WETO Software Stack User Workshops
High Fidelity Modeling

November 14, 2024

Rafael Mudafort
Pietro Bortolotti
Garrett Barter
Michael Sprague
Michael Kuhn
Marc Henry de Frahan
Jon Rood
Eliot Quon

Agenda

Section	Duration	Time	Speaker
Intro	5’	0:00 - 0:05	Rafael Mudafort
WETO Stack Overview	10’	0:05 - 0:15	Rafael Mudafort
ExaWind	10’	0:15 - 0:25	Mike Sprague
ExaWind:OpenTurbine	5’	0:25 – 0:30	Mike Sprague
ExaWind:AMR Wind	10’	0:30 - 0:40	Michael Kuhn
ExaWind Software & Performance	10’	0:40 - 0:50	Marc Henry de Frahan and Jon Rood
ERF	10’	0:50 - 1:00	Eliot Quon
Polls / Community discussion	**40’**	**1:00 - 1:40**	**YOU**
Wrap up	5’	1:40 - end	Rafael Mudafort

Holistic Modeling Project

WETO Software Portfolio Coordination

US DOE & Lab-based Wind Research Projects

WETO invests in wind energy **software** that enables and accelerates the innovations needed to advance wind energy.

NREL's active WETO projects

Study on the Potential Application of Additive Manufacturing in Wind Turbine Components and Tooling
Enabling Larger Rotors Through Modular, Customizable, Inflatable Blades
Eagle Topic Area 3 Funding Opportunity Announcement (FOA) Support
Co-Simulation Study and Control of a Wind Farm for Conversion Services
Continental-Scale Transmission Modeling Methods for Grid Integration Analysis
Atmosphere to Electrons to Grid (A2e2g)
Fusion Joining of Thermoplastic Composites Using Energy Efficient Processes (TCF)
Automating In-Situ Grinding and Repair for Thermoplastic Blades
Codesign and Intelligent Approaches for Cost-Effective Operation and Maintenance of Generators and Power Converters
Modeling and Validation for Offshore Wind
Wind Power as Virtual Synchronous Generation (WindVSG)
Technology Development and Innovation to Address Operational Challenges
Evaluating Deterrent Stimuli for Increasing Species-Specific Effectiveness of an Advanced Ultrasonic Acoustic Deterrent
North American Renewable Integration Study
High-Fidelity Modeling
Wind Turbine Drivetrain Reliability Assessment and Remaining Useful Life Prediction (TCF)
Enabling Autonomous Wind Plants through Consensus Control (TCF)
North American Energy Resiliency Model (NAERM)
Big Adaptive Rotor
Energy Sector Modeling and Impacts Analysis
Floating Downwind Turbines: A Conceptual System-Level Design and Feasibility Study for U.S. Waters
Wind Standards Development
Multiscale Integration of Control Systems (EMS/DMS/BMS)
Advanced Modeling, Dynamic Stability Analysis, and Mitigation of Control Interactions in Wind Power Plants
Wind Grid Integration Stakeholder Engagement
Atmosphere to Electrons (A2e) Performance Risk, Uncertainty and Finance (PRUF) Analysis Support
Working Together to Resolve Environmental Effects of Wind Energy (WREN)
High-Fidelity Modeling Toolkit for Wind Farm Development

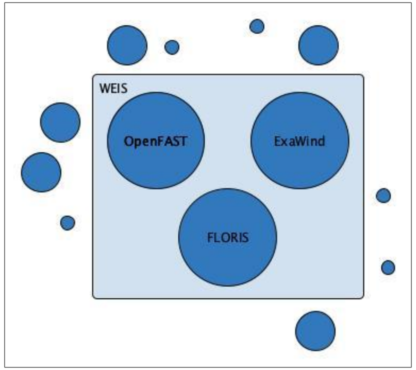
Argonne National Laboratory | Berkeley Lab Lawrence Berkeley National Laboratory | Lawrence Livermore National Laboratory | NREL National Renewable Energy Laboratory | Pacific Northwest National Laboratory | Sandia National Laboratories

https://www.energy.gov/eere/wind/wind-energy-technologies-office-projects-map

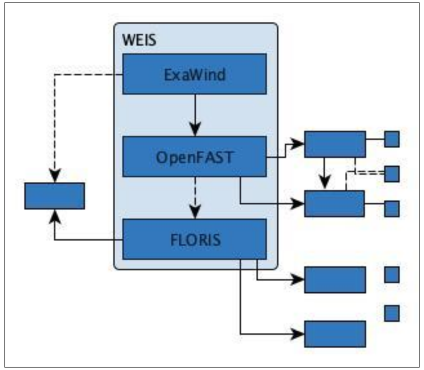
Holistic Modeling Project

Objective

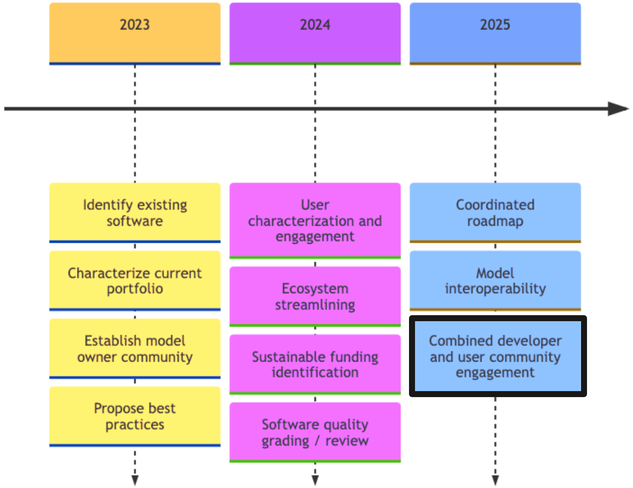
Past: Loose collection of software



Future: Cohesive software stack



Project Timeline

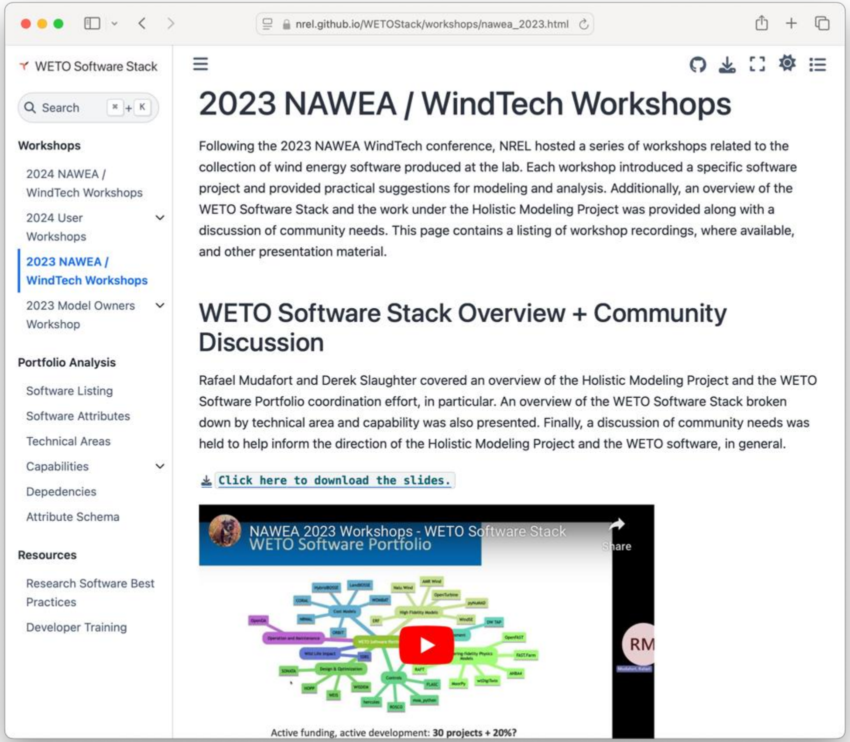


WETO Stack Dashboard

nrel.github.io/WETOStack

WETO Stack dashboard will provide the following information:
- The WETO-supported tools that enable a given task
- The state (maturity, stability) of included software
- The current and future capabilities
- Updates and community-focused materials

WETO Software Stack



WETO Stack Dashboard

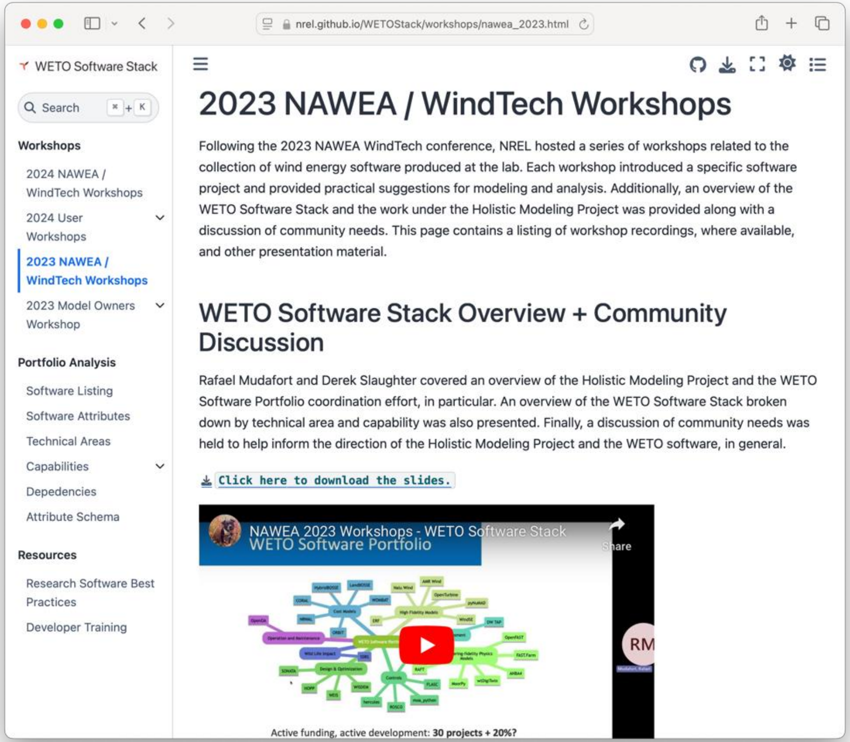
nrel.github.io/WETOStack

WETO Stack dashboard will provide the following information:

- The WETO-supported tools that enable a given task
- The state (maturity, stability) of included software
- The current and future capabilities
- Updates and community-focused materials

Current Contents:

- Workshop recordings and reports



WETO Stack Dashboard

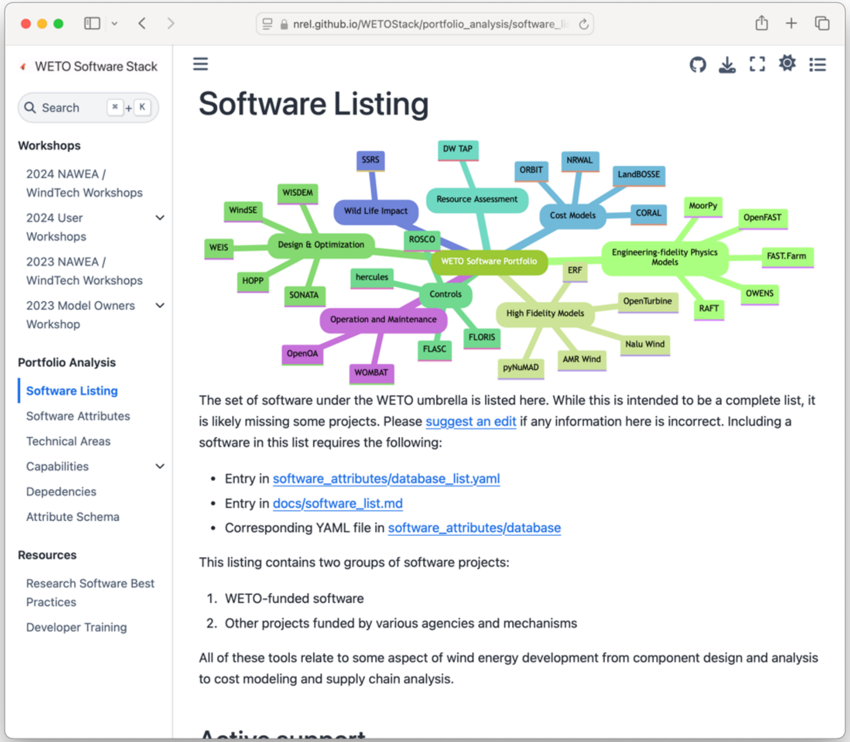
nrel.github.io/WETOStack

WETO Stack dashboard will provide the following information:

- The WETO-supported tools that enable a given task
- The state (maturity, stability) of included software
- The current and future capabilities
- Updates and community-focused materials

Current Contents:

- Workshop recordings and reports
- Software Listing: active, inactive, and “other status” software

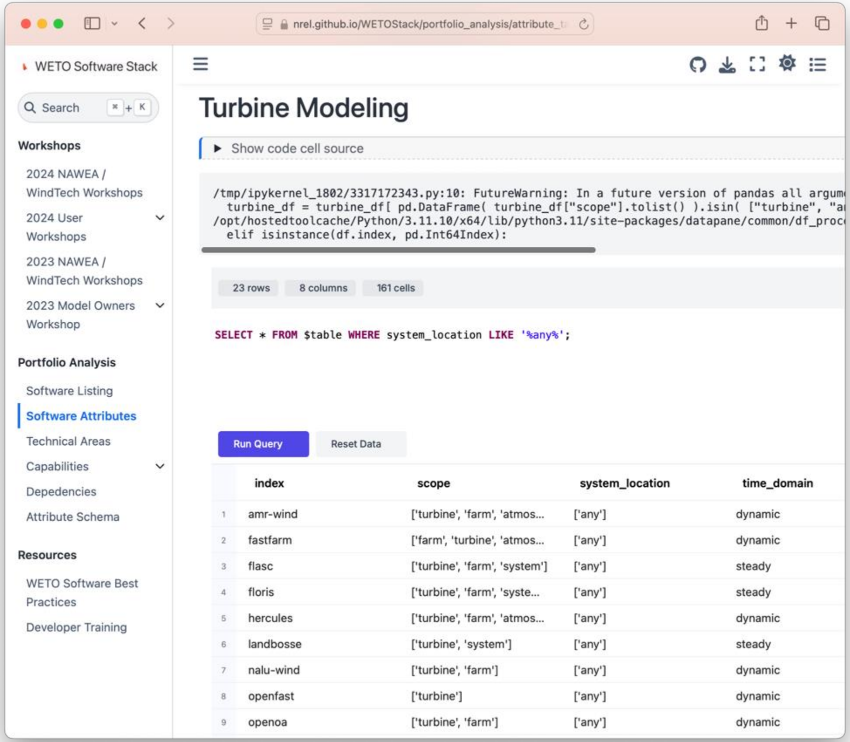


WETO Stack Dashboard

nrel.github.io/WETOStack

WETO Stack dashboard will provide the following information:
- The WETO-supported tools that enable a given task
- The state (maturity, stability) of included software
- The current and future capabilities
- Updates and community-focused materials

