FAST.Farm User's Guide and Theory Manual

Jason Jonkman and Kelsey Shaler

National Renewable Energy Laboratory

NREL is a national laboratory of the U.S. Department of Energy
Office of Energy Efficiency & Renewable Energy
Operated by the Alliance for Sustainable Energy, LLC

This report is available at no cost from the National Renewable Energy Laboratory (NREL) at www.nrel.gov/publications.

Contract No. DE-AC36-08GO28308

Technical Report
NREL/TP-5000-78485
April 2021

FAST.Farm User's Guide and Theory Manual

Jason Jonkman and Kelsey Shaler

National Renewable Energy Laboratory

Suggested Citation
Jonkman, Jason, and Kelsey Shaler. 2021. *FAST.Farm User's Guide and Theory Manual*. Golden, CO: National Renewable Energy Laboratory. NREL/TP-5000-78785. https://www.nrel.gov/docs/fy21osti/78485.pdf.

NREL is a national laboratory of the U.S. Department of Energy
Office of Energy Efficiency & Renewable Energy
Operated by the Alliance for Sustainable Energy, LLC

This report is available at no cost from the National Renewable Energy Laboratory (NREL) at www.nrel.gov/publications.

Contract No. DE-AC36-08GO28308

Technical Report
NREL/TP-5000-78485
April 2021

National Renewable Energy Laboratory
15013 Denver West Parkway
Golden, CO 80401
303-275-3000 • www.nrel.gov

NOTICE

This work was authored by the National Renewable Energy Laboratory, operated by Alliance for Sustainable Energy, LLC, for the U.S. Department of Energy (DOE) under Contract No. DE-AC36-08GO28308. Funding provided by the U.S. Department of Energy Office of Energy Efficiency and Renewable Energy Wind Energy Technologies Office. The views expressed herein do not necessarily represent the views of the DOE or the U.S. Government.

This report is available at no cost from the National Renewable Energy Laboratory (NREL) at www.nrel.gov/publications.

U.S. Department of Energy (DOE) reports produced after 1991 and a growing number of pre-1991 documents are available free via www.OSTI.gov.

Cover Photos by Dennis Schroeder: (clockwise, left to right) NREL 51934, NREL 45897, NREL 42160, NREL 45891, NREL 48097, NREL 46526.

NREL prints on paper that contains recycled content.

Acknowledgments

This work was authored by Alliance for Sustainable Energy, LLC, the manager and operator of the National Renewable Energy Laboratory for the U.S. Department of Energy (DOE) under Contract No. DE-AC36-08GO28308. Funding provided by Department of Energy Office of Energy Efficiency and Renewable Energy, Wind Energy Technologies Office. The views expressed in the article do not necessarily represent the views of the DOE or the U.S. Government. The U.S. Government retains and the publisher, by accepting the article for publication, acknowledges that the U.S. Government retains a nonexclusive, paid-up, irrevocable, worldwide license to publish or reproduce the published form of this work, or allow others to do so, for U.S. Government purposes.

Nomenclature

ABLSolver	atmospheric boundary layer solver
AWAE	ambient wind and array effects (module)
$a(r)$	axial induction factor, distributed radially
a_K	coherence decrement parameter
BEM	blade-element momentum
b_K	coherence offset parameter
$C^{\mathrm{O}}_{\mathrm{HWkDfl}}$, $C^{\mathrm{OY}}_{\mathrm{HWkDfl}}$, $C^{\mathrm{x}}_{\mathrm{HWkDfl}}$, and $C^{\mathrm{xY}}_{\mathrm{HWkDfl}}$	calibrated parameters in the horizontal wake-deflection correction
c_{max}	maximum blade chord length
C_{Meander}	calibrated parameter for wake meandering
C_{NearWake}	calibrated parameter in the near-wake correction
C_{WakeDiam}	calibrated parameter in the wake-diameter calculation
$C^{\mathrm{DMax}}_{\nu\mathrm{Amb}}$, $C^{\mathrm{DMin}}_{\nu\mathrm{Amb}}$, $C^{\mathrm{Exp}}_{\nu Amb}$, and $C^{\mathrm{FMin}}_{\nu Amb}$	calibrated parameters in the eddy-viscosity filter function for ambient turbulence
$C^{\mathrm{DMax}}_{\nu\mathrm{Shr}}$, $C^{\mathrm{DMin}}_{\nu\mathrm{Shr}}$, $C^{\mathrm{Exp}}_{\nu\mathrm{Shr}}$, and $C^{\mathrm{FMin}}_{\nu\mathrm{Shr}}$	calibrated parameters in the eddy-viscosity filter function for the wake shear layer
${}^{\mathrm{AzimAvg}}C_t(r)$ and ${}^{\mathrm{FiltAzimAvg}}C_t(r)$	azimuthally averaged thrust-force coefficient (normal to a rotor disk), distributed radially, and its low-pass time-filtered value
$Coh_{i,j}$	magnitude of partial coherence between points i and j
DLL	dynamic-link library
DWM	dynamic wake meandering
D_{Grid}	Assumed rotor diameter when generating TurbSim inflow
D^{Rotor} and ${}^{\mathrm{Filt}}D^{\mathrm{Rotor}}_{n_p}$	rotor diameter and its low-pass time-filtered value at wake plane n_p
$D^{\mathrm{Wake}}_{n_p}$	wake diameter at wake plane n_p
FLORIS	FLOw Redirection and Induction in Steady state
f	frequency
f_c	cutoff (corner) frequency of the low-pass time filter
$\vec{f}_{n_b}(r)$	aerodynamic applied loads distributed radially per unit length for blade n_b
f_{max}	maximum excitation frequency
$F_{\nu\mathrm{Amb}}(x)$	eddy-viscosity filter function associated with ambient turbulence
$F_{\nu\mathrm{Shr}}(x)$	eddy-viscosity filter function associated with the wake shear layer
HFM	high-fidelity modeling
HPC	high-performance computer
I	three-by-three identify matrix
K	velocity components u, v, and w
$k_{\nu\mathrm{Amb}}$	calibrated parameter for the influence of ambient turbulence in the eddy viscosity
$k_{\nu\mathrm{Shr}}$	calibrated parameter for the influence of the wake shear layer in the eddy viscosity
LES	large-eddy simulation
MFoR	moving frame of reference
MPI	message-passing interface
NaN	not a number
NREL	National Renewable Energy Laboratory
N and n	number of discrete-time steps and discrete-time-step counter
N_b and n_b	number of rotor blades and blade counter
$N^{\mathrm{Polar}}_{n_p}$ and n^{Polar}	number of points in the polar grid of wake plane n_p and point counter
N^{Wake} and n^{Wake}	number of wakes overlapping a given wind data point in the wind

	domain and wake counter
N_P and n_P	number of wake planes and wake-plane counter
N_r and n_r	number of radial nodes and radii counter
N_t and n_t	number of wind turbines and turbine counter
OF	OpenFAST (module)
OpenMP	open multiprocessing
$\vec{p}^{\text{Hub}}$	global position of a rotor center
$\vec{p}_{n_p}^{\text{Plane}}$	global position of the center of wake plane n_p
RAM	random-access memory
RSS	root-sum-squared
r and r^{Plane}	radius in the axisymmetric coordinate system
$\hat{r}^{\text{Plane}}$	radial unit vector in the axisymmetric coordinate system
S	global X-, Y-, and Z-coordinate
SC	super controller (module)
SOWFA	Simulator fOr Wind Farm Applications
t	simulation time
TI_{Amb} and ${}^{\text{Filt}}TI_{\text{Amb}_{n_p}}$	ambient turbulence intensity of the wind at a rotor and its low-pass time-filtered value for wake plane n_p
u^d	discrete-time inputs
V_{Advect}	advection speed of the synthetic wind data
$\vec{V}_{\text{Amb}}^{\text{High}}$	ambient wind across a high-resolution wind domain around a turbine
$\vec{V}_{\text{Amb}}^{\text{Low}}$	ambient wind across a low-resolution wind domain throughout the wind farm
$\vec{V}_{\text{Dist}}^{\text{High}}$	disturbed wind across a high-resolution wind domain around a turbine
$\vec{V}_{\text{Dist}}^{\text{Low}}$	disturbed wind across a low-resolution wind domain throughout the wind farm
V_{Hub}	mean hub-height wind speed
$\vec{V}_{n_p}^{\text{Plane}}$ and ${}^{\text{Filt}}\vec{V}_{n_p}^{\text{Plane}}$	advection, deflection, and meandering velocity and its low-pass time-filtered value of wake plane n_p
V_r	radial velocity in the axisymmetric coordinate system
$V_{r_{n_p}}^{\text{Wake}}(r)$	radial wake-velocity deficit at wake plane n_p, distributed radially
VTK	Visualization Toolkit
${}^{\text{DiskAvg}}V_x^{\text{Rel}}$ and ${}^{\text{FiltDiskAvg}}V_x^{\text{Rel}}$	rotor-disk-averaged relative wind speed (ambient plus wakes of neighboring turbines plus turbine motion), normal to the disk, and its low-pass time-filtered value
V_x	axial velocity in the axisymmetric coordinate system
$V_{x_{n_p}}^{\text{Wake}}(r)$	axial wake-velocity deficit at wake plane n_p, distributed radially
${}^{\text{DiskAvg}}V_x^{\text{Wind}}$ and ${}^{\text{FiltDiskAvg}}V_{x_{n_p}}^{\text{Wind}}$	rotor-disk-averaged ambient wind speed, normal to the disk, and its low-pass time-filtered value at wake plane n_p
$w_{n^{\text{Wind}}}$	weighting in the spatial averaging for wind data point n^{Wind}
WD	wake dynamics (module)
WISDEM	Wind-Plant Integrated System Design & Engineering Model
x and $x_{n_p}^{\text{Plane}}$	downwind distance from a rotor to wake plane n_p in the axisymmetric coordinate system
X, Y, and Z	inertial-frame coordinates, with Z directed vertically upward, opposite gravity, X directed horizontally nominally downwind (along the zero-degree wind direction), and Y directed horizontally transversely

$\hat{X}$, $\hat{Y}$, and $\hat{Z}$	unit vectors of the inertial-frame coordinate system, parallel to the X, Y, and X coordinates
x^d	discrete-time states
$X^d(\quad)$	discrete-time state functions
$\hat{x}^{\text{Disk}}$	orientation of a rotor centerline
$\hat{x}_{n_p}^{\text{Plane}}$	orientation of wake plane n_p
y^d	discrete-time outputs
$Y^d(\quad)$	discrete-time output functions
z_{bot}	bottom vertical location of synthetic turbulence inflow grid
α	low-pass time-filter parameter
Δt	discrete time step (increment)
γ^{YawErr} and ${}^{\text{Filt}}\gamma_{n_p}^{\text{YawErr}}$	nacelle-yaw error of a rotor and its low-pass time-filtered value at wake plane n_p
ν_T	eddy viscosity
ρ	air density
2D	two dimensional
3D	three dimensional

Contents

List of Figures

List of Tables

1 Introduction

FAST.Farm is a midfidelity multiphysics engineering tool for predicting the power performance and structural loads of wind turbines within a wind farm. FAST.Farm uses OpenFAST to solve the aero-hydro-servo-elastic dynamics of each individual turbine, but considers additional physics for wind farm-wide ambient wind in the atmospheric boundary layer; a wind-farm super controller; and wake deficits, advection, deflection, meandering, and merging. FAST.Farm is based on some of the principles of the dynamic wake meandering (DWM) model—including passive tracer modeling of wake meandering—but addresses many of the limitations of previous DWM implementations. FAST.Farm maintains low computational cost to support the often highly iterative and probabilistic design process. Applications of FAST.Farm include reducing wind farm underperformance and loads uncertainty, developing wind farm controls to enhance operation, optimizing wind farm siting and topology, and innovating the design of wind turbines for the wind-farm environment. The existing implementation of FAST.Farm also forms a solid foundation for further development of wind farm dynamics modeling as wind farm physics knowledge grows from future computations and experiments.

The main idea behind the DWM model is to capture key wake features pertinent to accurate prediction of wind farm power performance and wind turbine loads, including the wake-deficit evolution (important for performance) and the wake meandering and wake-added turbulence (important for loads). The wake-deficit evolution and wake meandering are illustrated in Figure 1. Although fundamental laws of physics are applied, appropriate simplifications

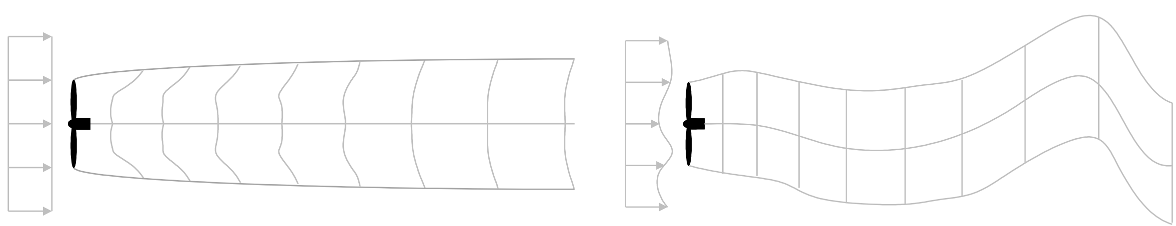


Figure 1. Axisymmetric wake deficit (left) and meandering (right) evolution.

have been made to minimize the computational expense, and high-fidelity modeling (HFM) solutions, e.g., using the Simulator fOr Wind Farm Applications (SOWFA), have been used to inform and calibrate the submodels. In the DWM model, the wake-flow processes are treated via the "splitting of scales," in which small turbulent eddies (less than two diameters) affect wake-deficit evolution and large turbulent eddies (greater than two diameters) affect wake meandering.

FAST.Farm is a nonlinear time-domain multiphysics engineering tool composed of multiple submodels, each representing different physics domains of the wind farm. FAST.Farm is implemented as open-source software that follows the programming requirements of the FAST modularization framework, whereby the submodels are implemented as modules interconnected through a driver code. The submodel hierarchy of FAST.Farm is illustrated in Figure 2. Wake advection, deflection, and meandering; near-wake correction; and wake-deficit increment are submodels of the wake-dynamics (*WD*) model, implemented in a single module. Ambient wind and wake merging are submodels of the ambient wind and array effects (*AWAE*) model, implemented in a single module. Combined with the super controller (*SC*) and OpenFAST (*OF*) modules, FAST.Farm has four modules and one driver. There are multiple instances of the *OF* and *WD* modules—one instance for each wind turbine/rotor.

1.1 FAST.Farm Driver

The FAST.Farm driver, also known as the "glue code," is the code that couples individual modules together and drives the overall time-domain solution forward. Additionally, the FAST.Farm driver reads an input file of simulation parameters, checks the validity of these parameters, initializes the modules, writes results to a file, and releases memory at the end of the simulation.

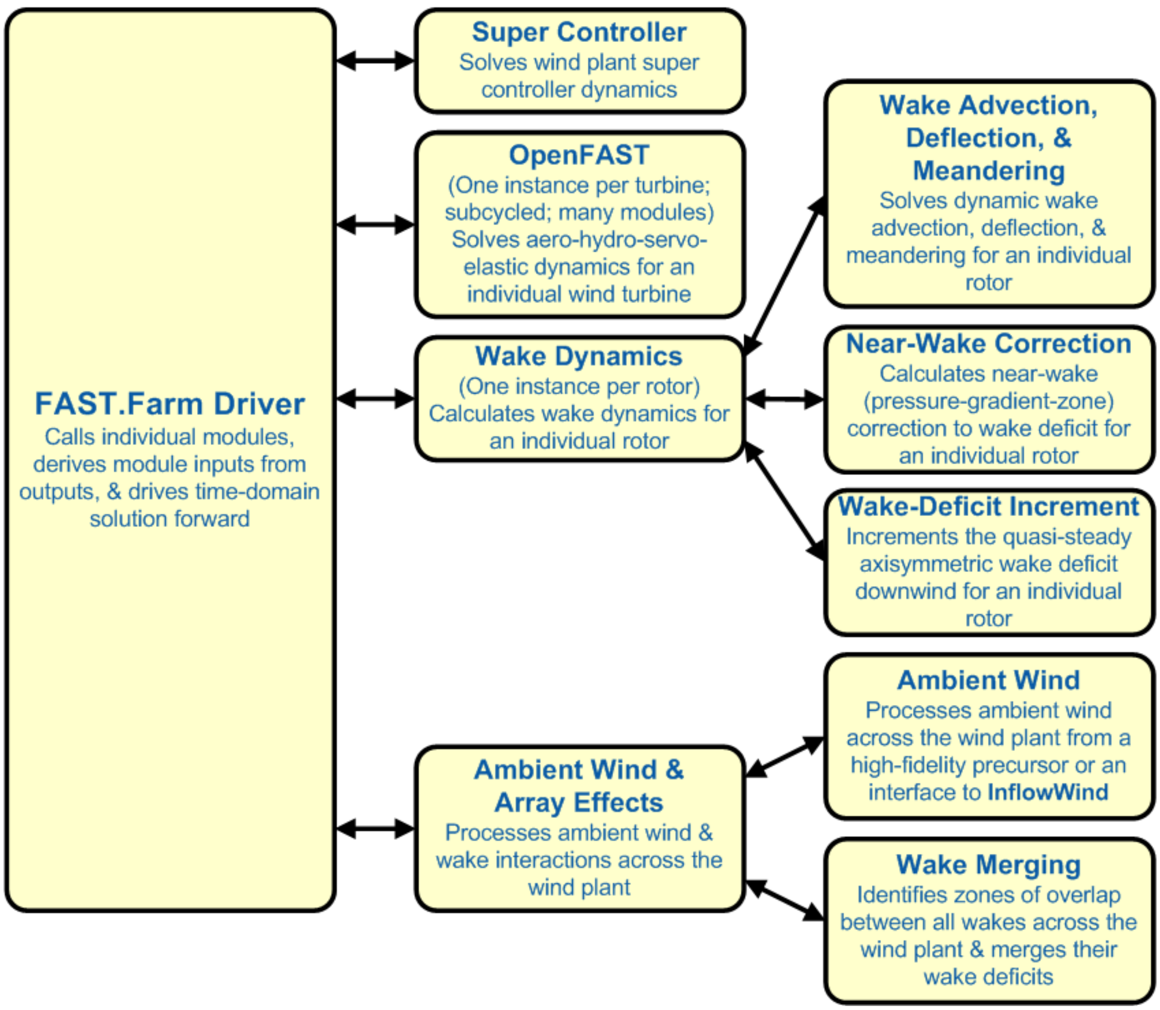


Figure 2. FAST.Farm submodel hierarchy.

1.2 Super Controller Module

The *SC* module of FAST.Farm—essentially identical to the super controller available in SOWFA—allows wind-farm-wide control logic to be implemented by the user, including sending and receiving commands from the individual turbine controllers in OpenFAST. The logic of such a super controller could be developed through the application of the National Renewable Energy Laboratory (NREL) code FLOw Redirection and Induction in Steady state (FLORIS).

1.3 OpenFAST Module

The *OF* module of FAST.Farm is a wrapper that enables the coupling of OpenFAST to FAST.Farm. OpenFAST models the dynamics (loads and motions) of distinct turbines in the wind farm, capturing the environmental excitations (wind inflow and, for offshore systems, waves, current, and ice) and coupled system response of the full system (the rotor, drivetrain, nacelle, tower, controller, and, for offshore systems, the substructure and station-keeping system). OpenFAST itself is an interconnection of various modules, each corresponding to different physical domains of the coupled aero-hydro-servo-elastic solution. There is one instance of the *OF* module for each wind turbine, which, in parallel mode, are parallelized through open multiprocessing (OpenMP). At initialization, the number of wind turbines, associated OpenFAST primary input file(s), and turbine origin(s) in the global *X*-*Y*-*Z* inertial-frame coordinate system are specified by the user of FAST.Farm. Turbine origins are defined as the intersection of the undeflected tower centerline and the ground or, for offshore systems, the mean sea level. The global inertial-frame coordinate system is defined with *Z* directed vertically upward (opposite gravity), *X* directed horizontally nominally downwind (along the zero-degree wind direction), and *Y* directed horizontally transversely. This coordinate system is not tied to specific compass directions. Among other time-dependent inputs from FAST.Farm, OpenFAST uses the disturbed wind (ambient plus wakes) across a high-resolution wind domain (in both time and space) around the turbine as input. This high-resolution domain ensures that the individual turbine loads and responses calculated by OpenFAST are accurately driven by flow through the wind farm, including wake and array effects.

1.4 Wake Dynamics Module

The *WD* module of FAST.Farm calculates wake dynamics for an individual rotor, including wake advection, deflection, and meandering; a near-wake correction; and a wake-deficit increment. The near-wake correction treats the near-wake (pressure-gradient zone) correction of the wake deficit. The wake-deficit increment shifts the quasi-steady-state axisymmetric wake deficit nominally downwind. There is one instance of the *WD* module for each rotor. The wake-dynamics calculations involve many user-specified parameters that may depend, e.g., on turbine operation or atmospheric conditions and can be calibrated to better match experimental data or by using an HFM solution as a benchmark. Default values have been derived for each calibrated parameter based on SOWFA simulations, but these can be overwritten by the user.

The wake-deficit evolution is solved in discrete time on an axisymmetric finite-difference grid consisting of a fixed number of wake planes, each with a fixed radial grid of nodes. The radial finite-difference grid can be considered a plane because the wake deficit is assumed to be axisymmetric. A wake plane can be thought of as a cross section of the wake wherein the wake deficit is calculated.

By simple extensions to the passive tracer solution for transverse (horizontal and vertical) wake meandering, the wake-dynamics solution in FAST.Farm is extended to account for wake deflection, as illustrated in Figure 3, and wake advection, as illustrated in Figure 4, among other physical improvements such as:

1. Calculating the wake-plane velocities by spatially averaging the disturbed wind instead of the ambient wind (in the AWAE module)
2. Orientating the wake planes with the rotor centerline instead of the wind direction
3. Low-pass time filtering the local conditions at the rotor, as input to the wake dynamics module, to account for transients in inflow, turbine control, and/or turbine motion instead of considering time-averaged conditions.

With these extensions, the passive tracer solution enables:

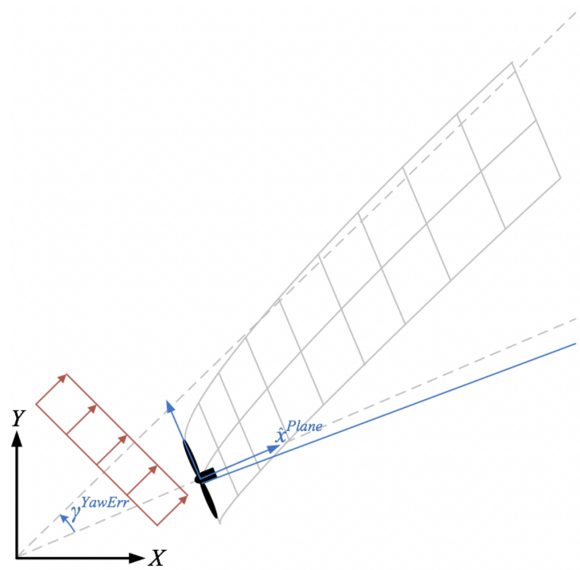


Figure 3. Wake deflection resulting from inflow skew, including a horizontal wake-deflection correction. The lower dashed line represents the rotor centerline, the upper dashed line represents the wind direction, and the solid blue line represents the horizontal wake-deflection correction (offset from the rotor centerline).

1. The wake centerline to deflect based on inflow skew, because in skewed inflow, the wake deficit normal to the disk introduces a velocity component that is not parallel to the ambient flow
2. The wake to accelerate from near wake to far wake, because the wake deficits are stronger in the near wake and weaken downwind
3. The wake-deficit evolution to change based on conditions at the rotor, because low-pass time filtering conditions are used instead of time-averaging
4. The wake to meander axially in addition to transversely, because local axial winds are considered
5. The wake shape to be elliptical instead of circular in skewed flow when looking downwind (the wake shape remains circular when looking down the rotor centerline).

From item 1 above, a horizontally asymmetric correction to the wake deflection is accounted for, i.e., a correction to the wake deflection resulting from the wake-plane velocity, which physically results from the combination of wake rotation and shear not modeled directly in the *WD* module (see Figure 3 for an illustration). This horizontal wake deflection correction is a simple linear correction (with a slope and offset), similar to the correction implemented in the wake model of FLORIS. Such a correction is important for accurate modeling of nacelle-yaw-based wake-redirection (wake-steering) wind farm control.

From item 3, low-pass time filtering is important because the wake reacts slowly to changes in local conditions at the rotor and because the wake evolution is treated in a quasi-steady-state fashion.

The near-wake correction submodel of the *WD* module computes the wake-velocity deficits at the rotor disk, as an inlet boundary condition for the wake-deficit evolution. To improve the accuracy of the far-wake solution, the near-wake correction accounts for the drop-in wind speed and radial expansion of the wake in the pressure-gradient zone behind the rotor that is not otherwise accounted for in the solution for the wake-deficit evolution.

As with most DWM implementations, the *WD* module of FAST.Farm models the wake-deficit evolution via the thin shear-layer approximation of the Reynolds-averaged Navier-Stokes equations under quasi-steady-state conditions in axisymmetric coordinates, with turbulence closure captured by using an eddy-viscosity formulation. The thin shear-layer approximation drops the pressure term and assumes that the velocity gradients are much bigger in the radial

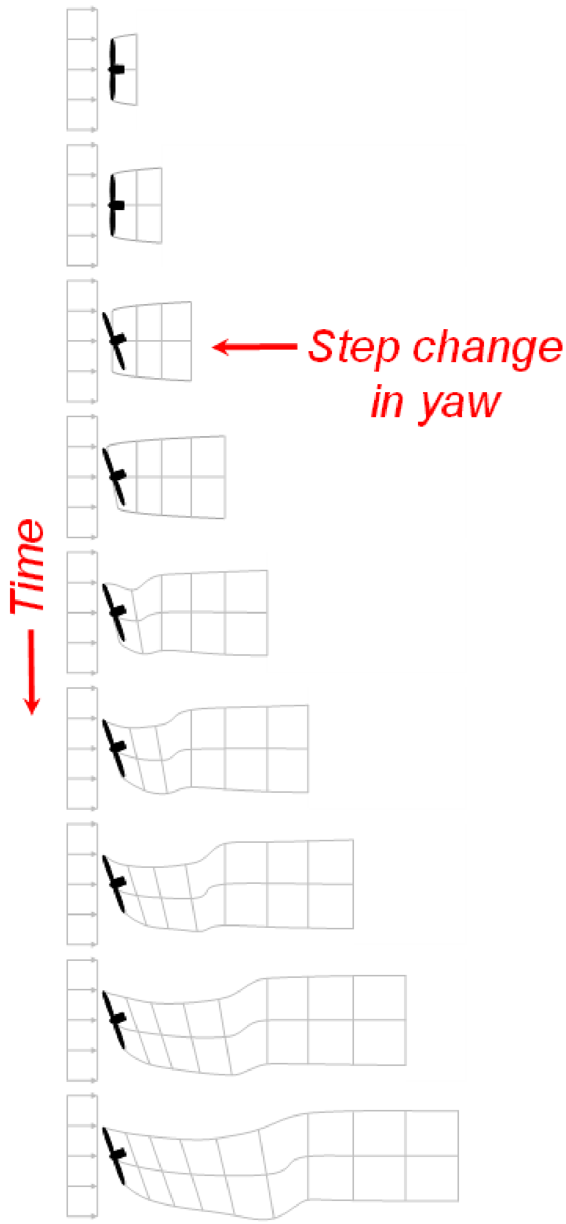


Figure 4. Wake advection for a single turbine resulting from a step change in yaw angle.

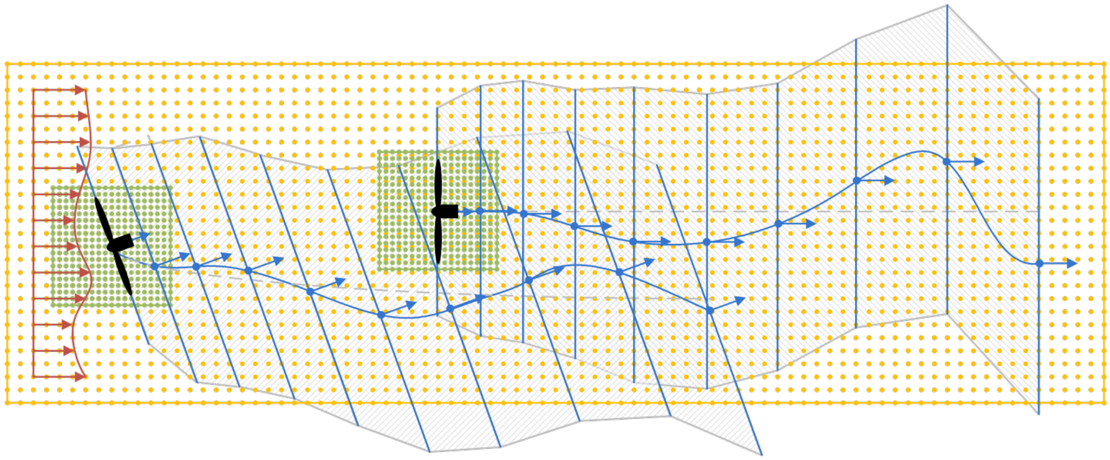


Figure 5. Wake planes, wake volumes, and zones of wake overlap for a two-turbine wind farm, with the upwind turbine yawed. The yellow points represent the low-resolution wind domain and the green points represent the high-resolution wind domains around each turbine. The blue points and arrows represent the centers and orientations of the wake planes, respectively, with the wake planes identified by the blue lines normal to their orientations. The gray dashed lines represent the mean trajectory of the wake and the blue curves represent the instantaneous [meandered] trajectories. The wake volumes associated with the upwind turbine are represented by the upward hatch patterns, the wake volumes associated with the downwind turbine are represented by the downward hatch patterns, and the zones of wake overlap are represented by the crosshatch patterns. (For clarity of the illustration, the instantaneous (meandered) wake trajectory is shown as a smooth curve, but will be modeled as piece-wise linear between wake planes when adjacent wake planes are parallel. The wake planes and volumes are illustrated with a diameter equal to twice the wake diameter, but the local diameter depends on the calculation. As illustrated, a wake plane or volume may extend beyond the boundaries of the low-resolution domain of ambient wind data.)

direction than in the axial direction.

1.5 Ambient Wind and Array Effects Module

The *AWAE* module of FAST.Farm processes ambient wind and wake interactions across the wind farm, including the ambient wind submodel, which processes ambient wind across the wind farm and the wake-merging submodel, which identifies zones of overlap between all wakes across the wind farm and merges their wake deficits. The calculations in the *AWAE* module make use of wake volumes, which are volumes formed by a (possibly curved) cylinder starting at a wake plane and extending to the next adjacent wake plane along a line connecting the centers of the two wake planes. If the adjacent wake planes (top and bottom of the cylinder) are not parallel, e.g., for transient simulations involving variations in nacelle-yaw angle, the centerline will be curved. Figure 5 illustrates some of the concepts. The calculations in the *AWAE* module also require looping through all wind data points, turbines, and wake planes; these loops have been sped up in the parallel mode of FAST.Farm by implementation of open multiprocessing (OpenMP) parallelization.

