increase in C_τ and C_T, which means higher efficiency is gained by the turbine. At the same time, the elimination of viscosity not only removes the higher frequency components (resulting in smoother curves) but also introduce phase differences on the curves.

Finally, the efficiency of the solver on this problem is tabled in Table 6. Again, it is found that the solver is very efficient.

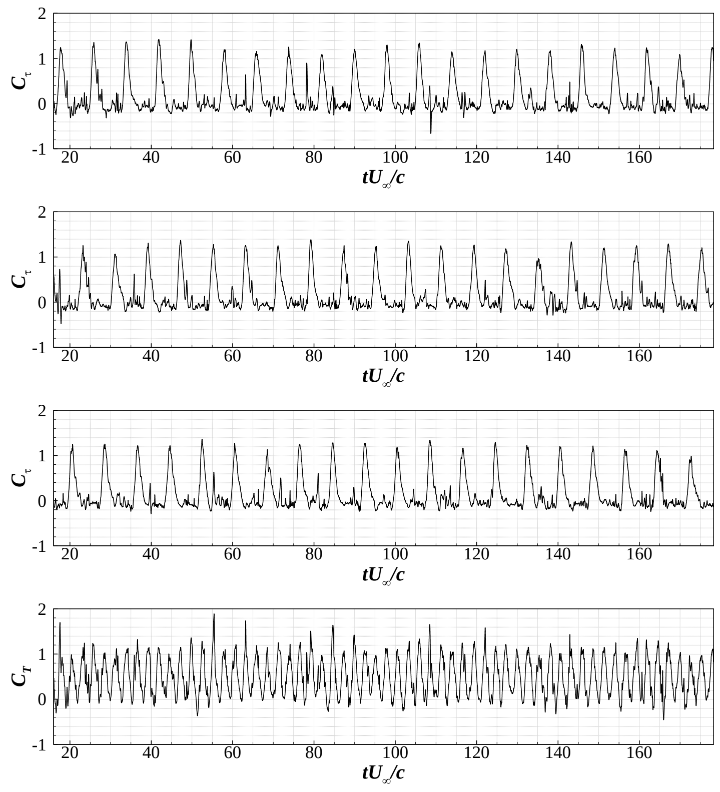


FIGURE 13. Coefficients of parallel force at $Re = 10000$ (from top to bottom: blade 1, blade 2, blade 3, total).

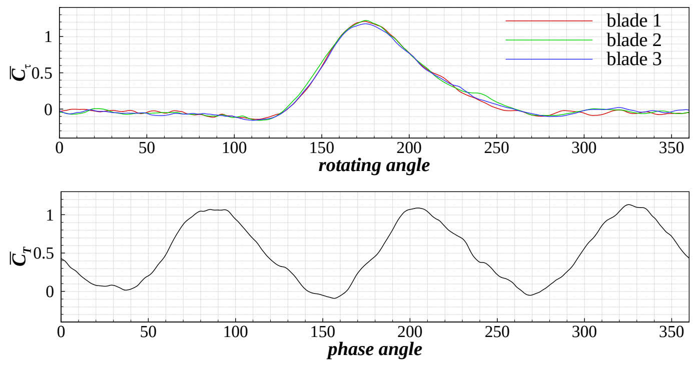


FIGURE 14. Phase averaged coefficient of parallel force at $Re = 10000$ (top, force on each blade; bottom, total force).

CONCLUSIONS

A new high-order accurate solver for the two-dimensional compressible Navier-Stokes equation on coupled rotating and deforming/stationary domains with sliding-mesh interfaces is developed and tested. The solver is shown to be very accurate, at the same time the communications on the sliding-mesh interfaces are shown to be very efficient and introduce negligible computa-

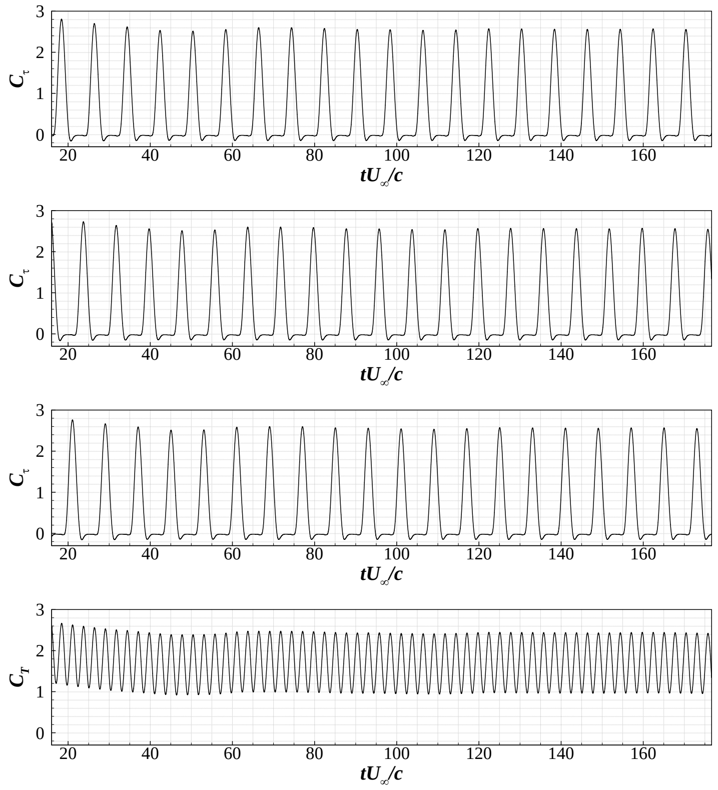


FIGURE 15. Coefficients of parallel force for inviscid flow (from top to bottom: blade 1, blade 2, blade 3, total).