Current Contents:
- Workshop recordings and reports
- Software Listing: active, inactive, and "other status" software
- Software Attributes: tabulated data describing each software, defined by an Attribute Schema



WETO Stack Dashboard

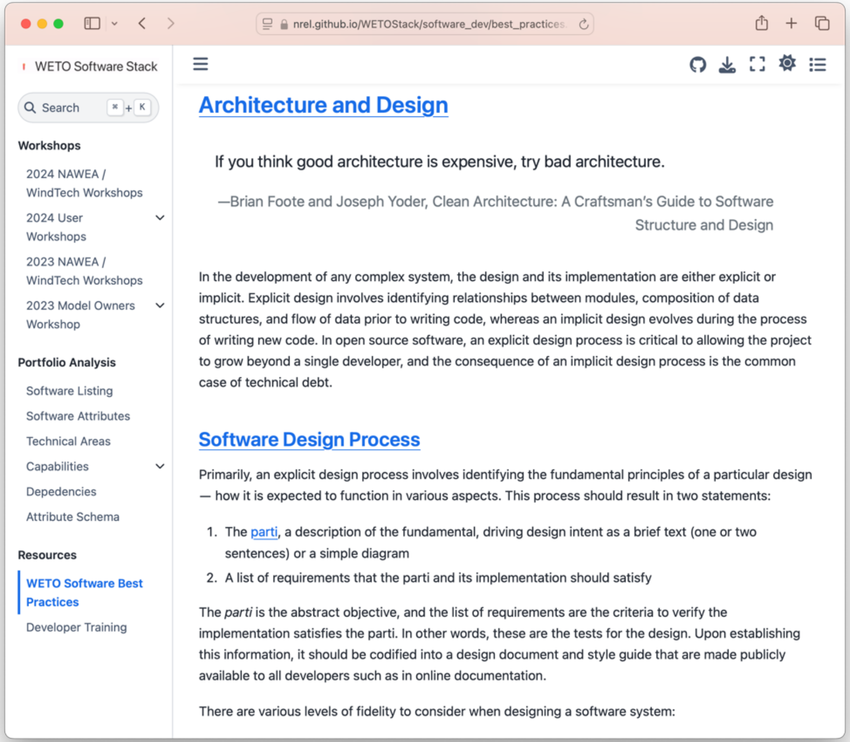
nrel.github.io/WETOStack

WETO Stack dashboard will provide the following information:

- The WETO-supported tools that enable a given task
- The state (maturity, stability) of included software
- The current and future capabilities
- Updates and community-focused materials

Current Contents:

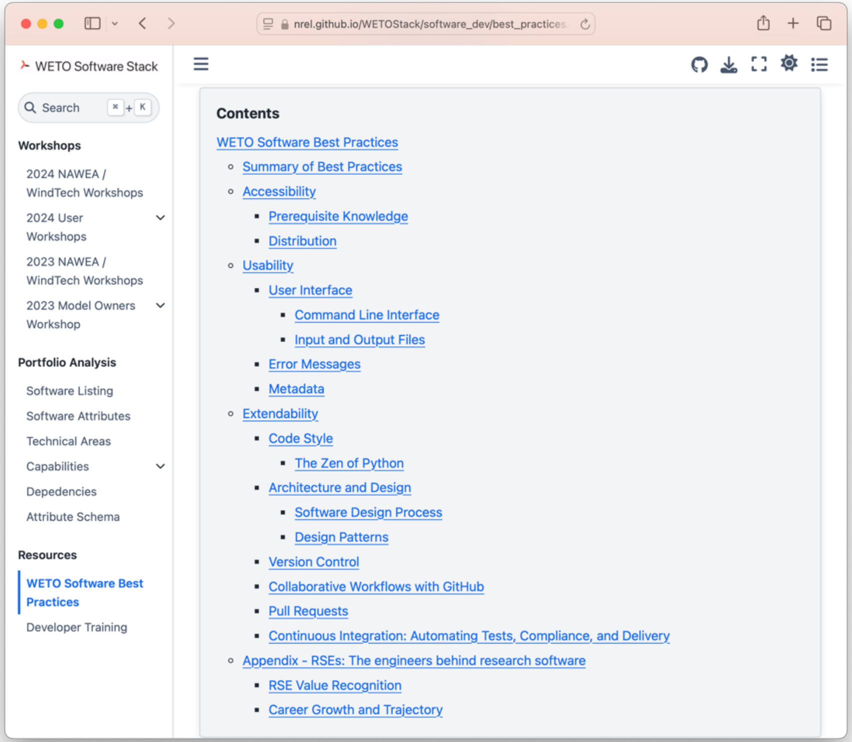
- Workshop recordings and reports
- Software Listing: active, inactive, and “other status” software
- Software Attributes: tabulated data describing each software, defined by an Attribute Schema
- Best Practices: guidance for creating software within the context of WETO and the research environment



WETO Software Best Practices

nrel.github.io/WETOStack

- **Accessibility:** How to obtain and integrate the software into your work
- **Usability:** How to get up to speed and become proficient at executing the software and understanding the results
- **Extendability:** How new features, bug fixes, and general maintenance are incorporated into the software by regular developers as well as new developers



WETO Software Stack

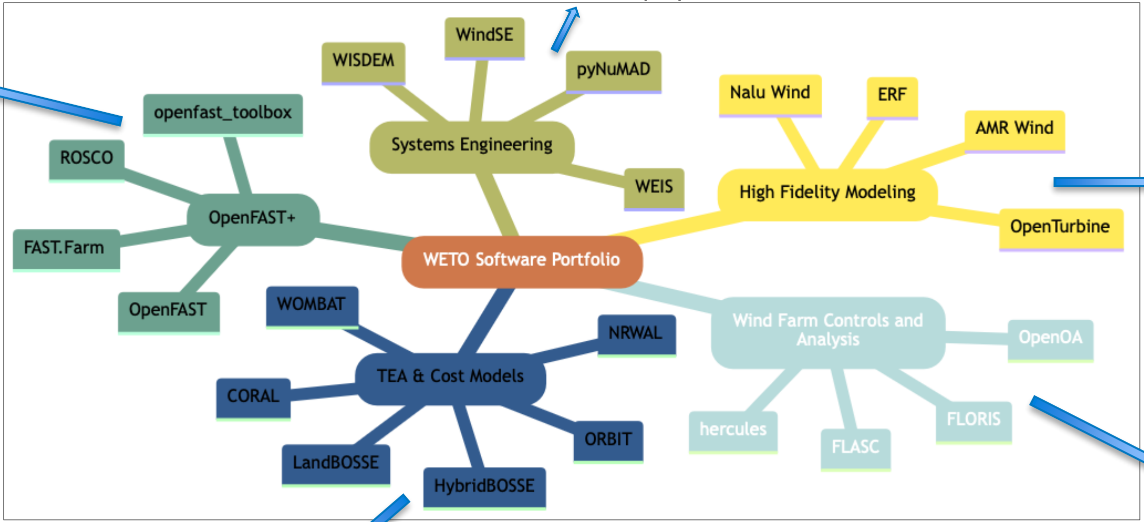
Overview

WETO Software Stack

Grouped by how its used

System-level optimizations with multi-physics and multi-fidelities

Wind turbine and wind farm mid-fidelity physics



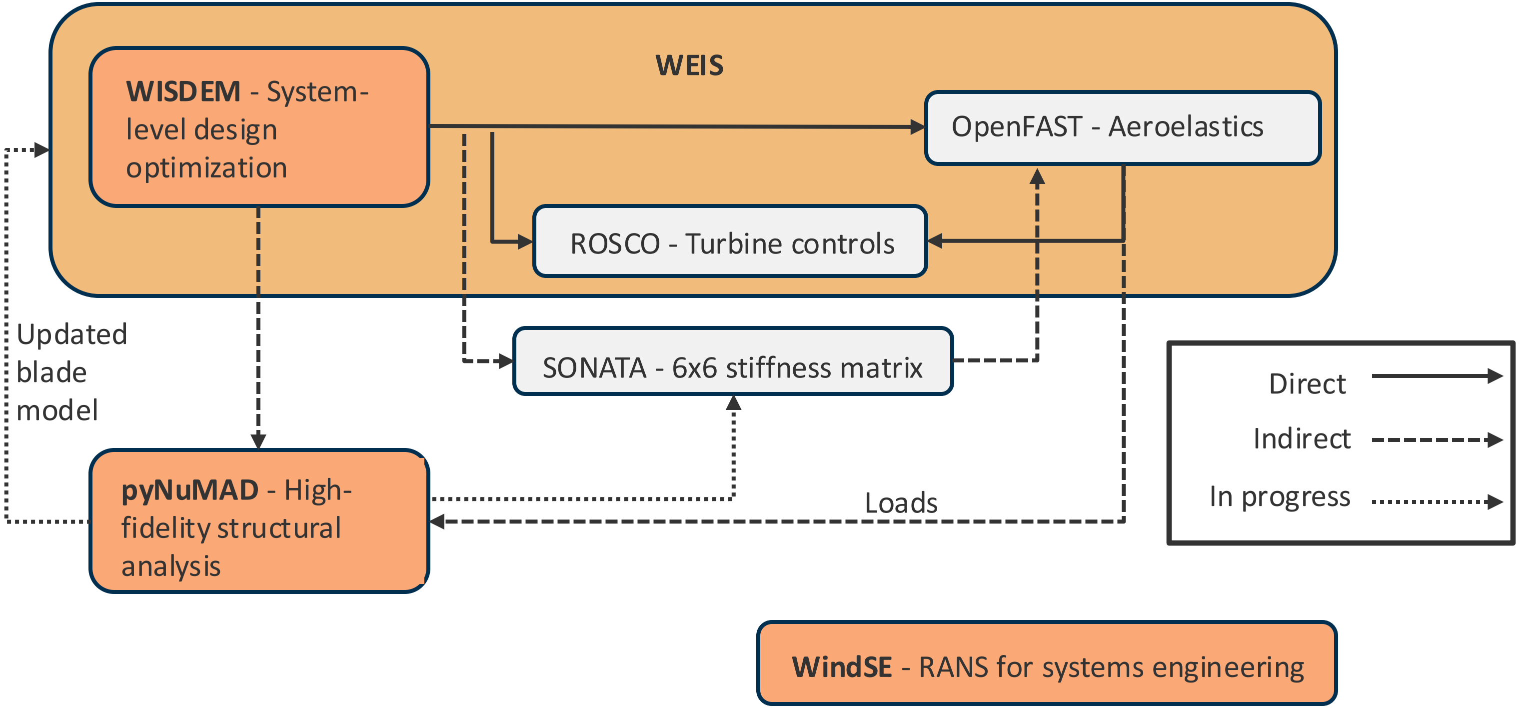
Detailed aerodynamics and structural dynamics

Wind farm energy production, energy loss identification, and loss reduction through farm-level controls

Wind turbine and wind plant costs for installation and operation, cost forecasting

Systems Engineering

Pietro.Bortolotti@nrel.gov
Garrett.Barter@nrel.gov



Adapted from Big Adaptive Rotor (BAR) project

Technoeconomic Analysis / Cost Modeling

Matt.Shields@nrel.gov

NRWAL: Offshore wind
system cost and scaling model

Energy Yield

Wind farm AEP estimate
FLORIS

CapEx

Balance-of-System
LandBOSSE
HybridBOSSE
ORBIT

Shared BOS Infrastructure
CORAL

OpEx

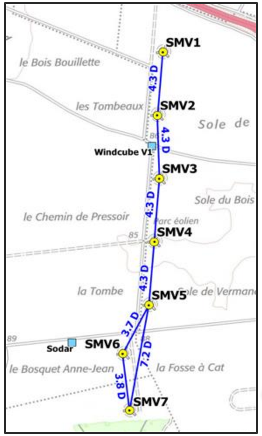
Operation & Maintenance
WOMBAT

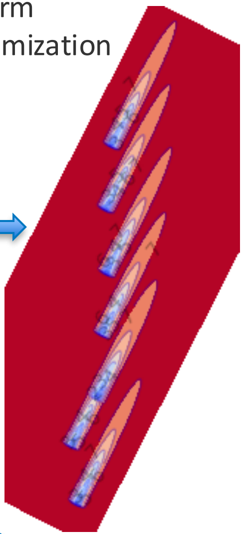
Wind Asset Value Estimate
WAVES

Wind Farm Controls and Analysis

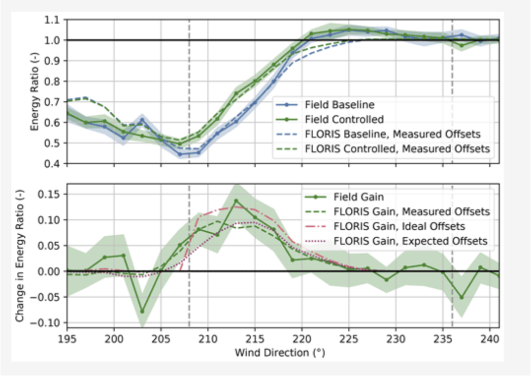
Paul.Fleming@nrel.gov

FLORIS: Steady-state modeling, farm controls optimization

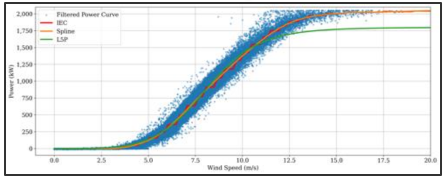




FLASC: Validate FLORIS model with SCADA, compare control methods



Hercules: Realtime high-fidelity simulator for hybrid power plants with a specific focus on wind farm controls.

