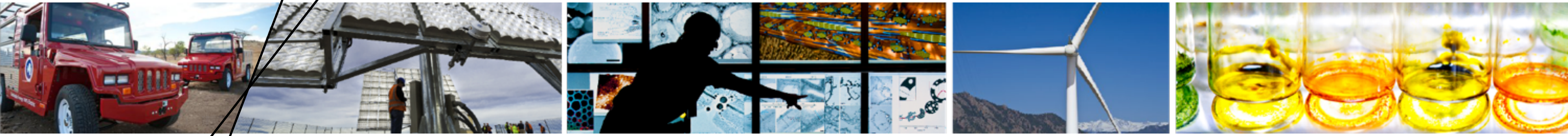
Simulator fOr Wind Farm Applications (SOWFA)

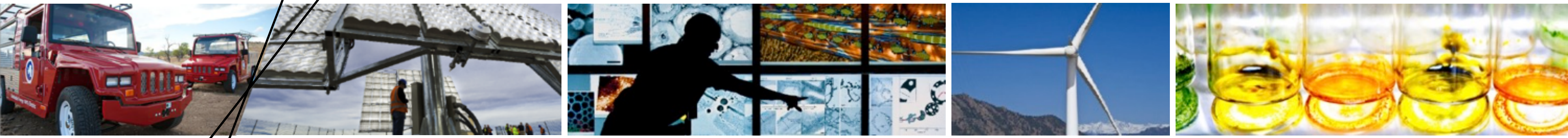


NAWEA 2017 Training Session

Ames, Iowa

Matthew Churchfield

25 September 2017



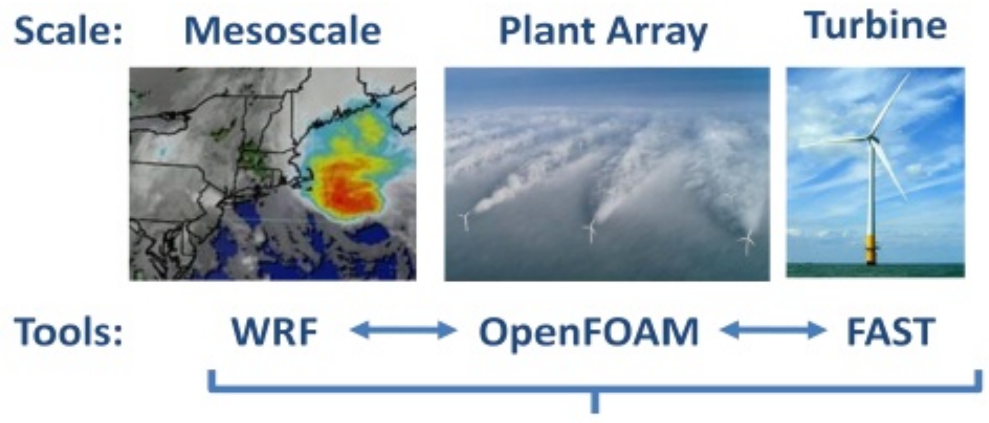
Overview

Overview of SOWFA

- Simulator fOr Wind Farm Applications
- Currently, it is composed of CFD tools based on OpenFOAM coupled with a NREL's FAST wind turbine structural/system dynamics model
- It is meant to be modular and open-source so that others can put in their own "modules"
- Open-source and freely available
- It can be downloaded at: https://github.com/NREL/SOWFA

Overview of SOWFA

- Simulator fOr Wind Farm Applications
- The overall vision is:



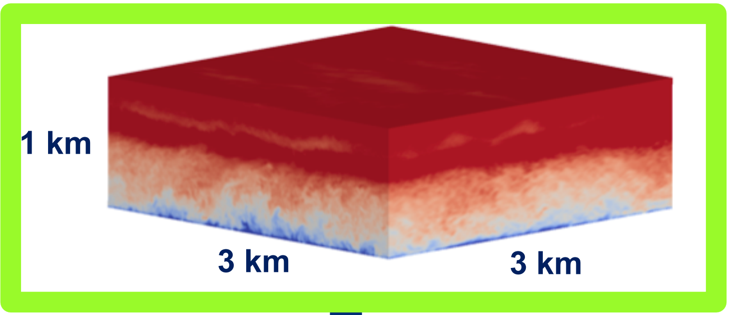
SOWFA

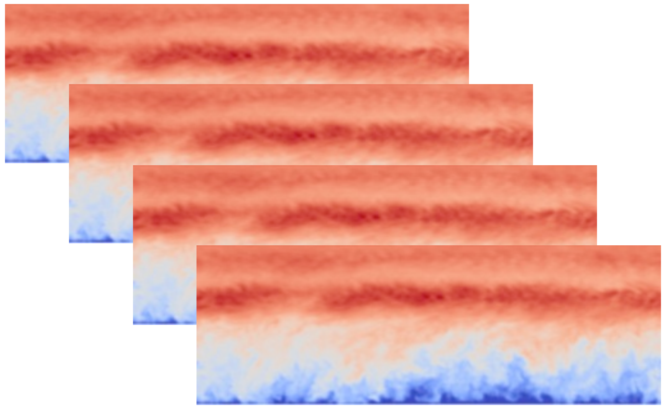
Simulation Method

"Precursor" atmospheric large-eddy simulation

OR

Mesoscale weather model input



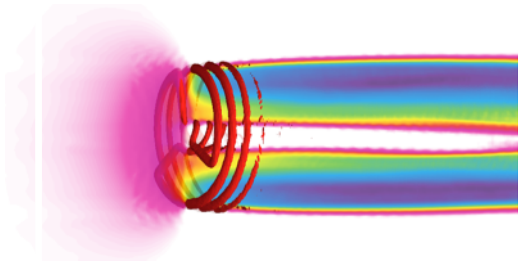


Save planes of data every N time steps

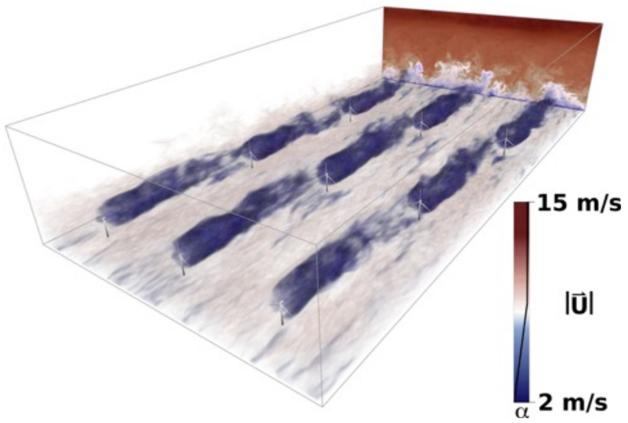
Initialize wind farm domain with precursor volume field

Use saved precursor data as inflow boundary conditions

Actuator line turbine aerodynamics models (coupled with NREL's FAST turbine dynamics model)



Wind farm large-eddy simulation



Simulation Method: Inflow

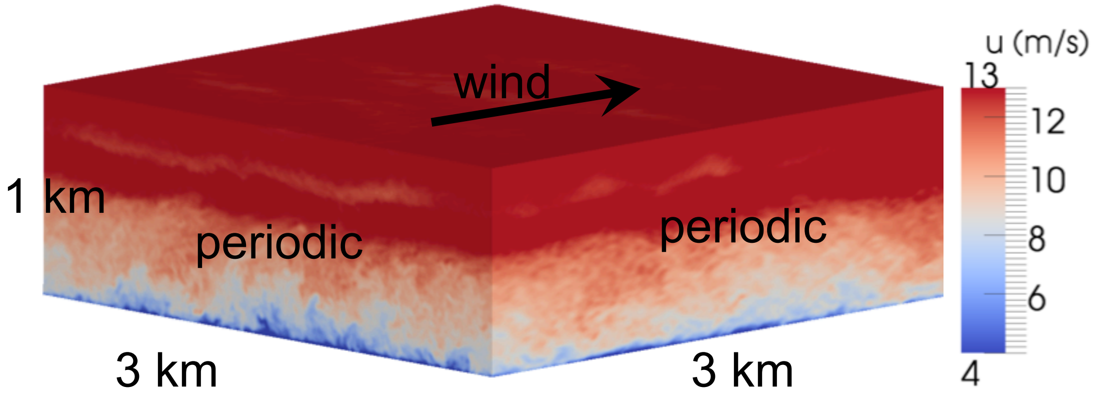
Precursor Atmospheric LES

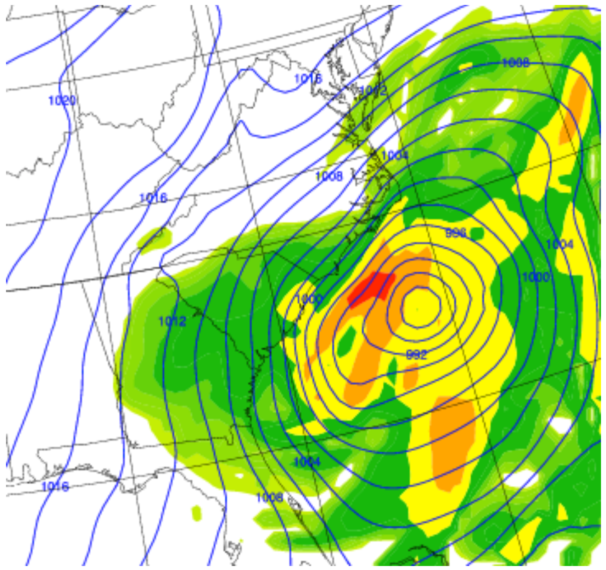
- Canonical, horizontally homogeneous
- Naturally generates realistic turbulence
- Atmospheric stability and Coriolis
- Idealized, not always the case in the real world

OR

Mesoscale Model Input

- Non-homogeneous
- Assimilated with real observations
- Downscaling problem —"spin-up" of turbulence



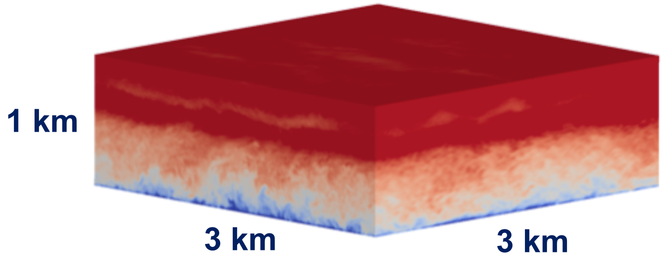


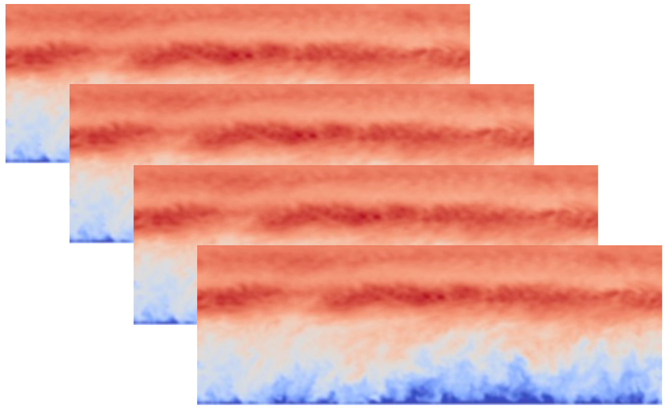
Simulation Method

"Precursor" atmospheric large-eddy simulation

OR

Mesoscale weather model input



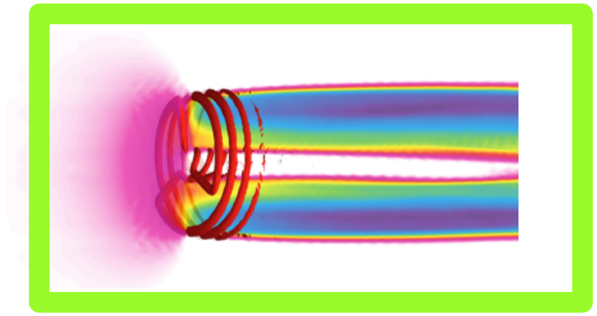


Save planes of data every N time steps

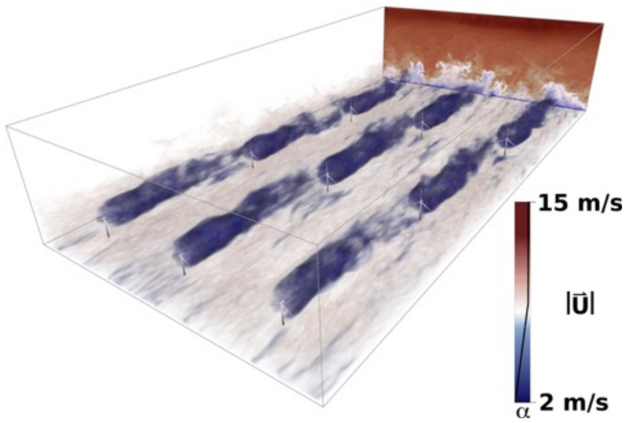
Initialize wind farm domain with precursor volume field

Use saved precursor data as inflow boundary conditions

Actuator line turbine aerodynamics models (coupled with NREL's FAST turbine dynamics model)