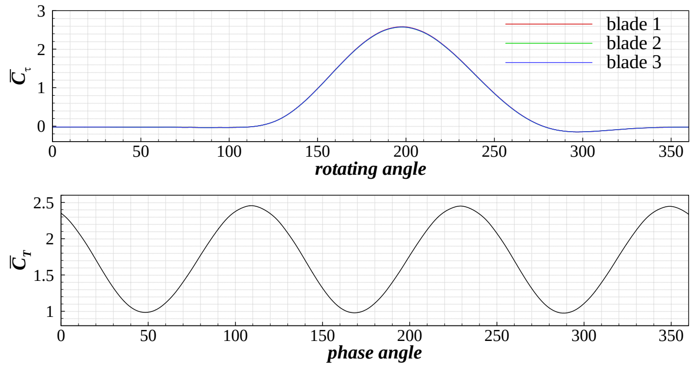


FIGURE 16. Phase averaged coefficient of parallel force for inviscid flow (top, force on each blade; bottom, total force).

tional cost to the solver. The high-order curved sliding-mesh interface method can also be extended to other discontinuous high-order methods for compressible flows, and can also be extend to 3*D* flow solvers. This solver can be applied to a wind range of problems, such as the aerodynamics of rotorcrafts, oscillating wing wind power generators.

order	total time	comm. time	percentage
3	83.933235	0.464098	0.55%
4	146.68811	0.617841	0.42%

TABLE 6. Total computation time and interface communication time (both in seconds) for 100 computational steps for simulation of flow around a 2D vertical axis wind turbine.

ACKNOWLEDGMENT

The authors would like to express our acknowledgments for the support from an ONR Young Investigator Program awarded to Chunlei Liang and administrated by Dr. Ki-Han Kim.

REFERENCES

[1] Zhang, B., and Liang, C., 2015. “A simple, efficient, high-order accurate sliding-mesh interface approach to FR/CPR method on coupled rotating and stationary domains”. *AIAA paper 2015-1742*.

[2] Wang, Z. J., Fidkowski, K., Abgrall, R., Bassi, F., Caraeni, D., Cary, A., Deconinck, H., Hartmann, R., Hillewaert, K., Huynh, H. T., Kroll, N., May, G., Persson, P.-O., van Leer, B., and Visbal, M., 2013. “High-order CFD methods: current status and perspective”. *International Journal for Numerical Methods in Fluids,* ***72***(8), pp. 811–845.

[3] Karniadakis, G. E., and Sherwin, S. J., 2005. *Spectral/hp Element Methods for Computational Fluid Dynamics*, 2^{nd} ed. Oxford University Press, Oxford.

[4] Hesthaven, J. S., and Warburton, T., 2008. *Nodal Discontinuous Galerkin Methods: Algorithms, Analysis, and Applications*, Vol. 54 of *Texts in Applied Mathematics*. Springer, New York.

[5] Wang, Z. J., ed., 2011. *Adaptive High-Order Methods in Computational Fluid Dynamics*, Vol. 2 of *Advances in Computational Fluid Dynamics*. World Scientific.

[6] Cockburn, B., Karniadakis, G. E., and Shu, C.-W., eds., 2011. *Discontinuous Galerkin Methods: Theory, Computation and Applications*, Vol. 11 of *Lecture Notes in Computational Science and Engineering*. Springer, New York.

[7] Ekaterinaris, J. A., 2005. “High-order accurate, low numerical diffusion methods for aerodynamics”. *Progress in Aerospace Sciences,* ***41***(3-4), April-May, pp. 192–300.

[8] Wang, Z. J., 2007. “High-order methods for the Euler and Navier-Stokes equations on unstructured grids”. *Progress in Aerospace Sciences,* ***43***(1-3), pp. 1–41.

[9] Reed, W. H., and Hill, T. R., 1973. Triangular mesh methods for the neutron transport equation. Tech. Rep. LA-UR–73-479; CONF-730414–2, Los Alamos Scientific Laboratory.

[10] Cockburn, B., Hou, S., and Shu, C.-W., 1990. “The Runge-

Kutta local projection discontinuous Galerkin finite element method for conservation laws IV: The multidimensional case". *Mathematics of Computation,* ***54***(190), April, pp. 545–581.