Ambient wind may come from either a high-fidelity precursor simulation or an interface to the *InflowWind* module in OpenFAST. The use of the *InflowWind* module enables the use of simple ambient wind, e.g., uniform wind, discrete wind events, or synthetically generated turbulent wind data. Synthetically generated turbulence can be generated from, e.g., TurbSim or the Mann model, in which the wind is propagated through the wind farm using Taylor's frozen-turbulence assumption. This method is most applicable to small wind farms or a subset of wind turbines within a larger wind farm. FAST.Farm can also use ambient wind generated by a high-fidelity precursor large-eddy simulation (LES) of the entire wind farm (without wind turbines present), such as the atmospheric boundary layer solver (ABLSolver) preprocessor of SOWFA. This atmospheric precursor simulation captures more physics than synthetic turbulence—as illustrated in Figure 6—including atmospheric stability, wind-farm-wide turbulent length scales, and complex terrain effects. This method is more computationally expensive than using the ambient wind

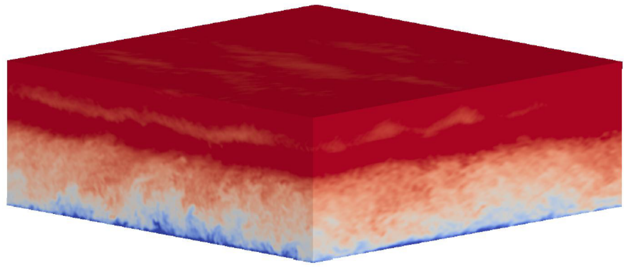


Figure 6. Example flow generated by ABLSolver.

modeling options of InflowWind, but it is much less computationally expensive than a SOWFA simulation with wind turbines present. FAST.Farm requires ambient wind to be available in two different resolutions in both space and time. Because wind will be spatially averaged across wake planes within the *AWAE* module, FAST.Farm needs a low-resolution wind domain throughout the wind farm wherever turbines may potentially reside. For accurate load calculation by OpenFAST, FAST.Farm also needs high-resolution wind domains around each wind turbine (encompassing any turbine displacement). The high-resolution domains will occupy the same space as portions of the low-resolution domain, requiring domain overlap.

When using ambient wind generated by a high-fidelity precursor simulation, the *AWAE* module reads in the three-component wind-velocity data across the high- and low-resolution domains that were computed by the high-fidelity solver within each time step. These values are stored in files for use in a given driver time step. The wind data files, including spatial discretizations, must be in Visualization Toolkit (VTK) format and are specified by users of FAST.Farm at initialization. Visualization Toolkit is an open-source, freely available software system for three-dimensional (3D) computer graphics, image processing, and visualization. When using the *InflowWind* inflow option, the ambient wind across the high- and low-resolution domains are computed by calling the *InflowWind* module. In this case, the spatial discretizations are specified directly within the FAST.Farm primary input file. These wind data from the combined low- and high-resolution domains within a given driver time step represent the largest memory requirement of FAST.Farm.

In previous implementations of DWM, the wind turbine and wake dynamics were solved individually or serially, not considering two-way wake-merging interactions. Additionally, there was no method available to calculate the disturbed wind in zones of wake overlap. Wake merging is illustrated by the FAST.Farm simulation of Figure 7. In FAST.Farm, the wake-merging submodel of the *AWAE* module identifies zones of wake overlap between all wakes across the wind farm by finding wake volumes that overlap in space. Wake deficits are superimposed in the axial direction based on the root-sum-squared (RSS) method. Transverse components (radial wake deficits) are superimposed by vector sum. The RSS method assumes that the local kinetic energy of the axial deficit in a merged wake equals the sum of the local energies of the axial deficits for each wake at the given wind data point. The RSS method only applies to an array of scalars. This method works well for axial deficits because overlapping wakes likely have similar axial directions; therefore, only the magnitude of the vector is important in the superposition. A vector sum is applied to the transverse components (radial wake deficits) because any given radial direction is dependent on the azimuth angle in the axisymmetric coordinate system.

To visualize the ambient wind and wake interactions across the wind farm, FAST.Farm includes visualization capability through the generation of output files in VTK format. OpenFAST can further generate VTK-formatted output files for visualizing the wind turbine based on either surface or stick-figure geometry. The VTK files generated by FAST.Farm and OpenFAST can be read with standard open-source visualization packages such as ParaView or VisIt.

1.6 FAST.Farm Parallelization

FAST.Farm can be compiled and run in serial or parallel mode. Parallelization has been implemented in FAST.Farm through OpenMP, which allows FAST.Farm to take advantage of multicore computers by dividing computational tasks among the cores/threads within a node (but not between nodes) to speed up a single simulation. The size of the

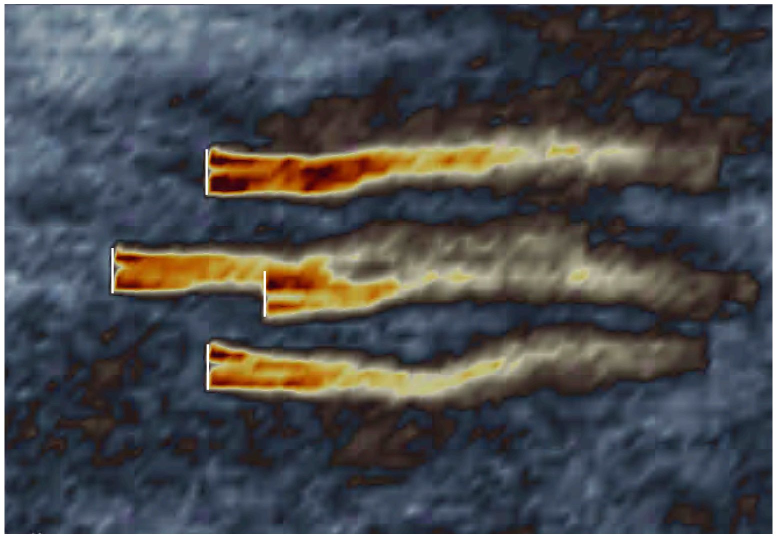


Figure 7. Wake merging for closely spaced rotors.

wind farm and number of wind turbines is limited only by the available random-access memory (RAM). In parallel mode, each instance of the OpenFAST submodel can be run in parallel on separate threads at the same time the ambient wind within the *AWAE* module is being read in another thread. Thus, the fastest simulations require at least one more core than the number of wind turbines in the wind farm. Furthermore, the output calculations within the *AWAE* module are parallelized into separate threads. Because of the small timescales involved and sophisticated physics, the *OF* submodel is the computationally slowest FAST.Farm module. The output calculation of the *AWAE* module is the only major calculation that cannot be solved in parallel to OpenFAST; therefore, at best, the parallelized FAST.Farm solution may execute only slightly more slowly than stand-alone OpenFAST simulations—computationally inexpensive enough to run the many simulations necessary for wind turbine/farm design and analysis.

To support the modeling of large wind farms, single simulations involving memory parallelization and parallelization between nodes of a multinode high-performance computer (HPC) through a message-passing interface (MPI) is likely required. MPI has not yet been implemented within FAST.Farm.

1.7 Organization of the Guide

The remainder of this documentation is structured as follows: Section 2 details how to obtain the FAST.Farm software archive and how to run FAST.Farm. Section 3 describes the FAST.Farm input files. Section 4 discusses the output files generated by FAST.Farm. Section 5 provides modeling guidance when using FAST.Farm. The FAST.Farm theory is covered in Section 6. Section 7 outlines future work, and the bibliography provides background and other information sources. Example FAST.Farm primary input and ambient wind data files are shown in Appendices A and B. A summary of available output channels is found in Appendix C.

2 Running FAST.Farm

This section discusses how to obtain and execute FAST.Farm for Mac/Linux and Windows®.

2.1 Downloading the FAST.Farm Software

This process is the same as OpenFAST, detailed in the OpenFAST documentation.

2.2 Compiling FAST.Farm Software

2.2.1 Windows Machine Using Visual Studio

A Visual Studio solution has been provided in the FAST.Farm download for compiling FAST.Farm on Windows®. Simply:

- Open up the Visual Studio solution file (*FAST-Farm.sln*)
- Select the desired Solution Configuration (*Release* for serial mode, *Release_OpenMP* with OpenMP parallelization, or *Debug* for serial debug mode [not optimized]), and the desired Solution Platform (*Win32* or *x64*) by using the drop-down boxes located below the menu bar
- Build the solution using the *Build->Build Solution* menu option.

For additional details, see the OpenFAST documentation.

2.2.2 Mac or Linux Machine Using CMake

Example FAST.Farm-specific build commands are as follows:

```
# For baseline FAST.Farm configuration

$ cmake .. -DBUILD_FAST_FARM=ON

# For memory and speed improvements (recommended)

$ cmake .. -DBUILD_FAST_FARM=ON -DCMAKE_Fortran_FLAGS_RELEASE="-O2 -xhost"
-DDOUBLE_PRECISION:BOOL=OFF

# For using OpenMP on a cluster

$ cmake .. -DBUILD_FAST_FARM=ON -DCMAKE_Fortran_FLAGS_RELEASE="-O2 -xhost"
-DDOUBLE_PRECISION:BOOL=OFF -DOPENMP=ON

# For all compilations

$ make
```

For additional details, see the OpenFAST documentation.

2.3 Running FAST.Farm Software

2.3.1 Windows Machine

Compiled FAST.Farm program executables are available for 64-bit Windows® machines for serial mode—*FAST.Farm_x64.exe*—or parallel mode—*FAST.Farm_x64_OMP.exe*. The serial-mode executable has been compiled with static-link libraries, and thus should run on any Windows® platform. The parallel-mode (OMP) executable must be compiled with dynamic-link libraries (DLLs), and thus the executable depends on Intel® Fortran DLLs that are not included in this distribution of FAST.Farm.

The OMP executable was compiled with Intel® Parallel Studio XE 2017 Update 2 Composer Edition for Fortran Windows® 2017.0.2.187. If you do not have this compiler (or similar version) installed, you must download and install the redistributable library found here: https://software.intel.com/sites/default/files/managed/a5/22/ww_ifort_redist_msi_2017.2.187.zip. If you do have this compiler installed, make sure the proper environment paths are set before running the OMP executable of FAST.Farm (e.g., open the “Compiler 17.0 Update 2 Intel® 64” command prompt).

To run FAST.Farm from a Windows® command prompt, the syntax is:

$ FAST.Farm_x64_OMP.exe MyInputFile.fstf

where *MyInputFile.fstf* is the name of the FAST.Farm primary input file, as described in Section 3.2.

2.3.2 Mac or Linux Machine

To run FAST.Farm from a Mac or Linux machine command prompt, the syntax is:

$ <rootdir>/build/glue-codes/fast-farm/FAST.Farm MyInputFile.fstf

Other than the name of the executable, running FAST.Farm is identical to running OpenFAST. For additional guidance, see the OpenFAST documentation.

Checkpoint-restart capability has not yet been implemented within FAST.Farm.

3 Input Files

The primary FAST.Farm input file defines ambient wind, the wind turbine layout within the wind farm, the wake axisymmetric finite-difference grid, calibrated parameters for wake dynamics, visualization output, output file specifications, and auxiliary parameters. Ambient wind data optionally generated from the high-fidelity precursor atmospheric simulation are stored in separate files referenced in the primary FAST.Farm input file. Properties for each wind turbine in the wind farm are stored in the standard OpenFAST input files, referenced by their primary OpenFAST input file (one for each wind turbine) in the primary FAST.Farm input file.

No lines should be added or removed from the input files, except in tables where the number of rows is specified.

3.1 Units

FAST.Farm uses the SI system (kg, m, s, N).

3.2 FAST.Farm Primary Input File

The FAST.Farm primary input file is organized into several functional sections:

- Simulation Control
- Super Controller
- Ambient Wind
- Wind Turbines
- Wake Dynamics
- Visualization
- Output

Each section corresponds to an aspect of the FAST.Farm model—see the subsections below. A sample FAST.Farm primary input file is given in Appendix A. Where there is a one-to-one equivalency between an input parameter and a variable in the FAST.Farm theory documented in Section 6, the variable in Section 6 is shown in parentheses after the input parameter in the subsections below.

The input file begins with two lines of header information that is for your use, but is not used by the software.

3.2.1 Simulation Control

Echo [flag] specifies if you wish to have FAST.Farm echo the contents of the FAST.Farm primary input file (useful for debugging errors in the input file). If ***Echo*** = TRUE, an echo file will be generated. The echo file has the naming convention of *<RootName>.ech*, where *<RootName>* is the name of the FAST.Farm primary input file, excluding its file extension.

AbortLevel [quoted string] indicating what error level should cause an abort. Options are: "WARNING," "SEVERE," or "FATAL." ***AbortLevel*** in FAST.Farm is used the same way as the level set in stand-alone OpenFAST, but the ***AbortLevel*** set in FAST.Farm will override the levels set in the OpenFAST primary input file of each wind turbine in the wind farm. Setting FAST.Farm to abort on fatal errors is typical, but see the FAST v8 ReadMe document for additional guidance.

TMax [sec] is the total length of the simulation to be run. The first output is calculated at $t = 0$; the last output is calculated at *t* = ***TMax***. The ***TMax*** set in FAST.Farm will override the simulation length set in the OpenFAST primary input file of each wind turbine in the wind farm.

UseSC [flag] indicates if the wind-farm-wide super controller is to be used. If ***UseSC*** = TRUE, the super controller will be called. If ***UseSC*** = FALSE, the super controller will not be called, but each wind turbine may still have an individual controller specified in the OpenFAST module *ServoDyn*.

Mod_AmbWind [switch] indicates the ambient wind source. There are three options: 1) use ambient wind data generated by a high-fidelity precursor simulation in VTK format ***[Mod_AmbWind=1]***, 2) use ambient wind data as defined by the FAST.Farm interface to the *InflowWind* module, with one instance of *InflowWind* ***[Mod_AmbWind=2]***, or 3) use ambient wind data as defined by the FAST.Farm interface to the *InflowWind* module, with multiple instances of *InflowWind* ***[Mod_AmbWind=3]***. The distinct AMBIENT WIND subsections below pertain to each option.

3.2.2 Super Controller

SC_FileName [quoted string] sets the name and location of the dynamic library containing the super controller code. It is only used when ***UseSC*** = TRUE. The dynamic library should be compiled as a *.dll* file in Windows® or a *.so* file in Linux or Mac OS. **The file name must be in quotations** and can contain an absolute or a relative path. The super controller is used in conjunction with individual wind turbine controllers defined in the style of the DISCON dynamic library of the DNV GL's Bladed wind turbine software package, with minor modification. See Section 5.5 for more information.

3.2.3 Ambient Wind: Precursor in Visualization Toolkit Format

The input parameters in this section are only used when ***Mod_AmbWind*** = 1, indicating the use of ambient wind generated by a high-fidelity precursor simulation. In this case, the ambient wind, including their spatial discretization, must be stored in VTK format—as described in Section 3.3—and is used directly without modification by FAST.Farm.

DT_Low-VTK [sec] (t) sets the time step of the low-resolution ambient wind data files and calculation, as well as the global (driver/glue-code) time step of FAST.Farm. ***DT_Low-VTK*** is the same as ***DT_Low*** in this documentation. The modules of FAST.Farm are called every ***DT_Low*** seconds, although OpenFAST and its modules may use a time step that is an integer multiple smaller than or equal to ***DT_Low***.

DT_High-VTK [sec] sets the time step of the high-resolution ambient wind data files and calculation and **must be an integer multiple smaller than or equal to *DT_Low***. ***DT_High-VTK*** is the same as ***DT_High*** in this documentation. It is essential that ***DT_Low*** and ***DT_High*** are small enough to ensure solution accuracy and match the time resolution used when generating the ambient wind data from the high-fidelity precursor simulation. ***DT_Low*** should be consistent with the timescales of wake dynamics, e.g., on the order of seconds and smaller for higher mean wind speeds. ***DT_High*** should be sufficient for accurate aerodynamic load calculations, e.g., on the order of fractions of a second. Further guidance on choosing appropriate time steps is given in Section 5.

WindFilePath [quoted string] specifies the path to the directory where the low- and high-resolution ambient wind data files are stored. The path can be specified relative to the location of the FAST.Farm primary input file or with an absolute path. It is recommended to use quotes around the path. If there are spaces in the file or path names, these quotes are required. **FAST.Farm requires that the ambient wind data files be stored in specific subdirectories of the directory specified by *WindFilePath* and with specific filenames.** The low-resolution ambient wind data files must be named *Amb.t*<n_{low}>*.vtk* and stored in a subdirectory named *Low*. In the file names, <n_{low}> is an integer (without leading zeros) between *0* (at $t = 0$) and *N-1*, where $N = FLOOR\left(\frac{TMax}{DT_{\text{Low}}}\right) + 1$ is the number of low-resolution time steps. The high-resolution ambient wind data files must be named *Amb.t*<n_{high}>*.vtk*, where <n_{high}> is an integer (without leading zeros) between 0 (at $t = 0$) and $\frac{DT_{\text{Low}}}{DT\_High}(N-1)$. The files must be stored in a subdirectory named *HighT*<n_t>, where <n_t> is an integer (without leading zeros) between 1 and the total number of wind turbines (***NumTurbines***). Subdirectory *HighT*<n_t> must contain the high-resolution ambient wind data corresponding to wind turbine <n_t>, specified in the WIND TURBINES section of the FAST.Farm primary input file—see Section 3.2.5. The VTK format of each ambient wind data file—for both the low-resolution and high-resolution domains—is identical, as described in Section 3.3.

ChkWndFiles [flag] specifies if FAST.Farm should check the ambient wind data files for consistency before running the simulation (preventing a possible crash later). As this check is time intensive, it is recommended that ***ChkWndFiles*** be set to FALSE (to disable the check) if the ambient wind data have previously been checked, such as in a prior simulation. If set to TRUE, FAST.Farm will check to ensure that:

- The number of low-resolution ambient wind data files is sufficient to run the entire simulation (up to $t =$ ***TMax***). If more files are in the subdirectory, only the first N will be used.
- The number of high-resolution ambient wind data files is sufficient to run the entire simulation (up to $t =$ ***TMax***) for all wind turbines. If there are more subdirectories, only the first ***NumTurbines*** will be used. If more files are in each subdirectory, only the first $\frac{DT_{\text{Low}}}{DT\_High}(N-1)+1$ will be used.
- The spatial resolution (number of grid points, origin, and spacing) of each low-resolution ambient wind data file is the same.
- The spatial resolution (number of grid points, origin, and spacing) of each high-resolution ambient wind data file is the same for a given wind turbine.
- The number of grid points in each high-resolution domain is the same for all wind turbines in the wind farm.

3.2.4 Ambient Wind: InflowWind Module

The input parameters in this section are only used when ***Mod_AmbWind*** = 2 or 3, indicating the use of ambient wind through one or multiple instances of the *InflowWInd* module. In this case, the ambient wind specified within *InflowWInd* is interpolated to the low- and high-resolution domains for use within FAST.Farm.

DT_Low [sec] (Δt) sets the time step of the low-resolution ambient wind calculation, as well as the global (driver/glue-code) time step of FAST.Farm. The modules of FAST.Farm are called every ***DT_Low*** seconds, although OpenFAST and its modules may choose to use a time step that is an integer multiple smaller than or equal to ***DT_Low***.

DT_High [sec] sets the time step of the high-resolution ambient wind data calculation and must be an integer multiple smaller than or equal to ***DT_Low***. It is essential that ***DT_Low*** and ***DT_High*** are small enough to ensure solution accuracy. ***DT_Low*** should be consistent with the timescales of wake dynamics, e.g., on the order of seconds and smaller for higher mean wind speeds. ***DT_High*** should be sufficient for accurate aerodynamic load calculations, e.g., on the order of fractions of a second. Further guidance on choosing appropriate time steps is given in Section 5.

The next nine input parameters set the spatial discretization of the low-resolution ambient wind domain. The low-resolution domain is stored as a structured 3D grid of wind data points (representing the corners of 3D cells) in the global *X*-*Y*-*Z* inertial-frame coordinate system, as illustrated generically in Figure 8.

NX_Low, ***NY_Low***, and ***NZ_Low*** [integer] set the number of wind data points in each direction.

X0_Low, ***Y0_Low***, and ***Z0_Low*** [m] set the origin of the grid (lowest-most *X*-*Y*-*Z* coordinate).

dX_Low, ***dY_Low***, and ***dZ_Low*** [m] set the spatial discretization in each direction.

The total low-resolution domain size has dimensions (***NX_Low***-1)***dX_Low*** × (***NY_Low***-1)***dY_Low*** × (***NZ_Low***-1)***dZ_Low***. The low-resolution domain should extend throughout the wind farm wherever turbines and wakes may potentially reside with a resolution sufficient so that the spatial averaging is accurate, e.g., on the order of tens of meters for utility-scale wind turbines. Further guidance on choosing appropriate spatial discretization is given in Section 5.

Like the low-resolution domain, each high-resolution domain is stored as a structured 3D grid of wind data points in the global *X*-*Y*-*Z* inertial-frame coordinate system—as illustrated generically in Figure 8.

NX_High, ***NY_High***, and ***NZ_High*** [integer] set the number of wind data points in each direction. These values are the same for each wind turbine and so only need to be set once.

The origin and spatial discretization for the high-resolution wind domain for each turbine are specified in the WIND TURBINES section of the FAST.Farm primary input file below.

InflowFile [quoted string] specifies the name of the primary input file for the *InflowWind* module, which can be specified relative to the location of the FAST.Farm primary input file or specified with an absolute path. It is recommended to use quotes around the file name. If there are spaces in the file or path names, these quotes are required. See Section 3.4 for information on the contents of this file.

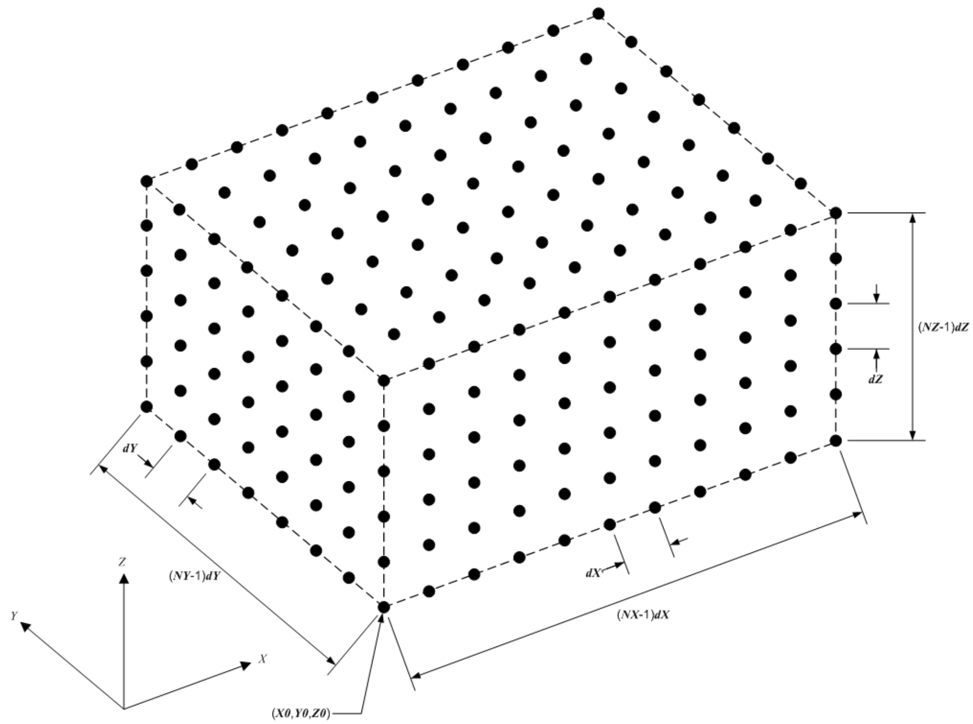


Figure 8. Structured 3D grid for the low- or high-resolution domains.

3.2.5 Wind Turbines

NumTurbines [integer] (N_t) is the number of wind turbines in the wind farm and determines the number of rows in the subsequent table (after two table header lines).

For each wind turbine:

- ***WT_X***, ***WT_Y***, and ***WT_Z*** [m] specify the origin in the global *X-Y-Z* inertial-frame coordinate system. The origin is defined as the intersection of the undeflected tower centerline and the ground or, for offshore systems, mean sea level.
- ***WT_FASTInFile*** [quoted string] specifies the name of the OpenFAST primary input file associated with each turbine. Each wind turbine is numbered within FAST.Farm as an integer (n_t) between 1 and ***NumTurbines*** corresponding to the row in the table. The OpenFAST primary input file name can be specified relative to the location of the FAST.Farm primary input file or with an absolute path. It is recommended to use quotes around the file name. Identical wind turbines can use the same OpenFAST primary input file, except if the corresponding OpenFAST model makes use of a Bladed-style controller in DLL format or, for offshore wind turbines, if different wave conditions are required for each turbine. If a Bladed-style DLL controller is being used, distinct Bladed-style controller DLLs must be used (each with a unique name). This requires the need for distinct *ServoDyn* primary input files, referencing the appropriate DLL name, and distinct OpenFAST primary input files, each referencing the appropriate *ServoDyn* primary input file name. If different wave conditions are required for each turbine, the distinct wave conditions (e.g., based on unique random wave seeds) for each wind turbine must be set in the *HydroDyn* primary input file and distinct OpenFAST primary input files must be used, each referencing the appropriate *HydroDyn* primary input file name. See Section 3.5 for information on the contents of the OpenFAST input files.
- When ***Mod_AmbWind*** = 2 or 3, the WIND TURBINES table has six additional columns to complete the spatial discretization of the high-resolution wind domain for each wind turbine:
  - ***X0_High***, ***Y0_High***, and ***Z0_High*** [m] set the origin of the grid.

– ***dX_High***, ***dY_High***, ***dZ_High*** [m] set spatial discretization in each direction.

The total high-resolution domain size has dimensions (***NX_High***-1)***dX_High*** × (***NY_High***-1)***dY_High*** × (***NZ_-High***-1)***dZ_High***. Each high-resolution domain must extend around the corresponding wind turbine, encompassing any turbine displacement. The domains should have a resolution sufficient for accurate aerodynamic load calculations, e.g., on the order of the blade chord length. The high-resolution domains will occupy the same space as portions of the low-resolution domain, requiring domains overlap.

3.2.6 Wake Dynamics

With FAST.Farm, each wake plane is treated as a radial finite-difference grid, as shown in Figure 9. These planes are defined by the following parameters:

- ***dr*** [m] sets the radial increment. To ensure the wake deficits are accurately computed by FAST.Farm, ***dr*** should be set so that FAST.Farm sufficiently resolves the wake deficit within each plane.
- ***NumRadii*** [integer] (N_r) sets the number of radii. To ensure the wake deficits are accurately computed by FAST.Farm, ***NumRadii*** should be set so that the diameter of each wake plane, 2(***NumRadii***-1)***dr***, is large relative to the rotor diameter.
- ***NumPlanes*** [integer] (N_p) sets the number of wake planes. To ensure the wake deficits are accurately captured by FAST.Farm, ***NumPlanes*** should be set so that the wake planes propagate a sufficient distance downstream, preferably until the wake deficit decays away.

The next 20 inputs are user-specified calibration parameters and options that influence the wake-dynamics calculations. The parameters may depend, e.g., on turbine operation or atmospheric conditions that can be calibrated to better match experimental data or by using an HFM benchmark. Default values have been derived for each calibrated parameter based on SOWFA simulations (Doubrawa et al. 2018), but these can be overwritten by the user.

f_c [Hz] (f_c) is the cutoff (corner) frequency of the low-pass time filter for the wake advection, deflection, and meandering model and must be greater than zero. If the DEFAULT keyword is specified in place of a numerical value, ***f_c*** is set to 0.0007.

C_HWkDfl_O [m] (C^{O}_{HWkDfl}) is the calibrated parameter for the wake deflection correction defining the horizontal offset at the rotor. If the DEFAULT keyword is specified in place of a numerical value, ***C_HWkDfl_O*** is set to 0.0.

C_HWkDfl_OY [m/deg] (C^{OY}_{HWkDfl}) is the calibrated parameter for the wake deflection correction defining the horizontal offset at the rotor scaled with yaw error. If the DEFAULT keyword is specified in place of a numerical value, ***C_HWkDfl_OY*** is set to 0.3.

C_HWkDfl_x [-] (C^{x}_{HWkDfl}) is the calibrated parameter for the wake deflection correction defining the horizontal offset scaled with downstream distance. If the DEFAULT keyword is specified in place of a numerical value, ***C_-HWkDfl_x*** is set to 0.0.

C_HWkDfl_xY [1/deg] (C^{xY}_{HWkDfl}) is the calibrated parameter for the wake deflection correction defining the horizontal offset scaled with downstream distance and yaw error. If the DEFAULT keyword is specified in place of a numerical value, ***C_HWkDfl_xY*** is set to -0.004.

C_NearWake ($C_{NearWake}$) [-] is the calibrated parameter for the near-wake correction and must be greater than one. If the DEFAULT keyword is specified in place of a numerical value, ***C_NearWake*** is set to 1.8.

k_vAmb [-] (k_{vAmb}) is the calibrated parameter for the ambient turbulence influence in the eddy viscosity and must be greater than zero. If the DEFAULT keyword is specified in place of a numerical value, ***k_vAmb*** is set to 0.05.

k_vShr [-] (k_{vShr}) is the calibrated parameter for the wake shear layer influence in the eddy viscosity and must be greater than zero. If the DEFAULT keyword is specified in place of a numerical value, ***k_vShr*** is set to 0.016.

C_vAmb_DMin [-] (C^{DMin}_{vAmb}) is a calibrated parameter in the eddy viscosity filter function for ambient turbulence. It defines the transitional diameter fraction between the minimum and exponential regions and must be greater than or equal to zero. If the DEFAULT keyword is specified in place of a numerical value, ***C_vAmb_DMin*** is set to 0.0.

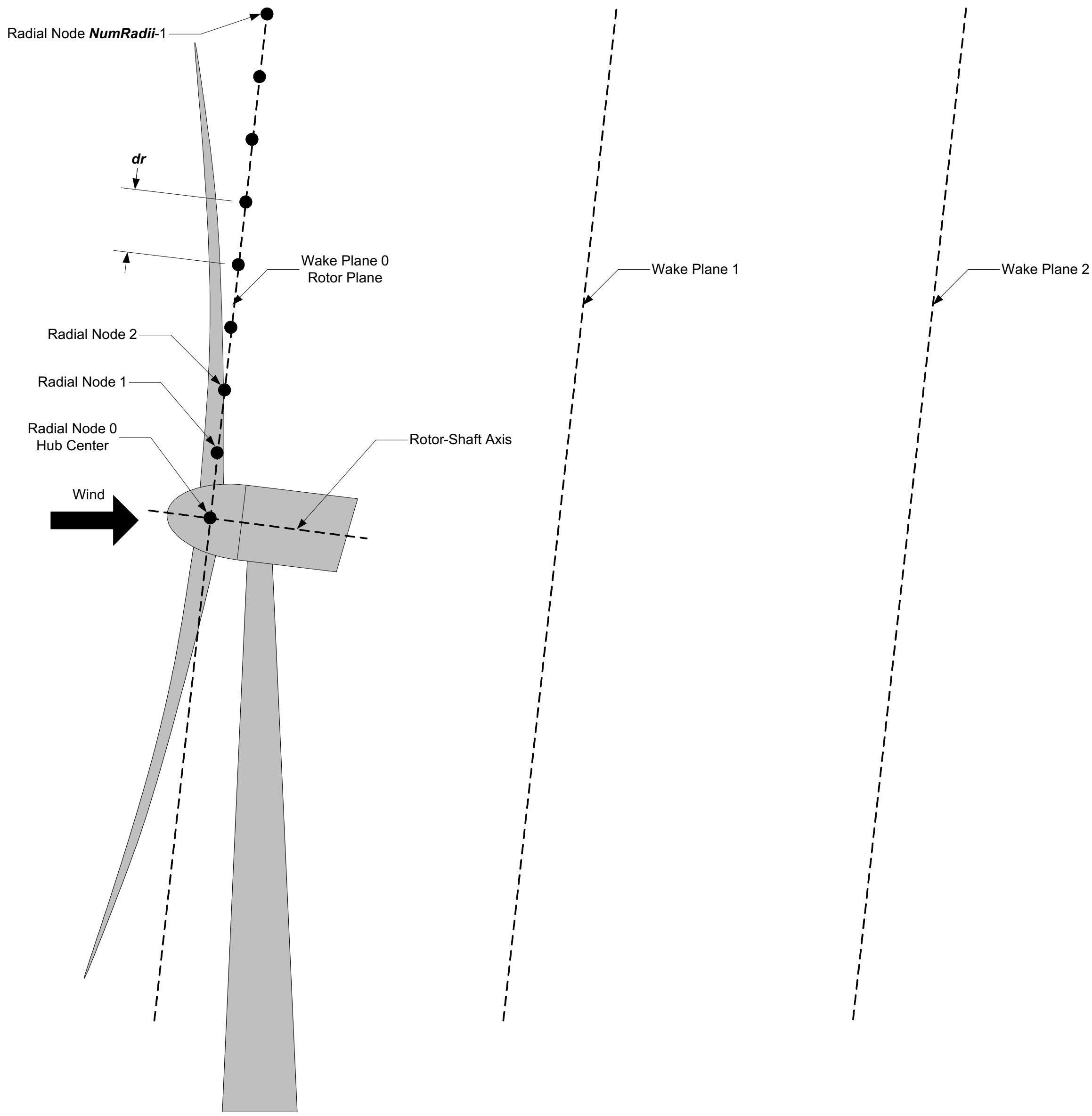


Figure 9. Radial finite-difference grid. For clarity of the illustration, the number and size of the wake planes are shown smaller than they should be.