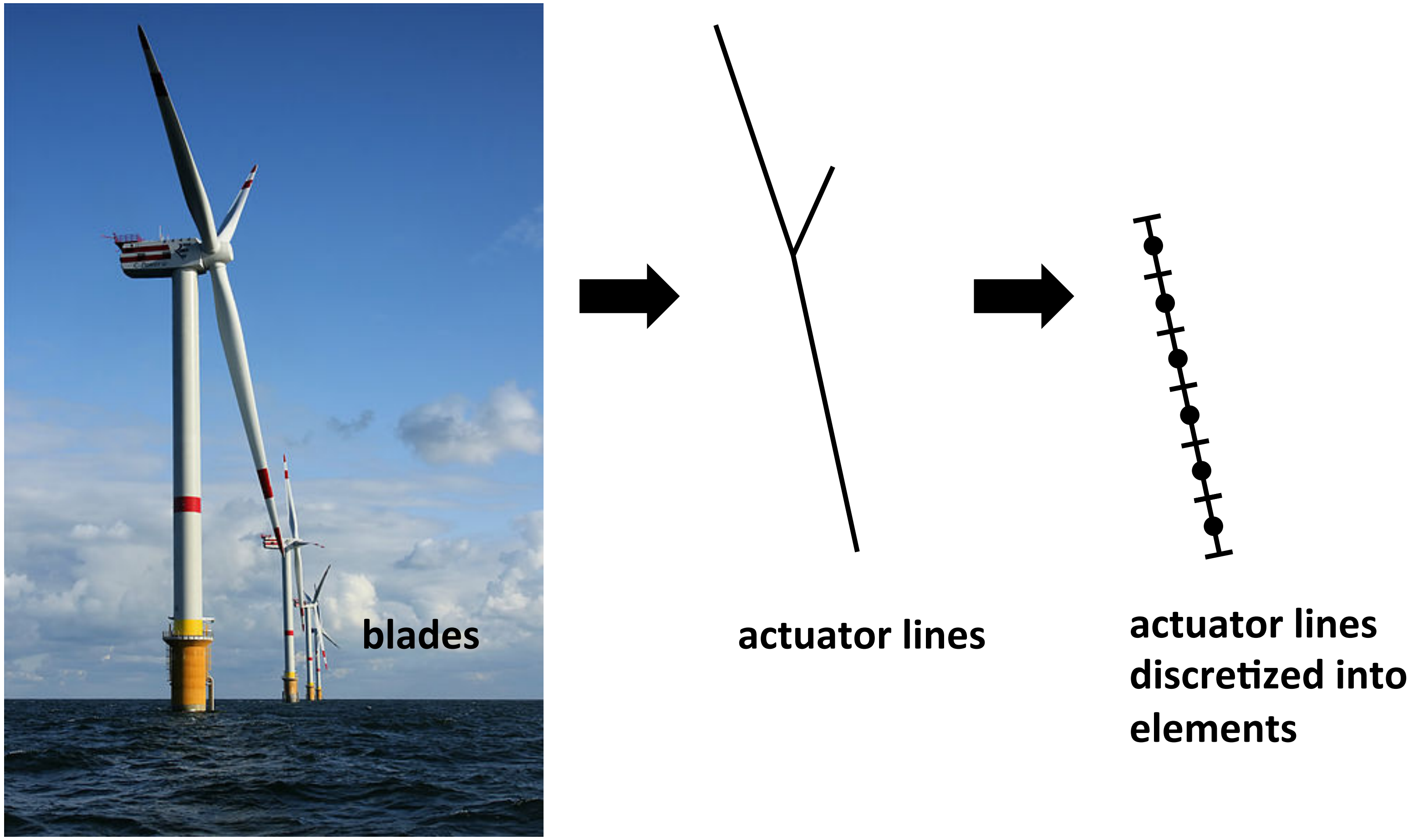


Wind farm large-eddy simulation

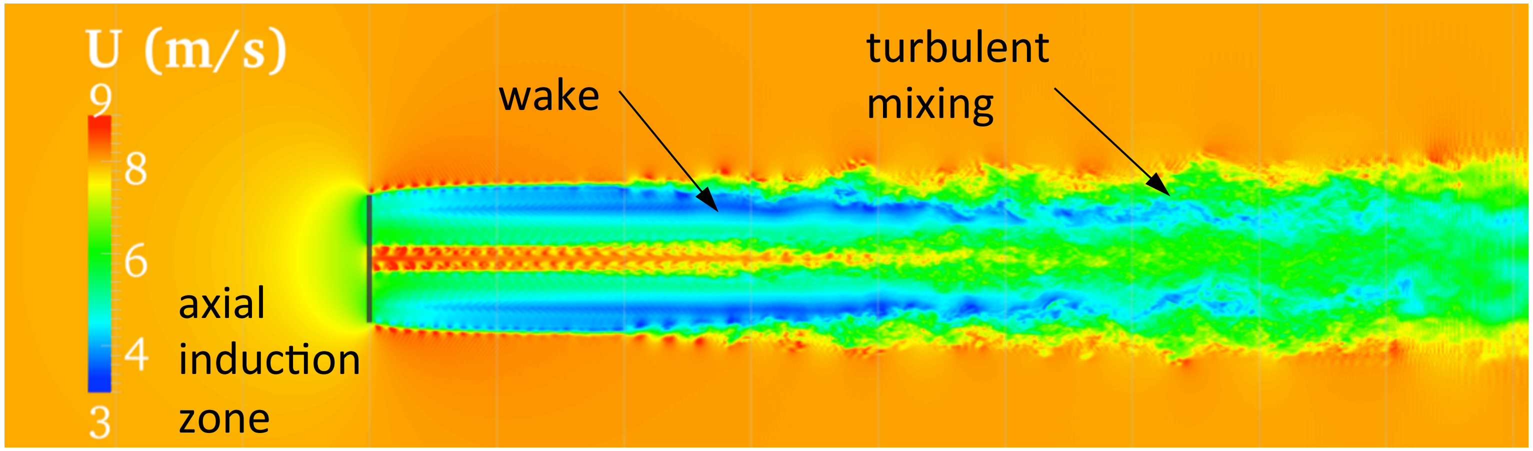


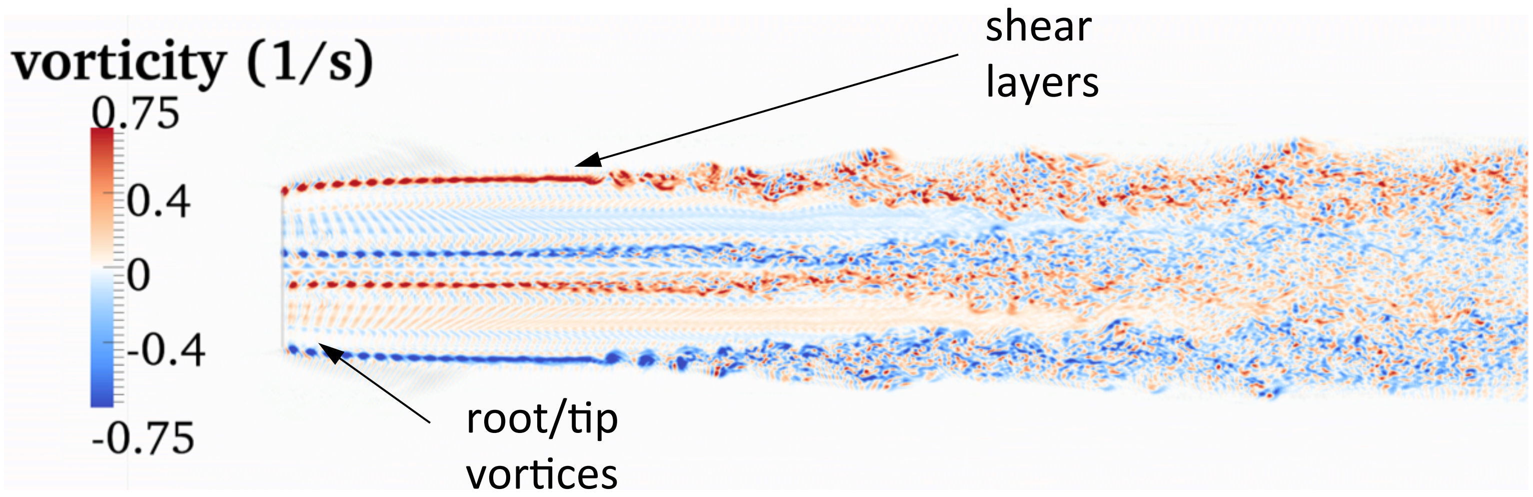
Simulation Method: Actuator Line

- **Actuator line model of turbine rotor**

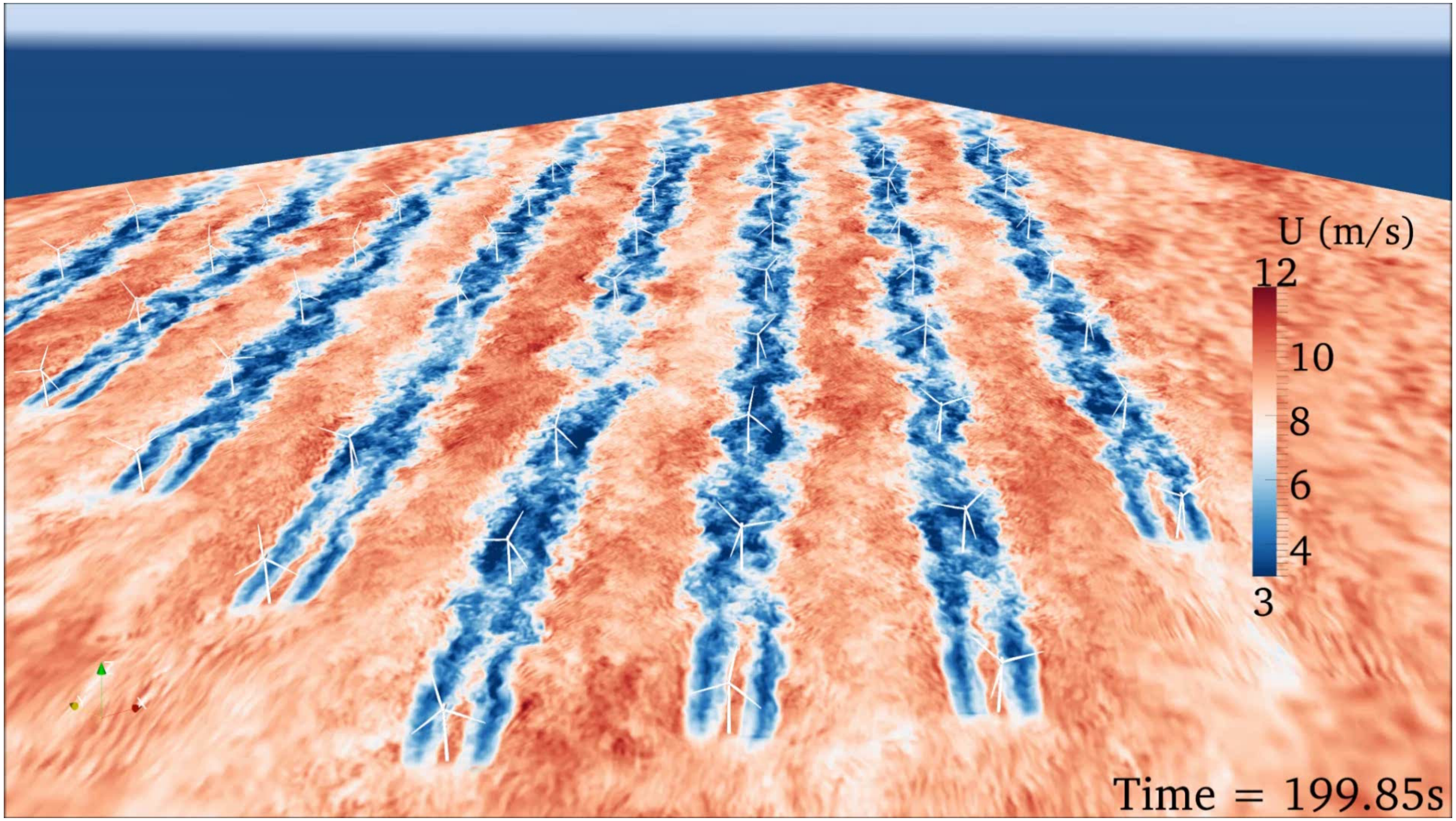


Simulation Method: Actuator Line



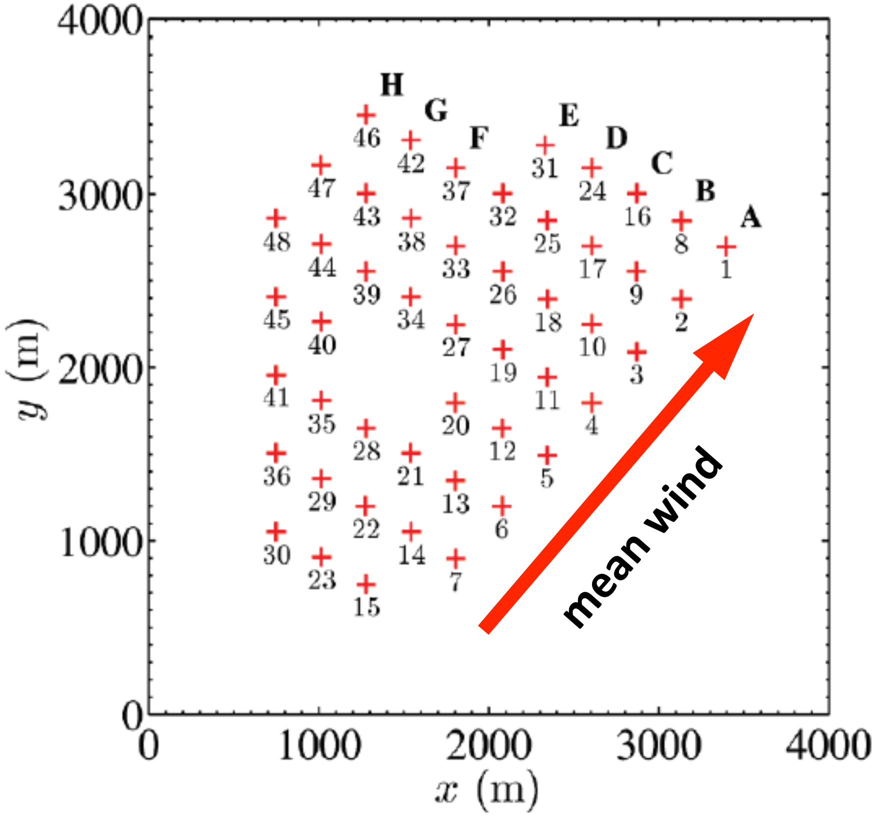


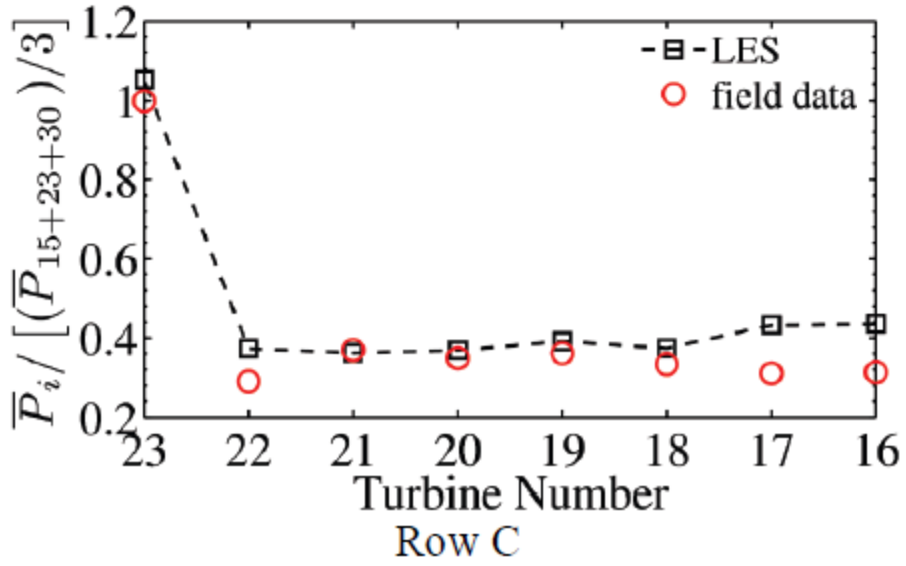
Simulation Method: Put It All Together

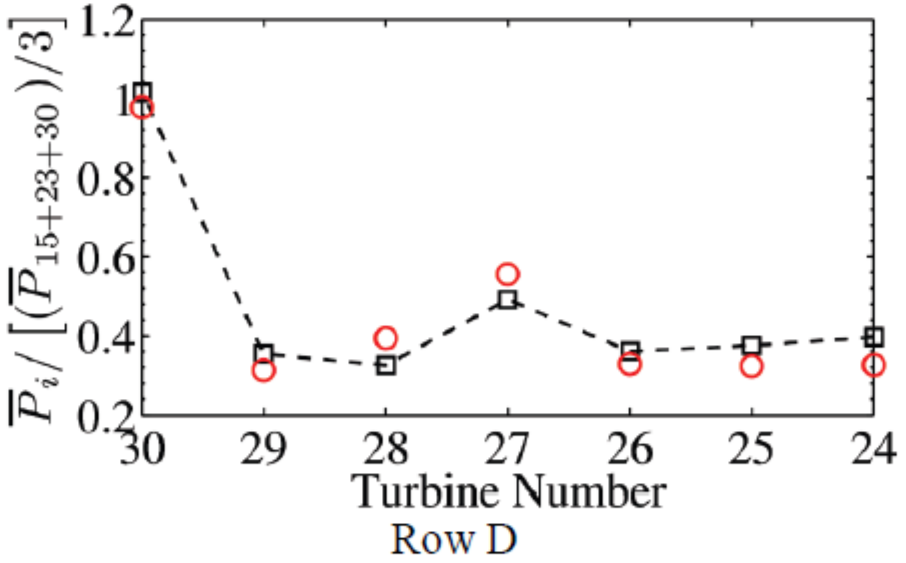


48 turbine Lillgrund offshore wind plant between Denmark and Sweden
Video played at 8X real time

How Well Does It Work?

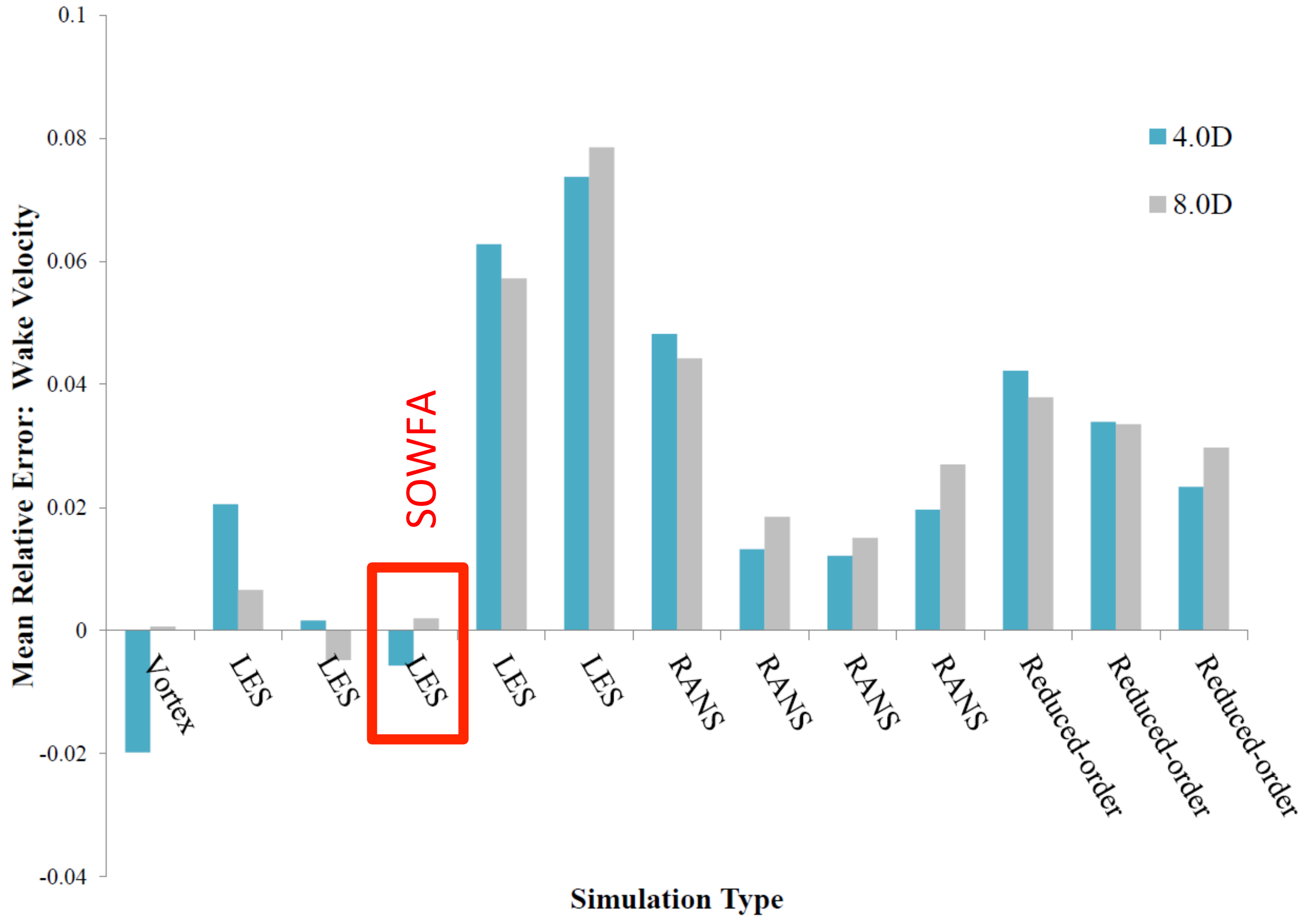






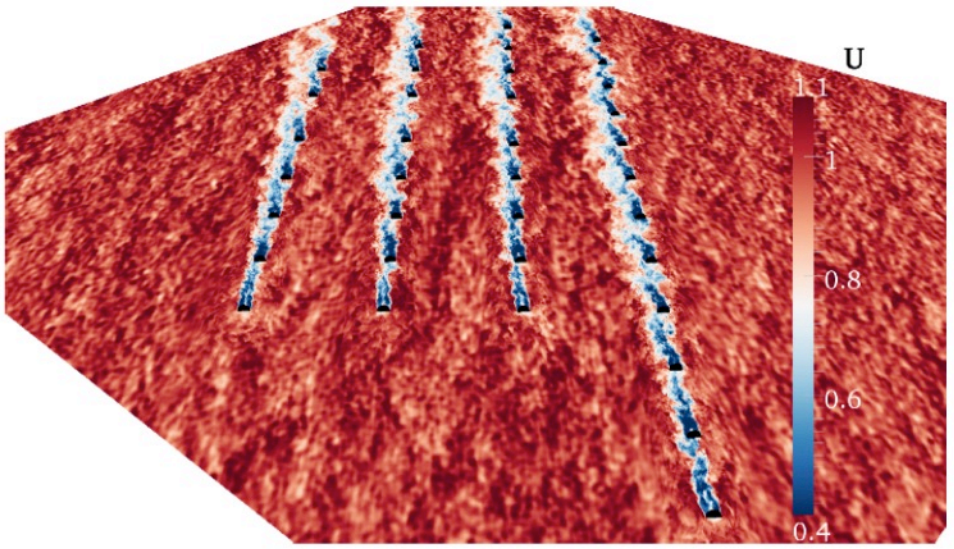
How Well Does It Work?

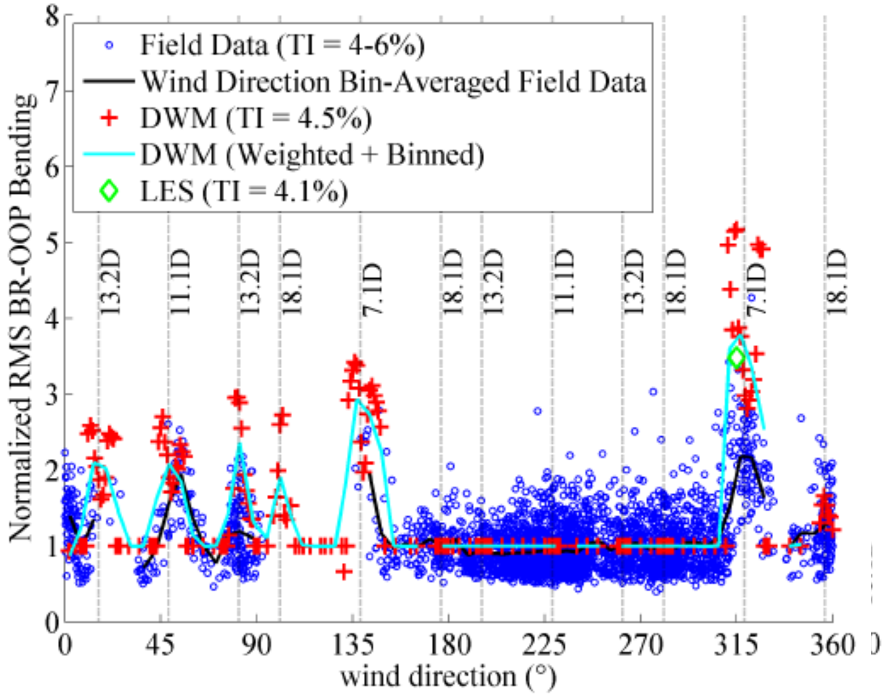
- **Sexbierum, single wake, turbulent atmospheric inflow**



How Well Does It Work?

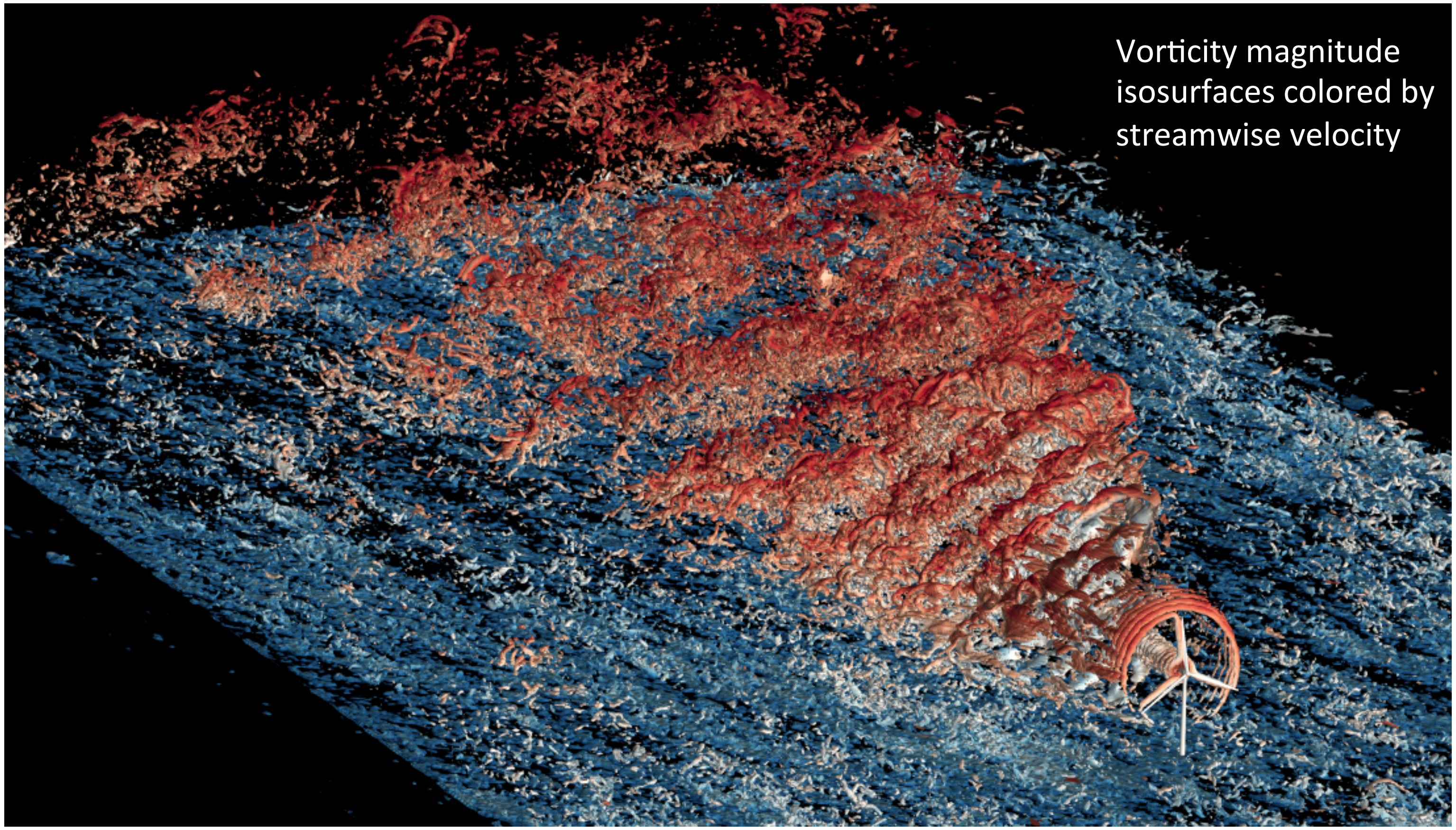
- **Simulation of the Egmond Aan Zee Wind Plant**
    - Power and loads





Studies of Atmospheric Stability Effects on Wakes

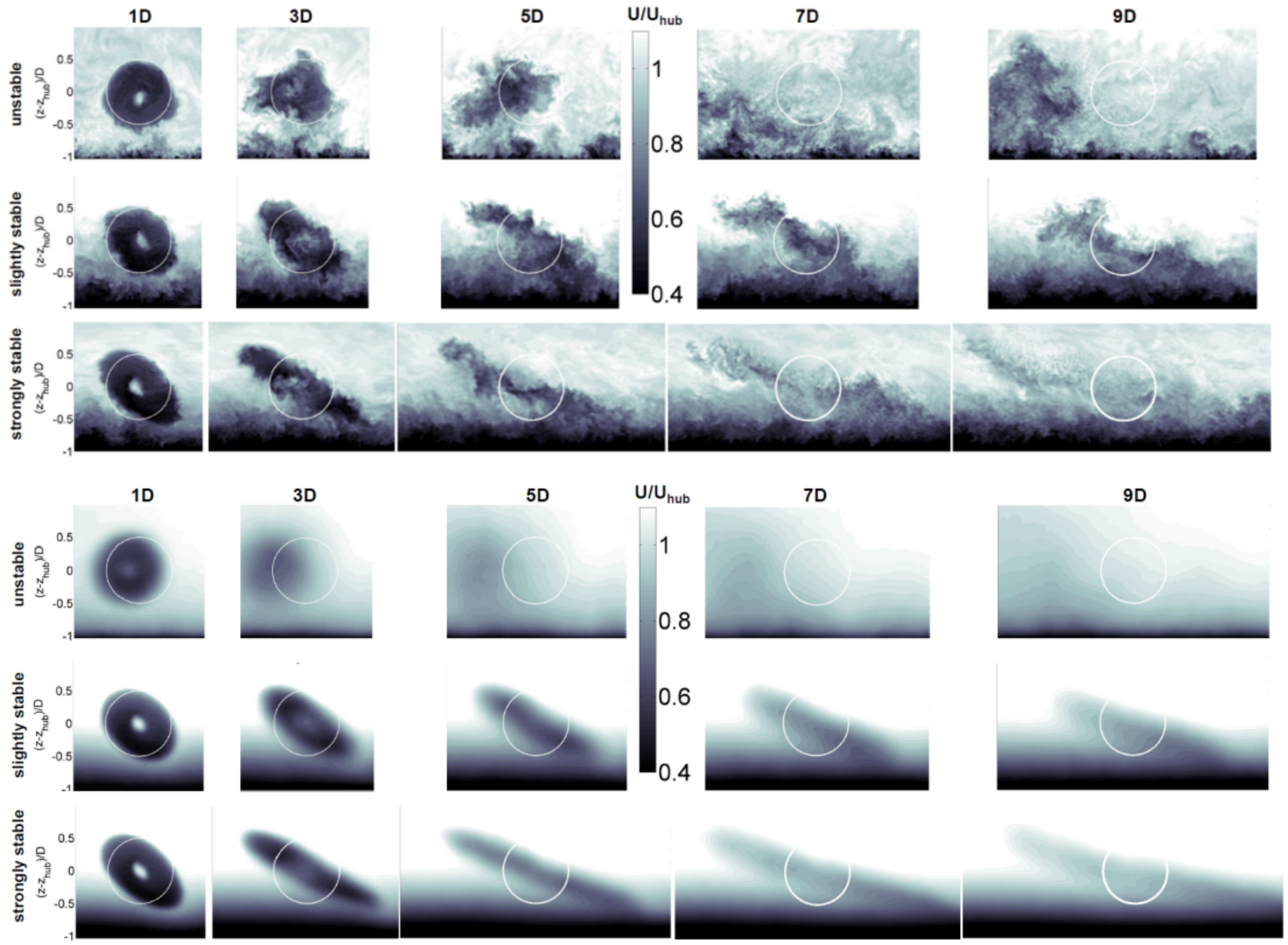
- **SOWFA can be used to study the effect of atmospheric stability on wakes**
  - Stability affects shear, veer, and turbulence levels and structure


Vorticity magnitude isosurfaces colored by streamwise velocity

Simulation of a single wind turbine subject to strongly stable conditions with 30° veer across the rotor disk

Studies of Atmospheric Stability Effects on Wakes

- **Contour planes of streamwise velocity through the wake at different stations downstream stations for different stability**



Using LES to Assess Vertical Profiling Lidar

- Virtual lidars placed behind turbines to measure vertical profiles of wakes
- Simulated lidar measurements compared to actual profiles
- We concluded that vertical scanning lidars are error prone where there is high shear, and have trouble sensing wake rotation

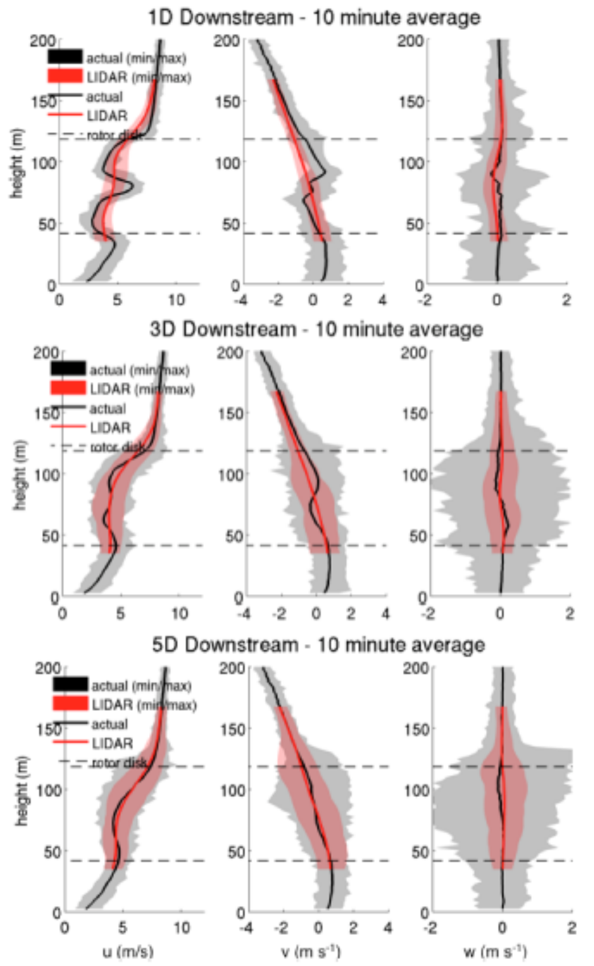
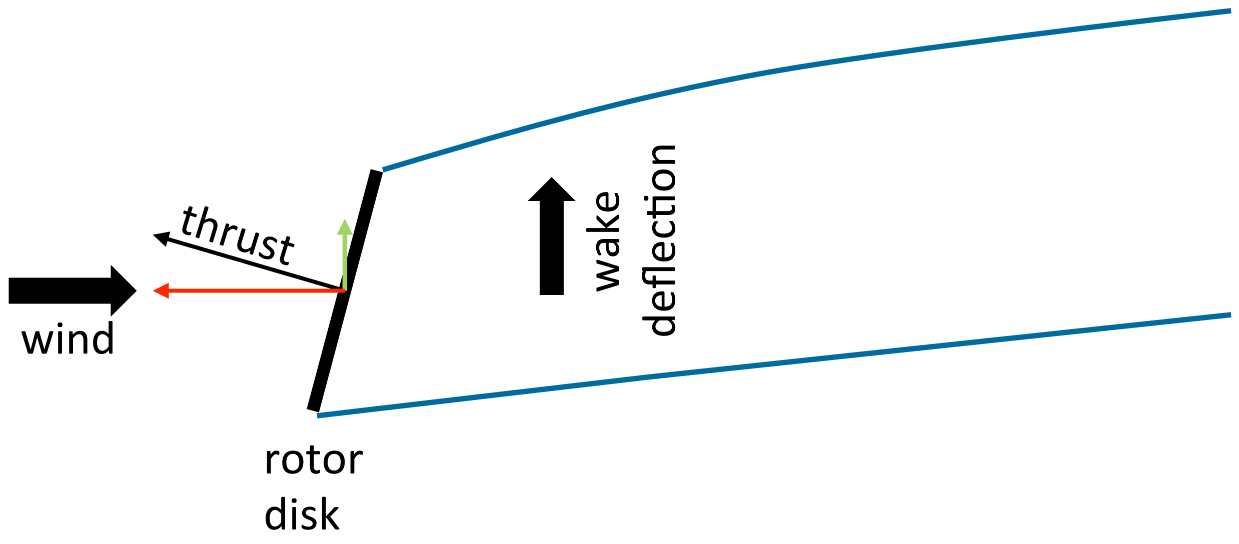


Figure 10. Vertical profiles of the stream-wise (left), cross-stream (middle), and vertical (right) components of wind speed at different locations downstream of the turbine on the wake centerline $y/D = 0$ as measured by the simulated lidars with weighting function applied (red lines) and directly sampled from the LES (black line). The shaded regions represent the envelope of all sampled values from the simulated lidar (pink) and from the direct LES sampling (gray).