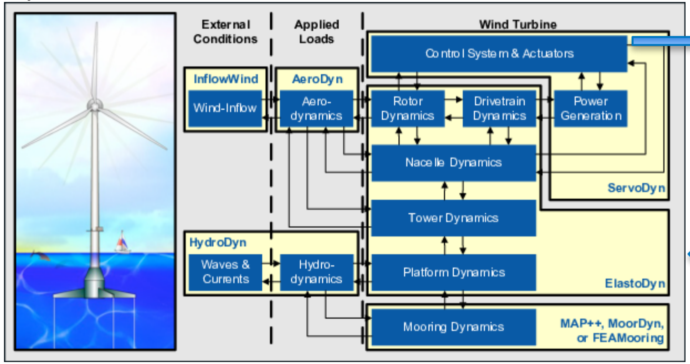


OpenOA: Characterize plant performance and quantify sources of operational loss

OpenFAST+

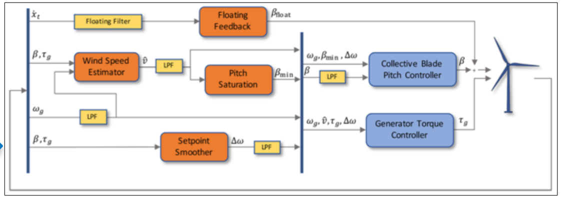
Jason.Jonkman@nrel.gov

OpenFAST



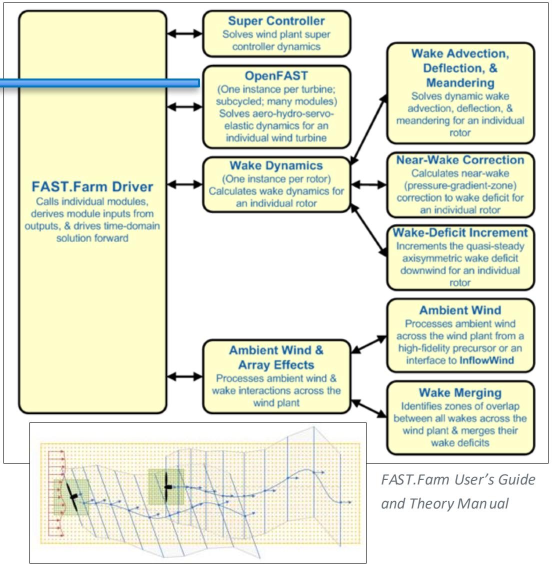
OpenFAST v3.5.3 documentation

ROSCO



N. J. Abbas et al.: A reference controller for wind turbines

FAST.Farm



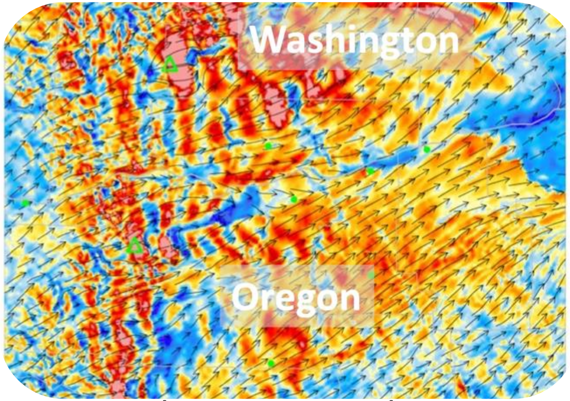
FAST.Farm User's Guide and Theory Manual

openfast_toolbox

High Fidelity Models

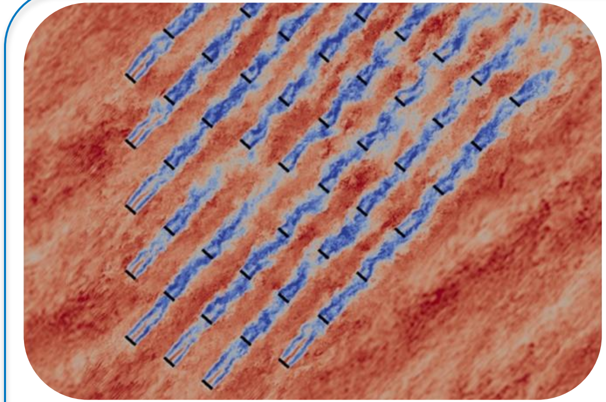
Michael.A.Sprague@nrel.gov

ExaWind



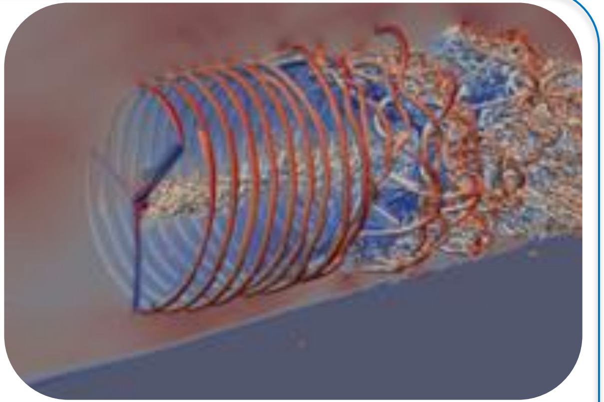
Mesoscale to microscale: ERF

- Regional scale weather down to actuator line and actuator disc models
- Scales <1 km to 1000 km
- WRF numerics & models, built on AMReX
- GPU compatible
- Compressible, incompressible, anelastic



Microscale: AMR-Wind

- Atmospheric boundary layer
- Scales less than 10 km
- Large Eddy Simulation built on AMReX
- GPU compatible
- Structured grid with refinement zones
- Incompressible
- Actuator line and disk turbine models
- Background solver for Nalu-Wind
- Couples with OpenFAST for FSI



Turbine scale: Nalu-Wind

- Turbine, rotor, tower, nacelle
- Scales less than 1 km
- Hybrid RANS-LES
- GPU compatible
- Unstructured grid, geometry resolving
- Incompressible
- Couples with OpenFAST for FSI

ExaWind

Mike Sprague

What is ExaWind?

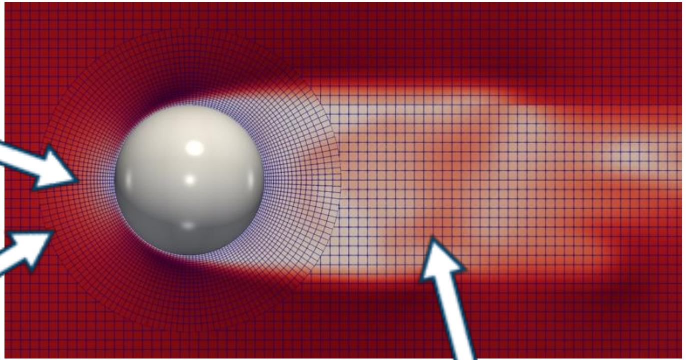
- ExaWind is an open-source suite of codes designed for high-fidelity and mid-fidelity modeling and simulation of wind turbines and wind farms
    - https://github.com/Exawind/
    - Nalu-Wind, AMR-Wind, OpenFAST/OpenTurbine
- Provide predictive tool for understanding wind energy and to validate and improve engineering models
- Designed to be “performance portable” in that it can run well on modern CPU- and GPU-based supercomputers
- Key capabilities are in place for land-based wind
    - Atmospheric turbulent flow
    - Hybrid-RANS/LES turbulence modeling
    - Fluid-structure interaction
    - Large-deformation nonlinear structural dynamics

ExaWind hybrid RANS/LES simulation of 16 NREL 5-MW turbines on the Frontier supercomputer; credit Brunhart-Lupo and Cheung

Hybrid-solver for geometry-resolved simulations

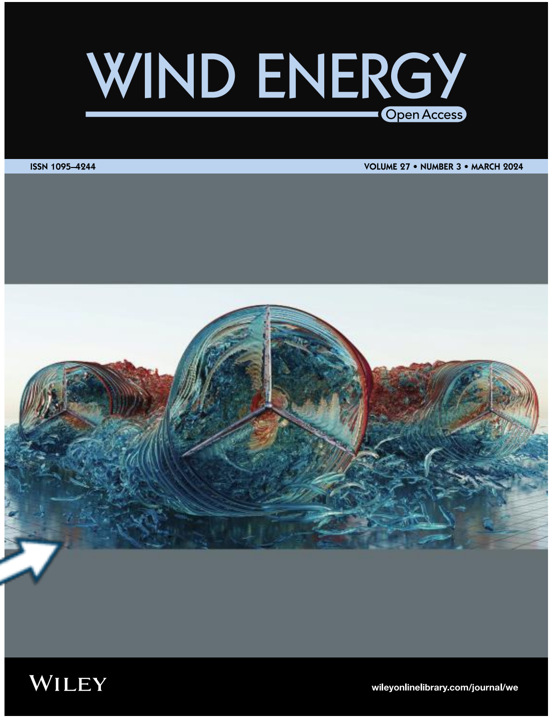
ExaWind (Nalu-Wind/AMR-Wind) flow over a sphere

Geometry-resolving unstructured-mesh Nalu-Wind model



TIOGA overset-mesh coupling

Structured-mesh AMR-Wind model



Journal of Computational Physics 428 (2021) 109987

Contents lists available at ScienceDirect

Journal of Computational Physics

www.elsevier.com/locate/jcp

Overset meshes for incompressible flows: On preserving accuracy of underlying discretizations

Ashesh Sharma [a,*], Shreyas Ananthan [a], Jayanarayanan Sitaraman [b], Stephen Thomas [a], Michael A. Sprague [a]

Received: 27 March 2023 | Revised: 27 July 2023 | Accepted: 24 November 2023

DOI: 10.1002/we.2886

RESEARCH ARTICLE

ExaWind: Open-source CFD for hybrid-RANS/LES geometry-resolved wind turbine simulations in atmospheric flows

Ashesh Sharma [1] | Michael J. Brazell [1] | Ganesh Vijayakumar [1] | Shreyas Ananthan [2] | Lawrence Cheung [3] | Nathaniel deVelder [3] | Marc T. Henry de Frahan [1] | Neil Matula [3] | Paul Mullowney [4] | Jon Rood [1] | Philip Sakievich [3] | Ann Almgren [5] | Paul S. Crozier [3] | Michael Sprague [1]

AMR-Wind for actuator-line/disc simulations

AMR-Wind methods paper submitted to Wind Energy

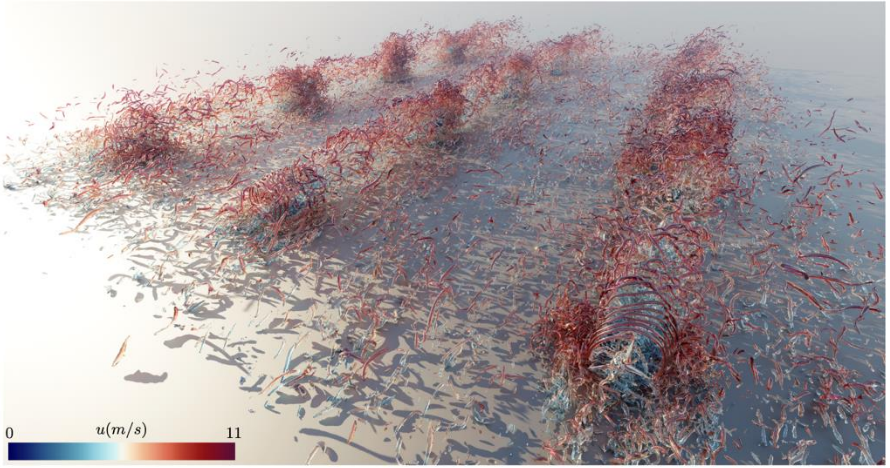
AMR-Wind: A performance-portable, high-fidelity flow solver for wind farm simulations

Michael B. Kuhn[1], Marc T. Henry de Frahan[1], Prakash Mohan[1], Georgios Deskos[1], Matthew Churchfield[1], Lawrence Cheung[3], Ashesh Sharma[1], Ann Almgren[2], Shreyas Ananthan[1], Michael Brazell[1], Luis A. Martinez-Tossas[1], Regis Thedin[1], Jon Rood[1], Philip Sakievich[4], Ganesh Vijayakumar[1], Weiqun Zhang[2], and Michael A. Sprague[1]

[1]National Renewable Energy Laboratory, Golden, CO, 80401, USA
[2]Lawrence Berkeley National Laboratory, Berkeley, CA, USA
[3]Sandia National Laboratories, Livermore, CA, 94550, USA
[4]Sandia National Laboratories, Albuquerque, NM, 87123, USA

Correspondence: Michael B. Kuhn (michael.kuhn@nrel.gov)

Abstract. We present AMR-Wind, a verified and validated multifidelity computational-fluid-dynamics code for wind farm flows. AMR-Wind is a block-structured, adaptive-mesh, incompressible-flow solver that enables predictive simulations of the atmospheric boundary layer and wind plants. It is a highly scalable code designed for parallel high-performance computing with a specific focus on performance portability for current and future computing architectures, including graphical processing



Instantaneous visualization of the flow field in the 12 turbine wind farm case at t = 1600 s. Isosurface of q-criterion at 0.019, colored by x-velocity. Image credit: Nicholas Brunhart-Lupo

- Detailed explanation of theory, implementation, and available features
- Thorough verification and validation
- Wind farm demonstration simulation with twelve IEA 15-MW turbines using actuator line model (see image)

ExaWind funding & development team (*past and present)

U.S. Department of Energy (DOE) Wind Energy Technologies Office (2016 – present)

ATMOSPHERE TO ELECTRONS
U.S. DEPARTMENT OF ENERGY

DOE Exascale Computing Project (2016-2024)

ECP
EXASCALE COMPUTING PROJECT

DOE Office of Science Floating Offshore Wind Energy Earthshot Research Center (2023 – present)

FLOWMAS

NREL Transforming ENERGY 45th anniversary

M. Sprague, PI
S. Ananthan*
R. Binyahib*
M. Brazell*
M. Churchfield*
G. Deskos
K. Gruchalla*
S. Hammond*
M. Henry de Frahan
M. Kuhn
M. Lawson*
T. Martinez*
P. Mullowney (AMD)
J. Rood
A. Sharma*
K. Swirydowicz*
S. Thomas*
G. Vijayakumar
S. Yellapantula*

Sandia National Laboratories

P. Crozier, co-PI
L. Berger-Vergiat
M. Barone*
M. Blaylock*
L. Cheung
N. Develder
S. Domino*
D. Glaze*
A. Hsieh*
J. Hu*
R. Knaus
D.H. Lee*
N. Matula
T. Okusanya*
J. Overfelt*
S. Rajamanickam*
P. Sakievich*
T. Smith
J. Vo*
A. Williams*
D. Womble*
I. Yamazaki*

OAK RIDGE National Laboratory

J. Turner*
A. Prokopenko*
S. Slattery
R. Wilson*

BERKELEY LAB
Bringing Science Solutions to the World

A. Almgren
W. Zhang
Aaron Lattanzi

TEXAS
The University of Texas at Austin

R. Moser*
J. Melvin*

UNIVERSITY OF WYOMING

D. Mavriplis
A. Kirby

Parallel Geometric Alg., Inc.

