Prepared for *Archives of Computational Methods in Engineering*

Yuri Bazilevs · Kenji Takizawa · Tayfun E. Tezduyar · Ming-Chen Hsu · Nikolay Kostov · Spenser McIntyre

Aerodynamic and FSI Analysis of Wind Turbines with the ALE-VMS and ST-VMS Methods

Abstract We provide an overview of the aerodynamic and FSI analysis of wind turbines the first three authors' teams carried out in recent years with the ALE-VMS and ST-VMS methods. The ALE-VMS method is the variational multiscale version of the Arbitrary Lagrangian–Eulerian (ALE) method. The VMS components are from the residual-based VMS (RBVMS) method. The ST-VMS method is the VMS version of the Deforming-Spatial-Domain/Stabilized Space–Time (DSD/SST) method. The techniques complementing these core methods include weak enforcement of the essential boundary conditions, NURBS-based isogeometric analysis, using NURBS basis functions in temporal representation of the rotor motion, mesh motion and also in remeshing, rotation representation with constant angular velocity, Kirchhoff–Love shell modeling of the rotor-blade structure, and full FSI coupling. The analysis cases include the aerodynamics of standalone wind-turbine rotors, wind-turbine rotor and tower, and the FSI that accounts for the deformation of the rotor blades. The specific wind turbines considered are NREL 5MW, NREL Phase VI and Micon 65/13M, all at full scale, and our analysis for NREL Phase VI and Micon 65/13M includes comparison with the experimental data.

Keywords Wind turbines · Aerodynamic analysis · Fluid–structure interaction analysis · Wind-turbine rotor · Wind-turbine tower · ALE-VMS method · ST-VMS method · DSD/SST method · Weak enforcement of essential boundary conditions · Finite elements · Isogeometric analysis · NURBS · Temporal NURBS · Rotation representation with constant angular velocity · Mesh motion · Remeshing · Rotation-free shells · NREL Phase VI wind turbine · Micon 65/13M wind turbine · NREL 5MW offshore wind turbine · Blade pre-bending.

Y. Bazilevs
Structural Engineering, University of California, San Diego
9500 Gilman Drive, La Jolla, CA 92093, USA
E-mail: jbazilevs@ucsd.edu

K. Takizawa
Department of Modern Mechanical Engineering and
Waseda Institute for Advanced Study, Waseda University
1-6-1 Nishi-Waseda, Shinjuku-ku,Tokyo 169-8050, Japan

T.E. Tezduyar
Mechanical Engineering, Rice University – MS 321
6100 Main Street, Houston, TX 77005, USA

M.-C. Hsu
Department of Mechanical Engineering
Iowa State University
2025 Black Engineering, Ames, IA 50011, USA

N. Kostov and S. McIntyre
Mechanical Engineering, Rice University – MS 321
6100 Main Street, Houston, TX 77005, USA

1 Introduction

Countries around the world are putting substantial effort into the development of wind energy technologies. The ambitious wind energy goals put pressure on the wind energy industry research and development to significantly enhance the current wind generation capabilities in a short period of time and decrease the associated costs. This calls for transformative concepts and designs (e.g., floating offshore wind turbines) that must be created and analyzed with high-precision methods and tools. These include complex-geometry, 3D, time-dependent, multi-physics predictive simulation methods and software that will play an increasingly important role as the demand for wind energy grows.

Currently most wind-turbine aerodynamics and aeroelasticity simulations are performed using low-fidelity methods, such as the Blade Element Momentum (BEM) theory for the rotor aerodynamics employed in conjunction with simplified structural models of the wind-turbine blades and tower (see, e.g., [1; 2]). These methods are very fast to implement and execute. However, the cases involving unsteady flow, turbulence, 3D details of the wind-turbine blade and tower geometry, and other similarly-important features, are beyond their range of applicability.

To obtain high-fidelity predictive simulation results for wind turbines, 3D modeling is essential. However, simula-

tion of wind turbines at full scale engenders a number of challenges: the flow is fully turbulent, requiring highly accurate methods and increased grid resolution. The presence of fluid boundary layers, where turbulence is created, complicates the situation further. Wind-turbine blades are long and slender structures, with complex distribution of material properties, for which the numerical approach must have good approximation properties and avoid locking. Wind-turbine simulations involve moving and stationary components, and the fluid–structure coupling must be accurate, efficient and robust to preclude divergence of the computations. These explain the current, modest nature of the state-of-the-art in wind-turbine simulations.

Fluid–structure interaction (FSI) simulations at full scale are essential for accurate modeling of wind turbines. The motion and deformation of the wind-turbine blades depend on the wind speed and air flow, and the air flow patterns depend on the motion and deformation of the blades. In order to simulate the coupled problem, the equations governing the air flow and the blade motions and deformations need to be solved simultaneously, with proper kinematic and dynamic conditions coupling the two physical systems. Without that the modeling cannot be realistic: unsteady blade deformation affects aerodynamic efficiency and noise generation, and response to wind gusts. Flutter analysis of large blades operating in offshore environments is of great importance and cannot be accomplished without FSI.

In recent years, several attempts were made to address the above mentioned challenges and to raise the fidelity and predictability levels of wind-turbine simulations. Standalone aerodynamics simulations of wind-turbine configurations in 3D were reported in [3; 4; 5; 6; 7; 8], while standalone structural analyses of rotor blades of complex geometry and material composition, but under assumed wind-load conditions or wind-load conditions coming from separate aerodynamic computations were reported in [9; 10; 11; 12; 13; 14]. In a recent work [15] it was shown that coupled FSI modeling and simulation of wind turbines is important for accurately predicting their mechanical behavior at full scale.

We feel that in order to address the above mentioned challenges one should employ a combination of numerical techniques, which are general, accurate, robust and efficient for the targeted class of problems. Such techniques are summarized in what follows and are described in greater detail in the body of this review paper.

Isogeometric Analysis (IGA), first introduced in [16] and further expanded on in [17; 18; 19; 20; 21; 22; 23; 24; 25; 26; 27; 28], is adopted as the geometry modeling and simulation framework for wind turbines in some of the examples presented in this paper. We use the IGA based on NURBS (non-uniform rational B-splines), which are more efficient than standard finite elements for representing complex, smooth geometries, such as wind-turbine blades. The IGA was successfully employed for computation of turbulent flows [29; 30; 31; 32; 33; 34], nonlinear structures [35; 36; 37; 38; 39; 13], and FSI [40; 41; 42; 43], and, in most cases, gave a clear advantage over standard low-order finite elements in terms of solution accuracy per-degree-of-freedom. This is in part attributable to the higher-order smoothness of the basis functions employed. Flows about rotating components are naturally handled in an isogeometric framework because all conic sections, and in particular, circular and cylindrical shapes, are represented exactly [44].

The blade structure is governed by the isogeometric rotation-free shell formulation with the aid of the bending-strip method [13]. The method is appropriate for thin-shell structures comprised of multiple C^1- or higher-order continuous surface patches that are joined or merged with continuity no greater than C^0. The Kirchhoff–Love shell theory that relies on higher-order continuity of the basis functions is employed in the patch interior as in [39]. Although NURBS-based IGA is employed in this work, other discretizations such as T-splines [24; 23] or subdivision surfaces [45; 46; 47], are perfectly suited for the proposed structural modeling method.

In addition, an isogeometric representation of the analysis-suitable geometry can be used in generating tetrahedral and hexahedral meshes for computations with the finite element method (FEM). In this paper, we use tetrahedral meshes generated that way in wind-turbine computations with the ALE-VMS and ST-VMS methods. The ALE-VMS method [42; 6] is the variational multiscale (VMS) version of the Arbitrary Lagrangian–Eulerian (ALE) method [48]. The VMS components are from the residual-based VMS (RBVMS) method given in [49; 50; 29; 34]. The ST-VMS method [51; 52] is the VMS version of the Deforming-Spatial-Domain/Stabilized Space–Time (DSD/SST) method [53; 54; 55; 56; 57]. Earlier it was called "DSD/SST-VMST" (i.e. the version with the VMS turbulence model) in [51]. For comparison purposes, we also report in this paper results from the original DSD/SST formulation, which was was named "DSD/SST-SUPS" in [51] (i.e. the version with the SUPG/PSPG stabilization), which was also called "ST-SUPS" in [58].

The ALE-VMS method originated from the RBVMS formulation of incompressible turbulent flows proposed in [29] for stationary meshes, and may be thought of as an extension of the RBVMS method to moving meshes. As such, it was presented for the first time in [42] in the context of FSI. Although ALE-VMS gave reasonably good results for several important turbulent flows, it was evident in [29; 32] that to obtain accurate results for wall-bounded turbulent flows the method required relatively fine resolution of the boundary layers. This fact makes ALE-VMS a somewhat costly technology for full-scale wall-bounded turbulent flows at high Reynolds numbers, which are characteristic of the present application. For this reason, weakly-enforced essential boundary condition formulation was introduced in [59], which significantly improved the performance of the ALE-VMS formulation in the presence of unresolved boundary layers [30; 31; 34]. The weak boundary condition formulation may be thought of as an extension of Nitsche's method [60] to the Navier–Stokes equations of incompressible flows. Another interpretation of the weak

boundary condition formulation is that it is a discontinuous Galerkin (DG) method (see, e.g., [61]), where the continuity of the basis functions is enforced everywhere in the domain interior, but not at the domain boundary.

The DSD/SST formulation was introduced in [53; 54; 55] as a general-purpose interface-tracking (moving-mesh) technique for flows with moving boundaries and interfaces, including FSI and flows with moving objects. Its stabilization components are the Streamline-Upwind/Petrov-Galerkin (SUPG) [62] and Pressure-Stabilizing/Petrov-Galerkin (PSPG) [53; 63] stabilizations. It also includes the "LSIC" (least-squares on incompressibility constraint) stabilization. Some of the earliest FSI computations with the DSD/SST formulation were reported in [64] for vortex-induced vibrations of a cylinder and in [65] for flow-induced vibrations of a flexible, cantilevered pipe (1D structure with 3D flow). The DSD/SST formulation has been used extensively in 3D computations of parachute FSI, starting with the 3D computations reported in [66] and evolving to computations with direct coupling [67]. New versions of the DSD/SST formulation introduced in [57] are the core technologies of the Stabilized ST FSI (SSTFSI) technique, which was also introduced in [57]. The ST-VMS method and SSTFSI technique, combined with a number of special techniques (see [68; 69; 70; 71] and references therein) have been used in some of the most challenging parachute FSI computations (see [68; 72; 73; 74] and references therein)), and also in a good number of patient-specific cardiovascular FSI and fluid mechanics computations (see [69; 70; 71; 75] and references therein). Computations with the SSTFSI technique also received a substantial attention in research related to iterative solution of large linear systems [76; 77].

In application of the DSD/SST formulation to flows with moving objects, the Shear–Slip Mesh Update Method (SSMUM) [78; 79; 80] has been very instrumental. The SSMUM was first introduced for computation of flow around two high-speed trains passing each other in a tunnel (see [78]). The challenge was to accurately and efficiently update the meshes used in computations based on the DSD/SST formulation and involving two objects in fast, linear relative motion. The idea behind the SSMUM was to restrict the mesh moving and remeshing to a thin layer of elements between the objects in relative motion. The mesh update at each time step can be accomplished by a "shear" deformation of the elements in this layer, followed by a "slip" in node connectivities. The slip in the node connectivities, to an extent, un-does the deformation of the elements and results in elements with better shapes than those that were shear-deformed. Because the remeshing consists of simply re-defining the node connectivities, both the projection errors and the mesh generation cost are minimized. A few years after the high-speed train computations, the SSMUM was implemented for objects in fast, rotational relative motion and applied to computation of flow past a rotating propeller [79] and flow around a helicopter with its rotor in motion [80].

The ST-VMS method was successfully tested on computation of wind-turbine rotor aerodynamics in [7; 81; 82]. Those computations did not include a wind-turbine tower, and therefore a mesh update method was not required. In [83], the ST-VMS method was applied to computation of wind-turbine rotor and tower aerodynamics. The presence of a tower requires a mesh update method that can handle the fast, rotational relative motion between the rotor and tower. The SSMUM would have been an option, but we decided to use a mesh update method that is more general. We use NURBS basis functions for the temporal representation of the rotor motion, mesh motion and also in remeshing. This is essentially the same computational technology used in the ST-VMS computations of flapping-wing aerodynamics reported in [84; 85; 86; 87]. We named it "ST/NURBS Mesh Update Method (STNMUM)" in [83]. The motion of the rotor surface mesh created from the NURBS geometry is represented by quadratic temporal NURBS basis functions, with sufficient number of temporal patches for one rotation. This enables us to represent the circular paths associated with the rotor motion exactly and, with a "secondary mapping" [51; 84; 52; 58], specify a constant angular velocity corresponding to the invariant speeds along those paths. Given the motion of the surface mesh, we compute meshes that serve as temporal-control points. This is done by creating with an automatic mesh generator a new mesh at the central control point of the temporal patch, and computing the meshes at the other two control points by using the mesh moving technique [88; 89; 90; 91; 57] developed earlier in conjunction with the DSD/SST method. The STNMUM allows us to do mesh computations with longer time in between, but get the mesh-related information for each ST slab, such as the coordinates and their time derivatives, from the temporal representation whenever we need. This approach where the mesh-related information is computed "directly" was called in [83] "Direct Temporal Representation (DTR)." In an alternative approach, we can obtain the mesh-related data after first computing the finite element meshes associated with each ST slab by interpolation from the temporal NURBS representation of the mesh. This approach was called "Interpolated-Mesh Temporal Representation (IMTR)." in [83]. For better mesh resolution, we use layers of thin elements near the blade surfaces. These layers of elements are created with a special mesh generation process and are not part of what we create with the automatic mesh generation process. They undergo rigid-body motion with the rotor. Despite the fast, rotational relative motion between the rotor and tower, the computations reported in [83] were carried out by using an automatic mesh generator only a total of 6 times during an entire computation.

In Section 2, we review the methods for the aerodynamics parts of this work, namely, the ALE-VMS and ST-VMS methods. We also describe, for the ST-VMS computations, a method with NURBS basis functions in temporal representation of the rotor motion with constant angular velocity and a recently-developed element length definition for the diffusion-dominated limits of the stabilization parame-

ters. In addition, for the ALE-VMS computations, we describe the formulation for weakly-enforced essential boundary conditions. In Section 3, we provide a description of the wind-turbine rotor geometry modeling and summarize the aerodynamics computations for the 5MW wind-turbine rotor defined in [2]. The results presented are from reference [6] and include simulations using the FEM-based ST-VMS method. In Section 4, we present the sliding-interface formulation from [44; 92; 93], which enables the simulation of rotor–tower interaction. The formulation was used in [93] for ALE-VMS aerodynamic simulations of the National Renewable Energy Lab (NREL) Phase VI wind turbine (see [94]) for comparison to the extensive set of experimental data available for this test case. We also present those simulations in Section 4. In Section 5, we describe, from [83], the FEM-based ST-VMS computations of the wind-turbine rotor and tower aerodynamics. NURBS basis functions are used in temporal representation of the rotor and volume mesh motion and in remeshing. In Section 6, we review the structural mechanics formulation, which is based on the Kirchhoff–Love thin-shell theory and the bending-strip method (see [39; 13; 15]). The FSI coupling is described in Section 7, where we also provide a discussion of the mesh motion procedures employed in the FSI simulations of the Micon 65/13M wind turbine reported in Section 8, which was reported earlier in [95]. In Section 9, we present a method for pre-bending of wind-turbine blades, which was recently proposed in [14]. We end with concluding remarks in Section 10.

2 ALE-VMS and ST-VMS Formulations of the Navier–Stokes Equations of Incompressible Flows

2.1 Continuous Problem

Let $\Omega_t \in \mathbb{R}^{n_{\text{sd}}}$, $d = 2, 3$, be the spatial domain of the aerodynamics problem with boundary Γ_t at time $t \in (0, T)$. The subscript t indicates that the fluid mechanics spatial domain is time-dependent. The Navier–Stokes equations of incompressible flows [1] in the ALE frame may be written on Ω_t and $\forall t \in (0, T)$ as

$$\rho\left(\left.\frac{\partial \mathbf{u}}{\partial t}\right|_{\hat{x}} + (\mathbf{u} - \hat{\mathbf{u}}) \cdot \boldsymbol{\nabla}\mathbf{u} - \mathbf{f}\right) - \boldsymbol{\nabla} \cdot \boldsymbol{\sigma} = \mathbf{0},$$
$$\boldsymbol{\nabla} \cdot \mathbf{u} = 0, \tag{1}$$

where ρ, $\mathbf{u}$, and $\mathbf{f}$ are the density, velocity and the external force, respectively, and the stress tensor $\boldsymbol{\sigma}$ is defined as

$$\boldsymbol{\sigma}(\mathbf{u}, p) = -p\mathbf{I} + 2\mu\boldsymbol{\varepsilon}(\mathbf{u}). \tag{2}$$

Here p is the pressure, $\mathbf{I}$ is the identity tensor, μ is the dynamic viscosity, and $\boldsymbol{\varepsilon}(\mathbf{u})$ is the strain-rate tensor given by

$$\boldsymbol{\varepsilon}(\mathbf{u}) = \frac{1}{2}\left(\boldsymbol{\nabla}\mathbf{u} + \boldsymbol{\nabla}\mathbf{u}^T\right). \tag{3}$$

[1] Although aerodynamic phenomena are generally described using the Navier–Stokes equations of compressible flows, the incompressible-flow assumption is valid for the present application.

In Eq. (1), the notation $\left|_{\hat{x}}\right.$ implies that the time derivative is taken with respect to a fixed referential-domain spatial coordinates $\hat{\mathbf{x}}$, and $\hat{\mathbf{u}}$ is the velocity of the fluid domain Ω_t. The spatial gradients are taken with respect to the spatial coordinates $\mathbf{x}$ of the current configuration.

2.2 ALE-VMS Method

The ALE-VMS formulation of the continuum aerodynamics formulation is given as follows: find $\mathbf{u}^h \in \mathcal{S}_u^h$ and $p^h \in \mathcal{S}_p^h$, such that $\forall \mathbf{w}^h \in \mathcal{V}_u^h$ and $\forall q^h \in \mathcal{V}_p^h$:

$$\begin{aligned}
&\int_{\Omega_t} \mathbf{w}^h \cdot \rho\left(\left.\frac{\partial \mathbf{u}^h}{\partial t}\right|_{\hat{x}} + \left(\mathbf{u}^h - \hat{\mathbf{u}}^h\right) \cdot \boldsymbol{\nabla}\mathbf{u}^h - \mathbf{f}^h\right) \mathrm{d}\Omega \\
&+ \int_{\Omega_t} \boldsymbol{\varepsilon}\left(\mathbf{w}^h\right) : \boldsymbol{\sigma}\left(\mathbf{u}^h, p^h\right) \mathrm{d}\Omega - \int_{(\Gamma_t)_{\text{h}}} \mathbf{w}^h \cdot \mathbf{h}^h \, \mathrm{d}\Gamma \\
&+ \int_{\Omega_t} q^h \boldsymbol{\nabla} \cdot \mathbf{u}^h \, \mathrm{d}\Omega \\
&+ \sum_{e=1}^{n_{\text{el}}} \int_{\Omega_t^e} \tau_{\text{SUPS}} \left(\left(\mathbf{u}^h - \hat{\mathbf{u}}^h\right) \cdot \boldsymbol{\nabla}\mathbf{w}^h + \frac{\boldsymbol{\nabla} q^h}{\rho}\right) \cdot \mathbf{r}_{\text{M}}\left(\mathbf{u}^h, p^h\right) \mathrm{d}\Omega \\
&+ \sum_{e=1}^{n_{\text{el}}} \int_{\Omega_t^e} \rho \nu_{\text{LSIC}} \boldsymbol{\nabla} \cdot \mathbf{w}^h r_{\text{C}}(\mathbf{u}^h, p^h) \, \mathrm{d}\Omega \\
&- \sum_{e=1}^{n_{\text{el}}} \int_{\Omega_t^e} \tau_{\text{SUPS}} \mathbf{w}^h \cdot \left(\mathbf{r}_{\text{M}}\left(\mathbf{u}^h, p^h\right) \cdot \boldsymbol{\nabla}\mathbf{u}^h\right) \mathrm{d}\Omega \\
&- \sum_{e=1}^{n_{\text{el}}} \int_{\Omega_t^e} \frac{\boldsymbol{\nabla}\mathbf{w}^h}{\rho} : \left(\tau_{\text{SUPS}} \mathbf{r}_{\text{M}}\left(\mathbf{u}^h, p^h\right)\right) \\
&\qquad \otimes \left(\tau_{\text{SUPS}} \mathbf{r}_{\text{M}}\left(\mathbf{u}^h, p^h\right)\right) \mathrm{d}\Omega = 0.
\end{aligned} \tag{4}$$

Here Ω_t is divided into n_{el} spatial finite element subdomains denoted by Ω_t^e. The finite-dimensional trial function spaces $\mathcal{S}_u^h$ for the velocity and $\mathcal{S}_p^h$ for the pressure, as well as the corresponding test function spaces $\mathcal{V}_u^h$ and $\mathcal{V}_p^h$ are assumed to be of equal order. In Eq. (4), $\mathbf{h}$ is the natural boundary condition, $(\Gamma_t)_{\text{h}}$ is the part of the boundary where we specify that natural boundary condition, $\hat{\mathbf{u}}^h$ is the mesh velocity, and $\mathbf{r}_{\text{M}}$ and r_{C} are the residuals of the momentum and continuity (incompressibility constraint) equations, given as

$$\begin{aligned}
&\mathbf{r}_{\text{M}}(\mathbf{u}^h, p^h) = \\
&\rho\left(\left.\frac{\partial \mathbf{u}^h}{\partial t}\right|_{\hat{x}} + \left(\mathbf{u}^h - \hat{\mathbf{u}}^h\right) \cdot \boldsymbol{\nabla}\mathbf{u}^h - \mathbf{f}^h\right) - \boldsymbol{\nabla} \cdot \boldsymbol{\sigma}\left(\mathbf{u}^h, p^h\right)
\end{aligned} \tag{5}$$

and

$$r_{\text{C}}(\mathbf{u}^h, p^h) = \boldsymbol{\nabla} \cdot \mathbf{u}^h. \tag{6}$$

Also in Eq. (4), τ_{SUPS} and ν_{LSIC} are the stabilization parameters defined in [42] as

$$\tau_{\text{SUPS}} = \left(\frac{4}{\Delta t^2} + \left(\mathbf{u}^h - \hat{\mathbf{u}}^h\right) \cdot \mathbf{G}\left(\mathbf{u}^h - \hat{\mathbf{u}}^h\right) + C_I \nu^2 \mathbf{G} : \mathbf{G}\right)^{-1/2} \tag{7}$$

and

$$\nu_{\text{LSIC}} = (\text{tr}\mathbf{G}\ \tau_{\text{SUPS}})^{-1}\,, \tag{8}$$

where

$$\text{tr}\mathbf{G} = \sum_{i=1}^{d} G_{ii} \tag{9}$$

is the trace of the element metric tensor $\mathbf{G}$, Δt is the time-step size, and C_I is a positive constant, independent of the mesh size, derived from an appropriate element-wise inverse estimate (see, e.g., [96; 97; 98]).

Remark 1 *The stabilization parameters τ_{SUPS} and ν_{LSIC} in the above equations originate from stabilized finite element methods for fluid dynamics (see, e.g., [62; 99; 100; 101; 102; 56]). The notation "SUPS," introduced in [51], indicates that there is a single stabilization parameter for the SUPG and PSPG stabilizations, instead of two separate parameters. The notation "LSIC", introduced in [102], denotes the stabilization based on least-squares on the incompressibility constraint. The stabilization parameters were designed and studied extensively in the context of stabilized finite element formulations of linear model problems of direct relevance to fluid mechanics. These model problems include advection–diffusion and Stokes equations. The design of τ_{SUPS} and ν_{LSIC} is such that optimal convergence with respect to the mesh size and polynomial order of discretization is attained for these cases. Furthermore, enhanced stability for advection-dominated flows and the ability to conveniently employ the same basis functions for velocity and pressure variables for incompressible flow are some of the attractive outcomes of this method. More recently, the stabilization parameters were derived in the context of the variational multiscale methods [49; 103] and were interpreted as the appropriate averages of the small-scale Green's function, a key mathematical object in the theory of VMS methods (see [104] for an elaboration).*

Remark 2 *The ALE-VMS formulation is a moving-mesh extension of the RBVMS turbulence modeling technique proposed for stationary meshes in [29]. It was also presented in [42] for moving meshes in the context of FSI. Recently, a VMS version of the DSD/SST formulation, which is called both DSD/SST-VMST and ST-VMS, was introduced in [51; 52] for computations with moving meshes (see next subsection).*

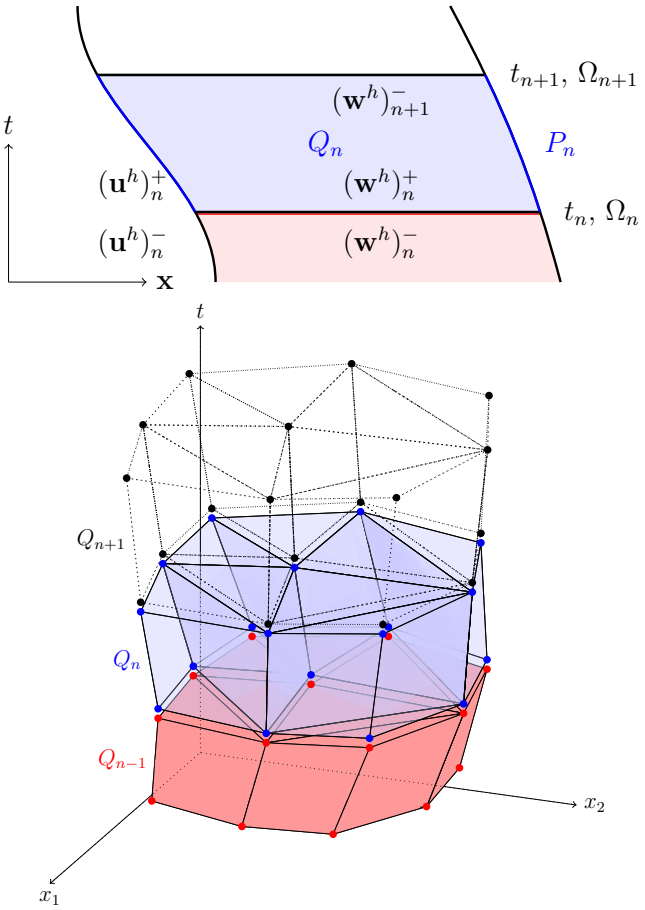


Fig. 1 ST slab in an abstract representation (top) and in a 2D context (bottom).

2.3 ST-VMS Method

In the DSD/SST method (see, e.g., [53; 54; 55; 56; 57; 51; 52]), the finite element formulation is written over a sequence of N ST slabs Q_n, where Q_n is the slice of the ST domain between the time levels t_n and t_{n+1} (see Figure 1). At each time step, the integrations are performed over Q_n. The ST finite element interpolation functions are continuous within a ST slab, but discontinuous from one ST slab to another. The notation $(\cdot)^-_n$ and $(\cdot)^+_n$ will denote the function values at t_n as approached from below and above. Each Q_n is decomposed into elements Q^e_n, where $e = 1, 2, \ldots, (n_{\text{el}})_n$. The subscript n used with n_{el} is for the general case where the number of ST elements may change from one ST slab to another. The essential and natural boundary conditions are enforced over $(P_n)_{\text{g}}$ and $(P_n)_{\text{h}}$, the complementary subsets of the lateral boundary of the ST slab. The finite element trial function spaces $\mathcal{S}^h_u$ for velocity and $\mathcal{S}^h_p$ for pressure, and the test function spaces $\mathcal{V}^h_u$ and $\mathcal{V}^h_p = \mathcal{S}^h_p$ are defined by using, over Q_n, first-order polynomials in space and time. [2]

The conservative form of the ST-VMS method is written as follows: given $(\mathbf{u}^h)^-_n$, find $\mathbf{u}^h \in \mathcal{S}^h_u$ and $p^h \in \mathcal{S}^h_p$, such that

[2] Although the trial and test function spaces for the ALE and ST formulations are different, to avoid introducing extra notation, we use the same symbols to denote these objects in both cases.