An Interesting Application: Wake Redirection

- **Wake Redirection Control to Improve Performance**



An Interesting Application: Wake Redirection

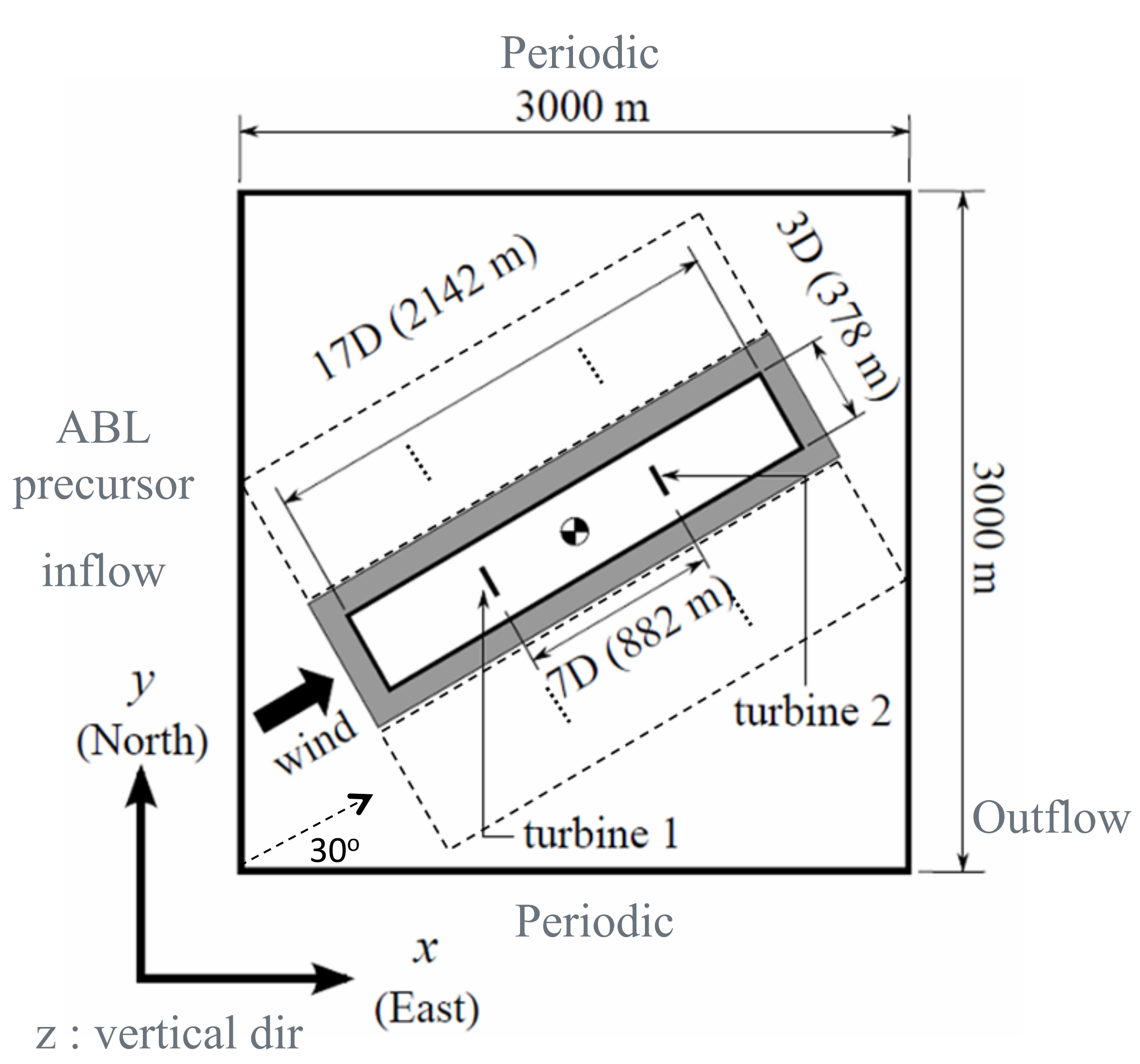
NREL Simulator for On/Offshore Wind Farm Applications (SOWFA)

NREL NATIONAL RENEWABLE ENERGY LABORATORY

TU Delft Delft University of Technology

Simulations and animations by Paul Fleming and Pieter Gebraad, NREL

Studying Turbine Mechanical Loads



2 turbines placed in different inflows and coupled with FAST to study loads

NREL 5-MW wind turbine

-Hub-height : 90 m

-Rotor diameter : 126 m

-U_{avg} at hub. : 8.0 m/s

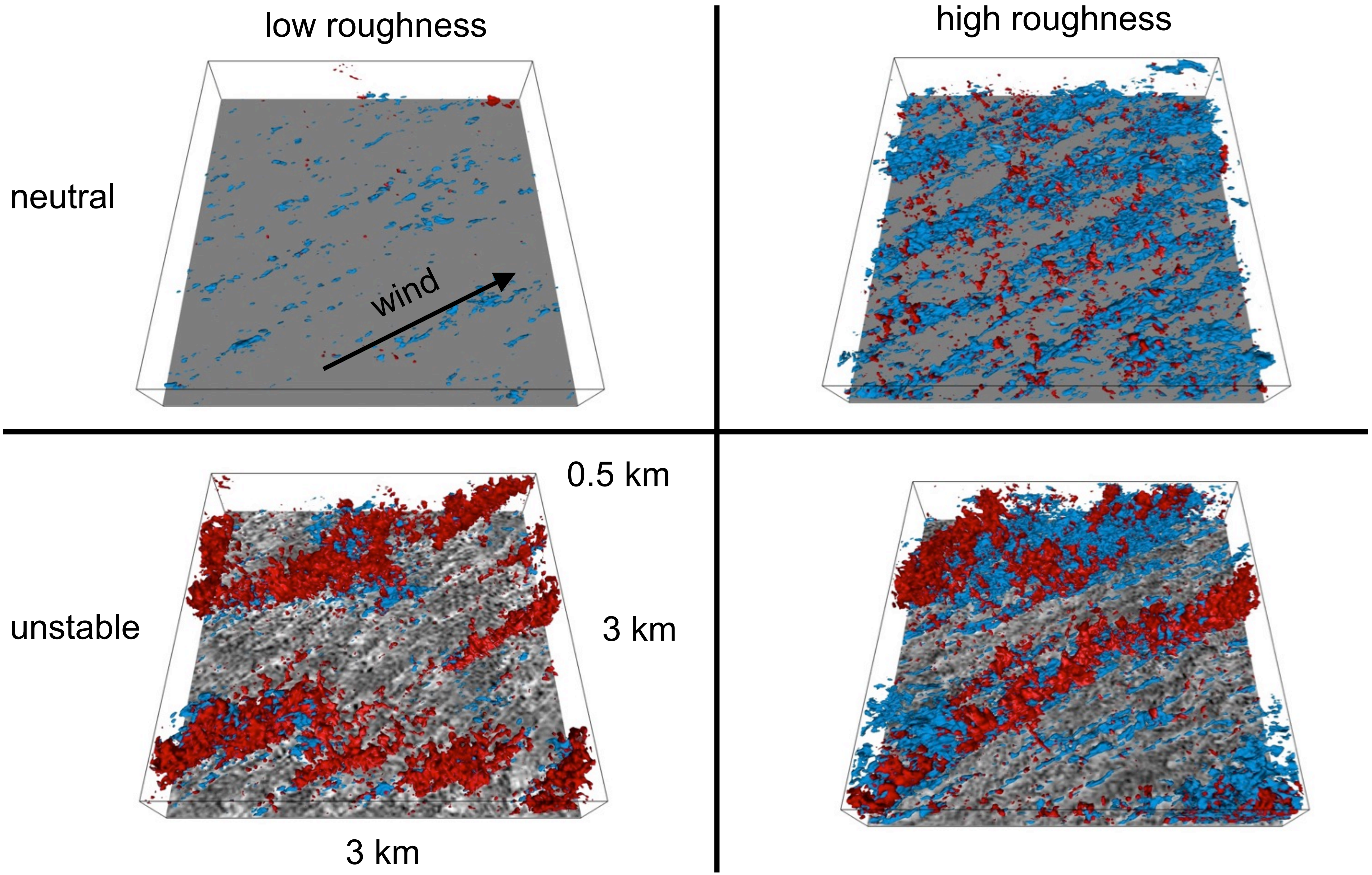
-Avg. rotational speed : 9.2 RPM

Domain and Grid

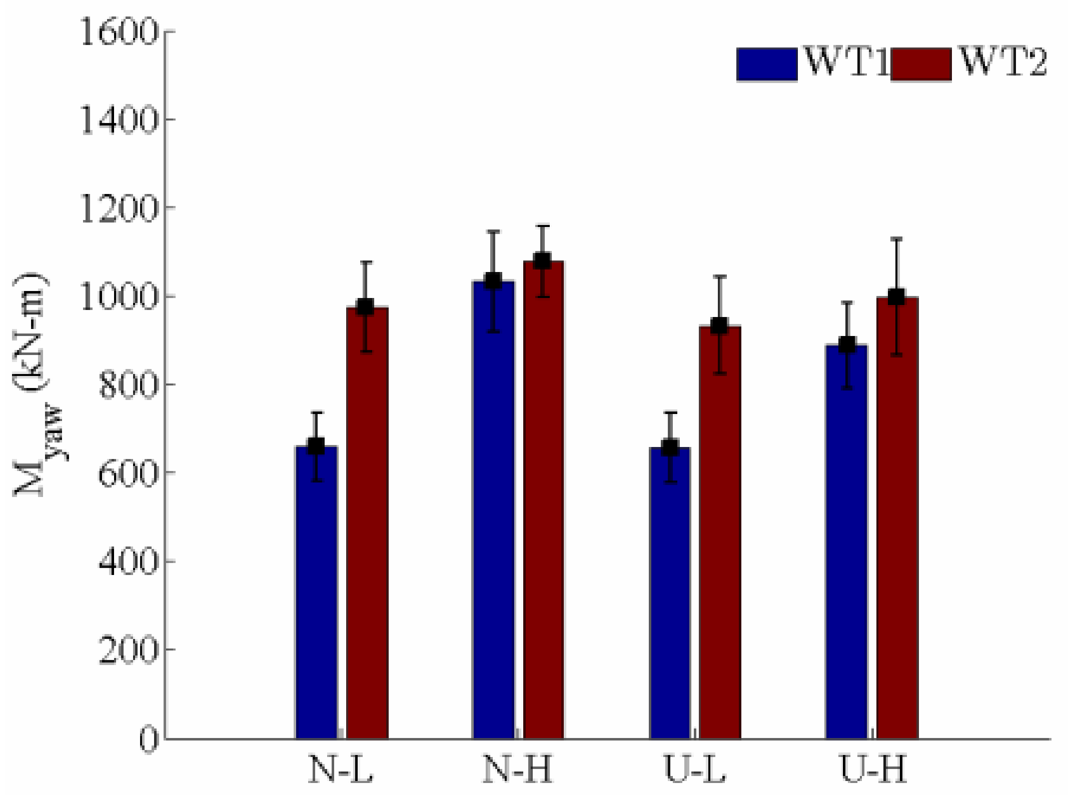
- 3km X 3km X 1km
- 10m, 5m, 2.5m resolution

5400 sec of load data collected

Studying Turbine Mechanical Loads



Studying Turbine Mechanical Loads

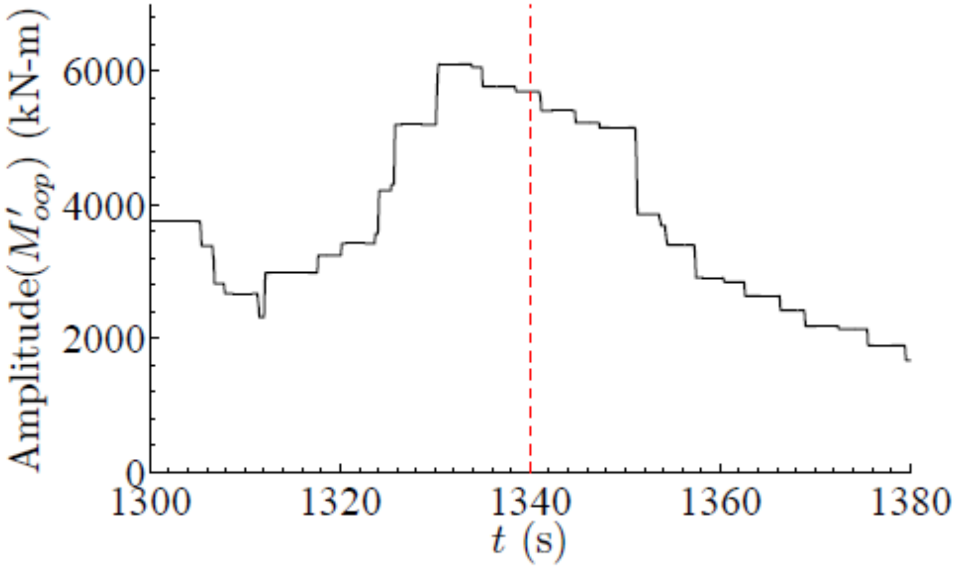


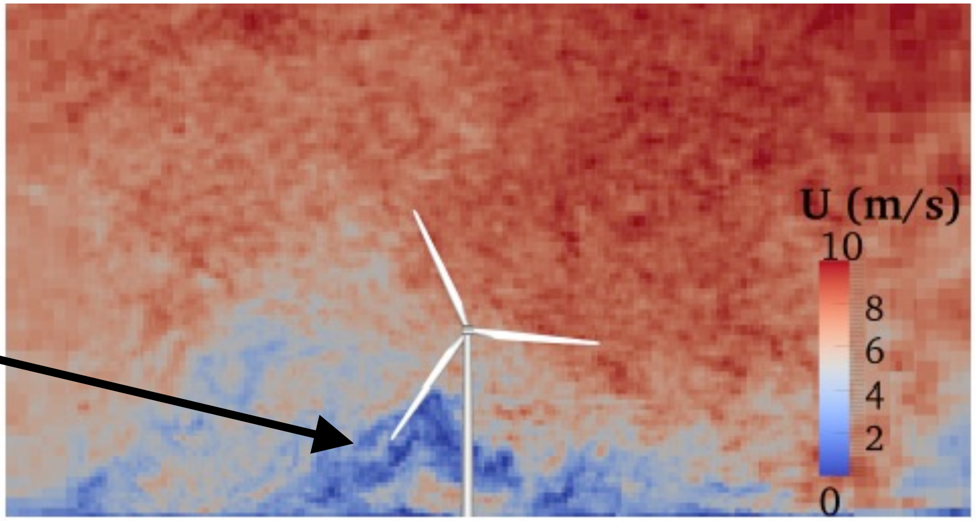
Damage Equivalent Loads (DEL)

Equivalent fatigue load under constant amplitude cycle that will produce the same amount of fatigue damage from actual loading history

Studying Turbine Mechanical Loads

Upstream turbine in high-roughness neutral

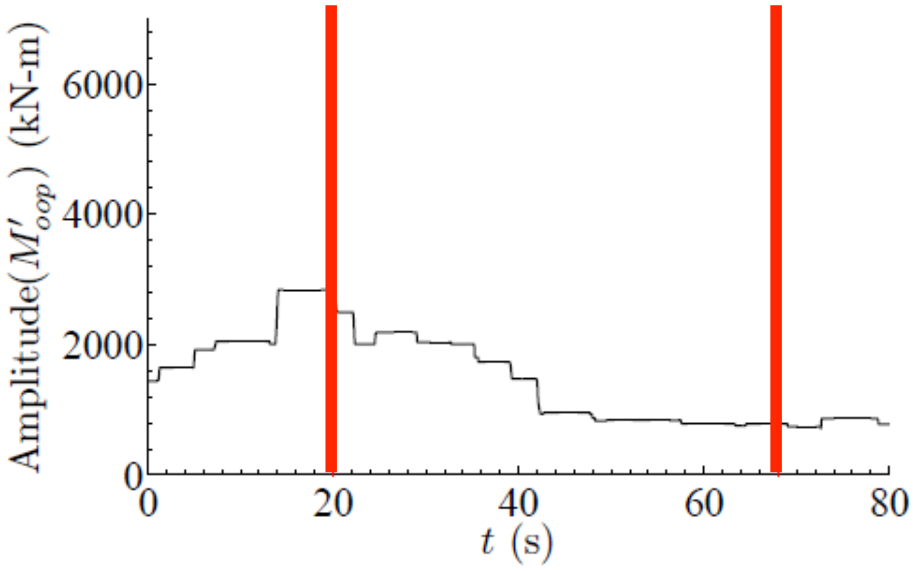


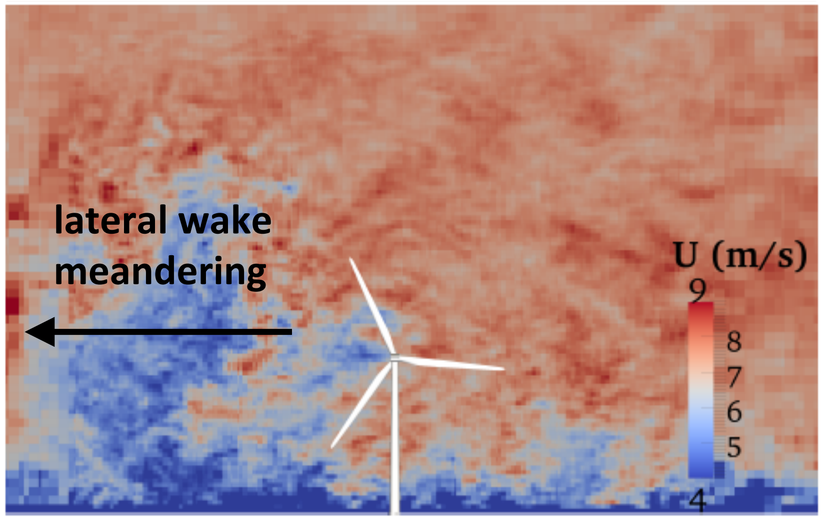


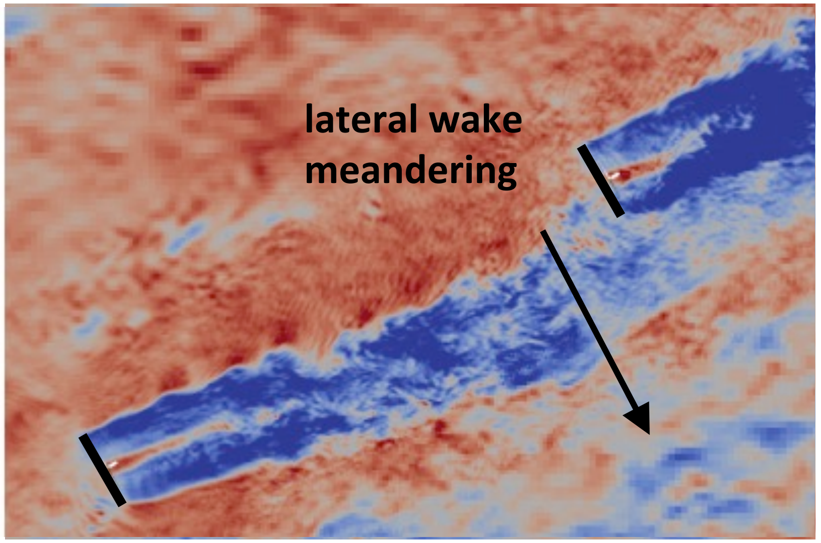
Shear-generated, low-speed turbulent structure