C_vAmb_DMax [-] (C_{vAmb}^{DMax}) is a calibrated parameter in the eddy viscosity filter function for ambient turbulence. It defines the transitional diameter fraction between the exponential and maximum regions and must be greater than ***C_vAmb_DMin***. If the DEFAULT keyword is specified in place of a numerical value, ***C_vAmb_DMax*** is set to 1.0.

C_vAmb_FMin [-] (C_{vAmb}^{FMin}) is a calibrated parameter in the eddy viscosity filter function for ambient turbulence. It defines the value in the minimum region and must be between zero and one (inclusive). If the DEFAULT keyword is specified in place of a numerical value, ***C_vAmb_FMin*** is set to 1.0.

C_vAmb_Exp [-] (C_{vAmb}^{Exp}) is a calibrated parameter in the eddy viscosity filter function for ambient turbulence. It defines the exponent in the exponential region and must be greater than zero. If the DEFAULT keyword is specified in place of a numerical value, ***C_vAmb_Exp*** is set to 0.01.

C_vShr_DMin [-] (C_{vShr}^{DMin}) is a calibrated parameter in the eddy viscosity filter function for the wake shear layer. It defines the transitional diameter fraction between the minimum and exponential regions and must be greater than or equal to zero. If the DEFAULT keyword is specified in place of a numerical value, ***C_vShr_DMin*** is set to 3.0.

C_vShr_DMax [-] (C_{vShr}^{DMax}) is a calibrated parameter in the eddy viscosity filter function for the wake shear layer. It defines the transitional diameter fraction between the exponential and maximum regions and must be greater than ***C_vShr_DMin***. If the DEFAULT keyword is specified in place of a numerical value, ***C_vShr_DMax*** is set to 25.0.

C_vShr_FMin [-] (C_{vShr}^{FMin}) is a calibrated parameter in the eddy viscosity filter function for the wake shear layer. It defines the value in the minimum region and must be between zero and one (inclusive). If the DEFAULT keyword is specified in place of a numerical value, ***C_vShr_FMin*** is set to 0.2.

C_vShr_Exp [-] (C_{vShr}^{Exp}) is a calibrated parameter in the eddy viscosity filter function for the wake shear layer. It defines the exponent in the exponential region and must be greater than zero. If the DEFAULT keyword is specified in place of a numerical value, ***C_vShr_Exp*** is set to 0.1.

Mod_WakeDiam [switch] specifies the wake diameter calculation model (method). There are four options: 1) use the rotor diameter [***Mod_WakeDiam=1***]; 2) use a velocity-based method [***Mod_WakeDiam=2***]; 3) use a mass-flux based method [***Mod_WakeDiam=3***]; or 4) use a momentum-flux based method [***Mod_WakeDiam=4***]. If the DEFAULT keyword is specified in place of a numerical value, ***Mod_WakeDiam*** is set to 1.

C_WakeDiam [-] ($C_{WakeDiam}$) is the calibrated parameter for the wake diameter calculation and must be greater than zero and less than 0.99. It is unused when ***Mod_WakeDiam=1***. If the DEFAULT keyword is specified in place of a numerical value, ***C_WakeDiam*** is set to 0.95.

Mod_Meander [switch] specifies the spatial filter model (method) for wake meandering. There are three options: 1) use a uniform spatial average [***Mod_Meander=1***]; 2) use a truncated *jinc* [***Mod_Meander=2***]; or 3) use a windowed *jinc* [***Mod_Meander=3***]. If the DEFAULT keyword is specified in place of a numerical value, ***Mod_Meander*** is set to 3.

C_Meander [-] ($C_{Meander}$) is the calibrated parameter for the wake meandering and must be greater than or equal to one. If the DEFAULT keyword is specified in place of a numerical value, ***C_Meander*** is set to 1.9.

3.2.7 Visualize

WrDisWind [flag] specifies whether full 3D low- and high-resolution disturbed wind data output files will be generated. These files show the ambient wind and wake interactions across the wind farm for visualization and are generated if ***WrDisWind***=TRUE. The VTK data format and spatial resolutions (number of grid points, origin, and spacing) of these output files match those of the corresponding low- and high-resolution ambient wind data used by the FAST.Farm simulation. The VTK files are written to a directory named *vtk_ff* where the FAST.Farm primary file is stored. The naming conventions of these output files are *<RootName>.Low.Dis.<n_{low}>.vtk* and *<RootName>.HighT*<n_t>*.Dis.<n_t>.vtk* for the low- and high-resolution disturbed wind data files, respectively, where *<RootName>* is the name of the FAST.Farm primary input file, excluding its file extension, where <n_t> and <n_{low}> are as specified in Section 3.2.3, but include leading zeros.

For visualization, FAST.Farm can also output low-resolution disturbed (including wakes) wind data output files that are two-dimensional (2D) slices of the full low-resolution domain, specified by the following 7 inputs. Up to nine 2D slices parallel to the *X*-*Y*, *Y*-*Z*, and/or *X*-*Z* planes can be output.

- ***NOutDisWindXY*** [integer] specifies the number of 2D slices parallel to the *X*-*Y* plane where low-resolution disturbed wind data output files are output (0 to 9).
- ***OutDisWindZ*** [m] is a list ***NOutDisWindXY*** values long of the *Z* coordinates of each plane that will be output. These values are in the **global inertial-frame coordinate system**, separated by any combination of commas, semicolons, spaces, and/or tabs.
- ***NOutDisWindYZ*** [integer] specifies the number of 2D slices parallel to the *Y*-*Z* plane where low-resolution disturbed wind data output files are output (0 to 9).
- ***OutDisWindX*** [m] is a list ***NOutDisWindYZ*** values long of the *X* coordinates of each plane that will be output. These values are in the **global inertial-frame coordinate system**, separated by any combination of commas, semicolons, spaces, and/or tabs.
- ***NOutDisWindXZ*** [integer] specifies the number of 2D slices parallel to the *X*-*Z* plane where low-resolution disturbed wind data output files are output (0 to 9).
- ***OutDisWindY*** [m] is a list ***NOutDisWindXZ*** values long of the *Y* coordinates of each plane that will be output. These values are in the **global inertial-frame coordinate system**, separated by any combination of commas, semicolons, spaces, and/or tabs.

The VTK files are written to a directory named *vtk_ff* where the FAST.Farm primary file is stored. The naming conventions of these output files are *<RootName>.Low.DisXY<*n_{Out}*>.<*n_{low}*>.vtk*, *<RootName>.Low.DisYZ<*n_{Out}*>.<*n_{low}*>.vtk*, and *<RootName>.Low.DisXZ<*n_{Out}*>.<*n_{low}*>.vtk* for the *X*-*Y*, *Y*-*Z*, and *X*-*Z* slices, respectively, where <n_{Out}> is an integer between 1 and 9 corresponding to which slice is output. *<RootName>* and <n_{low}> are as defined in Section 3.2.3, but include leading zeros.

WrDisDT [sec] specifies the time step (inverse of the frame rate) of all disturbed wind data output files and must be an integer multiple larger than or equal to ***DT_Low***. This input is unused when ***WrDisWind*** = FALSE and when ***NOutDisWindXY***, ***NOutDisWindYZ***, and ***NOutDisWindXZ*** are set to zero. If the DEFAULT keyword is specified in place of a numerical value, ***WrDisDT*** is set to ***DT_Low***. Note that the full high-resolution disturbed wind data output files are not output at a frame rate of 1/***DT_High***, but are only output every ***WrDisDT*** seconds.

Visualizing the ambient wind and wake interactions can be useful for interpreting results and debugging problems. However, FAST.Farm will generate $n+1$ files per output option when ***WrDisWind*** = TRUE and/or when ***NOutDisWindXY***, ***NOutDisWindYZ***, and/or ***NOutDisWindXZ*** are set greater than zero. This file generation will slow down FAST.Farm and take up a lot of disk space, especially when generating full low- and high-resolution disturbed wind data files. Therefore, disabling visualization is recommended when running many FAST.Farm simulations. See Section 4.3 for visualization output file details.

3.2.8 Output

SumPrint [flag] specifies if a summary file is generated. The file is generated if ***SumPrint***=TRUE, with the name *<RootName>.sum*, where *<RootName>* is as defined above. See Section 4.2 for summary file details.

ChkptTime [sec] specifies how frequently checkpoint files are written for a potential restart, but **is currently unused by FAST.Farm.**

TStart [sec] specifies the simulation time at which FAST.Farm will begin writing data in the time-series results output file. Note that output files may not be generated at ***TStart*** seconds if ***TStart*** is not an integer multiple of ***DT_Low***.

OutFileFmt [switch] specifies which type of time-series results output file will be generated. Three options are available, and are the same as those in OpenFAST: 1) generates an ASCII text file [***OutFileFmt=1***]; 2) generates a binary file [***OutFileFmt=2***]; or 3) generates both ASCII text and binary files [***OutFileFmt=3***]. **However, FAST.Farm currently only supports text-based output files. Therefore, *OutFileFmt* must be set to 1**.

TabDelim [flag] specifies how columns in the ASCII text output time-series results are delimited. If ***TabDelim*** = TRUE, the columns are tab-delimited. Otherwise, the columns are delimited with spaces. ***TabDelim*** is not used when ***OutFileFmt*** = 2.

OutFmt [string] specifies the ASCII text-based output file channel format (excluding the time channel). Values printed in the time-series results output file should result in a field that is 10 characters long; "ES10.3E2" is a common setting for ***OutFmt***. The time channel is printed using the "F10.4" format. ***OutFmt*** is not used when ***OutFileFmt*** = 2. See Section 4.4 for details on time-series results files.

FAST.Farm can output wake-related quantities for up to 9 individual turbines, not considering the effects of wake merging, at up to 20 radial nodes and up to 9 downstream distances. These outputs are specified with the 4 following inputs:

- ***NOutRadii*** [integer] specifies the number of radial nodes to be outputted (0 to 20).
- ***OutRadii*** [integer] specifies the node numbers between 0 (at the wake center) and ***NumRadii***-1 (at the outer extent of the radial finite-difference grid). Values are a list of length ***NOutRadii***, separated by any combination of commas, semicolons, spaces, and/or tabs.
- ***NOutDist*** [integer] specifies the number of downstream distances that output is requested for (0 to 9).
- ***OutDist*** [m] specifies the downstream distances (not wake-plane numbers) and each must be greater or equal to zero. Values are a list of length ***NOutDist***, separated by any combination of commas, semicolons, spaces, and/or tabs. The downstream distances are measured normal to the wake planes and **an *OutDist* of zero corresponds to the rotor plane**. Wake output quantities are linearly interpolated between wake planes. Only wake-related quantities for the first 9 turbines can be output and all wakes have the same output radial node numbers and downstream distances. The outputs specified in the ***OutList*** section determine which quantities are actually output at these output radial node numbers and downstream distances.

FAST.Farm can also output ambient wind velocities (not including wakes) and disturbed wind velocities (including wakes) at up to nine points (positions) in the low-resolution wind domain, defined with the following inputs:

- ***NWindVel*** [integer] specifies the number of points where wind will be output (0 to 9).
- ***WindVelX***, ***WindVelY***, and ***WindVelZ*** [m] specifies X, Y, Z and coordinates, respectively, in the **global inertial-frame coordinate system**. Values are lists of length ***NWindVel*** separated by any combination of commas, semicolons, spaces, and/or tabs. The outputs specified in the ***OutList*** section determine which wind velocities are actually output at these points.
- ***OutList*** [quoted strings] controls output quantities generated by FAST.Farm. Enter one or more lines containing quoted strings that in turn contain one or more output parameter names. Separate output parameter names by any combination of commas, semicolons, spaces, and/or tabs. If you prefix a parameter name with a minus sign, "-"; underscore, "_"; or the characters "m" or "M", FAST.Farm will multiply the value for that channel by -1 before writing the data. The output columns are written in the order they are listed in the input file. FAST.Farm allows for the use of multiple lines so that lists can be broken into meaningful groups and so the lines can be shorter. Comments may be entered after the closing quote on any of the lines. Entering a line with the string "END" at the beginning of the line or at the beginning of a quoted string found at the beginning of the line will cause FAST.Farm to quit scanning for more lines of channel names. Wake-related output quantities are generated for the requested output radial node numbers and downstream distances through the ***OutRadii*** and ***OutDist*** lists above. Ambient and disturbed wind velocities are generated for the requested points through the ***WindVelX***, ***WindVelY***, and ***WindVelZ*** lists above. If FAST.Farm encounters an unknown/invalid channel name, it warns the users but will remove the suspect channel from the output file. Please refer to Appendix C for a complete list of possible output parameters.

3.3 Ambient Wind Precursor Files in Visualization Toolkit Format

When using ambient wind generated by a high-fidelity precursor simulation with ***Mod_AmbWind*** = 1, ambient wind data files for both the low- and high-resolution domains must be pre-generated. Each of these ambient wind data files must follow the simple legacy serial VTK file format. A sample VTK-formatted file is given in Appendix B.

FAST.Farm requires that the ambient wind data files be stored in specific subdirectories of the directory specified by ***WindFilePath*** and with specific file names. The low-resolution ambient wind data files must be stored in a subdirectory named *Low* and be named *Amb.t*<n_{low}>*.vtk*, where <n_{low}> is as specified in Section 3.2.3. The high-resolution ambient wind data files must be stored in a subdirectory named *HighT*<n_t> and be named *Amb.t*<n_{high}>*.vtk*, where <n_t> and <n_{high}> are as specified in Section 3.2.3. Subdirectory *HighT*<n_t> should contain the high-resolution ambient wind data corresponding to wind turbine n_t specified in the WIND TURBINES section of the FAST.Farm primary input file—see Section 3.2.5.

Each VTK-formatted input file begins with a file version and identifier, but is not checked by FAST.Farm. The second line is the header information that is for identifying specific cases, but is not used by FAST.Farm. The third line must include the single word ASCII, designating the file format currently supported by FAST.Farm.

The fourth line must contain the words *DATASET STRUCTURED_POINTS*, designating the data set structure currently supported by FAST.Farm. The next three lines set the spatial discretization of the domain. Each domain is stored as a structured 3D grid of wind data points (representing the corners of 3D cells) in the global *X*-*Y*-*Z* inertial-frame coordinate system—as illustrated generically in Figure 8. The number of wind data points in each direction are set by DIMENSIONS followed by three integers separated by white space representing ***NX***, ***NY***, and ***NZ***; the origin of the grid (lowest-most *X*-*Y*-*Z* coordinate) is set by ORIGIN followed by three floating real numbers separated by white space representing ***X0***, ***Y0***, and ***Z0***; and the spatial discretization in each direction are set by SPACING followed by three floating real numbers separated by white space representing ***dX***, ***dY***, and ***dZ***. The total domain size has dimensions (***NX***-1)***dX*** × (***NY***-1)***dY*** × (***NZ***-1)***dZ***.

The eighth line must contain the word *POINT_DATA* followed by an integer number specifying the number of wind data points, i.e., ***NX*** × ***NY*** × ***NZ***. The ninth line must contain the word *VECTORS* followed by the data name (not used by FAST.Farm) and *FLOAT*, which defines the format of the data stored on the grid. Alternatively, the ninth line must contain the word *FIELD* followed by the data name (not used by FAST.Farm) and 1 and the tenth line must contain the array name (not used by FAST.Farm) followed by 3, the number of wind data points, i.e., ***NX*** × ***NY*** × ***NZ***, and *FLOAT*. The remaining ***NX*** × ***NY*** × ***NZ*** lines of the file contain the *X*-*Y*-*Z* components of the ambient wind velocity at each wind data point stored as three floating real numbers separated by white space. The first data point corresponds to the *ORIGIN* and the remaining points involve looping through *X*, then *Y*, and then *Z*. For a ground or wave surface that is not flat and level—e.g., complex terrain or time-varying sea-surface elevation for offshore systems—the wind velocity components at a given wind data point should be written as NaN (not a number)[1] if that point is below the surface (not exposed to the atmosphere).

3.4 Ambient Wind with InflowWind Module Input Files

When using ambient wind through the interface to the *InflowWind* module with ***Mod_AmbWind*** = 2 or 3, the ambient wind is specified within standard *InflowWind* input files described in the OpenFAST documentation. The name of the primary *InflowWind* input file is specified by input parameter ***InflowFile*** in FAST.Farm. Please note that ***InflowFile*** is independent of the *InflowWind* primary input file used by the OpenFAST model of each wind turbine.

The *InflowWind* primary input file is processed the same when running FAST.Farm simulations as it would when running simulations in stand-alone OpenFAST. The only difference is that input parameter ***OutList*** in the *InflowWind* primary input file is ignored and replaced with equivalent output settings in FAST.Farm. All wind file type options and their associated input options are supported by FAST.Farm. Wind file type options are specified with input parameter ***WindType*** in the *InflowWind* primary input file. The available input options include steady wind, uniform time-varying wind, e.g., discrete gusts, and, full-field turbulent wind (in TurbSim, Bladed, and HAWC formats).

[1]FAST.Farm will treat such wind data points as outside the domain, and so, not used in any calculations.

The wind data specified within *InflowWind* must encompass the entire low- and high-resolution domains defined within FAST.Farm for the entire simulation. This is because the ambient wind data specified within *InflowWind* will be interpolated to low- and high-resolution domains for use within FAST.Farm. To ensure this when using full-field turbulent wind data in *InflowWind*, it is recommend that:

- The full-field wind data files be generated periodically so that the wind domain in *InflowWind* effectively extends forever along the wind propagation direction.
- The input parameter ***PropagationDir*** in the *InflowWind* primary input file be set to 0, ±90, or 180 degrees so that the wind propagates along the $\pm X$ or $\pm Y$ axes of the FAST.Farm inertial-frame coordinate system (the exact direction should depend on the orientation of the wind turbines and farm).

When using full-field turbulent wind data in *InflowWind*, it is recommended that the 2D grid where the full-field turbulent wind data are defined be coincident with either the *Y-Z* grid of the high-resolution domain when ***PropogationDir*** = 0 or 180 degrees or the *X-Z* grid of the high-resolution domain when ***PropogationDir*** = ±90 degrees for each wind turbine. This is done to avoid doubly interpolating the wind data (once by FAST.Farm when generating the high-resolution domain and once by OpenFAST when accessing high-resolution wind at turbine analysis nodes).

When using ambient wind through multiple instances of the *InflowWind* module, i.e, when ***Mod_AmbWind*** = 3, only one *InflowWind* input file is specified. However, multiple wind data files are used, each with a different name. Specifically, the file name in the *InflowWind* input file in this case specifically refers only to the directory path of the wind files. The wind file root names are required to be *Low* for the low-resolution domain and *HighT*<n_t> for the high-resolution domain associated with turbine n_t.[2] Setting ***Mod_AmbWind*** to 2 or 3 has no influence when steady inflow is used (***WindType*** = 1). When using full-field turbulent wind data in *InflowWind* with ***Mod_AmbWind*** = 3, it is required that:

- The full-field wind data files be generated periodically. This effectively extends the wind domain forever along the wind propagation direction.
- The input parameter ***PropagationDir*** in the *InflowWind* input file be set to 0 degrees so that the wind propagates along the *X* axis of the FAST.Farm inertial-frame coordinate system.
- The wind data files associated with the high-resolution ambient wind be spatially and temporally synchronized with the low-resolution wind data file. The spatial synchronization must be based on the global *X-Y-Z* offsets of each turbine origin relative to the origin of the inertial frame coordinate system.

3.5 OpenFAST Input Files

In addition to the FAST.Farm-specific input files, the OpenFAST model of each wind turbine also requires input files.

WT_FASTInFile [quoted string] specifies the OpenFAST primary input file for each wind turbine, including path. This is required in addition to the FAST.Farm-specific input files. The OpenFAST primary file, in turn, identifies several module-level input files. These OpenFAST input files are described in the OpenFAST documentation. Identical wind turbines can use the same OpenFAST primary input file, except if the corresponding OpenFAST model makes use of a Bladed-style controller in DLL format or, for offshore wind turbines, if different wave conditions are required for each turbine. If a Bladed-style DLL controller is being used, distinct Bladed-style controller DLLs must be used (each with a unique name). This requires the need for distinct *ServoDyn* primary input files, referencing the appropriate DLL name, and distinct OpenFAST primary input files, each referencing the appropriate *ServoDyn* primary input file name. If different wave conditions are required for each turbine, the distinct wave conditions (e.g., based on unique random wave seeds) for each wind turbine must be set in the *HydroDyn* primary input file and distinct OpenFAST primary input files must be used, each referencing the appropriate *HydroDyn* primary input file name.

Please note that the following input parameters in OpenFAST are interpreted differently when running FAST.Farm simulations than when running simulations in stand-alone OpenFAST.

[2] When HAWC format is used (***WindType*** = 5), *_u*, *_v*, *_w* must be appended to the file names.

AbortLevel in the OpenFAST primary input file is ignored and replaced with the equivalent input set in the FAST.Farm primary input.

TMax in the OpenFAST primary input file is ignored and replaced with the equivalent input set in the FAST.Farm primary input.

CompInflow in the OpenFAST primary input file must be set to 1 (to use the *InflowWind* module).

CompAero in the OpenFAST primary input file must be set to 2 (to use the *AeroDyn v15* module).

WindType and its associated input parameters in the OpenFAST *InflowWind* module primary input file are ignored and replaced with the disturbed wind (including wakes) computed across the high-resolution domain for each wind turbine.

PropogationDir in the OpenFAST *InflowWind* module primary input file is ignored.

PCMode, ***VSContrl***, ***HSSBRMode***, and ***YCMode*** in the OpenFAST *ServoDyn* module primary input file must not be set to 4 because the Simulink/Labview interface is not currently supported by FAST.Farm.

All input parameters across the various OpenFAST input files pertaining to the wind turbine geometry defined relative to the origin of the OpenFAST inertial-frame coordinate system remain unchanged. Turbine origins are defined as the intersection of the undeflected tower centerline and the ground or, for offshore systems, mean sea level. Note, however, this origin ((0,0,0) in the OpenFAST inertial-frame coordinate system) is located at (***WT_X***,***WT_Y***,***WT_Z***) in the FAST.Farm global *X*-*Y*-*Z* inertial-frame coordinate system.

4 Output Files

FAST.Farm produces five types of output files: an echo file, a summary file, visualization output files, a time-series results file, and OpenFAST-related files. The following sections detail the purpose and contents of these files.

4.1 Echo File

If ***Echo*** = TRUE in the FAST.Farm primary input file, the contents of the file will be echoed to a file with the naming convention <*RootName.ech*, where <*RootName*> is as defined in Section 3.3. The echo file is helpful for debugging the primary input file. The contents of an echo file will be truncated if FAST.Farm encounters an error while parsing the primary input file. The error usually corresponds to the line after the last successfully echoed line.

4.2 Summary File

If ***SumPrint*** = TRUE in the FAST.Farm primary input file, FAST.Farm will generate a summary file with the naming convention of <*RootName.sum*>. This file summarizes key information about the wind farm model, including the wind turbine locations and OpenFAST primary input files; wake dynamics finite-difference grid and parameters; time steps of the various model components; and the name units and order of the outputs that have been selected.

4.3 Visualization Output Files

If ***WrDisWind*** = TRUE in the FAST.Farm primary input file, FAST.Farm will generate full 3D low- and high-resolution disturbed wind data output files, i.e., the ambient wind and wake interactions across the wind farm for visualization. The VTK data format and spatial resolutions (number of grid points, origin, and spacing) of these output files matches those of the corresponding low- and high-resolution ambient wind data used by the FAST.Farm simulation. The VTK files are written to a directory named *vtk_ff* where the FAST.Farm primary file is stored. The naming conventions of these output files are <*RootName*>*.Low.Dis.*<n_{low}>*.vtk* and <*RootName*>*.HighT*<n_t>*.Dis.*<n_{high}>*.vtk* for the low- and high-resolution disturbed wind data files, respectively, where <n_t>, <n_{low}>, and <n_{high} are as defined in Section 3.3, but with leading zeros.

Likewise, if ***NOutDisWindXY***, ***NOutDisWindYZ***, or ***NOutDisWindXZ*** are set to be greater than zero in the FAST.Farm primary input file, FAST.Farm will generate low-resolution disturbed wind data (including wakes) output files that are 2D slices of the full low-resolution domain. The 2D slices are parallel to the *X-Y*, *Y-Z*, and/or *X-Z* planes of the global inertial-frame coordinate system, respectively. The VTK files are written to a directory named *vtk_ff* where the FAST.Farm primary file is stored. The naming conventions of these output files are <*RootName*>*.Low.DisXY*<n_{Out}>*.*<n_{low}>*.vtk*, <*RootName*>*.Low.DisYZ*<n_{Out}>*.*<n_{low}>*.vtk*, and <*RootName*>*.Low.DisXZ*<n_{Out}>*.*<n_{low}>*.vtk* for the *X-Y*, *Y-Z*, and *X-Z* slices, respectively, where <n_{Out}> is as defined in Section 3.3, but with leading zeros.

The time step (inverse of the frame rate) of all disturbed wind data files is set by input parameter ***WrDisDT*** in the FAST.Farm primary input file. Note that the full high-resolution disturbed wind data output files are not output at a frame rate of 1/***DT_High***, but are only output every ***WrDisDT*** seconds.

Each visualization output file follows the same VTK format used for the ambient wind data files for the high-fidelity precursor simulations—see Section 3.4 for details on the file format.

Visualizing the ambient wind and wake interactions can be useful for interpreting results and debugging problems. However, FAST.Farm will generate many files per output option when ***WrDisWind*** = TRUE and/or when ***NOutDisWindXY***, ***NOutDisWindYZ***, and/or ***NOutDisWindXZ*** are set greater than zero. This file generation will slow down FAST.Farm and take up a lot of disk space, especially when generating full low- and high-resolution disturbed wind data files. Therefore, disabling visualization is recommended when running many FAST.Farm simulations.

4.4 Time-Series Results File

The FAST.Farm time-series results are written to an ASCII text-based file with the naming convention <*RootName.out*. The results are in table format, where each column is a data channel with the first column always being the simulation time; each row corresponds to a simulation output time step. The data channels are specified in the ***OutList*** section of the *OUTPUT* section of the FAST.Farm primary input file. The column format of the FAST.Farm-generated file is specified using the ***OutFmt*** parameter of the FAST.Farm primary input file.

4.5 OpenFAST Output Files

In addition to the FAST.Farm-generated output files, the OpenFAST model of each wind turbine may also generate output files. The various output files that OpenFAST may generate (at both the driver/glue code and module levels, as specified within the OpenFAST input files) are described in the OpenFAST documentation and include summary (*.sum*) files, time-series results (ASCI *.out* or binary *.outb*) files, visualization (*.vtk*) files, etc. FAST.Farm simulations will generate these same files, but with the the path/rootname changed to *<RootName of WT_FASTInFile>.T*<n_t>.

5 Modeling Guidance

This chapter includes modeling guidance for setting up and running a FAST.Farm simulation. This includes guidance on inflow wind generation; low- and high-resolution grid discretization; parameter selection; super controller use; and solutions for commonly encountered errors.

5.1 FAST.Farm Setup Overview

This section includes a high-level overview of how to set up ambient inflow and FAST.Farm simulations—in particular, the information needed to calculate various parameters, as shown in Figure 10. Note that this schematic only includes information relevant to FAST.Farm simulations. Typically, additional inflow information is required to generate inflow and the OpenFAST models. The specific equations that should be used to compute the input parameters are discussed in Section 5.4. It is highly recommended that the Python notebooks provided in the FAST.Farm tools repository be used when setting up new inflow or a FAST.Farm case. Improperly setting these parameters can lead to common errors and/or excessive interpolation, which should be avoided. Note that this chapter assumes a wind direction of 0°—i.e., ambient wind that propagates along the +*X* axis of the global inertial frame coordinate system.

When generating a FAST.Farm simulation setup and corresponding inflow, planning is important. Improper planning could results in FAST.Farm errors and/or needing to regenerate the inflow. Values that should be known *a priori* are:

- wind turbine rotor diameter (D^{Rotor});
- wind turbine hub height;
- maximum turbine chord length (c_{max});
- maximum turbine natural frequency (f_{max});
- *X*, *Y*, and *Z* locations of all turbines in the wind farm;
- desired mean inflow hub-height wind velocity; and
- mean inflow wind direction.

The values that must be computed using this information are:

- inflow and FAST.Farm domain size (height, width, and length);
- FAST.Farm high- and low-resolution domain origin locations (***S0_High*** and ***S0_Low***, where *S*=*X*, *Y*, or *Z*);
- high- and low-resolution temporal discretization values (***DT_High*** and ***DT_Low***);
- high- and low-resolution spatial discretization values (***DS_High*** and ***DS_Low***);
- number of grid points in the high- and low-resolution domains (***NS_High*** and ***NS_Low***);
- actual mean inflow hub-height wind velocity (V_{hub}); and
- additional wake dynamics properties (***dr***, ***NumRadii***, and ***NumPlanes***).

With this information, inflow generation can begin. Though not required, it is recommended to complete inflow generation before setting up the FAST.Farm simulation. This is because the realized spatial discretization values and/or mean hub height velocity can differ from what is desired. Having the correct values of these parameters leads to less interpolation of the wind data in FAST.Farm simulations, which would otherwise reduce the ambient turbulence.

When setting up the inflow generation, the recommended spatial and temporal discretizations should be used, as discussed in Section 5.3. If using:

- ***Mod_AmbWind*** = 1, a high-fidelity must be generated and all discretization values can be specified as the exact desired value.