$\forall \mathbf{w}^h \in \mathcal{V}_u^h$ and $\forall q^h \in \mathcal{V}_p^h$:

$$\begin{aligned}
&\int_{Q_n} \mathbf{w}^h \cdot \rho \left(\frac{\partial \mathbf{u}^h}{\partial t} + \boldsymbol{\nabla} \cdot (\mathbf{u}^h \mathbf{u}^h) - \mathbf{f}^h \right) \mathrm{d}Q \\
&+ \int_{Q_n} \boldsymbol{\varepsilon}(\mathbf{w}^h) : \boldsymbol{\sigma}(\mathbf{u}^h, p^h) \mathrm{d}Q - \int_{(P_n)_\mathrm{h}} \mathbf{w}^h \cdot \mathbf{h}^h \mathrm{d}P \\
&+ \int_{Q_n} q^h \boldsymbol{\nabla} \cdot \mathbf{u}^h \mathrm{d}Q + \int_{\Omega_n} (\mathbf{w}^h)_n^+ \cdot \rho \left((\mathbf{u}^h)_n^+ - (\mathbf{u}^h)_n^- \right) \mathrm{d}\Omega \\
&+ \sum_{e=1}^{(n_\mathrm{el})_n} \int_{Q_n^e} \frac{\tau_\mathrm{SUPS}}{\rho} \left[\rho \left(\frac{\partial \mathbf{w}^h}{\partial t} + \mathbf{u}^h \cdot \boldsymbol{\nabla} \mathbf{w}^h \right) + \boldsymbol{\nabla} q^h \right] \cdot \mathbf{r}_\mathrm{M} \mathrm{d}Q \\
&+ \sum_{e=1}^{(n_\mathrm{el})_n} \int_{Q_n^e} \rho \nu_\mathrm{LSIC} \boldsymbol{\nabla} \cdot \mathbf{w}^h r_\mathrm{C} \mathrm{d}Q \\
&+ \sum_{e=1}^{(n_\mathrm{el})_n} \int_{Q_n^e} \tau_\mathrm{SUPS} \mathbf{r}_\mathrm{M} \cdot \left(\boldsymbol{\nabla} \mathbf{w}^h \right) \mathbf{u}^h \mathrm{d}Q \\
&- \sum_{e=1}^{(n_\mathrm{el})_n} \int_{Q_n^e} \frac{\tau_\mathrm{SUPS}^2}{\rho} \mathbf{r}_\mathrm{M} \cdot \left(\boldsymbol{\nabla} \mathbf{w}^h \right) \mathbf{r}_\mathrm{M} \mathrm{d}Q = 0. \qquad (10)
\end{aligned}$$

The stabilization parameters τ_SUPS and ν_LSIC can be calculated using the definitions given by Eqs. (7)–(9). However, the definitions that are commonly used with the DSD/SST formulation are those given in [56]:

$$\tau_\mathrm{SUPS} = \left(\frac{1}{\tau_\mathrm{SUGN12}^2} + \frac{1}{\tau_\mathrm{SUGN3}^2} \right)^{-\frac{1}{2}}, \qquad (11)$$

$$\tau_\mathrm{SUGN12} = \left(\sum_{a=1}^{n_{en}} \left| \frac{\partial N_a}{\partial t} + \mathbf{u}^h \cdot \boldsymbol{\nabla} N_a \right| \right)^{-1}, \qquad (12)$$

$$\tau_\mathrm{SUGN3} = \frac{h_\mathrm{RGN}^2}{4\nu}, \qquad (13)$$

$$h_\mathrm{RGN} = 2 \left(\sum_{a=1}^{n_{en}} |\mathbf{r} \cdot \boldsymbol{\nabla} N_a| \right)^{-1}, \qquad (14)$$

$$\mathbf{r} = \frac{\boldsymbol{\nabla} \|\mathbf{u}^h\|}{\| \boldsymbol{\nabla} \|\mathbf{u}^h\| \|}, \qquad (15)$$

and in [57]:

$$\nu_\mathrm{LSIC} = \tau_\mathrm{SUPS} \, \|\mathbf{u}^h - \hat{\mathbf{u}}^h\|^2, \qquad (16)$$

where n_{en} is the number of (ST) element nodes, and N_a is the ST shape function associated with the ST node a. As an alternative to the construction of τ_SUPS as given by Eqs. (11)–(15), another option was introduced in [57]. In that option, τ_SUPS is constructed based on separate definitions for the advection-dominated and transient-dominated limits:

$$\tau_\mathrm{SUPS} = \left(\frac{1}{\tau_\mathrm{SUGN1}^2} + \frac{1}{\tau_\mathrm{SUGN2}^2} + \frac{1}{\tau_\mathrm{SUGN3}^2} \right)^{-\frac{1}{2}}, \qquad (17)$$

$$\tau_\mathrm{SUGN1} = \left(\sum_{a=1}^{n_{en}} \left| \left(\mathbf{u}^h - \hat{\mathbf{u}}^h \right) \cdot \boldsymbol{\nabla} N_a \right| \right)^{-1}, \qquad (18)$$

$$\tau_\mathrm{SUGN2} = \frac{\Delta t}{2}. \qquad (19)$$

It was noted in [57] that separating τ_SUGN12 into its advection- and transient-dominated components as given by Eqs. (18)–(19) is equivalent to excluding the $\frac{\partial N_a}{\partial t}\big|_{\boldsymbol{\xi}}$ part of $\frac{\partial N_a}{\partial t}$ in Eq. (12), making that the definition for τ_SUGN1, and accounting for $\frac{\partial N_a}{\partial t}\big|_{\boldsymbol{\xi}}$ in the definition for τ_SUGN2 given by Eq. (19). Here $\boldsymbol{\xi}$ is the vector of element coordinates, and $\frac{\partial}{\partial t}\big|_{\boldsymbol{\xi}}$ is equivalent to $\frac{\partial}{\partial t}\big|_{\hat{\mathbf{x}}}$. Both notations for the same partial derivative are kept in the interest of backward compatibility with prior articles. For more ways of calculating τ_SUPS and ν_LSIC, see [102; 56; 105; 106; 107; 108; 109; 110; 33; 111; 112; 113].

Remark 3 *The 6th and 7th terms of the ST-VMS method, given by Eq. (10), are the SUPG/PSPG and LSIC stabilization terms, respectively. If we exclude the last two terms in Eq. (10), the method reduces to the original DSD/SST method (with the advection term retained in the conservation-law form) under the condition $\tau_\mathrm{PSPG} = \tau_\mathrm{SUPG}$. This original method is now called both DSD/SST-SUPS and ST-SUPS.*

For completeness, we also provide here the ST-SUPS method (from [56]): given $(\mathbf{u}^h)_n^-$, find $\mathbf{u}^h \in \mathcal{S}_u^h$ and $p^h \in \mathcal{S}_p^h$, such that $\forall \mathbf{w}^h \in \mathcal{V}_u^h$ and $\forall q^h \in \mathcal{V}_p^h$:

$$\begin{aligned}
&\int_{Q_n} \mathbf{w}^h \cdot \rho \left(\frac{\partial \mathbf{u}^h}{\partial t} + \mathbf{u}^h \cdot \boldsymbol{\nabla} \mathbf{u}^h - \mathbf{f}^h \right) \mathrm{d}Q \\
&+ \int_{Q_n} \boldsymbol{\varepsilon}(\mathbf{w}^h) : \boldsymbol{\sigma}(\mathbf{u}^h, p^h) \mathrm{d}Q - \int_{(P_n)_\mathrm{h}} \mathbf{w}^h \cdot \mathbf{h}^h \mathrm{d}P \\
&+ \int_{Q_n} q^h \boldsymbol{\nabla} \cdot \mathbf{u}^h \mathrm{d}Q + \int_{\Omega_n} (\mathbf{w}^h)_n^+ \cdot \rho \left((\mathbf{u}^h)_n^+ - (\mathbf{u}^h)_n^- \right) \mathrm{d}\Omega \\
&+ \sum_{e=1}^{(n_\mathrm{el})_n} \int_{Q_n^e} \frac{1}{\rho} \left[\tau_\mathrm{SUPG} \rho \left(\frac{\partial \mathbf{w}^h}{\partial t} + \mathbf{u}^h \cdot \boldsymbol{\nabla} \mathbf{w}^h \right) + \tau_\mathrm{PSPG} \boldsymbol{\nabla} q^h \right] \cdot \mathbf{r}_\mathrm{M} \mathrm{d}Q \\
&+ \sum_{e=1}^{(n_\mathrm{el})_n} \int_{Q_n^e} \rho \nu_\mathrm{LSIC} \boldsymbol{\nabla} \cdot \mathbf{w}^h r_\mathrm{C} \mathrm{d}Q = 0. \qquad (20)
\end{aligned}$$

Remark 4 *One of the main differences between the ALE and ST forms of the VMS method is that the ST form retains the fine-scale time derivative term $\frac{\partial \mathbf{u}'}{\partial t}\big|_{\boldsymbol{\xi}}$. Dropping this term is called the "quasi-static" assumption (see [6] for the terminology). This is the same as the "WTSE" option in the DSD/SST formulation (see Remark 2 of [57]). We believe that this makes a significant difference, especially when the polynomial orders in space or time are higher (see [51]).*

Remark 5 *With the function spaces defined in the paragraph preceding Eq. (10), for each ST slab velocity and pressure assume double unknown values at each spatial node. One value corresponds to the lower end of the slab, and the other one the upper end. In [57], the option of using double unknown values at a spatial node is called "DV" for velocity and "DP" for pressure. In this case, as pointed out in [57], we use two integration points over the time interval of the ST slab, and this time-itegration option is called "TIP2". This version of the DSD/SST formulation, with the options set DV, DP and TIP2, is called "DSD/SST-DP".*

2.4 Rotation Representation with Constant Angular Velocity

This subsection, which is related to the ST-VMS computations, is from [83]. We use quadratic NURBS functions, as described in [51; 84; 52; 58], to represent a circular arc. We discretize time and position as follows:

$$t = \sum_{\alpha=1}^{n_{\text{ent}}} T^{\alpha}(\Theta_t(\theta)) t^{\alpha}, \tag{21}$$

$$\mathbf{x} = \sum_{\alpha=1}^{n_{\text{ent}}} T^{\alpha}(\Theta_x(\theta)) \mathbf{x}^{\alpha}. \tag{22}$$

Here n_{ent} is the number of temporal element nodes, T^{α} is the basis function, $\Theta_t(\theta)$ and $\Theta_x(\theta)$ are the secondary mappings for time and position, and t^{α} and $\mathbf{x}^{\alpha}$ are the time and position values corresponding to the basis function T^{α}. The basis functions could be finite element or NURBS basis functions. For the circular arc, $n_{\text{ent}} = 3$ and they are quadratic NURBS. The secondary mapping concept above was introduced in [51], and the velocity can be expressed as follows:

$$\frac{\mathrm{d}\mathbf{x}}{\mathrm{d}t} = \left(\sum_{\alpha=1}^{n_{\text{ent}}} \frac{\mathrm{d}T^{\alpha}}{\mathrm{d}\Theta_x} \frac{\mathrm{d}\Theta_x}{\mathrm{d}\theta} \mathbf{x}^{\alpha} \right) \left(\sum_{\alpha=1}^{n_{\text{ent}}} \frac{\mathrm{d}T^{\alpha}}{\mathrm{d}\Theta_t} \frac{\mathrm{d}\Theta_t}{\mathrm{d}\theta} t^{\alpha} \right)^{-1}, \tag{23}$$

leading to

$$\frac{\mathrm{d}\mathbf{x}}{\mathrm{d}t} = \left(\sum_{\alpha=1}^{n_{\text{ent}}} \frac{\mathrm{d}T^{\alpha}}{\mathrm{d}\Theta_x} \mathbf{x}^{\alpha} \right) \left(\sum_{\alpha=1}^{n_{\text{ent}}} \frac{\mathrm{d}T^{\alpha}}{\mathrm{d}\Theta_t} t^{\alpha} \right)^{-1} \left(\frac{\mathrm{d}\Theta_x}{\mathrm{d}\theta} \frac{\mathrm{d}\theta}{\mathrm{d}\Theta_t} \right). \tag{24}$$

Thus, the speed along the path can be specified only by modifying the secondary mapping. For a circular arc, two methods were introduced in [84; 52] and also described in [58]; one is modifying the secondary mapping for position and the other one is modifying both such that $\frac{\mathrm{d}t}{\mathrm{d}\theta}$ is constant. We note that, in theory, the secondary mapping selections do not make any difference as long as the relationship $\frac{\mathrm{d}\Theta_x}{\mathrm{d}\Theta_t}$ is the same.

In our implementation, to keep the process general, we search for the parametric coordinate θ by using an iterative solution method [84; 52; 58]. We use the latter set of the secondary mappings, having constant $\frac{\mathrm{d}t}{\mathrm{d}\theta}$.

Remark 6 *When we use a secondary mapping for discretization of unknowns, the selection of the mappings affects the numerical integration accuracy in the physical domain.*

For the IMTR, we find the parametric coordinate corresponding to each time level and interpolate the position to obtain the corresponding mesh. For the DTR, we first calculate time corresponding to each integration point, including the time step size because of the jump term, and then calculate Θ_x and Θ_t to interpolate the position and velocity from Eqs. (22) and (24).

2.5 Element Length Definition for the Diffusion-Dominated Limit

This subsection, which is related to the ST-VMS computations, is from [83]. The element length definition for the diffusion-dominated limit is used in calculating the diffusion-dominated limit of the stabilization parameters τ_{SUPG}, τ_{PSPG} and τ_{SUPS}, and, directly or indirectly (through τ_{SUPS}), in calculating all options of ν_{LSIC} except for the "TGI" option, which is given by Eq. (8). That includes the option that was defined in [81] as a component of the "LHC" option, which was named in [58] "HRGN":

$$\nu_{\text{LSIC–HRGN}} = \frac{h_{\text{RGN}}^2}{\tau_{\text{SUPS}}}. \tag{25}$$

In the expression for $\mathbf{r}$ given by Eq. (15), $\boldsymbol{\nabla}\|\mathbf{u}^h\|$ is calculated as

$$\boldsymbol{\nabla}\|\mathbf{u}^h\| = \boldsymbol{\nabla}\left(\left\|\mathbf{u}^h\right\|^2)^{\frac{1}{2}} \right) = \frac{1}{2\|\mathbf{u}^h\|} \boldsymbol{\nabla}\left(\left\|\mathbf{u}^h\right\|^2 \right), \tag{26}$$

$$\boldsymbol{\nabla}\|\mathbf{u}^h\| = \boldsymbol{\nabla}\mathbf{u}^h \cdot \frac{\mathbf{u}^h}{\|\mathbf{u}^h\|}, \tag{27}$$

resulting in

$$\mathbf{r} = \frac{\boldsymbol{\nabla}\mathbf{u}^h \cdot \mathbf{u}^h}{\|\boldsymbol{\nabla}\mathbf{u}^h \cdot \mathbf{u}^h\|}. \tag{28}$$

This expression becomes ill defined when $\boldsymbol{\nabla}\mathbf{u}^h \cdot \mathbf{u}^h = \mathbf{0}$.

We introduce a new element length definition for the diffusion-dominated limit:

$$h_{\text{RGNT}} = \left(\sum_{i=1}^{n} w_i^2 \frac{1}{h_i^2} \right)^{-\frac{1}{2}}, \tag{29}$$

where $h_i > 0$ is element length for the i^{th} direction, $w_i \geq 0$ is the weight for that direction, n is the number of directions, and

$$\sum_{i=1}^{n} w_i = 1. \tag{30}$$

Equation (29) is well defined if the element length for at least one of the directions is nonzero.

We define the element length and weight for each of the n directions based on an $n_{\text{sd}} \times n$ tensor $\mathbf{R}$:

$$h_i = 2\,\|\mathbf{R}_i\| \left(\sum_{a=1}^{n_{\text{en}}} |\mathbf{R}_i \cdot \boldsymbol{\nabla} N_a| \right)^{-1}, \tag{31}$$

$$w_i = \frac{\|\mathbf{R}_i\|}{\|\mathbf{R}\|}, \tag{32}$$

where $\mathbf{R}_i$ is the i^{th} column vector of $\mathbf{R}$, the vector norm is L^2, and the matrix norm is Frobenius. From Eqs. (29), (31) and (32) we obtain

$$h_{\text{RGNT}} = 2\,\|\mathbf{R}\|\left(\sum_{i=1}^{n}\left(\sum_{a=1}^{n_{\text{en}}}|\mathbf{R}_i\cdot\boldsymbol{\nabla}N_a|\right)^2\right)^{-\frac{1}{2}}, \tag{33}$$

which is well defined if at least one of the column vectors is nonzero. We propose to define the tensor $\mathbf{R}$ as

$$\mathbf{R}_i = \boldsymbol{\nabla}u_i, \tag{34}$$

with $n = n_{\text{sd}}$. If all column vectors are zero, which implies uniformness in the flow field, to make the expression for h_{RGNT} well defined, we define $\mathbf{R}$ as

$$\mathbf{R}_i = \boldsymbol{\nabla}N_i, \tag{35}$$

with $n = n_{\text{en}}$. The following is a brief justification for Eq. (35). Suppose where there is uniformness in the flow field we change only one coefficient, $(u_b)_i$, which means that the solution gradient is in the $\boldsymbol{\nabla}N_b$ direction. Therefore, it is reasonable to assume that there is an equal chance of having a solution gradient in all $\boldsymbol{\nabla}N_b$ directions. Therefore we use all those directions as the column vectors of $\mathbf{R}$. As an alternative to the definition given by Eq. (35), we propose

$$\mathbf{R}_i = \boldsymbol{\nabla}N_i - \left(\frac{\mathbf{u}^h}{\|\mathbf{u}^h\|}\cdot\boldsymbol{\nabla}N_i\right)\frac{\mathbf{u}^h}{\|\mathbf{u}^h\|}, \tag{36}$$

and this is based on the assumption that there is an equal chance of having a solution gradient in all $\boldsymbol{\nabla}N_b$ directions perpendicular to $\mathbf{u}^h$. In this alternative option, $\mathbf{u}^h = \mathbf{0}$ would revert the definition back to the one given by Eq. (35).

With the new element length h_{RGNT} for the diffusion-dominated limit, the HRGN option of ν_{LSIC} becomes

$$\nu_{\text{LSIC–HRGN}} = \frac{h_{\text{RGNT}}^2}{\tau_{\text{SUPS}}}. \tag{37}$$

2.6 Weakly-Enforced Essential Boundary Conditions

In this section we state the formulation of the weakly-enforced essential boundary conditions. This was first proposed in [59] for the advection–diffusion equation and Navier–Stokes equations of incompressible flows in an effort to improve the accuracy of stabilized and multiscale formulations in the presence of unresolved boundary layers. In [30; 31; 34], the method for the weakly-enforced boundary condition was further refined and studied in a set of challenging wall-bounded turbulent flows. Here, we apply the method to the aerodynamics of wind turbines at full spatial scale.

To account for the weak enforcement of the essential boundary conditions, we remove them from the trial and test function sets $\mathcal{S}_u^h$ and $\mathcal{V}_u^h$, and add the following terms to the left-hand-side of Eq. (4):

$$\begin{aligned}
&-\sum_{b=1}^{n_{\text{eb}}}\int_{\Gamma_t^b\cap(\Gamma_t)_{\text{g}}}\mathbf{w}^h\cdot\boldsymbol{\sigma}\left(\mathbf{u}^h,p^h\right)\mathbf{n}\,\mathrm{d}\Gamma\\
&-\sum_{b=1}^{n_{\text{eb}}}\int_{\Gamma_t^b\cap(\Gamma_t)_{\text{g}}}\left(2\mu\boldsymbol{\varepsilon}\left(\mathbf{w}^h\right)\mathbf{n}+q^h\mathbf{n}\right)\cdot\left(\mathbf{u}^h-\mathbf{g}^h\right)\,\mathrm{d}\Gamma\\
&-\sum_{b=1}^{n_{\text{eb}}}\int_{\Gamma_t^b\cap(\Gamma_t)_{\text{g}}^-}\mathbf{w}^h\cdot\rho\left(\left(\mathbf{u}^h-\hat{\mathbf{u}}^h\right)\cdot\mathbf{n}\right)\left(\mathbf{u}^h-\mathbf{g}^h\right)\,\mathrm{d}\Gamma\\
&+\sum_{b=1}^{n_{\text{eb}}}\int_{\Gamma_t^b\cap(\Gamma_t)_{\text{g}}}\tau_{\text{TAN}}^B\left(\mathbf{w}^h-\left(\mathbf{w}^h\cdot\mathbf{n}\right)\mathbf{n}\right)\cdot\\
&\qquad\qquad\left(\left(\mathbf{u}^h-\mathbf{g}^h\right)-\left(\left(\mathbf{u}^h-\mathbf{g}^h\right)\cdot\mathbf{n}\right)\mathbf{n}\right)\,\mathrm{d}\Gamma\\
&+\sum_{b=1}^{n_{\text{eb}}}\int_{\Gamma_t^b\cap(\Gamma_t)_{\text{g}}}\tau_{\text{NOR}}^B\left(\mathbf{w}^h\cdot\mathbf{n}\right)\left(\left(\mathbf{u}^h-\mathbf{g}^h\right)\cdot\mathbf{n}\right)\,\mathrm{d}\Gamma.
\end{aligned} \tag{38}$$

Here $(\Gamma_t)_{\text{g}}$ is the part of the boundary where the velocity boundary condition $\mathbf{g}$ is set and $\mathbf{n}$ is the unit normal vector. The boundary $(\Gamma_t)_{\text{g}}$ is decomposed into n_{eb} surface elements denoted by Γ_t^b, and $(\Gamma_t)_{\text{g}}^-$ is defined as the "inflow" part of $(\Gamma_t)_{\text{g}}$:

$$(\Gamma_t)_{\text{g}}^- = \left\{\mathbf{x}\;\middle|\;\left(\mathbf{u}^h-\hat{\mathbf{u}}^h\right)\cdot\mathbf{n}<0,\ \forall\mathbf{x}\subset(\Gamma_t)_{\text{g}}\right\}. \tag{39}$$

If $(\Gamma_t)_{\text{g}}$ coincides with the moving wall (rigid or flexible), then $\mathbf{g}$ is the prescribed wall velocity.

The term in the first line is the so-called consistency term. It is necessary to ensure that the discrete formulation is identically satisfied by the exact solution of the Navier–Stokes equations, which, in turn, has implications on the accuracy of the discrete formulation. Also note that this term cancels with the contributions coming from the integration-by-parts of the stress terms in Eq. (4), thus correctly removing traction boundary conditions from the no-slip boundary. The term in the second line is the so-called adjoint consistency term. Its role is less intuitive, as it ensures that the analytical solution of the adjoint equations, when introduced in place of the linear momentum and continuity equation test functions, also satisfies the discrete formulation. Adjoint consistency is linked to optimal convergence of the discrete solution in lower-order norms (see, e.g., [61]). The term in the third line leads to better satisfaction of the inflow boundary conditions. The last two terms are penalty-like, in that they penalize the deviation of the discrete solution from its prescribed value at the boundary. These terms are necessary to ensure the stability (or coercivity) of the discrete formulation, which may be lost due to the introduction of the consistency and adjoint consistency terms.

The weak boundary condition formulation is numerically stable if

$$\tau_{\text{TAN}}^B = \tau_{\text{NOR}}^B = \frac{C_I^B\mu}{h_n}, \tag{40}$$

where h_n is the wall-normal element size, and C_I^B is a sufficiently large positive constant computed from an appropriate element-level inverse estimate (see, e.g., [96; 97; 98]). The constant C_I^B depends on the space dimension d, the element type (tetrahedron, hexahedron, etc.), and the polynomial order of the finite element approximation. For a linear tetrahedron, it is sufficient to take $4.0 \leq C_I^B \leq 8.0$ to obtain a stable discrete solution. The wall-normal element size may be computed from the element metric tensor:

$$h_n = (\mathbf{n} \cdot \mathbf{G}\mathbf{n})^{1/2}. \tag{41}$$

Remark 7 *Rather than setting the no-slip boundary conditions exactly, the weak boundary condition formulation gives the no-slip solution only in the limit as $h_n \to 0$. As a result, coarse discretizations do not need to struggle to resolve the boundary layers; the flow simply slips on the solid boundary. Because of this added flexibility, the weak boundary condition enforcement tends to produce more accurate results on meshes that are too coarse to capture the boundary layer solution. However, as the mesh is refined to capture the boundary layer, the weak and strong boundary condition formulations produce nearly identical results (see [30]).*

Remark 8 *Although the weak boundary condition formulation is also stable for very large values of C_I^B, we do not favor that. Large values of C_I^B place a heavy penalization on the no-slip condition, and the above mentioned flexibility of the method is lost together with the associated accuracy benefits. We favor using a C_I^B that is just large enough to guarantee the stability of the discrete formulation.*

Remark 9 *In reference [30], a connection was identified between the weakly-enforced boundary conditions and wall functions. The latter are commonly employed in conjunction with RANS formulations of turbulent flows (see, e.g., [114; 115]). In the case of wall function formulation, a no-slip boundary condition is replaced with a tangential traction boundary condition, where the traction direction is given by that of the local slip velocity, and the traction magnitude is computed by invoking the "law-of-the-wall". This is an empirical relationship between the flow speed and the normal distance to the wall, both appropriately normalized (see, e.g., [114]). The penalty parameter τ_{TAN}^B may be defined as*

$$\tau_{TAN}^B = \frac{\rho u^{*2}}{||\mathbf{u}_{TAN}^h||}, \tag{42}$$

where $\mathbf{u}_{TAN}^h = \left(\left(\mathbf{u}^h - \mathbf{g}^h\right) - \left(\left(\mathbf{u}^h - \mathbf{g}^h\right) \cdot \mathbf{n}\right)\mathbf{n}\right)$ is the tangential slip velocity, and u^* is the so-called friction velocity, which, among other factors, depends on the magnitude of the slip velocity, and is computed from the law-of-the-wall formula by nonlinear iterations. It was shown in [30], however, that when the boundary layer mesh is fine enough, τ_{TAN}^B from Eq. (42) is independent of the local flow solution, and reverts to the definition given by Eq. (40). This fact is remarkable in that Eq. (40) is purely based on considerations of numerical stability, while Eq. (42) derives from the physics of wall-bounded turbulent flows. In our limited experience, both the "numerics-based" and "physics-based" definitions of the penalty parameter τ_{TAN}^B give very similar results.

3 Aerodynamic Simulations of a 5MW Wind-Turbine Rotor

In this section we begin with a careful definition of the 5MW wind-turbine rotor geometry. We then present the NURBS-based and FEM-based simulations of the wind-turbine rotor. In this section, we only present pure aerodynamic simulations. Structural and FSI modeling and simulations will be presented in the later sections.

3.1 5MW Wind-Turbine Rotor Geometry Definition

As a first step we construct a template for the structural model of the rotor. Here, the structural model is limited to a surface (shell) representation of the wind-turbine blade, the hub, and their attachment zone. The blade surface is assumed to be composed of a collection of airfoil shapes that are lofted in the blade axis direction.

The geometry of the rotor blade is based on the NREL 5MW offshore baseline wind-turbine described in [2]. The blade geometry data taken from the reference is summarized in Table 1.

RNodes	AeroTwst	Chord	AeroCent	AeroOrig	Airfoil
2.0000	0.000	3.542	0.2500	0.50	Cylinder
2.8667	0.000	3.542	0.2500	0.50	Cylinder
5.6000	0.000	3.854	0.2218	0.44	Cylinder
8.3333	0.000	4.167	0.1883	0.38	Cylinder
11.7500	13.308	4.557	0.1465	0.30	DU40
15.8500	11.480	4.652	0.1250	0.25	DU35
19.9500	10.162	4.458	0.1250	0.25	DU35
24.0500	9.011	4.249	0.1250	0.25	DU30
28.1500	7.795	4.007	0.1250	0.25	DU25
32.2500	6.544	3.748	0.1250	0.25	DU25
36.3500	5.361	3.502	0.1250	0.25	DU21
40.4500	4.188	3.256	0.1250	0.25	DU21
44.5500	3.125	3.010	0.1250	0.25	NACA64
48.6500	2.310	2.764	0.1250	0.25	NACA64
52.7500	1.526	2.518	0.1250	0.25	NACA64
56.1667	0.863	2.313	0.1250	0.25	NACA64
58.9000	0.370	2.086	0.1250	0.25	NACA64
61.6333	0.106	1.419	0.1250	0.25	NACA64
62.9000	0.000	0.700	0.1250	0.25	NACA64

Table 1 Wind-turbine rotor geometry definition with "RNodes" (m), "AeroTwst" (°), "Chord" (m), "AeroCent" (-), "AeroOrig", and "Airfoil" type.

A 61 m blade is attached to a hub with radius of 2 m, which gives the total rotor radius of 63 m. The blade is composed of several airfoil types provided in the rightmost column of the table. The first portion of the blade is a perfect cylinder. Further away from the root the cylinder is smoothly blended into a series of DU (Delft University) airfoils. At the

44.55 m location away from the root the NACA64 profile is used to define the blade all the way to the tip (see Figures 2 and 3). The remaining parameters from Table 1 are defined

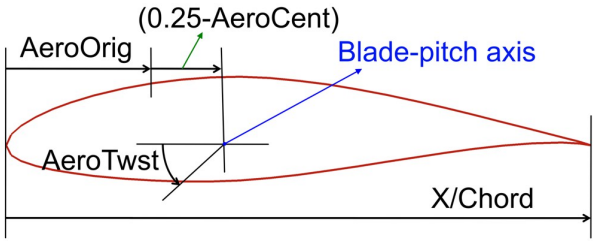


Fig. 2 Illustration of quantities at a cross-section from Table 1.

in Figures 2 and 3: "RNodes" is the distance from the rotor center to the airfoil cross-section in the blade axis direction. "AeroTwst" is the twist angle for a given cross-section. The blades are twisted to enhance the aerodynamic performance. "Chord" is the chord length of the airfoil. "AeroOrig" is the location of the aerodynamic center. For most of the blade airfoil cross-sections, the aerodynamic center is taken at 25% of the chord length from the leading edge. To accommodate the cylindrical shape at the root, the aerodynamic center is gradually moved to 50% of the chord length. This is not reported in [2], but mentioned in [116].