Studying Turbine Mechanical Loads

Downstream turbine in high-roughness unstable ABL



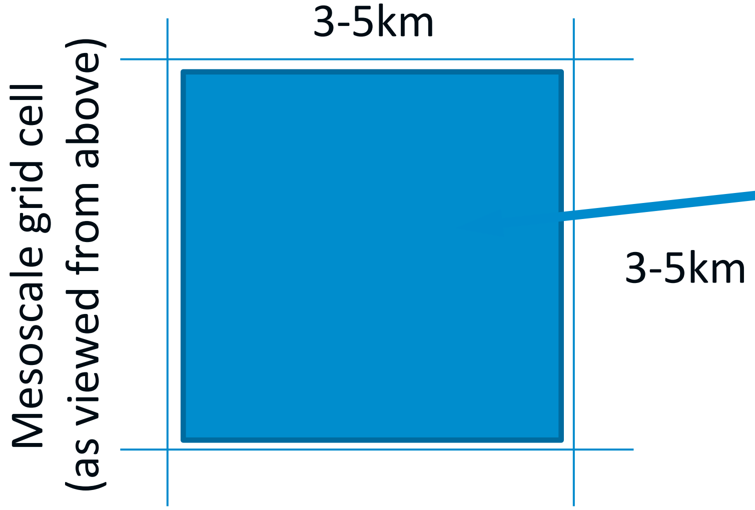




Meso-Micro Coupling

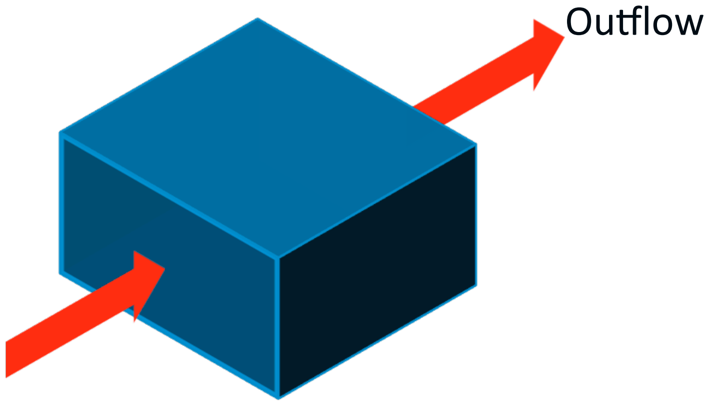
- **Flow is not always horizontally homogeneous and slowly varying in time**
- **Inhomogeneity often driven by mesoscale weather effects**
  - Frontal passages, weather patterns, microbursts, etc.
- **Mesoscale weather models provide, smooth RANS-like boundary data**
- **The challenge is to rapidly develop correct turbulent content**
  - Some sort of perturbation is necessary

Meso-Micro Coupling



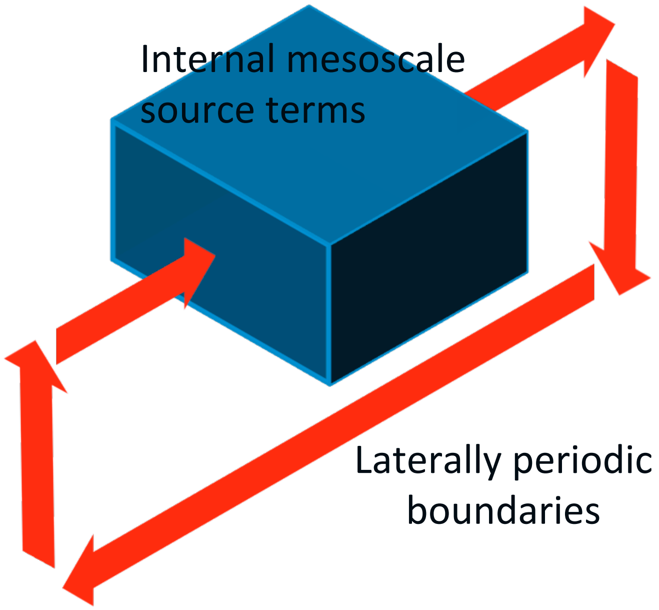
Microscale/wind plant domain footprint fits roughly within a single or a few mesoscale grid cell footprints

Approach 1

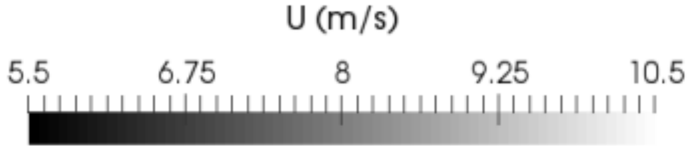


Mean inflow profiles interpolated from mesoscale weather model
+
Perturbations to initiate resolved turbulence

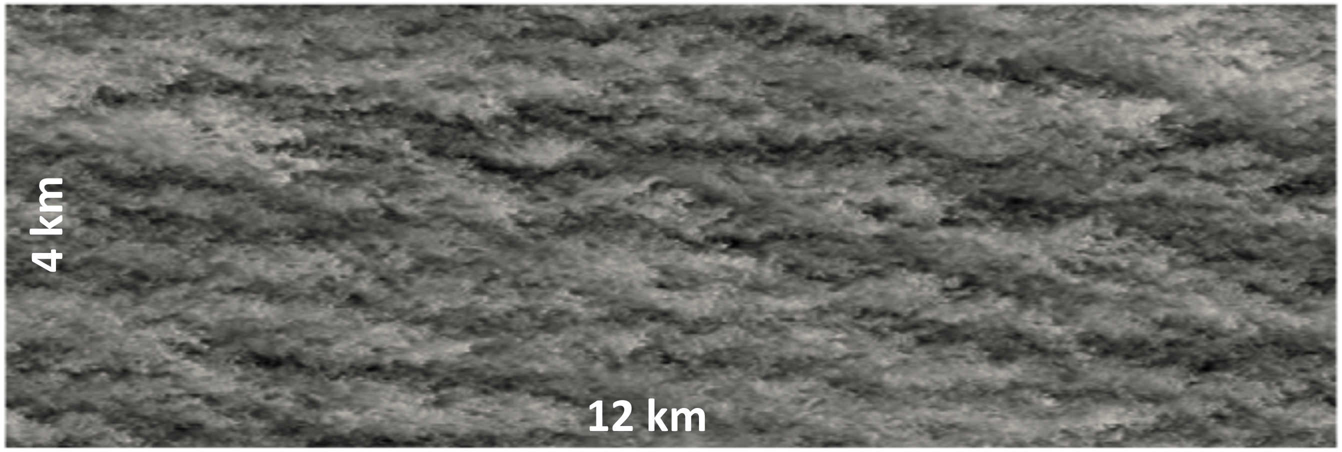
Approach 2


Internal mesoscale source terms
Laterally periodic boundaries

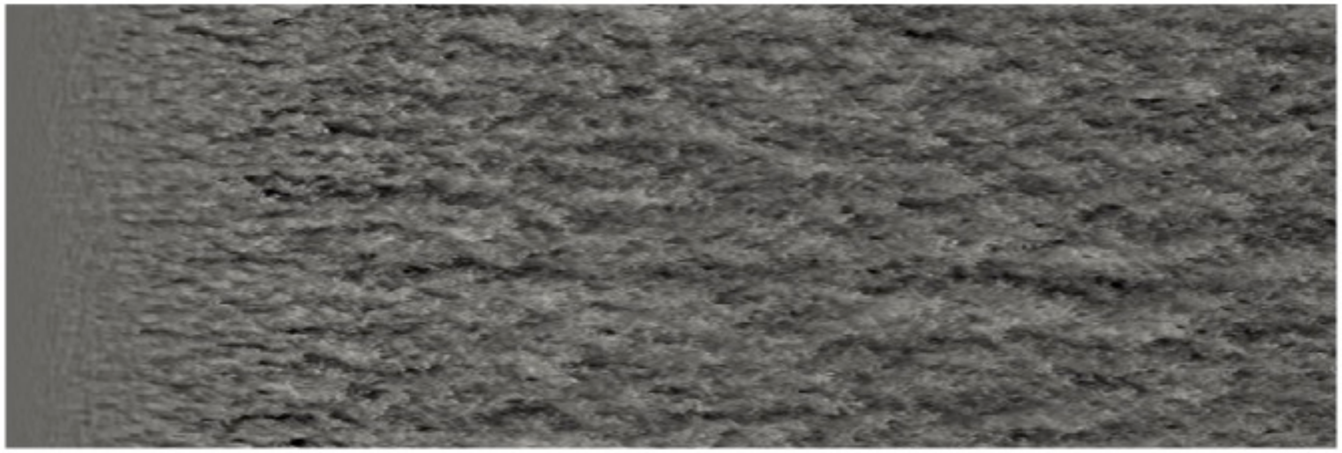
Meso-Micro Coupling



Temperature Perturbation



periodic precursor



±0.5K, 5s, 50m patch

±0.5K, 5s, 200m patch

±1.0K, 5s, 50m patch

±0.5K, 20s, 50m patch

Instantaneous streamwise velocity at 80 m above surface

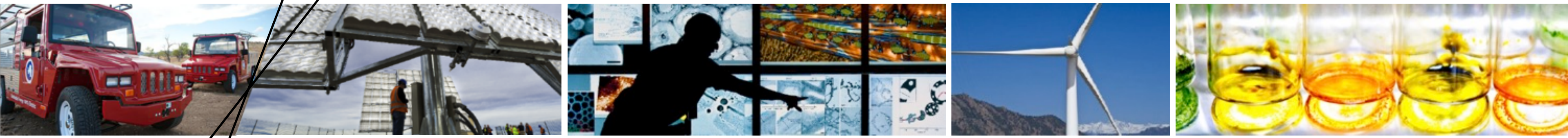
Meso-Micro Coupling

Inflow, a Major Source of Uncertainty

- Usually we try to generate turbulent ABL inflow given a desired hub-height mean wind speed, direction, and turbulence intensity (based only on the horizontal fluctuations)
- The nature of the turbulence is highly dependent on the initial temperature profile, surface roughness, surface heating/cooling, and wind speed at a given height, most of which is usually not measured in the field
- Two different stability conditions can yield similar turbulence intensity (based on the horizontal fluctuations), but different 3-component, or z-component turbulence intensity

Areas for Improvement

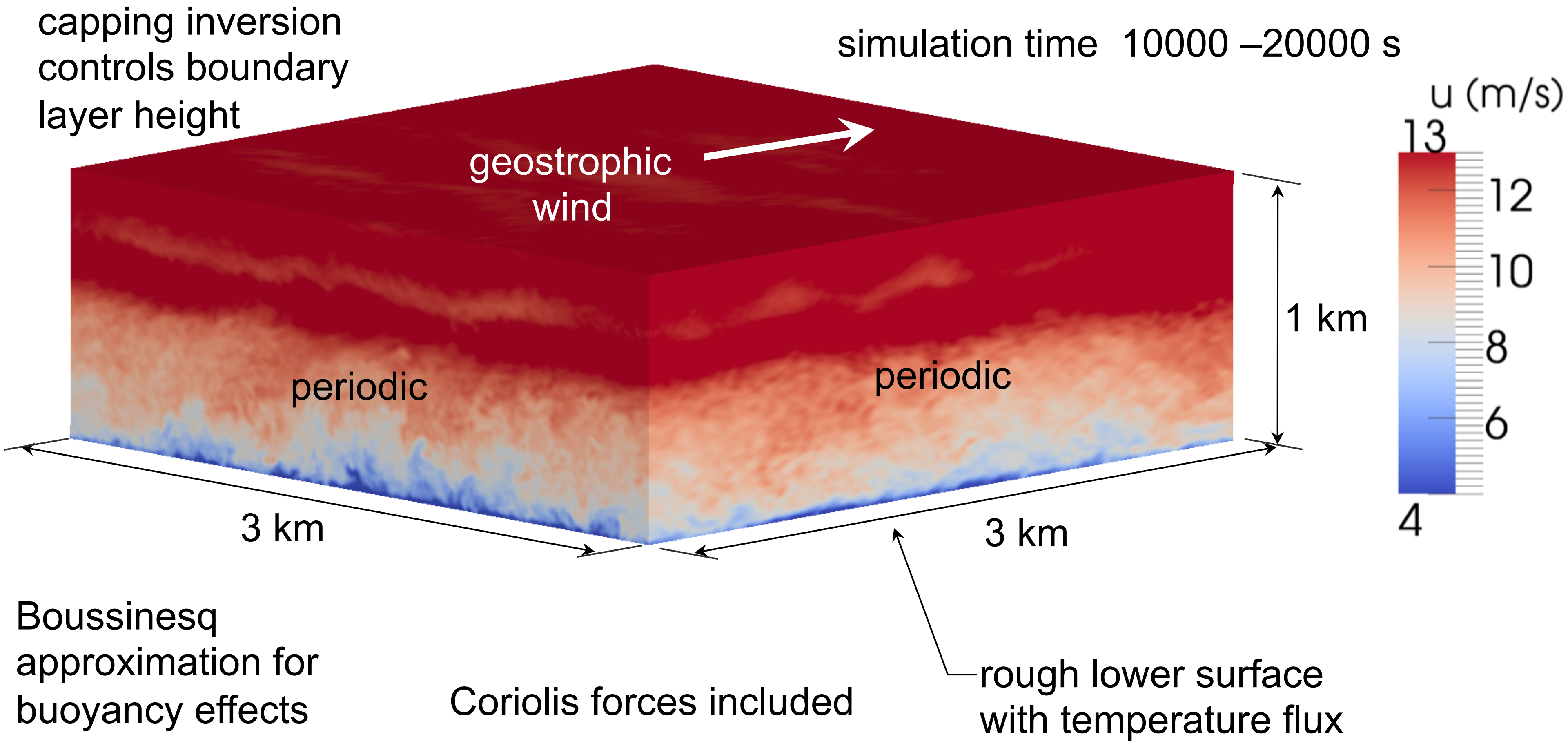
- More user-friendly (lots of inputs, but lots of flexibility)
- Currently, SOWFA is RANS capable, but untested
- More work for mesoscale coupling
- Improved local wall shear-stress models



Atmospheric Boundary Layer

Overview

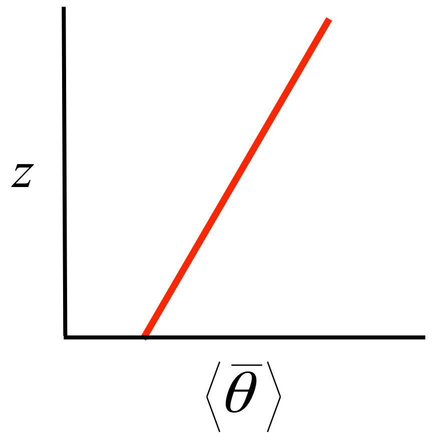
ABLSolver is an atmospheric solver based on the OpenFOAM standard buoyantBoussinesqPimpleFoam solver. It can be run in PISO or SIMPLE mode for either LES or RANS (or a blend). It can simulate a variety of atmospheric stabilities. We use it in LES mode to compute turbulent atmospheric precursor wind fields.


capping inversion controls boundary layer height
simulation time 10000 –20000 s
u (m/s)
13
12
10
8
6
4
geostrophic wind
1 km
periodic
periodic
3 km
3 km
Boussinesq approximation for buoyancy effects
Coriolis forces included
rough lower surface with temperature flux

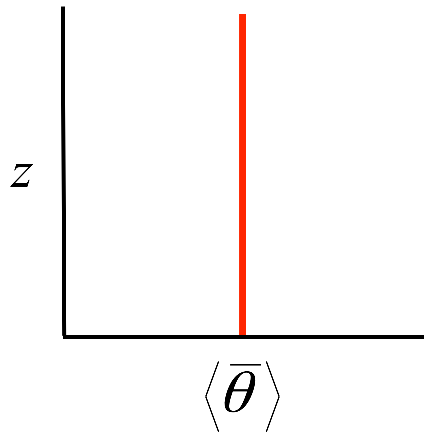
Atmospheric Stability

Potential Temperature and Stability

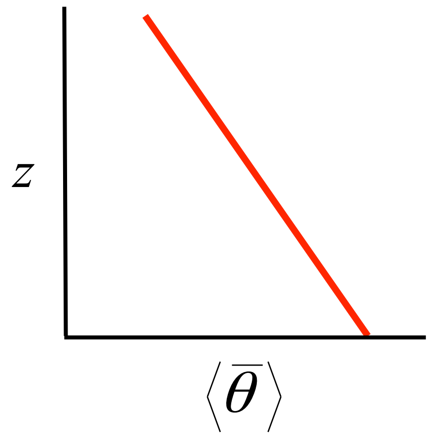
- Potential temperature (θ), is the temperature that a parcel of dry air would have if adiabatically brought from some pressure level to a reference pressure, usually 100kPA
- Simplifies the study of atmospheric stability



stable



neutral



unstable

Boussinesq Buoyancy Force

- This is an incompressible formulation, with constant density, so we need a way to account for buoyancy effects caused by variable density
- Use the Boussinesq approximation
- Ratio of “buoyant density” to constant density is

$$\frac{\rho_k}{\rho_0} = 1 - \left(\frac{\overline{\theta} - \theta_0}{\theta_0} \right)$$

- θ_0 is a reference temperature, usually some constant temperature that is related to initial temperature near the ground, often set to 300K
- Where temperature is greater than (less than) than θ_0, ρ_k/ρ_0 is greater (less than) 1.

Boussinesq Buoyancy Force

- **But it is the vertical gradient of the "Boussinesq density" that is important**

density gradient	buoyancy force	
$\frac{\partial}{\partial z}\left(\frac{\rho_k}{\rho_0}\right) = 0,$	$g\frac{\partial}{\partial z}\left(\frac{\rho_k}{\rho_0}\right) = 0$	locally neutral (air pushed into air of equal temperature): **zero force**
$\frac{\partial}{\partial z}\left(\frac{\rho_k}{\rho_0}\right) > 0,$	$g\frac{\partial}{\partial z}\left(\frac{\rho_k}{\rho_0}\right) < 0$	locally stable (**cool** air pushed **up** into **warm** air): **negative force**
$\frac{\partial}{\partial z}\left(\frac{\rho_k}{\rho_0}\right) < 0,$	$g\frac{\partial}{\partial z}\left(\frac{\rho_k}{\rho_0}\right) > 0$	locally unstable (**warm** air pushed **up** into **cool** air): **positive force**

Coriolis Force

Coriolis Force

- Due to planetary rotation, there is an apparent force called Coriolis force
- If +x is east, +y is north, and +z is up (at your location on the planet), then the force is:

$$-2\varepsilon_{ijk}\Omega_j\overline{u}_k$$

$$\Omega_j = \omega\begin{bmatrix} 0 \\ \cos\phi \\ \sin\phi \end{bmatrix}$$

- Ω_j is the rotation rate vector at a location on the planetary surface, ω is the planetary rotation rate (rad/s), and ϕ is the lattitude

Coriolis Force

- **Expanding out the Coriolis term by taking the cross product**

$$-2\varepsilon_{ijk}\Omega_j\bar{u}_k = -2\left[\left(\cancel{\Omega_2}\bar{u}_3 - \Omega_3\bar{u}_2\right)e_1 - \left(\cancel{\Omega_1}\bar{u}_3 - \Omega_3\bar{u}_1\right)e_2 + \cancel{\left(\Omega_1\bar{u}_2 - \Omega_2\bar{u}_1\right)e_3}\right]$$

- **Assumptions/simplifications:**
  - $\Omega_1 = 0$
  - $\bar{u}_3$ is small
  - vertical component of Coriolis force small compared to other vertical forces

$$-2\varepsilon_{ijk}\Omega_j\bar{u}_k = -2\left[-\Omega_3\bar{u}_2 e_1 + \Omega_3\bar{u}_1 e_2\right]$$

Governing Equations

Momentum Equation

Momentum transport

$$\underbrace{\frac{\partial \overline{u}_i}{\partial t}}_{\text{I}} + \underbrace{\frac{\partial}{\partial x_j}\left(\overline{u}_j \overline{u}_i\right)}_{\text{II}} = \underbrace{-2\varepsilon_{i3k}\Omega_3 \overline{u}_k}_{\text{III}} \underbrace{- \frac{\partial \widetilde{p}}{\partial x_i}}_{\text{IV}} \underbrace{- \frac{\partial}{\partial x_i}\left\langle \frac{\overline{p}_0(x,y)}{\rho_0} \right\rangle}_{\text{V}} \underbrace{- \frac{\partial}{\partial x_j}\left(\tau_{ij}^D\right)}_{\text{VI}} \underbrace{- g_3 z \frac{\partial}{\partial x_i}\left(\frac{\rho_k}{\rho_0}\right)}_{\text{VII}} \underbrace{+ \frac{1}{\rho_0} f_i^T}_{\text{VIII}}$$

I. time rate of change
II. convection
III. Coriolis force due to planetary rotation
IV. density-normalized pressure gradient (deviation from hydrostatic and horizontal-mean gradient)
V. horizontal-mean driving pressure gradient
VI. stresses (viscous + SGS/Reynolds)
VII. buoyancy
VIII. other density-normalized forces (from turbine actuator line model)

Explanation of the Pressure Variable

The pressure variable is:

$$\tilde{p} = \frac{\bar{p}}{\rho_0} - \frac{\rho_k}{\rho_0} g_j x_j - \left\langle \frac{\bar{p}_0(x_1, x_2)}{\rho_0} \right\rangle \qquad (1)$$

Normally, the momentum equation contains these terms on the RHS:

$$-\frac{1}{\rho_0}\frac{\partial \bar{p}}{\partial x_i} + \frac{\rho_k}{\rho_0} g_i \qquad (2)$$

Replacing (1) into (2) yields:

$$\begin{aligned}
-\frac{1}{\rho_0}\frac{\partial \bar{p}}{\partial x_i} &= -\frac{\partial \tilde{p}}{\partial x_i} - \frac{1}{\rho_0}\frac{\partial}{\partial x_i}\left(\rho_k g_j x_j\right) + \frac{\rho_k}{\rho_0} g_i - \frac{\partial}{\partial x_i}\left\langle \frac{\bar{p}_0(x_1, x_2)}{\rho_0} \right\rangle \qquad (3) \\
&= -\frac{\partial \tilde{p}}{\partial x_i} - g_j x_j \frac{\partial}{\partial x_i}\left(\frac{\rho_k}{\rho_0}\right) - \frac{\rho_k}{\rho_0} g_j \frac{\partial x_j}{\partial x_i} + \frac{\rho_k}{\rho_0} g_i - \frac{\partial}{\partial x_i}\left\langle \frac{\bar{p}_0(x_1, x_2)}{\rho_0} \right\rangle \\
&= -\frac{\partial \tilde{p}}{\partial x_i} - g_j x_j \frac{\partial}{\partial x_i}\left(\frac{\rho_k}{\rho_0}\right) - \frac{\rho_k}{\rho_0} g_i + \frac{\rho_k}{\rho_0} g_i - \frac{\partial}{\partial x_i}\left\langle \frac{\bar{p}_0(x_1, x_2)}{\rho_0} \right\rangle \\
&= -\frac{\partial \tilde{p}}{\partial x_i} - g_j x_j \frac{\partial}{\partial x_i}\left(\frac{\rho_k}{\rho_0}\right) - \frac{\partial}{\partial x_i}\left\langle \frac{\bar{p}_0(x_1, x_2)}{\rho_0} \right\rangle
\end{aligned}$$

Explanation of the Pressure Variable

Replacing (1) into (2) yields:

$$-\frac{1}{\rho_0}\frac{\partial \overline{p}}{\partial x_i} = -\frac{\partial \widetilde{p}}{\partial x_i} - g_j x_j \frac{\partial}{\partial x_i}\left(\frac{\rho_k}{\rho_0}\right) - \frac{\partial}{\partial x_i}\left\langle \frac{\overline{p}_0(x_1, x_2)}{\rho_0}\right\rangle \qquad (3)$$

For gravity in the standard z-direction (i.e., g = [0,0,-9.81] m/s^2), this is:

$$-\frac{1}{\rho_0}\frac{\partial \overline{p}}{\partial x_i} = -\frac{\partial \widetilde{p}}{\partial x_i} - g_3 z \frac{\partial}{\partial x_i}\left(\frac{\rho_k}{\rho_0}\right) - \frac{\partial}{\partial x_i}\left\langle \frac{\overline{p}_0(x_1, x_2)}{\rho_0}\right\rangle \qquad (4)$$

Thus we have this form of the momentum equation:

$$\frac{\partial \overline{u}_i}{\partial t} + \frac{\partial}{\partial x_j}\left(\overline{u}_j \overline{u}_i\right) = -2\varepsilon_{i3k}\Omega_3 \overline{u}_k - \frac{\partial \widetilde{p}}{\partial x_i} - \frac{\partial}{\partial x_i}\left\langle \frac{\overline{p}_0(x, y)}{\rho_0}\right\rangle - \frac{\partial}{\partial x_j}\left(\tau_{ij}^D\right) - g_3 z \frac{\partial}{\partial x_i}\left(\frac{\rho_k}{\rho_0}\right) + \frac{1}{\rho_0} f_i^T$$

This pressure variable varies much less with height than absolute pressure, making it easier to solve for numerically. Also, there is no variation in this pressure laterally (in the mean) because the lateral pressure gradient is separated, which is convenient for periodic atmospheric boundary layer simulations.