J. Sitaraman*

ExaWind status & outreach

- Capabilities established for high- and mid-fidelity simulations of land-based wind turbines and farms
- Active development:
  - Performance and robustness improvements
  - Coupling to OpenTurbine
  - Two-phase flow for floating offshore wind; validation in 2025
- Creating benchmark repository and webpage
  - https://github.com/Exawind/exawind-benchmarks
  - https://exawind.github.io/exawind-benchmarks/
  - Will contain hierarchy of cases relevant to wind energy
  - ExaWind input files, meshes, outputs, performance data
  - We welcome results from other codes
- AMR-Wind user workshops at NAWEA 2022, 2023, 2024

ExaWind:OpenTurbine

Mike Sprague

OpenTurbine Overview

A flexible multi-body dynamics code that provides a wind turbine structural model for CFD codes -> specifically targeting ExaWind.

Current Status

- In active development
- Key library development is nearing completion
- Proof-of-concept studies have been completed

Key software and algorithm design choices

- C++ with Kokkos for GPU-based computing
- Index-3 differential-algebraic-equation (DAE-3) formulation
- Second-order Lie-group generalized-alpha time integrator
- Rigid body, geometrically exact beam, and constraint member types
- High-order beam finite elements

Dev Team

Derek Slaughter
Faisal Bhuiyan
David Dement
Paul Crozier
Mike Sprague

Funding

DOE EERE WETO
DOE Office of Science FLOWMAS Energy Earthshot Research Center

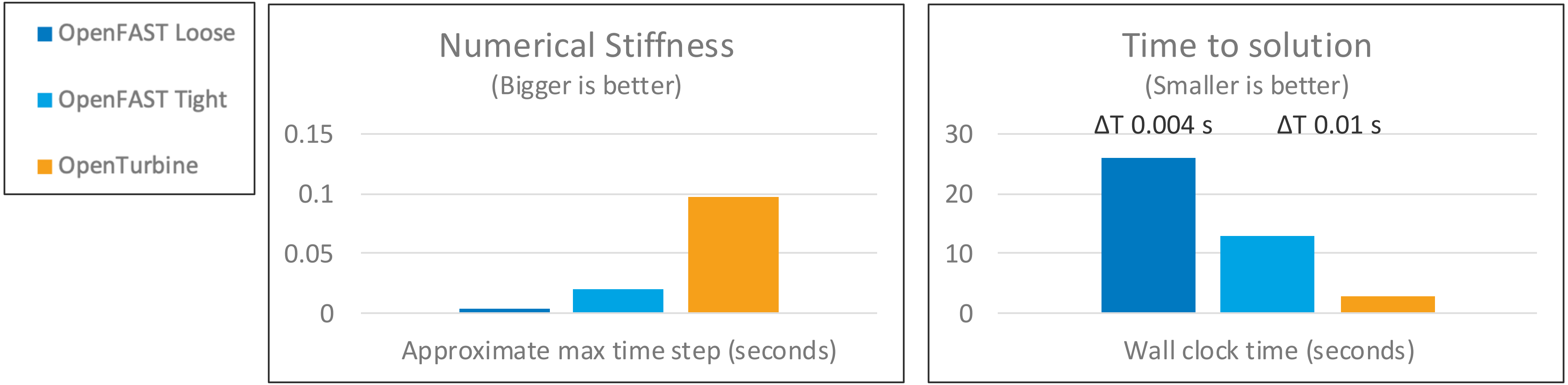
https://github.com/Exawind/openturbine

OpenTurbine Proof of Concept 1

Objective:

- Examine stability (maximum stable time step) and computational speed
- Compare OpenTurbine and OpenFAST standard (loose coupling) and new tight coupling

- Turbine: IEA 15-MW
- Rotational speed: Fixed
- Loading: Gravity (no fluid)
- Models: OpenTurbine standalone



Stable with significantly larger time steps

5x faster than OpenFAST tight coupling; supports GPUs

OpenTurbine Proof of Concept 2

Objective:

- Examine stability (maximum stable time step) and computational speed
- Compare OpenTurbine and OpenFAST standard (loose coupling) and new tight coupling

- Turbine: IEA 15-MW
- Load: wind gusts from 10.59 m/s to 20 m/s over 5 to 15 sec with pitch control
- Models: OpenTurbine coupled to AeroDyn and the ROSCO controller

Model	GitHub ID	Time Step	Simulation Wall-Clock Time	Wall-clock time per simulated time (smaller is better)
OpenTurbine	22157a20ff	0.01 s	35 sec	1.2
OpenFAST (tightly coupled)	3f0e899d1b	0.01 s	194 sec	6.5
OpenFAST 3.5.3 (loosely coupled)	3.5.3	10^{-6} - 0.01 s	N/A (unstable)	N/A (unstable)

OpenTurbine is about five times faster that the tightly coupled OpenFAST; unable to produce stable solution with standard, loosely coupled OpenFAST

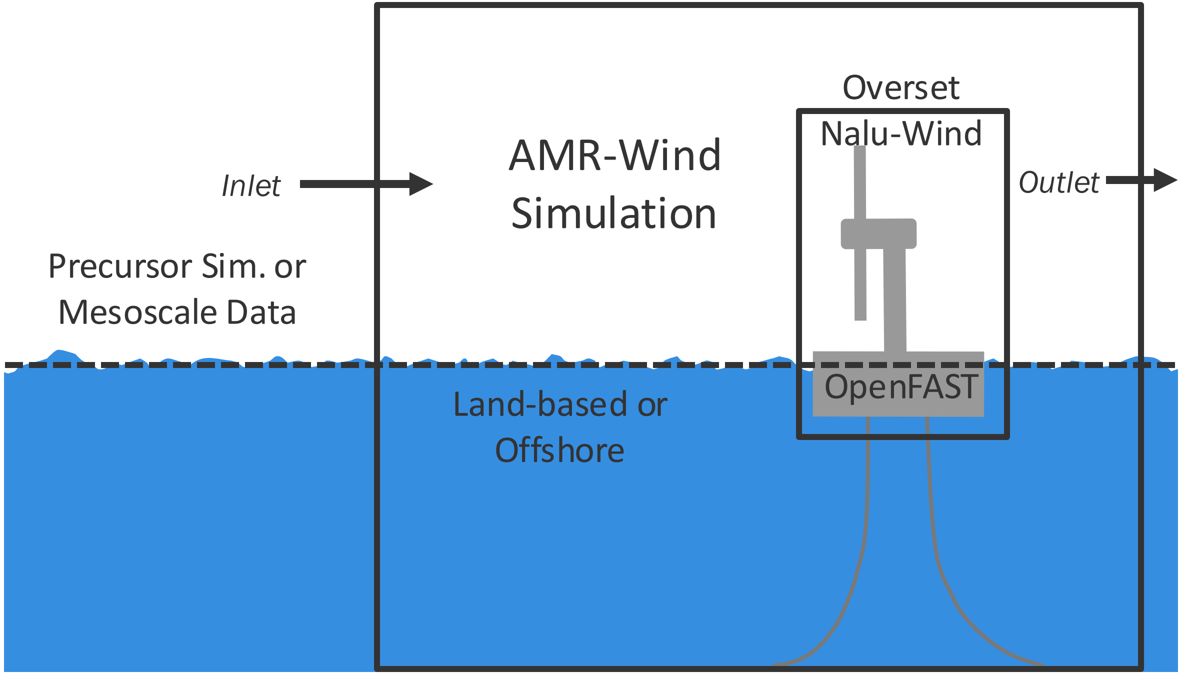
OpenTurbine Next Steps

- Generalizing implementation for horizontal-axis wind turbines and floating rigid bodies
- Creating interface for input through WindIO turbine definition schema
- Designing and implementing application programming interface for fluid-structure-interaction coupling with ExaWind
  - Nalu-Wind for geometric-resolved simulations
  - AMR-Wind for actuator-line simulations
- Goal to release production version late summer 2025

ExaWind:AMR-Wind

Michael Kuhn

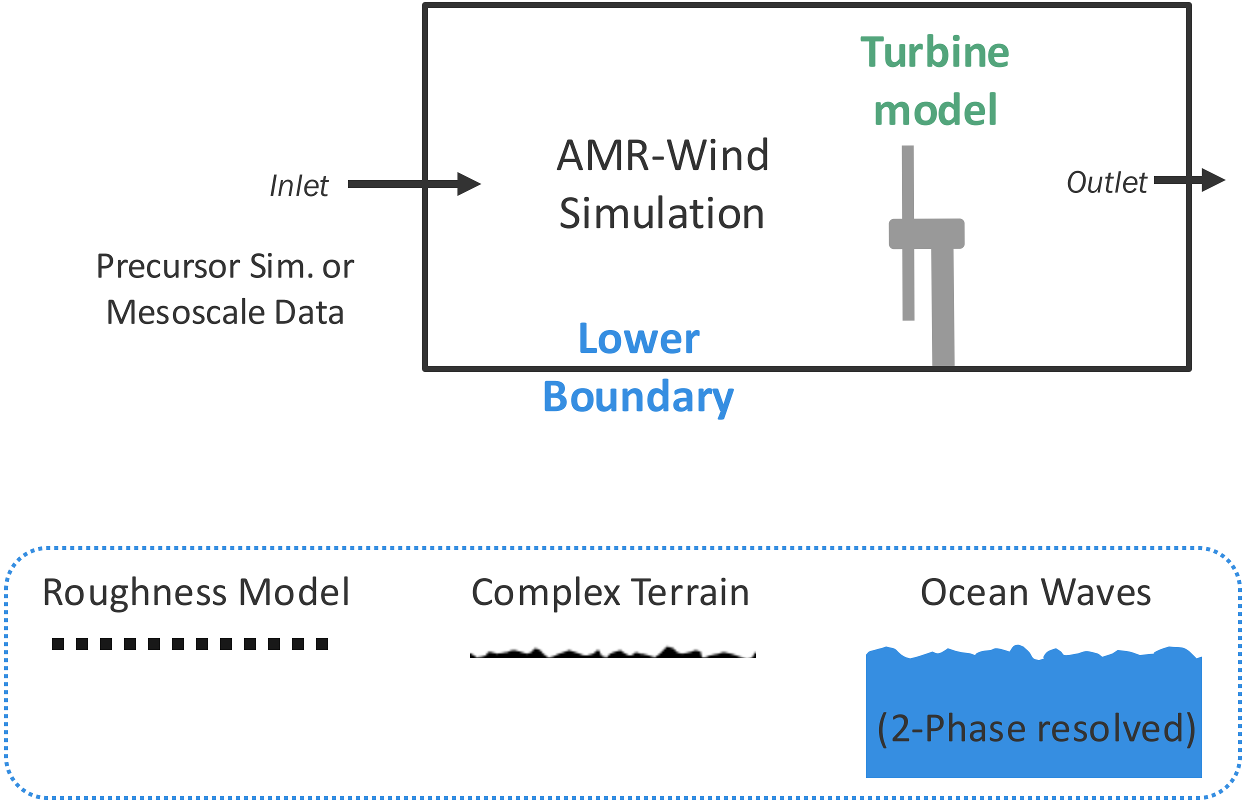
Exawind: AMR-Wind

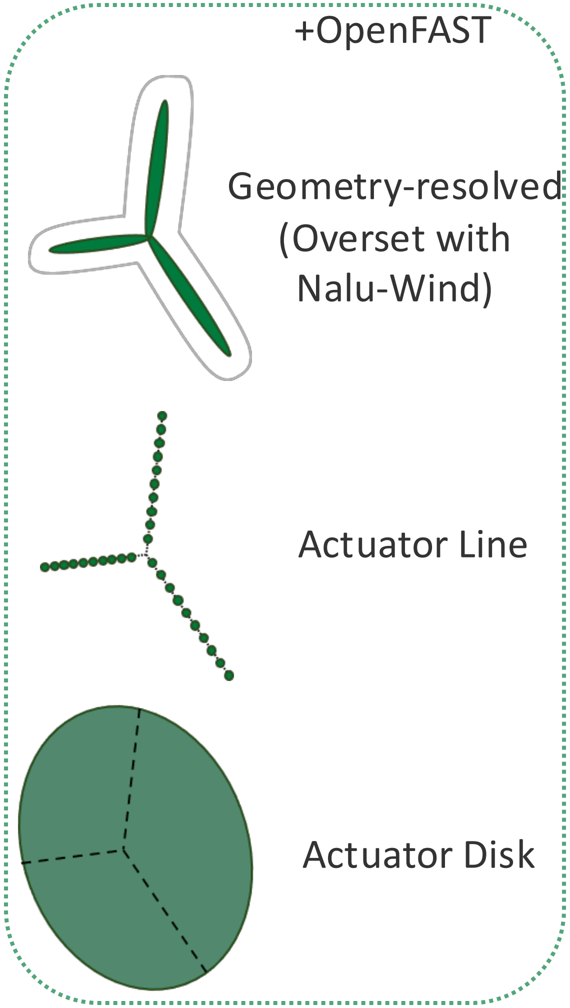


<u>Far-field flow solver for turbine simulations:</u>

- Atmospheric Boundary Layer (ABL)
    - Precursor
    - Mesoscale Solver
- Turbulence modeling for LES
- Terrain boundary modeling
- Mesh refinement around turbine