Remark 10 *There is some redundancy in the parameters given in Table 1. The variable "AeroCent" is used as an input to FAST [1], which is the aerodynamics modeling software that is typically used for wind-turbine rotor computations. FAST assumes that the blade-pitch axis passes through each airfoil section at 25% chord length, and defines AeroCent* $-$ *0.25 to be the fractional distance to the aerodynamic center from the blade-pitch axis along the chordline, positive toward the trailing edge. Therefore, AeroOrig* $+ (0.25 - AeroCent)$ *gives the location of where the blade-pitch axis passes through each airfoil cross-section. Although for our purposes this added complexity is unnecessary, the same naming system is used for backward compatibility with the referenced reports.*

For each blade cross-section, we use quadratic NURBS to represent the 2D airfoil shape. The weights of the NURBS functions are set to unity. The weights are adjusted near the root to represent the circular cross-sections of the blade exactly. The cross-sections are lofted in the blade axis direction, also using quadratic NURBS and unity weights. This geometry modeling procedure produces a smooth rotor blade surface using a relatively small number of input parameters, which is an advantage of the isogeometric representation. Figure 4 shows a top view of the blade in which the twisting of the cross-sections is evident.

To carry out the simulations, rotationally-periodic boundary conditions must be imposed. Denoting by $\mathbf{u}_l^h$ and $\mathbf{u}_r^h$ the discrete fluid velocities at the left and right boundary,

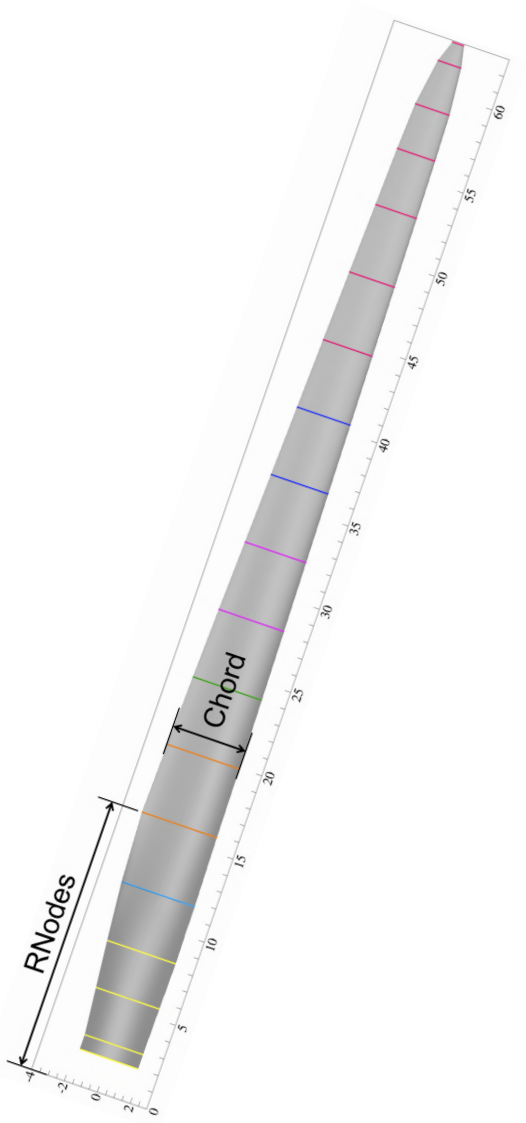


Fig. 3 Illustration of quantities on the blade surface from Table 1.

respectively (see Figure 5), and by p_l^h and p_r^h the corresponding pressures, we set

$$p_l^h = p_r^h, \tag{43}$$

$$\mathbf{u}_l^h = \mathbf{R}(2/3\pi)\, \mathbf{u}_r^h, \tag{44}$$

where $\mathbf{R}(2/3\pi)$ is the rotation matrix evaluated at $\alpha = 2/3\pi$. That is, while the pressure degrees-of-freedom take on the same values, the fluid velocity degrees-of-freedom are related through a linear transformation corresponding to a rotation by $2/3\pi$ radians. Note that the transformation matrix is independent of the current domain position. Rotationally-periodic boundary conditions are implemented through standard master-slave relationships. We note that rotationally-periodic boundary conditions were employed earlier in [117; 118; 119] for parachute simulations.

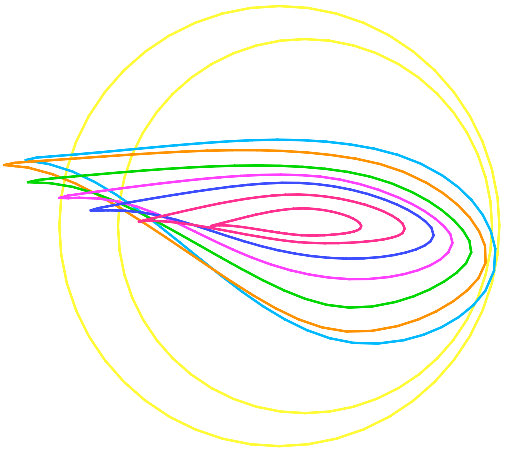


Fig. 4 Top view of a subset of the airfoil cross-sections illustrating blade twisting.

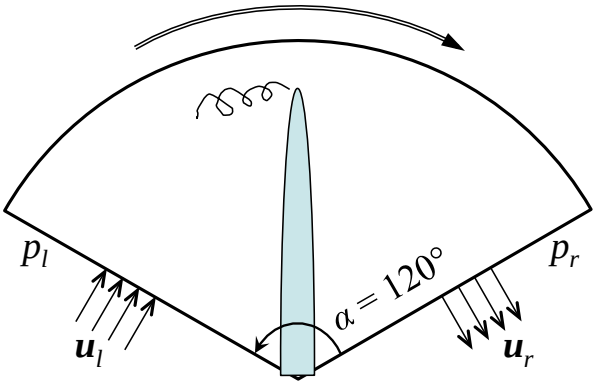


Fig. 5 Rotationally-periodic boundary conditions.

We compute the aerodynamics of the wind-turbine rotor with prescribed speed using a rotating mesh. The wind speed is uniform at 9 m/s and the rotor speed is 1.08 rad/s, giving a tip speed ratio of 7.55 (see [120] for wind-turbine terminology). We use air properties at standard sea-level conditions. The Reynolds number (based on the cord length at $\frac{3}{4}R$ and the relative velocity there) is approximately 12 million. At the inflow boundary the velocity is set to the wind velocity, at the outflow boundary the stress vector is set to zero, and at the radial boundary the radial component of the velocity is set to zero. We start from a flow field where the velocity is equal to the inflow velocity everywhere in the domain except on the rotor surface, where the velocity matches the rotor velocity.

The chosen wind velocity and rotor speed correspond to one of the cases given in [2], where the aerodynamics simulations were performed using FAST [1]. We note that FAST is based on look-up tables for airfoil cross-sections, which give planar, steady-state lift and drag data for a given wind speed and angle of attack. The effects of trailing edge turbulence, hub, and tip are incorporated through empirical models. It was reported in [2] that at these wind conditions and rotor speed, no blade pitching takes place and the rotor develops a favorable aerodynamic torque (i.e., torque in the direction of the rotation) of 2500 kN m. Although this value is used for comparison with our simulations, the exact match is not expected, as our computational modeling is very different than the one in [2]. Nevertheless, we feel that this value of the aerodynamic torque is close to what is expected in reality, given the vast experience of NREL with wind-turbine rotor simulations employing FAST.

3.2 ST-VMS Computations with Finite Elements

This subsection is from [81]. We compute the problem with the DSD/SST method and linear finite elements. To generate the triangular mesh on the rotor surface, we started with a quadrilateral surface mesh generated by interpolating the NURBS geometry of the rotor at each knot intersection. We subdivided each quadrilateral element into triangles and then made minor modifications to improve the mesh quality near the hub. We use three different meshes: Mesh-2, Mesh-3 and Mesh-4, with the surface mesh refined along the blade 2, 3 and 4 times, respectively, compared to the finite element mesh used in [6]. The number of nodes and elements for each blade surface mesh is shown in Table 2, and Figure 6 shows the surface mesh for Mesh-4. For computational efficiency, rotational-periodicity [117; 119] is utilized so that the domain includes only one of three blades, as shown in Figure 7. The inflow, outflow and radial boundaries lie $0.5R$, $2R$ and $1.43R$ from the hub center, respectively. This can be more easily seen in Figure 8, where the inflow, outflow, and radial boundaries are the left, right and top edges, respectively, of the cut plane along the rotation axis. Each periodic boundary contains 1,430 nodes and 2,697 triangles. Near the rotor surface, we have 22 layers of refined mesh with first-layer thickness of 1 cm and a progression factor of 1.1. The boundary layer mesh at $\frac{3}{4}R$ is shown in Figure 9. The number of nodes and elements for each volume mesh is shown in Table 2.

	Surface		Volume	
	nn	*ne*	*nn*	*ne*
Mesh-2	5,748	11,452	155,494	898,640
Mesh-3	7,552	15,060	205,855	1,195,452
Mesh-4	9,268	18,492	253,340	1,475,175

Table 2 Summary of the meshes. Here *nn* and *ne* are the number of nodes and elements.

We compute the problem with the ST-SUPS and ST-VMS methods, which are labeled in some of the figures in this section as "DSD/SST-DP-SUPS" and "DSD/SST-DP-VMST" (see Remark 5). For τ_{SUPG}, τ_{PSPG} and τ_{SUPS}, we use the definition given by Eq. (11). The ST-SUPS method is used without the LSIC stabilization ($\nu_{\text{LSIC}} = 0$), while for the ST-VMS method we use the TGI option of ν_{LSIC}, given by Eq. (8).

Fig. 6 Rotor surface mesh (Mesh-4).

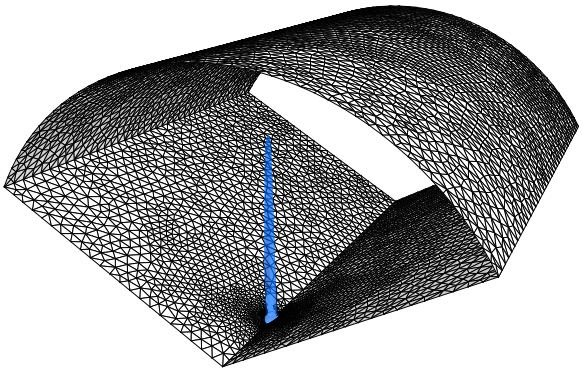


Fig. 7 Rotationally-periodic domain with wind-turbine blade shown in blue.

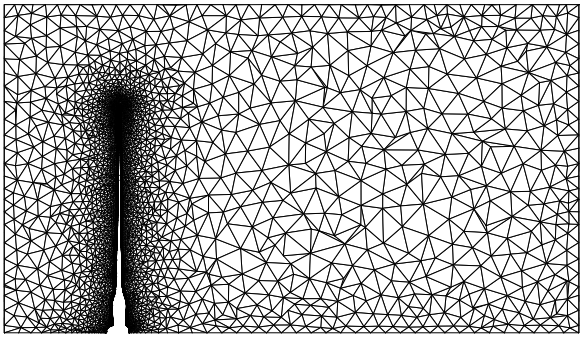


Fig. 8 Cut plane of the fluid volume mesh along rotor axis (Mesh-4).

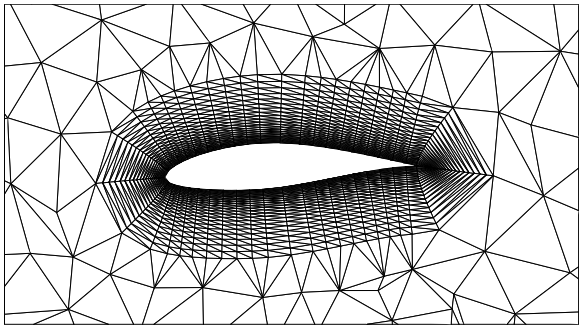


Fig. 9 Boundary layer mesh at $\frac{3}{4}R$.

starting from a lower Reynolds number and gradually reaching the actual Reynolds number. This solution is used as the initial condition also for the computations with the ST-VMS method. The purpose is to generate a divergence-free and reasonable flow field at this Reynolds number. We note that it was especially difficult with the ST-VMS method to start from non-physical conditions, such as setting all nodes except those on the blade to the inflow velocity.

The blade is segmented into 18 spanwise "patches" shown in Figure 10 to investigate how the aerodynamic torque distribution varies along the blade span. Figures 11–13 show the time history of the aerodynamic torque and the torque contribution from each patch for a single blade at t = 1.0 s. Figure 14 shows the pressure coefficient at t = 1.0 s for Patch 16 (at 0.90R), which is a representative section of the blade. For most of the patches, the angle of attack and Reynolds number do not vary much from one patch to another. For example, the angle of attack and Reynolds number are 7.4° and 9.9×10^6 at 0.65R for Patch 12 (at 0.65R) and 7.6° and 9.6×10^6 for Patch 16 (at 0.90R).

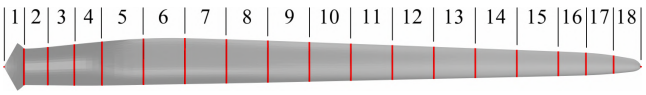


Fig. 10 Patches along the blade spanwise dimension.

In solving the linear equation systems involved at every nonlinear iteration, the GMRES search technique [121] is used with a diagonal preconditioner. The computation is carried out in a parallel computing environment, using PC clusters. The mesh is partitioned to enhance the parallel efficiency of the computations. Mesh partitioning is based on the METIS algorithm [122]. The time-step size is 4.67×10^{-4} s. The number of nonlinear iterations per time step is 3 with 30, 60 and 500 GMRES iterations for the first, second and third nonlinear iterations, respectively.

Prior to the computations reported here, we performed a series of brief computations with the ST-SUPS method,

Mesh refinement studies for both the ST-SUPS and ST-VMS methods indicate good convergence in the quantities

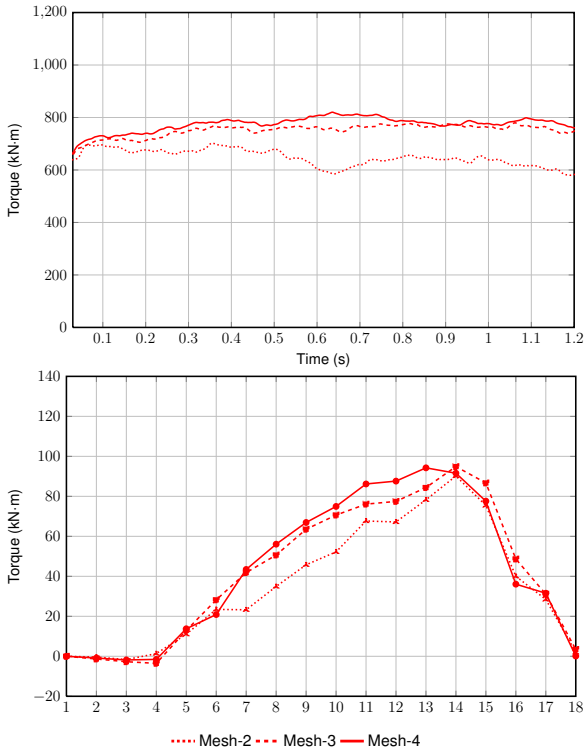


Fig. 11 The aerodynamic torque generated by a single blade. Comparison between different meshes with the ST-SUPS method. Time history (top). The torque contribution from each patch at $t = 1.0$ s (bottom).

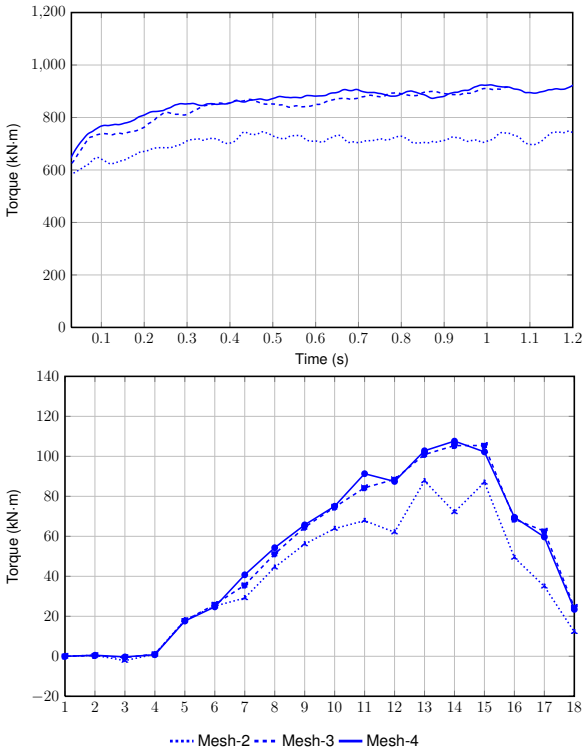


Fig. 12 The aerodynamic torque generated by a single blade. Comparison between different meshes with the ST-VMS method. Time history (top). The torque contribution from each patch at $t = 1.0$ s (bottom).

of interest such as the aerodynamic torque and pressure coefficient. The ST-VMS method with the finest mesh gives more or less the same value of the aerodynamic torque as the ALE-VMS simulation from [6] using NURBS, which is taken as a reference solution for this study. The results for the ST-SUPS method are also very good, however the torque is slightly under-predicted with respect to the ST-VMS and NURBS-based ALE-VMS simulations. Figure 14 indicates that smoother (i.e., more stable) pressure solution is obtained with the ST-VMS method. We note that the main reason behind the higher ST-VMS torque is the wider low-pressure region on the upper surface of the NACA64 geometry, as can be seen in the figure. The lower pressure indicates that the flow is attached; thus, the ST-VMS method, for the level of mesh refinement used, is able to better represent the turbulent boundary layer solution than the ST-SUPS method.

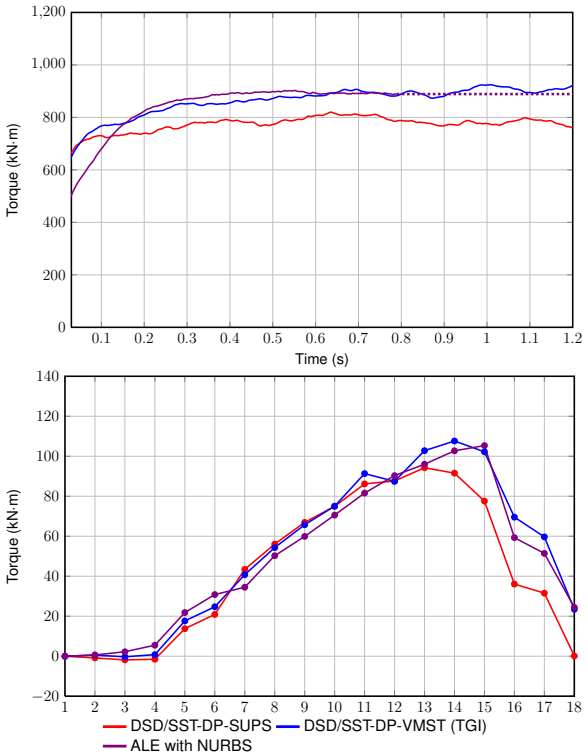
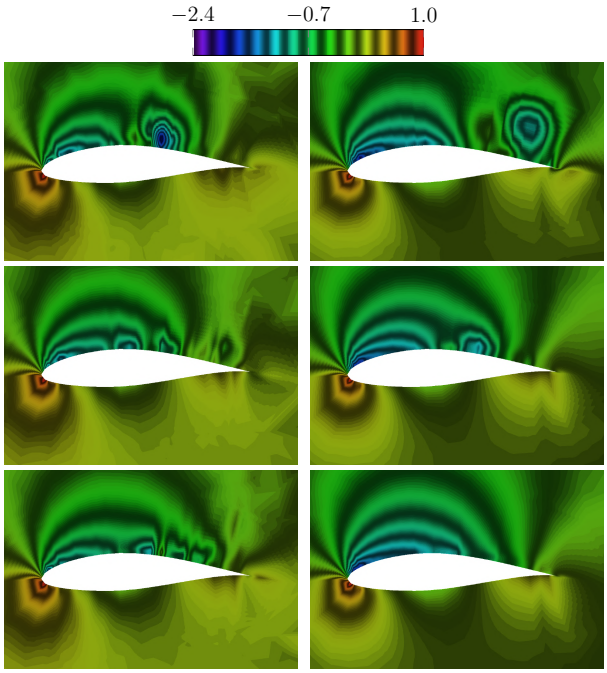


Fig. 13 The aerodynamic torque generated by a single blade. Computed with different methods using Mesh-4. Time history (top). The torque contribution from each patch at $t = 1.0$ s (bottom). The curve labeled "ALE with NURBS" is from the ALE-VMS simulation from [6].



(a) ST-SUPS (b) ST-VMS

Fig. 14 Pressure coefficient at t = 1.0 s for Patch 16 (at 0.90R). Top: Mesh-2. Middle: Mesh-3. Bottom: Mesh-4.

4 Sliding-Interface Formulation and Simulation of Rotor–Tower Interaction

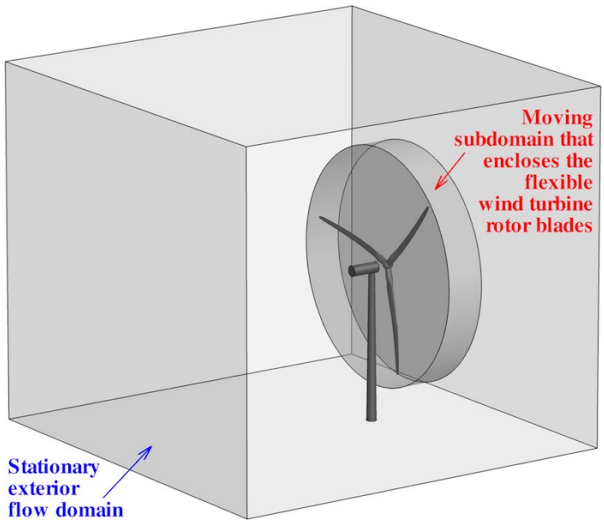


Fig. 15 Setup for the simulation of a full machine. The interior moving subdomain, which encloses the wind turbine rotor, and the exterior stationary subdomain, which houses the nacelle and tower, are shown.

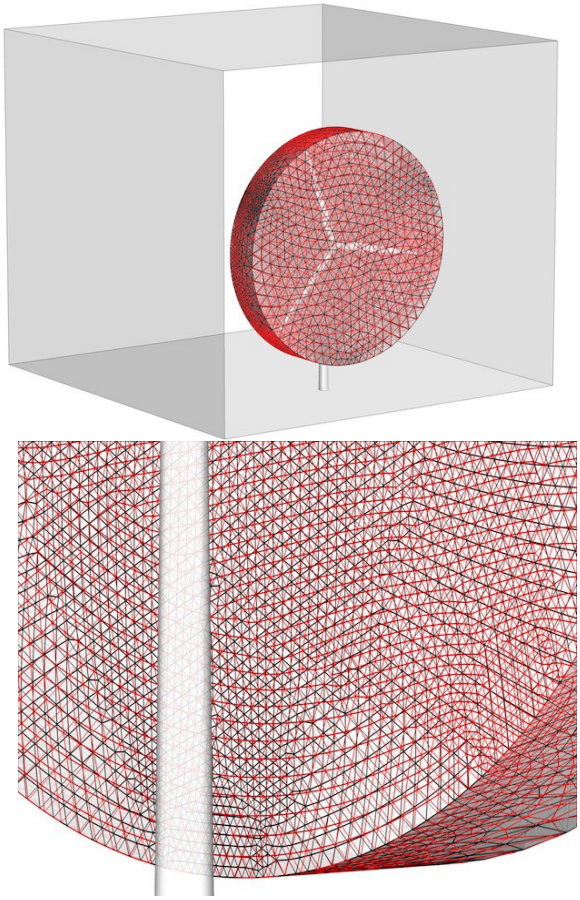


Fig. 16 Nonmatching meshes at the sliding interface between the stationary and moving subdomains. Top: Full domain. Bottom: Zoom on the sliding interface.

4.1 Sliding-Interface Formulation

In order to simulate the full wind turbine configuration and investigate the rotor–tower interaction, we consider an approach that makes use of a moving subdomain, which encloses the entire wind turbine rotor, and a stationary subdomain that contains the rest of the wind turbine (see Figure 15). The two domains are in relative motion and share a sliding cylindrical interface. The meshes on each side of the interface are nonmatching because of the relative motion (see Figure 16). As a result, a numerical procedure is needed to impose the continuity of the kinematics and tractions at the stationary and rotating subdomain interface despite the fact that the interface discretizations are incompatible. Such a procedure was developed in [44] in the context of IGA for computing flows about rotating components. The advantage of IGA for rotating-component flows is that the cylindrical sliding interfaces are represented exactly and no geometry errors are incurred. In the case of standard FEM employed here, the geometric compatibility is only approximate. The sliding-interface coupling was successfully tested on the NREL Phase VI wind turbine in [93] and is presented in what follows.

Let the subscripts S and M denote the quantities pertaining to the fluid mechanics problem on the stationary and moving subdomains, respectively. The subdomain that encloses the rotor rotates with it, and the interior of the rotating subdomain is allowed to deflect to accommodate the motion of the blades. However, the motion of the outer boundary of the rotor subdomain is restricted to a rigid rotation to maintain geometric compatibility with the stationary subdomain. To enforce the compatibility of the flow kinematics and tractions at the sliding interface, we add the following terms to the ALE-VMS formulation, which is now assumed to hold

in both the stationary and moving subdomains:

$$
\begin{aligned}
&-\sum_{b=1}^{n_{\mathrm{eb}}} \int_{\Gamma_t^b \cap (\Gamma_t)_{\mathrm{SI}}} \left(\mathbf{w}_{\mathrm{S}}^h - \mathbf{w}_{\mathrm{M}}^h\right) \cdot \frac{1}{2} \left(\boldsymbol{\sigma}_{\mathrm{S}} \mathbf{n}_{\mathrm{S}} - \boldsymbol{\sigma}_{\mathrm{M}} \mathbf{n}_{\mathrm{M}}\right) \mathrm{d}\Gamma \\
&-\sum_{b=1}^{n_{\mathrm{eb}}} \int_{\Gamma_t^b \cap (\Gamma_t)_{\mathrm{SI}}} \frac{1}{2} \left(\delta\boldsymbol{\sigma}_{\mathrm{S}} \mathbf{n}_{\mathrm{S}} - \delta\boldsymbol{\sigma}_{\mathrm{M}} \mathbf{n}_{\mathrm{M}}\right) \cdot \left(\mathbf{u}_{\mathrm{S}}^h - \mathbf{u}_{\mathrm{M}}^h\right) \mathrm{d}\Gamma \\
&-\sum_{b=1}^{n_{\mathrm{eb}}} \int_{\Gamma_t^b \cap (\Gamma_t)_{\mathrm{SI}}} \mathbf{w}_{\mathrm{S}}^h \cdot \rho \left\{\left(\mathbf{u}_{\mathrm{S}}^h - \hat{\mathbf{u}}_{\mathrm{S}}^h\right) \cdot \mathbf{n}_{\mathrm{S}}\right\}_- \left(\mathbf{u}_{\mathrm{S}}^h - \mathbf{u}_{\mathrm{M}}^h\right) \mathrm{d}\Gamma \\
&-\sum_{b=1}^{n_{\mathrm{eb}}} \int_{\Gamma_t^b \cap (\Gamma_t)_{\mathrm{SI}}} \mathbf{w}_{\mathrm{M}}^h \cdot \rho \left\{\left(\mathbf{u}_{\mathrm{M}}^h - \hat{\mathbf{u}}_{\mathrm{M}}^h\right) \cdot \mathbf{n}_{\mathrm{M}}\right\}_- \left(\mathbf{u}_{\mathrm{M}}^h - \mathbf{u}_{\mathrm{S}}^h\right) \mathrm{d}\Gamma \\
&+\sum_{b=1}^{n_{\mathrm{eb}}} \int_{\Gamma_t^b \cap (\Gamma_t)_{\mathrm{SI}}} \frac{C_I^B \mu}{h_n} \left(\mathbf{w}_{\mathrm{S}}^h - \mathbf{w}_{\mathrm{M}}^h\right) \cdot \left(\mathbf{u}_{\mathrm{S}}^h - \mathbf{u}_{\mathrm{M}}^h\right) \mathrm{d}\Gamma = 0, \quad (45)
\end{aligned}
$$

where $\delta\boldsymbol{\sigma}$ is given by

$$
\delta\boldsymbol{\sigma}\left(\mathbf{w}, q\right) \mathbf{n} = 2\mu\boldsymbol{\varepsilon}(\mathbf{w})\mathbf{n} + q\mathbf{n}, \quad (46)
$$

$(\Gamma_t)_{\mathrm{SI}}$ is the sliding interface, and $\{\mathcal{A}\}_-$ denotes the negative part of $\mathcal{A}$, that is, $\{\mathcal{A}\}_- = \mathcal{A}$ if $\mathcal{A} < 0$ and $\{\mathcal{A}\}_- = 0$ if $\mathcal{A} \geq 0$. The sliding-interface formulation may be see as a DG method, where the continuity of the basis function is enforced everywhere in the interior of the two subdomains, but not at the sliding interface between them. The structure of the terms on the sliding interface is similar to that of the weak enforcement of essential boundary conditions. The significance of each term is explained in detail in [44]. Note that, in the current application, $\hat{\mathbf{u}}_{\mathrm{S}}^h = \mathbf{0}$, because the subdomain S is stationary. However, the formulation is able to handle situations where both subdomains are in motion.