Momentum Equation in the Code

Momentum transport

```
fvVectorMatrix UEqn
(
    fvm::ddt(U)                                    // time derivative
  + fvm::div(phi, U)                               // convection
  + turbulence->divDevReff(U)                      // stresses (interior faces)
  + fvc::div(Rwall)                                // stresses at boundary (wall model)
  - fCoriolis                                      // Coriolis force
  + gradPd                                         // driving pressure gradient
);

UEqn.relax();

if (pimple.momentumPredictor())
{
    solve
    (
        UEqn
        ==
        fvc::reconstruct
        (
            (
              - fvc::snGrad(p_rgh)                 // modified pressure gradient
              - ghf*fvc::snGrad(rhok)              // buoyancy force
            ) * mesh.magSf()
        )
    );
}
```

Potential Temperature Equation

Potential temperature transport

$$\underbrace{\frac{\partial \bar{\theta}}{\partial t}}_{\textbf{I}} + \underbrace{\frac{\partial}{\partial x_j}\left(\bar{u}_j \bar{\theta}\right)}_{\textbf{II}} = - \underbrace{\frac{\partial}{\partial x_j}\left(q_j\right)}_{\textbf{III}}$$

I. time rate of change
II. convection
III. viscous + SGS/Reynolds temperature fluxes

[1] provides a good explanation of atmospheric boundary layer physics.

[2] is a good outline of atmospheric boundary layer LES.

[1] R. B. Stull. *An Introduction to Boundary Layer Meteorology*. Springer Science + Business Media B. V., 2009.
[2] C.-H. Moeng. A Large-Eddy Simulation Model for the Study of Planetary Boundary Layer Turbulence. *Journal of the Atmospheric Sciences*, Vol. 41, No. 13,

Potential Temperature Equation in the Code

Potential temperature transport

```
kappat = turbulence->nut()/Prt;
kappat.correctBoundaryConditions();

volScalarField kappaEff("kappaEff", turbulence->nu()/Pr + kappat);

fvScalarMatrix TEqn
(
    fvm::ddt(T)                             // time derivative
  + fvm::div(phi, T)                        // convection
  - fvm::laplacian(kappaEff, T)             // diffusion (molecular + turbulent)
  - fvc::div(qwall)                         // temperature flux at boundary
);

TEqn.relax();
TEqn.solve();

rhok = 1.0 - ( (T - TRef)/TRef );           // Boussinesq buoyancy density
```

Subgrid-Scale Modeling

Subgrid-Scale Model

Gradient-diffusion hypothesis

$$\tau_{ij}^{D} = -\upsilon^{SGS}\left(\frac{\partial \overline{u}_i}{\partial x_j} + \frac{\partial \overline{u}_j}{\partial x_i}\right)$$

$$q_j = -\kappa^{SGS}\,\frac{\partial \overline{\theta}}{\partial x_j}$$

Standard Smagorinsky model[1]

$$\upsilon^{SGS} = \left(C_s \Delta\right)^2 \left[2\left(\frac{\partial \overline{u}_i}{\partial x_j} + \frac{\partial \overline{u}_j}{\partial x_i}\right)\left(\frac{\partial \overline{u}_i}{\partial x_j} + \frac{\partial \overline{u}_j}{\partial x_i}\right)\right]^{1/2}$$

$$\kappa^{SFS} = \frac{\upsilon^{SFS}}{\mathrm{Pr}_t}$$

[1] J. Smagorinsky. General Circulation Experiments with the Primitive Equations, *Monthly Weather Review*, Vol. 91, 1963, pp. 99–164.

Subgrid-Scale Model

$C_s = 0.13 - 0.17$ (we use 0.135) Smagorinsky constant

$\Delta = V^{1/3}$ SGS filter width (V is grid cell volume)

$\mathrm{Pr}_t = \frac{1}{3}$ neutral and unstable

$\mathrm{Pr}_t = 1$ stable

Turbulent Prandtl number

Subgrid-Scale Model

A more advanced turbulent Prandtl number formulation that we need to implement[1]

$$\mathrm{Pr}_t = \frac{1}{\left(1 + 2\frac{l}{\Delta}\right)}$$

Turbulent Prandtl number

$$l = \begin{cases} \min\left(7.6\frac{\upsilon^{SGS}}{\Delta}\left(\sqrt{\frac{1}{s}}\right), \Delta\right) & \text{if} \quad s > 0 \\ \Delta & \text{if} \quad s \leq 0 \end{cases}$$

Length-scale for Prt

$$s = \frac{|g_i|}{\theta_0}\frac{\partial\overline{\theta}}{\partial z}$$

Measure of stability

If locally unstable or neutral ($s \leq 0$): $\mathrm{Pr}_t = 1/3$ $\kappa^{SFS} = 3\upsilon^{SFS}$
If locally stable ($s > 0$): Pr_t approaches 1 $\kappa^{SFS} \rightarrow \upsilon^{SFS}$

[1] C.-H. Moeng. A Large-Eddy Simulation Model for the Study of Planetary Boundary Layer Turbulence. *Journal of the Atmospheric Sciences*, Vol. 41, No. 13,

Subgrid-Scale Model

- The weak point of the Standard Smagorinsky model is the need to choose the model constant, C_s, and the constant is non-local
- That lead to the development of dynamic variants of the Smagorinsky model, which the model choses C_s based on minimizing the error between the "test filtered" Leonard stresses and the modeled ones
- The error is a tensor, and Lilly showed that mean square error could be minimized using

$$C_s^2 = \frac{M_{ij}L_{ij}}{M_{kl}M_{kl}}$$

- where

$$M_{ij} = 2\left[\bar{\Delta}^2 \widetilde{\bar{S}\bar{S}_{ij}} - \tilde{\Delta}^2 \tilde{\bar{S}}\tilde{\bar{S}}_{ij}\right] \qquad L_{ij} = T_{ij} - \tilde{\tau}_{ij} = \widetilde{\bar{U}_i\bar{U}_j} - \tilde{\bar{U}}_i\tilde{\bar{U}}_j$$

- and the overbar denotes the actual filter and the tilde denotes the test filter (usually 2x the actual filter). The actual and test filter widths are

$\bar{\Delta}$ and $\tilde{\Delta}$

Subgrid-Scale Model

- In practice, computing $C_s^2 = \frac{M_{ij}L_{ij}}{M_{kl}M_{kl}}$ locally is often unstable

- To remedy this, $C_s^2 = \frac{\langle M_{ij}L_{ij}\rangle}{\langle M_{kl}M_{kl}\rangle}$ is often computed where the brackets denote some sort of averaging

- Often the averaging is planar averages, but that does not work for horizontally inhomogeneous flow (i.e., wind plants, flow over terrain)

- The averaging can be done in a Lagrangian sense, backwards along streamlines using an exponential weighting, putting more weight close to the point of interest[1]

[1] C. Meneveau, T.S. Lund, W.H. Cabot, "A Lagrangian Averaged Dynamic Subgrid-Scale Model of Turbulence. *J. of Fluid Mech*., 319, pp. 353-385..

Subgrid-Scale Model

- **If exponential weighting is used, then the Lagrangian averages can be solved for with a relaxation transport equation (i.e., a lag equation)**

- **Let** $\mathscr{I}_{LM} = \left\langle M_{ij} L_{ij} \right\rangle$ and $\mathscr{I}_{MM} = \langle M_{kl} M_{kl} \rangle$

- **Then the Lagrangian averages can be solved with**

$$\frac{\partial \mathscr{I}_{LM}}{\partial t} + \frac{\bar{U}_j \mathscr{I}_{LM}}{\partial x_j} = \frac{1}{\theta \bar{\Delta} \left(\mathscr{I}_{LM} \mathscr{I}_{MM} \right)^{-1/8}} \left(L_{ij} M_{ij} - \mathscr{I}_{LM} \right),$$

$$\frac{\partial \mathscr{I}_{MM}}{\partial t} + \frac{\bar{U}_j \mathscr{I}_{MM}}{\partial x_j} = \frac{1}{\theta \bar{\Delta} \left(\mathscr{I}_{LM} \mathscr{I}_{MM} \right)^{-1/8}} \left(M_{ij} M_{ij} - \mathscr{I}_{MM} \right)$$

- **The RHS coefficient controls the amount of averaging or relaxation, but what is shown here is what is recommended by Meneveau et al.[1]**

[1] C. Meneveau, T.S. Lund, W.H. Cabot, "A Lagrangian Averaged Dynamic Subgrid-Scale Model of Turbulence. *J. of Fluid Mech*., 319, pp. 353-385..

Subgrid-Scale Model

- **Meneveau et al. do not recommend actually solving these equations because of the expense, but rather use an approximate method backward interpolating in space and time to perform the average[1]**

$$\frac{\partial \mathscr{I}_{LM}}{\partial t} + \frac{\bar{U}_j \mathscr{I}_{LM}}{\partial x_j} = \frac{1}{\theta \bar{\Delta} \left(\mathscr{I}_{LM}\mathscr{I}_{MM}\right)^{-1/8}} \left(L_{ij}M_{ij} - \mathscr{I}_{LM}\right),$$

$$\frac{\partial \mathscr{I}_{MM}}{\partial t} + \frac{\bar{U}_j \mathscr{I}_{MM}}{\partial x_j} = \frac{1}{\theta \bar{\Delta} \left(\mathscr{I}_{LM}\mathscr{I}_{MM}\right)^{-1/8}} \left(M_{ij}M_{ij} - \mathscr{I}_{MM}\right)$$

- **But, Meneveau's code is structured; this is much harder to do with an unstructured code like OpenFOAM, so OpenFOAM actually solves the lag equations (relatively inexpensive compared to pressure equation)**
- **Also, Meneveau's code is pseudo spectral, so test filtering is easily done in wave number space; OpenFOAM uses an approximate filter based on the Laplacian of the velocity field**

[1] C. Meneveau, T.S. Lund, W.H. Cabot, "A Lagrangian Averaged Dynamic Subgrid-Scale Model of Turbulence. *J. of Fluid Mech*., 319, pp. 353-385..

Subgrid-Scale Model

- We found the Lagrangian-averaged dynamic Smagorinsky model that comes standard with OpenFOAM to create too noisy and fields
- Much improved behavior was created by clipping the resultant Cs field from above and below at 0.07 and 0.14
- However, we used pure central differencing on the convective term of the relaxation equations, and the and fields are naturally somewhat noisy, so linear central differencing may have exacerbated this noisiness
- We have not tested it, but better behavior (without clipping) likely could be achieved by using a more diffusive convection scheme
- We also find the Lagrangian-averaged dynamic model to have a difficult time on more complex meshes (WP1, for example), and have had to revert to the standard Smagorinsky model
- We need to investigate OpenFOAM's approximate test filtering, also

Subgrid-Scale Model, Conclusions

- Since the atmospheric solvers are based on PIMPLE, the codes can be run either as RANS in SIMPLE or LES in PISO, all in one code
- Soon will implement the NCAR (Deardorff) one-equation model for SGS turbulent kinetic energy

  AND

- Kosovic's nonlinear anisotropic backscatter one-equation SGS model
- Also will further study the Lagrangian-averaged dynamic Smagorinsky model and make it more in line with Meneveau et al.

Domain Size, Boundary and Initial Conditions

Wall Shear Stress and Temp. Flux Models

- **The cost of high-Re fully-resolved LES of wall-bounded flow scales strongly with Re.**

  "The only economical way to perform LES of high Reynolds-number attached flow, therefore, is by computing the outer layer only." "Because the grid is too coarse to resolve the inner-layer structures, the effect of the wall layer must be modeled. In particular, the momentum flux at the wall (i.e., the wall stress) cannot be evaluated by discrete differentiation because the grid cannot resolve either the sharp velocity gradients in the inner layer or the quasi-streamwise and hairpin vortices that transfer momentum in this region of the flow. Therefore, some phenomelogical relation must be found to relate the wall stress to the outer-layer flow."[1]

- **The planetary surface is covered with roughness elements (dirt, rocks, vegetation) that would be extremely expensive to resolved with the grid.**
- **It is inappropriate to apply no-slip at the surface**
- **Instead apply a model for surface stress**

[1] U. Piomelli and E. Balaras, "Wall-Layer Models for Large-Eddy Simulations," Annual Review of Fluid Mechanics, Vol. 34, pp. 349–374, 2002.

Wall Shear Stress Model

- Surface stress model predicts total (viscous + SGS) stress at surface
- Assumes that first cell centers away from surface lie within surface layer of the atmospheric boundary layer
- So at the surface

$$\tau_{ij}^{D} = \begin{bmatrix} 0 & 0 & \tau_{13}^{tot} \\ 0 & 0 & \tau_{23}^{tot} \\ \tau_{13}^{tot} & \tau_{23}^{tot} & 0 \end{bmatrix}$$

- The wall model models τ_{13}^{tot} and τ_{23}^{tot}

Wall Shear Stress Model

- **SOWFA contains the wall models of**
  - Schumann[1]
  - Moeng[2] (Not yet reimplemented)
- **Schumann's model**

$$\tau_{13}^{tot} = -u_*^2 \frac{\left(\overline{u}_{1/2} - \left\langle \overline{u}_{1/2} \right\rangle\right)}{\left(\left\langle \overline{u}_{1/2} \right\rangle^2 + \left\langle \overline{v}_{1/2} \right\rangle^2\right)^{1/2}}$$

$$\tau_{23}^{tot} = -u_*^2 \frac{\left(\overline{v}_{1/2} - \left\langle \overline{v}_{1/2} \right\rangle\right)}{\left(\left\langle \overline{u}_{1/2} \right\rangle^2 + \left\langle \overline{v}_{1/2} \right\rangle^2\right)^{1/2}}$$

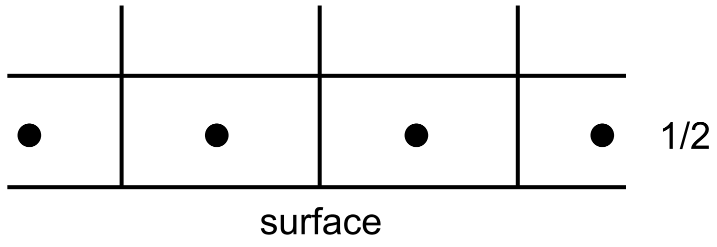
[1] U. Schumann. Subgrid-Scale Model for Finite-Difference Simulations of Turbulent Flow in Plane Channels and Annuli. Journal of Computational Physics, Vol. 18, 1975, pp. 76–404.
[2] C.-H. Moeng. A Large-Eddy Simulation Model for the Study of Planetary Boundary Layer Turbulence. Journal of the Atmospheric Sciences, Vol. 41, No. 13, 1984, pp. 2052–2062.

Wall Shear Stress Model

$$\tau_{13}^{tot} = -u_*^2 \frac{\left(\overline{u}_{1/2} - \left\langle \overline{u}_{1/2} \right\rangle\right)}{\left(\left\langle \overline{u}_{1/2} \right\rangle^2 + \left\langle \overline{v}_{1/2} \right\rangle^2\right)^{1/2}}$$

$$\tau_{23}^{tot} = -u_*^2 \frac{\left(\overline{v}_{1/2} - \left\langle \overline{v}_{1/2} \right\rangle\right)}{\left(\left\langle \overline{u}_{1/2} \right\rangle^2 + \left\langle \overline{v}_{1/2} \right\rangle^2\right)^{1/2}}$$

- **1/2 denotes values at first cell centers away from surface**



- **Angle brackets denote a horizontal average at a certain height**

Wall Shear Stress Model

$$\tau_{13}^{tot} = -u_*^2 \frac{\left(\overline{u}_{1/2} - \left\langle \overline{u}_{1/2} \right\rangle\right)}{\left(\left\langle \overline{u}_{1/2} \right\rangle^2 + \left\langle \overline{v}_{1/2} \right\rangle^2\right)^{1/2}}$$

$$\tau_{23}^{tot} = -u_*^2 \frac{\left(\overline{v}_{1/2} - \left\langle \overline{v}_{1/2} \right\rangle\right)}{\left(\left\langle \overline{u}_{1/2} \right\rangle^2 + \left\langle \overline{v}_{1/2} \right\rangle^2\right)^{1/2}}$$

- **Friction velocity is defined as**

$$u_*^2 = \left(\left\langle \tau_{13}^{tot} \right\rangle^2 + \left\langle \tau_{23}^{tot} \right\rangle^2\right)^{1/2}$$

- **It needs to be approximated. Use rough wall log law**

$$\frac{\left(\left\langle \overline{u}_{1/2} \right\rangle + \left\langle \overline{v}_{1/2} \right\rangle\right)^{1/2}}{u_*} = \frac{1}{\kappa} \ln\left(\frac{z}{z_0} + f(L)\right)$$

Wall Shear Stress Model

$$\frac{\left(\langle \overline{u}_{1/2} \rangle + \langle \overline{v}_{1/2} \rangle\right)^{1/2}}{u_*} = \frac{1}{\kappa} \ln\left(\frac{z}{z_0} + f(L) \right)$$

- **$f(L)$ is an atmospheric stability-related function that is zero for neutral stability. See Etling[1] for more information**
- **L is the Obuhkov length**
- **z_0 is the aerodynamic roughness height. It depends on height, distribution, and shape of roughness elements on planetary surface. See Stull[2] for more information**

z_0 (m)	Terrain
$1\times10^{-1} – 5\times10^{-1}$	Many trees, hedges, few buildings
$3\times10^{-3} – 2\times10^{-2}$	Level grass plains
$1\times10^{-4} – 1\times10^{-3}$	Large expanses of water

[1] D. Etling. Modelling the Vertical ABL Structure, in *Modelling of Atmospheric Flow Fields*, D. P. Lalas and C. F. Ratto, editors, World Scientific, 1996, pp. 56–57.
[2] R. B. Stull. *An Introduction to Boundary Layer Meteorology*. Springer Science

Wall Temperature Flux Model

- A similar approach is taken to model the total temperature flux at the surface[1]

$$q_j = \begin{bmatrix} 0 \\ 0 \\ q_3^{tot} \end{bmatrix}$$

- Total average temperature flux, Q_s, is specified, and the wall model creates the fluctuating temperature flux q_3^{tot}
- Or surface heating/cooling rate is specified and q_3^{tot} is calculated.[2] If simulating stable atmospheric conditions, it is best to specify a surface cooling rate

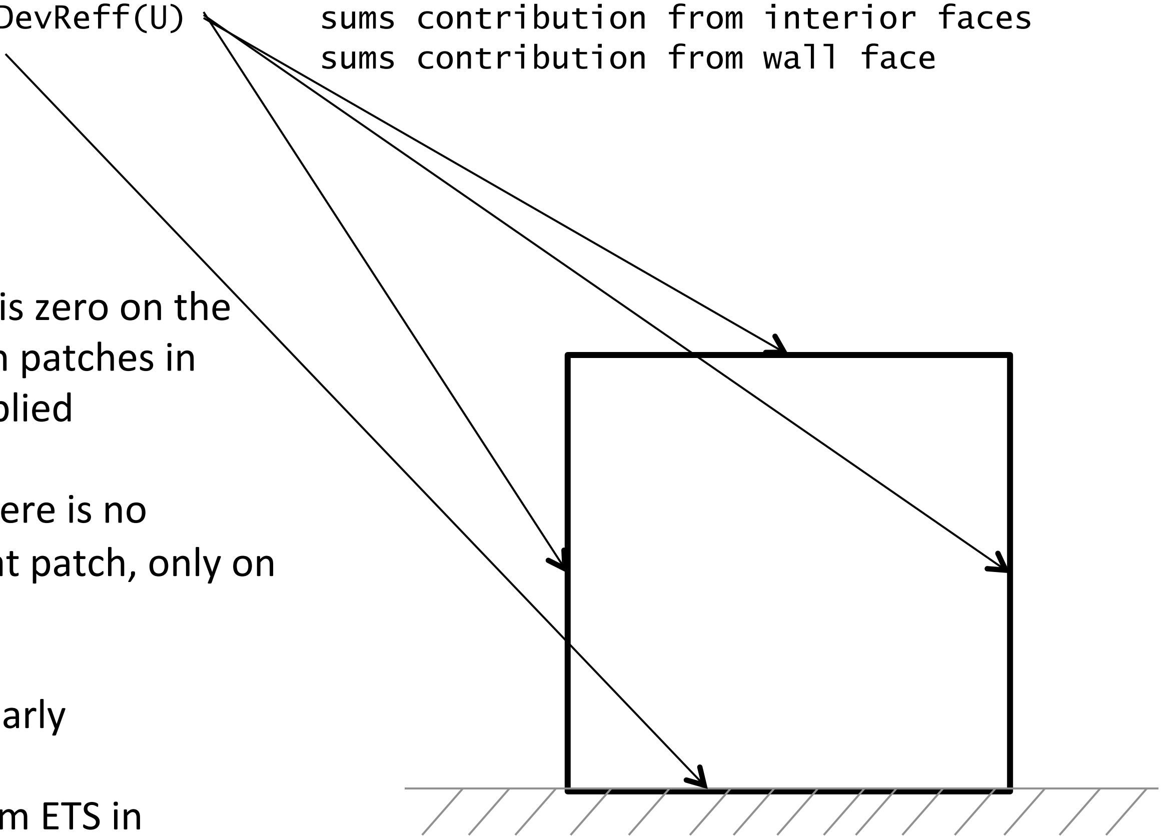
[1] C.-H. Moeng. A Large-Eddy Simulation Model for the Study of Planetary Boundary Layer Turbulence. Journal of the Atmospheric Sciences, Vol. 41, No. 13, 1984, pp. 2052–2062.
[2] S. Basu, A. A. M. Holtslag, B. J. H. Van de Wiel, A. F. Moene, G.-J. Steeneveld, "An inconvenient "truth" about using sensible heat flux as a surface boundary condition in models under stably stratified regimes," Acta Geophysica, Vol. 56, No. 1, 2008, pp. 88-99.