Exawind: AMR-Wind





AMR Wind Background

Incompressible, constant density, Boussinesq buoyancy

(Also available: single-phase anelastic; two-phase incompressible)

Turbulence Modeling – Bulk (LES):

- Constant Smagorinsky
- One-equation k_{SGS} (Moeng 1984)
- Anisotropic minimum dissipation (AMD)

$$\frac{\partial k}{\partial t} + \frac{\partial k u_j}{\partial x_j} = \frac{\partial}{\partial x_j}\left(2\nu_t \frac{\partial k}{\partial x_j}\right) - \tau_{ij} S_{ij} + \frac{g}{\theta_\circ}\tau_{\theta 3} - C_\epsilon \frac{k^{3/2}}{l}$$

Turbulence Modeling – Boundary:

- Moeng (1984) + Monin-Obukhov similarity theory (MOST)
    - Applies to flat boundary or complex terrain

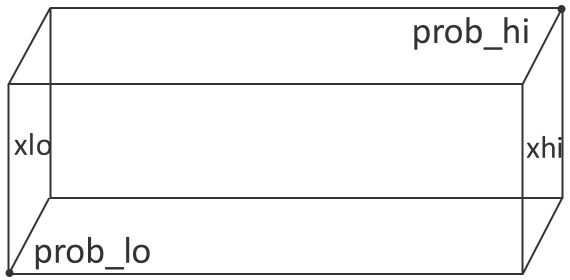
$$\tau_{i3} = \frac{\overline{u_i}(z_b) s + \bar{s}\left(u_i(z_b) - \overline{u_i}(z_b)\right)}{\bar{s}^2} u_\tau^2$$

RANS modeling also available for bulk (k-ω SST and others)

AMR Wind Background

Structured mesh (Cartesian); patch-based refinement

- Cells are tagged, finer patches are introduced
- Aspect ratio of cells is constant



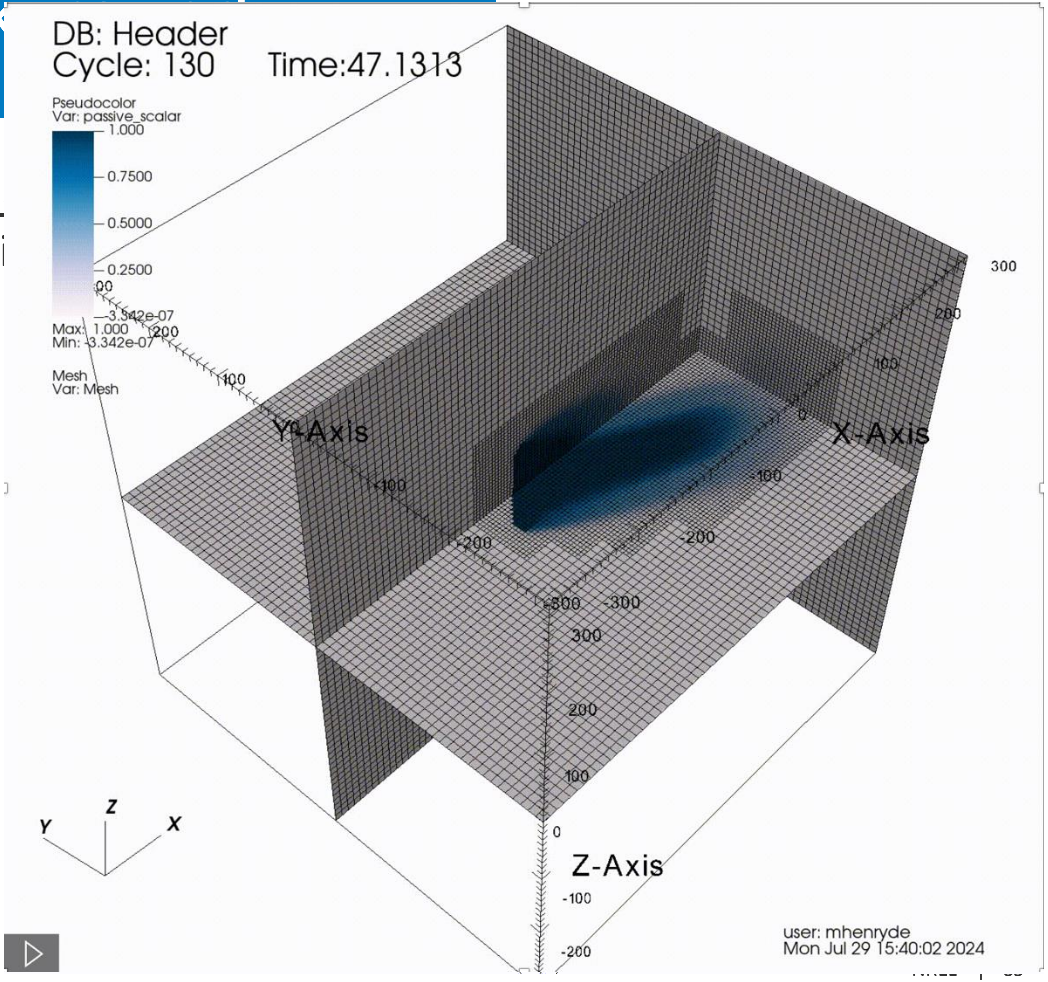
AMR Wind Back

Structured mesh (Cartesian); patch-b

- Cells are tagged, finer patches are i
- Aspect ratio of cells is constant

Mesh refinement – Adaptive:

- Tagging based on flow variables



AMR Wind Background

Structured mesh (Cartesian); patch-based refinement

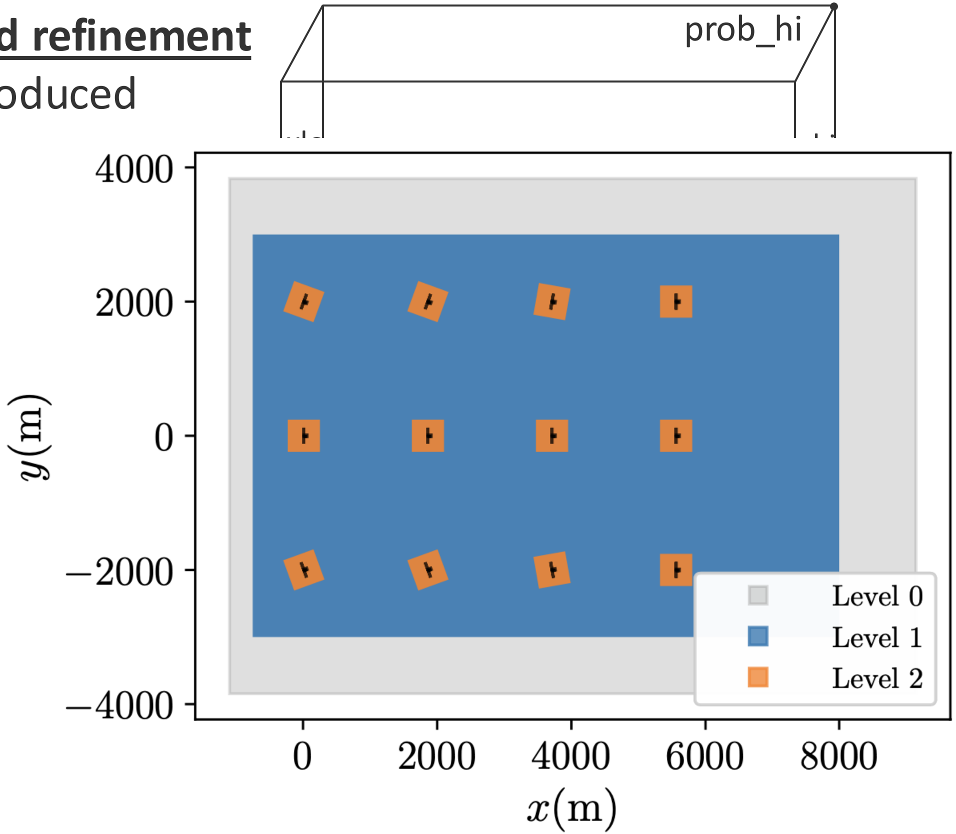
- Cells are tagged, finer patches are introduced
- Aspect ratio of cells is constant

Mesh refinement – Adaptive:

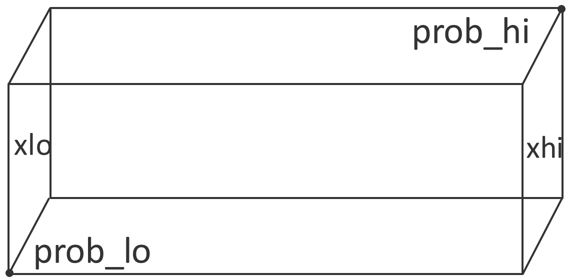
- Tagging based on flow variables

Mesh refinement – Static:

- Tagging based on geometric sections



AMR Wind Background



Structured mesh (Cartesian); patch-based refinement

- Cells are tagged, finer patches are introduced
- Aspect ratio of cells is constant

Mesh refinement – Adaptive:

- Tagging based on flow variables

Mesh refinement – Static:

- Tagging based on geometric sections of domain

AMR-Wind Frontend:

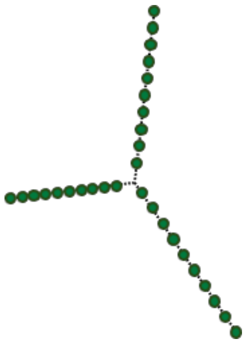
- GUI or Python interface to set up input file
- Helpful for many refinements, many turbines
- Powerful post-processing tools included

AMR Wind Background

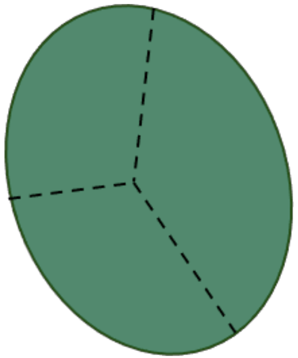
Actuator-based turbine models (non-overset)

- Actuator Line Method
  - OpenFAST-linked
  - Required mesh resolution depends on epsilon scale
  - Required time step size depends on rotor speed
  - Gold standard for turbines without resolving geometry
  - Non-uniform point distribution can save comp. cost
  - Filtered-lifting line correction can improve accuracy

- Actuator Disk Method
  - OpenFAST-linked or standalone
  - Less stringent mesh and time step requirements
  - Larger variety of modeling options



Actuator Line



Actuator Disk

AMR Wind Usage

To do a turbine study with AMR-Wind, you need:

- **Target flow conditions:**
    - Constant flow profile
    - Time-varying inflow - Precursor
        - Standalone: target velocity, thermal stratification, forcing terms
        - Linking to data from mesoscale solver (WRF, ERF*)
- **Turbine model:**
    - Standalone (uniform C_t or Joukowsky disk): parameters and turbine geometry
    - OpenFAST-linked: CFD-facing parameters and OpenFAST turbine input files
- **Solver settings pertinent to the problem type** (desired models and outputs)
- **Sufficient computational resources**

AMR Wind Usage

When doing a turbine study with AMR-Wind, you get:

- **Flow characterization:**
    - ABL statistics
    - Flow quantities at sampling locations: planes, lines, spinner lidar, etc.
    - Flow data for 3D visualization software
- **Turbine performance:**
    - Actuator-based quantities, such as velocities and forces
    - OpenFAST turbine outputs

AMR Wind Workflow

Online Documentation:

⊟ **Walkthrough**

- Compiling using Spack, with Exawind-Manager
- Precursor (ABL) walkthrough
- Turbine simulation walkthrough
- LES with Terrain
- Actuator Line Model Calibration

User Manual

Theory Manual

Developer Documentation

Walkthrough

This section demonstrates a typical AMR-Wind workflow, walking through the steps required to simulate wind turbines in a turbulent atmospheric boundary flow. The compilation instructions outline how to use Exawind-Manager, a Spack-based package management tool customized for the ExaWind software suite, of which AMR-Wind is an integral part. The turbulent flow conditions are established through precursor simulations, and then turbines are placed in the flow.

Note

This tutorial is intended to provide an example of how AMR-Wind is often used, but there are many variations and alternative workflows that AMR-Wind provides. Please consult the User Manual, especially the capabilities list and input file reference, for additional details on other AMR-Wind features and options.