Remark 11 *Nonmatching interface discretizations in the FSI and sliding-interface problems necessitate the use of interpolation or projection of kinematic and traction data between the nonmatching surface meshes (see, e.g., [51; 123; 52], where [52] is more comprehensive than [51]). A computational procedure, which can simultaneously handle the data transfer for IGA and FEM discretizations, was proposed in [123]. The procedure also includes a robust approach in identifying "closest points" for arbitrary shaped surfaces. While such interface projections are rather straightforward for weakly-coupled FSI algorithms, they require special techniques [57; 68; 52] for strongly-coupled, direct and quasi-direct methods [124; 67; 57; 68; 52] that are monolithic-like (i.e. become monolithic for matching discretizations).*

4.2 Aerodynamics Simulation of the NREL Phase VI Wind Turbine

The computational results in this section make use of the ALE-VMS technique and are taken from [93]. The sliding-interface formulation is applied to the simulation of the full NREL Phase VI wind turbine configuration, including the rotor (blades and hub), nacelle and tower. The tower is composed of two cylinders with diameters of 0.6096 m and 0.4064 m that are connected with a short conical section. The tower height is 11.144 m above the wind tunnel floor. The detailed geometry and configuration of the tower and nacelle can be found in Hand *et al.* [94]. For this study, wind speeds of 7 and 10 m/s were selected from the experimental sequence S. The experimental sequence S setup consists the wind turbine rotor in the upwind configuration, 0° yaw angle, 0° cone angle, rotational speed of 72 rpm, and blade tip pitch angle of 3°. The air density and viscosity are 1.23 kg/m^3 and 1.78×10^{-5} kg/(m·s), respectively.

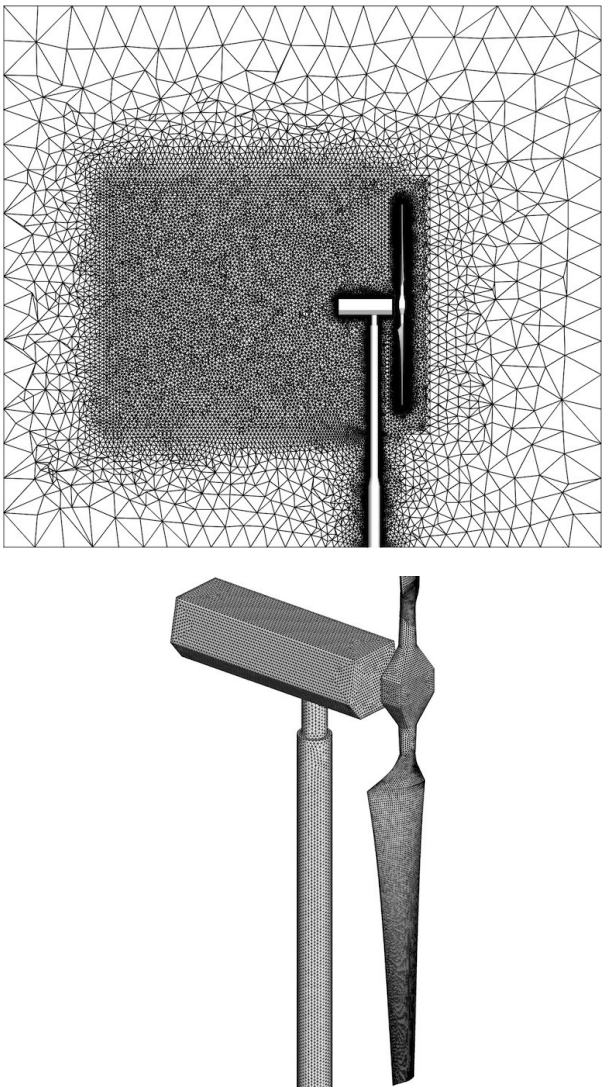


Fig. 17 Meshes used in the full-wind-turbine simulation. Top: 2D cut at $x = 0$ to show the flow domain mesh quality. Bottom: Rotor, nacelle, and tower surface mesh.

Figure 17 shows the mesh quality and resolution used in the full-wind-turbine computation. The mesh is highly

refined near the rotor, nacelle and tower, as well as downstream of the wind turbine to better capture the wake turbulence. The mesh is comprised of 6,835,647 linear elements and 1,603,377 nodes. The size of the first boundary-layer element in the wall-normal direction is 0.002 m, and 15 layers of prismatic elements were generated with a growth ratio of 1.2. The time step size is set to 1.0×10^{-5} s.

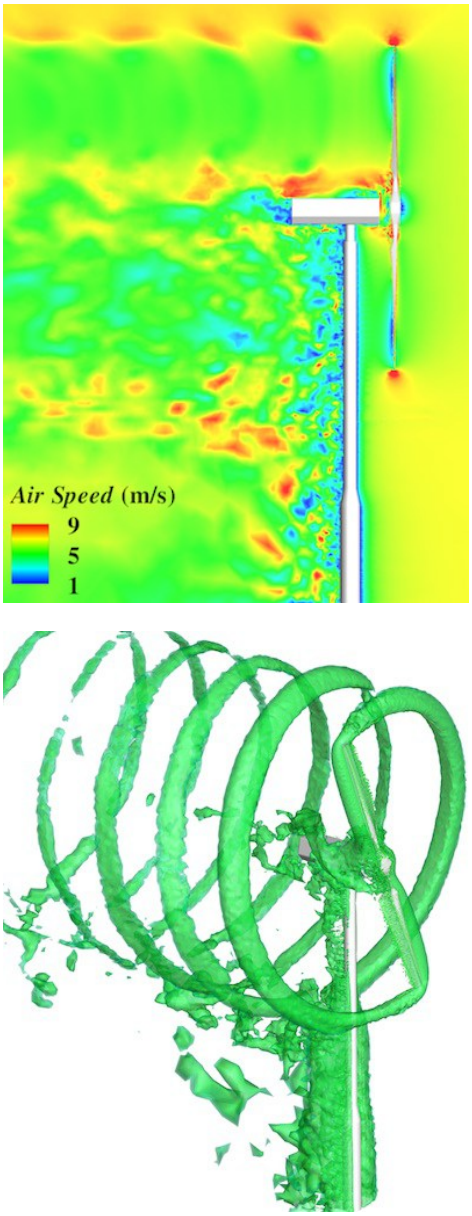


Fig. 18 Air speed contours at a planar cut (top), and isosurfaces of air speed (bottom) at an instant for the 7 m/s case.

Figures 18 and 19 show the flow visualization of the full-wind-turbine simulations of the 7 and 10 m/s cases, respectively. The flow structures are different between the two cases. The tip vortex for the 7 m/s case decays very slowly as it is convected downstream, while the tip vortex breaks

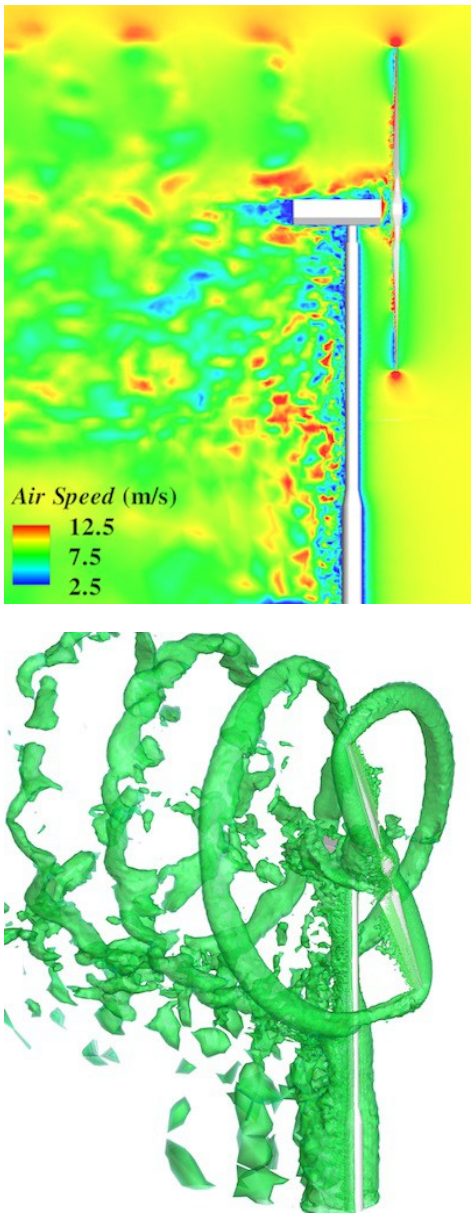


Fig. 19 Air speed contours at a planar cut (top), and isosurfaces of air speed (bottom) at an instant for the 10 m/s case.

down quickly for the 10 m/s case. Note that no visible discontinuities are present in the flow field at the sliding interface, which indicates that the method correctly handles the kinematic compatibility conditions in this location.

To see the influence of the tower, the single-blade aerodynamic torque over a full revolution is plotted in Figure 20 for both 7 and 10 m/s cases. The results of the full-wind-turbine computations are compared with the experimental data, as well as with the results of the rotor-only computations. For the full-wind-turbine simulation of the 7 m/s case, Figure 20(a) clearly shows the drop in the aerodynamic torque at an instant when the blade passes in front of the tower, which corresponds to the azimuthal angle of

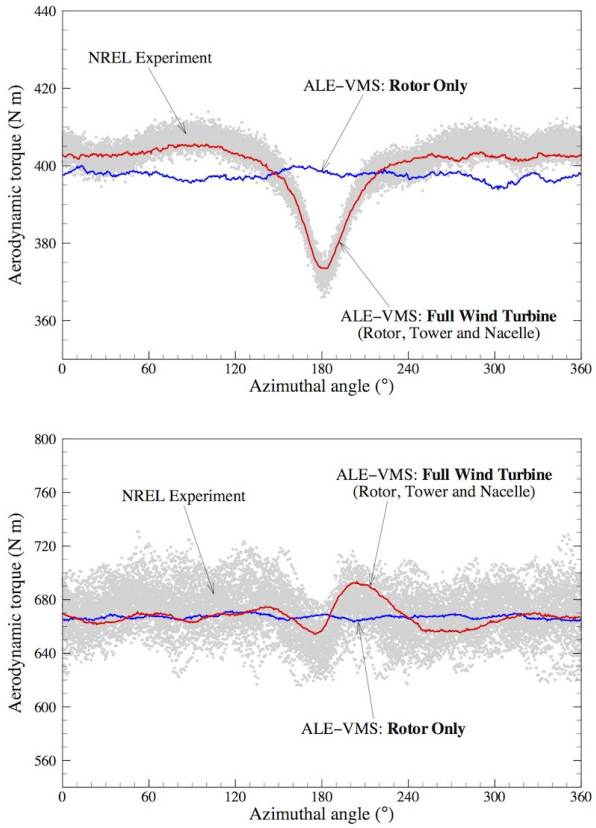


Fig. 20 Single-blade aerodynamic torque over a full revolution for 7 m/s (top) and 10 m/s (bottom) cases. The 180° azimuthal angle corresponds to the instant when the blade passes in front the tower. The tower effect is clearly pronounced in the 7 m/s case. It is also present in the 10 m/s case, but is not as significant. The results in both cases are in very good agreement with the experimental data.

180°. The drop in the torque is about 8% relative to its value when the blade is away from the tower. These results are in good agreement with the experimental data. The rotor-only computation, which is also shown in the figure, is obviously unable to predict this feature, which may be important for the transient structural response of the blades. It should be noted, however, that the cycle-averaged aerodynamic torque is nearly identical for the full-wind-turbine and the rotor-only simulations. The picture is completely different for the 10 m/s case, where the influence of the tower, although clearly present, is a lot less pronounced.

5 ST-VMS Computation of the Wind-Turbine Rotor and Tower Aerodynamics

This section is from [83]. Starting from the rotor surface geometry given in Section 3, we generate a quadratic NURBS surface with G^2 and G^1 continuity between the patches around and along the blade, respectively. The tower geometry was created based on the tower design specified for the NREL 5MW offshore baseline wind turbine, which describes a circular tower with a height of 87.6 m, a base diameter of 6 m, and a top diameter of 3 m. This geometry was generated by lofting between NURBS curves for the top and base of the tower. The rotor axis is 90° to the tower, and there is no tilt or precone, since it is only a fluid mechanics computation.

5.1 Problem Setup

We compute the aerodynamics of the rotor with and without its tower for a given rotor shape and wind speed and a specified rotor speed. The rotor and tower geometries are shown in Figure 21.

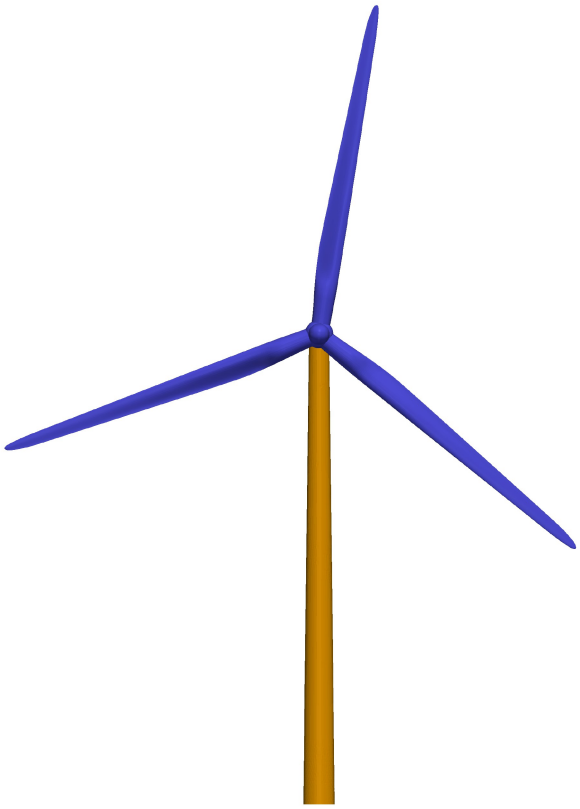


Fig. 21 Wind-turbine rotor and tower geometries.

The wind and rotor speeds are the same as those in Section 3. At the inflow boundary the velocity is set to the wind velocity, at the outflow boundary the stress vector is set to zero, and at the top, side, and bottom boundaries slip conditions are imposed.

5.2 Rotor Motion

The circular turbine rotation is represented with temporal NURBS basis functions and secondary mapping, described in Section 2.4. Because the 3 blades of the turbine are 120° apart, rotational geometric periodicity is used such that a full 360° rotation is defined by 3 identical 120° segments. Each 120° segment is divided into 6 patches to keep the mesh distortion under control. Each patch is a 20° arc, with 3 temporal-control points. The 6 temporal patches and their control points are illustrated in Figure 22 and Table 3.

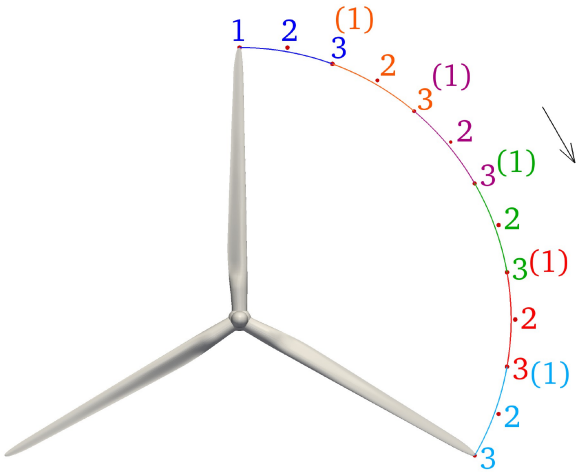


Fig. 22 Path of a blade tip with temporal patches and control point numbering local to each patch. A control point at the start of a patch and colocated with a control point at the end of the previous patch is in parentheses. For the color code, see Table 3.

Temporal patch	Fig. 22 color
1	Blue
2	Orange
3	Purple
4	Green
5	Red
6	Teal

Table 3 Figure 22 color code for the temporal patches.

5.3 Surface Mesh

The rotor surface mesh is generated by discretizing the NURBS surface geometry at each knot intersection, subdividing the knot spans into quadrilateral finite elements in a

structured way, and subdividing the quadrilateral elements into two triangles. Small adjustments are made to improve the mesh near the hub. The surface mesh position is calculated at each temporal-control point shown in Figure 22. Figure 23 shows the rotor surface at the three temporal-control points of the first patch. We note that control points 1 and

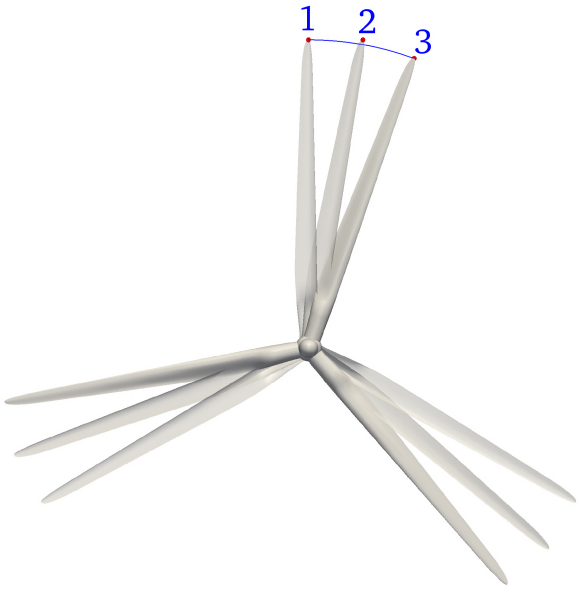


Fig. 23 Rotor surface at the three temporal-control points of the first patch.

3 lie on the path traveled by the points on the blades and a portion of the hub at the start and end of the 20° rotation, but control point 2 lies outside the circular arc. This means that the temporal-control mesh 2 is deformed compared to the temporal-control meshes 1 and 3. A temporal-control mesh 2 has to be generated for the part of the surface between the hub cross-sections rotating with the blades and fixed to the tower. The tower surface mesh is generated from the NURBS representation of the surface by using an unstructured triangular mesh generator and matched with the previously generated hub mesh at the intersection. The rotor surface mesh has 34,087 nodes and 68,112 triangles. The tower surface mesh has 6,952 nodes and 13,806 triangles.

5.4 Volume Mesh

5.4.1 Boundary-Layer Mesh

The layers of thin elements near the blades are generated by extruding the NURBS surface geometry into NURBS volume representation, subdividing the knot spans into hexahedral finite elements in a structured way, and subdividing the hexahedral elements into six tetrahedral elements. The resulting boundary-layer mesh for each blade consists of 4 layers with a first-layer thickness of about 2.85×10^{-2} m and a total thickness of about 2.85×10^{-1} m, 52 nodes in the circumferential direction around the blade, and approximately 145 nodes in the longitudinal direction. The tower boundary-layer mesh is generated by extruding the tower surface mesh to layers of prismatic elements, which are then subdivided into 3 tetrahedral elements each. It consists of 4 layers, with a first-layer thickness of 2.85×10^{-2} m and a total thickness of 3.0×10^{-1} m. The blade and tower boundary-layer meshes do not undergo any mesh deformation. This maintains the mesh quality in the boundary-layer regions. Figure 24 shows the outer surface of the blade boundary-layer mesh and cut-planes showing the tower and blade boundary-layer meshes.

5.4.2 Overall Mesh

Three different meshes are used in the computations: Mesh 1, Mesh 2, and Mesh 3. Mesh 2 has both the rotor and the tower, with boundary-layer mesh only for the blades. Mesh 1 has only the rotor, and is identical to Mesh 2 except the tower is filled with volume elements. Mesh 3 has both the rotor and the tower, with boundary-layer mesh for both the blades and the tower, and a mesh refinement region downstream of the tower. All three meshes have an outer, coarser region, with an inner cylindrical refinement region surrounding the rotor. This inner refinement region includes most of the tower for Mesh 2 and Mesh 3, and the mesh refinement region downstream of the tower for Mesh 3. Figure 25 illustrates, as an example, cut planes of Mesh 3, and Figure 26 shows zoomed longitudinal cut planes of all three meshes. The inflow and outflow boundaries are at $3.79R$ and $10.35R$ from the hub center, respectively. The side, top, and bottom boundaries are at $2.29R$, $3.17R$, and $1.43R$, respectively (see Figure 25). The volume mesh is generated once per patch using an automatic mesh generator (a total of 6 times). The mesh is generated at control point 2 of each patch to minimize mesh distortion between control points. We note that only the mesh in the inner cylindrical refinement region surrounding the rotor is generated for each patch. The outer, coarser mesh is generated only once, and is kept the same when the inner meshes are generated for each patch. The mesh moving technique [88; 89; 90; 91; 57] developed earlier in conjunction with the DSD/SST method is used for computing the mesh position for control points 1 and 3. The outer surfaces of the boundary-layer meshes serve as the boundaries where we specify the inner boundary conditions for the mesh motion. The external boundaries of the computational domain serve as the boundaries where we specify the outer boundary conditions, with zero displacement. In the elasticity equations of the mesh moving technique, a Young's modulus of 1.0, a Poisson's ratio of -0.20, and a Jacobian-based stiffening exponent of 1.5 are used. We use 1,500 GMRES iterations for each step of the mesh motion, with diagonal preconditioner. Each 10° range of motion is computed over 40 steps. Number of nodes and elements for all 6 temporal patches of the 3 volume meshes are given in Table 4.

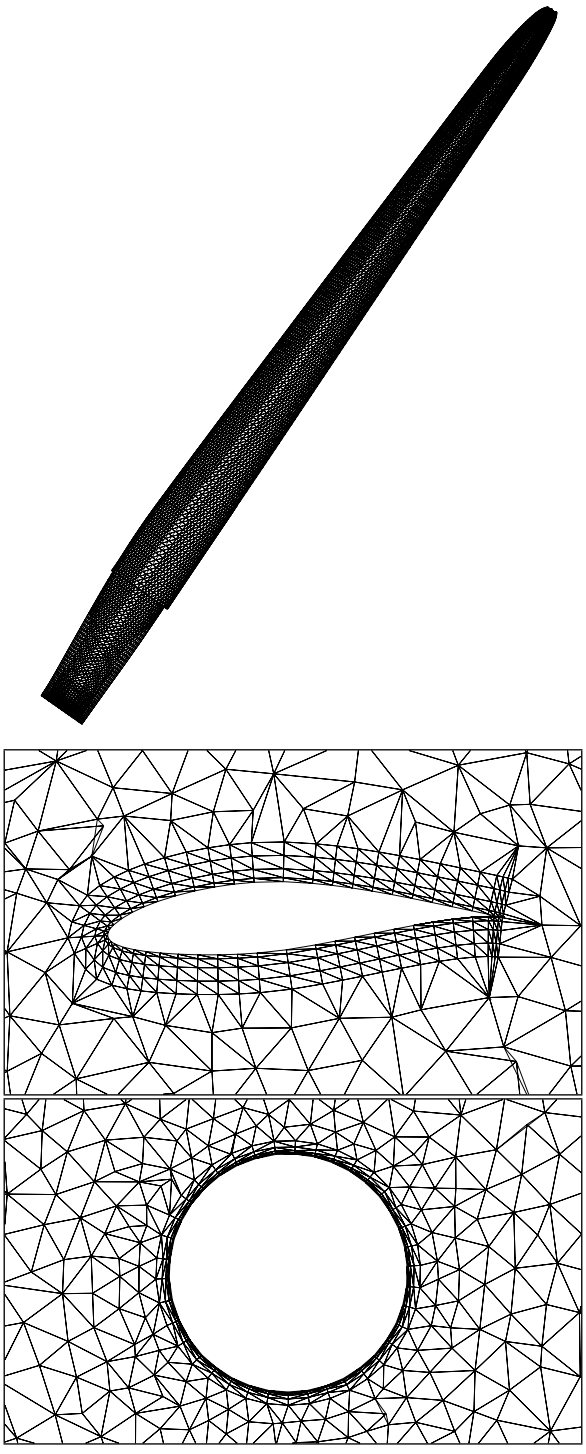


Fig. 24 Top: Outer surface of blade and boundary-layer mesh. Middle: Boundary-layer mesh at $\frac{3}{4}R$. Bottom: Tower boundary-layer mesh.

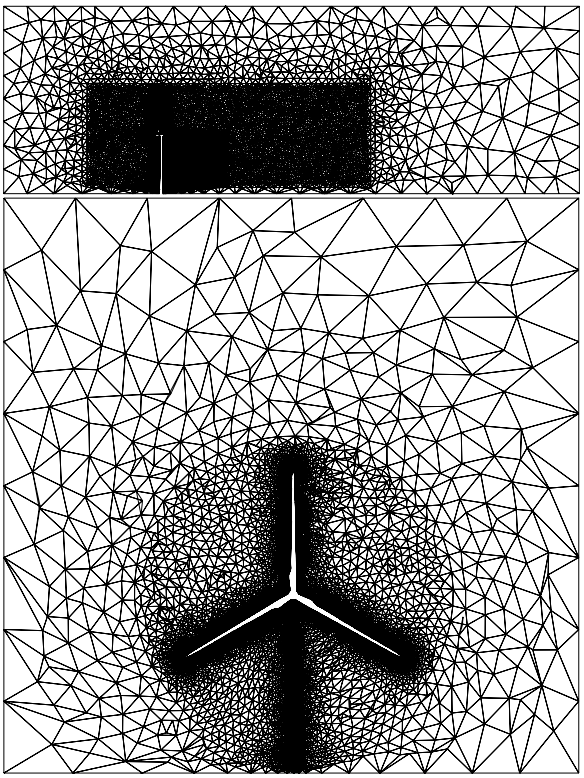


Fig. 25 Cut planes of temporal-control mesh 1 of patch 1 for Mesh 3.

Mesh	Temporal patch	*nn*	*ne*
1	1	470,880	2,725,614
1	2	466,983	2,701,657
1	3	460,932	2,665,562
1	4	462,733	2,676,747
1	5	464,712	2,687,745
1	6	468,529	2,711,069
2	1	446,709	2,553,100
2	2	442,876	2,529,556
2	3	436,825	2,493,524
2	4	438,802	2,505,789
2	5	440,870	2,517,233
2	6	444,517	2,539,512
3	1	598,125	3,454,865
3	2	596,111	3,442,699
3	3	592,345	3,420,273
3	4	590,628	3,410,226
3	5	595,719	3,440,031
3	6	596,522	3,445,407

Table 4 Number of nodes (*nn*) and elements (*ne*) for the fluid mechanics meshes used in each temporal patch.

5.5 Computational Conditions

For τ_{SUPS}, we use the definition given by Eq. (11), with h_{RGN} (= h_{RGNT}) given by Eq. (33). For ν_{LSIC}, we use the $\nu_{\text{LSIC–HRGN}}$ definition given by Eq. (37). The DTR and IMTR approaches are used on all three meshes. Least-

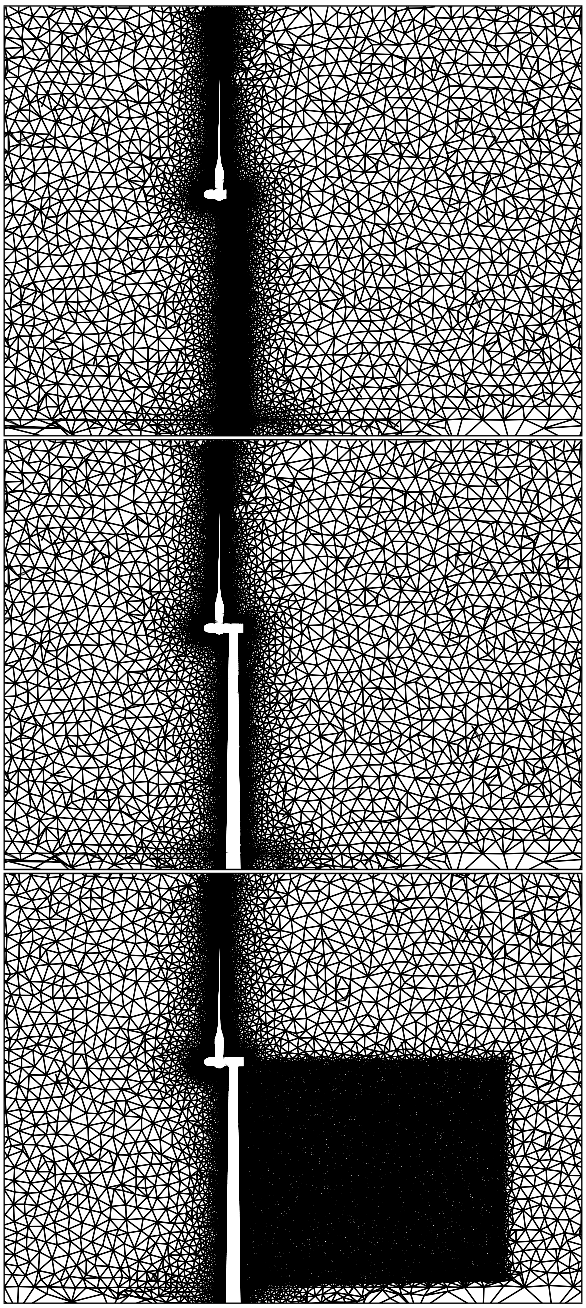


Fig. 26 Zoomed cut planes of temporal-control mesh 1 of patch 1 for Mesh 1 (top), Mesh 2 (middle), and Mesh 3 (bottom).

Mesh	Tower	Temporal representation
1	No	DTR
2	Yes	DTR
3	Yes	DTR
1	No	IMTR
2	Yes	IMTR
3	Yes	IMTR

Table 5 Summary of the computations.

steps from zero to the wind speed using a cosine ramp. During this flow-development stage of the computation, we use 150, 150, 200, and 400 GMRES iterations for the 4 nonlinear iterations. In computations with the rotor in motion, we use 150, 150, 200, and 400 GMRES iterations for Mesh 1, and 150, 250, 350, and 500 GMRES iterations for Mesh 2 and Mesh 3. With the GMRES iterations in flow computations, we use nodal-block-diagonal preconditioner. The mesh is partitioned based on the METIS algorithm to improve parallel efficiency in the computations.