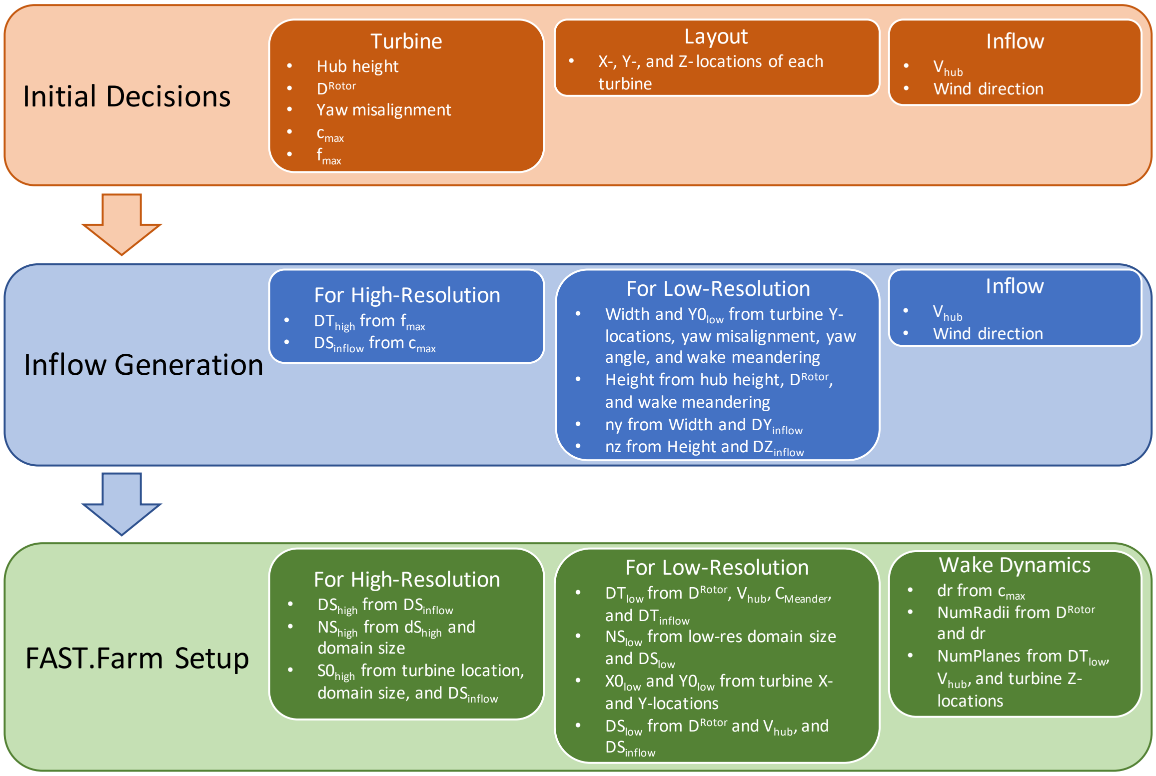


Figure 10. Information flowchart for setting up inflow generation and FAST.Farm simulations. Here, *S*=*X*, *Y*, or *Z*.

- ***Mod_AmbWind*** = 2, a single synthetic inflow (TurbSim or Mann) must be generated using the high-resolution discretization values recommended herein.
- ***Mod_AmbWind*** = 3, multiple synthetic inflows must be generated. In this case, the recommended high-resolution discretizations should be used for all high-resolution inflows generated. For the low-resolution inflow generation, the recommended high-resolution temporal discretization and low-resolution spatial discretization should be used.

If using synthetic inflow (TurbSim or Mann), the inflow streamwise spatial discretization, DX_Inflow, is not specified by the user, but is instead based on Taylor's frozen-turbulence assumption. Because the streamwise discretization of the FAST.Farm domain should be based on the inflow streamwise discretization, the user should compute this value using the inflow time step (***DT_High***) and the advection speed of the synthetic wind data, V_{Advect}. The V_{Advect} may differ from the actual wind speed at hub height, V_{Hub}, as discussed in Section 5.2.3, and should be computed directly from the generated synthetic inflow. Therefore, the exact resulting DX_Inflow will not be known until after the inflow has been generated. Additionally, DX_Inflow will likely be much smaller than the desired values of ***DX_Low*** and ***DX_High***.

When setting up the FAST.Farm simulation itself, many of the values that were used for inflow generation will be used again here to specify the FAST.Farm domain. Note that this domain specification in FAST.Farm is only needed when using synthetic turbulence inflow. The origin of the low-resolution domain (***X0_Low***, ***Y0_Low***, and ***Z0_Low***) should be determined based on:

- the minimum turbine *X*- and *Y*-locations;
- turbine yaw misalignment;
- inflow wind direction; and

- the expected range of wake meandering.

Specifically, ***X0_Low*** must accommodate all turbine locations and, if desired, enough room to analyze the undisturbed inflow upstream of the wind farm. ***Y0_Low*** must accommodate all turbine locations as well as the horizontal wake meandering. When using TurbSim, which cannot generate wind at ground level, ***Z0_Low*** should be close to but above ground level.

The FAST.Farm domain width and height are then computed using:

- the turbine locations;
- the calculated ***Y0_Low*** and ***Z0_Low*** values;
- the horizontal and vertical meandering distance requirements;
- turbine yaw misalignment; and
- the inflow wind direction.

The domain length should be based on the streamwise extent of the wind farm and, if desired, allow enough room to analyze the waked outflow downstream of the wind farm.

The FAST.Farm low-resolution domain lateral and vertical spacial discretization (***DY_Low*** and ***DZ_Low***) and number of grid points (***NY_Low*** and ***NZ_Low***) can then be computed using:

- the domain width and height;
- the lateral and vertical spacing of the generated inflow; and
- DY_Inflow and DZ_Inflow.

The low-resolution temporal discretization (***DT_Low***) should be computed using:

- the turbine diameter;
- inflow hub-height velocity; and
- the inflow temporal discretization.

The streamwise spacing and number of grid points (***DX_Low*** and ***NX_Low***) should also be based on ***DT_Low*** and the mean wind speed.

The final domain parameters to calculate are the locations of the high-resolution domains (***X0_High***, ***Y0_High***, and ***Z0_High***) and the number of grid points required to make up the domains (***NX_High***, ***NY_High***, and ***NZ_High***). These quantities should be determined from:

- ***DS_High*** values;
- turbine locations; and
- the size of the high-resolution domains.

The ***DS_High*** values should be selected based on recommended high-resolution domain discretization criteria, discussed in Section 5.3.

Additional wake dynamics quantities are needed when specifying the FAST.Farm input file, as discussed further in Section 5.4.2. It is recommended to base ***dr*** on $c_{\max}$; ***NumRadii*** on wake diameter and ***dr***; and ***NumPlanes*** on ***DT_Low***, inflow hub-height velocity, and the distance between turbine locations.

A sample turbine layout and domain locations are shown in Figure 11.

5.2 Inflow Wind Generation

This section includes guidelines by which turbulent inflow should be generated for use with FAST.Farm.

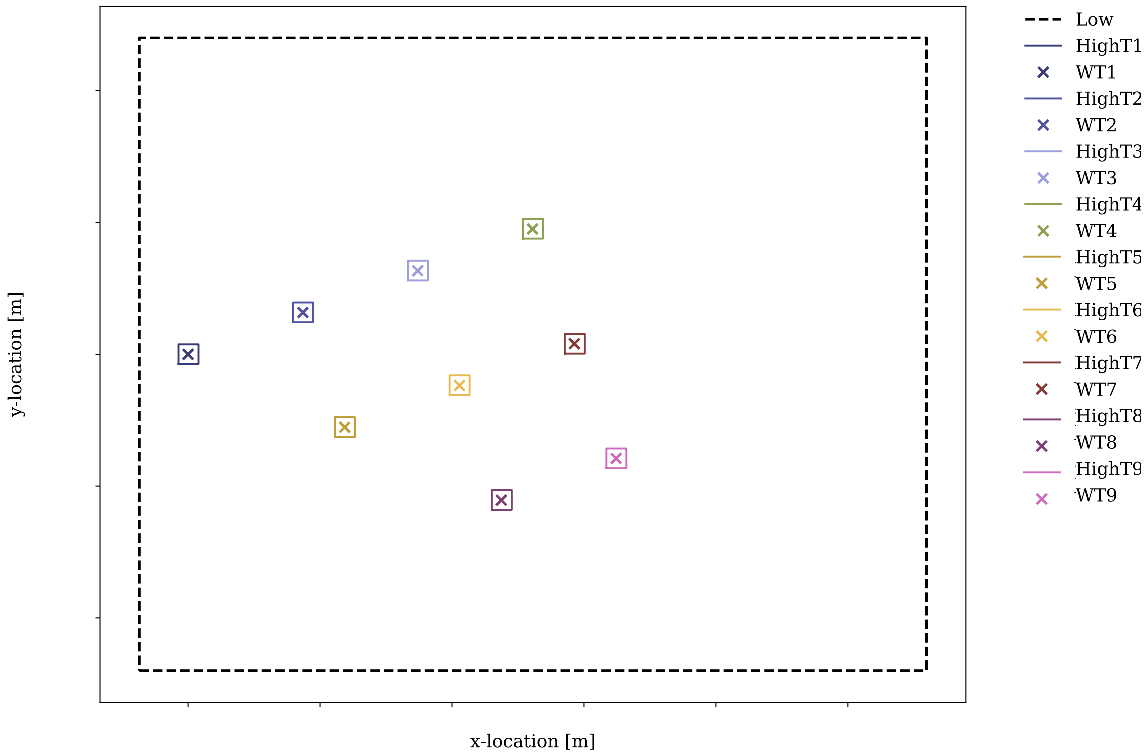


Figure 11. Schematic of example 9-turbine wind farm layout, including low- and high-resolution domains and turbine locations.

5.2.1 High-Fidelity Precursor Ambient Inflow

There are many different methods by which high-fidelity precursor ambient inflow can be generated. This section focuses on generating such inflow using SOWFA.

When using SOWFA to generate FAST.Farm precursor inflow, the *ABLSolver* preprocessor is used. It is important to note the baseline high-fidelity solution is not directly used as inflow for FAST.Farm, but is instead sampled within a specified domain and discretization. This sampling is done through SOWFA and specified in a SOWFA input file. The inflow data are written out in 3D volume VTK-formatted files, as described in Section 3.3. These are large ASCII-formatted files; as such, decreasing the precision to, e.g., 3 digits is recommended. The domain size and low-resolution domain discretization used for SOWFA simulations is much larger than what is required for FAST.Farm simulations. Therefore, sampling files must be set up to generate boundary conditions for use with FAST.Farm, based on FAST.Farm discretization suggestions detailed in Section 5.3. Two sampling files are needed: one for the low-resolution sampling for the farm-scale domain and one for the high-resolution sampling for the turbine-scale domains. Each sampling file defines the spatial and temporal discretization that will be used in the FAST.Farm simulations. The low-resolution domain file defines a single low-resolution domain that will be used for the FAST.Farm simulations; the high-resolution domain file defines each high-resolution domain that will be used for the FAST.Farm simulations. Thus, it is important to know exactly where all turbines will be located in the FAST.Farm simulation before generating the inflow. Note that this FAST.Farm sampling step can be computationally expensive. Therefore, it is recommended that users make sure all inputs are correct before executing SOWFA, including turbine locations and discretization levels.

An example Python notebook is provided in the FAST.Farm tools repository to assist in setting up these files for a given FAST.Farm simulation.

5.2.2 Complex Terrain

Complex terrain, or a time-varying sea-surface elevation for offshore systems, can be modeled in FAST.Farm by providing ambient inflow data that are terrain following, e.g., by modeling the surface boundary condition in an LES precursor. The VTK format used by FAST.Farm is spatially uniform. To accommodate complex terrain or waves with a uniform grid, the wind speed for points below the terrain surface should be set to NaN. Any NaN value will be trapped by FAST.Farm and marked as outside of the domain, and so, unused by calculations within the AWAE module. When the ambient wind inflow is terrain following, the wakes will naturally follow the terrain as well, even though FAST.Farm does not include any explicit models for complex terrain, flow recirculation or separation, or local pressure gradients.

If using a SOWFA inflow precursor, the complex terrain is accounted for in the SOWFA inflow precursor generation, and so, no modification to the *vtk* files is required to account for complex terrain when sampling for a FAST.Farm simulation.

5.2.3 Synthetic Turbulence Ambient Inflow

Synthetically generated turbulent inflow can be used in FAST.Farm to accurately predict turbine response and wake dynamics across different atmospheric conditions. There are several ways to achieve this; any method can be used as long as it produces an output file in a format supported by *InflowWind*. Modeling guidance for TurbSim and the Mann model are discussed next.

TurbSim

When using the NREL tool TurbSim v2, different options are available to drive the synthetic turbulence towards specific desired outcomes, e.g.;

1. standard or user-defined time-averaged wind profile (shear, veer);
2. standard or user-defined velocity spectra in three directions (along the wind, u, and transverse, v and w);
3. standard or user-defined spatial point-to-point coherence; and
4. standard or user-defined component-to-component correlations (Reynolds stresses).

Additionally, TurbSim v2 allows the user to generate turbulent wind that is consistent with user-defined three-component wind time series at one or more points (i.e., constrained wind). These options can be used separately or in some combination (though user-defined spectra and user-defined time series can not be used together). When defined appropriately, all these methods can result in good statistical comparison of turbine response and wake dynamics between FAST.Farm results and a reference data set, e.g., compared with an LES precursor or physically measured inflow. However, attention must be paid when generating these inflows to ensure atmospheric conditions are modeled properly.

In particular, TurbSim generates wind velocities transversely throughout the domain using u-, v-, and w-spatial-coherence models based on a selection of coherence model equations and their associated parameters. These models and parameters can either be specified explicitly or left as *default* values in TurbSim. When the IEC spatial-coherence model is selected, spatial coherence is computed using Eq. (5.1) (Jonkman 2014).

$$Coh_{i,j_K}(f) = exp\left(-a_K\sqrt{\left(\frac{fr}{V_{\text{Advect}}}\right)^2 + (rb_K)^2}\right) \tag{5.1}$$

where V_{Advect} is the average wind speed at the hub height specified in TurbSim, which is also the advection speed in *InflowWind*; Coh_{i,j_K} is the spatial coherence between points i and j for the velocity components $K = u, v, w$; r is the distance between points i and j; a_K is the coherence decrement parameter; and b_K is the coherence offset parameter. It was discovered in Shaler et al. 2019b that the use of the IEC coherence model with default coherence parameters together with the IEC Kaimal spectra results in negligible wake meandering. This is because the default v- and w-coherence parameters in TurbSim are set such that a_K are very large numbers and $b_K = 0$, effectively resulting in no coherence ($Coh_{i,j_K}(f) = 0$) (Jonkman 2014). [3] This lack of meandering is nonphysical and will have a nonphysical impact on the response of downstream turbines. Instead of using the default values, the v- and w-coherence parameters were specified in Shaler et al. 2019b to identically equal the u-coherence parameters specified in the IEC standard, such that: $SCMod2 = SCMod3 = IEC$; $a_K = 12.0$ and $b_K = 0.00035273$ m^{-1}; and $CohExp = 0.0$. (Jonkman 2014). Properly setting spatial coherence parameters for the transverse wind velocity components is necessary to accurately predict wake meandering. It is also important to note that, in TurbSim, the a_K and b_K values must be specified within quotation marks (e.g., `"12.0 0.00035273"`) or, at present, the values are set to 0.

When using TurbSim to generate the full-field turbulent wind data for FAST.Farm, one often wants the TurbSim grid to extend well above the hub height to capture vertical wake meandering due to the *w* component of turbulence. Because TurbSim requires that ***HubHt***$> 0.5*$***GridHeight***, it is often necessary to specify an artificially high ***HubHt*** in TurbSim. To properly set the ***HubHt*** parameter, the following equation is suggested:

$$\boldsymbol{HubHt} = z_{\text{bot}} + \boldsymbol{GridHeight} - 0.5D_{\text{grid}} \tag{5.2}$$

where z_{bot} is the desired bottom vertical location of the grid (just above ground level) and $D_{\text{grid}} = MIN(\boldsymbol{GridWidth}, \boldsymbol{GridHeight})$. Note that the ***HubHt*** parameter is used by TurbSim as the reference height for the wind speed (V_{Advect}) used to define the wind-speed standard deviations and spatial coherence in the IEC turbulence models, as well as the advection speed (in *InflowWind*) for all models. Thus, the resulting wind-speed standard deviations and spatial coherence in the IEC turbulence models will not be what is expected without explicit consideration of the difference in wind speeds between the ***HubHt*** used by TurbSim and the actual turbine hub height. The advection speed (in *InflowWind*) will likely also be faster than it would be when the actual hub height speed is used. A separate reference height (***RefHt***) is specified in TurbSim, which is the height at which, e.g., the reference wind speed is enforced. This value is also used to properly set the power law velocity profile. Future work is needed to decouple the ***HubHt*** parameter from the TurbSim grid generation.

It is generally recommended that the full-field wind data files be generated periodically. This effectively extends the wind domain forever along the wind propagation direction.

When using ambient wind through multiple instances of the *InflowWind* module, i.e, when ***Mod_AmbWind*** = 3, only one *InflowWind* input file is specified. However, multiple wind data files are used, each with a different name. Specifically, the file name in the *InflowWind* input file in this case refers only to the directory path of the wind files. The wind file root names are required to be *Low* for the low-resolution domain and *HighT*<n_t> for the high-resolution domain associated with turbine n_{t}. [4] When steady inflow in *InflowWind* is used (***WindType*** = 1), setting ***Mod_AmbWind*** to 2 or 3 produces identical results. When using full-field turbulent wind data in *InflowWind* with ***Mod_AmbWind*** = 3, it is required that:

- The full-field wind data files be generated periodically. This effectively extends the wind domain forever along the wind propagation direction.
- The input parameter ***PropagationDir*** in the *InflowWind* input file be set to 0 degrees so that the wind propagates along the *X* axis of the FAST.Farm inertial-frame coordinate system.
- The wind data files associated with the high-resolution ambient wind be spatially and temporally synchronized with the low-resolution wind data file. The spatial synchronization must be based on the global *X-Y-Z* offsets of each turbine origin relative to the origin of the inertial frame coordinate system. For each wind turbine, the velocity time series at the turbine location should be extracted from the low-resolution TurbSim domain. To account for turbine downstream distance, each time series should then be offset in time based on the freestream velocity and turbine location. This time series should then be used to generate the high-resolution TurbSim inflow for each turbine. The TurbSim user's manual contains details on how to generate a TurbSim inflow using a specified time series Jonkman 2014.

[3] TurbSim effectively neglects the spatial v- and w-coherence in the default IEC case because these are not prescribed by the IEC design standards.

[4] When HAWC format is used (***WindType*** = 5), *_u*, *_v*, *_w* must be appended to the file names.

Mann Model

When generating stochastic turbulence with the Mann model, 11 user-defined inputs are required: ***prefix***, ***alpha_-epsilon***, ***L***, ***gamma***, ***seed***, ***nx***, ***ny***, ***nz***, ***dx***, ***dy***, and ***dz***. The parameters that should be selected in conjunction with FAST.Farm parameters are discussed here.

dx, ***dy***, and ***dz***—These parameters should be selected based on the high-resolution spatial discretization recommendations discussed below in Section 5.3.

nx—This value is required to be a power of 2. To ensure no repetition of the turbulence box for the duration of the simulation, the following equation is recommended:

$$\boldsymbol{nx} = 2^{CEILING\left[log_2\left(\frac{V_{\text{Advect}}\boldsymbol{T\_Max}}{\boldsymbol{dx}}\right)\right]} \tag{5.3}$$

where V_{Advect} is the advection speed of the Mann box and $CEILING[x]$ rounds x to the next highest integer. This equation ensures that the turbulence box will not repeat during the simulation and also that the power of two criteria is satisfied.

ny and ***nz***—These values are also required to be powers of 2. With this requirement in mind, these values should be selected to ensure the entire desired domain width (Y) and height (Z) are captured, as discussed below in Section 5.4.1.

The *InflowWind* input file has a specific section for using a Mann turbulence box. This section requires the input of ***nx***, ***ny***, ***nz***, ***dx***, ***dy***, ***dz***, ***RefHt***, and ***URef***. These values should be specified exactly as those used to generate the inflow. Note that ***dx***, ***dy***, and ***dz*** specified in *InflowWind* should be the same as ***dX_High***, ***dY_High***, and ***dZ_High*** in FAST.Farm, respectively. ***RefHt*** should be defined as follows:

$$\boldsymbol{RefHt} = 0.5\boldsymbol{dz}(\boldsymbol{nz}-1) + z_{\text{bot}} \tag{5.4}$$

URef is the mean wind speed at the reference height, and dictates the advection speed of the Mann box, identified here as V_{Advect}.

When using a Mann box, it is important to know that **the x-axis direction is opposite the convention used by *InflowWind*. Although the interpretation in *InflowWind* (including OpenFAST and FAST.Farm) is consistent with how Mann boxes are used in other aeroelastic software, the interpretation is nonphysical**. If desired, the user can adjust the FAST.Farm source code to read the x-axis in reverse. Correcting this error universally across all aeroelastic software that use Mann boxes is needed future work.

5.3 Low- and High-Resolution Domain Discretization

Spatial and temporal discretization can affect wake meandering, turbine structural response, and resulting wake and load calculations. This section summarizes recommendations for discretization values in terms of geometry and wind speed that will ensure a converged solution, while maximizing computational efficiency. For details on how these recommendations were formed, see Shaler et al. 2019a. Though developed for FAST.Farm use, these guidelines are likely applicable to any DWM-type model or aeroelastic analysis.

5.3.1 Low-Resolution Domain

The low-resolution domain in FAST.Farm is primarily responsible for wake meandering and merging. As such, convergence was assessed by comparing trends in standard deviation of horizontal and vertical meandering wake center positions for the wakes behind each turbine at various distances downstream. It was found that the mean horizontal and vertical wake trajectories have negligible dependence of ***DT_Low*** or ***DS_Low***. The following equation can be used to ensure convergence of wake meandering in the low-resolution domain:

$$\boldsymbol{DT\_Low} \leq \frac{C_{\text{Meander}}D^{\text{Wake}}}{10V_{\text{Hub}}} \tag{5.5}$$

This equation is based on the low-pass cutoff frequency for wake meandering $\left(\frac{V_{\text{Hub}}}{C_{\text{Meander}} D^{\text{Wake}}}\right)$ from Larsen et al. 2008 (in which $C_{\text{Meander}} = 2$, but C_{Meander} defaults to 1.9 in FAST.Farm) and effectively specifies that the highest frequency of wake meandering should be resolved by at least 10 time steps. Note that D^{Wake} can be approximated as D^{Rotor} in this calculation.

Spatial discretization convergence was assessed in the same manner as temporal discretization. Minimal sensitivity to spatial discretization was found for the low-resolution domain in the range of spatial discretizations considered. Nonetheless, the following equation is recommended for identifying the maximum suggested ***DS_Low***, where S refers to X, Y, or Z and the denominator has the units [m/s]:

$$\boldsymbol{DS\_Low} \leq \frac{C_{\text{Meander}} D^{\text{Wake}} V_{\text{Hub}}}{150 m/s} = \frac{\boldsymbol{DT\_Low} V_{\text{Hub}}^2}{15 m/s} \tag{5.6}$$

For all synthetic turbulence methods, it is recommended that ***DX_Low***$= V_{\text{Advect}}$***DT_Low*** to avoid interpolating in X-direction. Note the use of the advection speed, V_{Advect}, to calculate ***DX_Low***, rather than the actual hub-height wind speed, V_{Hub}. Additionally, ***X0_Low*** should be an integer multiple of ***DX_Low***.

5.3.2 High-Resolution Domain

The high-resolution wind domain in FAST.Farm is primarily responsible for ambient and waked inflow local to a turbine. As such, convergence was assessed by comparing trends in mean and standard deviation of turbine structural motions and loads for each turbine.

Required discretization levels vary depending on the quantity of interest. Thus, it is important to decide what structural components will be considered when selecting a high-resolution discretization level. Most notably, tower-base moments are the most sensitive to ***DT_High***, whereas generator power and blade deflections and moments show little dependence on this value. To capture the full structural response, ***DT_High*** should be selected based on the highest frequencies influencing the structural excitation, including rotational sampling of turbulence and response, i.e., natural frequencies, of the pertinent structural components, f_{max} (in Hz), as in Equation 5.7, where the factor of 2 is taken from the Nyquist sampling theorem. This is a frequently used rule of thumb in wind turbine aeroelastic analysis under excitation from turbulent inflow.

$$\boldsymbol{DT\_High} \leq \frac{1}{2 f_{\text{max}}} \tag{5.7}$$

The required ***DS_High*** approximately corresponds to the maximum blade chord length of the turbine, c_{max}, as in Equation 5.8. Selecting a ***DS_High*** equivalent to this value has long been a rule-of-thumb in wind turbine aeroelastic analysis under excitation from turbulent inflow.

$$\boldsymbol{DS\_High} \leq c_{\text{max}} \tag{5.8}$$

5.4 Parameter Selection

Setting up a FAST.Farm simulation can involve specifying a large number of parameters, especially if the *InflowWind* module is used for the ambient wind. This section summarizes best practices for selecting some of these parameters. References are made to desired versus realized values. The discrepancies between these values are discussed in Section 5.1.

5.4.1 InflowWind Domain Parameters

Care must be taken when setting up a FAST.Farm simulation using the *InflowWind* ambient wind inflow option. It is highly recommended that the distributed Python notebooks be used when setting up a new case. Improperly setting these parameters can lead to common errors and/or excessive interpolation, which should be avoided. The methods and rules of thumb that are used in those Python notebooks are also discussed here.

Low-Resolution Domain

NX_Low, ***NY_Low***, ***NZ_Low***—These quantities should be based on ***DS_Low*** and the desired domain size (*Sdist_Low*), where *S*=*X*, *Y* or *Z*. This integer quantity should be computed as:

$$\boldsymbol{NS\_Low} = CEILING\left(\frac{Sdist\_Low}{\boldsymbol{DS\_Low}}\right) + 1 \tag{5.9}$$

X0_Low—This quantity must be less than the *X* location of the furthest upstream turbine. It is recommended to set this value further upstream to allow for analysis of the ambient inflow. If using a Mann box, this value should be 0.

Y0_Low—This quantity must be less than the lowest *Y* location of any turbine (***WT_Y_min***). Additional clearance is required to accommodate wake meandering, wake deflection, and spatial averaging used in the *AWAE* module. This value may be computed as:

$$\boldsymbol{Y0\_Low} \leq \boldsymbol{WT\_Y\_min} - 3D^{\text{Rotor}} \tag{5.10}$$

Additional clearance should be allowed for appreciable wake meandering and/or yaw. For ***Mod_AmbWind*** = 2, the synthetic inflow data are centered around Y=0. Because of this, ***Y0_Low*** should equal -*Ydist_Low*/2. This is the same for the low-resolution domain with ***Mod_AmbWind*** = 3.

Z0_Low—It is recommended that this value be set close to but above ground level. When using TurbSim, this value can not be at or below ground level because TurbSim cannot generate wind at these locations.

DX_Low, ***DY_Low***, ***DZ_Low***—Desired spatial values are not discussed here, as they are covered in detail in Section 5.3. However, the actual quantities used might differ from the desired values when using synthetic inflow, as discussed in Section 5.2.3. To determine the actual quantity, the following equation is suggested when using synthetic inflow:

$$\boldsymbol{DS\_Low} = FLOOR\left(\frac{DS\_Low\_Desired}{\boldsymbol{DS\_High}}\right) * \boldsymbol{DS\_High} \tag{5.11}$$

Use of this equation is the best way to ensure that ***DS_Low*** will be a multiple integer of ***DS_High***, reducing interpolation smoothing.

High-Resolution Domain

Xdist_High, *Ydist_High*, *Zdist_High*—Though not direct inputs, these lengths, widths, and heights of the high-resolution domains should be selected based on the size and location of the turbines. The following values are recommended:

$$Xdist\_High = Ydist\_High = Zdist\_High \geq 1.1\ D^{\text{Rotor}} \tag{5.12}$$

If tower aerodynamic loads are desired, the high-resolution domain should span the entire tower and rotor:

$$Zdist\_High = \boldsymbol{HubHt} + \frac{1.1\ D^{\text{Rotor}}}{2} \tag{5.13}$$

These parameters might need to be increased to account for large structural motion, such as for floating offshore wind applications.

NX_High, ***NY_High***, ***NZ_High***—These quantities should be based on ***DS_High*** and the desired domain size (*Sdist_High*), where *S*=*X*, *Y*, or *Z*. This integer quantity should be computed as:

$$\boldsymbol{NS\_High} = CEILING\left(\frac{Sdist\_High}{\boldsymbol{DS\_High}}\right) + 1 \tag{5.14}$$

X0_High, ***Y0_High***, ***Z0_High***—These quantities are set for each turbine. They should be based on turbine location and set so that the turbine is contained inside the high-resolution domain. It is recommended that ***X0_High*** and ***Y0_High*** are set approximately $1.1D^{\text{Rotor}}/2$ lower than the turbine location. For the high-resolution domains with ***Mod_AmbWind*** = 3, the synthetic inflow data are centered around each turbine, based on ***WT_X/Y/Z***.

DX_High, ***DY_High***, ***DZ_High***—Desired spatial values are not discussed here, as they are covered in detail in Section 5.3.

5.4.2 Wake Dynamics Parameters

Wake dynamics parameters define the axisymmetric finite-difference grid used for each wake plane. These planes are defined by the following parameters:

- ***dr***—This value should be set so that FAST.Farm sufficiently resolves the wake deficit within each plane. The following value is suggested:

$$\boldsymbol{dr} \leq c_{\max} \tag{5.15}$$

- ***NumRadii***—To ensure the wake deficits are accurately computed by FAST.Farm, ***NumRadii*** should be set so that the diameter of each wake plane, 2(***NumRadii***-1)***dr***, is large relative to the rotor diameter. The following value is suggested:

$$\boldsymbol{NumRadii} \geq \frac{3D^{Rotor}}{2\,\boldsymbol{dr}} + 1 \tag{5.16}$$

- ***NumPlanes***—To ensure the wake deficits are accurately captured by FAST.Farm, ***NumPlanes*** should be set so that the wake planes propagate a sufficient distance downstream, preferably until the wake deficit decays away (x_{dist}), with typical values between $10-20\times D^{Rotor}$. The following value is suggested:

$$\boldsymbol{NumPlanes} \geq \frac{x_{\text{dist}}}{\boldsymbol{DT\_Low}\overline{V}} \tag{5.17}$$

  where $\overline{V}$ is the average convection speed of the wake, which can be approximated as

$$\overline{V} = V_{\text{Hub}}\left(1-\frac{\overline{a}}{2}\right) \tag{5.18}$$

  where $\overline{a}$ is the time- and spatial-temporal-average of the axial induction at the rotor disk. $\overline{a}$ is expected to be around $1/3$ below rated wind speed (for optimal aerodynamic efficiency) and decreases above rated wind speed to near zero before the cut-out wind speed.

Note that because new wake planes are added each time step as the simulation begins, increasing ***NumPlanes*** will also increase the initial transient time of the simulation. The start-up transient time is estimated by Equation 5.19.

$$t_{\text{startup}} = \boldsymbol{DT\_Low}(\boldsymbol{NumPlanes}-2) \tag{5.19}$$

- ***Mod_WakeDiam***—A value of ***1*** is recommended. For further details on the options for this parameter, see Equation 6.26.
- ***Mod_Meander***—A value of ***3*** is recommended. For further details on the options for this parameter, see Equation 6.35.

The remaining 20 inputs are user-specified calibration parameters and options that influence the wake-dynamics calculations. The parameters may depend, e.g., on turbine operation or atmospheric conditions that can be calibrated to better match experimental data or by using an HFM benchmark. Default values have been derived for each calibrated parameter based on SOWFA simulations for the NREL 5MW turbine (Doubrawa et al. 2018), but these can be overwritten by the user.

5.5 Super Controller

When ***UseSC*** is set to TRUE, the super controller is enabled. The super controller code must be compiled as a dynamic library file—a *.dll* file in Windows® or a *.so* file in Linux or Mac OS. This super controller dynamic library is essentially identical to the super controller available in SOWFA. The super controller is used in conjunction with individual wind turbine controllers defined in the style of the DISCON dynamic library of the DNV GL's Bladed wind turbine software package, with minor modification.