AMR Wind Workflow

⊟ **Walkthrough**

- Compiling using Spack, with Exawind-Manager
- Precursor (ABL) walkthrough
- Turbine simulation walkthrough

<u>Compiling:</u>

1. Clone exawind-manager and activate
2. Create environment choosing amr-wind version, desired options, and compiler
3. Install and load

1.

```
$ git clone --recursive https://github.com/Exawind/exawind-manager.git
$ export EXAWIND_MANAGER=~/exawind-manager/
$ source $EXAWIND_MANAGER/start.sh
$ spack-start
```

2.

```
$ mkdir env_walkthrough
$ quick-create-dev -d env_walkthrough/ -s amr-wind@main+hypre+netcdf
```

3.

```
$ spack install
$ spack load amr-wind
```

AMR Wind Workflow

⊟ **Walkthrough**

- Compiling using Spack, with Exawind-Manager
- Precursor (ABL) walkthrough
- Turbine simulation walkthrough

Running precursor simulation:

- Design atmospheric flow of interest
  - Target velocity at target height
  - Thermal profile
  - Doubly periodic

```
incflo.velocity                          = 10.0 0.0 0.0
ABLForcing.abl_forcing_height            = 86.5
ABL.temperature_heights                  = 0.0 600.0 700.0 1700.0
ABL.temperature_values                   = 290.0 290.0 298.0 301.0
ABL.surface_temp_flux                    = 0.05
geometry.is_periodic                     = 1   1   0
```

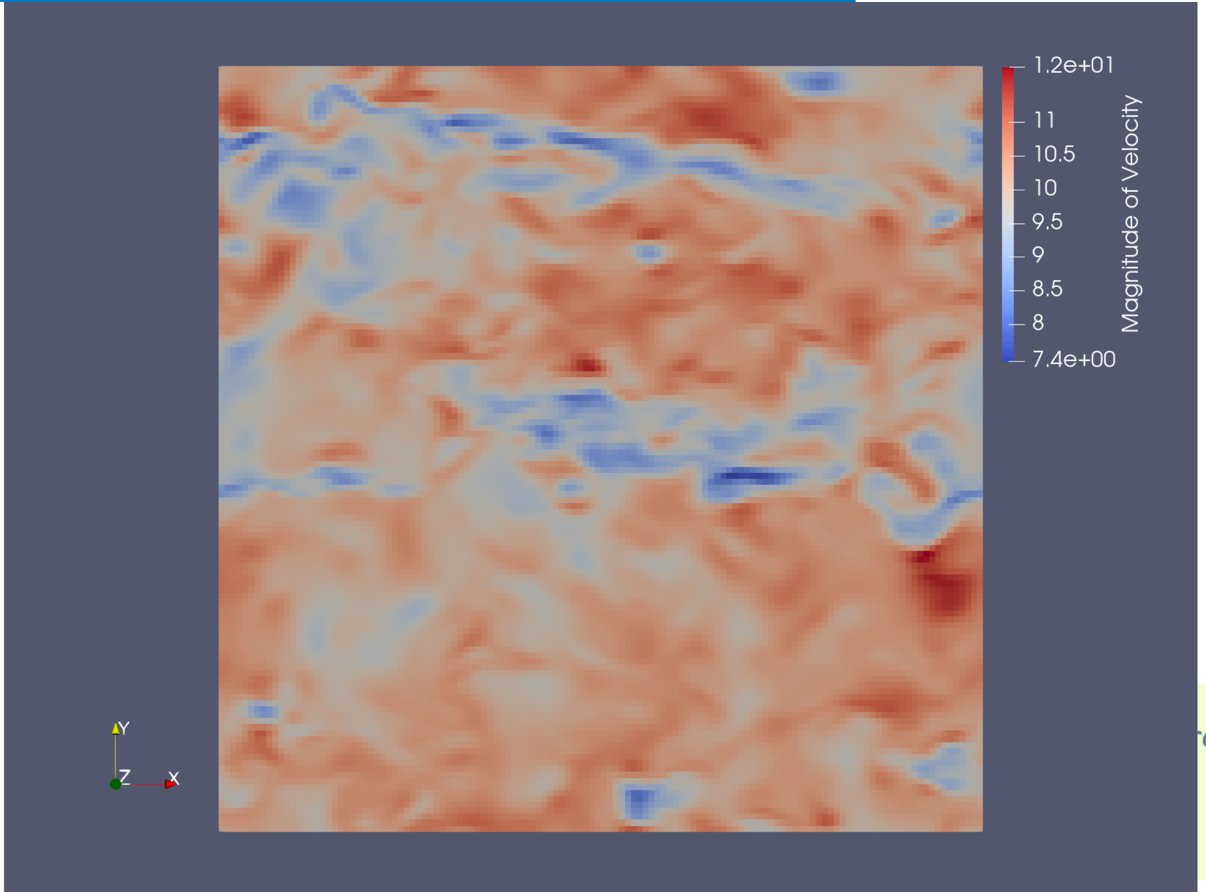
- Save boundary planes when sufficiently developed

```
ABL.bndry_file                           = bndry_file.native
ABL.bndry_io_mode                        = 0           # 0 = write, 1 = read
ABL.bndry_planes                         = xlo
ABL.bndry_output_start_time              = 7200.0
ABL.bndry_var_names                      = velocity temperature tke
```

AMR Wind Workflow

Walkthrough

- Compiling using Spack, with Exawind-Manager
- Precursor (ABL) walkthrough
- Turbine simulation walkthrough



ead

AMR Wind Workflow

Walkthrough

- Compiling using Spack, with Exawind-Manager
- Precursor (ABL) walkthrough
- Turbine simulation walkthrough

Running turbine simulation:

- Place turbines in domain

```
Actuator.T0.type                        = TurbineFastDisk
Actuator.T0.openfast_input_file         = T0_OpenFAST/NREL-2p8-127.fst
Actuator.T0.base_position               = 640.0 1280.0 0.0
Actuator.T0.rotor_diameter              = 126.9
```

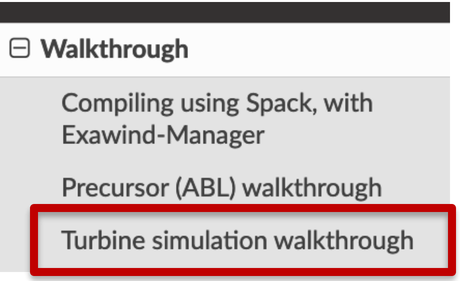
- Accompany turbines with mesh refinement

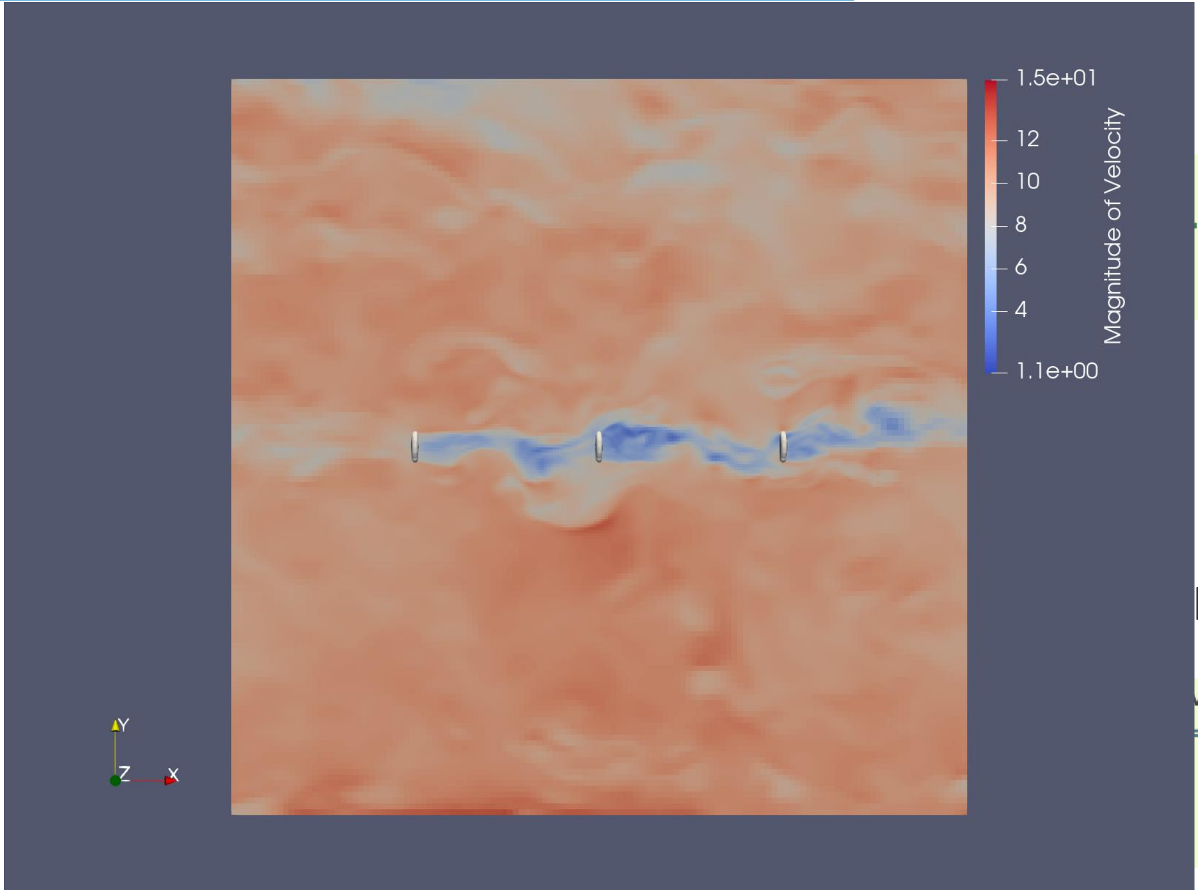
```
# 1st refinement level
tagging.T0_level_0_zone.type                  = GeometryRefinement
tagging.T0_level_0_zone.shapes                = T0_level_0_zone
tagging.T0_level_0_zone.level                 = 0
tagging.T0_level_0_zone.T0_level_0_zone.type = box
```

- Begin from saved precursor checkpoint and use inflow plane data

```
io.restart_file                              = ../spinup/chk14400
ABL.bndry_file                               = ../precursor/bndry_file.native
ABL.bndry_io_mode                            = 1            # 0 = write, 1 = read
ABL.bndry_planes                             = xlo
ABL.bndry_output_start_time                  = 7200.0
ABL.bndry_var_names                          = velocity temperature tke
```

AMR Wind Workflow





.fst

lane data

ve
= read

AMR Wind Roadmap

Current capabilities (https://exawind.github.io/amr-wind/user/features.html) :

In active development:

- Direct coupling with ERF mesoscale simulations
- Dynamic wave boundary model
- Overset coupling with Nalu-Wind for two-phase simulations of floating platforms

Planned development:

- Near-interface turbulence modeling for ocean waves + ABL

For bug reports and feature requests, please submit an issue on GitHub!

To get on our user mailing list, send an email to *amr-wind-maintainers@groups.nrel.gov*
GitHub is preferred for most inquiries, but this email can also be used for direct correspondence.

ExaWind Software and Performance

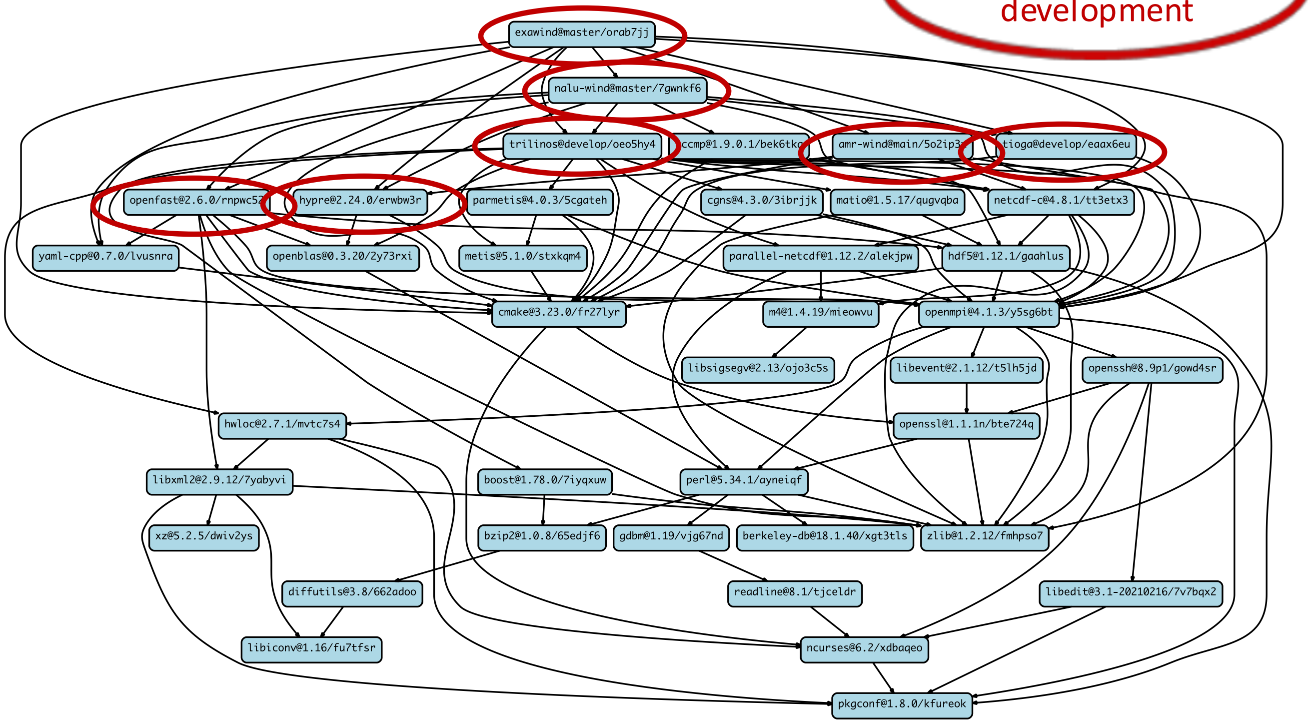
Jon Rood and Marc Henry de Frahan

The concrete ExaWind software stack

- ExaWind software stack:
  - Living on the develop branch of multiple dependencies
  - Actively supporting development of 7+ software packages in the stack (CPU+GPU)
  - Built and used ExaWind on over 25 HPC machines
  - Entire stack can take a significant amount of time to build with high risk to encounter errors
    - Build time is mostly solved now in Spack with both high and low level parallelism

ExaWind Software Stack (DAG)