5.6 Results

Figure 27 shows the torque for Mesh 1 with the DTR approach, for the last 360° rotation of a blade, with the rotation amount measured from the orientation seen in Figure 22. For reference purposes, Figure 27 includes the NREL data. The torque is within 8% of the NREL data.

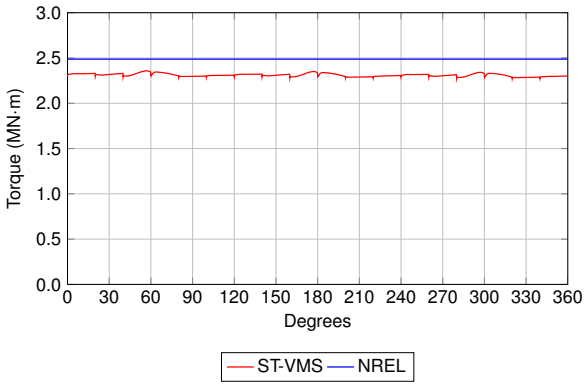


Fig. 27 Torque for Mesh 1 with the DTR approach, compared with the NREL data.

squares projection is used to interpolate the velocity and pressure between temporal patches. Because the boundary-layer meshes and the tower and rotor surface meshes remain identical between temporal patches, the velocity values are transferred exactly for those nodes. The computations performed are summarized in Table 5.

The time-step size is 2.23×10^{-3} s (145 time steps per patch), with 4 nonlinear iterations per time-step. First we develop the flow field for 500 time steps while the rotor is static, ramping up the inflow velocity during the first 300

Figure 28 shows the torque for the last 80° rotation of a single blade of Mesh 1 with the DTR approach, compared with the torque from the single-blade ST-VMS computation with Mesh-4 from Section 3. The higher torque seen for the single-blade computation may be due to the fact that the computation was carried out for a much shorter duration, only 80° of rotation versus 1,080° for the Mesh 1 computation. Therefore the current computation likely represents a more settled torque value. The higher torque for the single-blade computation may also be due the fact that the com-

putation was carried out using a computational domain with significantly nearer lateral boundaries.

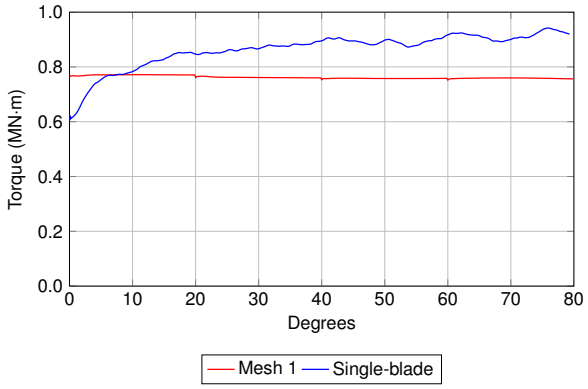


Fig. 28 Torque for a single blade of Mesh 1 with the DTR approach, compared with the torque from the single-blade ST-VMS computation with Mesh-4 from Section 3.

Figures 29 and 30 show the torque for all three meshes with the DTR and IMTR approaches. As can be seen from

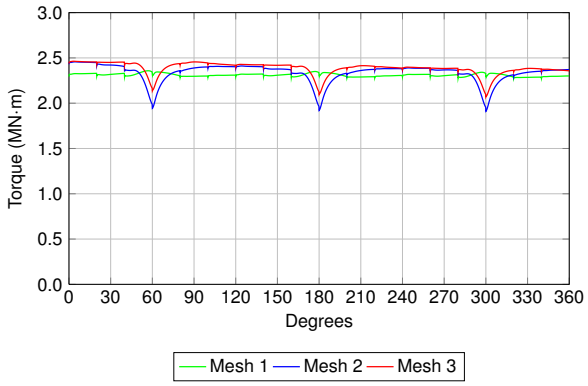


Fig. 29 Torque for Mesh 1, Mesh 2 and Mesh 3 with the DTR approach.

these figures, Mesh 1 (no tower) has a very stable torque, while Mesh 2 and Mesh 3 (with tower) exhibit a significant but expected drop in torque each time a blade passes the tower.

Figure 31 shows, for each of the three meshes, the torque obtained with the DTR and IMTR approaches. The figure illustrates that the DTR and IMTR approaches result in a nearly identical torque magnitude for all 3 meshes.

Figure 32 shows the torque for Mesh 1 with the DTR approach, using two different time-step sizes: 2.23×10^{-3} s (145 time steps per patch) and 4.49×10^{-3} s (72 time steps per patch). Doubling the time-step size still yields a comparable torque value, within 10% of the value for the smaller time-step size.

We also carried out a computation with the convective form of the ST-VMS formulation (see Eq. (8.17) in [52]), but

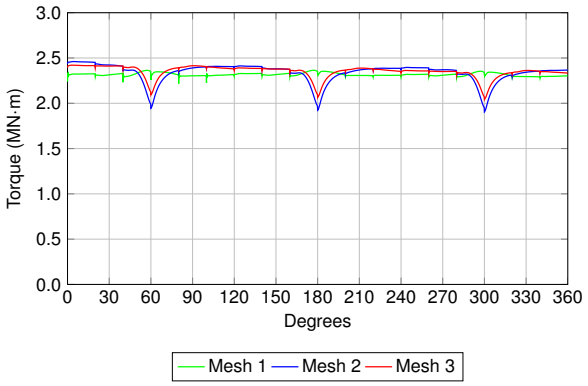


Fig. 30 Torque for Mesh 1, Mesh 2 and Mesh 3 with the IMTR approach.

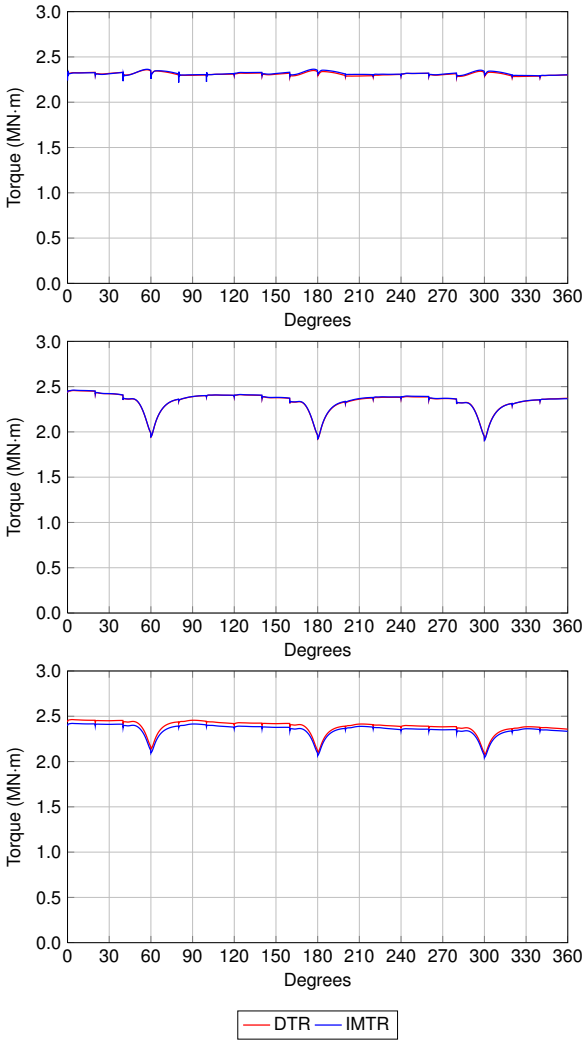


Fig. 31 Torque with the DTR and IMTR approaches for Mesh 1 (top), Mesh 2 (middle), and Mesh 3 (bottom).

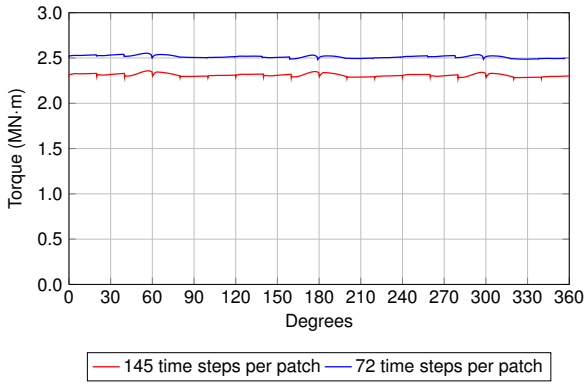


Fig. 32 Torque for Mesh 1 with the DTR approach, using two different time-step sizes: 2.23×10^{-3} s (145 time steps per patch) and 4.49×10^{-3} s (72 time steps per patch).

with a smaller time-step size: 4.46×10^{-4} s (725 time steps per patch). Figure 33 shows the torque for Mesh 2 with the DTR approach and the conservative and convective forms of the ST-VMS formulation. The conservative-form computation is with the standard time-step size: 2.23×10^{-3} s (145 time steps per patch).

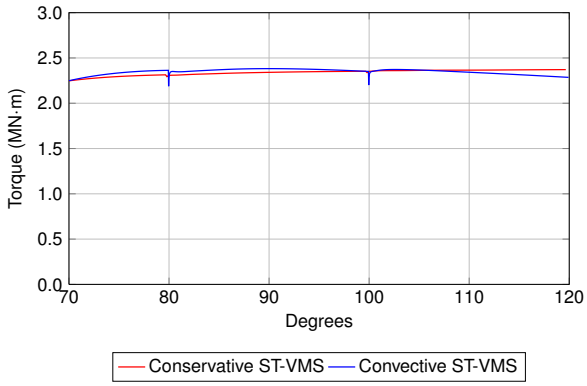


Fig. 33 Torque for Mesh 2 with the DTR approach and the conservative and convective forms of the ST-VMS formulation. The time-step sizes: 4.46×10^{-4} s (725 time steps per patch) for the convective form and 2.23×10^{-3} s (145 time steps per patch) for the conservative form. The torques shown are from the same period in a rotation cycle, but the conservative-form torque is from the last 360° of the computation, and the convective-form torque is from a recently-started, ongoing computation.

Figure 34 shows the torque for the individual blades of Mesh 2 with the DTR approach. The figure clearly shows the expected torque drop for each blade as it passes the tower, while the other two blades maintain relatively constant torque.

Figure 35 shows the torque for 10 equal-length spanwise sections of a blade of Mesh 2 with the DTR approach. Greatest amount of torque is generated in sections 6–9 of the blade, while section 10 at the tip and the other lower sections generate less torque.

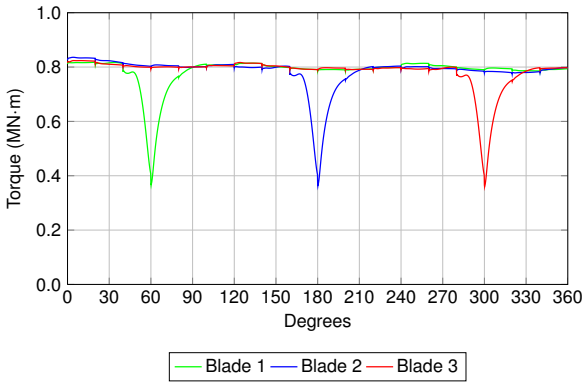


Fig. 34 Torque for the individual blades of Mesh 2 with the DTR approach.

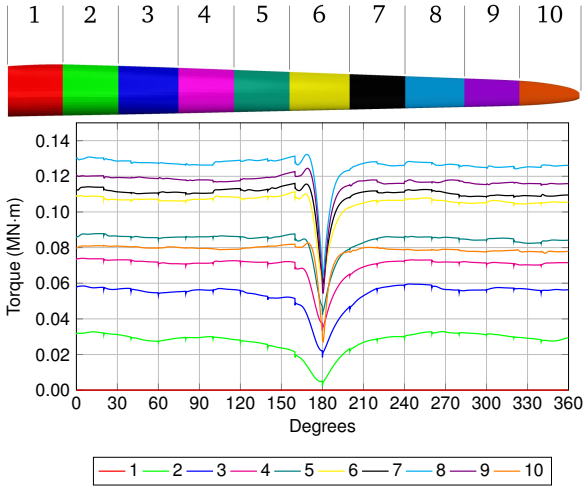


Fig. 35 Torque for 10 equal-length spanwise sections of a blade of Mesh 2 with the DTR approach.

Figure 36 shows a volume rendering of the vorticity for Mesh 2 with the DTR approach. The flow patterns vary considerably along each blade length, illustrating the necessity to carry out the computations in 3D.

Figure 37 shows the pressure coefficient at $0.90R$ for the last 0° orientation of a blade of Mesh 2, with the DTR and IMTR approaches, with the last 0° orientation being common between the two computations. There is very little difference in the pressure coefficient around the blades between the DTR and IMTR approaches.

Figure 38 shows the pressure coefficient at $0.90R$ for the last 180° orientation of a blade of Mesh 1, Mesh 2 and Mesh 3, with the DTR approach, with the last 180° orientation being common between Mesh 2 and Mesh 3 computations.

Table 6 provides the averaged torque for the last 360° rotation in all 6 computations. The values show that the difference in torque between the DTR and IMTR approaches, and between Mesh 2 and Mesh 3, is rather small. The difference in torque between Mesh 1 and Mesh 2 and 3 illustrates effect of the tower.

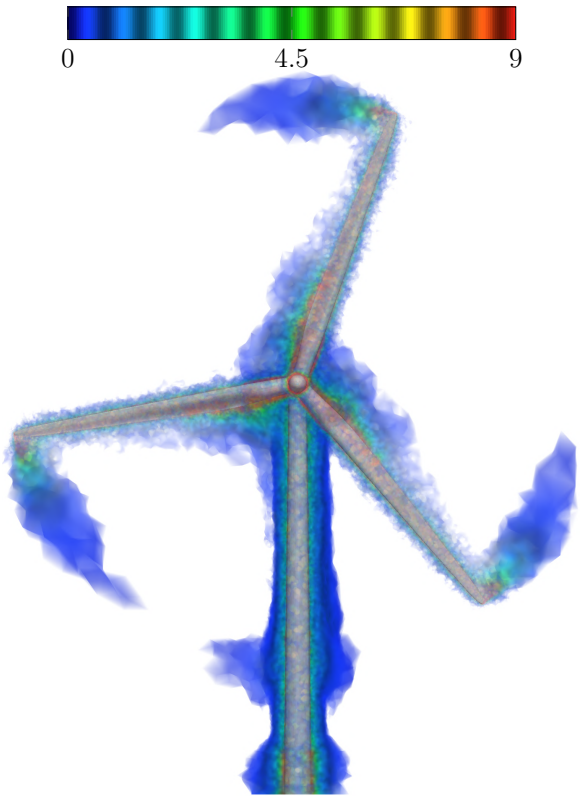


Fig. 36 Volume rendering of the vorticity (in s^{-1}) from the last 360° of the computation for Mesh 2 with the DTR approach.

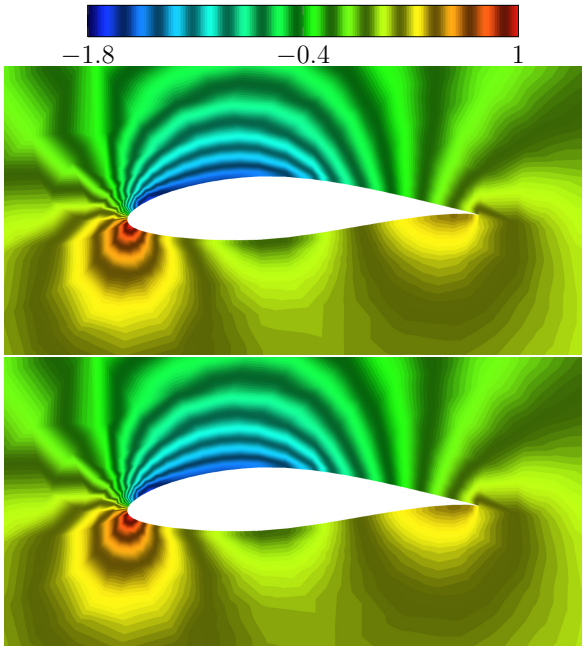


Fig. 37 Pressure coefficient at 0.90R for the last 0° orientation of a blade of Mesh 2, with the DTR (top) and IMTR (bottom) approaches.

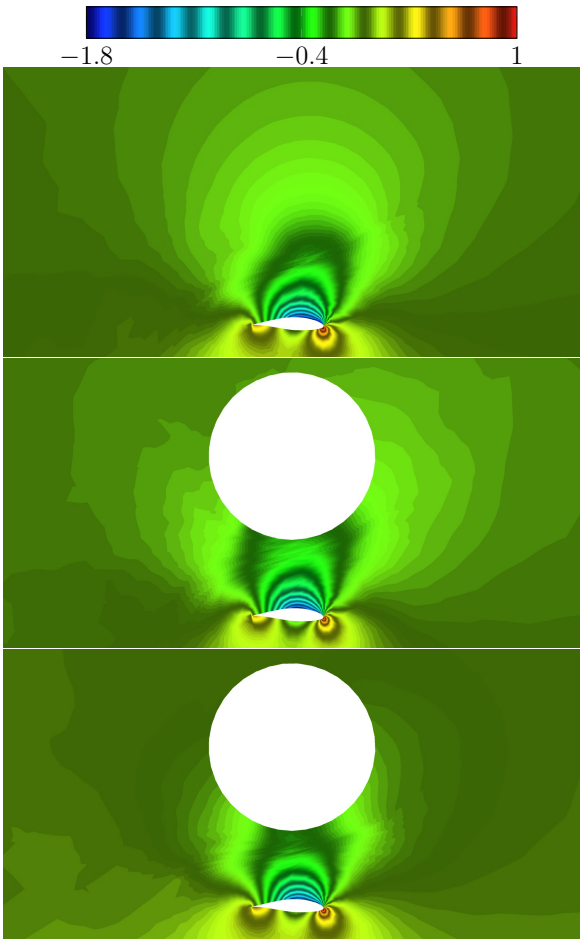


Fig. 38 Pressure coefficient at 0.90R for the last 180° orientation of a blade of Mesh 1 (top), Mesh 2 (middle), and Mesh 3 (bottom), with the DTR approach.

Mesh	DTR	IMTR
1	2.31	2.32
2	2.34	2.34
3	2.39	2.35

Table 6 Averaged torque (MN·m) for the last 360° rotation in all 6 computations.

Figure 39 shows the vorticity around the tower for Mesh 2 and Mesh 3 with the DTR approach. Mesh 3 is able to represent the wake behind the tower far more effectively, although no vortex shedding is observed at this stage of the computation, possibly due to insufficient computing duration or mesh refinement.

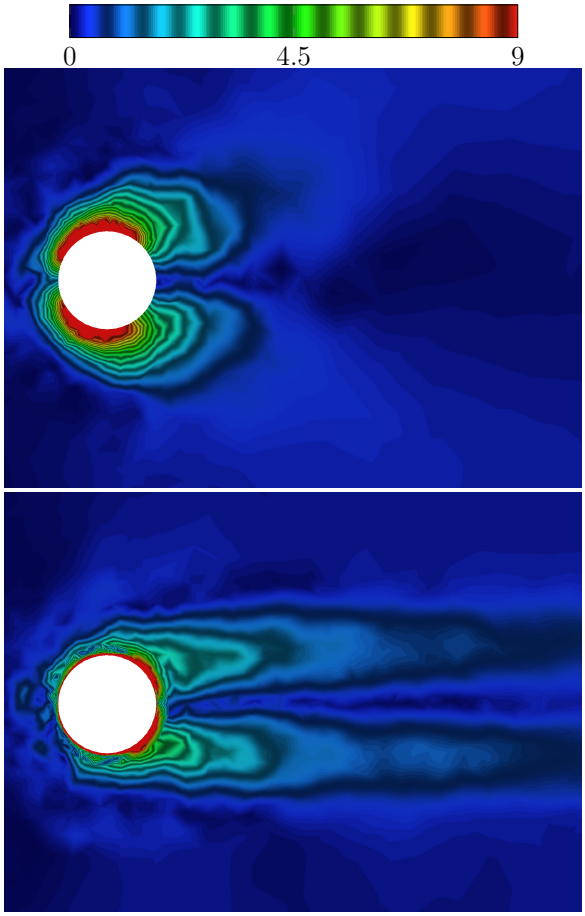


Fig. 39 Vorticity (in s^{-1}) around the tower at a cross-section $1.1R$ from the hub center for Mesh 2 (top) and Mesh 3 (bottom) with the DTR approach. The pictures are from the last time step of the computation.

6 Governing Equations of Structural Mechanics: Isogeometric Kirchhoff–Love Composite Shell and the Bending-Strip Method

6.1 Kirchhoff–Love Shell

In this section we follow the developments of [39; 13; 15] and present the governing equations of the Kirchhoff–Love shell theory. The theory is appropriate for thin-shell structures and requires no rotational degrees of freedom. The variational formulation of a Kirchhoff–Love shell is based on the principle of virtual work expressed as

$$\delta W = \delta W_{int} + \delta W_{ext} = 0\,, \tag{47}$$

where W, W_{int}, and W_{ext} denote the total, internal, and external work, respectively, and δ denotes a variation with respect to the virtual displacement variables $\delta\mathbf{y}$, that is

$$\delta W = \frac{\partial W}{\partial \mathbf{y}}\delta\mathbf{y}\,. \tag{48}$$

The internal virtual work is defined by (see, e.g., [125])

$$\delta W_{int} = -\int_{\Omega_0^s} (\delta\mathbf{E} \,:\, \mathbf{S})\ \mathrm{d}\Omega\,, \tag{49}$$

where Ω_0^s is the shell volume in the reference configuration (the total Lagrangian approach is adopted in this work), $\mathbf{E}$ is the Green–Lagrange strain tensor, $\delta\mathbf{E}$ is its variation with respect to virtual displacements $\delta\mathbf{y}$, and $\mathbf{S}$ is the energetically conjugate second Piola–Kirchhoff stress tensor.

In the case of shells, the 3D continuum description is reduced to that of the shell midsurface, and the transverse normal stress is neglected. Furthermore, the Kirchhoff–Love theory assumes that the shell director remains normal to its middle surface during the deformation, which implies that the transverse shear strains are zero. As a result, only in-plane stress and strain tensors are considered, and the indices $\alpha = 1,2$ and $\beta = 1,2$ are employed to denote their components. We denote by Γ_0^s the shell midsurface in the undeformed reference configuration, h_{th} is the (variable) shell thickness, and $\xi_3 \in [-0.5h_{th}, 0.5h_{th}]$ is the through-thickness coordinate.

We introduce the following standard shell kinematic quantities and relationships (see [126; 39] for more details):

$$E_{\alpha\beta} = \varepsilon_{\alpha\beta} + \xi_3\kappa_{\alpha\beta}, \tag{50}$$

$$\varepsilon_{\alpha\beta} = \frac{1}{2}\left(\mathbf{g}_\alpha\cdot\mathbf{g}_\beta - \mathbf{G}_\alpha\cdot\mathbf{G}_\beta\right), \tag{51}$$

$$\kappa_{\alpha\beta} = -\frac{\partial\mathbf{g}_\alpha}{\partial\xi_\beta}\cdot\mathbf{g}_3 - (-\frac{\partial\mathbf{G}_\alpha}{\partial\xi_\beta}\cdot\mathbf{G}_3), \tag{52}$$

$$\mathbf{g}_\alpha = \frac{\partial\mathbf{x}}{\partial\xi_\alpha}, \tag{53}$$

$$\mathbf{G}_\alpha = \frac{\partial\mathbf{X}}{\partial\xi_\alpha}, \tag{54}$$

$$\mathbf{g}_3 = \frac{\mathbf{g}_1\times\mathbf{g}_2}{\|\mathbf{g}_1\times\mathbf{g}_2\|}, \tag{55}$$

$$\mathbf{G}_3 = \frac{\mathbf{G}_1\times\mathbf{G}_2}{\|\mathbf{G}_1\times\mathbf{G}_2\|}, \tag{56}$$

$$\mathbf{G}^\alpha = (\mathbf{G}_\alpha\cdot\mathbf{G}_\beta)^{-1}\mathbf{G}_\beta. \tag{57}$$

Here, $E_{\alpha\beta}$, $\varepsilon_{\alpha\beta}$, and $\kappa_{\alpha\beta}$ are the contravariant components of the in-plane Green–Lagrange strain, membrane strain, and curvature tensors, respectively. The spatial coordinates of the *shell midsurface* in the current and reference configurations are $\mathbf{x} = \mathbf{x}(\xi_1,\xi_2)$ and $\mathbf{X} = \mathbf{X}(\xi_1,\xi_2)$, parameterized by ξ_1 and ξ_2. The covariant surface basis vectors in the current and reference configurations are $\mathbf{g}_\alpha$ and $\mathbf{G}_\alpha$. The unit outward normal vectors to the shell midsurface in the current and reference configurations are $\mathbf{g}_3$ and $\mathbf{G}_3$. The contravariant surface basis vectors in the reference configuration are denoted by $\mathbf{G}^\alpha$.

We select the *local Cartesian basis* vectors as follows:

$$\bar{\mathbf{e}}_1 = \frac{\mathbf{G}_1}{\|\mathbf{G}_1\|}, \tag{58}$$

$$\bar{\mathbf{e}}_2 = \frac{\mathbf{G}_2 - (\mathbf{G}_2\cdot\mathbf{e}_1)\mathbf{e}_1}{\|\mathbf{G}_2 - (\mathbf{G}_2\cdot\mathbf{e}_1)\mathbf{e}_1\|}, \tag{59}$$

that is, the first local basis vector is the normalized first covariant basis vector in the reference configuration. The local Cartesian basis vectors $\overline{\mathbf{e}}_\alpha$ are used in expressing a constitutive relationship for the shell. Because the local basis is orthonormal, we make no distinction between covariant and contravariant quantities, which are expressed with respect to it.

With the above definitions, we calculate the components of the Green–Lagrange strain tensor and its variation in the local coordinate system as

$$\overline{E}_{\alpha\beta} = \overline{\varepsilon}_{\alpha\beta} + \xi_3 \overline{\kappa}_{\alpha\beta}, \tag{60}$$

$$\delta\overline{E}_{\alpha\beta} = \delta\overline{\varepsilon}_{\alpha\beta} + \xi_3 \delta\overline{\kappa}_{\alpha\beta}, \tag{61}$$

$$\overline{\varepsilon}_{\alpha\beta} = \varepsilon_{\gamma\delta}(\mathbf{G}^\gamma \cdot \overline{\mathbf{e}}_\alpha)(\mathbf{G}^\delta \cdot \overline{\mathbf{e}}_\beta), \tag{62}$$

$$\overline{\kappa}_{\alpha\beta} = \kappa_{\gamma\delta}(\mathbf{G}^\gamma \cdot \overline{\mathbf{e}}_\alpha)(\mathbf{G}^\delta \cdot \overline{\mathbf{e}}_\beta), \tag{63}$$

$$\delta\overline{\varepsilon}_{\alpha\beta} = \delta\varepsilon_{\gamma\delta}(\mathbf{G}^\gamma \cdot \overline{\mathbf{e}}_\alpha)(\mathbf{G}^\delta \cdot \overline{\mathbf{e}}_\beta), \tag{64}$$

$$\delta\overline{\kappa}_{\alpha\beta} = \delta\kappa_{\gamma\delta}(\mathbf{G}^\gamma \cdot \overline{\mathbf{e}}_\alpha)(\mathbf{G}^\delta \cdot \overline{\mathbf{e}}_\beta). \tag{65}$$

The variations $\delta\varepsilon_{\gamma\delta}$ and $\delta\kappa_{\gamma\delta}$ may be computed directly by taking the variational derivatives of the expressions given by Eqs. (51) and (52) with respect to the displacement vector.