Table 1. Arguments for Each Procedure of the Super Controller Dynamic Library

Procedure	Inputs	Outputs	Comments
sc_init	• nTurbines	• nInpGlobal • NumCtrl2SC • NumParamGlobal • NumParamTurbine • NumStatesGlobal • NumStatesTurbine • NumSC2CtrlGlob • NumSC2Ctrl • errStat • errMsg	• Set numbers of inputs, outputs, states, and parameters • nInpGlobal must currently be set to zero in FAST.Farm
sc_ getinitData	• nTurbines • NumParamGlobal • NumParamTurbine • NumSC2CtrlGlob • NumSC2Ctrl • NumStatesGlobal • NumStatesTurbine	• ParamGlobal(1:NumParamGlobal) • ParamTurbine (1:NumParamTurbine*nTurbines) • from_SCglob(1:NumSC2CtrlGlob) • from_SC(1:NumSC2Ctrl*nTurbines) • StatesGlob(1:NumStatesGlobal) • StatesTurbine (1:NumStatesTurbine*nTurbines) • errStat • errMsg	• Set parameters • Initialize states at time zero • Initial outputs are not currently used by FAST.Farm
sc_ calcOutputs	• nTurbines • NumParamGlobal • ParamGlobal(1:NumParamGlobal) • NumParamTurbine • ParamTurbine (1:NumParamTurbine*nTurbines) • nInpGlobal • to_SCglob(1:nInpGlobal) • NumCtrl2SC • to_SC(1:NumCtrl2SC*nTurbines) • NumStatesGlobal • StatesGlob(1:NumStatesGlobal) • NumStatesTurbine • StatesTurbine (1:NumStatesTurbine*nTurbines) • NumSC2CtrlGlob • NumSC2Ctrl	• from_SCglob(1:NumSC2CtrlGlob) • from_SC(1:NumSC2Ctrl*nTurbines) • errStat • errMsg	• Calculate outputs at the current time step • nInpGlobal is currently zero in FAST.Farm • to_SCglob is currently null in FAST.Farm
sc_ updateStates	• nTurbines • NumParamGlobal • ParamGlobal(1:NumParamGlobal) • NumParamTurbine • ParamTurbine (1:NumParamTurbine*nTurbines) • nInpGlobal • to_SCglob(1:nInpGlobal) • NumCtrl2SC • to_SC(1:NumCtrl2SC*nTurbines) • NumStatesGlobal • NumStatesTurbine	• StatesGlob(1:NumStatesGlobal) • StatesTurbine (1:NumStatesTurbine*nTurbines) • errStat • errMsg	• Update states from one time step to the next • nInpGlobal Is currently zero in FAST.Farm • to_SCglob Is currently null in FAST.Farm
sc_end		• errStat • errMsg	• Release memory • Close files

The inputs to the super controller are commands or measurements from individual turbine controllers.[5] The outputs of super controller module are the global controller commands and individual turbine controller commands.

The super controller dynamic library must be compiled with five procedures, whose arguments are outlined in Table 1.

To interact with the super controller, the individual turbine controllers within each instance of OpenFAST must also be compiled as a dynamic library. The single procedure, `DISCON`, is unchanged from the standard `DISCON` interface for the Bladed wind turbine software package, as defined by DNV GL, but with three extra arguments, as outlined in Table 2.

Note that at time zero, the super controller output calculation (`sc_calcOutputs`) is called before the call to the individual turbine controllers (`DISCON`). So, the initial outputs from the super controller (`from_SC`, `from_SCglob`) are sent as inputs to the individual turbine controllers, but the initial inputs to the super controller from the individual turbine controller outputs (`to_SC`) at time zero are always zero. At subsequent time steps, the individual

[5] The super controller also has as input a placeholder for future global (e.g., wind) measurements in addition to commands or measurements from the individual turbine controllers. But the global inputs are currently null.

Table 2. Arguments of the `DISCON` Procedure for Individual Turbine Controller Dynamic Library, Updated for the Super Controller

Procedure	Inputs	Outputs	Comments
`DISCON`	• `avrSWAP(*)` • `from_SCglob (1:NumSC2CtrlGlob)` • `from_SC (1:NumSC2Ctrl)` • `accInFILE` • `avcOUTNAME`	• `avrSWAP(*)` • `to_SC(1:NumCtrl2SC)` • `aviFAIL` • `avcMSG`	• New inputs: `from_SCglob` and `from_SC` • New output: `to_SC`

turbine controllers (`DISCON`) are called before the output calculation of the super controller (`sc_calcOutputs`). As a result, at each time step other than time zero, the outputs from the super controller (`from_SC`, `from_SCglob`) are extrapolated in time based on past values within *OF* before being sent as input to the individual turbine controllers. Thus, care should be taken to ensure that the outputs from the super controller (`from_SC`, `from_SCglob`) vary smoothly over time (without steps). See Figure 13 for more information.

5.6 Commonly Encountered Errors

This section covers errors that have been commonly encountered by users during the development, verification, and use of FAST.Farm. Submit any additional errors or questions to the NWTC forum.

5.6.1 *InflowWind* Errors

InflowWind errors tend to be related to improperly setting the high- or low-resolution domain sizes. Two such common errors are detailed here.

Turbine Leaving the Domain

The following error is commonly encountered:

```
T<n_t>:<routine name>:FAST_Solution0:CalcOutputs_And_SolveForInputs:
SolveOption2:InflowWind_CalcOutput:CalcOutput:IfW_4Dext_CalcOutput
[position=(-1.8753, 0, 32.183) in wind-file coordinates]:Interp4D:Outside
the grid bounds.
```

This error occurs when a turbine leaves the specified high-resolution domain. This typically happens through improper domain specification or large blade deflections/structural motions. Note that coordinates in this error are in the local frame of reference of the turbine and are case dependent.

If the cause is improper domain specification, the error will trigger in the initialization stage of the simulation (*<routine name>=FARM_InitialCO:FWrap_t0*). In this case, a review of the primary FAST.Farm input file is suggested. In particular, the values of ***NX_High***, ***NY_High***, ***NZ_High***, ***X0_High***, ***Y0_High***, ***Z0_High***, ***dX_High***, ***dY_High***, and ***dZ_High***, as these parameters define the size and location of the high-resolution domain. Note that the error specifies which turbine (T<n_t>) the error has occurred for, which will aid in debugging where the error is.

If the cause is large blade deflection or structural motion, the error will trigger at some point during the simulation (*<routine name>=FARM_UpdateStates:FWrap_t0*). In this case, increasing the overall size of the high-resolution domain could alleviate this problem. However, the user should first confirm that such large deflections/motions are expected and realistic and not due to a turbine modeling error.

Undefined Location

The following error is commonly encountered:

```
Farm_Initialize:InflowWind_CalcOutput:CalcOutput:IfW_TSFFWind_CalcOutput
[position=(5, 565, 5) in wind-file coordinates]:  FF wind array boundaries
violated:  Grid too small in Y direction.  Y=565; Y boundaries =
[-555, 555]
```

This error occurs when FAST.Farm tries to access a point in the low-resolution domain that is not contained in the ambient wind file. Note that coordinates in this error are in the global frame of reference and are case dependent. For this error, a review of the primary FAST.Farm input file is suggested. In particular, the values of ***NX_Low***, ***NY_-Low***, ***NZ_Low***, ***X0_Low***, ***Y0_Low***, ***Z0_Low***, ***dX_Low***, ***dY_Low***, and ***dZ_Low***, as these parameters define the size and location of the low-resolution domain. The error specifies along which axis the error has occurred, aiding in debugging.

6 FAST.Farm Theory

FAST.Farm is a multiphysics engineering tool for predicting the performance and loads of wind turbines within a wind farm. FAST.Farm uses OpenFAST to solve the aero-hydro-servo-elastic dynamics of each individual turbine, but considers additional physics for wind-farm-wide ambient wind in the atmospheric boundary layer; a wind-farm super controller; and wake deficits, advection, deflection, meandering, and merging. FAST.Farm is based on the principles of the DWM model—including passive tracer modeling of wake meandering—but addresses many of the limitations of previous DWM implementations.

6.1 Dynamic Wake Meandering Principles and Limitations Addressed

The main idea behind the DWM model is to capture key wake features pertinent to accurate prediction of wind farm power performance and wind turbine loads, including the wake-deficit evolution (important for performance) and the wake meandering and wake-added turbulence (important for loads). Although fundamental laws of physics are applied, appropriate simplifications have been made to minimize the computational expense, and HFM solutions are used to inform and calibrate the submodels. In the DWM model, the wake-flow processes are treated via the "splitting of scales," in which small turbulent eddies (less than two diameters) affect wake-deficit evolution and large turbulent eddies (greater than two diameters) affect wake meandering.

The presence of thrust from the wind turbine rotor causes the wind speed to decrease and the pressure to increase just upwind of the rotor. In the near-wake region just downwind of the rotor—illustrated in Figure 12—coherent vortices break down, the pressure recovers to free stream, the wind speed decreases further, and the wake expands radially. In the far-wake region further downwind, the wake deficit is approximately Gaussian and recovers to free stream due to the turbulent transfer of momentum into the wake from the ambient wind across the wake shear layer. This flow-speed reduction and gradual recovery to free stream is known as the wake-deficit evolution. In most DWM implementations, the wake-deficit evolution is modeled via the thin shear-layer approximation of the Reynolds-averaged Navier-Stokes equations under quasi-steady-state conditions in axisymmetric coordinates—illustrated in Figure 4. The turbulence closure is captured by using an eddy-viscosity formulation, dependent on small turbulent eddies. This wake-deficit evolution solution is only valid in the far wake. This far wake is most important for wind farm analysis because wind turbines are not typically spaced closely. However, because the wake-deficit evolution solution begins at the rotor, a near-wake correction is applied at the inlet boundary condition to improve the accuracy of the far-wake solution.

Wake meandering is the large-scale movement of the wake deficit transported by large turbulent eddies. This wake-meandering process is treated pragmatically in DWM (Larsen et al. 2008) by modeling the meandering as a passive tracer, which transfers the wake deficit transversely (horizontally and vertically) to a moving frame of reference (MFoR)—as illustrated in Figure 1—based on the ambient wind (including large turbulent eddies) spatially averaged across planes of the wake.

Wake-added turbulence is the additional small-scale turbulence generated from the turbulent mixing in the wake. It is often modeled in DWM by scaling up the background (undisturbed) turbulence.

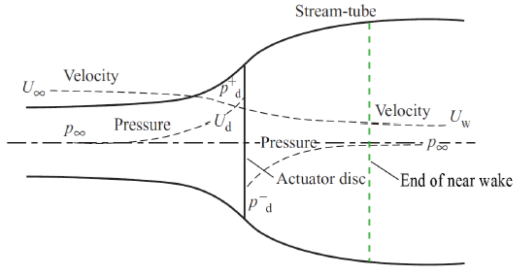


Figure 12. Near-wake region.

Several variations of DWM have been implemented, e.g., by the Technical University of Denmark (Madsen et al. 2010, 2016) and the University of Massachusetts (Hao et al. 2014; Churchfield et al. 2015; Hao 2016). Although the exact limitations of existing DWM implementations depend on the implementation, specific limitations that are addressed in developing FAST.Farm are summarized in Table 3 and are discussed where appropriate in the next section.

6.2 FAST.Farm Theory Basis

FAST.Farm is a nonlinear time-domain multiphysics engineering tool composed of multiple submodels, each representing different physics domains of the wind farm. FAST.Farm is implemented as open-source software that follows the programming requirements of the FAST modularization framework (Jonkman 2013), whereby the submodels are implemented as modules interconnected through a driver code. The submodel hierarchy of FAST.Farm is illustrated in Figure 2. Wake advection, deflection, and meandering; near-wake correction; and wake-deficit increment are submodels of the wake-dynamics (*WD*) model, implemented in a single module. Ambient wind and wake merging are submodels of the ambient wind and array effects (*AWAE*) model, implemented in a single module. Combined with the super controller (*SC*) and OpenFAST (*OF*) modules, FAST.Farm has four modules and one driver. There are multiple instances of the *OF* and *WD* modules—one instance for each wind turbine/rotor. Each submodel/module is described in the subsections below.

FAST.Farm can be compiled and run in serial or parallel mode. Parallelization has been implemented in FAST.Farm through OpenMP, which allows FAST.Farm to take advantage of multicore computers by dividing computational tasks among the cores/threads within a node (but not between nodes) to speed up a single simulation. This process is illustrated in Figure 13 for a node where the number of threads (N_{Th}) is greater than the number of wind turbines (N_t). There is one instance of the *AWAE* and *SC* modules and N_t instances of the *OF* and *WD* modules. The initialization, update states, calculate output, and end calls to each module are shown. The output calculation of *AWAE* is parallelized across all threads. During time marching, each instance of *OF* is solved in parallel while the ambient wind data are read by *AWAE*.

The size of the wind farm and number of wind turbines is limited only by the available RAM. In parallel mode, each instance of the OpenFAST submodel can be run in parallel on separate threads. At the same time, the ambient wind within the *AWAE* module is being read into memory on another thread. Thus, the fastest simulations require at least one more core than the number of wind turbines in the wind farm. Furthermore, the output calculations within the *AWAE* module are parallelized into separate threads. To support the modeling of large wind farms, single simulations involving memory parallelization and parallelization between nodes of a multinode HPC through MPI is likely required. MPI has not yet been implemented within FAST.Farm. However, a multinode HPC can be used to run multiple serial or parallelized simulations in parallel (in batch mode) on separate nodes. In serial mode, multiple serial simulations can be run in parallel (in batch mode) on separate cores and/or nodes.

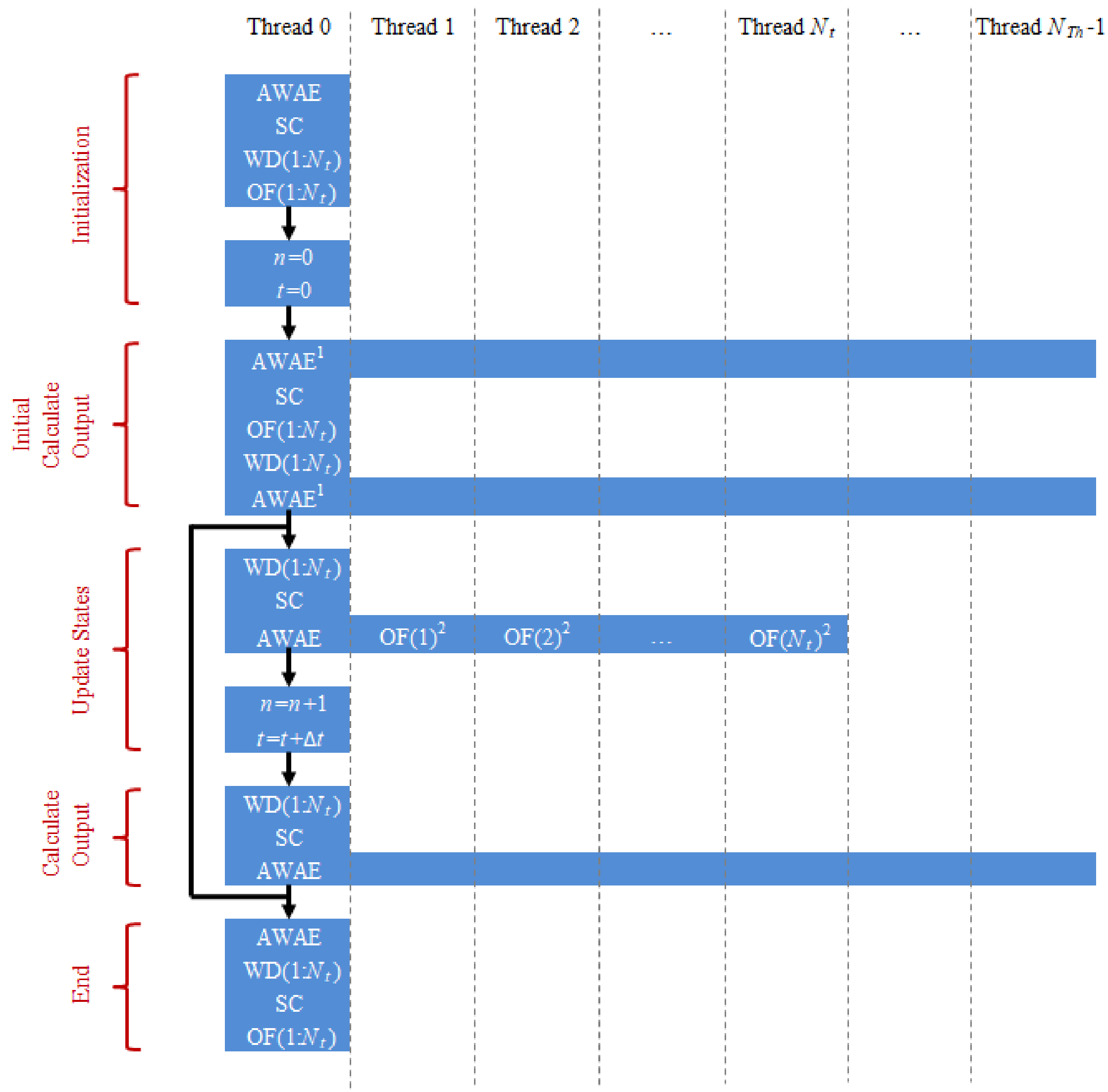
6.2.1 FAST.Farm Driver

The FAST.Farm driver, also known as the "glue code," is the code that couples individual modules together and drives the overall time-domain solution forward. Additionally, the FAST.Farm driver reads an input file of simulation parameters, checks the validity of these parameters, initializes the modules, writes results to a file, and releases memory at the end of the simulation.

To simplify the coupling algorithm in the FAST.Farm driver and ensure computational efficiency, all module states (x^d), inputs (u^d), outputs (y^d), and functions (X^d for state updates and Y^d for outputs) in FAST.Farm are expressed in discrete time, $t = n\Delta t$, where t is time, n is the discrete-time-step counter, and Δt is the user-specified discrete time step (increment). Thus, the most general form of a module in FAST.Farm is simpler than that permitted by the FAST

Table 3. Dynamic Wake Meandering Limitations Addressed by FAST.Farm

Limitation	Solution/Innovation
• Ambient wind is solved per individual rotor and generated synthetically based on the Taylor's frozen-turbulence assumption; not coherent across the wind farm or based on mesoscale conditions or local terrain.	• Optionally compute ambient wind-farm-wide from a high-fidelity precursor.
• No treatment of a wind farm super controller.	• Optional inclusion of a wind farm super controller.
• Wake advects at mean ambient wind speed, not accelerating from near wake to far wake or affected by local flow conditions.	• Wake advects based on the local spatially averaged ambient wind speed and wake deficit.
• Wake deficit is not distorted by inflow skew (i.e., when looking downwind, the wake looks circular, not elliptical). • Wake centerline is not deflected by inflow skew.	• Wake deficit solved in planes parallel to rotor disk. • Wake centerline deflected based on inflow skew.
• Wake deficit and centerline based only on mean conditions, not updated for transients in inflow, turbine control, or wind turbine motion (the latter is especially important for floating offshore wind turbines).	• Wake deficit and centerline updated based on low-pass-filtered inflow, wind turbine control, and wind turbine motion.
• Individual wind turbine and wake dynamics solved individually or serially, not considering two-way wake-merging interactions. • Wake impingement based only on the strongest wake deficit—not considering cumulative effects from multiple upwind wind turbines—and/or the wake impingement approach is treated differently below and above rated wind speed (i.e., a discrete change). • No available method to calculate disturbed wind in zones of wake overlap.	• Individual wind turbine and wake dynamics solved in parallel on multiple cores. • Wake merging allowed to influence wake dynamics. • Wake deficits of downwind wind turbines dependent on impingement of wakes from upwind wind turbines. • Wake deficits superimposed in the axial direction based on the RSS method.
• Wake meandering velocity calculated with uniform spatial averaging, resulting in less meandering than expected and at improper frequencies. • The wakes meander laterally, but not axially.	• Wake meandering velocity calculated with optional weighted spatial averaging based on the jinc function to result in closer-to-ideal low-pass filtering. • Wakes meander both laterally and axially.



[1]The output calculation of AWAE is called twice at initialization--first to return the high-resolution ambient wind before initializing OF and second to fully initialize the output

[2]In OF, state update and output calculations are combined into a single routine

Figure 13. FAST.Farm parallelization process.

modularization framework (Jonkman 2013), represented mathematically as:[6]

$$x^d[n+1] = X^d\left(x^d[n], u^d[n], n\right) \tag{6.1}$$

$$y^d[n] = Y^d\left(x^d[n], u^d[n], n\right) \tag{6.2}$$

The *SC*, *OF*, and *WD* modules do not have direct feedthrough of input to output, meaning that the corresponding output functions simplify to $y^d[n] = Y^d\left(x^d[n], n\right)$. The ability of the *OF* module to be written in the above form is explained in Section 6.2.3. Additionally, the *AWAE* module does not have states, reducing the module to a feed-forward-only system and a module form that simplifies to $y^d[n] = Y^d\left(u^d[n], n\right)$. For functions in this manual, square brackets [ ] denote discrete functions and round parentheses ( ) denote continuous functions; the brackets/parentheses are dropped when implied. The states, inputs, and outputs of each of the FAST.Farm modules (*SC*, *OF*, *WD*, and *AWAE*) are listed in Table 4 and explained further in the sections below.

After initialization and within each time step, the states of each module (*SC*, *OF*, and *WD*) are updated (from time t to time $t+\Delta t$, or equivalently, n to $n+1$); time is incremented; and the module outputs are calculated and transferred as inputs to other modules. Because of the form simplifications, the state updates of each module can be solved in parallel; the output-to-input transfer does not require a large nonlinear solve; and overall correction steps of the solution are not needed. The lack of a correction step is a major simplification of the coupling algorithm used within OpenFAST (Sprague et al. 2015, 2014). Furthermore, the output calculations of the *SC*, *OF*, and *WD* modules can be parallelized, followed then by the output calculation of the *AWAE* module.[7] In parallel mode, parallelization has been implemented in FAST.Farm through OpenMP.

Because of the small timescales and sophisticated physics, the OpenFAST submodel is the computationally slowest of the FAST.Farm modules. Additionally, the output calculation of the *AWAE* module is the only major calculation that cannot be solved in parallel to OpenFAST. Because of this, the parallelized FAST.Farm solution at its fastest may execute only slightly more slowly than stand-alone OpenFAST simulations. This results in simulations that are computationally inexpensive enough to run the many simulations necessary for wind turbine/farm design and analysis.

6.2.2 Super Controller (SC Module)

Wind-farm-wide super controllers have the potential to achieve the global benefit of improving overall power performance and reducing turbine loads, based on modifying wake deficits through variations in blade pitch or generator torque and/or redirecting (steering) wakes through variations in nacelle yaw or tilt, as illustrated in Figure 14.

The *SC* module of FAST.Farm provides an interface to the super controller dynamic library—essentially identical to the super controller available in SOWFA—which allows the user of FAST.Farm to implement their own wind-farm-wide control logic in discrete time and without direct feedthrough of input to output—perhaps developed through the application of FLORIS. The inputs to the *SC* module are commands or measurements from individual turbine controllers (output from the *OF* module).[8] The outputs of the *SC* module are the global controller commands and individual turbine controller commands (inputs to the *OF* module).

Note that at time zero, the *SC* module is called before the call to the *OF* module and the associated individual turbine controllers. So, the initial outputs from the super controller are sent as inputs to the individual turbine controllers, but the initial inputs to the super controller from the individual turbine controller outputs at time zero are always zero. At subsequent time steps, the *OF* module and the associated individual turbine controllers are called before the output calculation of the *SC* module. As a result, at each time step other than time zero, the outputs from the super

[6] x^d and X^d are identical to what is described in Jonkman 2013. u^d, y^d, and Y^d are identical to u, y, and Y from Jonkman 2013, but are only evaluated in discrete time, $t = n\Delta t$, and so, are marked here with superscript ^{d}.

[7] Not all of these possible parallel tasks have been implemented within FAST.Farm because profiling did not show adequate computational speedup. However, to minimize the computational expense of the output calculation of the *AWAE* module, the ambient wind data files are read in parallel to the state updates of the *SC*, *OF*, and *WD* modules. See the introduction to section 6.2 for more information.

[8] The *SC* module also has as input a placeholder for future global (e.g., wind) measurements (output from the *AWAE* module) in addition to commands or measurements from the individual turbine controllers. But the global inputs are currently null.

Table 4. Module States, Inputs, and Outputs in FAST.Farm

Module	States (Discrete Time)	Inputs	Outputs
Super Controller (SC)	• User-defined	• Global measurements • Commands/measurements from individual turbine controllers	• Global controller commands • Commands to individual turbine controllers
OpenFAST (OF)	• None in the OpenFAST wrapper, but there are many states internal to OpenFAST	• Global controller commands • Commands to the individual turbine controller • $\vec{V}^{\text{High}}_{\text{Dist}}$	• Commands/measurements from the individual turbine controller • $\hat{x}^{\text{Disk}}$ • $\vec{p}^{\text{Hub}}$ • D^{Rotor} • γ^{YawErr} • ${}^{\text{DiskAvg}}V^{\text{Rel}}_{x}$ • ${}^{\text{AzimAvg}}C_t\left(r\right)$
Wake Dynamics (WD)	• ${}^{\text{FiltDiskAvg}}V^{\text{Rel}}_{x}$ • ${}^{\text{FiltAzimAvg}}C_t\left(r\right)$ For $0 \le n_p \le N_p - 1$: • ${}^{\text{Filt}}D^{\text{Rotor}}_{n_p}$ • ${}^{\text{Filt}}\gamma^{\text{YawErr}}_{n_p}$ • ${}^{\text{Filt}}\vec{V}^{\text{Plane}}_{n_p}$ • ${}^{\text{FiltDiskAvg}}V^{\text{Wind}}_{x_{n_p}}$ • ${}^{\text{Filt}}TI_{\text{Amb}_{n_p}}$ • $x^{\text{Plane}}_{n_p}$ • $\hat{x}^{\text{Plane}}_{n_p}$ • $\vec{p}^{\text{Plane}}_{n_p}$ • $V^{\text{Wake}}_{x_{n_p}}\left(r\right)$ • $V^{\text{Wake}}_{r_{n_p}}\left(r\right)$	• $\hat{x}^{\text{Disk}}$ • $\vec{p}^{\text{Hub}}$ • D^{Rotor} • γ^{YawErr} • ${}^{\text{DiskAvg}}V^{\text{Rel}}_{x}$ • ${}^{\text{AzimAvg}}C_t\left(r\right)$ • $\vec{V}^{\text{Plane}}_{n_p}$ for $0 \le n_p \le N_p - 1$ • ${}^{\text{DiskAvg}}V^{\text{Wind}}_{x}$ • TI_{Amb}	For $0 \le n_p \le N_p - 1$: • $\hat{x}^{\text{Plane}}_{n_p}$ • $\vec{p}^{\text{Plane}}_{n_p}$ • $V^{\text{Wake}}_{x_{n_p}}\left(r\right)$ • $V^{\text{Wake}}_{r_{n_p}}\left(r\right)$ • $D^{\text{Wake}}_{n_p}$
Ambient Wind and Array Effects (AWAE)	• None	For each turbine and $0 \le n_p \le N_p - 1$: • $\hat{x}^{\text{Plane}}_{n_p}$ • $\vec{p}^{\text{Plane}}_{n_p}$ • $V^{\text{Wake}}_{x_{n_p}}\left(r\right)$ • $V^{\text{Wake}}_{r_{n_p}}\left(r\right)$ • $D^{\text{Wake}}_{n_p}$	For each turbine: • $\vec{V}^{\text{High}}_{\text{Dist}}$ • $\vec{V}^{\text{Plane}}_{n_p}$ for $0 \le n_p \le N_p - 1$ • ${}^{\text{DiskAvg}}V^{\text{Wind}}_{x}$ • TI_{Amb}

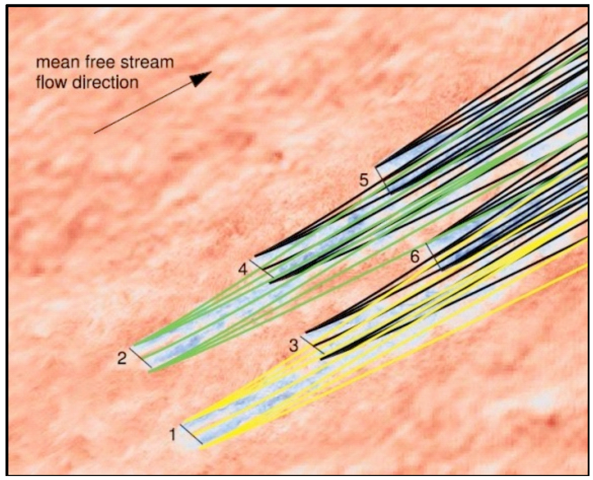


Figure 14. Nacelle-yaw control used to redirect wakes away from downwind wind turbines. Gebraad and al. 2016

controller are extrapolated in time based on past values within *OF* before being sent as input to the individual turbine controllers. Thus, care should be taken to ensure that the outputs from the super controller vary smoothly over time (without steps). See Figure 13 for more information.

6.2.3 OpenFAST (OF Module)

FAST.Farm makes use of OpenFAST to model the dynamics (loads and motions) of distinct turbines in the wind farm. OpenFAST captures the environmental excitations (wind inflow; for offshore systems, waves, current, and ice) and coupled system response of the full system (the rotor, drivetrain, nacelle, tower, controller; for offshore systems, the substructure and station-keeping system). OpenFAST itself is an interconnection of various modules, each corresponding to different physical domains of the coupled aero-hydro-servo-elastic solution. The details of the OpenFAST solution are outside the scope of this document, but can be found in the hyperlink above and associated references.

The *OF* module of FAST.Farm is a wrapper that enables the coupling of OpenFAST to FAST.Farm—similar to the OpenFAST wrapper available in SOWFA, but with different inputs and outputs (described below). This wrapper also controls subcycling of the OpenFAST state updates. The timescales solved within OpenFAST are much smaller than those within FAST.Farm. Therefore, for accuracy and numerical stability reasons, the OpenFAST time step is typically much smaller than that required of FAST.Farm, as depicted in Figure 15. There is one instance of the *OF* module for each wind turbine. In parallel mode, these instances are parallelized through OpenMP. OpenFAST itself has various modules with different inputs, outputs, states, and parameters—including continuous-time, discrete-time, algebraic, and other (e.g., logical) states. However, for the purposes of coupling OpenFAST to FAST.Farm, the *OF* module functions in discrete time and without direct feedthrough of input to output. This is achieved by calling the *OF* module at the rate dictated by the FAST.Farm time step, Δt, and by introducing a one-time-step (Δt) delay of the output relative to the input; this one-time-step delay is not expected to be problematic because of the slow timescales solved within FAST.Farm.

At initialization, the number of wind turbines (N_t, with n_t the turbine counter such that $1 \leq n_t \leq N_t$), the corresponding OpenFAST primary input files, and turbine origins in the global *X*-*Y*-*Z* inertial-frame coordinate system are specified by the user. Turbine origins are defined as the intersection of the undeflected tower centerline and the ground or, for offshore systems, mean sea level. The global inertial-frame coordinate system is defined with *Z* directed vertically upward (opposite gravity), *X* directed horizontally nominally downwind (along the zero-degree wind direction), and *Y* directed horizontally transversely.

The global and turbine-dependent commands from the super controller (outputs from the *SC* module) are used as inputs to the *OF* module to enable the individual turbine controller to be guided by wind farm-level effects; likewise,

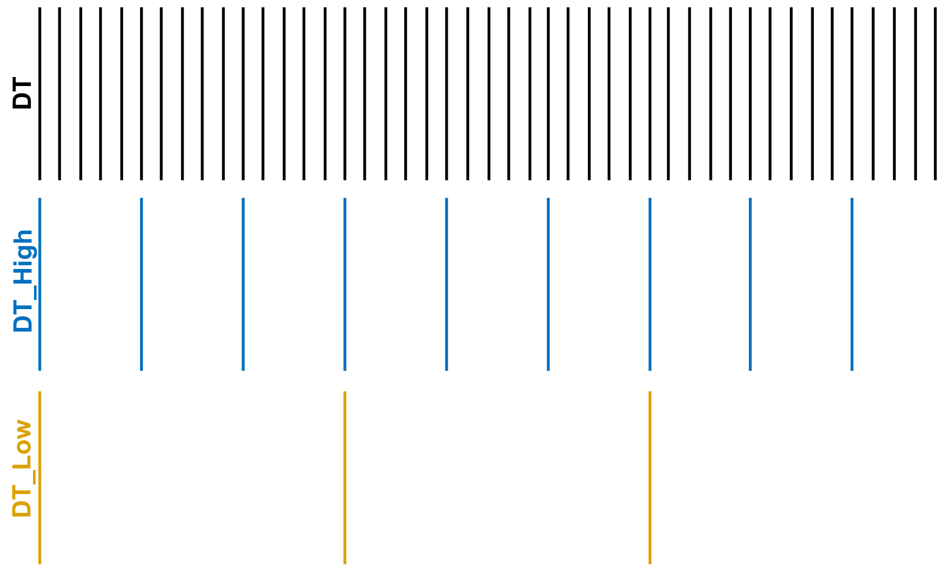


Figure 15. Illustration of timescale ranges for OpenFAST (DT), the FAST.Farm high-resolution domain (DT_High), and the FAST.Farm low-resolution domain (DT_Low).

the turbine-dependent commands or measurements are output from the *OF* module for access by the super controller (inputs to the *SC* module).