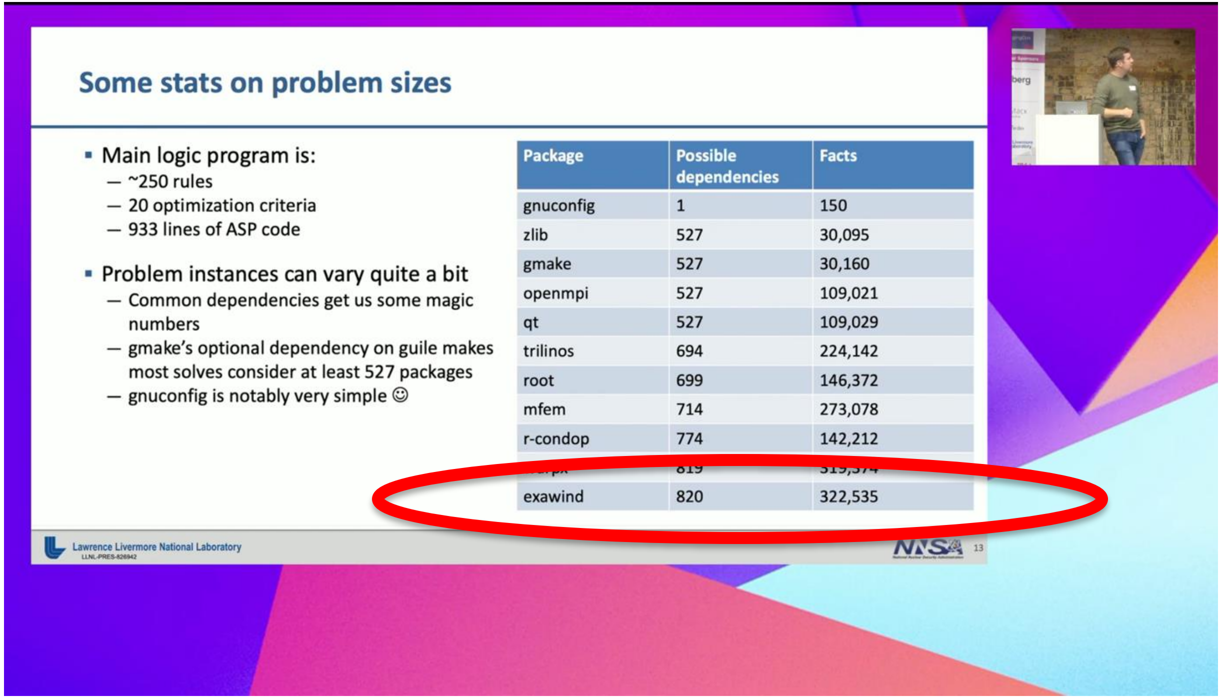
Packages under active development



ExaWind dependency complexity

- ExaWind relies on several very active third party libraries (TPLs):
  - Trilinos, Kokkos, AMReX, hypre, NetCDF, Parallel-NetCDF, HDF5, yaml-cpp, Boost, BLAS/LAPACK, etc
- Relying on several TPLs allows us to leverage work from many other people we could not do on our own
- However, there is a significant burden to managing the interplay between constantly changing libraries
- We have spent much time providing several fixes to TPLs
- Lots of custom patches perpetually in play
- Spack essential for machine portability and development tasks

Spack at PackagingCon



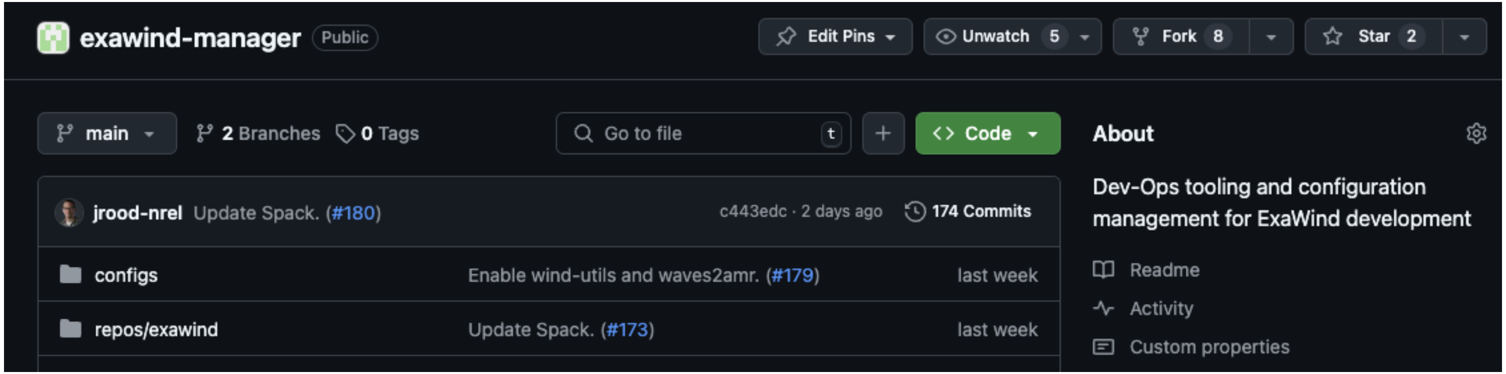
ExaWind-Manager

The software stack is complex so we have developed tooling for developers and users:

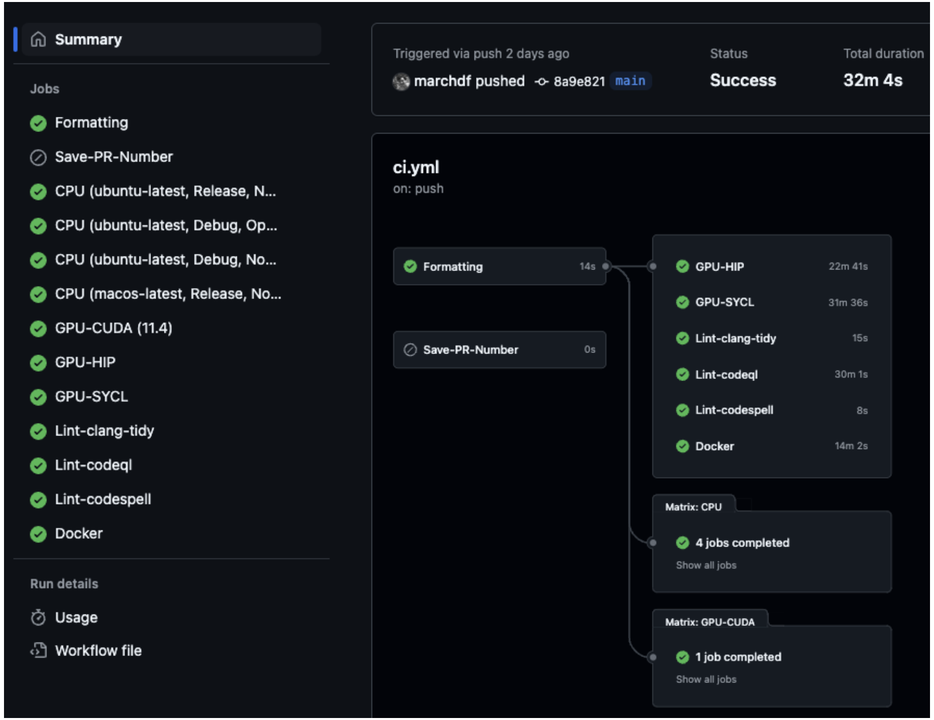
- Multiple machines, compilers, architectures
- Multiple user profiles: developers and users

Single line to build the entire stack:

```
$ git clone --recursive git@github.com:Exawind/exawind-manager.git && cd exawind-manager && \
        source shortcut.sh && nice deploy.py --ranks 24 --depfile --overwrite --name exawind
```



Continuous testing, monitoring, updating



ExaWind: Programming models for performance portability

Nalu-Wind
- Kokkos abstraction layer

TIOGA
- Currently restricted to CPUs

AMR-Wind
- AMReX abstraction layer

OpenFAST
- Legacy Fortran; restricted to CPUs

ExaWind is able to effectively utilize CPUs and NVIDIA, AMD, and Intel GPUS.
Minimizes duplicate code for multiple architectures.

Portability libraries give us multi-device capability

ExaWind Device Capability

Device	AMR-Wind	Nalu-Wind	TIOGA	OpenFAST
CPU	✅	✅	✅	✅
NVIDIA GPUs	✅	✅		
AMD GPUs	✅	✅		
Intel GPUs	✅	In progress		

ExaWind Binary Device Modes

Device	AMR-Wind	Nalu-Wind
CPU	✅	✅
GPU	✅	✅

Note: we can set all four binaries to build within a single Spack environment:

- exawind+amr_wind_gpu+nalu_wind_gpu
- exawind+amr_wind_gpu~nalu_wind_gpu
- exawind~amr_wind_gpu+nalu_wind_gpu
- exawind~amr_wind_gpu~nalu_wind_gpu

At runtime spack load exawind+amr_wind_gpu~nalu_wind_gpu

Multi-device utilization gives further flexibility

- Running one MPI rank per GPU is straightforward, but AMR-Wind and Nalu-Wind must overlap within ranks or allocate their own GPUs in separate ranks so they may run concurrently
- ExaWind turbine simulations typically involve a 40:1 ratio of gridpoints between AMR-Wind and Nalu-Wind so a natural mapping is to:
  - Run AMR-Wind with one rank per GPU
  - Run Nalu-Wind on what would be idle CPU cores (strong scaling is good on CPUs)
  - This mode is slightly less performant than Nalu-Wind on GPUs, but requires much less nodes and saturates the entire node
- Mixed-device mapping is not trivial
  - Requires advanced functionality from resource managers (explicit resource files on Summit, MPICH_RANK_REORDER_METHOD=3 on Frontier, etc)
  - Need a script to write the MPI rank mapping file for the following example list:

Example ExaWind MPI_COMM_WORLD

Exawind 0	Exawind 1	Exawind 2	Exawind 3	Exawind 4	Exawind 5
AMR-Wind 0	AMR-Wind 1	Nalu-Wind 0	Nalu-Wind 1	Nalu-Wind 3	Nalu-Wind 4

AMR-Wind and Nalu-Wind strong scaling

<u>Problem description</u>

- ABL, neutral atmospheric boundary layer problem
- 5 x 5 x 1 km^3
- 10 m grid spacing

<u>Summit</u>

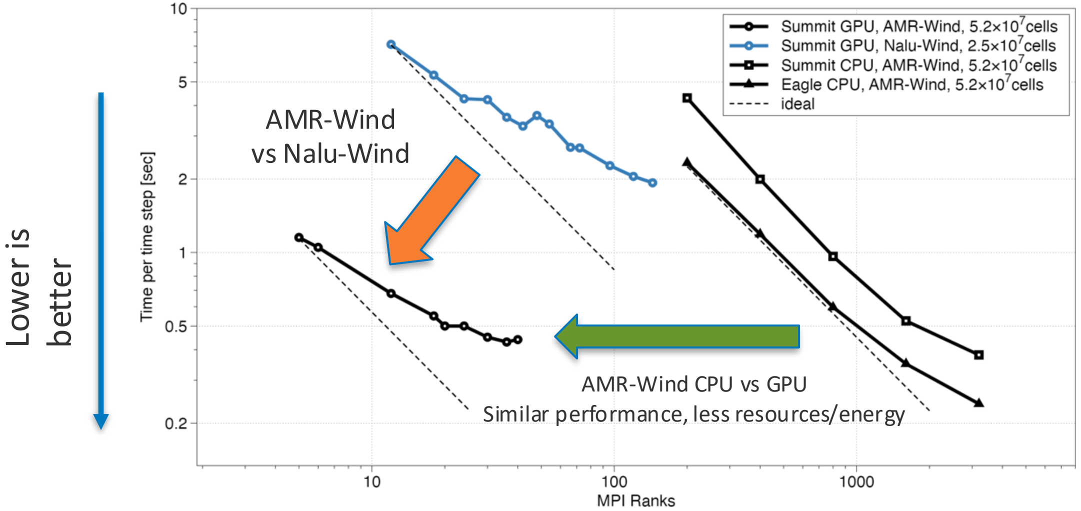
- 4608 compute nodes, each with:
  - 2x IBM POWER9 CPUs w/ 42 cores total
  - 6x Nvidia V100 GPUs

<u>Eagle</u>

- 2618 compute nodes, each with:
  - 2x x86-based CPU w/ 36 cores total

<u>Results</u>

- AMR-Wind ~10x faster than Nalu-Wind
- GPUs better but not necessarily fastest for us -- more efficient than CPUs



Code	CPU Strong Scaling Limit	GPU Strong Scaling Limit
AMR-Wind	1.5E4 gridpoints/core	2.0E6 gridpoints/GPU
Nalu-Wind	2.0E4 gridpoints/core	2.5E5 gridpoints/GPU

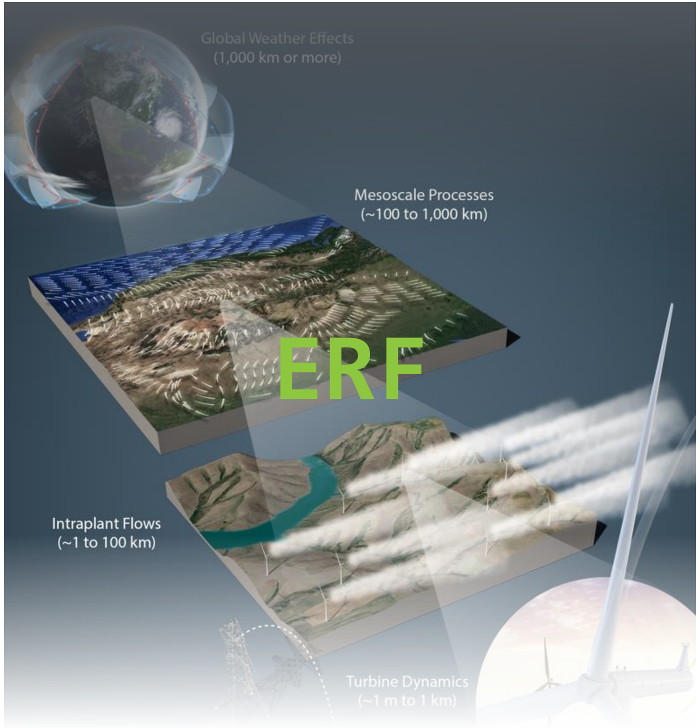
Energy Research & Forecasting

Eliot Quon

Overview of ERF

ERF provides an atmospheric modeling capability that runs on the latest high-performance computing architectures.