We define the vectors of membrane strain and curvature components in the local coordinate system as

$$\overline{\boldsymbol{\varepsilon}} = \begin{bmatrix} \overline{\varepsilon}_{11} \\ \overline{\varepsilon}_{22} \\ \overline{\varepsilon}_{12} \end{bmatrix} \tag{66}$$

and

$$\overline{\boldsymbol{\kappa}} = \begin{bmatrix} \overline{\kappa}_{11} \\ \overline{\kappa}_{22} \\ \overline{\kappa}_{12} \end{bmatrix}, \tag{67}$$

together with a Green–Lagrange strain vector

$$\overline{\mathbf{E}} = \overline{\boldsymbol{\varepsilon}} + \xi_3 \overline{\boldsymbol{\kappa}}. \tag{68}$$

We assume St. Venant–Kirchhoff material law and write the following stress–strain relationship in the local coordinate system:

$$\overline{\mathbf{S}} = \overline{\mathbf{C}}\,\overline{\mathbf{E}}, \tag{69}$$

where $\overline{\mathbf{S}}$ is a vector of components of the second Piola–Kirchhoff stress tensor in the local coordinate system, and $\overline{\mathbf{C}}$ is a constitutive material matrix, which is symmetric. Introducing Eqs. (68) and (69) into the expression for the internal virtual work given by Eq. (49), we obtain

$$\delta W_{int} = -\int_{\Omega_0^s} \delta\overline{\mathbf{E}} \cdot \overline{\mathbf{S}}\, \mathrm{d}\Omega \tag{70}$$

$$= -\int_{\Gamma_0^s} \left(\int_{h_{th}} \delta\overline{\mathbf{E}} \cdot \overline{\mathbf{C}}\,\overline{\mathbf{E}}\, d\xi_3 \right) \mathrm{d}\Gamma \tag{71}$$

$$\begin{aligned} = &-\int_{\Gamma_0^s} \delta\overline{\boldsymbol{\varepsilon}} \cdot \left(\left(\int_{h_{th}} \overline{\mathbf{C}}\, d\xi_3 \right) \overline{\boldsymbol{\varepsilon}} + \left(\int_{h_{th}} \xi_3 \overline{\mathbf{C}}\, d\xi_3 \right) \overline{\boldsymbol{\kappa}} \right) \mathrm{d}\Gamma \\ &-\int_{\Gamma_0^s} \delta\overline{\boldsymbol{\kappa}} \cdot \left(\left(\int_{h_{th}} \xi_3 \overline{\mathbf{C}}\, d\xi_3 \right) \overline{\boldsymbol{\varepsilon}} + \left(\int_{h_{th}} {\xi_3}^2 \overline{\mathbf{C}}\, d\xi_3 \right) \overline{\boldsymbol{\kappa}} \right) \mathrm{d}\Gamma. \end{aligned} \tag{72}$$

For a general orthotropic material,

$$\overline{\mathbf{C}}_{ort} = \begin{bmatrix} \dfrac{E_1}{(1-\nu_{12}\nu_{21})} & \dfrac{\nu_{21}E_1}{(1-\nu_{12}\nu_{21})} & 0 \\ \dfrac{\nu_{12}E_2}{(1-\nu_{12}\nu_{21})} & \dfrac{E_2}{(1-\nu_{12}\nu_{21})} & 0 \\ 0 & 0 & G_{12} \end{bmatrix}. \tag{73}$$

In Eq. (73), E_1 and E_2 are the Young's moduli in the directions defined by the local basis vectors, ν_{12} and ν_{21} are the Poisson ratios, G_{12} is the shear modulus, and $\nu_{21}E_1 = \nu_{12}E_2$ to ensure the symmetry of the constitutive material matrix $\overline{\mathbf{C}}_{ort}$. In the case of an isotropic material, $E_1 = E_2 = E$, $\nu_{21} = \nu_{12} = \nu$, and $G_{12} = E/(2(1+\nu))$.

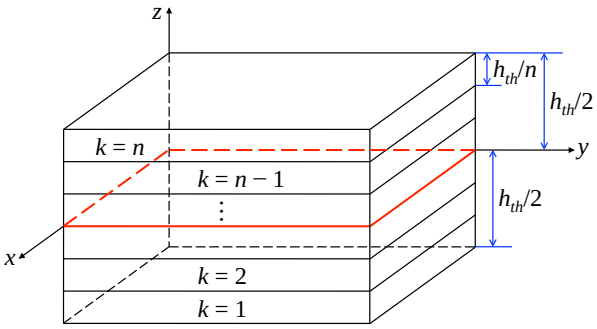


Fig. 40 Schematic of a composite laminate.

In the case of composite materials, which are used in the manufacturing of modern wind-turbine blades, we assume that the structure is composed of a set of plies, each modeled as an orthotropic material. We use the classical laminated-plate theory [127], and homogenize the material through-thickness constitutive behavior for a given composite ply layout. Let k denotes the k^{th} ply (or lamina) and let n be the total number of plies (see Figure 40). We assume each ply has the same thickness h_{th}/n. Pre-integrating through the shell thickness in Eq. (72), the extensional stiffness **A**, coupling stiffness **B**, and bending stiffness **D** are given by

$$\mathbf{A} = \int_{h_{th}} \overline{\mathbf{C}}\, d\xi_3 = \frac{h_{th}}{n} \sum_{k=1}^{n} \overline{\mathbf{C}}_k, \tag{74}$$

$$\mathbf{B} = \int_{h_{th}} \xi_3 \overline{\mathbf{C}}\, d\xi_3 = \frac{h_{th}^2}{n^2} \sum_{k=1}^{n} \overline{\mathbf{C}}_k \left(k - \frac{n}{2} - \frac{1}{2} \right), \tag{75}$$

$$\mathbf{D} = \int_{h_{th}} {\xi_3}^2 \overline{\mathbf{C}}\, d\xi_3 = \frac{h_{th}^3}{n^3} \sum_{k=1}^{n} \overline{\mathbf{C}}_k \left(\left(k - \frac{n}{2} - \frac{1}{2} \right)^2 + \frac{1}{12} \right), \tag{76}$$

where

$$\overline{\mathbf{C}}_k = \mathbf{T}^T(\phi_k)\,\overline{\mathbf{C}}_{ort}\,\mathbf{T}(\phi_k), \tag{77}$$

$$\mathbf{T}(\phi) = \begin{bmatrix} \cos^2\phi & \sin^2\phi & \sin\phi\cos\phi \\ \sin^2\phi & \cos^2\phi & -\sin\phi\cos\phi \\ -2\sin\phi\cos\phi & 2\sin\phi\cos\phi & \cos^2\phi - \sin^2\phi \end{bmatrix}. \tag{78}$$

In the above equations, ϕ is the fiber orientation angle in each ply, Eq. (77) transforms $\overline{\mathbf{C}}_{ort}$ from the principal material coordinates to the laminate coordinates (defined by the local Cartesian basis) for each ply, and $\overline{\mathbf{C}}_k$ is constant within each ply.

With the above definitions, the expression for the internal virtual work for a composite shell may now be compactly written as

$$\delta W_{int} = -\int_{\Gamma_0^s} \delta\bar{\boldsymbol{\varepsilon}} \cdot \left(\mathbf{A}\bar{\boldsymbol{\varepsilon}} + \mathbf{B}\bar{\boldsymbol{\kappa}}\right) \mathrm{d}\Gamma - \int_{\Gamma_0^s} \delta\bar{\boldsymbol{\kappa}} \cdot \left(\mathbf{B}\bar{\boldsymbol{\varepsilon}} + \mathbf{D}\bar{\boldsymbol{\kappa}}\right) \mathrm{d}\Gamma. \tag{79}$$

Remark 12 *Setting $n = 1$ and $\overline{\mathbf{C}}_k = \overline{\mathbf{C}}_{ort}$ in Eqs. (74)–(76), we get $\mathbf{B} = \mathbf{0}$ and*

$$\mathbf{A} = h_{th}\overline{\mathbf{C}}_{ort}, \tag{80}$$

$$\mathbf{D} = \frac{h_{th}^3}{12}\overline{\mathbf{C}}_{ort}, \tag{81}$$

which are the classical membrane and bending stiffnesses for an orthotropic shell.

6.2 The Bending-Strip Method and a Complete Variational Statement of the Structural Mechanics Problem

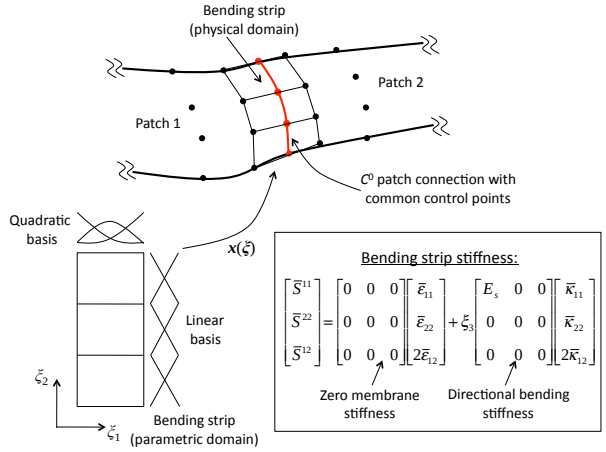


Fig. 41 Schematic of the bending-strip method.

The expression for of the internal virtual work given by Eq. (79) is only meaningful when the shell midsurface is described using a smooth geometrical mapping. In the case when the regularity of the mapping reduces to the C^0 level, the terms involving the curvature tensors, which rely on the second derivatives of the geometrical mapping, lead to non-integrable singularities, and the formulation may not be used as is. However, for complex structures, the geometry definition often requires that the continuity of the geometrical mapping is reduced to the C^0 level (think of a trailing edge of an airfoil or an I-beam, the latter being a non-manifold surface).

In [13], a method was prosed to handle complex multi-patch shell structures in the context of the rotation-free Kirchhoff–Love theory, called the "bending-strip method" [3]. The main idea behind the method, illustrated in Figure 41, consists of the following. It is assumed that the shell structure is comprised of smooth subdomains, such as NURBS patches, that are joined with C^0-continuity. In addition, thin strips of fictitious material, also modeled as surface NURBS patches, are placed at patch intersections. The triples of control points at the patch interface, consisting of a shared control point and one on each side, are extracted and used as a control mesh for the bending strips. The parametric domain of each bending strip consists of one quadratic element in the direction transverse to the interface and, for simplicity and computational efficiency, of as many linear elements as necessary to accommodate all the control points along the length of the strip. The material is assumed to have zero mass, zero membrane stiffness, and *non-zero bending stiffness only in the direction transverse to the interface*. The transverse direction may be obtained using the local basis construction given by Eqs. (58) and (59), however, other options may be explored.

Let Γ_0^s and Γ_t^s denote the structure midsurface in the reference and deformed configurations, respectively, and let Γ_0^b denote the bending-strip domain, where Γ_0^b is a union of the bending-strip patch subdomains. Let $\mathcal{S}_y^h$ and $\mathcal{V}_y^h$ denote the discrete trial and test function spaces for the structural problem. We seek the displacement of the shell midsurface $\mathbf{y}^h \in \mathcal{S}_y^h$, such that $\forall\ \delta\mathbf{y}^h \in \mathcal{V}_y^h$:

$$\begin{aligned} &\int_{\Gamma_t^s} \delta\mathbf{y}^h \cdot \rho h_{th} \left(\left.\frac{\partial^2 \mathbf{y}^h}{\partial t^2}\right|_{\mathbf{X}} - \mathbf{f}^h \right) \mathrm{d}\Gamma \\ &+ \int_{\Gamma_0^s} \delta\bar{\boldsymbol{\varepsilon}}^h \cdot \left(\mathbf{A}\bar{\boldsymbol{\varepsilon}}^h + \mathbf{B}\bar{\boldsymbol{\kappa}}^h\right) \mathrm{d}\Gamma \\ &+ \int_{\Gamma_0^s} \delta\bar{\boldsymbol{\kappa}}^h \cdot \left(\mathbf{B}\bar{\boldsymbol{\varepsilon}}^h + \mathbf{D}\bar{\boldsymbol{\kappa}}^h\right) \mathrm{d}\Gamma \\ &+ \int_{\Gamma_0^b} \delta\bar{\boldsymbol{\kappa}}^h \cdot \mathbf{D}^b\bar{\boldsymbol{\kappa}}^h \,\mathrm{d}\Gamma - \int_{(\Gamma_t^s)_h} \delta\mathbf{y}^h \cdot \mathbf{h}^h \,\mathrm{d}\Gamma = 0. \end{aligned} \tag{82}$$

In the above, the superscript h denotes all the discrete quantities, ρ is the structural mass density in the deformed con-

[3] The method in its current form was developed and implemented at the University of California, San Diego, when J. Kiendl, at the time a PhD student in the group of K.-U. Bletzinger at the Technical University of Munich, was visiting the research group of Y. Bazilevs. The method has similarities with the concept of "continuity patches", introduced by K.-U. Bletzinger and collaborators in [128].

figuration, $\mathbf{f}$ is the body force (e.g. gravity), and $\mathbf{h}$ is the prescribed surface traction on $(\Gamma_t^s)_h$. The terms on the first line of Eq. (82) represent the inertial and body forces. The terms on the second and third lines were derived in the previous section. The term $\int_{\Gamma_0^b} \delta\overline{\boldsymbol{\kappa}}^h \cdot \mathbf{D}^b \overline{\boldsymbol{\kappa}}^h \, \mathrm{d}\Gamma$ is penalty-like, and represents the contribution of the bending strips to the structural formulation given by Eq. (82). Here $\mathbf{D}^b$ is the bending stiffness of the strips:

$$\mathbf{D}^b = \frac{h_{th}^3}{12}\mathbf{C}^b, \tag{83}$$

where

$$\mathbf{C}^b = \begin{bmatrix} E_s & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \tag{84}$$

and E_s is the scalar bending-strip stiffness, typically chosen as a multiple of the local Young's modulus of the shell. This design of the material constitutive matrix ensures that the bending strips add no extra stiffness to the structure. They only penalize the change in the angle during the deformation between the triples of control points at the patch interface. The stiffness E_s must be high enough so that the change in angle is within an acceptable tolerance. However, if E_s is chosen too high, the global stiffness matrix becomes badly conditioned, which may lead to divergence in the computations.

Remark 13 *Because of the structure of the bending-strip term in Eq. (82), the method may be interpreted as a physically-motivated penalty formulation.*

Remark 14 *In IGA, the possibility to employ smooth surface descriptions directly in analysis has led to the development of new shell element formulations. Besides the references cited in this section, the reader is referred to [45; 46; 47; 38; 129] for relevant work on shells. We would also like to note that references [45; 46; 47] predate the development of IGA.*

6.3 Time Integration of the Structural Mechanics Equations

In the case of wind-turbine rotors, the structural motions are dominated by the rotation of the blades around the hub axis. In [15], the authors proposed to take advantage of this fact and modify a class of standard time integration techniques to exactly account for the rotational part of the structural motion.

For this, as a first step, it is useful to decompose the structural displacement $\mathbf{y}$ into its rotation and deflection components as

$$\mathbf{y} = \mathbf{y}_\theta + \mathbf{y}_d. \tag{85}$$

The rotational component of the displacement may be computed as

$$\mathbf{y}_\theta = (\mathbf{R}(\theta) - \mathbf{I})\,(\mathbf{X} - \mathbf{X}_0)\,, \tag{86}$$

where $\mathbf{X}$ are the coordinates of the structure reference configuration, $\mathbf{X}_0$ is a fixed point, θ is the time-dependent angle of rotation, $\mathbf{R}(\theta)$ is the rotation matrix, and $\mathbf{I}$ is the identity matrix. We specialize to the case of rotation about the z-axis, which gives

$$\mathbf{R}(\theta) = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{87}$$

The total structural velocity and acceleration may be computed as

$$\left.\frac{\partial \mathbf{y}^h}{\partial t}\right|_{\mathbf{X}} = \dot{\mathbf{y}} = \dot{\mathbf{y}}_\theta + \dot{\mathbf{y}}_d = \dot{\mathbf{R}}(\theta)(\mathbf{X} - \mathbf{X}_0) + \dot{\mathbf{y}}_d, \tag{88}$$

$$\left.\frac{\partial^2 \mathbf{y}^h}{\partial t^2}\right|_{\mathbf{X}} = \ddot{\mathbf{y}} = \ddot{\mathbf{y}}_\theta + \ddot{\mathbf{y}}_d = \ddot{\mathbf{R}}(\theta)(\mathbf{X} - \mathbf{X}_0) + \ddot{\mathbf{y}}_d, \tag{89}$$

where

$$\dot{\mathbf{R}}(\theta) = \begin{bmatrix} -\sin\theta & -\cos\theta & 0 \\ \cos\theta & -\sin\theta & 0 \\ 0 & 0 & 0 \end{bmatrix}\dot{\theta}, \tag{90}$$

$$\ddot{\mathbf{R}}(\theta) = \begin{bmatrix} -\cos\theta & \sin\theta & 0 \\ -\sin\theta & -\cos\theta & 0 \\ 0 & 0 & 0 \end{bmatrix}\dot{\theta}^2 + \begin{bmatrix} -\sin\theta & -\cos\theta & 0 \\ \cos\theta & -\sin\theta & 0 \\ 0 & 0 & 0 \end{bmatrix}\ddot{\theta}. \tag{91}$$

We repeat this decomposition at the discrete level, where we operate directly on the nodal or control-point displacement degrees of freedom. For this, we let $\mathbf{U}$, $\dot{\mathbf{U}}$, and $\ddot{\mathbf{U}}$ be the vectors of nodal or control point displacements, velocities, and accelerations, respectively. We set

$$\mathbf{U} = \mathbf{U}_\theta + \mathbf{U}_d, \tag{92}$$

$$\dot{\mathbf{U}} = \dot{\mathbf{U}}_\theta + \dot{\mathbf{U}}_d, \tag{93}$$

$$\ddot{\mathbf{U}} = \ddot{\mathbf{U}}_\theta + \ddot{\mathbf{U}}_d, \tag{94}$$

where $\mathbf{U}_\theta$, $\dot{\mathbf{U}}_\theta$, and $\ddot{\mathbf{U}}_\theta$ are given by

$$\mathbf{U}_\theta = (\mathbf{R}(\theta) - \mathbf{I})\,(\mathbf{X} - \mathbf{X}_0), \tag{95}$$

$$\dot{\mathbf{U}}_\theta = \dot{\mathbf{R}}(\theta)(\mathbf{X} - \mathbf{X}_0), \tag{96}$$

$$\ddot{\mathbf{U}}_\theta = \ddot{\mathbf{R}}(\theta)(\mathbf{X} - \mathbf{X}_0). \tag{97}$$

The above Eqs. (95)–(97) present an exact relationship between the nodal or control point displacements, velocities, and accelerations corresponding to the rotational motion. To relate the deflection degrees of freedom between time levels t_n and t_{n+1}, we make use of the standard Newmark formulas (see e.g., [130]):

$$\dot{\mathbf{U}}_d^{n+1} = \dot{\mathbf{U}}_d^n + \Delta t\left((1-\gamma)\ddot{\mathbf{U}}_d^n + \gamma\ddot{\mathbf{U}}_d^{n+1}\right), \tag{98}$$

$$\mathbf{U}_d^{n+1} = \mathbf{U}_d^n + \Delta t\dot{\mathbf{U}}_d^n + \frac{\Delta t^2}{2}\left((1-2\beta)\ddot{\mathbf{U}}_d^n + 2\beta\ddot{\mathbf{U}}_d^{n+1}\right), \tag{99}$$

where γ and β are the time integration parameters chosen to maintain second-order accuracy and unconditional stability of the method.

Combining exact rotations given by Eqs. (95)–(97) and time-discrete deflections given by Eqs. (98)–(99), we obtain the following modified Newmark formulas for the total discrete solution:

$$\dot{\mathbf{U}}^{n+1} = \Delta\dot{\mathbf{R}}^{n+1}(\mathbf{X}-\mathbf{X}_0) + \dot{\mathbf{U}}^n + \Delta t\left((1-\gamma)\ddot{\mathbf{U}}^n + \gamma\ddot{\mathbf{U}}^{n+1}\right), \quad (100)$$

$$\mathbf{U}^{n+1} = \Delta\mathbf{R}^{n+1}(\mathbf{X}-\mathbf{X}_0) + \mathbf{U}^n + \Delta t\dot{\mathbf{U}}^n + \frac{\Delta t^2}{2}\left((1-2\beta)\ddot{\mathbf{U}}^n + 2\beta\ddot{\mathbf{U}}^{n+1}\right), \quad (101)$$

where

$$\Delta\dot{\mathbf{R}}^{n+1} = \dot{\mathbf{R}}^{n+1} - \left[\dot{\mathbf{R}}^n + \Delta t\left((1-\gamma)\ddot{\mathbf{R}}^n + \gamma\ddot{\mathbf{R}}^{n+1}\right)\right], \quad (102)$$

and

$$\Delta\mathbf{R}^{n+1} = \mathbf{R}^{n+1} - \left[\mathbf{R}^n + \Delta t\dot{\mathbf{R}}^n + \frac{\Delta t^2}{2}\left((1-2\beta)\ddot{\mathbf{R}}^n + 2\beta\ddot{\mathbf{R}}^{n+1}\right)\right]. \quad (103)$$

We employ Eqs. (100)–(103) for the time integration of the structure.

Remark 15 *In the case of no rotation, for which* $\mathbf{R}$ *is an identity, Eqs. (100)–(103) reduce to the standard Newmark formulas. In the case of no deflection, pure rotation is likewise recovered.*

7 FSI Coupling and Aerodynamics Mesh Update

In this section we briefly summarize our FSI coupling procedures for wind-turbine simulations. The fluid and structural equations are integrated in time using the generalized-alpha method. In the case of the structure, the modified Newmark formulas given by Eqs. (100)–(103) are employed to enhance the accuracy of the time integration procedures in the presence of large rotation. Within each time step, the coupled equations are solved using an inexact Newton approach. For every Newton iteration the following steps are performed. 1. We obtain the fluid solution increment holding the structure and mesh fixed. 2. We update the fluid solution, compute the aerodynamic force on the structure and compute the structural solution increment. The aerodynamic force at control points or nodes is computed using the conservative definition (see, e.g., [131; 132] for the importance of using the conservative definitions of fluxes near essential boundaries and in coupled problems). 3. We update the structural solution and use elastic mesh motion to update the fluid domain velocity and position. We note that only the deflection part of the mesh motion is computed using linear elastostatics, while the rotation part is computed exactly. This three-step iteration is repeated until convergence to an appropriately coupled discrete solution is achieved. The proposed approach, also referred to as "block-iterative coupling" (see [124; 67; 57] for the terminology), is stable because the wind-turbine blades are relatively heavy structures.

Remark 16 *For this coupling strategy the fluid and structural meshes may or may not be conforming. In the case of conforming meshes, the conservative nodal or control-point traction vector from the fluid side is applied directly at the nodes or control points of the structure, while the structural nodal or control-point kinematic data is applied directly to the nodes or control points of the fluid. When the fluid and structural meshes are non-conforming, additional projection of the traction and kinematic data is necessary before they are transferred to the neighboring subdomain. In this paper we only present conforming mesh simulations.*

We conclude this section with the discussion of a special technique we devised in [15] to update the kinematics (position and velocity) of the fluid mesh. Typically, one employes the equations of linear elastostatics subject to dynamic boundary conditions coming from the structural displacement to update the position and velocity of the fluid mesh (see, for example, the elasticity-based mesh-moving method introduced in [88; 89]). In the case of wind turbines, which are dominated by rotation, this may not be a preferred procedure due to the fact that the linear elastostatics operator does not vanish on large rotational motions. This, in turn, may lead to the loss of the fluid mesh quality if one plans to simulate the FSI problem for many revolutions of the wind-turbine rotor. As a result, for the present application, we modify our fluid mesh motion strategy as follows. We take advantage of the fact that the structural displacement vector is already decomposed into the rotation and deflection parts. As a result, as the increment of the structural displacement is computed, we extract the deflection part, apply the elasticity-based mesh-moving method [88; 89] to computation of just the deflection part of the mesh displacement, rotate the (deformed) mesh from the previous time level to the current time, and add the mesh deflection increment to obtain its current position. For a precise mathematical formulation of this procedure see [15].

Remark 17 *For a variety of other mesh update strategies the reader is referred to [90; 91; 84].*

8 Simulation of the Micon 65/13M Wind Turbine with a CX-100 Blade

This section is adapted from [95]. We simulate the Micon 65/13M wind turbine at field test conditions [133]. Micon 65/13M is a three-blade, horizontal-axis, fixed-pitch, upwind turbine with the total rotor diameter of 19.3 m and rated power of 100 kW. The hub is located at the height of 23 m. The wind turbine stands on a tubular steel tower, with a base diameter of 1.9 m. The drive train generator operates at 1200 rpm, while the rotor spins at a nominal speed of 55 rpm. The Micon 65/13M wind turbine was used for the Long-Term Inflow and Structural Testing (LIST) program [134] initiated by Sandia National Laboratories in 2001 to explore the use of carbon fiber in wind

turbine blades. Three experimental blade prototypes, GX-100, CX-100 and TX-100, were developed specifically for this project. We use the CX-100 conventional carbon-spar blade design [133; 135]. The NREL S821, S819 and S820 airfoils are used to define the blade geometry. The details of the blade geometry definition are provided in Table 7.

RNodes	Chord	AeroTwst	Airfoil
0.200	0.356	29.6	Cylinder
0.600	0.338	24.8	Cylinder
1.000	0.569	20.8	Cylinder
1.400	0.860	17.5	NREL S821
1.800	1.033	14.7	NREL S821
2.200	0.969	12.4	NREL S821
3.200	0.833	8.3	NREL S821
4.200	0.705	5.8	NREL S819
5.200	0.582	4.0	NREL S819
6.200	0.463	2.7	NREL S819
7.200	0.346	1.4	NREL S819
8.200	0.232	0.4	NREL S819
9.000	0.120	0.0	NREL S820

Table 7 CX-100 blade geometry definition with "RNodes" (m), "Chord" (m), "AeroTwst" (°), and "Airfoil" type.

8.1 Eigenfrequency Analysis of the CX-100 Blade

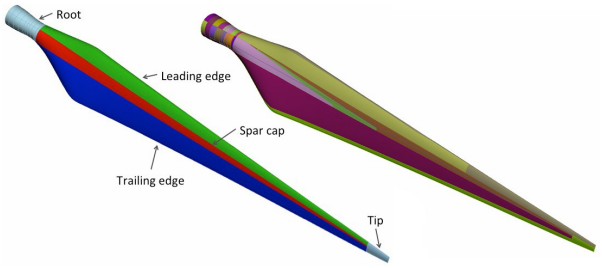


Fig. 42 Left: Five primary sections of the CX-100 blade; Right: 32 distinct material zones of the CX-100 blade.

The blade structure is comprised of five primary sections: leading edge, trailing edge, root, spar cap, and shear web. The sections are shown in Figure 42. Each section is further subdivided into zones, each consisting of a multi-layer composite layup. There is a total of 32 zones with constant total thickness and unique laminate stacking. The effective material properties for each of the zones are computed using the procedures described Section 6.1. All 32 zones are identified on the blade surface and are shown in Figure 42. For more details of the material composition of the CX-100 blade see [95].

	Control points	Elements
Mesh 1	3,469	1,846
Mesh 2	7,411	4,647
Mesh 3	25,896	18,611

Table 8 NURBS blade meshes used in the eigenfrequency analysis.

	Mode 1	Mode 2	Mode 3
Mesh 1	8.28	15.92	19.26
Mesh 2	8.22	15.61	18.21
Mesh 3	8.22	15.6	18.01
Experiment	7.6–8.2	15.7–18.1	20.2–21.3

Table 9 Comparison of experimentally measured and computed natural frequencies (in Hz) for the free case. Mode 1 is the first flapwise mode, Mode 2 is the first edgewise mode, and Mode 3 is the second flapwise mode.

	Mode 1	Mode 2	Mode 3
Mesh 1	4.33	11.82	19.69
Mesh 2	4.29	11.61	19.08
Mesh 3	4.27	11.54	18.98
Experiment	4.35	11.51	20.54

Table 10 Comparison of experimentally measured and computed natural frequencies (in Hz) for the clamped case. Modes 1-3 are the first three flapwise bending modes.

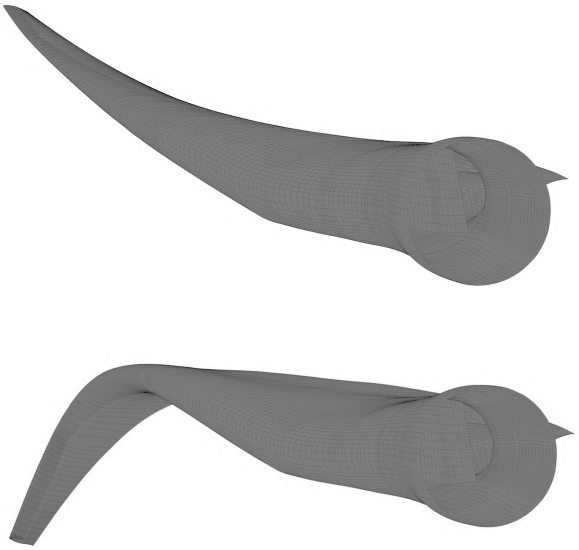


Fig. 43 First flapwise bending mode (top) and second flapwise bending mode (bottom) for the clamped case.

We perform eigenfrequency calculations of the CX-100 blade using three quadratic NURBS meshes. The coarsest mesh has 1,846 elements, while the finest mesh has 18,611 elements. The mesh statistics are summarized in Table 8. The eigenfrequency results are compared with the experimental data from [136; 137]. We compute the case with free boundary conditions and the case when the blade is clamped at the root. In both cases, the computed natural frequencies

are in good agreement with the experimental data (see Tables 9 and 10). The medium mesh shows a good balance between the computational cost and accuracy of the results. For this reason, this mesh is chosen for the FSI computations presented in what follows. The mode shapes computed using the medium mesh for the clamped case are shown in Figure 43.