The *OF* module also uses the disturbed wind (ambient plus wakes of neighboring turbines) across a high-resolution wind domain (in both time and space) around the turbine (output from the *AWAE* module—see Section 6.2.5 for more information), $\vec{V}_{\text{Dist}}^{\text{High}}$, as input, to ensure that the individual turbine loads and response calculated by OpenFAST are accurately driven by flow through the wind farm, including wake and array effects. Spatially, the high-resolution wind domain must be large enough to encompass yawing of the rotor, blade deflection, and motion of the support structure (the latter is especially important for floating offshore wind turbines). OpenFAST uses a four-dimensional (three space dimensions plus one time dimension) interpolation to determine the wind local to its analysis nodes.

The *OF* module computes several outputs needed for calculating wake dynamics (inputs to the *WD* module). These include:

- $\hat{x}^{\text{Disk}}$—the orientation of the rotor centerline
- $\vec{p}^{\text{Hub}}$—the global position of the rotor center
- D^{Rotor}—the rotor diameter
- γ^{YawErr}—the nacelle-yaw error of the rotor
- ${}^{\text{DiskAvg}}V_x^{\text{Rel}}$—the rotor-disk-averaged relative wind speed (ambient plus wakes of neighboring turbines plus turbine motion), normal to the disk
- ${}^{\text{AzimAvg}}C_t\,(r)$—the azimuthally averaged thrust-force coefficient (normal to the rotor disk), distributed radially, where r is the radius.

In this manual, an over arrow ($\vec{}$) denotes a three-component vector and a hat ($\hat{}$) denotes a three-component unit vector. For clarity in this manual, (r) is used to denote radial dependence as a continuous function, even though the radial dependence is stored/computed on a discrete radial finite-difference grid within FAST.Farm. Except for γ^{YawErr} and ${}^{\text{AzimAvg}}C_t\,(r)$, all of the listed variables were computed within OpenFAST before the development of FAST.Farm. γ^{YawErr} is defined as the angle about global Z from the rotor centerline to the rotor-disk-averaged relative wind velocity (ambient plus wakes of neighboring turbines plus turbine motion), both projected onto the horizontal global

X-*Y* plane—see Figure 3 for an illustration. ${}^{\text{AzimAvg}}C_t(r)$ is computed by Equation 6.3

$$ {}^{\text{AzimAvg}}C_t(r) = \frac{\sum_{n_b=1}^{N_b} \left\{\hat{x}^{\text{Disk}}\right\}^T \vec{f}_{n_b}(r)}{\frac{1}{2}\rho 2\pi r \left({}^{\text{DiskAvg}}V_x^{\text{Rel}}\right)^2} \tag{6.3} $$

where:

- N_b—number of rotor blades, with n_b as the blade counter such that $1 \leq n_b \leq N_b$
- $\{\ \ \}^T$—vector transpose
- ρ—air density
- $\vec{f}_{n_b}(r)$—aerodynamic applied loads[9] distributed per unit length along a line extending radially outward in the plane of the rotor disk for blade n_b.

The numerator of Equation 6.3 is the aerodynamic applied loads distributed per unit length projected normal to the rotor disk, i.e., the radially dependent thrust force. The denominator is the normalizing factor for the radially dependent thrust coefficient, composed of the circumference at the given radius, $2\pi r$, and the dynamic pressure of the rotor-disk-averaged relative wind speed, $\frac{1}{2}\rho\left({}^{\text{DiskAvg}}V_x^{\text{Rel}}\right)^2$.

6.2.4 Wake Dynamics (WD Module)

The *WD* module of FAST.Farm calculates wake dynamics for an individual rotor, including wake advection, deflection, and meandering; a near-wake correction; and a wake-deficit increment. The near-wake correction treats the near-wake (pressure-gradient zone) expansion of the wake deficit. The wake-deficit increment shifts the quasi-steady-state axisymmetric wake deficit nominally downwind. Each submodel is described in the subsections below. There is one instance of the *WD* module for each rotor.

The wake-dynamics calculations involve many user-specified parameters that may depend, e.g., on turbine operation or atmospheric conditions that can be calibrated to better match experimental data or HFM, e.g., by running SOWFA (or equivalent) as a benchmark. Default values have been derived for each calibrated parameter based on SOWFA simulations (Doubrawa et al. 2018), but these can be overwritten by the user of FAST.Farm.

The wake-deficit evolution is solved in discrete time on an axisymmetric finite-difference grid consisting of a fixed number of wake planes, N_p (with n_p the wake-plane counter such that $0 \leq n_p \leq N_p - 1$), each with a fixed radial grid of nodes. Because the wake deficit is assumed to be axisymmetric, the radial finite-difference grid can be considered a plane. A wake plane can be thought of as a cross section of the wake wherein the wake deficit is calculated.

Inputs to the *WD* module include $\hat{x}^{\text{Disk}}$, $\vec{p}^{\text{Hub}}$, D^{Rotor}, γ^{YawErr}, ${}^{\text{DiskAvg}}V_x^{\text{Rel}}$, and ${}^{\text{AzimAvg}}C_t(r)$. Additional inputs are the advection, deflection, and meandering velocity of the wake planes for the rotor ($\vec{V}_{n_p}^{\text{Plane}}$); the rotor-disk-averaged ambient wind speed, normal to the disk (${}^{\text{DiskAvg}}V_x^{\text{Wind}}$); and the ambient turbulence intensity of the wind at the rotor (TI_{Amb}) (output from the *AWAE* module—see Section 6.2.5 for more information). $\vec{V}_{n_p}^{\text{Plane}}$ is computed for $0 \leq n_p \leq N_p - 1$ by spatial averaging of the disturbed wind.

The *WD* module computes several outputs needed for the calculation of disturbed wind, to be used as input to the *AWAE* module. These outputs include:

- $\hat{x}_{n_p}^{\text{Plane}}$—the orientations of the wake planes defined using the unit vectors normal to each plane, i.e., the orientation of the wake-plane centerline
- $\vec{p}_{n_p}^{\text{Plane}}$—the global positions of the centers of the wake planes
- $V_{x_{n_p}}^{\text{Wake}}(r)$ and $V_{r_{n_p}}^{\text{Wake}}(r)$—the axial and radial wake-velocity deficits, respectively, at the wake planes, distributed radially

[9]Derived using the Line2-to-Line2 mesh-mapping algorithm of FAST (Sprague et al. 2015, 2014) to transfer the aerodynamic applied loads distributed per unit length along the deflected/curved blade as calculated within FAST.

- $D_{n_p}^{\text{Wake}}$—the wake diameters at the wake planes, each for $0 \leq n_p \leq N_p - 1$.

Though the details are left out of this manual, during start-up—whereby a wake has not yet propagated through all of the wake planes—the number of wake planes is limited by the elapsed time to avoid having to set inputs, outputs, and states in the *WD* and *AWAE* modules beyond where the wake has propagated.

Wake Advection, Deflection, and Meandering

By simple extensions to the passive tracer solution for transverse (horizontal and vertical) wake meandering, the wake-dynamics solution in FAST.Farm is extended to account for wake deflection—as illustrated in Figure 3—and wake advection—as illustrated in Figure 4—among other physical improvements. The following extensions are introduced:

1. Calculating the wake plane velocities, $\vec{V}_{n_p}^{\text{Plane}}$ for $0 \leq n_p \leq N_p - 1$, by spatially averaging the disturbed wind instead of the ambient wind (see Section 6.2.5)
2. Orientating the wake planes with the rotor centerline instead of the wind direction
3. Low-pass filtering the local conditions at the rotor, as input to the *WD* module, to account for transients in inflow, turbine control, and/or turbine motion instead of considering time-averaged conditions.

With these extensions, the passive tracer solution enables:

1. The wake centerline to deflect based on inflow skew. This is achieved because in skewed inflow, the wake deficit normal to the disk introduces a velocity component that is not parallel to the ambient flow.
2. The wake to accelerate from near wake to far wake because the wake deficits are stronger in the near wake and weaken downwind.
3. The wake-deficit evolution to change based on conditions at the rotor because low-pass time filtered conditions are used instead of time-averaged.
4. The wake to meander axially in addition to transversely because local axial winds are considered.
5. The wake shape to be elliptical instead of circular in skewed flow when looking downwind (the wake shape remains circular when looking down the rotor centerline).

For item 3, low-pass time filtering is important because the wake reacts slowly to changes in local conditions at the rotor and because the wake evolution is treated in a quasi-steady-state fashion. Furthermore, a correction to the wake deflection resulting from item 1 is needed to account for the physical combination of wake rotation and shear, which is not modeled directly in the *WD* module. This is achieved through a horizontally asymmetric correction to the wake deflection from item 1 (see Figure 3 for an illustration). This horizontal wake-deflection correction is a simple linear correction with slope and offset, similar to the correction implemented in the wake model of FLORIS. It is important for accurate modeling of nacelle-yaw-based wake-redirection (wake-steering) wind farm control.

Mathematically, the low-pass time filter is implemented using a recursive, single-pole filter with exponential smoothing (Smith 2006). The discrete-time recursion (difference) equation for this filter is (Jonkman et al. 2009):

$$x_{n_p}^{d}[n+1] = x_{n_p}^{d}[n]\,\alpha + u^{d}[n]\,(1-\alpha) \qquad \text{for } n_p = 0 \tag{6.4}$$

where

- x^d—discrete-time state storing the low-pass time-filtered value of input u^d
- $\alpha = e^{-2\pi\Delta t f_c}$—low-pass time-filter parameter, with a value between 0 (minimum filtering) and 1 (maximum filtering) (exclusive)
- f_c—user-specified cutoff (corner) frequency (the time constant of the low-pass time filter is $\frac{1}{f_c}$).

Subscript n_p is used to denote the state associated with wake-plane n_p; Equation 6.4 applies at the rotor disk, where $n_p = 0$.

To be consistent with the quasi-steady-state treatment of the wake-deficit evolution (see Section 6.2.4), the conditions at the rotor are maintained as fixed states of a wake plane as the plane propagates downstream

$$x^d_{n_p}[n+1] = x^d_{n_p-1}[n] \qquad \text{for } 1 \le n_p \le N_p - 1 \tag{6.5}$$

Equations 6.4 and 6.5 apply directly to the *WD* module inputs D^{Rotor}[10], γ^{YawErr}, ${}^{\text{DiskAvg}}V_x^{\text{Rel}}$, and TI_{Amb}. The associated states are ${}^{\text{Filt}}D^{\text{Rotor}}_{n_p}$, ${}^{\text{Filt}}\gamma^{\text{YawErr}}_{n_p}$, ${}^{\text{FiltDiskAvg}}V^{\text{Wind}}_{x_{n_p}}$, and ${}^{\text{Filt}}TI_{\text{Amb}_{n_p}}$ respectively (each for $0 \le n_p \le N_p - 1$). The *WD* module inputs ${}^{\text{DiskAvg}}V_x^{\text{Rel}}$ and ${}^{\text{AzimAvg}}C_t(r)$ are needed for the boundary condition at the rotor, but are not otherwise needed in the wake-deficit evolution calculation and are therefore not propagated downstream with the wake planes. Therefore, Equation 6.4 applies to these inputs but Equation 6.5 does not. The associated states are ${}^{\text{FiltDiskAvg}}V_x^{\text{Rel}}$ and ${}^{\text{FiltAzimAvg}}C_t(r)$. Likewise, only Equation 6.4 is used to low-pass time filter the *WD* module input $\vec{V}^{\text{Plane}}_{n_p}$ with state ${}^{\text{Filt}}\vec{V}^{\text{Plane}}_{n_p}$ (for $0 \le n_p \le N_p - 1$). Equations 6.4 and 6.5 apply in a modified form to the *WD* module inputs $\hat{x}^{\text{Disk}}$ and $\vec{p}^{\text{Hub}}$ to derive the state associated with the downwind distance from the rotor to each wake plane in the axisymmetric coordinate system ($x^{\text{Plane}}_{n_p}$), and the states and outputs associated with the orientations of the wake planes, normal to the planes, ($\hat{x}^{\text{Plane}}_{n_p}$), and the global center positions of the wake planes, ($\vec{p}^{\text{Plane}}_{n_p}$) as follows:

$$\hat{x}^{\text{Plane}}_{n_p}[n+1] = \begin{cases} \dfrac{\hat{x}^{\text{Plane}}_{n_p}[n]\alpha + \hat{x}^{\text{Disk}}(1-\alpha)}{\left\| \hat{x}^{\text{Plane}}_{n_p}[n]\alpha + \hat{x}^{\text{Disk}}(1-\alpha) \right\|_2} & \text{for } n_p = 0 \\ \hat{x}^{\text{Plane}}_{n_p-1}[n] & \text{for } 1 \le n_p \le N_p - 1 \end{cases} \tag{6.6}$$

$$x^{\text{Plane}}_{n_p}[n+1] = \begin{cases} 0 & \text{for } n_p = 0 \\ x^{\text{Plane}}_{n_p-1}[n] + \left| d\hat{x}_{n_p-1} \right| & \text{for } 1 \le n_p \le N_p - 1 \end{cases} \tag{6.7}$$

$$\vec{p}^{\text{Plane}}_{n_p}[n+1] = \begin{cases} \vec{p}^{\text{Plane}}_{n_p}[n]\,\alpha + \left\{ \vec{p}^{\text{Hub}}[n] + \left(C^{\text{O}}_{\text{HWkDfl}} + C^{\text{OY}}_{\text{HWkDfl}}\ {}^{\text{Filt}}\gamma^{\text{YawErr}}_{n_p}[n+1] \right) \widehat{XY_{n_p}} \right\} (1-\alpha) & for\ n_p = 0 \\ \vec{p}^{\text{Plane}}_{n_p-1}[n] + \hat{x}^{\text{Plane}}_{n_p-1}[n]\ d\hat{x}_{n_p-1} + \left[I - \hat{x}^{\text{Plane}}_{n_p-1}[n] \left\{ \hat{x}^{\text{Plane}}_{n_p-1}[n] \right\}^T \right] \vec{V}^{\text{Plane}}_{n_p-1} \Delta t & \\ \quad + \left(\left(C^{\text{x}}_{\text{HWkDfl}} + C^{\text{xY}}_{\text{HWkDfl}}\ {}^{\text{Filt}}\gamma^{\text{YawErr}}_{n_p-1}[n] \right) d\hat{x}_{n_p-1} \right) \widehat{XY_{n_p-1}} & for\ 1 \le n_p \le N_p - 1 \end{cases} \tag{6.8}$$

where:

$$d\hat{x}_{n_p-1} = \left\{ \hat{x}^{\text{Plane}}_{n_p-1}[n] \right\}^T\ {}^{\text{Filt}}\vec{V}^{\text{Plane}}_{n_p-1}[n+1]\,\Delta t \tag{6.9}$$

$$\widehat{XY_{n_p}} = \left\{ \frac{\left(\left\{ \hat{x}^{\text{Plane}}_{n_p}[n+1] \right\}^T \hat{X} \right) \hat{Y} - \left(\left\{ \hat{x}^{\text{Plane}}_{n_p}[n+1] \right\}^T \hat{Y} \right) \hat{X}}{\left\| \left(\left\{ \hat{x}^{\text{Plane}}_{n_p}[n+1] \right\}^T \hat{X} \right) \hat{X} + \left(\left\{ \hat{x}^{\text{Plane}}_{n_p}[n+1] \right\}^T \hat{Y} \right) \hat{Y} \right\|_2} \right\} \tag{6.10}$$

Equation 6.6 differs from Equations 6.4 and 6.5 in that after applying Equation 6.4 to low-pass time-filter input $\hat{x}^{\text{Disk}}$, the state is renormalized to ensure that the vector remains unit length; Equation 6.6 ensures that the wake-plane orientation is maintained as the planes propagate nominally downwind. Equation 6.7 expresses that each wake plane propagates downwind in the axisymmetric coordinate system by a distance equal to that traveled by the low-pass time-filtered wake-plane velocity projected along the plane orientation over the time step;[11] the initial wake plane

[10] Variations in the rotor diameter, D^{Rotor}, are possible as a result of blade deflection. These variations are likely small, but this variable is treated the same as other inputs for consistency.

[11] The absolute value is added because, as far as wake evolution is concerned, if a wake plane travels opposite of its original propagation direction (e.g., due to a localized wind gust), the total downwind distance traveled is used rather than the instantaneous downwind distance from the rotor.

($n_p = 0$) is always at the rotor disk. Equation 6.8 expresses the global center positions of the wake plane following the passive tracer concept, similar to Equation 6.7, but considering the full three-component movement of the wake plane, including deflection and meandering. The last term on the right-hand side of Equation 6.8 for each wake plane is the horizontal wake-deflection correction, where:

- C^{O}_{HWkDfl}—user-specified parameter defining the horizontal offset at the rotor
- C^{OY}_{HWkDfl}—user-specified parameter defining the horizontal offset at the rotor scaled with nacelle-yaw error
- C^{x}_{HWkDfl}—user-specified parameter defining the horizontal offset scaled with downstream distance
- C^{xY}_{HWkDfl}—user-specified parameter defining the horizontal offset scaled with downstream distance and nacelle-yaw error
- $\hat{X}$, $\hat{Y}$, and $\hat{Z}$—unit vectors parallel to the inertial-frame coordinates X, Y and, Z respectively
- $\widehat{XY_{np}}$—three-component unit vector in the horizontal global X-Y plane orthogonal to $\hat{x}^{\text{Plane}}_{n_p}[n+1]$
- $C^{\text{O}}_{\text{HWkDfl}} + C^{\text{OY}}_{\text{HWkDfl}}\ {}^{\text{Filt}}\gamma^{\text{YawErr}}_{n_p}[n+1]$—offset at the rotor
- $C^{\text{x}}_{\text{HWkDfl}} + C^{\text{xY}}_{\text{HWkDfl}}\ {}^{\text{Filt}}\gamma^{\text{YawErr}}_{n_p}[n+1]$—slope
- $d\hat{x}_{n_p-1}$—nominally downwind increment of the wake plane (from Equation 6.7)
- I—three-by-three identity matrix
- $\left[I - \hat{x}^{\text{Plane}}_{n_p-1}[n]\left\{\hat{x}^{\text{Plane}}_{n_p-1}[n]\right\}^T\right]$—used to calculate the transverse component of $V^{\text{Plane}}_{n_p-1}$ normal to $\hat{x}^{\text{Plane}}_{n_p-1}[n]$.

It is noted that the advection, deflection, and meandering velocity of the wake planes, $\vec{V}^{\text{Plane}}_{n_p-1}$, is low-pass time filtered in the axial direction, but not in the transverse direction. Low-pass time filtering in the axial direction is useful for minimizing how often wake planes get close to or pass each other while they travel axially; this filtering is not needed transversely because an appropriate transverse meandering velocity is achieved through spatially averaging the disturbed wind (see Section 6.2.5).

The consistent output equation corresponding to the low-pass time filter of Equation 6.4 is $y^d[n] = x^d[n]\,\alpha + u^d[n](1-\alpha)$, i.e., $Y^d(\quad) = X^d(\quad)$, or equivalently, $y^d[n] = x^d[n+1]$ (Jonkman et al. 2009). However, the output is delayed by one time step (Δt) to avoid having direct feedthrough of input to output within the *WD* module, yielding $y^d[n] = x^d[n]$. This one-time-step delay is applied to all outputs of the *WD* module and is not expected to be problematic because of the slow timescales solved within FAST.Farm.

Near-Wake Correction

The near-wake correction submodel of the *WD* module computes the axial and radial wake-velocity deficits at the rotor disk as an inlet boundary condition for the wake-deficit evolution described in Section 6.2.4. To improve the accuracy of the far-wake solution, the near-wake correction accounts for the drop in wind speed and radial expansion of the wake in the pressure-gradient zone behind the rotor that is not otherwise accounted for in the solution for the wake-deficit evolution. For clarity, the equations in this section are expressed using continuous variables, but within FAST.Farm the equations are solved discretely on an axisymmetric finite-difference grid.

The near-wake correction is computed differently for low thrust conditions ($C_T < \frac{24}{25}$), momentum theory is valid, and high thrust conditions ($1.1 < C_T \leq 2$), where C_T is the rotor disk-averaged thrust coefficient, derived from the low-pass time-filtered azimuthally averaged thrust-force coefficient (normal to the rotor disk), ${}^{\text{FiltAzimAvg}}C_t(r)$, evaluated at $n+1$. The propeller brake region occurs for very high thrust-force coefficients ($C_T \geq 2$) and is not considered. Between the low and high thrust regions, a linear blending of the two solutions, based on C_T, is implemented.

At low thrust ($C_T < \frac{24}{25}$) conditions, the axial induction at the rotor disk, distributed radially, $a(r)$, is derived from the low-pass time-filtered azimuthally averaged thrust-force coefficient (normal to the rotor disk), ${}^{\text{FiltAzimAvg}}C_t(r)$, evaluated at $n+1$ using Equation 6.11, which follows from the momentum region of blade-element momentum (BEM) theory.

$$a(r) = \frac{1}{2}\left(1 - \sqrt{1 - MIN\left[{}^{\text{FiltAzimAvg}}C_t(r), \frac{24}{25}\right]}\right) \tag{6.11}$$

To avoid unrealistically high induction at the ends of a blade, Equation 6.11 does not directly consider hub- or tip-loss corrections, but these may be accounted for in the calculation of the applied aerodynamic loads within OpenFAST (depending on the aerodynamic options enabled within OpenFAST), which have an effect on ${}^{\text{FiltAzimAvg}}C_t(r)$. Moreover, ${}^{\text{FiltAzimAvg}}C_t(r)$ is capped at $\frac{24}{25}$ to avoid ill-conditioning of the radial wake expansion discussed next.

The states and outputs associated with the axial and radial wake-velocity deficits, distributed radially ($V_{x_{n_p}}^{\text{Wake}}(r)$ and $V_{r_{n_p}}^{\text{Wake}}(r)$), are derived at the rotor disk ($n_p = 0$) from $a(r)$ and the low-pass time-filtered rotor-disk-averaged relative wind speed (ambient plus wakes of neighboring turbines plus turbine motion), normal to the disk (${}^{\text{FiltDiskAvg}}V_x^{\text{Rel}}$), evaluated at $n+1$ using Equations 6.12 and 6.13.

$$V_{x_{n_p}}^{\text{Wake}}(r^{\text{Plane}})|_{n_p=0} = -{}^{\text{FiltDiskAvg}}V_x^{\text{Rel}}C_{\text{NearWake}}a(r) \tag{6.12}$$

$$V_{r_{n_p}}^{\text{Wake}}(r^{\text{Plane}})|_{n_p=0} = 0 \tag{6.13}$$

where

$$r^{\text{Plane}} = \sqrt{2\int_0^r \frac{1-a(r')}{1-C_{\text{NearWake}}a(r')} r'\, dr'} \tag{6.14}$$

In Equation 6.12:

- r^{Plane}—radial expansion of the wake associated with r
- r'—dummy variable of r
- C_{NearWake}—user-specified calibration parameter greater than unity and less than 2.5 which determines how far the wind speed drops and wake expands radially in the pressure-gradient zone before recovering in the far wake.[12]

The right-hand side of Equation 6.12 represents the axial-induced velocity at the end of the pressure-gradient zone; the negative sign appears because the axial wake deficit is in the opposite direction of the free stream axial wind—see Section 6.2.4 for more information. The radial expansion of the wake in the left-hand side of Equation 6.12 results from the application of the conservation of mass within an incremental annulus in the pressure-gradient zone.[13] The radial wake deficit is initialized to zero, as given in Equations 6.13. Because the near-wake correction is applied directly at the rotor disk, the solution to the wake-deficit evolution for downwind distances within the first few diameters of the rotor, i.e., in the near wake, is not expected to be accurate; as a result, modifications to FAST.Farm would be needed to accurately model closely spaced wind farms.

At high thrust ($1.1 < C_T \leq 2$) conditions, the axial wake-velocity deficit, distributed radially ($V_{x_{n_p}}^{\text{Wake}}(r)$), is derived at the rotor disk ($n_p = 0$) by a Gaussian fit to LES solutions at high thrust per Equation 6.16, as derived by Martinez-Tossas 2021. The radial wake deficit is again initialized to zero.

$$V_{x_{n_p}}^{\text{Wake}}(r)|_{n_p=0} = -\mu(C_T)\,{}^{\text{FiltDiskAvg}}V_x^{\text{Rel}}\, e^{-\left(\frac{r}{\sigma(C_T)\,{}^{\text{Filt}}D_{n_p}^{\text{Rotor}}|_{n_p=0}}\right)^2} \tag{6.16}$$

[12]A value of $C_{\text{NearWake}} = 2$ is expected from first principles, but can be calibrated by the user of FAST.Farm to better match the far wake to known solutions.

[13]The incremental mass flow is given by:

$$d\dot{m} = 2\pi r dr \rho\; {}^{\text{FiltDiskAvg}}V_x^{\text{Rel}}(1-a(r)) = 2\pi r^{\text{Plane}} dr^{\text{Plane}} \rho\; {}^{\text{FiltDiskAvg}}V_x^{\text{Rel}}(1-C_{\text{NearWake}}a(r)) \tag{6.15}$$

Following from this, $r^{\text{Plane}} dr^{\text{Plane}} = \frac{1-a(r)}{1-C_{\text{NearWake}}a(r)} r dr$, which can then be integrated along the radius.

where

$$\mu(C_T) = \frac{0.3}{2C_T^2 - 1} + \frac{1}{5} \tag{6.17}$$

$$\sigma(C_T) = \frac{C_T}{2} + \frac{4}{25} \tag{6.18}$$

Wake-Deficit Increment

As with most DWM implementations, the *WD* module of FAST.Farm models the wake-deficit evolution via the thin shear-layer approximation of the Reynolds-averaged Navier-Stokes equations under quasi-steady-state conditions in axisymmetric coordinates, with turbulence closure captured by using an eddy-viscosity formulation (Ainslie 1988). The thin shear-layer approximation drops the pressure term and assumes that the velocity gradients are much bigger in the radial direction than in the axial direction. With these simplifications, analytical expressions for the conservation of momentum (Equation 6.19) and conservation of mass (continuity, Equation 6.20) are as follows:

$$V_x \frac{\partial V_x}{\partial x} + V_r \frac{\partial V_x}{\partial r} = \frac{1}{r}\frac{\partial}{\partial r}\left(r\nu_T \frac{\partial V_x}{\partial r}\right), \text{ or equivalently, } rV_x \frac{\partial V_x}{\partial x} + rV_r \frac{\partial V_x}{\partial r} = \nu_T \frac{\partial V_x}{\partial r} + r\nu_T \frac{\partial^2 V_x}{\partial r^2} + r\frac{\partial \nu_T}{\partial r}\frac{\partial V_x}{\partial r} \tag{6.19}$$

$$\frac{\partial V_x}{\partial x} + \frac{1}{r}\frac{\partial}{\partial r}(rV_r) = 0, \text{ or equivalently, } V_r + r\frac{\partial V_r}{\partial r} + r\frac{\partial V_x}{\partial x} = 0 \tag{6.20}$$

where V_x and V_r are the axial and radial velocities in the axisymmetric coordinate system, respectively, and ν_T is the eddy viscosity (all dependent on x and r). The equations on the left are written in a form common in literature. The equivalent equations on the right are written in the form implemented within FAST.Farm. For clarity, the equations in this section are first expressed using continuous variables, but within FAST.Farm the equations are solved discretely on an axisymmetric finite-difference grid consisting of a fixed number of wake planes, as summarized at the end of this section. For the continuous variables, subscript n_p, corresponding to wake plane n_p, is replaced with (x). The subscript is altogether dropped for variables that remain constant as the wake propagates downstream, following Equation 6.5. For example, ${}^{\text{Filt}}D^{\text{Rotor}}_{n_p}$, ${}^{\text{FiltDiskAvg}}V^{\text{Wind}}_{x_{n_p}}$, and ${}^{\text{Filt}}TI_{\text{Amb}_{n_p}}$ are written as ${}^{\text{Filt}}D^{\text{Rotor}}$, ${}^{\text{FiltDiskAvg}}V^{\text{Wind}}_x$, and ${}^{\text{Filt}}TI_{\text{Amb}}$, respectively.

V_x and V_r are related to the low-pass time-filtered rotor-disk-averaged ambient wind speed, normal to the disk (${}^{\text{FiltDiskAvg}}V^{\text{Wind}}_x$), and the states and outputs associated with radially distributed axial and radial wake-velocity deficits, $V^{\text{Wake}}_x(x,r)$ and $V^{\text{Wake}}_r(x,r)$, respectively, by Equations 6.21 and 6.22.

$$V_x(x,r) = {}^{\text{FiltDiskAvg}}V^{\text{Wind}}_x + V^{\text{Wake}}_x(x,r) \tag{6.21}$$

$$V_r(x,r) = V^{\text{Wake}}_r(x,r) \tag{6.22}$$

$V_x(x,r)$ and $V_r(x,r)$ can be thought of as the change in wind velocity in the wake relative to free stream; therefore, $V^{\text{Wake}}_x(x,r)$ usually has a negative value. Several variations of the eddy-viscosity formulation have been used in prior implementations of DWM. The eddy-viscosity formulation currently implemented within FAST.Farm is given by Equation 6.23.

$$\begin{aligned}\nu_T(x,r) =& F_{\nu\text{Amb}}(x) k_{\nu\text{Amb}}\, {}^{\text{Filt}}TI_{\text{Amb}}\, {}^{\text{FiltDiskAvg}}V^{\text{Wind}}_x \frac{{}^{\text{Filt}}D^{\text{Rotor}}}{2} \\ &+ F_{\nu\text{Shr}}(x) k_{\nu\text{Shr}} MAX\left[\left(\frac{D^{\text{Wake}}(x)}{2}\right)^2 \left|\frac{\partial V_x}{\partial r}(x,r)\right|, \frac{D^{\text{Wake}}(x)}{2} MIN\Big|_r \{V_x(x,r)\}\right]\end{aligned} \tag{6.23}$$

where:

- $F_{\nu\text{Amb}}(x)$—filter function associated with ambient turbulence
- $F_{\nu\text{Shr}}(x)$—filter function associated with the wake shear layer
- $k_{\nu\text{Amb}}$—user-specified calibration parameters weighting the influence of ambient turbulence on the eddy viscosity

- $k_{\nu\text{Shr}}$—user-specified calibration parameters weighting the influence of the wake shear layer on the eddy viscosity
- $\frac{D^{\text{Wake}}(x)}{2}$—wake half-width
- $|\frac{\partial V_x}{\partial r}|$—absolute value of the radial gradient of the axial velocity
- $MIN|_r(V_x(x,r))$—used to denote the minimum value of V_x along the radius for a given downstream distance.