[11] Cockburn, B., and Shu, C.-W., 2001. "Runge-Kutta discontinuous Galerkin methods for convection-dominated problems". *Journal of Scientific Computing,* ***16***(3), September, pp. 173–261.

[12] F.Bassi, and Rebay, S., 1997. "High-order accurate discontinuous finite element solution of the 2D Euler equations". *Journal of Computational Physics,* ***138***(2), pp. 251–285.

[13] Bassi, F., and Rebay, S., 1997. "A high-order accurate discontinuous finite element method for the numerical solution of the compressible Navier-Stokes equations". *Journal of Computational Physics,* ***131***, pp. 267–279.

[14] Jameson, A., and Lodato, G., 2014. "A note on the numerical dissipation from high-order discontinuous finite element schemes". *Computers & Fluids,* ***98***(2), September, pp. 186–195.

[15] Kopriva, D. A., and Kolias, J. H., 1996. "A conservative staggered-grid Chebyshev multidomain method for compressible flows.". *Journal of Computational Physics,* ***125***, pp. 244–261.

[16] Liu, Y., Vinokur, M., and Wang, Z. J., 2006. "Spectral difference method for unstructured grids I: Basic formulation". *Journal of Computational Physics,* ***216***, pp. 780–801.

[17] Wang, Z. J., Liu, Y., May, G., and Jameson, A., 2007. "Spectral difference method for unstructured grids II: Extension to the Euler equations". *Journal of Scientific Computing,* ***32***, pp. 45–71.

[18] Liang, C., Premasuthan, S., Jameson, A., and Wang, Z. J., 2009. "Large eddy simulation of compressible turbulent channel flow with spectral difference method". *AIAA paper 2009-402*.

[19] Mohammad, A. H., Wang, Z. J., and Liang, C., 2010. "LES of turbulent flow past a cylinder using spectral difference method". *Advances in Applied Mathematics and Mechanics,* ***2***, pp. 451–466.

[20] Parsani, M., Ghorbaniasl, G., Lacor, C., and Turkel, E., 2010. "An implicit high-order spectral difference approach for large eddy simulation". *Journal of Computational Physics,* ***229***, pp. 5373–5393.

[21] Lodato, G., Castonguay, P., and Jameson, A., 2014. "Structural wall-modeled LES using a high-order spectral difference scheme for unstructured meshes". *Flow, Turbulence and Combustion,* ***92***(1-2), pp. 579–606.

[22] Ou, K., Liang, C., and Jameson, A., 2010. "High-order spectral difference method for the Navier-Stokes equations on unstructured moving deforming grids". *AIAA paper 2010-0541*.

[23] Yu, M. L., Wang, Z. J., and Hu, H., 2011. "A high-order spectral difference method for unstructured dynamic grids". *Computers & Fluids,* ***48***, pp. 84–97.

[24] Liang, C., Ou, K., Premasuthan, S., Jameson, A., and Wang, Z., 2011. "High-order accurate simulations of unsteady flow past plunging and pitching airfoils". *Computers & Fluids,* ***40***, pp. 236–248.

[25] Thomas, P. D., and Lombard, C. K., 1979. "Geometric conservation law and its application to flow computations on moving grids". *AIAA Journal,* ***17***.

[26] Spiteri, R. J., and Ruuth, S. J., 2002. "A new class of optimal high-order strong-stability-preserving time discretization methods". *SIAM J. Numer. Anal.,* ***40***, pp. 469–491.

[27] Liang, C., Jameson, A., and Wang, Z. J., 2009. "Spectral difference method for two-dimensional compressible flow on unstructured grids with mixed elements". *Journal of Computational Physics,* ***228***, pp. 2847–2858.

[28] Liang, C., Premasuthan, S., and Jameson, A., 2009. "High-order accurate simulation of low-mach laminar flow past two side-by-side cylinders using spectral difference method". *Computers & Structures,* ***87***, pp. 812–817.

[29] Rusanov, V. V., 1961. "Calculation of interaction of nonsteady shock waves with obstacles". *Journal of Computational and Mathematical Physics USSR,* ***1***, pp. 267–279.

[30] Kopriva, D. A., 1996. "A conservative staggered-grid Chebyshev multidomain method for compressible flows. II. A semi-structured method". *Journal of Computational Physics,* ***128***, pp. 475–488.

[31] Erlebacher, G., Hussaini, M. Y., and Shu, C.-W., 1997. "Interaction of a shock with a longitudinal vortex". *Journal of Fluid Mechanics,* ***337***.

[32] Liang, C., Miyaji, K., and Zhang, B., 2014. "An efficient correction procedure via reconstruction for simulation of viscous flow on moving and deforming domains". *Journal of Computational Physics,* ***256***, pp. 55–68.

[33] Li, C., Zhu, S., Xu, Y., and Xiao, Y., 2013. "2.5D large eddy simulation of vertical axis wind turbine in consideration of high angle of attack flow". *Renewable Energy,* ***51***, March, pp. 317–330.