8.2 Aerodynamics and FSI Computations of the Micon 65/13M Wind Turbine

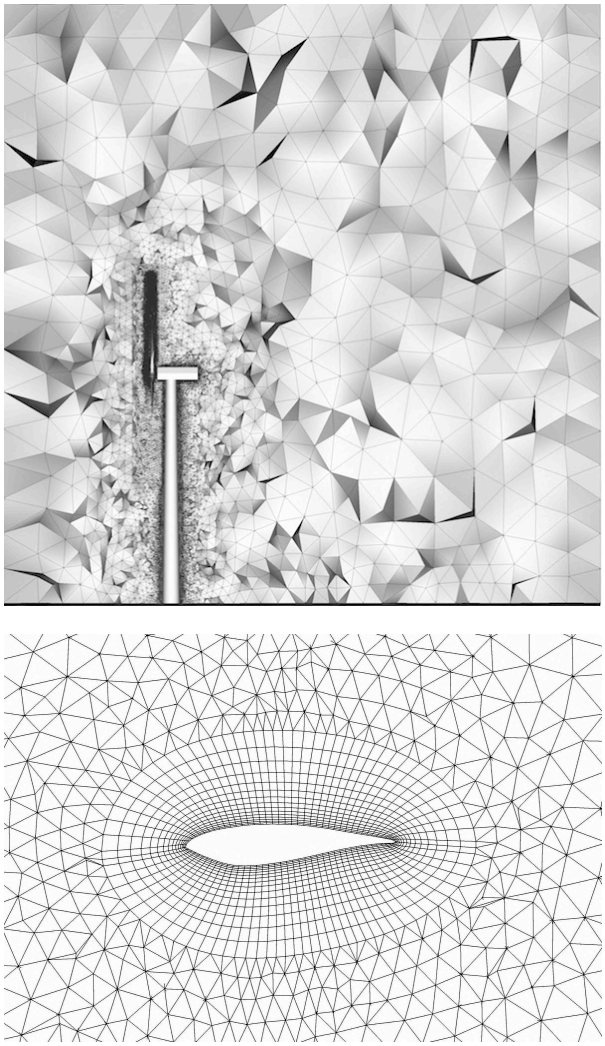


Fig. 44 Top: The computational domain and problem mesh with the refined inner region for better flow resolution near the rotor; Bottom: 2D blade cross-section at $r/R = 70\%$ and the boundary-layer mesh.

In this section, we present aerodynamic and FSI simulations of the full Micon 65/13M wind turbine. For both cases, a constant inflow wind speed of 10.5 m/s and fixed rotor speed of 55 rpm are prescribed. These correspond to the operating conditions reported for the field tests in [133]. The air density and viscosity are 1.23 kg/m^3 and 1.78×10^{-5} kg/(m·s), respectively. Zero traction boundary conditions are prescribed at the outflow and no-penetration boundary conditions are prescribed at the top, bottom, and side surfaces of the outer (stationary) computational domain. No-slip boundary conditions are prescribed at the rotor, nacelle, and tower, and are imposed weakly.

Figure 44 shows the computational domain and mesh used in this study. The mesh consists of 5,134,916 linear elements, which are triangular prisms in the rotor boundary layers and tetrahedra everywhere else in the domain. The mesh is refined in the rotor and tower regions for better flow resolution near the wind turbine. The size of the first element in the wall-normal direction is 0.002 m, and 15 layers of prismatic elements were generated with a growth ratio of 1.2. Figure 44 shows a 2D blade cross-section at 70% spanwise station to illustrate the boundary-layer mesh used in the computations. The time-step size is set to 3.0×10^{-5} s.

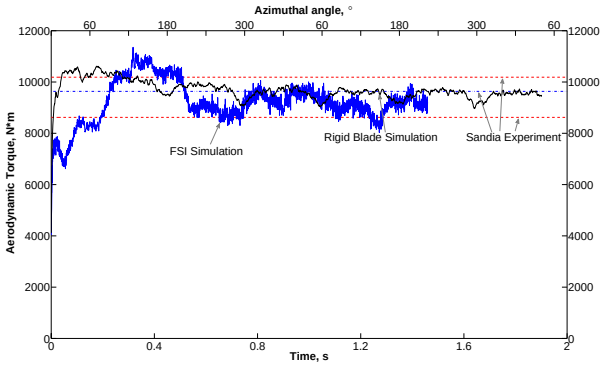


Fig. 45 Aerodynamic torque history for the FSI and rigid-blade simulations. The experimental range for the aerodynamic torque and its average value are provided for comparison and are plotted using dashed lines.

In Figure 45 the time history of the aerodynamic torque is plotted. As can be seen from the plot, using FSI, we capture the high frequency oscillations caused by the bending and torsional motions of the blades. In the case of the rigid blade the only high-frequency oscillations in the torque curve are due to the trailing-edge turbulence. For the rigid blade case the effect of the tower on the aerodynamic torque is more pronounced, while in the case of FSI it is not as visible due to the relatively high torque oscillations. The "dips" in the aerodynamic torque can be seen at 60°, 180°, and 300° azimuthal angle, which is precisely when one of the three blades is passing the tower.

The computed values of the aerodynamic torque are plotted together with field test results from [133]. The upper and lower dashed lines indicate the aerodynamic torque bounds, while the middle dashed line gives its average value. Both the aerodynamic and FSI results compare very well with the field test data.

Figure 46 shows the relative wind speed at the 70% spanwise station rotated to the reference configuration to illus-

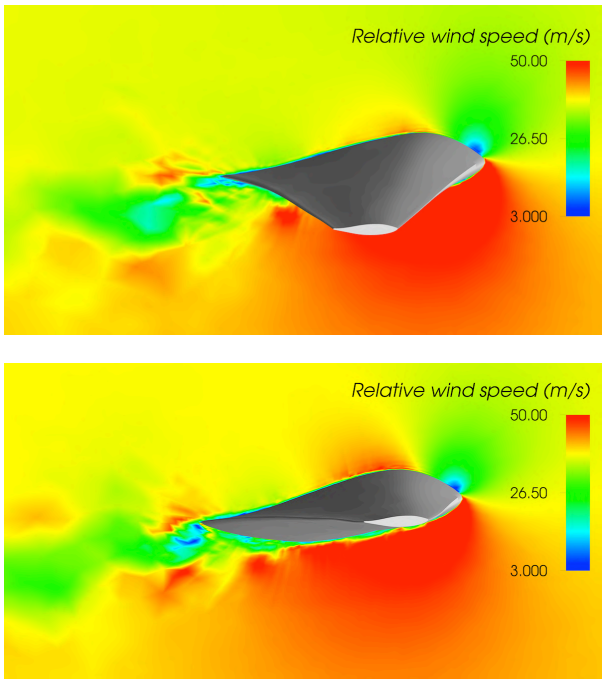


Fig. 46 Relative wind speed at the 70% spanwise station for the FSI simulation at $t = 0.86$ s (top) and $t = 1.06$ s (bottom). The blade deflection is clearly visible.

trate the blade deflection and complexity of boundary-layer turbulent flow. Figure 47 shows the flow field as the blade passes the tower.

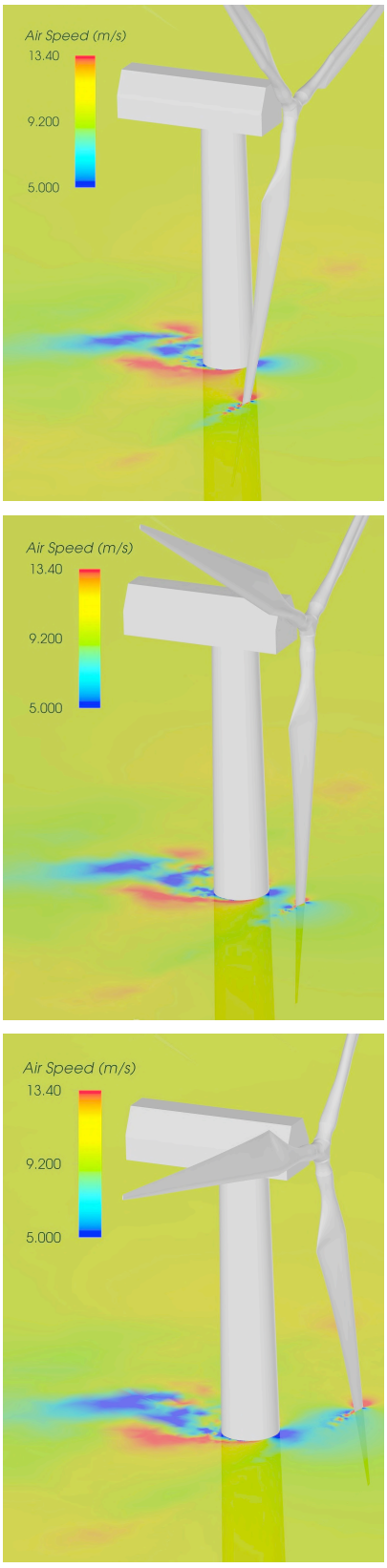
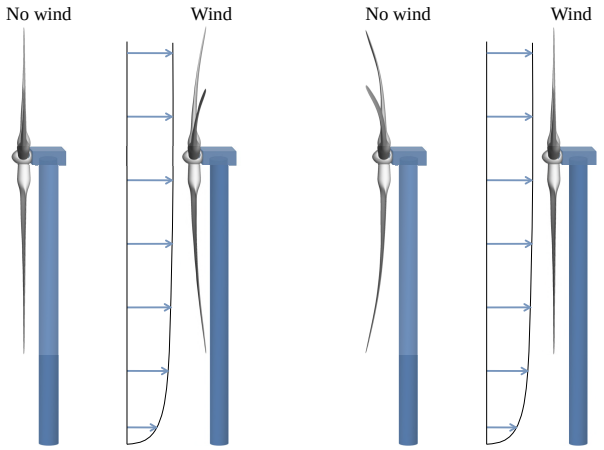


Fig. 47 Wind speed contours at 80% spanwise station as the blade passes the tower.

9 Pre-Bending of the Wind-Turbine Blades



(a) Without pre-bending (b) With pre-bending

Fig. 48 Using pre-bent blades to ensure tower clearance and rotor operation in its design configuration.

The rotor blades of a wind turbine need to be designed such that they do not strike the tower as the rotor turns in strong winds. This may be accomplished with blade pre-bending. In this case, the blades are manufactured to flex toward the wind when the rotor is mounted on the tower. Once the blades are exposed to the wind, and the rotor starts turning, the blades are straightened to achieve their design shape. This situation is graphically illustrated in Figure 48. Besides tower clearance, pre-bending of the blades engenders additional benefits. For example, the blades need not be quite as rigid because the amount of allowable deflection is greater. This makes it possible to use less material overall, and fewer processed materials, resulting in lighter and more economical blades. Pre-bending of the blades also results in a more compact nacelle design. During operation, the pre-bent blades straighten to their designed configuration, which is typically optimized for best possible aerodynamic performance.

Given the above advantages, it is important that one is able to determine the correct pre-bent shape given the blade structural and aerodynamic design, and the wind-turbine operating conditions (i.e., wind and rotor speeds). In [14], we proposed a method that makes use of standalone computational fluid and structural mechanics procedures to obtain a pre-bent shape of the wind-turbine blades. The main idea consists of performing an aerodynamics simulation of a rigidly-rotating rotor to obtain the aerodynamic load acting on the blade. Given the aerodynamic and inertial loads acting in the design configuration, a stress-free pre-bent blade configuration is found using a simple iterative procedure that requires a sequence of structural mechanics simulations. Note that in the proposed approach the aerodynamic and structural computations are decoupled, which avoids the challenges involved in solving the coupled FSI problem. In this section, we summarize the method and show the supporting computations. We follow the developments of [14].

9.1 Problem Statement and the Pre-bending Algorithm

We begin with the statement of virtual work for the structure from Eq. (82), where only the stress terms are left on the left-hand-side: find the displacement of the shell midsurface $\mathbf{y}^h \in \mathcal{S}_y^h$, such that for $\forall\ \delta\mathbf{y}^h \in \mathcal{V}_y^h$:

$$\begin{aligned}
&\int_{\Gamma_0^s} \delta\overline{\boldsymbol{\varepsilon}}^h \cdot \left(\mathbf{A}\overline{\boldsymbol{\varepsilon}}^h + \mathbf{B}\overline{\boldsymbol{\kappa}}^h\right) \mathrm{d}\Gamma \\
&+ \int_{\Gamma_0^s} \delta\overline{\boldsymbol{\kappa}}^h \cdot \left(\mathbf{B}\overline{\boldsymbol{\varepsilon}}^h + \mathbf{D}\overline{\boldsymbol{\kappa}}^h\right) \mathrm{d}\Gamma \\
&+ \int_{\Gamma_0^b} \delta\overline{\boldsymbol{\kappa}}^h \cdot \mathbf{D}^b\overline{\boldsymbol{\kappa}}^h \,\mathrm{d}\Gamma = \\
&- \int_{\Gamma_t^s} \delta\mathbf{y}^h \cdot \rho h_{th} \left(\left.\frac{\partial^2 \mathbf{y}^h}{\partial t^2}\right|_{\mathbf{X}} - \mathbf{f}^h \right) \mathrm{d}\Gamma + \int_{(\Gamma_t^s)_h} \delta\mathbf{y}^h \cdot \mathbf{h}^h \,\mathrm{d}\Gamma.
\end{aligned} \tag{104}$$

Although the virtual work equations hold true, the problem setup is unusual in that the stress-free reference configuration Γ_0^s is unknown and the final configuration Γ_t^s is given. The formulation given by Eq. (104) is a form of the inverse deformation problem, whose general formulation and treatment was proposed in [138], and further analyzed and studied computationally in [139]. In these references, the focus was placed on developing the right kinematic and stress measures for the inverse deformation problem. Here we develop a simple algorithm for the solution of the inverse deformation equations with application to wind-turbine blades.

We assume that the rotor spins around its axis at a constant angular speed and that the inflow wind conditions do not change. With this setup, the blade is subjected to a constant-in-time centripetal force density (per unit volume) given by

$$\rho \left.\frac{\partial^2 \mathbf{y}^h}{\partial t^2}\right|_{\mathbf{X}} = \rho\boldsymbol{\omega} \times (\boldsymbol{\omega} \times (\mathbf{x} - \mathbf{x}_0)), \tag{105}$$

where the coordinate system of the current configuration is assumed to rotate with the blade, $\boldsymbol{\omega}$ is the vector of angular velocities, and $\mathbf{x}_0$ is a fixed point. The centripetal force density per unit volume may be directly computed as

$$\rho\boldsymbol{\omega} \times \boldsymbol{\omega} \times (\mathbf{x} - \mathbf{x}_0) = \begin{bmatrix} -\rho x\dot{\theta}^2 \\ -\rho y\dot{\theta}^2 \\ 0 \end{bmatrix}, \tag{106}$$

where we assume that the coordinate system of the blade is chosen such that the y-axis is aligned with the blade axis, and the blade rotates around the z-axis with a constant angular speed $\dot{\theta}$.

The time-averaged aerodynamics traction vector $\mathbf{h}^h$ in Eq. (104) may be obtained from a separate aerodynamics

computation of a rigidly-spinning rotor using the methods described in the earlier sections. (See [140] for the computation of time-averaged traction vector for moving boundary simulations.)

In [14], we proposed the following two-stage iterative approach to solve the variational equations (104) for the shell midsurface displacement, which, in turn, gives the stress-free reference configuration.

Initialization: We initialize the unknown reference configuration to coincide with the current configuration, that is,

$$\Gamma_0^s = \Gamma_t^s, \tag{107}$$

which implies

$$\mathbf{y}^h = \mathbf{0}. \tag{108}$$

Step 1: Given the reference configuration Γ_0^s, we solve the standard nonlinear structural problem: find the structural displacement $\mathbf{y}^h \in \mathcal{S}_y^h$ relative to Γ_0^s, such that $\forall\ \delta\mathbf{y}^h \in \mathcal{V}_y^h$:

$$\begin{aligned}
&\int_{\Gamma_0^s} \delta\overline{\boldsymbol{\varepsilon}}^h \cdot \left(\mathbf{A}\overline{\boldsymbol{\varepsilon}}^h + \mathbf{B}\overline{\boldsymbol{\kappa}}^h\right) \mathrm{d}\Gamma \\
&+ \int_{\Gamma_0^s} \delta\overline{\boldsymbol{\kappa}}^h \cdot \left(\mathbf{B}\overline{\boldsymbol{\varepsilon}}^h + \mathbf{D}\overline{\boldsymbol{\kappa}}^h\right) \mathrm{d}\Gamma \\
&+ \int_{\Gamma_0^b} \delta\overline{\boldsymbol{\kappa}}^h \cdot \mathbf{D}^b\overline{\boldsymbol{\kappa}}^h \, \mathrm{d}\Gamma = \\
&- \int_{\Gamma_t^s} \delta\mathbf{y}^h \cdot (\rho h_{th}\boldsymbol{\omega} \times (\boldsymbol{\omega} \times (\mathbf{x} - \mathbf{x}_0))) \, \mathrm{d}\Gamma + \int_{(\Gamma_t^s)_h} \delta\mathbf{y}^h \cdot \mathbf{h}^h \, \mathrm{d}\Gamma.
\end{aligned} \tag{109}$$

Standard Newton–Raphson iteration is employed in this work to compute the solution of the nonlinear structural problem given by Eq. (109).

Step 2: Given the displacement solution $\mathbf{y}^h$ from Step 1, we update the reference configuration as

$$\Gamma_0^s = \{\mathbf{X} \mid \mathbf{X} = \mathbf{x} - \mathbf{y}^h,\ \forall \mathbf{x} \in \Gamma_t^s\}, \tag{110}$$

and return to Step 1 using $\mathbf{y}^h$ as the initial data.

Steps 1–2 are repeated until convergence, that is, until $\mathbf{y}^h$ satisfies Eq. (109).

The above algorithm is based on the idea of computing negative increments of the displacement, or increments of the displacement away from the current configuration, until the reference configuration is found. The mathematical justification for this approach may be found in the appendix of [14]. In what follows, we will illustrate the good performance of the proposed algorithm on a full-scale wind-turbine blade subject to realistic wind and inertial loads.

9.2 Pre-bending Results for the NREL 5MW Wind-Turbine Blade

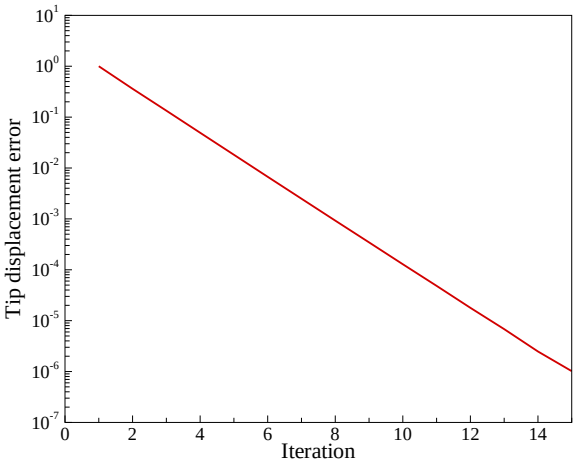


Fig. 49 Blade tip displacement convergence as a function of the iteration number.

The blade design and wind conditions for the pre-bending computations presented here are taken from [15]. Figure 49 shows the tip displacement convergence of the iterative pre-bending algorithm. After a few (5–6) iterations of the two-step pre-bending algorithm the tip exhibits no further visible displacements, and the computation is stopped after a total of 15 iterations. Figure 50 shows the initial and

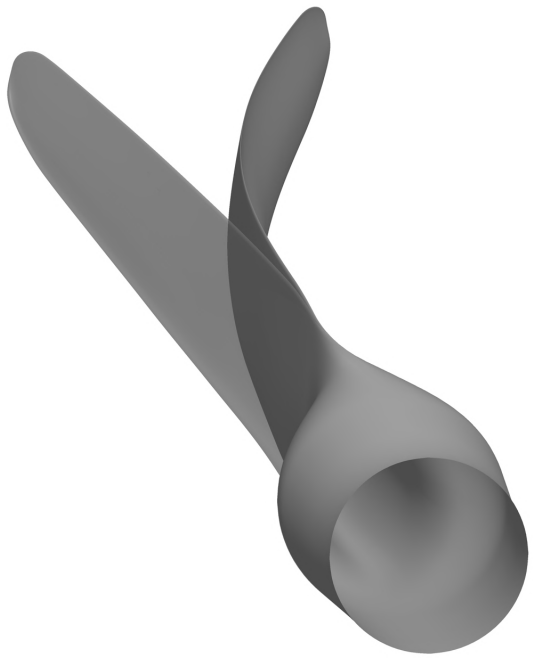


Fig. 50 Rotor blade design and pre-bent configurations superposed.

the final stress-free blade shapes. As expected, the blade bends into the wind. The tip deflection is predicted to be 5.61 m.

We examine the stress distribution in the composite plies of the blade skin. For each ply we compute the Cauchy stress tensor components with respect to the local Cartesian basis that is aligned with the material axes. The first basis vector points in the direction of the fiber and the second in the direction of the matrix, which is orthogonal to the fiber direction (see Eqs. (58) and (59)). The maximum values of the tensile (σ^t), compressive (σ^c), and in-plane shear stresses are computed for each ply. The highest ratio of the predicted Cauchy stress and the composite strength, i.e, $\sigma_2^t/\sigma_2^{t,u}$, occurs for the tensile stress in the direction of the matrix material. Although the ratio does not exceed 0.6, which means the predicted stress is below the composite failure strength, we feel this value is somewhat high. In the rest of the stress components the ratios are significantly lower. Figure 51 shows

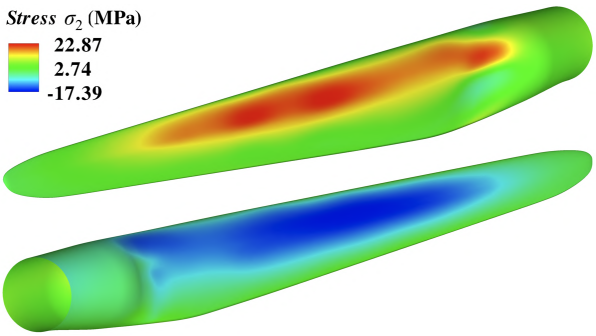


Fig. 51 Normal stress distribution in the direction of the matrix material for the 0° ply number 14. Top: Pressure side. Bottom: Suction side.

the distribution of σ_2 in the 0° ply number 14, which has the highest ratio of $\sigma_2^t/\sigma_2^{t,u}$. The pressure side of the blade is in tension, while the suction side of the blade is in compression as expected. However, the level of the tensile stress is not very far from the tensile failure strength, which suggests that stronger matrix material may be desirable for this blade design.

10 Concluding Remarks

We provided an extensive overview of the aerodynamic and FSI analysis of wind turbines carried out in recent years with the ALE-VMS and ST-VMS methods. The techniques complementing these core methods include weak enforcement of the essential boundary conditions, NURBS-based isogeometric analysis, using NURBS basis functions in temporal representation of the rotor motion, mesh motion and also in remeshing, rotation representation with constant angular velocity, Kirchhoff–Love shell modeling of the rotor-blade structure, and full FSI coupling. These techniques were included in our overview. The wind-turbine analysis cases presented include the aerodynamics of standalone wind-turbine rotors, wind-turbine rotor and tower, and the FSI that accounts for the deformation of the rotor blades. The specific wind turbines considered were NREL 5MW, NREL Phase VI and Micon 65/13M, all at full scale. In the case of NREL Phase VI and Micon 65/13M we also presented a successful comparison with the experimental data. Overall, this article demonstrates that the ALE-VMS and ST-VMS methods, together with a number of new supporting techniques, have brought the aerodynamic and FSI analysis of wind turbines to a new level, where such analyses can contribute more to simulation-based design and testing.

Acknowledgements

We wish to thank the Texas Advanced Computing Center (TACC) and the San Diego Supercomputing Center (SDSC) for providing HPC resources that have contributed to the research results reported in this paper. The first author acknowledges the support of the NSF CAREER Award, the NSF Award CBET-1306869, and the Air Force Office of Scientific Research Award FA9550-12-1-0005. The ST-VMS part of the work was supported by ARO grants W911NF-09-1-0346 and W911NF-12-1-0162 (third author) and Rice–Waseda Research Agreement (second author).

References

1. J.M. Jonkman and M.L. Buhl, “FAST user’s guide”, Technical Report NREL/EL-500-38230, National Renewable Energy Laboratory, Golden, CO, 2005.
2. J. Jonkman, S. Butterfield, W. Musial, and G. Scott, “Definition of a 5-MW reference wind turbine for offshore system development”, Technical Report NREL/TP-500-38060, National Renewable Energy Laboratory, Golden, CO, 2009.
3. N.N. Sørensen, J.A. Michelsen, and S. Schreck, “Navier–Stokes predictions of the NREL Phase VI rotor in the NASA Ames 80 ft × 120 ft wind tunnel”, *Wind Energy*, **5** (2002) 151–169.
4. A.L. Pape and J. Lecanu, “3D Navier–Stokes computations of a stall-regulated wind turbine”, *Wind Energy*, **7** (2004) 309–324.
5. F. Zahle, N.N. Sørensen, and J. Johansen, “Wind turbine rotor-tower interaction using an incompressible overset grid method”, *Wind Energy*, **12** (2009) 594–619.
6. Y. Bazilevs, M.-C. Hsu, I. Akkerman, S. Wright, K. Takizawa, B. Henicke, T. Spielman, and T.E. Tezduyar, “3D simulation of wind turbine rotors at full scale. Part I: Geometry modeling and aerodynamics”, *International Journal for Numerical Methods in Fluids*, **65** (2011) 207–235, doi: 10.1002/fld.2400.
7. K. Takizawa, B. Henicke, T.E. Tezduyar, M.-C. Hsu, and Y. Bazilevs, “Stabilized space–time computation of wind-turbine rotor aerodynamics”, *Computational Mechanics*, **48** (2011) 333–344, doi: 10.1007/s00466-011-0589-2.
8. Y. Li and P.M.C. Kim-Jong Paik, T. Xing, “Dynamic overset CFD simulations of wind turbine aerodynamics”, *Renewable Energy*, **37** (2012) 285–298.
9. E. Guttierez, S. Primi, F. Taucer, P. Caperan, D. Tirelli, J. Mieres, I. Calvo, J. Rodriguez, F. Vallano, G. Galiotis, and D. Mouzakis, “A wind turbine tower design based on fibre-reinforced composites”, Technical report, Joint Research Centre - Ispra, European