Although not matching any specific eddy-viscosity formulation found in prior implementations of DWM, the chosen implementation within FAST.Farm is simple to apply and inherently tailorable, allowing the user to properly calibrate the wake evolution to known solutions. The eddy-viscosity formulation expresses the influence of the ambient turbulence (first term on the right-hand side) and wake shear layer (second term) on the turbulent stresses in the wake. The dependence of the eddy viscosity on x and r is explicitly given in Equations 6.23 to make it clear which terms depend on the downwind distance and/or radius. The first term on the right-hand side of Equations 6.23 is similar to that given by Madsen et al. 2010 with a characteristic length taken to be the rotor radius, $\frac{^{\text{Filt}}D^{\text{Rotor}}}{2}$. The second term is similar to that given by Keck et al. 2013, but without consideration of atmospheric shear, which is considered by the *AWAE* module in the definition of ambient turbulence—see Section 6.2.5 for more information. In this second term, the characteristic length is taken to be the wake half-width and the $MAX(\quad)$ operator is used to denote the maximum of the two wake shear-layer methods. The second shear-layer method is needed to avoid underpredicting the turbulent stresses from the first method at radii where the radial gradient of the axial velocity approaches zero.

The filter functions currently implemented within FAST.Farm are given by Equations 6.24 and 6.25, where $C_{\nu\text{Amb}}^{DMax}$, $C_{\nu\text{Amb}}^{DMin}$, $C_{\nu\text{Amb}}^{Exp}$, $C_{\nu\text{Amb}}^{FMin}$, $C_{\nu\text{Shr}}^{DMax}$, $C_{\nu\text{Shr}}^{DMin}$, $C_{\nu\text{Shr}}^{Exp}$, and $C_{\nu\text{Shr}}^{FMin}$ are user-specified calibration parameters for the functions associated with ambient turbulence and the wake shear layer, respectively.

$$F_{\nu\text{Amb}}(x) = \begin{cases} C_{\nu\text{Amb}}^{\text{FMin}} & \text{for } x \le C_{\nu\text{Amb}}^{\text{DMin}}\, {}^{\text{Filt}}D^{\text{Rotor}} \\ C_{\nu\text{Amb}}^{\text{FMin}} + \left(1 - C_{\nu\text{Amb}}^{\text{FMin}}\right)\left[\dfrac{\frac{x}{^{\text{Filt}}D^{\text{Rotor}}} - C_{\nu\text{Amb}}^{\text{DMin}}}{C_{\nu\text{Amb}}^{\text{DMax}} - C_{\nu\text{Amb}}^{\text{DMin}}}\right]^{C_{\nu\text{Amb}}^{\text{Exp}}} & \text{for } C_{\nu\text{Amb}}^{\text{DMin}}\, {}^{\text{Filt}}D^{\text{Rotor}} < x < C_{\nu\text{Amb}}^{\text{DMax}}\, {}^{\text{Filt}}D^{\text{Rotor}} \\ 1 & \text{for } x \ge C_{\nu\text{Amb}}^{\text{DMax}}\, {}^{\text{Filt}}D^{\text{Rotor}} \end{cases} \tag{6.24}$$

$$F_{\nu\text{Shr}}(x) = \begin{cases} C_{\nu\text{Shr}}^{\text{FMin}} & \text{for } x \le C_{\nu\text{Shr}}^{\text{DMin}}\, {}^{\text{Filt}}D^{\text{Rotor}} \\ C_{\nu\text{Shr}}^{\text{FMin}} + \left(1 - C_{\nu\text{Shr}}^{\text{FMin}}\right)\left[\dfrac{\frac{x}{^{\text{Filt}}D^{\text{Rotor}}} - C_{\nu\text{Shr}}^{\text{DMin}}}{C_{\nu\text{Shr}}^{\text{DMax}} - C_{\nu\text{Shr}}^{\text{DMin}}}\right]^{C_{\nu\text{Shr}}^{\text{Exp}}} & \text{for } C_{\nu\text{Shr}}^{\text{DMin}}\, {}^{\text{Filt}}D^{\text{Rotor}} < x < C_{\nu\text{Shr}}^{\text{DMax}}\, {}^{\text{Filt}}D^{\text{Rotor}} \\ 1 & \text{for } x \ge C_{\nu\text{Shr}}^{\text{DMax}}\, {}^{\text{Filt}}D^{\text{Rotor}} \end{cases} \tag{6.25}$$

The filter functions of Equations 6.24 and 6.25 represent the delay in the turbulent stress generated by ambient turbulence and the development of turbulent stresses generated by the wake shear layer, respectively, and are made general in FAST.Farm. Each filter function is split into three regions of downstream distance, including:

1. A fixed minimum value (between zero and unity, inclusive) near the rotor
2. A fixed value of unity far downstream from the rotor
3. A transition region for intermediate distances, where the value can transition linearly or via any rational exponent of the normalized downstream distance within the transition region.

The definition of wake diameter is somewhat ambiguous and not defined consistently in DWM literature. FAST.Farm allows the user to choose one of several methods to calculate the wake diameter, $D^{\text{Wake}}(x)$, including taking the wake diameter to be:

1. The rotor diameter
2. The diameter at which the axial velocity of the wake is the C_{WakeDiam} fraction of the ambient wind speed, where C_{WakeDiam} is a user-specified calibration parameter between zero and 0.99 (exclusive)
3. The diameter that captures the C_{WakeDiam} fraction of the mass flux of the axial wake deficit across the wake plane
4. The diameter that captures the C_{WakeDiam} fraction of the momentum flux of the axial wake deficit across the wake plane.

Through the use of a $MAX(\quad)$ operator, models 2 through 4 have a lower bound set equal to the rotor diameter when the wake-diameter calculation otherwise returns smaller values. This is done to avoid numerical problems resulting from too few wind data points in the spatial averaging used to compute the wake-meandering velocity—see Section 6.2.5 for more information. Although the implementation in FAST.Farm is numerical, analytical expressions for these four methods are given in Equation 6.26. Here, $|x$ means the mean conditioned on x.

$$D^{\text{Wake}}(x) = \begin{cases} {}^{\text{Filt}}D^{\text{Rotor}} \\ \text{for } method\,1 - rotor\,diameter \\ MAX\left({}^{\text{Filt}}D^{\text{Rotor}}, \left\{2r|\left(V_x(x,r) = C_{\text{WakeDiam}}\,{}^{\text{FiltDiskAvg}}V_x^{\text{Wind}}\right)\right\}\right) \\ \text{for } method\,2 - velocity\,based \\ MAX\left({}^{\text{Filt}}D^{\text{Rotor}}, \left\{D^{\text{Wake}}(x)\,|\int\limits_0^{\frac{D^{\text{Wake}}(x)}{2}} V_x^{\text{Wake}}(x,r)\,2\pi r dr = C_{\text{WakeDiam}}\int\limits_0^{\infty} V_x^{\text{Wake}}(x,r)\,2\pi r dr\right\}\right) \\ \text{for } method\,3 - mass\text{-}flux\,based \\ MAX\left({}^{\text{Filt}}D^{\text{Rotor}}, \left\{D^{\text{Wake}}(x)\,|\int\limits_0^{\frac{D^{\text{Wake}}(x)}{2}} \left(V_x^{\text{Wake}}(x,r)\right)^2 2\pi r dr = C_{\text{WakeDiam}}\int\limits_0^{\infty} \left(V_x^{\text{Wake}}(x,r)\right)^2 2\pi r dr\right\}\right) \\ \text{for } method\,4 - momentum\text{-}flux\,based \end{cases} \tag{6.26}$$

The momentum and continuity equations are solved numerically in the wake-deficit-increment submodel of the *WD* module using a second-order accurate finite-difference method at $n+\frac{1}{2}$, following the implicit Crank-Nicolson method (Crank and Nicolson 1996). Following this method, central differences are used for all derivatives, e.g., Equation 6.27 for the momentum equation.

$$\frac{\partial V_x}{\partial x} = \frac{V_{x_{n_p}}^{\text{Wake}}(r)[n+1] - V_{x_{n_p-1}}^{\text{Wake}}(r)[n]}{\Delta x} \tag{6.27}$$

Here,

$$\Delta x = |x_{n_p}^{\text{Plane}}[n+1] - x_{n_p-1}^{\text{Plane}}[n]| \tag{6.28}$$

or equivalently from Equation 6.9

$$\Delta x = |\left\{\hat{x}_{n_p-1}^{\text{Plane}}[n]\right\}^T\,{}^{\text{Filt}}\vec{V}_{n_p-1}^{\text{Plane}}[n+1]\,\Delta t| \qquad \text{for } 1 \le n_p \le N_p - 1 \tag{6.29}$$

For the momentum equation, for each wake plane downstream of the rotor ($1 \le n_p \le N_p - 1$), the terms V_x, V_r, ν_T, and $\frac{\partial \nu_T}{\partial r}$ are calculated at n (or equivalently $x = x_{n_p-1}^{\text{Plane}}[n]$), e.g., $V_x = {}^{\text{FiltDiskAvg}}V_{x_{n_p-1}}^{\text{Wind}}[n] + V_{x_{n_p-1}}^{\text{Wake}}(r)[n]$ and $V_r = V_{r_{n_p-1}}^{\text{Wake}}(r)[n]$, to avoid nonlinearities in the solution for $n+1$. This will prevent the solution from achieving second-order convergence, but has been shown to remain numerically stable. Although the definition of each central difference is outside the scope of this document, the end result is that for each wake plane downstream of the rotor,

$V^{\text{Wake}}_{x_{np}}(r)[n+1]$ can be solved via a linear tridiagonal matrix system of equations in terms of known solutions of $V^{\text{Wake}}_{x_{np-1}}(r)[n]$, $V^{\text{Wake}}_{r_{np-1}}(r)[n]$, and other previously calculated states, e.g., ${}^{\text{FiltDiskAvg}}V^{\text{Wind}}_{x_{np-1}}[n]$. The linear tridiagonal matrix system of equations is solved efficiently in FAST.Farm via the Thomas algorithm (Thomas 1949).

For the continuity equation, a different finite-difference scheme is needed because the resulting tridiagonal matrix is not diagonally dominant when the same finite-difference scheme used for the momentum equation is used for the continuity equation, resulting in a numerically unstable solution. Instead, the finite-difference scheme used for the continuity equation is based on a second-order accurate scheme at $n+\frac{1}{2}$ and $n_r-\frac{1}{2}$. However, the terms involving V_r and $\frac{\partial V_r}{\partial r}$ are calculated at $n+1$, e.g., $V_r = \frac{1}{2}\left(V^{\text{Wake}}_{r_{np,nr}}[n+1] + V^{\text{Wake}}_{r_{np,nr-1}}[n+1]\right)$, where n_r is the radii counter for N_r radial nodes ($0 \le n_r \le N_r - 1$).[14] Although the definition of each central difference is outside the scope of this document, the end result is that for each wake plane downstream of the rotor, $V^{\text{Wake}}_{r_{np,nr}}[n+1]$ can be solved explicitly sequentially from known solutions of $V^{\text{Wake}}_{x_{np}}(r)[n+1]$ (from the solution of the momentum equation), $V^{\text{Wake}}_{x_{np-1}}(r)[n]$, and $V^{\text{Wake}}_{r_{np,nr-1}}[n+1]$ for $1 \le n_r \le N_r - 1$.[15]

6.2.5 Ambient Wind and Array Effects (AWAE Module)

The *AWAE* module of FAST.Farm processes ambient wind and wake interactions across the wind farm, including the ambient wind and wake-merging submodels. The ambient wind submodule processes ambient wind across the wind farm from either a high-fidelity precursor simulation or an interface to the *InflowWind* module in OpenFAST. The wake-merging submodule identifies zones of overlap between all wakes across the wind farm and merges their wake deficits. Both submodels are described in the subsections below.

The calculations in the *AWAE* module make use of wake volumes, which are volumes formed by a (possibly curved) cylinder starting at a wake plane and extending to the next adjacent wake plane along a line connecting the centers of the two wake planes. If the adjacent wake planes (top and bottom of the cylinder) are not parallel, e.g., for transient simulations involving variations in nacelle-yaw angle, the centerline will be curved instead of straight. Figure 5 illustrates some of the concepts that will be detailed in the subsections below. The calculations in the *AWAE* module also require looping through all wind data points, turbines, and wake planes; these loops have been sped up in the parallel mode of FAST.Farm by implementation of OpenMP parallelization.

The *AWAE* module does not have states, reducing the module to a feed-forward-only system whereby the module outputs are computed directly from the module inputs (with direct feedthrough of input to output). The *AWAE* module uses as input $\hat{x}^{\text{Plane}}_{np}$, $\vec{p}^{\text{Plane}}_{np}$, $V^{\text{Wake}}_{x_{np}}(r)$, $V^{\text{Wake}}_{r_{np}}(r)$, and D^{Wake}_{np} (each for $0 \le n_p \le N_p - 1$) as computed by the wake-dynamics model for each individual wind turbine (output by the *WD* module). The *AWAE* module computes output $\vec{V}^{\text{High}}_{\text{Dist}}$ needed for the calculation of OpenFAST for each individual wind turbine (input to the *OF* module) as well as outputs for $\vec{V}^{\text{Plane}}_{np}$ for $0 \le n_p \le N_p - 1$, ${}^{\text{DiskAvg}}V^{\text{Wind}}_{x}$, and TI_{Amb} needed for the calculation of wake dynamics for each individual wind turbine (input to the *WD* module).

Ambient Wind

The ambient wind data used by FAST.Farm can be generated in one of two ways. The use of the *InflowWind* module in OpenFAST enables the use of simple ambient wind, e.g., uniform wind, discrete wind events, or synthetically generated turbulent wind data. Synthetically generated turbulence can be from, e.g., TurbSim or the Mann model, in which the wind is propagated through the wind farm using Taylor's frozen-turbulence assumption. This method is most applicable to small wind farms or a subset of wind turbines within a larger wind farm. FAST.Farm can also use ambient wind generated by a high-fidelity precursor LES simulation of the entire wind farm (without wind turbines present), such as the ABLSolver preprocessor of SOWFA. This atmospheric precursor simulation captures more physics than synthetic turbulence—as illustrated in Figure 6—including atmospheric stability, wind-farm-wide turbulent length scales, and complex terrain effects. It is more computationally expensive than using the ambient

[14] Subscript n_r has been used here in place of (r)

[15] Note that the radial wake-velocity deficit at the centerline of the axisymmetric coordinate system ($n_r = 0$) is always zero ($V^{\text{Wake}}_{r_{np}}(r)|_{r=0} = 0$).

wind modeling options of *InflowWind*, but it is much less computationally expensive than a SOWFA simulation with multiple wind turbines present.

FAST.Farm requires ambient wind to be available in two different resolutions. Because wind will be spatially averaged across wake planes within the *AWAE* module, FAST.Farm needs a low-resolution wind domain (in both space and time) throughout the wind farm. The spatial resolution of the low-resolution domain—consisting of a structured 3D grid of wind data points—should be sufficient so that the spatial averaging is accurate, e.g., on the order of tens of meters for utility-scale wind turbines. The time step of the low-resolution domain dictates that of the FAST.Farm driver (Δt) and all FAST.Farm modules. It should therefore be consistent with the timescales of wake dynamics, e.g., on the order of seconds and smaller for higher mean wind speeds. Note that OpenFAST is subcycled within the *OF* module with a smaller time step. For accurate load calculation by OpenFAST, FAST.Farm also needs high-resolution wind domains (in both space and time) around each wind turbine and encompassing any turbine displacement. The spatial and time resolution of each high-resolution domain should be sufficient for accurate aerodynamic load calculations, e.g., on the order of the blade chord length and fractions of a second (Shaler et al. 2019b). The high-resolution domains overlap portions of the low-resolution domain. For simplicity of and to minimize computational expense within FAST.Farm, the time step of the high-resolution domain must be an integer divisor of the low-resolution domain time step.

When using ambient wind generated by a high-fidelity precursor simulation, the *AWAE* module reads in the three-component wind-velocity data across the high- and low-resolution domains—$\vec{V}^{\text{High}}_{\text{Amb}}$ for each turbine and $\vec{V}^{\text{Low}}_{\text{Amb}}$, respectively—that were computed by the high-fidelity solver within each time step. These values are stored in files for use in a given driver time step. The wind data files, including spatial discretizations, must be in VTK format and are specified by users of FAST.Farm at initialization. When using the *InflowWind* inflow option, the ambient wind across the high- and low-resolution domains are computed by calling the *InflowWind* module. In this case, the spatial discretizations of these domains are specified directly within the FAST.Farm primary input file. These wind data from the combined low- and high-resolution domains within a given driver time step represent the largest memory requirement of FAST.Farm.

After the ambient wind is processed at a given time step, the ambient wind submodel computes as output the rotor-disk-averaged ambient wind speed, normal to the disk, ${}^{\text{DiskAvg}}V^{\text{Wind}}_{x}$, for each turbine using Equation 6.30.

$$ {}^{\text{DiskAvg}}V^{\text{Wind}}_{x} = \left(\left\{ \hat{x}^{\text{Plane}}_{n_p} \right\}^T \left\{ \frac{1}{N^{\text{Polar}}_{n_p}} \sum_{n^{\text{Polar}}=1}^{N^{\text{Polar}}_{n_p}} \vec{V}^{\text{Low}}_{\text{Amb}_{n^{\text{Polar}}}} \right\} \right) \Bigg|_{n_p=0} \tag{6.30} $$

In Equation 6.30, $N^{\text{Polar}}_{n_p}$ is the number of points in a polar grid on wake plane n_p of the given wind turbine, n^{Polar} is the point counter such that $1 \leq n^{\text{Polar}} \leq N^{\text{Polar}}_{n_p}$ for wake plane n_p, and the equation is evaluated for the wake plane at the rotor disk ($n_p = 0$). The polar grid on wake plane n_p has a uniform radial and azimuthal discretization equal to the average *X-Y-Z* spatial discretization of the low-resolution domain (independent from the radial finite-difference grid used within the *WD* module) and a diameter of $C_{\text{Meander}} D^{\text{Wake}}_{n_p}$; C_{Meander} is discussed further in Section 6.2.5 below. Subscript n^{Polar} is appended to $\vec{V}^{\text{Low}}_{\text{Amb}}$ in Equation 6.30 to identify wind data that have been trilinearly interpolated from the low-resolution domain to the polar grid on the wake plane. Intuitively, Equation 6.30 states that the rotor-disk-averaged ambient wind speed, normal to the disk, for each turbine is calculated as the uniform spatial average of the ambient wind velocity on the wake plane at the rotor disk projected along the low-pass time-filtered rotor centerline.

The ambient wind submodel of the *AWAE* module also calculates as output the ambient turbulence intensity around each rotor, TI_{Amb}, using Equation 6.31:

$$ TI_{\text{Amb}} = \left(\frac{ \sqrt{ \frac{1}{3N^{\text{Polar}}_{n_p}} \sum_{n^{\text{Polar}}=1}^{N^{\text{Polar}}_{n_p}} \left\| \vec{V}^{\text{Low}}_{\text{Amb}_{n^{\text{Polar}}}} - \left\{ \frac{1}{N^{\text{Polar}}_{n_p}} \sum_{n^{\text{Polar}}=1}^{N^{\text{Polar}}_{n_p}} \vec{V}^{\text{Low}}_{\text{Amb}_{n^{\text{Polar}}}} \right\} \right\|_2^2 } }{ \left\| \left\{ \frac{1}{N^{\text{Polar}}_{n_p}} \sum_{n^{\text{Polar}}=1}^{N^{\text{Polar}}_{n_p}} \vec{V}^{\text{Low}}_{\text{Amb}_{n^{\text{Polar}}}} \right\} \right\|_2 } \right) \Bigg|_{n_p=0} \tag{6.31} $$

The bracketed term in Equation 6.31 is the same as in Equation 6.30, representing the uniform spatial average of the ambient wind velocity on the wake plane at the rotor disk. In contrast to the common definition of turbulence intensity used in the wind industry, which consists of a time-averaged quantity of the axial wind component, the turbulence intensity calculated in the ambient wind submodel of the *AWAE* module is based on a uniform spatial average of the three vector components. Not using time averaging ensures that only ambient wind at the current time step needs to be processed, which decreases memory requirements. Moreover, any time variation in the spatial average is moderated by the low-pass time filter in the *WD* module. Using spatial averaging and the three vector components allows for atmospheric shear, wind veer, and other ambient wind characteristics to influence the eddy viscosity and wake-deficit evolution in the *WD* module. The incorporation of wake-added turbulence is left for future work. Note that Equation 6.31 uses the eight wind data points from the low-resolution domain surrounding each point in the polar grid rather than interpolation. This is because calculating wind data in the polar grid on the wake plane via trilinear interpolation from the low-resolution domain would smooth out spatial variations and artificially reduce the calculated turbulence intensity.

Wake Merging

In previous implementations of DWM, the wind turbine and wake dynamics were solved individually or serially, not considering two-way wake-merging interactions. Additionally, there was no method available to calculate the disturbed wind in zones of wake overlap. Wake merging is illustrated by the SOWFA simulation of Figure 7. In FAST.Farm, the wake-merging submodel of the *AWAE* module identifies zones of wake overlap between all wakes across the wind farm by finding wake volumes that overlap in space. Wake deficits are superimposed in the axial direction based on the RSS method (Katič et al. 1986); transverse components (radial wake deficits) are superimposed by vector sum. In Katič et al. (Katič et al. 1986), the RSS method is applied to wakes with axial deficits that are uniform across the wake diameter and radial deficits are not considered. In contrast, the RSS method in FAST.Farm is applied locally at a given wind data point. The RSS method assumes that the local kinetic energy of the axial deficit in a merged wake equals the sum of the local energies of the axial deficits for each wake at the given wind data point. The RSS method only applies to an array of scalars, which works well for axial deficits because overlapping wakes likely have similar axial directions. This means, however, that only the magnitude of the vector is important in the superposition. A vector sum is applied to the transverse components (radial wake deficits) because any given radial direction is dependent on the azimuth angle in the axisymmetric coordinate system.

The disturbed (ambient plus wakes) wind velocities across the high- and low-resolution domains—$\vec{V}_{\text{Dist}}^{\text{High}}$ for each turbine and $\vec{V}_{\text{Dist}}^{\text{Low}}$, respectively—are computed using Equations 6.32 and 6.33, respectively.

$$\begin{aligned}\vec{V}_{\text{Dist}}^{\text{High}} &= \vec{V}_{\text{Amb}}^{\text{High}} \\ &- \left\{ \sqrt{\sum_{n^{\text{Wake}}=1}^{N^{\text{Wake}}} \begin{cases} \left(\left\{ \hat{x}^{Plane} \right\}^T \left\{ V_{x_{n^{\text{Wake}}}}^{\text{Wake}} \hat{x}_{n^{\text{Wake}}}^{\text{Plane}} + V_{r_{n^{\text{Wake}}}}^{\text{Wake}} \hat{r}_{n^{\text{Wake}}}^{\text{Plane}} \right\} \right)^2 & for \left(n_{t_{n^{\text{Wake}}}} \neq n_t \right) \\ 0 & otherwise \end{cases}} \right\} \hat{x}^{\text{Plane}} \\ &+ \sum_{n^{\text{Wake}}=1}^{N^{\text{Wake}}} \begin{cases} \left[I - \hat{x}^{\text{Plane}} \left\{ \hat{x}^{\text{Plane}} \right\}^T \right] \left\{ V_{x_{n^{\text{Wake}}}}^{\text{Wake}} \hat{x}_{n^{\text{Wake}}}^{\text{Plane}} + V_{r_{n^{\text{Wake}}}}^{\text{Wake}} \hat{r}_{n^{\text{Wake}}}^{\text{Plane}} \right\} & for \left(n_{t_{n^{\text{Wake}}}} \neq n_t \right) \\ \vec{0} & otherwise \end{cases}\end{aligned} \tag{6.32}$$

$$\begin{aligned}\vec{V}_{\text{Dist}}^{\text{Low}} &= \vec{V}_{\text{Amb}}^{\text{Low}} \\ &- \left\{ \sqrt{\sum_{n^{\text{Wake}}=1}^{N^{\text{Wake}}} \left(\left\{ \hat{x}^{\text{Plane}} \right\}^T \left\{ V_{x_{n^{\text{Wake}}}}^{\text{Wake}} \hat{x}_{n^{\text{Wake}}}^{\text{Plane}} + V_{r_{n^{\text{Wake}}}}^{\text{Wake}} \hat{r}_{n^{\text{Wake}}}^{\text{Plane}} \right\} \right)^2} \right\} \hat{x}^{\text{Plane}} \\ &+ \sum_{n^{\text{Wake}}=1}^{N^{\text{Wake}}} \left[I - \hat{x}^{\text{Plane}} \left\{ \hat{x}^{\text{Plane}} \right\}^T \right] \left\{ V_{x_{n^{\text{Wake}}}}^{\text{Wake}} \hat{x}_{n^{\text{Wake}}}^{\text{Plane}} + V_{r_{n^{\text{Wake}}}}^{\text{Wake}} \hat{r}_{n^{\text{Wake}}}^{\text{Plane}} \right\}\end{aligned} \tag{6.33}$$

Here, $(n_{t_{n^{\text{Wake}}}} \neq n_t)$ signifies that wake n^{Wake} is not associated with the given turbine n_t. The first, second, and third terms on the right-hand side of Equations 6.32 and 6.33 represent the ambient wind velocity, the RSS superposition of the axial wake-velocity deficits, and the vector sum of the transverse wake-velocity deficits, respectively. Although many mathematical details are outside the scope of this paper, the nomenclature of Equations 6.32 and 6.33 is as follows:

- N^{Wake}—number of wake volumes overlapping a given wind data point in the wind domain
- n^{Wake}—wake counter such that $1 \leq n^{\text{Wake}} \leq N^{\text{Wake}}$ which, when used as a subscript, is used to identify the specific point in a wake plane in place of (r) and subscript n_p
- $V^{\text{Wake}}_{x_{n^{\text{Wake}}}}$—axial wake-velocity deficit associated with where the given wind data point lies within the specific wake volume and corresponding wake plane
- $V^{\text{Wake}}_{r_{n^{\text{Wake}}}}$—radial wake-velocity deficit associated with where the given wind data point lies within the specific wake volume and corresponding wake plane
- $\hat{x}^{\text{Plane}}_{n^{\text{Wake}}}$—axial orientation associated with where the given wind data point lies within the specific wake volume and corresponding wake plane
- $\hat{r}^{\text{Plane}}_{n^{\text{Wake}}}$—radial unit vector associated with where the given wind data point lies within the specific wake volume and corresponding wake plane
- $\bar{\hat{x}}^{\text{Plane}}$—weighted-average axial orientation associated with a given point in the wind spatial domain
- $\{\bar{\hat{x}}^{\text{Plane}}\}^T$—projects $\{V^{\text{Wake}}_{x_{n^{\text{Wake}}}} \hat{x}^{\text{Plane}}_{n^{\text{Wake}}} + V^{\text{Wake}}_{r_{n^{\text{Wake}}}} \hat{r}^{\text{Plane}}_{n^{\text{Wake}}}\}$ along $\hat{r}^{\text{Plane}}_{n^{\text{Wake}}}$
- $\left[I - \hat{x}^{\text{Plane}}_{n^{\text{Wake}}} \{\bar{\hat{x}}^{\text{Plane}}\}^T\right]$—calculates the transverse component of $\{V^{\text{Wake}}_{x_{n^{\text{Wake}}}} \hat{x}^{\text{Plane}}_{n^{\text{Wake}}} + V^{\text{Wake}}_{r_{n^{\text{Wake}}}} \hat{r}^{\text{Plane}}_{n^{\text{Wake}}}\}$ normal to $\bar{\hat{x}}^{\text{Plane}}$.

Wake volumes are found by looping through all points, turbines, and wake planes and spatially determining if the given point resides in a wake volume that has a diameter equal to the radial extent of the wake planes. Wake volume n_p (for $0 \leq n_p \leq N_p - 2$) starts at wake plane n_p and extends to wake plane $n_p + 1$. Wake volumes have a centerline determined by $\vec{p}^{\text{Plane}}_{n_p}$, $\hat{x}^{\text{Plane}}_{n_p}$, $\vec{p}^{\text{Plane}}_{n_p+1}$, and $\hat{x}^{\text{Plane}}_{n_p+1}$—this centerline is curved if $\hat{x}^{\text{Plane}}_{n_p}$ and $\hat{x}^{\text{Plane}}_{n_p+1}$ are not parallel. The calculations of $V^{\text{Wake}}_{x_{n^{\text{Wake}}}}$ and $V^{\text{Wake}}_{r_{n^{\text{Wake}}}}$ involve bilinear interpolation of the wake deficits in the axial and radial directions. The axial interpolation is complicated when the adjacent wake planes are not parallel. The vector quantity $\{V^{\text{Wake}}_{x_{n^{\text{Wake}}}} \hat{x}^{\text{Plane}}_{n^{\text{Wake}}} + V^{\text{Wake}}_{r_{n^{\text{Wake}}}} \hat{r}^{\text{Plane}}_{n^{\text{Wake}}}\}$ represents the total wake-velocity deficit associated with where the given wind data point lies within the specific wake volume and corresponding wake plane. Because each wake plane may have a unique orientation, what constitutes "axial" and "radial" in the superposition at a given wind data point is determined by weighted-averaging the orientations of each wake volume overlapping that point (weighted by the magnitude of each axial wake deficit). A similar equation is used to calculate the distributed wind velocities across the high-resolution domain ($\vec{V}^{\text{High}}_{\text{Dist}}$) for each turbine, which is needed to calculate the disturbed wind inflow to a turbine. Note that for the high-resolution domain, a turbine is prevented from interacting with its own wake.