How to Incorporate Wall Model

- **OpenFOAM has standard wall shear stress models that look at the surface velocity gradient and assign a surface eddy-viscosity such that their product gives the correct stress**
- **This does not allow for the use of a non-linear surface shear stress model, like that of Moeng**
- **SOWFA includes wall model BCs in the Rwall and qwall variables**

How to Incorporate Wall Model

```
fvVectorMatrix UEqn
    (
        fvm::ddt(U)
      + fvm::div(phi, U)
      + turbulence->divDevReff(U)       sums contribution from interior faces
      + fvc::div(Rwall)                 sums contribution from wall face
      - fCoriolis
      + gradPd
    );
```



Rwall is a volSymmTensorField that is zero on the interior and only takes on a value on patches in which the wall shear stress BC is applied

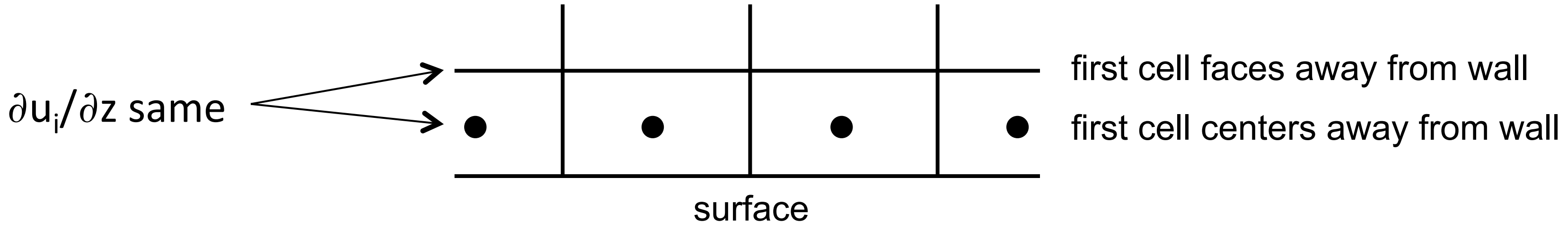
If nuEff is zero on the patch, then there is no contribution from divDevReff on that patch, only on the interior

Temperature flux model works similarly

I credit David Lapointe-Thériault from ETS in Montreal for this method of splitting up the stresses at faces between interior and wall and treating them separately while still usind divDevReff()

Other Boundary Conditions

- **Velocity, U**
  - Lower:
    - no wall-normal flow, but no need to specify wall-parallel flow (surface shear stress model takes care of that)
    - However, SGS models need velocity gradients at cell centers adjacent to wall, and that is computed by differencing across cell faces, so a wall-parallel velocity must be specified. We follow Penn State ABL code and set wall-parallel velocity such that cell center vertical gradient same as next cell face gradient, but Moeng[1] suggests another method


$\partial u_i/\partial z$ same
first cell faces away from wall
first cell centers away from wall
surface

[1] C.-H. Moeng. A Large-Eddy Simulation Model for the Study of Planetary Boundary Layer Turbulence. *Journal of the Atmospheric Sciences*, Vol. 41, No. 13,

Other Boundary Conditions

- **Velocity, U**
  - Upper
    - Slip
  - Lateral
    - If a precursor, purely ABL, periodic simulation, then set to cyclic (periodic)
    - If wind plant, use time-varying inflow based on saved data planes from a precursor ABL simulation, and outflow $\partial u/\partial n = 0$, and if flow tries to re-enter, $u=0$

Other Boundary Conditions

- **Potential temperature, T**
  - Lower
    - Zero gradient
  - Upper
    - Fixed gradient equal to capping inversion gradient
  - Lateral
    - If a precursor, purely ABL, periodic simulation, then set to cyclic (periodic)
    - If wind plant, use time-varying inflow based on saved data planes from a precursor ABL simulation, and outflow $\partial T/\partial n = 0$

Other Boundary Conditions

- **Pressure variable, p_rgh**
  - If a periodic simulation, set periodic boundaries to cyclic (periodic)
  - Otherwise, all boundaries are set to OpenFOAM's "buoyantPressure" boundary condition, which starts with the momentum equation

$$\cancel{\frac{\partial \bar{u}_i}{\partial t}} + \cancel{\frac{\partial}{\partial x_j}\left(\bar{u}_j \bar{u}_i\right)} = \cancel{-2\varepsilon_{i3k}\Omega_3 \bar{u}_k} - \frac{\partial \tilde{p}}{\partial x_i} - \cancel{\frac{\partial}{\partial x_i}\left\langle \frac{\bar{p}_0(x,y)}{\rho_0} \right\rangle} - \cancel{\frac{\partial}{\partial x_j}\left(\tau_{ij}^D\right)} - g_3 z \frac{\partial}{\partial x_i}\left(\frac{\rho_k}{\rho_0}\right) + \cancel{\frac{1}{\rho_0} f_i^T}$$

$$\frac{\partial \tilde{p}}{\partial n} = -g_3 z \frac{\partial}{\partial n}\left(\frac{\rho_k}{\rho_0}\right)$$

Domain Size and Resolution

- **Make domain at least as large and with the resolution given in the table below:**

	Horizontal Extent	Vertical Extent	Minimum Resolution
Neutral	3 km	1 km	20 m
Unstable	5 km	2 km	20 m
Stable	500 m	300 m	10 m

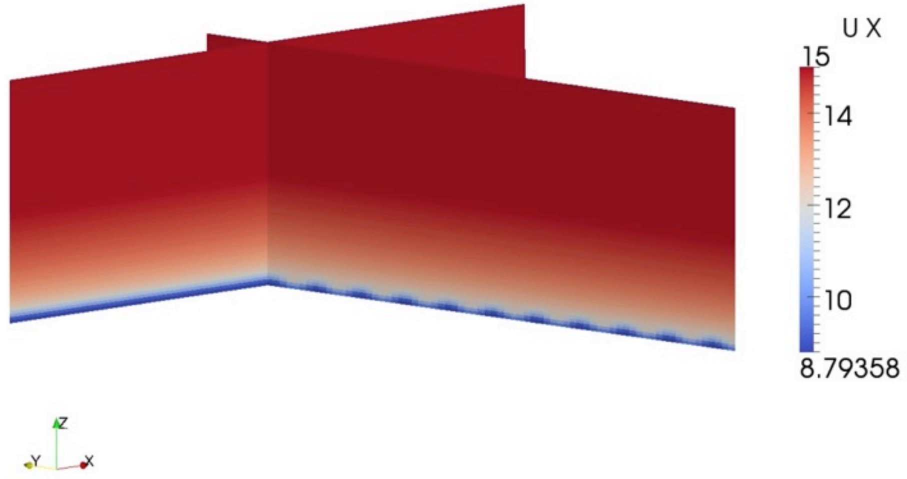
Initial Conditions for Precursor Flow

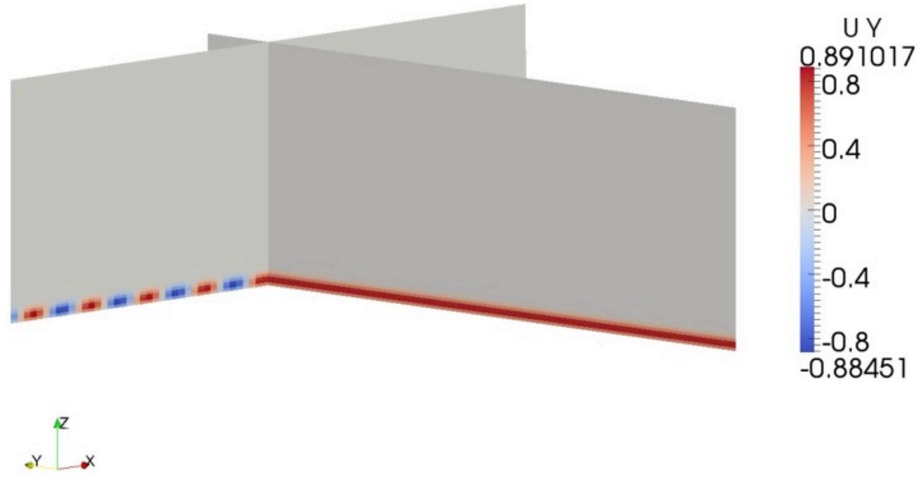
Initial conditions

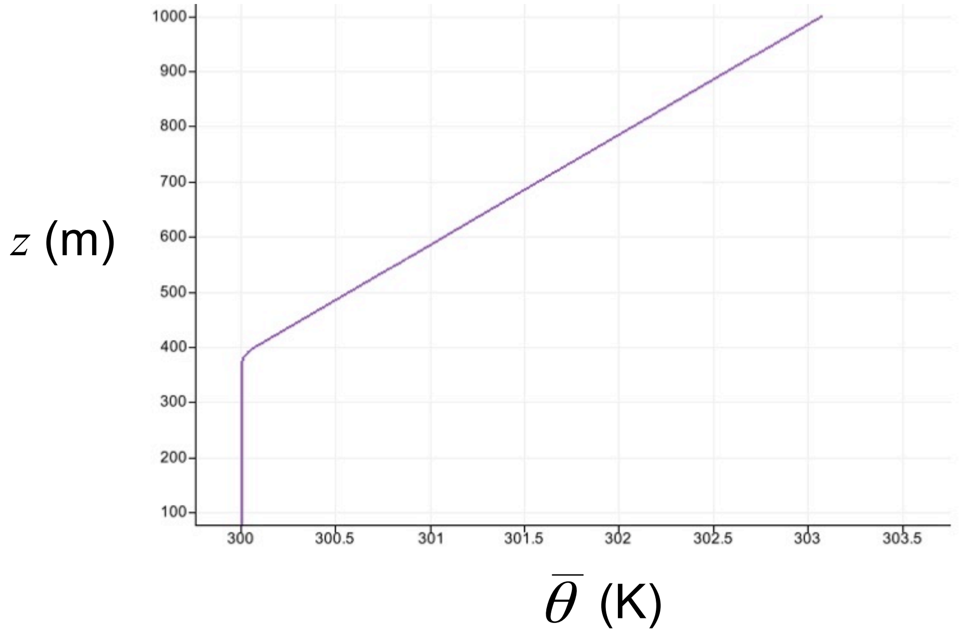
- **Velocity**
  - Constant at geostrophic wind speed or log law
  - Non-random, divergence-free perturbations added near surface to cause turbulence to quickly happen (similar to method used by DeVillier's in channel flow[1]).
- **Temperature**
  - Constant temperature (300K) up to some height, then temperature increases
  - This creates a capping inversion that caps the boundary layer and slows boundary layer vertical growth
- **Pressure variable**
  - Initialized to zero
- **Initial conditions set using "setABLFields" utility, but could use something like "funkySetFields"**

[1] De Villiers, E., "The Potential of Large Eddy Simulation for the Modeling of Wall Bounded Flows", PhD Thesis, Imperial College, London, 2006.

Initial Conditions for Precursor Flow







Numerical Schemes and Solver Algorithms

Numerical Scheme

- **ABLSolver uses the PISO[1] (Pressure Implicit Splitting Operation) – SIMPLE (Semi-Implicit Method for Pressure Linked Equations) to "implicitly" solve the momentum and pressure equation**
  - Predictor-Corrector approach with Rhie-Chow interpolation
  - I started with buoyantBoussinesqPimpleFoam, which only includes a temperature predictor
  - Having a temperature predictor only does not closely enough couple equations, so I also call the temperature equation in the corrector loop
  - Fixed the velocity overshoot problem seen at the top of the boundary layer in older ABLPisoSolver

- **I credit David Lapointe-Thériault from ETS in Montreal for figuring out the better temperature coupling through inclusion of T equation in corrector loop**

[1] R. I. Issa. Solution of the Implicitly Discretized Fluid Flow Equations by Operator-Splitting. *Journal of Computational Physics*, Vol. 62, 1985, pp. 40–65.

Numerical Scheme

- **Finite-volume formulation**
  - Linear interpolate of cell-center values to cell faces when needed
  - Equivalent to second-order central differencing
  - Sometimes pure linear differencing, which provides no artificial diffusion, causes instability, so a blend of linear plus a small amount of upwind is used.
  - Rhie-Chow[1]-like flux interpolation is used to avoid pressure-velocity decoupling

[1] C. M. Rhie and W. L. Chow. Numerical Study of the Turbulent Flow Past an Airfoil with Trailing Edge Separation. *AIAA Journal*, Vol. 21, No. 11, 1983, pp. 1552–1532.

Linear System Solvers

- **Velocity and Temperature**
  - Preconditioned Biconjugate Gradient (PBCG)
  - Diagonal incomplete LU matrix preconditioner
- **Pressure**
  - Preconditioned Conjugate Gradient (PCG)
  - Diagonal incomplete Cholesky or multigrid matrix preconditioner

OR

  - Geometric agglomerated algebraic multigrid solver
  - Diagonal incomplete Cholesky smoother

  - We recommend PCG with multigrid preconditioning

Solver Outputs

Solver Outputs

- **"averaging" file structure**
  - Within averaging directory are time directories corresponding to run start times. If you start a run at 0, there will be a "0" directory. If you restart a run at 1000, there will also be a "1000" directory.
  - Most files are structured as follows where each line represents a different time step, and starting at the third column, each column represents a horizontally-averaged value at a progressively greater height on the grid

$$\begin{array}{llllll}
time^0 & dt^0 & value_0 & value_1 & value_2 \ldots & value_J \\
time^1 & dt^1 & value_0 & value_1 & value_2 \ldots & value_J \\
 & & & & \ldots & \\
time^N & dt^N & value_0 & value_1 & value_2 \ldots & value_J
\end{array}$$

  - Heights corresponding the $value_0$ through $value_J$ are in the hLevelsCell file
    - hLevelsCell are cell-centered heights

Solver Outputs

- **"averaging" file structure**

Cell-center quantities	Description
T_mean	$\langle \bar{\theta} \rangle$
U_mean, V_mean, W_mean	$\langle \bar{u} \rangle$ $\langle \bar{v} \rangle$ $\langle \bar{w} \rangle$
uu_mean, vv_mean, ww_mean	$\langle u'u' \rangle$ $\langle v'v' \rangle$ $\langle w'w' \rangle$
uv_mean, uw_mean, vw_mean	$\langle u'v' \rangle$ $\langle u'w' \rangle$ $\langle v'w' \rangle$
wuu_mean, wvv_mean, www_mean	$\langle w'u'u' \rangle$ $\langle w'v'v' \rangle$ $\langle w'w'w' \rangle$
wuv_mean, wuw_mean, wvw_mean	$\langle w'u'v' \rangle$ $\langle w'u'w' \rangle$ $\langle w'v'w' \rangle$
Tu_mean, Tv_mean, Tw_mean	$\langle \theta'u' \rangle$ $\langle \theta'v' \rangle$ $\langle \theta'w' \rangle$

Solver Outputs

- **"averaging" file structure**

Cell-face quantities	Description
R11_mean, R22_mean, R33_mean	$\langle \tau_{11}^{D} \rangle$ $\langle \tau_{22}^{D} \rangle$ $\langle \tau_{33}^{D} \rangle$
R12_mean, R13_mean, R23_mean	$\langle \tau_{12}^{D} \rangle$ $\langle \tau_{13}^{D} \rangle$ $\langle \tau_{23}^{D} \rangle$
q1_mean, q2_mean, q3_mean	$\langle q_1 \rangle$ $\langle q_2 \rangle$ $\langle q_3 \rangle$

Guidelines for Use
Performance Metrics

Guidelines for Use

- **ABLSolver meant for flat terrain with "structured" mesh**
  - Only because it has built in planar averaging to give vertical mean profiles

- **Use ABLTerrainSolver if the bottom is not flat (exactly same solver, but takes time averages)**
  - At some point I want to make the averaging type function objects so that there is one ABL solver, and you choose either horizontal or time averaging function objects depending on the situation

- **Flat-bottom precursors should be run with periodic lateral boundaries**

- **The more stable the case, in general the longer the time to quasi-equilibrium (up to 50,000 s).**

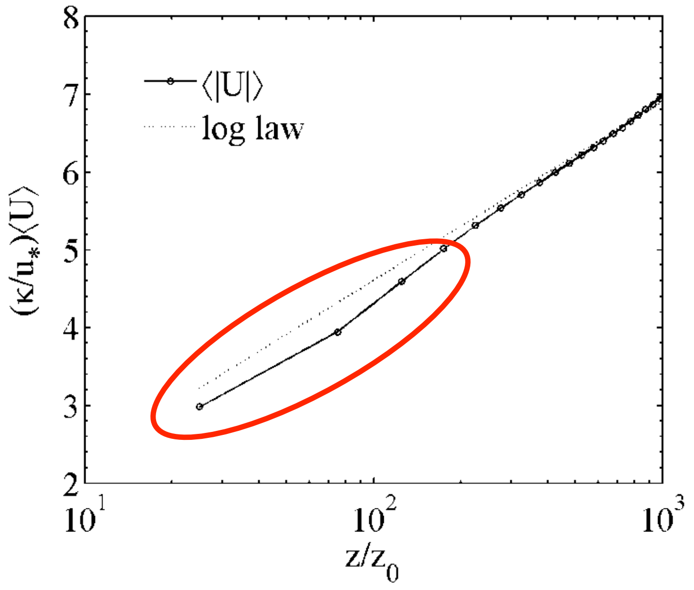
- **If the domain is the smallest recommended, drive hub-height wind at some angle no aligned with x-y; otherwise low-speed structures become "stuck" by periodicity and cycle through over and over.**

- $+x$ **must be east,** $+y$ **must be north,** $+z$ **must be up**

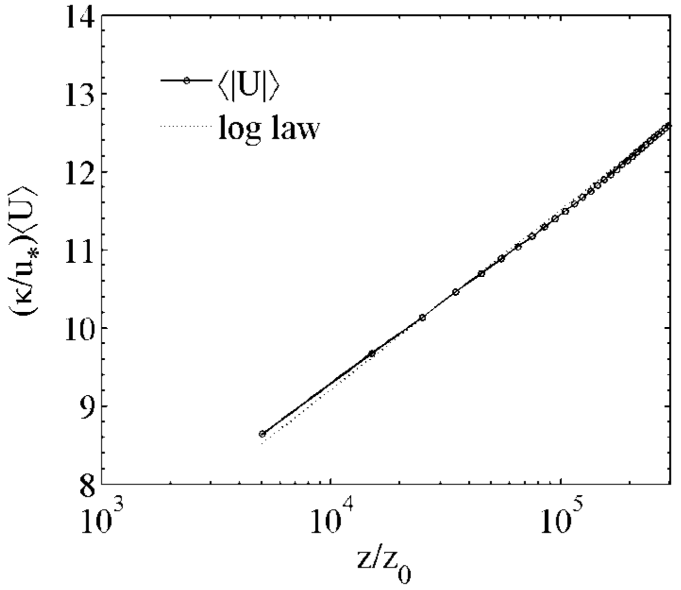
- **Must use adequate vertical grid resolution, small enough cell aspect ratio, and proper Smagorinsky constant to recover law-of-the-wall scaling**

Performance Metrics

- **Law-of-the-wall scaling**
  - This follows the work of Brasseur and Wei[1]
  - The problem:



Log-law mismatch



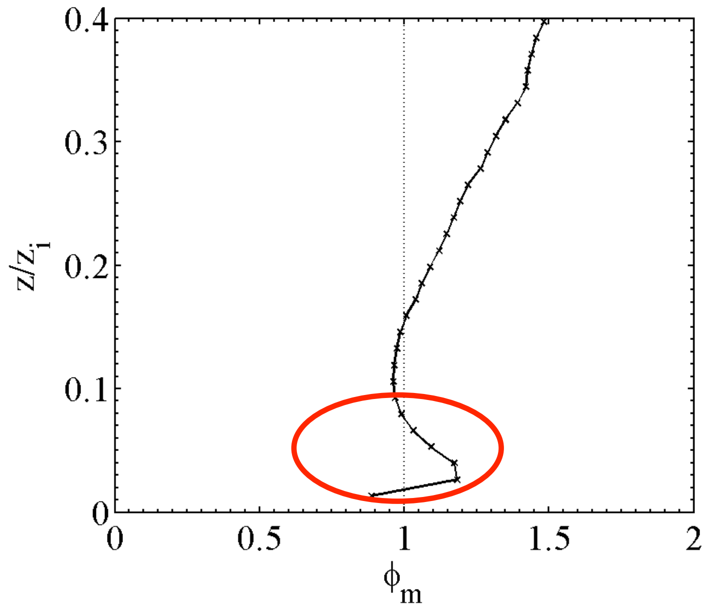
Improved log-law agreement

[1] J. Brasseur and T. Wei. Designing Large-Eddy Simulation of the Turbulent Boundary Layer to Capture Law-of-the-Wall Scaling, *Physics of Fluids*, Vol. 22, No. 2, 2010.

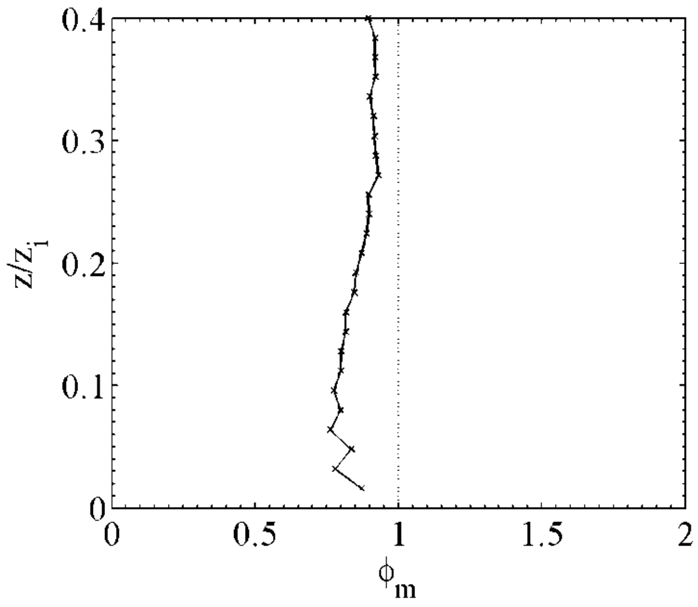
Performance Metrics

- **Law-of-the-wall scaling**
  - This follows the work of Brasseur and Wei[1]
  - The problem:

$$\phi_m = \frac{\kappa z}{u_*}\frac{\partial\langle U\rangle}{\partial z}$$



overshoot

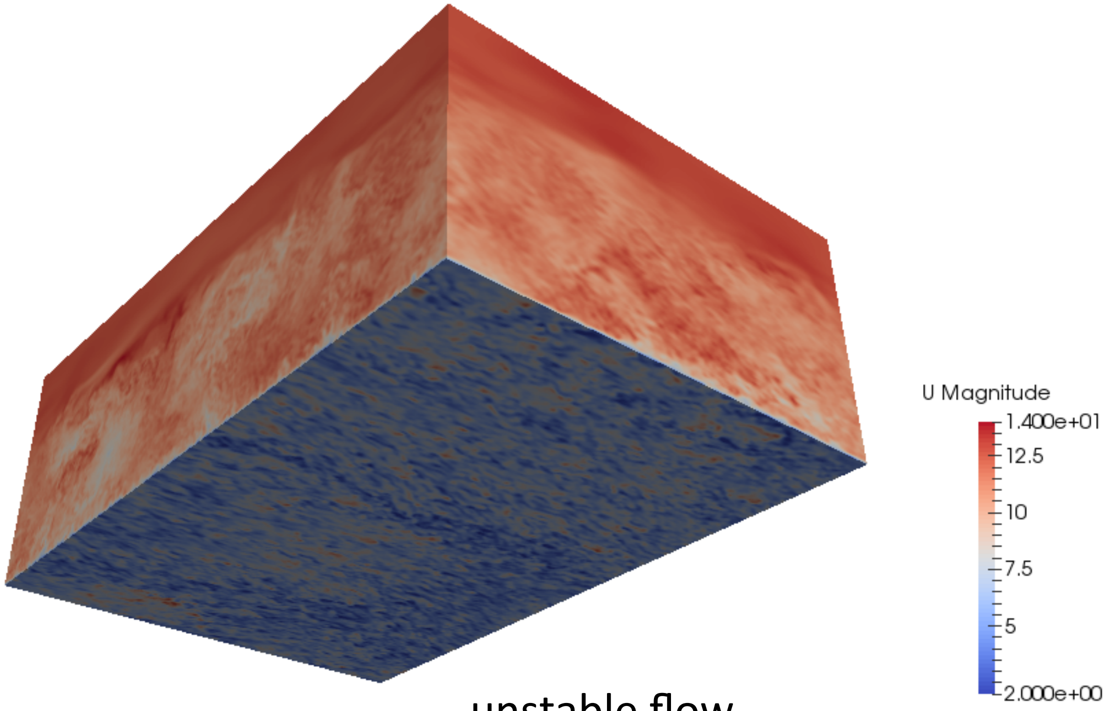


Improved log-law agreement

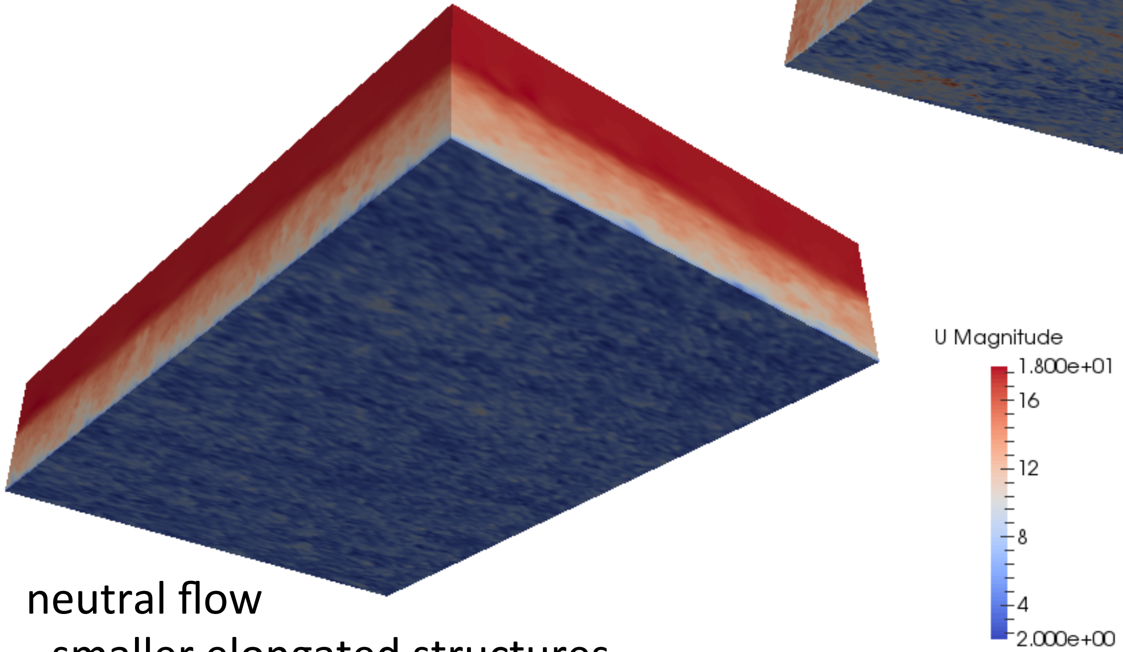
[1] J. Brasseur and T. Wei. Designing Large-Eddy Simulation of the Turbulent Boundary Layer to Capture Law-of-the-Wall Scaling, *Physics of Fluids*, Vol. 22, No. 2, 2010.