Laboratory for Structural Assessment (ELSA), Institute For Protection and Security of the Citizen (IPSC), European Commission, 2003.
10. C. Kong, J. Bang, and Y. Sugiyama, "Structural investigation of composite wind turbine blade considering various load cases and fatigue life", *Energy*, **30** (2005) 2101–2114.
11. M.O.L. Hansen, J.N. Sørensen, S. Voutsinas, N. Sørensen, and H.A. Madsen, "State of the art in wind turbine aerodynamics and aeroelasticity", *Progress in Aerospace Sciences*, **42** (2006) 285–330.
12. F.M. Jensen, B.G. Falzon, J. Ankersen, and H. Stang, "Structural testing and numerical simulation of a 34 m composite wind turbine blade", *Composite Structures*, **76** (2006) 52–61.
13. J. Kiendl, Y. Bazilevs, M.-C. Hsu, R. Wüchner, and K.-U. Bletzinger, "The bending strip method for isogeometric analysis of Kirchhoff–Love shell structures comprised of multiple patches", *Computer Methods in Applied Mechanics and Engineering*, **199** (2010) 2403–2416.
14. Y. Bazilevs, M.-C. Hsu, J. Kiendl, and D.J. Benson, "A computational procedure for pre-bending of wind turbine blades", *International Journal for Numerical Methods in Engineering*, **89** (2012) 323–336.
15. Y. Bazilevs, M.-C. Hsu, J. Kiendl, R. Wüchner, and K.-U. Bletzinger, "3D simulation of wind turbine rotors at full scale. Part II: Fluid–structure interaction modeling with composite blades", *International Journal for Numerical Methods in Fluids*, **65** (2011) 236–253.
16. T.J.R. Hughes, J.A. Cottrell, and Y. Bazilevs, "Isogeometric analysis: CAD, finite elements, NURBS, exact geometry, and mesh refinement", *Computer Methods in Applied Mechanics and Engineering*, **194** (2005) 4135–4195.
17. J.A. Cottrell, A. Reali, Y. Bazilevs, and T.J.R. Hughes, "Isogeometric analysis of structural vibrations", *Computer Methods in Applied Mechanics and Engineering*, **195** (2006) 5257–5297.
18. Y. Bazilevs, L.B. da Veiga, J.A. Cottrell, T.J.R. Hughes, and G. Sangalli, "Isogeometric analysis: Approximation, stability and error estimates for *h*-refined meshes", *Mathematical Models and Methods in Applied Sciences*, **16** (2006) 1031–1090.
19. J.A. Cottrell, T.J.R. Hughes, and A. Reali, "Studies of refinement and continuity in isogeometric structural analysis", *Computer Methods in Applied Mechanics and Engineering*, **196** (2007) 4160–4183.
20. W.A. Wall, M.A. Frenzel, and C. Cyron, "Isogeometric structural shape optimization", *Computer Methods in Applied Mechanics and Engineering*, **197** (2008) 2976–2988.
21. J.A. Cottrell, T.J.R. Hughes, and Y. Bazilevs, *Isogeometric Analysis: Toward Integration of CAD and FEA*. Wiley, Chichester, 2009.
22. J.A. Evans, Y. Bazilevs, I. Babuška, and T.J.R. Hughes, "*n*-Widths, sup-infs, and optimality ratios for the *k*-version of the isogeometric finite element method", *Computer Methods in Applied Mechanics and Engineering*, **198** (2009) 1726–1741.
23. M.R. Dörfel, B. Jüttler, and B. Simeon, "Adaptive isogeometric analysis by local *h*-refinement with T-splines", *Computer Methods in Applied Mechanics and Engineering*, **199** (2010) 264–275.
24. Y. Bazilevs, V.M. Calo, J.A. Cottrell, J.A. Evans, T.J.R. Hughes, S. Lipton, M.A. Scott, and T.W. Sederberg, "Isogeometric analysis using T-splines", *Computer Methods in Applied Mechanics and Engineering*, **199** (2010) 229–263.
25. F. Auricchio, L. Beirão da Veiga, C. Lovadina, and A. Reali, "The importance of the exact satisfaction of the incompressibility constraint in nonlinear elasticity: Mixed FEMs versus NURBS-based approximations", *Computer Methods in Applied Mechanics and Engineering*, **199** (2010) 314–323.
26. W. Wang and Y. Zhang, "Wavelets-based NURBS simplification and fairing", *Computer Methods in Applied Mechanics and Engineering*, **199** (2010) 290–300.
27. E. Cohen, T. Martin, R.M. Kirby, T. Lyche, and R.F. Riesenfeld, "Analysis-aware modeling: Understanding quality considerations in modeling for isogeometric analysis", *Computer Methods in Applied Mechanics and Engineering*, **199** (2010) 334–356.
28. V. Srinivasan, S. Radhakrishnan, and G. Subbarayan, "Coordinated synthesis of hierarchical engineering systems", *Computer Methods in Applied Mechanics and Engineering*, **199** (2010) 392–404.
29. Y. Bazilevs, V.M. Calo, J.A. Cottrell, T.J.R. Hughes, A. Reali, and G. Scovazzi, "Variational multiscale residual-based turbulence modeling for large eddy simulation of incompressible flows", *Computer Methods in Applied Mechanics and Engineering*, **197** (2007) 173–201.
30. Y. Bazilevs, C. Michler, V.M. Calo, and T.J.R. Hughes, "Weak Dirichlet boundary conditions for wall-bounded turbulent flows", *Computer Methods in Applied Mechanics and Engineering*, **196** (2007) 4853–4862.
31. Y. Bazilevs, C. Michler, V.M. Calo, and T.J.R. Hughes, "Isogeometric variational multiscale modeling of wall-bounded turbulent flows with weakly enforced boundary conditions on unstretched meshes", *Computer Methods in Applied Mechanics and Engineering*, **199** (2010) 780–790.
32. I. Akkerman, Y. Bazilevs, V.M. Calo, T.J.R. Hughes, and S. Hulshoff, "The role of continuity in residual-based variational multiscale modeling of turbulence", *Computational Mechanics*, **41** (2008) 371–378.
33. M.-C. Hsu, Y. Bazilevs, V.M. Calo, T.E. Tezduyar, and T.J.R. Hughes, "Improving stability of stabilized and multiscale formulations in flow simulations at small time steps", *Computer Methods in Applied Mechanics and Engineering*, **199** (2010) 828–840, doi: 10.1016/j.cma.2009.06.019.
34. Y. Bazilevs and I. Akkerman, "Large eddy simulation of turbulent Taylor–Couette flow using isogeometric analysis and the residual–based variational multiscale method", *Journal of Computational Physics*, **229** (2010) 3402–3414.
35. T. Elguedj, Y. Bazilevs, V.M. Calo, and T.J.R. Hughes, "B-bar and F-bar projection methods for nearly incompressible linear and nonlinear elasticity and plasticity using higher-order nurbs elements", *Computer Methods in Applied Mechanics and Engineering*, **197** (2008) 2732–2762.
36. S. Lipton, J.A. Evans, Y. Bazilevs, T. Elguedj, and T.J.R. Hughes, "Robustness of isogeometric structural discretizations under severe mesh distortion", *Computer Methods in Applied Mechanics and Engineering*, **199** (2010) 357–373.
37. D.J. Benson, Y. Bazilevs, E. De Luycker, M.-C. Hsu, M. Scott, T.J.R. Hughes, and T. Belytschko, "A generalized finite element formulation for arbitrary basis functions: from isogeometric analysis to XFEM", *International Journal for Numerical Methods in Engineering*, **83** (2010) 765–785.
38. D.J. Benson, Y. Bazilevs, M.-C. Hsu, and T.J.R. Hughes, "Isogeometric shell analysis: The Reissner–Mindlin shell", *Computer Methods in Applied Mechanics and Engineering*, **199** (2010) 276–289.
39. J. Kiendl, K.-U. Bletzinger, J. Linhard, and R. Wüchner, "Isogeometric shell analysis with Kirchhoff–Love elements", *Computer Methods in Applied Mechanics and Engineering*, **198** (2009) 3902–3914.
40. Y. Zhang, Y. Bazilevs, S. Goswami, C. Bajaj, and T.J.R. Hughes, "Patient-specific vascular nurbs modeling for isogeometric analysis of blood flow", *Computer Methods in Applied Mechanics and Engineering*, **196** (2007) 2943–2959.
41. Y. Bazilevs, V.M. Calo, Y. Zhang, and T.J.R. Hughes, "Isogeometric fluid–structure interaction analysis with applications to arterial blood flow", *Computational Mechanics*, **38** (2006) 310–322.
42. Y. Bazilevs, V.M. Calo, T.J.R. Hughes, and Y. Zhang, "Isogeometric fluid–structure interaction: theory, algorithms, and computations", *Computational Mechanics*, **43** (2008) 3–37.
43. J.G. Isaksen, Y. Bazilevs, T. Kvamsdal, Y. Zhang, J.H. Kaspersen, K. Waterloo, B. Romner, and T. Ingebrigtsen, "Determination of wall tension in cerebral artery aneurysms by numerical simulation", *Stroke*, **39** (2008) 3172–3178.

44. Y. Bazilevs and T.J.R. Hughes, "NURBS-based isogeometric analysis for the computation of flows about rotating components", *Computational Mechanics*, **43** (2008) 143–150.
45. F. Cirak, M. Ortiz, and P. Schröder, "Subdivision surfaces: a new paradigm for thin shell analysis", *International Journal for Numerical Methods in Engineering*, **47** (2000) 2039–2072.
46. F. Cirak and M. Ortiz, "Fully C^1-conforming subdivision elements for finite deformation thin shell analysis", *International Journal for Numerical Methods in Engineering*, **51** (2001) 813–833.
47. F. Cirak, M.J. Scott, E.K. Antonsson, M. Ortiz, and P. Schröder, "Integrated modeling, finite-element analysis, and engineering design for thin-shell structures using subdivision", *Computer-Aided Design*, **34** (2002) 137–148.
48. T.J.R. Hughes, W.K. Liu, and T.K. Zimmermann, "Lagrangian–Eulerian finite element formulation for incompressible viscous flows", *Computer Methods in Applied Mechanics and Engineering*, **29** (1981) 329–349.
49. T.J.R. Hughes, "Multiscale phenomena: Green's functions, the Dirichlet-to-Neumann formulation, subgrid scale models, bubbles, and the origins of stabilized methods", *Computer Methods in Applied Mechanics and Engineering*, **127** (1995) 387–401.
50. T.J.R. Hughes, A.A. Oberai, and L. Mazzei, "Large eddy simulation of turbulent channel flows by the variational multiscale method", *Physics of Fluids*, **13** (2001) 1784–1799.
51. K. Takizawa and T.E. Tezduyar, "Multiscale space–time fluid–structure interaction techniques", *Computational Mechanics*, **48** (2011) 247–267, doi: 10.1007/s00466-011-0571-z.
52. K. Takizawa and T.E. Tezduyar, "Space–time fluid–structure interaction methods", *Mathematical Models and Methods in Applied Sciences*, **22** (2012) 1230001, doi: 10.1142/S0218202512300013.
53. T.E. Tezduyar, "Stabilized finite element formulations for incompressible flow computations", *Advances in Applied Mechanics*, **28** (1992) 1–44, doi: 10.1016/S0065-2156(08)70153-4.
54. T.E. Tezduyar, M. Behr, and J. Liou, "A new strategy for finite element computations involving moving boundaries and interfaces – the deforming-spatial-domain/space–time procedure: I. The concept and the preliminary numerical tests", *Computer Methods in Applied Mechanics and Engineering*, **94** (1992) 339–351, doi: 10.1016/0045-7825(92)90059-S.
55. T.E. Tezduyar, M. Behr, S. Mittal, and J. Liou, "A new strategy for finite element computations involving moving boundaries and interfaces – the deforming-spatial-domain/space–time procedure: II. Computation of free-surface flows, two-liquid flows, and flows with drifting cylinders", *Computer Methods in Applied Mechanics and Engineering*, **94** (1992) 353–371, doi: 10.1016/0045-7825(92)90060-W.
56. T.E. Tezduyar, "Computation of moving boundaries and interfaces and stabilization parameters", *International Journal for Numerical Methods in Fluids*, **43** (2003) 555–575, doi: 10.1002/fld.505.
57. T.E. Tezduyar and S. Sathe, "Modeling of fluid–structure interactions with the space–time finite elements: Solution techniques", *International Journal for Numerical Methods in Fluids*, **54** (2007) 855–900, doi: 10.1002/fld.1430.
58. Y. Bazilevs, K. Takizawa, and T.E. Tezduyar, *Computational Fluid–Structure Interaction: Methods and Applications*. Wiley, February 2013, ISBN 978-0470978771.
59. Y. Bazilevs and T.J.R. Hughes, "Weak imposition of Dirichlet boundary conditions in fluid mechanics", *Computers and Fluids*, **36** (2007) 12–26.
60. J. Nitsche, "Uber ein variationsprinzip zur losung von Dirichlet-problemen bei verwendung von teilraumen, die keinen randbedingungen unterworfen sind", *Abh. Math. Univ. Hamburg*, **36** (1971) 9–15.
61. D.N. Arnold, F. Brezzi, B. Cockburn, and L.D. Marini, "Unified analysis of Discontinuous Galerkin methods for elliptic problems", *SIAM Journal of Numerical Analysis*, **39** (2002) 1749–1779.
62. A.N. Brooks and T.J.R. Hughes, "Streamline upwind/Petrov-Galerkin formulations for convection dominated flows with particular emphasis on the incompressible Navier-Stokes equations", *Computer Methods in Applied Mechanics and Engineering*, **32** (1982) 199–259.
63. T.E. Tezduyar, S. Mittal, S.E. Ray, and R. Shih, "Incompressible flow computations with stabilized bilinear and linear equal-order-interpolation velocity-pressure elements", *Computer Methods in Applied Mechanics and Engineering*, **95** (1992) 221–242, doi: 10.1016/0045-7825(92)90141-6.
64. S. Mittal and T.E. Tezduyar, "A finite element study of incompressible flows past oscillating cylinders and aerofoils", *International Journal for Numerical Methods in Fluids*, **15** (1992) 1073–1118, doi: 10.1002/fld.1650150911.
65. S. Mittal and T.E. Tezduyar, "Parallel finite element simulation of 3D incompressible flows – Fluid-structure interactions", *International Journal for Numerical Methods in Fluids*, **21** (1995) 933–953, doi: 10.1002/fld.1650211011.
66. V. Kalro and T.E. Tezduyar, "A parallel 3D computational method for fluid–structure interactions in parachute systems", *Computer Methods in Applied Mechanics and Engineering*, **190** (2000) 321–332, doi: 10.1016/S0045-7825(00)00204-8.
67. T.E. Tezduyar, S. Sathe, R. Keedy, and K. Stein, "Space–time finite element techniques for computation of fluid–structure interactions", *Computer Methods in Applied Mechanics and Engineering*, **195** (2006) 2002–2027, doi: 10.1016/j.cma.2004.09.014.
68. K. Takizawa and T.E. Tezduyar, "Computational methods for parachute fluid–structure interactions", *Archives of Computational Methods in Engineering*, **19** (2012) 125–169, doi: 10.1007/s11831-012-9070-4.
69. T.E. Tezduyar, K. Takizawa, T. Brummer, and P.R. Chen, "Space–time fluid–structure interaction modeling of patient-specific cerebral aneurysms", *International Journal for Numerical Methods in Biomedical Engineering*, **27** (2011) 1665–1710, doi: 10.1002/cnm.1433.
70. K. Takizawa, Y. Bazilevs, and T.E. Tezduyar, "Space–time and ALE-VMS techniques for patient-specific cardiovascular fluid–structure interaction modeling", *Archives of Computational Methods in Engineering*, **19** (2012) 171–225, doi: 10.1007/s11831-012-9071-3.
71. K. Takizawa, K. Schjodt, A. Puntel, N. Kostov, and T.E. Tezduyar, "Patient-specific computer modeling of blood flow in cerebral arteries with aneurysm and stent", *Computational Mechanics*, **50** (2012) 675–686, doi: 10.1007/s00466-012-0760-4.
72. K. Takizawa, M. Fritze, D. Montes, T. Spielman, and T.E. Tezduyar, "Fluid–structure interaction modeling of ringsail parachutes with disreefing and modified geometric porosity", *Computational Mechanics*, **50** (2012) 835–854, doi: 10.1007/s00466-012-0761-3.
73. K. Takizawa, D. Montes, M. Fritze, S. McIntyre, J. Boben, and T.E. Tezduyar, "Methods for FSI modeling of spacecraft parachute dynamics and cover separation", *Mathematical Models and Methods in Applied Sciences*, **23** (2013) 307–338, doi: 10.1142/S0218202513400058.
74. K. Takizawa, T.E. Tezduyar, J. Boben, N. Kostov, C. Boswell, and A. Buscher, "Fluid–structure interaction modeling of clusters of spacecraft parachutes with modified geometric porosity", *Computational Mechanics*, **52** (2013) 1351–1364, doi: 10.1007/s00466-013-0880-5.
75. K. Takizawa, K. Schjodt, A. Puntel, N. Kostov, and T.E. Tezduyar, "Patient-specific computational analysis of the influence of a stent on the unsteady flow in cerebral aneurysms", *Computational Mechanics*, **51** (2013) 1061–1073, doi: 10.1007/s00466-012-0790-y.
76. M. Manguoglu, K. Takizawa, A.H. Sameh, and T.E. Tezduyar, "Nested and parallel sparse algorithms for arterial fluid mechanics computations with boundary layer mesh refinement", *International Journal for Numerical Methods in Fluids*, **65** (2011) 135–

149, doi: 10.1002/fld.2415.
77. M. Manguoglu, K. Takizawa, A.H. Sameh, and T.E. Tezduyar, "A parallel sparse algorithm targeting arterial fluid mechanics computations", *Computational Mechanics*, **48** (2011) 377–384, doi: 10.1007/s00466-011-0619-0.
78. T. Tezduyar, S. Aliabadi, M. Behr, A. Johnson, V. Kalro, and M. Litke, "Flow simulation and high performance computing", *Computational Mechanics*, **18** (1996) 397–412, doi: 10.1007/BF00350249.
79. M. Behr and T. Tezduyar, "The Shear-Slip Mesh Update Method", *Computer Methods in Applied Mechanics and Engineering*, **174** (1999) 261–274, doi: 10.1016/S0045-7825(98)00299-0.
80. M. Behr and T. Tezduyar, "Shear-slip mesh update in 3D computation of complex flow problems with rotating mechanical components", *Computer Methods in Applied Mechanics and Engineering*, **190** (2001) 3189–3200, doi: 10.1016/S0045-7825(00)00388-1.
81. K. Takizawa, B. Henicke, D. Montes, T.E. Tezduyar, M.-C. Hsu, and Y. Bazilevs, "Numerical-performance studies for the stabilized space–time computation of wind-turbine rotor aerodynamics", *Computational Mechanics*, **48** (2011) 647–657, doi: 10.1007/s00466-011-0614-5.
82. Y. Bazilevs, M.-C. Hsu, K. Takizawa, and T.E. Tezduyar, "ALE-VMS and ST-VMS methods for computer modeling of wind-turbine rotor aerodynamics and fluid–structure interaction", *Mathematical Models and Methods in Applied Sciences*, **22** (2012) 1230002, doi: 10.1142/S0218202512300025.
83. K. Takizawa, T.E. Tezduyar, S. McIntyre, N. Kostov, R. Kolesar, and C. Habluetzel, "Space–time VMS computation of wind-turbine rotor and tower aerodynamics", *Computational Mechanics*, **53** (2014) 1–15, doi: 10.1007/s00466-013-0888-x.
84. K. Takizawa, B. Henicke, A. Puntel, T. Spielman, and T.E. Tezduyar, "Space–time computational techniques for the aerodynamics of flapping wings", *Journal of Applied Mechanics*, **79** (2012) 010903, doi: 10.1115/1.4005073.
85. K. Takizawa, B. Henicke, A. Puntel, N. Kostov, and T.E. Tezduyar, "Space–time techniques for computational aerodynamics modeling of flapping wings of an actual locust", *Computational Mechanics*, **50** (2012) 743–760, doi: 10.1007/s00466-012-0759-x.
86. K. Takizawa, N. Kostov, A. Puntel, B. Henicke, and T.E. Tezduyar, "Space–time computational analysis of bio-inspired flapping-wing aerodynamics of a micro aerial vehicle", *Computational Mechanics*, **50** (2012) 761–778, doi: 10.1007/s00466-012-0758-y.
87. K. Takizawa, B. Henicke, A. Puntel, N. Kostov, and T.E. Tezduyar, "Computer modeling techniques for flapping-wing aerodynamics of a locust", *Computers & Fluids*, **85** (2013) 125–134, doi: 10.1016/j.compfluid.2012.11.008.
88. T.E. Tezduyar, M. Behr, S. Mittal, and A.A. Johnson, "Computation of unsteady incompressible flows with the finite element methods – space–time formulations, iterative strategies and massively parallel implementations", in *New Methods in Transient Analysis*, PVP-Vol.246/AMD-Vol.143, ASME, New York, (1992) 7–24.
89. T. Tezduyar, S. Aliabadi, M. Behr, A. Johnson, and S. Mittal, "Parallel finite-element computation of 3D flows", *Computer*, **26** (1993) 27–36, doi: 10.1109/2.237441.
90. A.A. Johnson and T.E. Tezduyar, "Mesh update strategies in parallel finite element computations of flow problems with moving boundaries and interfaces", *Computer Methods in Applied Mechanics and Engineering*, **119** (1994) 73–94, doi: 10.1016/0045-7825(94)00077-8.
91. T.E. Tezduyar, "Finite element methods for flow problems with moving boundaries and interfaces", *Archives of Computational Methods in Engineering*, **8** (2001) 83–130, doi: 10.1007/BF02897870.
92. M.-C. Hsu and Y. Bazilevs, "Fluid–structure interaction modeling of wind turbines: simulating the full machine", *Computational Mechanics*, **50** (2012) 821–833.
93. M.-C. Hsu, I. Akkerman, and Y. Bazilevs, "Finite element simulation of wind turbine aerodynamics: Validation study using NREL Phase VI experiment", *Wind Energy*, **17** (2014) 461–481.
94. M.M. Hand, D.A. Simms, L.J. Fingersh, D.W. Jager, J.R. Cotrell, S. Schreck, and S.M. Larwood, "Unsteady aerodynamics experiment phase VI: Wind tunnel test configurations and available data campaigns", Technical Report NREL/TP-500-29955, National Renewable Energy Laboratory, Golden, CO, 2001.
95. A. Korobenko, M.-C. Hsu, I. Akkerman, J. Tippmann, and Y. Bazilevs, "Structural mechanics modeling and FSI simulation of wind turbines", *Mathematical Models and Methods in Applied Sciences*, **23** (2013) 249–272.
96. C. Johnson, *Numerical solution of partial differential equations by the finite element method*. Cambridge University Press, Sweden, 1987.
97. S.C. Brenner and L.R. Scott, *The Mathematical Theory of Finite Element Methods, 2nd ed.* Springer, 2002.
98. A. Ern and J.-L. Guermond, *Theory and Practice of Finite Elements*. Springer, 2004.
99. T.J.R. Hughes and T.E. Tezduyar, "Finite element methods for first-order hyperbolic systems with particular emphasis on the compressible Euler equations", *Computer Methods in Applied Mechanics and Engineering*, **45** (1984) 217–284, doi: 10.1016/0045-7825(84)90157-9.
100. T.E. Tezduyar and Y.J. Park, "Discontinuity capturing finite element formulations for nonlinear convection-diffusion-reaction equations", *Computer Methods in Applied Mechanics and Engineering*, **59** (1986) 307–325, doi: 10.1016/0045-7825(86)90003-4.
101. T.J.R. Hughes, L.P. Franca, and M. Balestra, "A new finite element formulation for computational fluid dynamics: V. Circumventing the Babuška–Brezzi condition: A stable Petrov–Galerkin formulation of the Stokes problem accommodating equal-order interpolations", *Computer Methods in Applied Mechanics and Engineering*, **59** (1986) 85–99.
102. T.E. Tezduyar and Y. Osawa, "Finite element stabilization parameters computed from element matrices and vectors", *Computer Methods in Applied Mechanics and Engineering*, **190** (2000) 411–430, doi: 10.1016/S0045-7825(00)00211-5.
103. T.J.R. Hughes, G.R. Feijóo, L. Mazzei, and J.-B. Quincy, "The variational multiscale method–A paradigm for computational mechanics", *Computer Methods in Applied Mechanics and Engineering*, **166** (1998) 3–24.
104. T.J.R. Hughes and G. Sangalli, "Variational multiscale analysis: the fine-scale Green's function, projection, optimization, localization, and stabilized methods", *SIAM Journal of Numerical Analysis*, **45** (2007) 539–557.
105. J.E. Akin, T. Tezduyar, M. Ungor, and S. Mittal, "Stabilization parameters and Smagorinsky turbulence model", *Journal of Applied Mechanics*, **70** (2003) 2–9, doi: 10.1115/1.1526569.
106. J.E. Akin and T.E. Tezduyar, "Calculation of the advective limit of the SUPG stabilization parameter for linear and higher-order elements", *Computer Methods in Applied Mechanics and Engineering*, **193** (2004) 1909–1922, doi: 10.1016/j.cma.2003.12.050.
107. E. Onate, A. Valls, and J. Garcia, "FIC/FEM formulation with matrix stabilizing terms for incompressible flows at low and high Reynolds numbers", *Computational Mechanics*, **38** (2006) 440–455.
108. A. Corsini, F. Rispoli, A. Santoriello, and T.E. Tezduyar, "Improved discontinuity-capturing finite element techniques for reaction effects in turbulence computation", *Computational Mechanics*, **38** (2006) 356–364, doi: 10.1007/s00466-006-0045-x.
109. F. Rispoli, A. Corsini, and T.E. Tezduyar, "Finite element computation of turbulent flows with the discontinuity-capturing directional dissipation (DCDD)", *Computers & Fluids*, **36** (2007) 121–126, doi: 10.1016/j.compfluid.2005.

07.004.
110. A. Corsini, C. Iossa, F. Rispoli, and T.E. Tezduyar, "A DRD finite element formulation for computing turbulent reacting flows in gas turbine combustors", *Computational Mechanics*, **46** (2010) 159–167, doi: 10.1007/s00466-009-0441-0.
111. A. Corsini, F. Rispoli, and T.E. Tezduyar, "Stabilized finite element computation of NOx emission in aero-engine combustors", *International Journal for Numerical Methods in Fluids*, **65** (2011) 254–270, doi: 10.1002/fld.2451.
112. A. Corsini, F. Rispoli, and T.E. Tezduyar, "Computer modeling of wave-energy air turbines with the SUPG/PSPG formulation and discontinuity-capturing technique", *Journal of Applied Mechanics*, **79** (2012) 010910, doi: 10.1115/1.4005060.
113. A. Corsini, F. Rispoli, A.G. Sheard, and T.E. Tezduyar, "Computational analysis of noise reduction devices in axial fans with stabilized finite element formulations", *Computational Mechanics*, **50** (2012) 695–705, doi: 10.1007/s00466-012-0789-4.
114. B.E. Launder and D.B. Spalding, "The numerical computation of turbulent flows", *Computer Methods in Applied Mechanics and Engineering*, **3** (1974) 269–289.
115. D.C. Wilcox, *Turbulence Modeling for CFD*. DCW Industries, La Canada, CA, 1998.
116. H.J.T. Kooijman, C. Lindenburg, D. Winkelaar, and E.L. van der Hooft, "DOWEC 6 MW pre-design: Aero-elastic modelling of the DOWEC 6 MW pre-design in PHATAS", Technical Report DOWEC-F1W2-HJK-01-046/9, 2003.
117. K. Takizawa, C. Moorman, S. Wright, T. Spielman, and T.E. Tezduyar, "Fluid–structure interaction modeling and performance analysis of the Orion spacecraft parachutes", *International Journal for Numerical Methods in Fluids*, **65** (2011) 271–285, doi: 10.1002/fld.2348.
118. K. Takizawa, C. Moorman, S. Wright, and T.E. Tezduyar, "Computer modeling and analysis of the Orion spacecraft parachutes", in H.-J. Bungartz, M. Mehl, and M. Schafer, editors, *Fluid–Structure Interaction II – Modelling, Simulation, Optimization*, volume 73 of *Lecture Notes in Computational Science and Engineering*, Chapter 3, 53–81, Springer, 2010, ISBN 978-3-642-14206-2.
119. K. Takizawa, S. Wright, C. Moorman, and T.E. Tezduyar, "Fluid–structure interaction modeling of parachute clusters", *International Journal for Numerical Methods in Fluids*, **65** (2011) 286–307, doi: 10.1002/fld.2359.
120. D.A. Spera, "Introduction to modern wind turbines", in D.A. Spera, editor, *Wind Turbine Technology: Fundamental Concepts of Wind Turbine Engineering*, 47–72, ASME Press, 1994.
121. Y. Saad and M. Schultz, "GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems", *SIAM Journal of Scientific and Statistical Computing*, **7** (1986) 856–869.
122. G. Karypis and V. Kumar, "A fast and high quality multilevel scheme for partitioning irregular graphs", *SIAM Journal of Scientific Computing*, **20** (1998) 359–392.
123. Y. Bazilevs, M.-C. Hsu, and M.A. Scott, "Isogeometric fluid–structure interaction analysis with emphasis on non-matching discretizations, and with application to wind turbines", *Computer Methods in Applied Mechanics and Engineering*, **249-252** (2012) 28–41.
124. T.E. Tezduyar, S. Sathe, and K. Stein, "Solution techniques for the fully-discretized equations in computation of fluid–structure interactions with the space–time formulations", *Computer Methods in Applied Mechanics and Engineering*, **195** (2006) 5743–5753, doi: 10.1016/j.cma.2005.08.023.
125. T. Belytschko, W.K. Liu, and B. Moran, *Nonlinear Finite Elements for Continua and Structures*. Wiley, 2000.
126. M. Bischoff, W.A. Wall, K.-U. Bletzinger, and E. Ramm, "Models and finite elements for thin-walled structures", in E. Stein, R. de Borst, and T.J.R. Hughes, editors, *Encyclopedia of Computational Mechanics, Vol. 2, Solids, Structures and Coupled Problems*, Chapter 3, Wiley, 2004.
127. J.N. Reddy, *Mechanics of Laminated Composite Plates and Shells: Theory and Analysis, 2nd ed.* CRC Press, Boca Raton, FL, 2004.
128. K.-U. Bletzinger, S. Kimmich, and E. Ramm, "Efficient modeling in shape optimal design", *Computing Systems in Engineering*, **2** (1991) 483–495.
129. D.J. Benson, Y. Bazilevs, M.-C. Hsu, and T.J.R. Hughes, "A large deformation, rotation-free, isogeometric shell", *Computer Methods in Applied Mechanics and Engineering*, **200** (2011) 1367–1378.
130. T.J.R. Hughes, *The finite element method: Linear static and dynamic finite element analysis*. Prentice Hall, Englewood Cliffs, NJ, 1987.
131. H. Melbø and T. Kvamsdal, "Goal oriented error estimators for Stokes equations based on variationally consistent postprocessing", *Computer Methods in Applied Mechanics and Engineering*, **192** (2003) 613–633.
132. E.H. van Brummelen, V.V. Garg, S. Prudhomme, and K.G. van der Zee, "Flux evaluation in primal and dual boundary-coupled problems", *Journal of Applied Mechanics*, **79** (2011) 010904.
133. J.R. Zayas and W.D. Johnson, "3X-100 blade field test", Report of the Sandia National Laboratories, Wind Energy Technology Department, 2008.
134. J.H. Sutherland, P.L. Jones, and B.A. Neal, "The long-term inflow and structural test program", Proceedings of the 2001 ASME Wind Energy Symposium, p.162, 2001.
135. D. Berry and T. Ashwill, "Design of 9-meter carbon-fiberglass prototype blades: CX-100 and TX-100", Report of the Sandia National Laboratories, 2007.
136. J.R. White, D.E. Adams, and M.A. Rumsey, "Modal analysis of CX-100 rotor blade and Micon 65/13 wind turbine", Structural Dynamics and Renewable Energy, Volume 1, Conference Proceedings of the Society for Experimental Mechanics Series 10, 2011.
137. T. Marinone, B. LeBlanc, J. Harvie, C. Niezrecki, and P. Avitabile, "Modal testing of a 9 m CX-100 turbine blade", Topics in Experimental Dynamics Substructuring and Wind Turbine Dynamics, Volume 2, Conference Proceedings of the Society for Experimental Mechanics Series 27, 2012.
138. R.T. Shield, "Inverse deformation results in finite elasticity", *ZAMP*, **18** (1967) 381–389.
139. S. Govindjee and P.A. Mihalic, "Computational methods for inverse finite elastostatics", *Computer Methods in Applied Mechanics and Engineering*, **136** (1996) 47–57.
140. K. Takizawa, C. Moorman, S. Wright, J. Christopher, and T.E. Tezduyar, "Wall shear stress calculations in space–time finite element computation of arterial fluid–structure interactions", *Computational Mechanics*, **46** (2010) 31–41, doi: 10.1007/s00466-009-0425-0.