Once the distributed wind velocities across the low-resolution domain have been found, the wake merging submodel of the *AWAE* module computes as output the advection, deflection, and meandering velocity of each wake plane, $\vec{V}^{\text{Plane}}_{n_p}$ for $0 \leq n_p \leq N_p - 1$, for each turbine as the weighted spatial average of the disturbed wind velocity across the wake plane, using Equation 6.34.

$$\vec{V}^{\text{Plane}}_{n_p} = \frac{\sum\limits_{n^{\text{Polar}}=1}^{N^{\text{Polar}}_{n_p}} w_{n^{\text{Polar}}} \vec{V}^{\text{Low}}_{\text{Dist}_{n^{\text{Polar}}}}}{\sum\limits_{n^{\text{Polar}}=1}^{N^{\text{Polar}}_{n_p}} w_{n^{\text{Polar}}}} \tag{6.34}$$

The polar grid on wake plane n_p has a uniform radial and azimuthal discretization equal to the average *X-Y-Z* spatial discretization of the low-resolution domain (independent from the radial finite-difference grid used within the *WD* module) and a local diameter described below. Subscript n^{Polar} is appended to $\vec{V}^{\text{Low}}_{\text{Dist}}$ in Equation 6.34 to identify wind data that have been trilinearly interpolated from the low-resolution domain to the polar grid on the wake plane. Unlike Equation 6.30, Equation 6.34 includes a spatial weighting factor, $w_{n^{\text{Polar}}}$, dependent on the radial distance of point n^{Polar} from the center of the wake plane (discussed below). FAST.Farm will issue a warning if the center of any wake plane has left the boundaries of the low-resolution domain and set the meandering velocity of each wake plane, $\vec{V}_{n_p}^{\text{Plane}}$, to zero for any wake plane that has entirely left the boundaries of the low-resolution domain. Qualitatively, Equation 6.34 states that the advection, deflection, and meandering velocity of each wake plane for each turbine is calculated as the weighted spatial average of the disturbed wind velocity on the wake plane. Larsen et al. (Larsen et al. 2008) proposed a uniform spatial average where all points within a circle of diameter $2D_{n_p}^{\text{Wake}}$ are given equal weight. However, the Fourier transform of the circular function in a polar spatial domain results in a *jinc* function in the polar wave-number domain,[16] implying a gentle roll-off of energy below the cutoff wave number and pockets of energy at distinct wave numbers above the cutoff wave number. Experience with FAST.Farm development has shown that this approach results in less overall wake meandering and at improper frequencies. As such, three weighted spatial averaging methods have been implemented in FAST.Farm, as defined in Equation 6.35.

$$w_{n^{\text{Polar}}} = \begin{cases} 1 & \text{for method 1—uniform} \\ jinc\left(\frac{r_{n^{\text{Polar}}}}{C_{\text{Meander}} D^{\text{Wake}}}\right) & \text{for method 2—truncated jinc} \\ jinc\left(\frac{r_{n^{\text{Polar}}}}{C_{\text{Meander}} D^{\text{Wake}}}\right) jinc\left(\frac{r_{n^{\text{Polar}}}}{2C_{\text{Meander}} D^{\text{Wake}}}\right) & \text{for method 3—windowed jinc} \end{cases} \tag{6.35}$$

The first method is a spatial average with a uniform weighting with a local polar-grid diameter of $C_{\text{Meander}} D_{n_p}^{\text{Wake}}$ at wake plane n_p, resulting in a cutoff wave number of $\frac{1}{C_{\text{Meander}} D^{\text{Wake}}}$. The second and third methods weight each point in the spatial average by a form of the *jinc* function dependent on the radius of the point from the wake centerline, $r_{n^{\text{Polar}}}$, normalized by $C_{\text{Meander}} D^{\text{Wake}}$. This results in a more ideal low-pass filter with a sharper cutoff of energy in the polar wave-number domain with a cutoff wave number of $\frac{1}{C_{\text{Meander}} D^{\text{Wake}}}$. However, because the *jinc* function decays slowly with increasing argument, the *jinc* function must be windowed to be applied in practice. The second method truncates the *jinc* function at its first zero crossing, corresponding to a local polar-grid diameter of $1.21967 C_{\text{Meander}} D_{n_p}^{\text{Wake}}$ at wake plane n_p. The third method windows the *jinc* function by multiplying it with a *jinc* function of half the argument (the polar-domain equivalent of a one-dimensional Lanczos/sinc window), which tapers the weighting to zero at its second zero crossing (the weighting is positive below the first zero crossing and negative past the first zero crossing until it tapers to zero). This corresponds to a local polar-grid diameter of $2.23313 C_{\text{Meander}} D_{n_p}^{\text{Wake}}$ at wake plane n_p. These weighted spatial averaging methods improve the overall level and frequency content of the wake meandering at the expense of a bit heavier computations due to the larger polar-grid diameters (i.e., the truncated *jinc* method has roughly 50% more points within the polar grid than the uniform method, and the windowed *jinc* method has roughly five times more points than the uniform method). A value of $C_{\text{Meander}} = 2$, resulting in a polar-grid diameter of $2D^{\text{Wake}}$ and cutoff wave number of $\frac{1}{2D^{\text{Wake}}}$, follows the characteristic dimension important to transverse wake meandering proposed by Larsen et al. (Larsen et al. 2008) C_{Meander} is included in all methods to enable the user of FAST.Farm to better match the meandering to known solutions. Note that the lower the value of C_{Meander}, the more the wake will meander.

[16]In this context, the *jinc* function is defined as $jinc(r) = \frac{J_1(2\pi r)}{r}$ (with the limiting value at the origin of $jinc(0) = \pi$), where $J_1(r)$ is the Bessel function of the first kind and order one. The *jinc* function is normalized such that $\int_0^\infty jinc(r)\, 2\pi r dr = 1$. The *jinc* function is the polar-equivalent of the one-dimensional sinc function defined as $\text{sinc}(x) = \frac{\sin(\pi x)}{\pi x}$ (with the limiting value at the origin of $\text{sinc}(0) = 1$, which is the Fourier transform of a rectangular function, i.e., an ideal low-pass filter, and normalized such that $\int_{-\infty}^{\infty} \text{sinc}(x)\, dx = 1$.

7 Future Work

This list contains features that could be implemented in future releases:

- Develop more efficient methods of generating/processing ambient wind from a high-fidelity precursor simulation, including:
  - Propagate 2D planes of ambient wind data using Taylor's frozen turbulence hypothesis as an alternative to 3D volumes
  - Allow for nonuniform grids in Turbsim
  - Use Dynamic Mode Decomposition to compress the file size of the low-resolution domains
  - Implement Gabor mode enrichment to replace the high-resolution domains
  - Develop a more efficient ABLSolver based on a simple rectangular (rather than a generally unstructured) grid.
- Improve the eddy-viscosity formulation with additional physics.
- Pursue additional wake-modeling approaches, including:
  - Introduce simpler wake-deficit models, e.g., the Gaussian wake model by Bastankhah and Porté-Agel and the super-Gaussian model by Blondel and Cathelain
  - Introduce simpler wake-deflection models, e.g., the model by Jiménez or the model by Qian and Ishihara
  - Apply a free-vortex method for the near wake
  - Incorporate a kidney-shaped wake under skewed-flow conditions, e.g., by incorporating opposing vortices from the skew-induced horseshoe vortex
  - Deform the base-wake deficit (introduce asymmetry) as a result of background turbulence (in addition to wake meandering)
  - Incorporate wake-added turbulence
  - Improve the treatment of complex terrain (beyond specifying ambient wind data as NaN in VTK format)
  - Include wakes from the nacelle and support structure
  - Reflect wakes off of the ground.
- Address deep-array effects for large wind farms and account for flow speedup around the edges of the wind farm – i.e., account for the wind-farm blockage effect – e.g., by mimicking the wind farm-induced boundary layer with surface roughness in the LES ambient wind precursor.
- Implement a model to mimic the measurements taken from a LIDAR and other remote sensing technologies.
- Incorporate MPI to support the modeling of large wind farms by taking advantage of memory parallelization and parallelization between nodes of an HPC.
- Allow for a more general module form, e.g.:
  - Support continuous states
  - Support direct feedthrough of input to output
  - Support full-system linearization.
- Support an interface to Simulink for super and individual wind turbine controllers.
- Implement checkpoint-restart capability.

- Enable binary wind data input and output formats and binary time-series results output format.
- Add ability to output disturbed wind in VTK format on 2D slices that need not be parallel to the *X-Y, Y-Z* and/or *X-Z* planes of the global inertial-frame coordinate system.
- Rename the ambient wind data input files in VTK format following the naming convention used for the FAST.Farm-generated visualization output files in VTK format (with leading zeros and without the *t*).
- Support super controller-, inflow-, and wake-related output channels for more than the first 9 wind turbines in the wind farm.
- Interface FAST.Farm to the Wind-Plant Integrated System Design & Engineering Model (WISDEM™) for systems-engineering applications (multidisciplinary design, analysis, and optimization; uncertainty quantification; and so on).
- Develop a wrapper for stand-alone AeroDyn – the aerodynamics module of OpenFAST (or an equivalent BEM tool) – as an alternative to OpenFAST to support advanced performance-only wind-farm analysis that is much more computationally efficient than FAST.Farm analysis using OpenFAST.
- Address unique offshore wind energy challenges, e.g.:
  - Ensure consistent waves across an offshore wind farm
  - Support the air-water interface
  - Consider shared mooring and anchoring arrangements (for floating offshore wind farms).
- Adopt the capability to support undersea marine turbine arrays (which may require supporting direct feedthrough of input to output to handle the added-mass effects).

Bibliography

Ainslie, J. F. 1988. "Calculating the Flowfield in the Wake of Wind Turbines". *Journal of Wind Engineering and Industrial Aerodynamics* 27:213–224. doi:https://doi.org/10.1016/0167-6105(88)90037-2.

Churchfield, M. J., et al. 2015. "A Comparison of the Dynamic Wake Meandering Model, Large-Eddy Simulations, and Field Data at the Egmond aan Zee Offshore Wind Plant". 33rd Wind Energy Symposium. Kissimmee, FL: AIAA. doi:http://dx.doi.org/10.2514/6.2015-0724.

Crank, J., and P. Nicolson. 1996. "A Practical Method for Numerical Evaluation of Solutions of Partial Differencial Equations of the Heat-Conduction Type". *Advances in Computaional Mathematics* 6:207–226.

Doubrawa, P., et al. 2018. "Optimization-Based Calibration of FAST.Farm Parameters Against SOWFA". 36th Wind Energy Symposium. Kissimmee, FL: AIAA. doi:https://arc.aiaa.org/doi/pdf/10.2514/6.2018-0512.

Gebraad, P. M. O., and et al. 2016. "Wind Plant Power Optimization Through Yaw Control Using a Parametric Model for Wake Effects – a CFD Simulation Study". *Wind Energy* 19 (1): 95–114. doi:http://onlinelibrary.wiley.com/doi/10.1002/we.1822/epdf.

Hao, Y. 2016. "Wind Farm Wake Modeling and Analysis of Wake Impacts in a Wind Farm". Phd thesis, University of Massachusetts.

Hao, Y., et al. 2014. "Implementing the Dynamic Wake Meandering Model in the NWTC Design Codes". 32nd Wind Energy Symposium. National Harbor, MD: AIAA. doi:http://dx.doi.org/10.2514/6.2014-1089.

Jonkman, B. 2014. *TurbSim User's Guide v2.00.00*. Tech. rep. NREL/TP-xxxx-xxxxx. Golden, CO: National Renewable Energy Laboratory.

Jonkman, J. 2013. "The New Modularization Framework for the FAST wind Turbine CAE Tool". 51st AIAA Aerospace Sciences Meeting. Dallas, TX: AIAA.

Jonkman, J., et al. 2009. *Definition of a 5-MW Reference Wind Turbine for Offshore System Development*. Tech. rep. NREL/TP-500-38060. Golden, CO: National Renewable Energy Laboratory.

Katič, I., et al. 1986. "A Simple Model for Cluster Efficiency". European Wind Energy Association Conference and Exhibition. Rome, Italy.

Keck, R.-E., et al. 2013. "A Consistent Turbulence Formulation for the Dynamic Wake Meandering Model in the Atmospheric Boundary Layer". Phd thesis, DTU.

Larsen, G. C., et al. 2008. "Wake Meander: A Pragmatic Approach". *Wind Energy* 11:337–95. doi:http://onlinelibrary.wiley.com/doi/10.1002/we.267/epdf.

Madsen, H. A., et al. 2010. "Calibration and Validation of the Dynamic Wake Meandering Model for Implementation in an Aeroelastic Code". *Journal of Solar Energy Engineering* 132 (4). doi:https://doi.org/10.1115/1.4002555.

– . 2016. "Wake Flow Characteristics at High Wind Speed." 34th Wind Energy Symposium. San Diego, CA: AIAA. doi:http://dx.doi.org/10.2514/6.2016-1522..

Martinez-Tossas, L. A. 2021. "Wind Turbine Wakes: High-Thrust Coefficient". Publication pending, *Wind Energy*.

Shaler, K., et al. 2019a. "Effects of Inflow Spatiotemporal Discretization on Wake Meandering and Turbine Structural Response Using FAST.Farm". *Journal of Physics: Conference Series* 1256. doi:10.1088/1742-6596/1256/1/012023.

– . 2019b. "FAST.Farm Response of Varying Wind Inflow Techniques". 37th Wind Energy Symposium. San Diego, CA: AIAA. doi:https://arc.aiaa.org/doi/pdf/10.2514/6.2019-2086.

Smith, S. W. 2006. *The Scientist and Engineer's Guide to Digital Signal Processing*. Californial Technical Publishing. ISBN: 978-0966017632.

Sprague, M. A., et al. 2014. "FAST Modular Wind Turbine CAE Tool: Nonmatching Spatial and Temporal Meshes". 50th AIAA Aerospace Sciences Meeting. National Harbor, MD: AIAA. doi:http://arc.aiaa.org/doi/pdf/10.2514/6.2014-0520.

– . 2015. "FAST Modular Framework for Wind Turbine Simulation: New Algorithms and Numerical Examples". 51th AIAA Aerospace Sciences Meeting. Kissimmee, FL: AIAA. doi:http://arc.aiaa.org/doi/pdf/10.2514/6.2014-0520.

Thomas, L. H. 1949. *Elliptic Problems in Linear Difference Equations Over a Network*. Tech. rep. New York, NY: Watson Science Computer Laboratory.

A FAST.Farm Primary Input File

The symbol ↪ in the following text indicates a continuation of the previous line, it should not be implemented as a new line in the actual input file. When a default value is available, DEFAULT may be used instead of the value.

```
FAST.Farm v1.00.* INPUT FILE
Sample FAST.Farm input file
--- SIMULATION CONTROL ---
False       Echo          Echo input data to <RootName>.ech? (flag)
FATAL       AbortLevel    Error level when simulation should abort (string) {"WARNING", "SEVERE",
                           ↪ "FATAL"}
2000.0      TMax          Total run time (s) [>=0.0]
False       UseSC         Use a super controller? (flag)
1           Mod_AmbWind   Ambient wind model (-) (switch) {1: high-fidelity precursor in VTK
                           ↪ format, 2: one InflowWind module, 3: multiple instances of
                           ↪ InflowWind module}
--- SUPER CONTROLLER --- [used only for UseSC=True]
"SC_DLL.dll" SC_FileName Name/location of the dynamic library {.dll [Windows] or .so [Linux]}
                           ↪ containing the Super Controller algorithms (quoted string)
--- AMBIENT WIND: PRECURSOR IN VTK FORMAT --- [used only for Mod_AmbWind=1]
2.0         DT_Low-VTK    Time step for low-resolution wind data input files; will be used as the
                           ↪ global FAST.Farm time step (s) [>0.0]
0.5         DT_High-VTK   Time step for high-resolution wind data input files (s) [>0.0]
                           ↪ "/AmbWind/steady" WindFilePath Path name to wind data files from
                           ↪ precursor (string)
False       ChkWndFiles   Check all the ambient wind files for data consistency (flag)
--- AMBIENT WIND: INFLOWWIND MODULE --- [used only for Mod_AmbWind=2 or 3]
2.0         DT_Low        Time step for low-resolution wind data interpolation; will be used as
                           ↪ the global FAST.Farm time step (s) [>0.0]
0.5         DT_High       Time step for high-resolution wind data interpolation (s) [>0.0]
300         NX_Low        Number of low-resolution spatial nodes in X direction for wind data
                           ↪ interpolation (-) [>=2]
300         NY_Low        Number of low-resolution spatial nodes in Y direction for wind data
                           ↪ interpolation (-) [>=2]
35          NZ_Low        Number of low-resolution spatial nodes in Z direction for wind data
                           ↪ interpolation (-) [>=2]
5.0         X0_Low        Origin of low-resolution spatial nodes in X direction for wind data
                           ↪ interpolation (m)
5.0         Y0_Low        Origin of low-resolution spatial nodes in Y direction for wind data
                           ↪ interpolation (m)
5.0         Z0_Low        Origin of low-resolution spatial nodes in Z direction for wind data
                           ↪ interpolation (m)
10.0        dX_Low        Spacing of low-resolution spatial nodes in X direction for wind data
                           ↪ interpolation (m) [>0.0]
10.0        dY_Low        Spacing of low-resolution spatial nodes in Y direction for wind data
                           ↪ interpolation (m) [>0.0]
10.0        dZ_Low        Spacing of low-resolution spatial nodes in Z direction for wind data
                           ↪ interpolation (m) [>0.0]
16          NX_High       Number of high-resolution spatial nodes in X direction for wind data
                           ↪ interpolation (-) [>=2]
16          NY_High       Number of high-resolution spatial nodes in Y direction for wind data
                           ↪ interpolation (-) [>=2]
17          NZ_High       Number of high-resolution spatial nodes in Z direction for wind data
                           ↪ interpolation (-) [>=2]
"InflowWind.dat" InflowFile Name of file containing InflowWind module input parameters (quoted
                           ↪ string)
--- WIND TURBINES ---
1           NumTurbines   Number of wind turbines (-) [>=1] [last 6 columns below used only for
                           ↪ Mod_AmbWind=2 or 3]
WT_X     WT_Y      WT_Z    WT_FASTInFile        X0_High Y0_High  Z0_High dX_High dY_High dZ_High
(m)      (m)       (m)      (string)              (m)      (m)      (m)      (m)      (m)      (m)
605.0    1500.0    0.0     "/FAST/Test18.fst"    525.0    1425.0    5.0     10.0     10.0     10.0
--- WAKE DYNAMICS ---
5.0         dr            Radial increment of radial finite-difference grid (m) [>0.0]
40          NumRadii      Number of radii in the radial finite-difference grid (-) [>=2]
```

```
140         NumPlanes     Number of wake planes (-) [>=2]
DEFAULT     f_c           Cutoff (corner) frequency of the low-pass time-filter for the wake
                           ↪ advection, deflection, and meandering model (Hz) [>0.0] or DEFAULT
                           ↪ [=0.0007]
DEFAULT     C_HWkDfl_O    Calibrated parameter in the correction for wake deflection defining the
                           ↪ horizontal offset at the rotor (m) or DEFAULT [=0.0]
DEFAULT     C_HWkDfl_OY   Calibrated parameter in the correction for wake deflection defining the
                           ↪ horizontal offset at the rotor scaled with yaw error (m/deg) or
                           ↪ DEFAULT [=0.3]
DEFAULT     C_HWkDfl_x    Calibrated parameter in the correction for wake deflection defining the
                           ↪ horizontal offset scaled with downstream distance (-) or DEFAULT
                           ↪ [=0.0]
DEFAULT     C_HWkDfl_xY   Calibrated parameter in the correction for wake deflection defining the
                           ↪ horizontal offset scaled with downstream distance and yaw error
                           ↪ (1/deg) or DEFAULT [=-0.004]
DEFAULT     C_NearWake    Calibrated parameter for the near-wake correction (-) [>1. and <2.5] or
                           ↪ DEFAULT [=1.8]
DEFAULT     k_vAmb        Calibrated parameter for the influence of ambient turbulence in the
                           ↪ eddy viscosity (-) [>=0.0] or DEFAULT [=0.05]
DEFAULT     k_vShr        Calibrated parameter for the influence of the shear layer in the eddy
                           ↪ viscosity (-) [>=0.0] or DEFAULT [=0.016]
DEFAULT     C_vAmb_DMin   Calibrated parameter in the eddy viscosity filter function for ambient
                           ↪ turbulence defining the transitional diameter fraction between the
                           ↪ minimum and exponential regions (-) [>=0.0] or DEFAULT [=0.0]
DEFAULT     C_vAmb_DMax   Calibrated parameter in the eddy viscosity filter function for ambient
                           ↪ turbulence defining the transitional diameter fraction between the
                           ↪ exponential and maximum regions (-) [> C_vAmb_DMin] or DEFAULT
                           ↪ [=1.0]
DEFAULT     C_vAmb_FMin   Calibrated parameter in the eddy viscosity filter function for ambient
                           ↪ turbulence defining the value in the minimum region (-) [>=0.0 and
                           ↪ <=1.0] or DEFAULT [=1.0]
DEFAULT     C_vAmb_Exp    Calibrated parameter in the eddy viscosity filter function for ambient
                           ↪ turbulence defining the exponent in the exponential region (-) [>
                           ↪ 0.0] or DEFAULT [=0.01]
DEFAULT     C_vShr_DMin   Calibrated parameter in the eddy viscosity filter function for the
                           ↪ shear layer defining the transitional diameter fraction between the
                           ↪ minimum and exponential regions (-) [>=0.0] or DEFAULT [=3.0]
DEFAULT     C_vShr_DMax   Calibrated parameter in the eddy viscosity filter function for the
                           ↪ shear layer defining the transitional diameter fraction between the
                           ↪ exponential and maximum regions (-) [> C_vShr_DMin] or DEFAULT
                           ↪ [=25.0]
DEFAULT     C_vShr_FMin   Calibrated parameter in the eddy viscosity filter function for the
                           ↪ shear layer defining the value in the minimum region (-) [>=0.0 and
                           ↪ <=1.0] or DEFAULT [=0.2]
DEFAULT     C_vShr_Exp    Calibrated parameter in the eddy viscosity filter function for the
                           ↪ shear layer defining the exponent in the exponential region (-) [>
                           ↪ 0.0] or DEFAULT [=0.1]
DEFAULT     Mod_WakeDiam  Wake diameter calculation model (-) (switch) {1: rotor diameter, 2:
                           ↪ velocity based, 3: mass-flux based, 4: momentum-flux based} or
                           ↪ DEFAULT [=1]
DEFAULT     C_WakeDiam    Calibrated parameter for wake diameter calculation (-) [>0.0 and <0.99]
                           ↪ or DEFAULT [=0.95] [unused for Mod_WakeDiam=1]
DEFAULT     Mod_Meander   Spatial filter model for wake meandering (-) (switch) {1: uniform, 2:
                           ↪ truncated jinc, 3: windowed jinc} or DEFAULT [=3]
DEFAULT     C_Meander     Calibrated parameter for wake meandering (-) [>=1.0] or DEFAULT [=1.9]
--- VISUALIZATION ---
False       WrDisWind     Write low- and high-resolution disturbed wind data to
                           ↪ <RootName>.Low.Dis.t<n>.vtk etc. (flag)
1           NOutDisWindXY Number of XY planes for output of disturbed wind data across the
                           ↪ low-resolution domain to <RootName>.Low.DisXY<n_out>.t<n>.vtk (-)
                           ↪ [0 to 9]
90.0        OutDisWindZ   Z coordinates of XY planes for output of disturbed wind data across the
                           ↪ low-resolution domain (m) [1 to NOutDisWindXY] [unused for
                           ↪ NOutDisWindXY=0]
2           NOutDisWindYZ Number of YZ planes for output of disturbed wind data across the
                           ↪ low-resolution domain to <RootName>/Low.DisYZ<n_out>.t<n>.vtk (-)
```

```
                             ↪ [0 to 9]
600.0,978.0 OutDisWindX X coordinates of YZ planes for output of disturbed wind data across the
                             ↪ low-resolution domain (m) [1 to NOutDisWindYZ] [unused for
                             ↪ NOutDisWindYZ=0]
1           NOutDisWindXZ Number of XZ planes for output of disturbed wind data across the
                             ↪ low-resolution domain to <RootName>/Low.DisXZ<n_out>.t<n>.vtk (-)
                             ↪ [0 to 9]
1500.0      OutDisWindY   Y coordinates of XZ planes for output of disturbed wind data across the
                             ↪ low-resolution domain (m) [1 to NOutDisWindXZ] [unused for
                             ↪ NOutDisWindXZ=0]
4.0        WrDisDT       Time step for disturbed wind visualization output (s) [>0.0] or DEFAULT
                             ↪ [=DT_Low or DT_Low-VTK] [unused for WrDisWind=False and
                             ↪ NOutDisWindXY=NOutDisWindYZ=NOutDisWindXZ=0]
--- OUTPUT ---
True        SumPrint      Print summary data to <RootName>.sum? (flag)
99999.9     ChkptTime     Amount of time between creating checkpoint files for potential restart
                             ↪ (s) [>0.0]
200.0       TStart        Time to begin tabular output (s) [>=0.0]
1           OutFileFmt    Format for tabular (time-marching) output file (-) (switch) {1: text
                             ↪ file [<RootName>.out], 2: binary file [<RootName>.outb], 3: both}
True        TabDelim      Use tab delimiters in text tabular output file? (flag) {uses spaces if
                             ↪ False}
"ES10.3E2" OutFmt         Format used for text tabular output, excluding the time channel.
                             ↪ Resulting field should be 10 characters. (quoted string)
3           NOutRadii     Number of radial nodes for wake output for an individual rotor (-) [0
                             ↪ to 20]
0, 15, 39  OutRadii       List of radial nodes for wake output for an individual rotor (-)
2           NOutDist      Number of downstream distances for wake output for an individual rotor
                             ↪ (-) [1 to NOutRadii] [unused for NOutRadii=0] rotor (-) [0 to 9]
0.0, 378.0 OutDist        List of downstream distances for wake output for an individual rotor
                             ↪ (m) [1 to NOutDist] [unused for NOutDist =0]
1           NWindVel      Number of points for wind output (-) [0 to 9]
600.0       WindVelX      List of coordinates in the X direction for wind output (m) [1 to
                             ↪ NWindVel] [unused for NWindVel=0]
1500.0      WindVelY      List of coordinates in the Y direction for wind output (m) [1 to
                             ↪ NWindVel] [unused for NWindVel=0]
90.0        WindVelZ      List of coordinates in the Z direction for wind output (m) [1 to
                             ↪ NWindVel] [unused for NWindVel=0]
OutList The next line(s) contains a list of output parameters. (quoted string)
"RtAxsXT1, RtAxsYT1, RtAxsZT1"
"RtPosXT1, RtPosYT1, RtPosZT1"
"YawErrT1"
"TIAmbT1"
"CtT1N01, CtT1N02, CtT1N03, CtT1N04, CtT1N05"
"WkAxsXT1D1, WkAxsXT1D2, WkAxsXT1D3"
"WkAxsYT1D1, WkAxsYT1D2, WkAxsYT1D3"
"WkAxsZT1D1, WkAxsZT1D2, WkAxsZT1D3"
"WkPosXT1D1, WkPosXT1D2, WkPosXT1D3"
"WkPosYT1D1, WkPosYT1D2, WkPosYT1D3"
"WkPosZT1D1, WkPosZT1D2, WkPosZT1D3"
"WkDfVxT1N01D1, WkDfVxT1N02D1, WkDfVxT1N03D1, WkDfVxT1N04D1, WkDfVxT1N05D1"
"WkDfVxT1N01D2, WkDfVxT1N02D2, WkDfVxT1N03D2, WkDfVxT1N04D2, WkDfVxT1N05D2"
"WkDfVxT1N01D3, WkDfVxT1N02D3, WkDfVxT1N03D3, WkDfVxT1N04D3, WkDfVxT1N05D3"
"WkDfVrT1N01D1, WkDfVrT1N02D1, WkDfVrT1N03D1, WkDfVrT1N04D1, WkDfVrT1N05D1"
"WkDfVrT1N01D2, WkDfVrT1N02D2, WkDfVrT1N03D2, WkDfVrT1N04D2, WkDfVrT1N05D2"
"WkDfVrT1N01D3, WkDfVrT1N02D3, WkDfVrT1N03D3, WkDfVrT1N04D3, WkDfVrT1N05D3"
END of input file (the word "END" must appear in the first 3 columns of
this last OutList line)
```

B Ambient Wind File

```
 # vtk DataFile Version 3.0
Amb.coarse
ASCII
DATASET STRUCTURED_POINTS
DIMENSIONS 300 300 35
ORIGIN 5 5 5
SPACING 10 10 10
POINT_DATA 3150000
VECTORS Amb FLOAT
```

```
4.852      0.217      -0.009
5.092      0.213      -0.077
5.248      0.418      -0.060
5.280      0.794      -0.045
5.125      0.993      -0.046
5.067      0.799      -0.118
5.272      0.481      -0.242
5.624      0.410      -0.312
5.724      0.531      -0.266
5.267      0.430      -0.167
4.597      0.055      -0.109
4.204      -0.322     -0.069
4.248      -0.441     -0.055
4.569      -0.159     -0.171
4.793      0.259      -0.423
4.658      0.431      -0.647
4.287      0.395      -0.572
3.955      0.355      -0.208
3.849      0.252      0.233
3.938      0.099      0.534
```

[3,149,970 lines removed]

```
10.576     -0.273     0.244
10.910     -0.287     -0.051
11.207     -0.241     -0.349
11.393     -0.172     -0.524
11.513     -0.139     -0.528
11.581     -0.210     -0.411
11.647     -0.315     -0.219
11.516     -0.348     -0.039
11.514     -0.252     0.185
10.977     0.058      0.245
```

C List of Output Channels

This is a list of all possible output parameters available within FAST.Farm (except those that are available from OpenFAST, which are specified within the OpenFAST input file(s) and output separately for each turbine). The names are grouped by meaning, but can be ordered in the OUTPUTS section of the FAST.Farm primary input file as you see fit.

Tα refers to turbine α, where α is a one-digit number in the range [1,9], corresponding to row α in the wind turbine input table. If ***NumTurbines*** > 9, only values for the first 9 turbines can be output. Setting α > ***NumTurbines*** yields invalid output.

Inζ and Otζ refer to super-controller input and output ζ, respectively, where ζ is a one-digit number in the range [1,9], corresponding to element ζ in the input and output arguments of the super-controller source code. If there are more than 9 elements, only values for the first 9 inputs and outputs can be output. Setting ζ greater than the number of elements yields invalid output.

Nβ refers to radial output node β, where β is a two-digit number in the range [01,20], corresponding to entry β in the ***OutRadii*** list, where node β is at radius ***dr*** × ***OutRadii***[β]. Setting β > ***NOutRadii*** yields invalid output.

Wη refers to wind point η, where η is a one-digit number in the range [1,9], corresponding to entry η in the ***WindVelX***, ***WindVelY***, and ***WindVelZ*** lists. Setting η > ***NWindVel*** yields invalid output. Setting ***WindVelX***, ***WindVelY***, and ***WindVelZ*** outside the low-resolution wind domain also yields invalid output.

δ refers to the X, Y, or Z coordinate axis.

Dγ refers to downstream distance γ, where γ is a one-digit number in the range [1,9], corresponding to entry γ in the ***OutDist*** list. Setting γ > ***NOutDist*** yields invalid output. The output is also invalid if ***OutDist*** is a distance further downstream than the wake has been calculated or for any distance where the wake from the turbine has overlapped itself.

Table C.1. List of Available FAST.Farm Output Channels

Channel Name	Units	Description
Super Controller		
SCGblInζ	(user)	Global (turbine independent) super controller input ζ
SCTαInζ	(user)	Turbine-dependent super controller input ζ for turbine α
SCGblOtζ	(user)	Global (turbine independent) super controller output ζ
SCTαOtζ	(user)	Turbine-dependent super controller input ζ for turbine α
Wind Turbine and Inflow		
RtAxsδTα	(-)	Orientation of the rotor centerline for turbine α in the global coordinate system
RtPosδTα	(m)	Position of the rotor (hub) center for turbine α in the global coordinate system
RtDiamTα	(m)	Rotor diameter for turbine α
YawErrTα	(deg)	Nacelle-yaw error for turbine α
TIAmbTα	(%)	Ambient turbulence intensity of the wind at the rotor disk for turbine α. The ambient turbulence intensity is based on a spatial-average of the three vector components, instead of just the axial component.
RtVAmbTα	(m/s)	Rotor-disk-averaged ambient wind speed (normal to disk, not including structural motion, local induction or wakes from upstream turbines) for turbine α
RtVRelTα	(m/s)	Rotor-disk-averaged relative wind speed (normal to disk, including structural motion and wakes from upstream turbines, but not including local induction) for turbine α
CtTαNβ	(-)	Azimuthally averaged thrust force coefficient (normal to disk) for radial output node β of turbine α
Wake (for an Individual Rotor)		
WkAxsδTαDγ	(-)	Orientation of the wake centerline for downstream distance γ of turbine α in the global coordinate system
WkPosδTαDγ	(m)	Center position of the wake centerline for downstream distance γ of turbine α in the global coordinate system
WkVelδTαDγ	(m/s)	Advection, deflection, and meandering velocity (not including the horizontal wake-deflection correction or low-pass time-filtering) of the wake for downstream distance γ of turbine α in the global coordinate system
WkDiamTαDγ	(m)	Wake diameter for downstream distance γ of turbine α
WkDfVxTαNβDγ	(m/s)	Axial wake velocity deficits for radial output node β and downstream distance γ of turbine α
WkDfVrTαNβDγ	(m/s)	Radial wake velocity deficits for radial output node β and downstream distance γ of turbine α
EddVisTαNβDγ	(m^2/s)	Total eddy viscosity for radial output node β and downstream distance γ of turbine α
EddAmbTαNβDγ	(m^2/s)	Individual contribution to the eddy viscosity from ambient turbulence for radial output node β and downstream distance γ of turbine α
EddShrTαNβDγ	(m^2/s)	Individual contributions to the eddy viscosity from the shear layer for radial output node β and downstream distance γ of turbine α
Ambient Wind and Array Effects		
WηVAmbδ	(m/s)	Ambient wind velocity (not including wakes) for point η in the global coordinate system (from the low-resolution domain)
WηVDisδ	(m/s)	Disturbed wind velocity (including wakes) for point η in the global coordinate system (from the low-resolution domain)