Performance Metrics

- **Flowfield visualization**



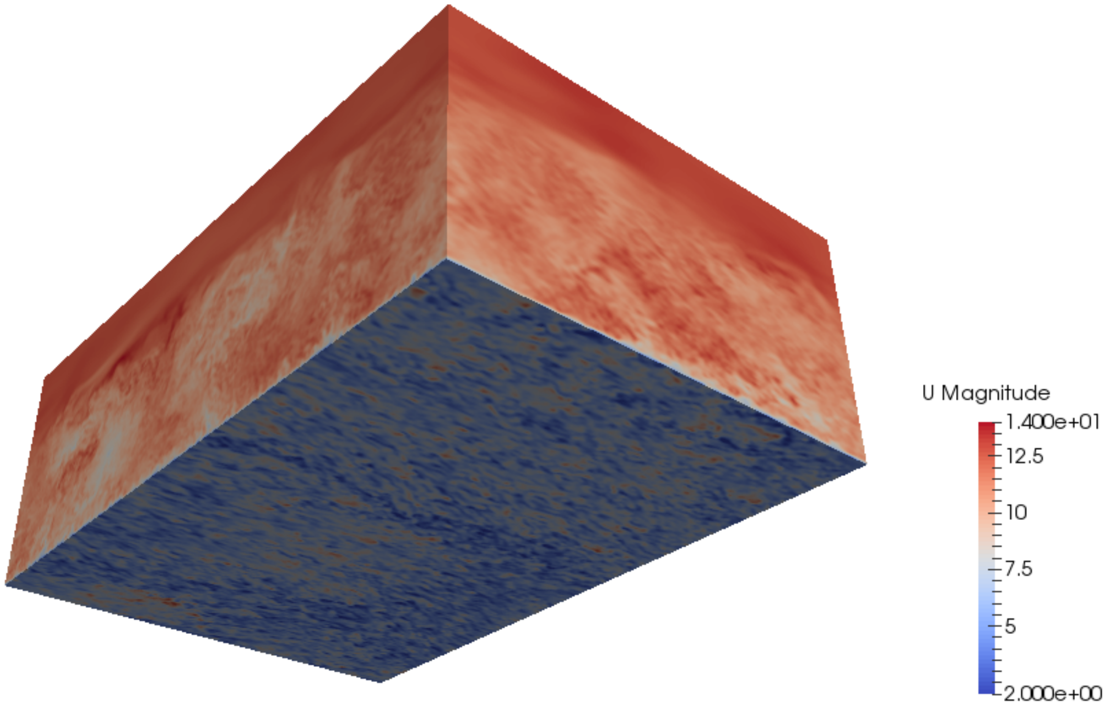
unstable flow
- larger flow structures

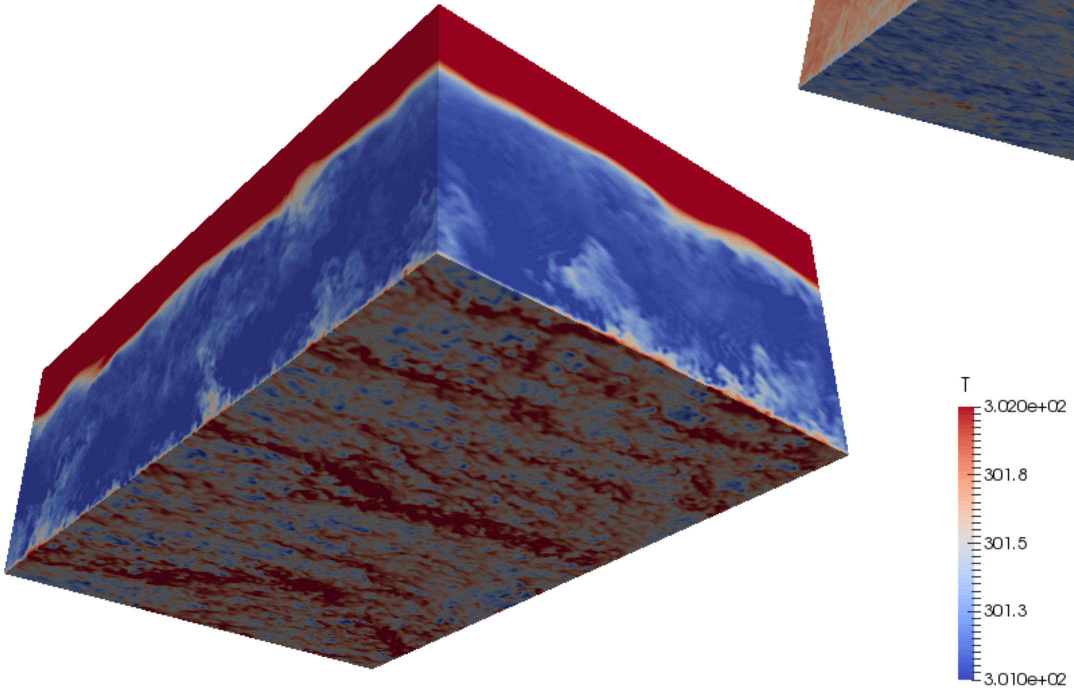


neutral flow
- smaller elongated structures

Performance Metrics

- **Flowfield visualization**

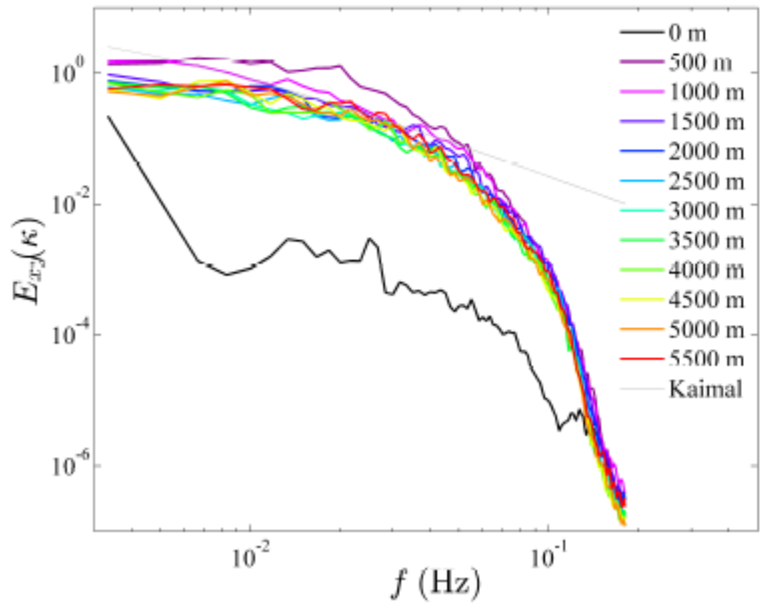


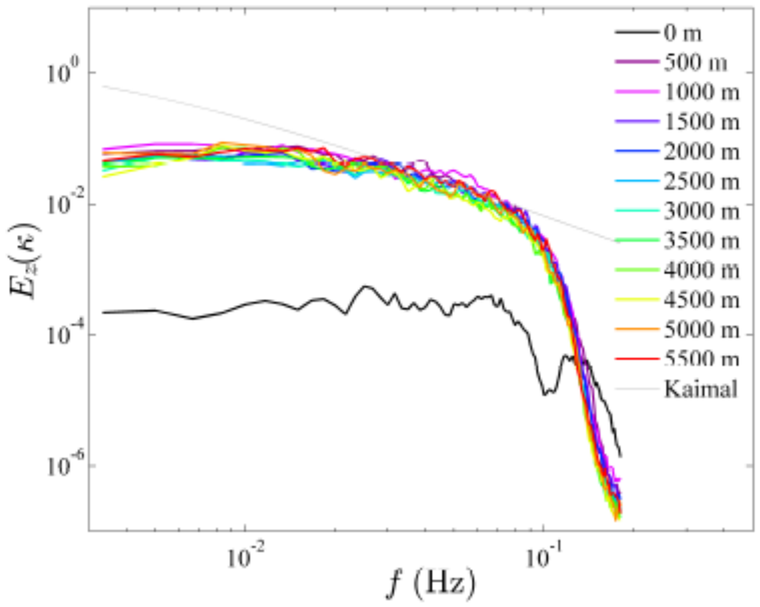


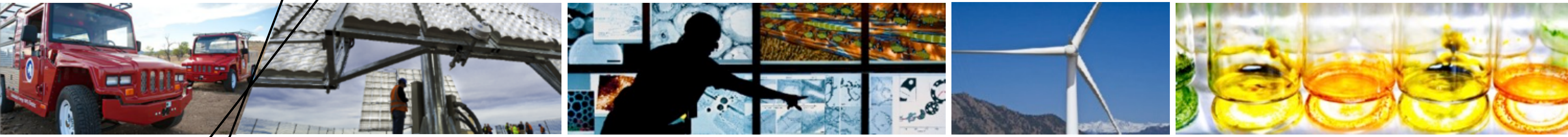
unstable flow
- velocity and temperature structures negatively correlated (i.e., where there are warm updrafts, low speed flow is being lifted into high speed flow

Performance Metrics

- **Spectra**



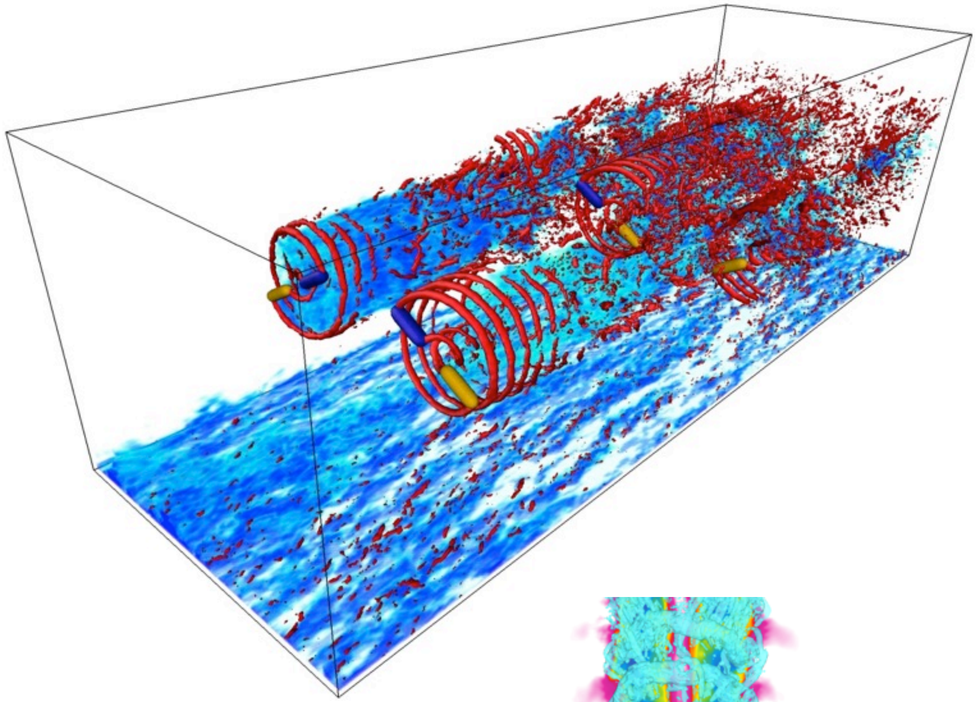


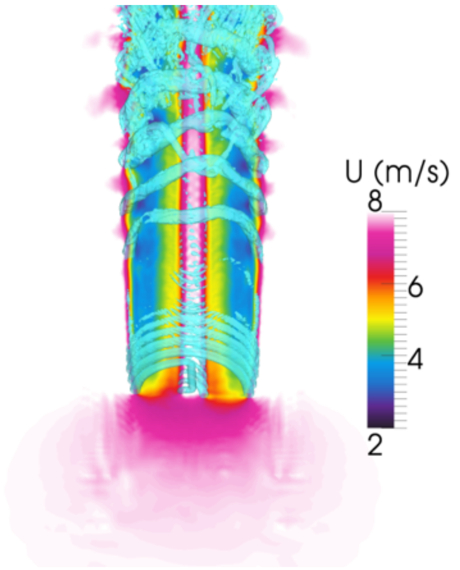


Actuator Line Turbine Model

Overview

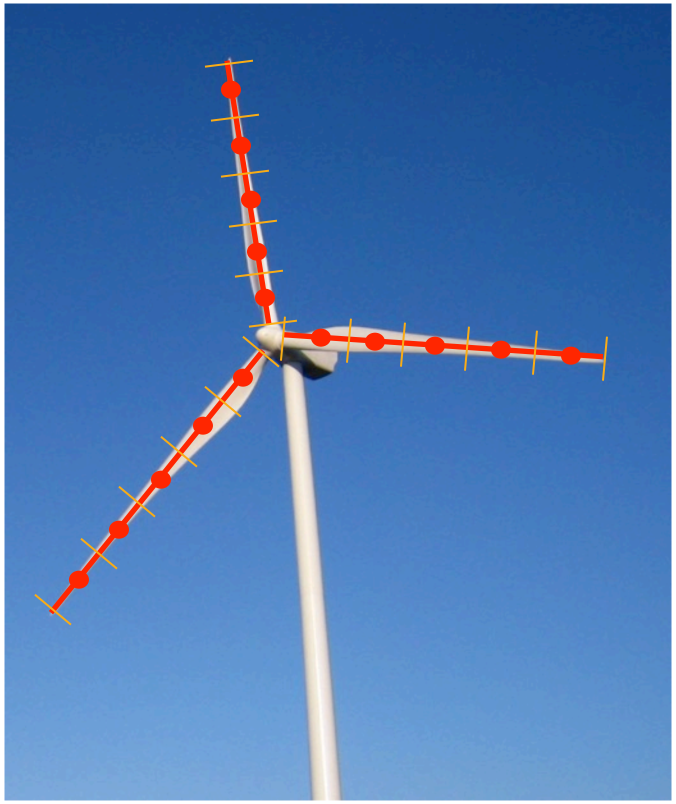
- Resolving turbine blade geometry with high-Re LES is infeasible
- An actuator approach does not require a very fine grid around turbine blades
- Creates wake, tip, root, and bound vortices
- Does not create blade boundary layer turbulence
- Depends upon airfoil look-up tables





Formulation

Theory



- Method of Sørensen and Shen[1]
- Blades discretized into spanwise sections of constant airfoil, chord, twist, oncoming wind
- Airfoil lookup tables used to calculate lift and drag at each actuator section
- Force on flow is equal and opposite to blade force
- Force is normalized and projected back to flow

$$\frac{\partial \bar{u}_i}{\partial t} + \frac{\partial}{\partial x_j}\left(\bar{u}_j \bar{u}_i\right) = -2\varepsilon_{i3k}\Omega_3 \bar{u}_k - \frac{\partial \tilde{p}}{\partial x_i} - \frac{\partial}{\partial x_i}\left\langle \frac{\bar{p}_0(x,y)}{\rho_0} \right\rangle - \frac{\partial}{\partial x_j}\left(\tau_{ij}^D\right) - g_3 z \frac{\partial}{\partial x_i}\left(\frac{\rho_k}{\rho_0}\right) + \frac{1}{\rho_0} f_i^T$$

[1] Sørensen, J. N. and Shen, W. Z., "Numerical Modeling of Wind Turbine Wakes", *Journal of Fluids Engineering* 124, 2002, pp. 393-399.

Theory

- **Force Projection**
  - How do you take force calculated at actuator line points and project it onto the CFD grid as a body force?
  - How do you smooth the force to avoid numerical oscillation?
  - Sørensen and Shen use a Gaussian projection

$$f_i^T(r) = \frac{F_i^A}{\varepsilon^3 \pi^{3/2}} \exp\left[-\left(\frac{r}{\varepsilon}\right)^2\right]$$

  - F_i^A is the actuator element force
  - f_i^T is the force field projected as a body force onto CFD grid
  - r is distance between CFD cell center and actuator point
  - ε controls Gaussian width.

Theory

- **Projection Width**
  - Troldborg[1] recommends $\varepsilon / \Delta x = 2$ where Δx is the grid cell length near actuator line
  - We found this to be the minimum in order to maintain an oscillation-free solution using central differences
  - We think ε should be tied to some physical blade length, like chord, but have not come up with a definitive guideline.
  - See the AIAA paper by Martínez et al.[2]
  - A good way to choose epsilon is to choose a wind speed/TSR and run a case and see how power compares to what it should be. If power is low, make epsilon bigger and vice versa, and try again. Repeat. Now you have 3 data points that should bracket the power you want. Fit a cubic spline to epsilon vs. power and find the epsilon that corresponds to the desired power. In our experience, this epsilon then holds for all other wind speeds and TSR.

[1] Troldborg, N., "Actuator Line Modeling of Wind Turbine Wakes", PhD Thesis, Technical University of Denmark, Lyngby, Denmark, 2008.
[2] Martinez, L. A., Leonardi, S., Churchfield, M. J., Moriarty, P. J., "A Comparison of Actuator Disk and Actuator Line Wind Turbine Models and Best Practices for Their Use", AIAA Paper 2012-900, Jan. 2012.

Modeling the Control System

Control System Functions

- **Generator Torque Control**
  - 5 region control like NREL 5MW (see 5MW Reference Turbine Report)
  - Generator speed vs. generator torque lookup table
- **Pitch Control**
  - PID, based on NREL 5MW Reference Turbine Report
  - Can provide P, I, and D gains, but must compute those gains following NREL 5MW Reference Turbine Report
- **Yaw Control**
  - Not yet implemented, but coming soon
- **Can be run in a FAST-coupled mode (we will discuss this later in the tutorial)**

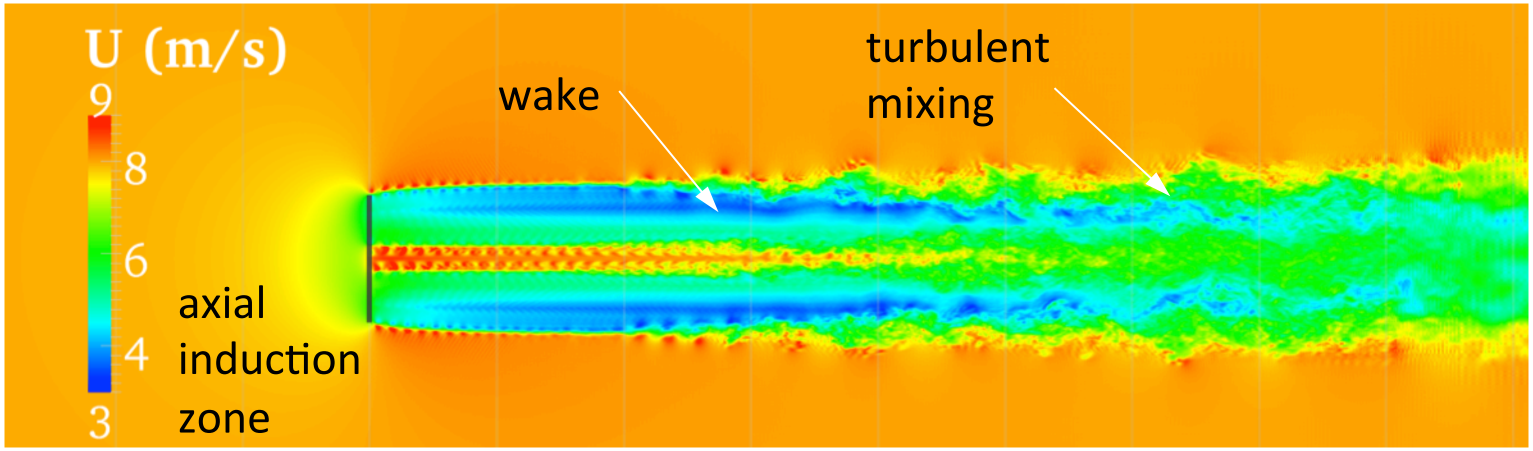
---

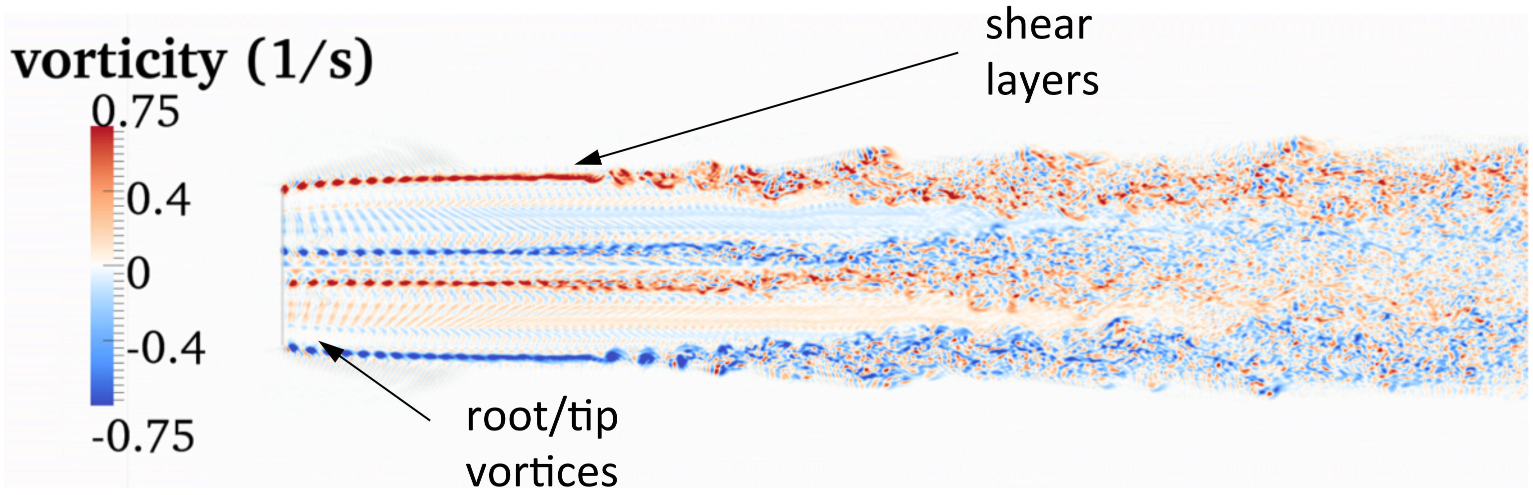
[1]J. Jonkman, S. Butterfield, W. Musial, and G. Scott, "Definition of a 5-MW Reference Wind Turbine for Offshore System Development," NREL Report TP-500-38060, Feb. 2009

Model Output

Sample Output

High-resolution simulation of a single wake in uniform, non-turbulent flow





Actuator Line Model Outputs

- **Solution files (inside time directories)**
  - bodyForce: body force projected onto flow field

- **“turbineOutput” directory**
  - Outputs various turbine information such as power, torque, rotor speed, etc.
  - Outputs information at each blade point such as angle of attack, velocity magnitude, lift, drag, etc.

Actuator Line Model Outputs

- **“turbineOutput” file structure**
    - Within turbineOutput directory are time directories corresponding to run start times. If you start a run at 0, there will be a “0” directory. If you restart a run at 1000, there will also be a “1000” directory.
        - Within the specific time directories are a files for global turbine data files for quantities like power, torque, rotor speed, etc.
        - Also there are files for blade local quantities like lift, drag, angle of attack, etc. vs. span.

Actuator Line Model Outputs

- **Global quantity file structure**

$\texttt{turbine}^{0}$ $\texttt{time}^{0}$ $\texttt{dt}^{0}$ $\texttt{value}$
$\texttt{turbine}^{1}$ $\texttt{time}^{0}$ $\texttt{dt}^{0}$ $\texttt{value}$
…
$\texttt{turbine}^{M}$ $\texttt{time}^{1}$ $\texttt{dt}^{0}$ $\texttt{value}$

$\texttt{turbine}^{0}$ $\texttt{time}^{1}$ $\texttt{dt}^{1}$ $\texttt{value}$
$\texttt{turbine}^{1}$ $\texttt{time}^{1}$ $\texttt{dt}^{1}$ $\texttt{value}$
…
$\texttt{turbine}^{M}$ $\texttt{time}^{1}$ $\texttt{dt}^{1}$ $\texttt{value}$

…

$\texttt{turbine}^{0}$ $\texttt{time}^{N}$ $\texttt{dt}^{N}$ $\texttt{value}$
$\texttt{turbine}^{1}$ $\texttt{time}^{N}$ $\texttt{dt}^{N}$ $\texttt{value}$
…
$\texttt{turbine}^{M}$ $\texttt{time}^{N}$ $\texttt{dt}^{N}$ $\texttt{value}$

Actuator Line Model Outputs

- **Blade radius dependent file structure**

turbine^0 blade^0 time^0 dt^0 value$_0$ value$_1$ value$_2$ … value$_J$
turbine^0 blade^1 time^0 dt^0 value$_0$ value$_1$ value$_2$ … value$_J$
turbine^0 blade^2 time^0 dt^0 value$_0$ value$_1$ value$_2$ … value$_J$

turbine^1 blade^0 time^0 dt^0 value$_0$ value$_1$ value$_2$ … value$_J$
turbine^1 blade^1 time^0 dt^0 value$_0$ value$_1$ value$_2$ … value$_J$
turbine^1 blade^2 time^0 dt^0 value$_0$ value$_1$ value$_2$ … value$_J$
…

turbine^M blade^0 time^0 dt^0 value$_0$ value$_1$ value$_2$ … value$_J$
turbine^M blade^1 time^0 dt^0 value$_0$ value$_1$ value$_2$ … value$_J$
turbine^M blade^2 time^0 dt^0 value$_0$ value$_1$ value$_2$ … value$_J$

…

turbine^0 blade^0 time^N dt^N value$_0$ value$_1$ value$_2$ … value$_J$
turbine^0 blade^1 time^N dt^N value$_0$ value$_1$ value$_2$ … value$_J$
turbine^0 blade^2 time^N dt^N value$_0$ value$_1$ value$_2$ … value$_J$

turbine^1 blade^0 time^N dt^N value$_0$ value$_1$ value$_2$ … value$_J$
turbine^1 blade^1 time^N dt^N value$_0$ value$_1$ value$_2$ … value$_J$
turbine^1 blade^2 time^N dt^N value$_0$ value$_1$ value$_2$ … value$_J$
…

turbine^M blade^0 time^N dt^N value$_0$ value$_1$ value$_2$ … value$_J$
turbine^M blade^1 time^N dt^N value$_0$ value$_1$ value$_2$ … value$_J$
turbine^M blade^2 time^N dt^N value$_0$ value$_1$ value$_2$ … value$_J$

Actuator Line Model Outputs

Global turbine quantities	Description
powerRotor	Rotor power/density (W)
rotSpeed	Rotor speed (rpm)
thrust	Thrust (N)
torqueRotor	Rotor torque (N-m)
torqueGen	Generator torque (N-m)
azimuth	Rotor azimuth angle (degrees)
nacYaw	Nacelle yaw angle (degrees)
pitch	Blade collective pitch (degrees)

Actuator Line Model Outputs

Blade Local quantities	Description
alpha	Angle of attack (degrees)
axialForce	Force along rotor shaft axis (N)
Cd	Coefficient of drag
Cl	Coefficient of lift
drag	Drag force (N)
lift	Lift force (N)
tangentialForce	Force in rotor rotation tangential direction (N)
Vaxial	Component of velocity along rotor shaft axis (m/s)
Vradial	Component of velocity along blade radius (m/s)
Vtangential	Component of velocity in rotation tangential direction (m/s)
x, y, z	Actuator point position in space (m)

General Recommendations

Guidelines for Use

- $+x$ must be east, $+y$ must be north, $+z$ must be up
- Use at least 20 CFD grid cells across the rotor diameter
- Use at least 50 CFD grid cells across the rotor if you want to well resolve tip/root vortices
- Set epsilon based on cubic fit approach, or for large utility-scale turbines, ε = 0.035D
- Time step should not exceed that which allows blade tip to pass through more than one cell per time step for smoothness of solution

Software Implementation

Implementation

- **Turbine model implemented as a class**
  - "horizontalAxisWindTurbinesALM"
  - See src/turbineModels/horizontalAxisWindTuribinesALM
- **Any solver can be modified to contain an object of the class**
- **That object is the entire turbine array**

Implementation

- **Modifying pisoFoam to include turbine class**
  - Add this to createFields.H to declare object of turbine class

```
// Create an object of the horizontalWindTurbineArray class if there
// is to be a turbine array
//
turbineModels::horizontalAxisWindTurbinesALM turbines(U);
```

  - Add this to the includes part of the solver code

```
#include “horizontalAxisWindTurbinesALM.H”
```

  - Add this line to solver code momentum equation to apply forces

```
fvVectorMatrix UEqn
(
       fvm::ddt(U)
     + fvm::div(phi, U)
     + turbulence->divDevReff(U)
     - turbines.force()
```

  - Add this line at the beginning or end of the time loop to advance the turbine one time step

```
turbines.update();
```

Implementation

- **Make/options file needs to be modified**

```
 EXE_INC = \
    -I$(LIB_SRC)/turbulenceModels/incompressible/turbulenceModel \
    -I$(LIB_SRC)/transportModels \
    -I$(LIB_SRC)/transportModels/incompressible/singlePhaseTransportModel \
    -I$(LIB_SRC)/finiteVolume/lnInclude \
    -I$(WM_PROJECT_USER_DIR)/src/turbineModels/lnInclude

EXE_LIBS
    -L$(FOAM_USER_LIBBIN) \
    -lincompressibleTurbulenceModel \
    -lincompressibleRASModels \
    -lincompressibleLESModels \
    -lincompressibleTransportModels \
    -lfiniteVolume \
    -lmeshTools \
    -llduSolvers \
    -luserTurbineModels
```

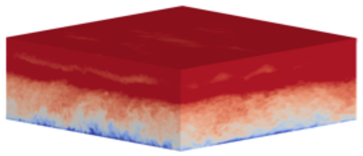
Coupling to NREL's FAST Structural and System Dynamics Tool

Coupling FAST to OpenFOAM

- **NREL's FAST[1] (Fatigue, Aerodynamics, Stress, and Turbulence) tool is a model for wind turbine structural, aero, and system dynamics**
- **Its aerodynamics part is through blade element momentum theory (BEM)**
- **Here, we coupled FAST to the actuator line model**
- **The "momentum" part of BEM is replaced by CFD**
  - CFD feeds FAST inflow information at blade elements
  - Aerodynamic forces computed by look-up table ("blade element" theory--just like normal actuator line)
  - Turbine structural and system response computed
  - Aerodynamic forces fed back to CFD

[1] Jonkman, J. and Buhl, M., FAST User's Guide, NREL/EL-500-38230, NREL technical report, 2005. Accessible at: http://wind.nrel.gov/designcodes/simulators/fast/FAST.pdf

Coupling FAST to OpenFOAM



OpenFOAM

Multiple-Turbine capability

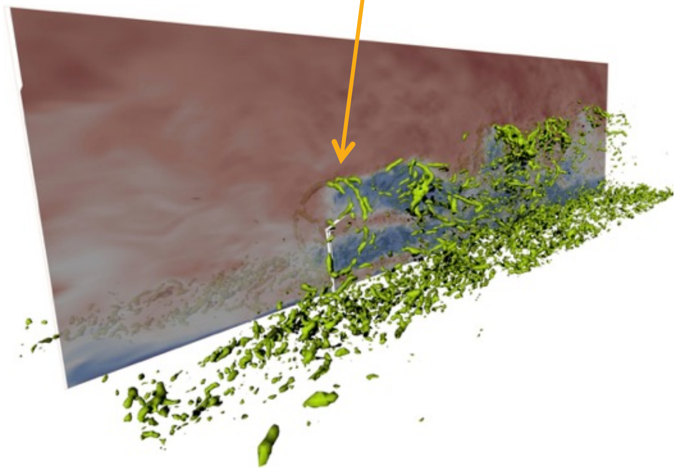
(NREL aero-elastic code)

Apply rotor body forces and compute flow field

velocity

Apply blade velocities, and compute structural response and positions

aeroforces w/ blade coord.
in actuator line representation



Implementation

- **Similar to standard actuator line**
- **Turbine model implemented as a class**
  - “horizontalAxisWindTurbinesFAST”
  - See src/fastturb/horizontalAxisWindTuribinesFAST
- **Any solver can be modified to contain an object of the class**
- **That object is the entire turbine array**

Lift/Drag Table Preparation

Lift/Drag Table Preparation

- **Interpolation between airfoil types or thicknesses**
  - It is best to use some sort of shape preserving interpolation

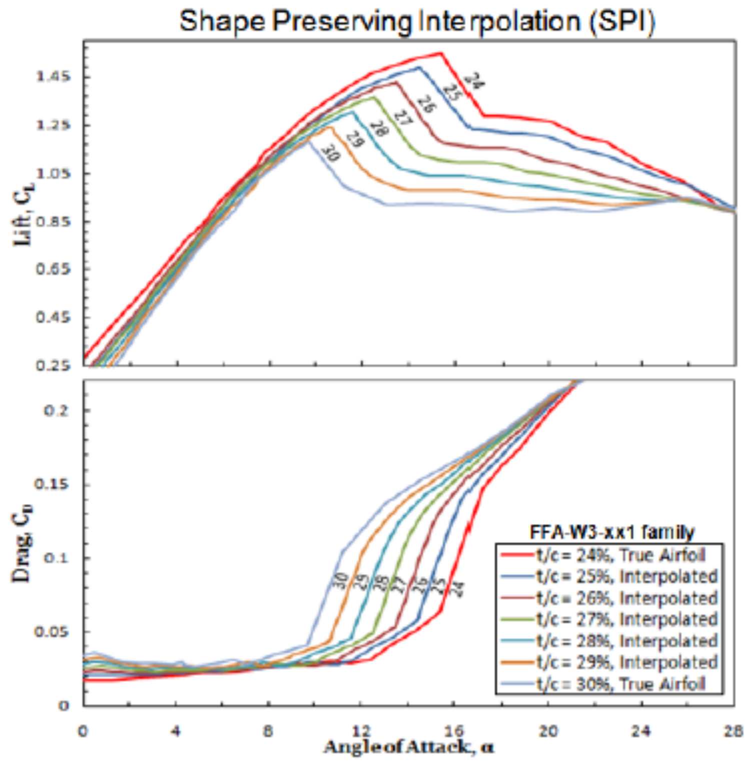


Figure 5: Interpolation of lift and drag coefficients with percent thickness as the variable using Shape Preserving Interpolation (SPI).

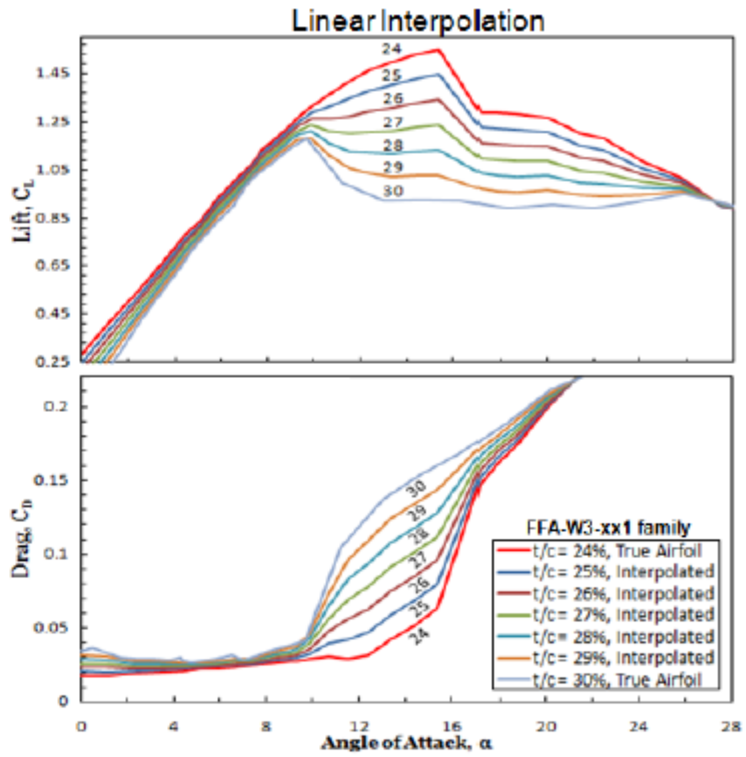
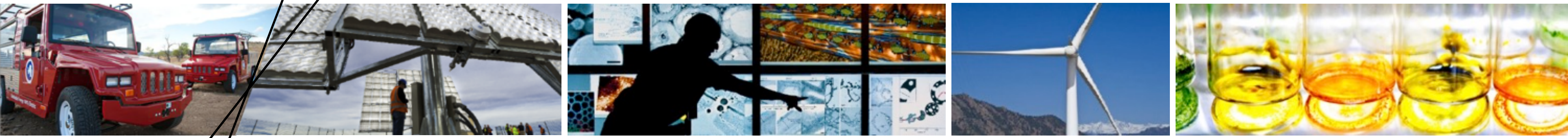


Figure 4: Interpolation of lift and drag coefficients with percent thickness as the variable using simple linear interpolation.

From the NREL HarpOpt User's Manual, also in the SOWFA-Tools Matlab directory

Correction for 3D Effects

- **We have tried both the Du-Selig (lift), Eggars (drag) method and Snel's (lift only) method.**
- **Sometimes the Du-Selig and Eggars method gives strange results**
- **Snel's method seems more robust**
  - Apply the correction up to 25° angle of attack, and then blend to no correction smoothly by 40°
  - See the "correctSnel.m" file in the SOWFA-Tools directory.

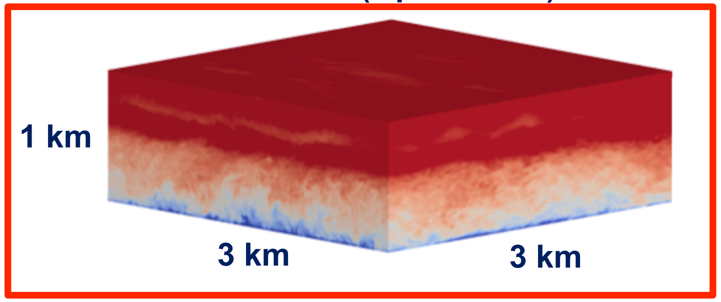


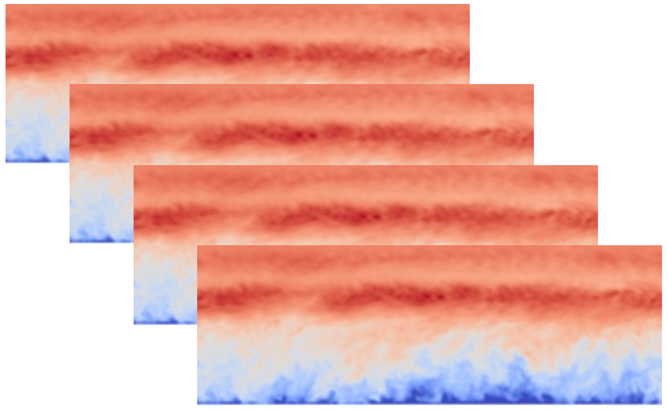
Wind Plant Solver

Wind Plant Simulation

- **Combination of the elements discussed above**

“Precursor” atmospheric simulation (OpenFOAM)



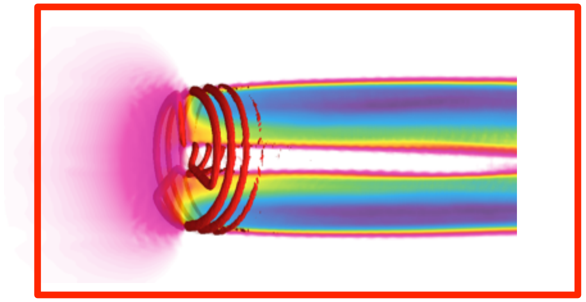


Save planes of data every N time steps

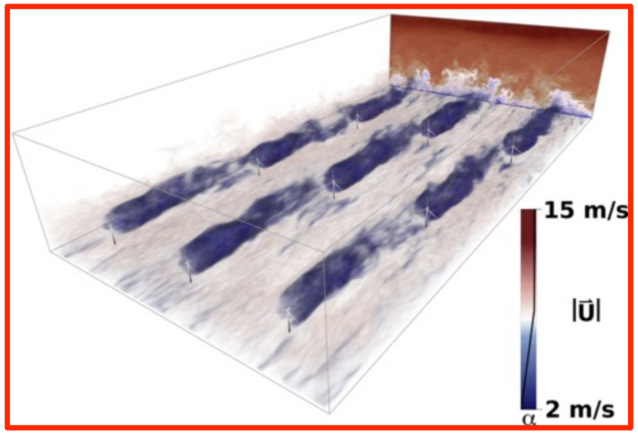
Initialize wind farm domain with precursor volume field

Use saved precursor data as inflow boundary conditions

Actuator line turbine aerodynamics models (coupled with NREL’s FAST turbine dynamics model)



Wind farm simulation (OpenFOAM)



windPlantSolver

- **It is basically the ABLSolver with the horizontalAxisALM class included (but can still be used on flat terrain—it just does time averages)**

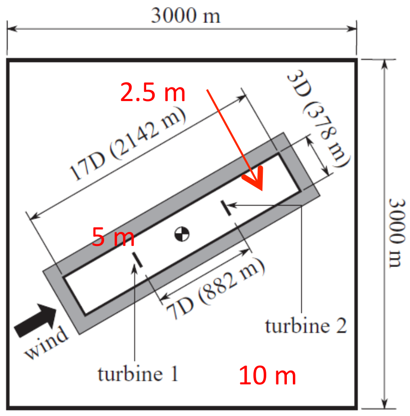
Output

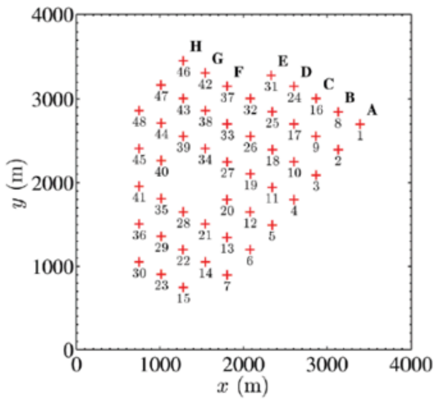
- **All the turbine information**
- **Instantaneous Fields**
  - U, T, p, u', T'
- **Mean Fields**
  - Umean, Tmean
- **Correlation Fields**
  - $\langle u'_i u'_j \rangle$, $\langle T' u'_j \rangle$

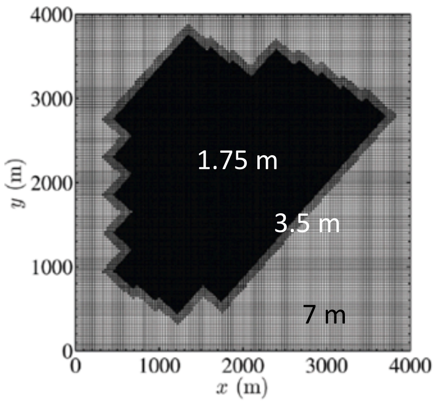
Guidelines for Use

- **Make sure domain boundaries have either predominant inflow or outflow**
    - Remember that with Coriolis, wind changes directions with altitude
    - Possible to have wind flowing in near ground and flowing out above
    - We do not have a good boundary condition for that case

- **Use local mesh refinement around the turbines**
    - but do it gently (i.e. give the turbulence time to cascade down before going to the next local refinement region)
    - We use toposet (with rotatedBox option) and refineMesh







Guidelines for Use

- We generally use a time step such that the actuator line tip does not travel through more than one cell per time step
- Can use larger time steps with actuator disk.
- The grid cell count can get large quickly. Try to estimate before refining the mesh as to how large it will be.

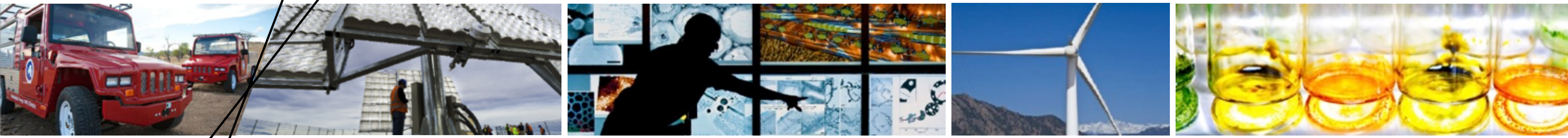
Some References

Churchfield, M. J., Lee, S., Michalakes, J., and Moriarty, P. J., “A Numerical Study of the Effects of Atmospheric and Wake Turbulence on Wind Turbine Dynamics,” Journal of Turbulence, Vol. 13, No. 14, pp. 1-32, 2012.

Churchfield, M. J., Lee, S., Moriarty, P. J., Martinez, L. A., Leonardi, S., Vijayakumar, G., and Brasseur, J. G., “A Large-Eddy Simulation of Wind-Plant Aerodynamics,” AIAA Paper AIAA-2012-537, 2012.

Lee, S., Churchfield, M. J., Moriarty, P. J., Jonkman, J., “Atmospheric and Wake Turbulence Impacts on Wind Turbine Fatigue Loading,” AIAA Paper AIAA-2012-540, 2012.

Martinez, L. A, Leonardi, S., Churchfield, M. J., Moriarty, P. J., “A Comparison of Actuator Disk and Actuator Line Wind Turbine Models and Best Practices for Their Use,” AIAA Paper AIAA-2012-900, 2012.



Compiling

Compiling the codes

- **Make sure you have OpenFOAM 2.0 or higher installed (but SOWFA will not work with OpenFOAM 3.0 and greater yet)**

- **Download the SOWFA codes from our git repository**
  - https://github.com/NREL/SOWFA
    - git clone https://github.com/NREL/SOWFA
    - cd SOWFA
    - git pull

- **I keep a clean SOWFA directory, but do a copy of directory structure with soft linked files to my user-2.0.x directory**
  - cp –rs /home/mchurchf/OpenFOAM/SOWFA /home/mchurchf/OpenFOAM/mchurchf-2.0.x
  - cd mchurchf-2.0.x

- **Then in mchurchf-2.0.x, I run**
  - ./Allwclean
  - ./Allwmake

- **In this way, you can have once central SOWFA directory that you periodically git pull to, and have multiple different compiled versions of it (i.e, you may have user-2.0.x, user 2.2.x, and user-2.3.x directories that all link back SOWFA, but each run with the different versions of OpenFOAM, and which also contain your own non-SOWFA custom files)**

- **See the README files**