- Many widely used atmospheric modeling codes today (such as WRF) do not have the ability to use GPU acceleration.
- DOE is investing in supercomputers in which GPUs provide much/most of the compute power.
- ERF can run on machines from laptops to supercomputers, whether CPU-only or GPU-accelerated. GPUs from all three major vendors (NVIDIA, AMD and Intel) are supported.
- ERF is completely open source and supported by an established software framework (AMReX).



Based on illustration by Josh Bauer and Besiki Kazaishvili, NREL

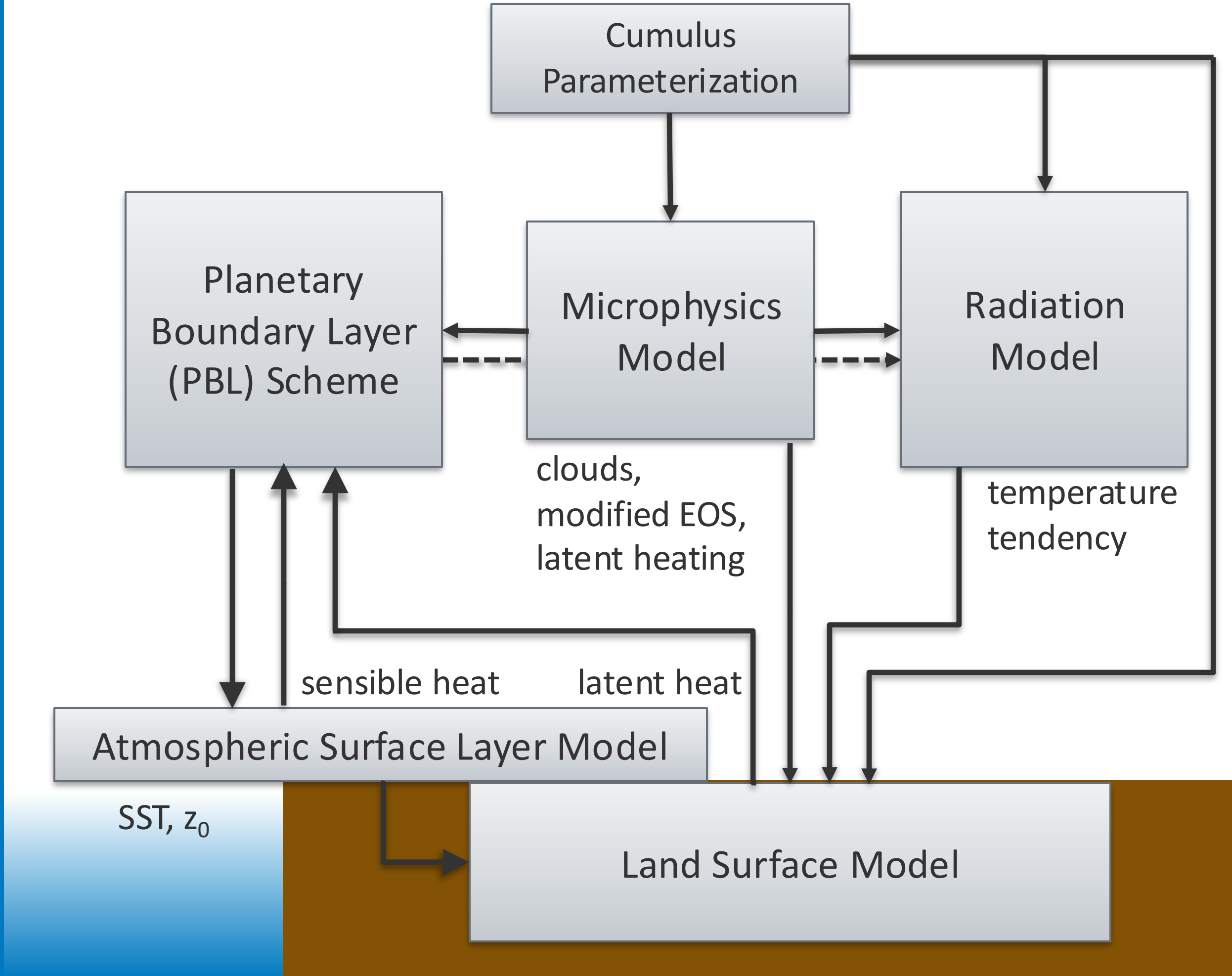
Design Choices

- ERF supports both the **fully compressible** equation set and the **anelastic** approximation (Solvers for the Poisson equation include multigrid and FFT)
  - Compressible: RK3 time integration with acoustic substepping (3rd order)
  - Anelastic: RK2 time integration (2nd order)
  - 2nd–6th order advection
- ERF uses a **height-based, terrain-fitted vertical coordinate** with grid stretching to represent complex terrain
- ERF supports **grid nesting** and **adaptive mesh refinement** (dynamically changing fine regions)
- ERF has local implementations of:
  - Turbulence closures: LES models, Planetary Boundary Layer (PBL) schemes
  - Monin–Obukhov Similarity Theory (with mesoscale corrections)
  - Moisture physics
  - Land surface models
- ERF is in the process of connecting to additional models used by E3SM (these 1-D models have been re-written in C++ and use Kokkos for portability to GPUs)

Compare with: WRF, WRF-LES, PINACLES, CM1, DALES, …

Atmospheric Physics Modeling

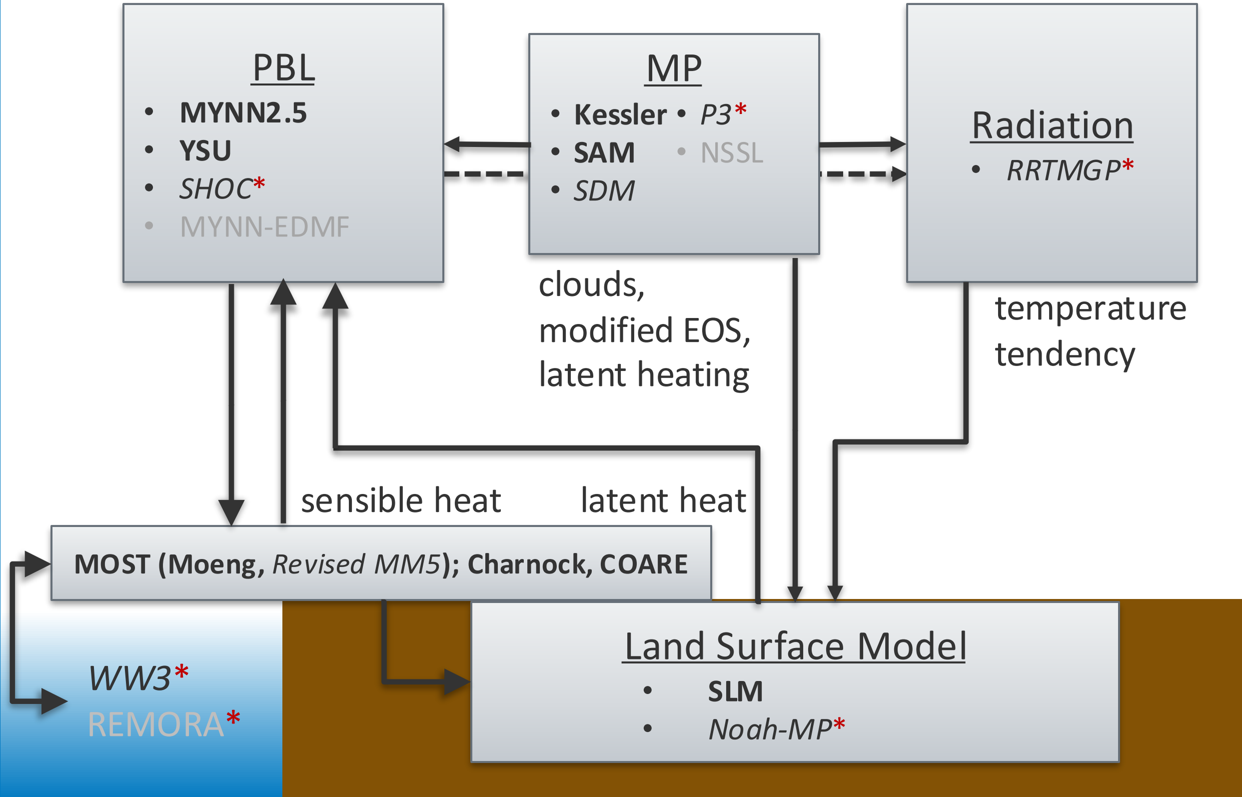
- Dry idealized ABL
- Moist idealized ABL
- Real ABL
  - + microphysics
  - + radiation
  - + land-surface modeling
  - + cumulus parameterization



Atmospheric Physics Modeling in ERF

- Dictated by **community needs**
- Informed by **survey of WRF users**

Turbulence closure
- Smagorinsky model
- Deardorff 1-equation model (prognostic TKE)



Legend
Implemented
In progress
Planned for FY25
(External code)*

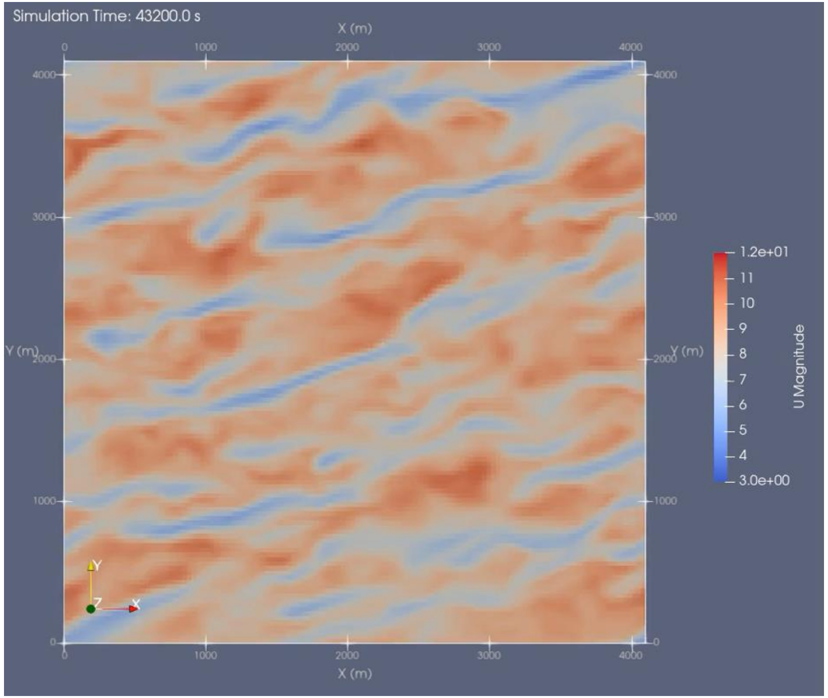
We will support a variety of real-data simulation workflows

	Large-scale data (reanalysis, HRRR, …)	Intermediate Processing	Mesoscale/microscale simulation
WRF	Manual download	WPS tools + `real.exe`	`wrf.exe`
WRF → ERF	Manual download	WPS tools + `real.exe` –or– `ndown.exe`	**`erf_abl`**
WPS → ERF	Manual download	WPS tools	**`erf_abl`**
E3SM → ERF	Run E3SM	See below ↓	**`erf_abl`**
ERF standalone	**Python tools**: HRRR → ERF, E3SM → ERF, … *(others under development)*		**`erf_abl`**

WRF Preprocessing System (WPS) tools: `geogrid.exe`, `ungrib.exe`, `metgrid.exe`

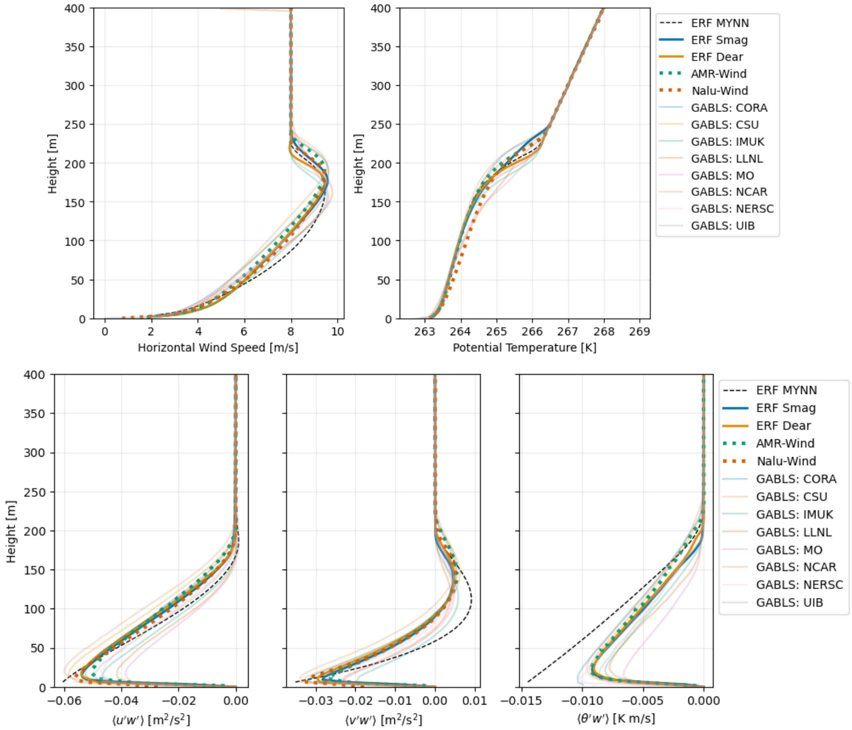
V&V: Dry Atmospheric Boundary Layer

Neutral ABL, WRF-like configuration



Contours of horizontal wind speed at hub height

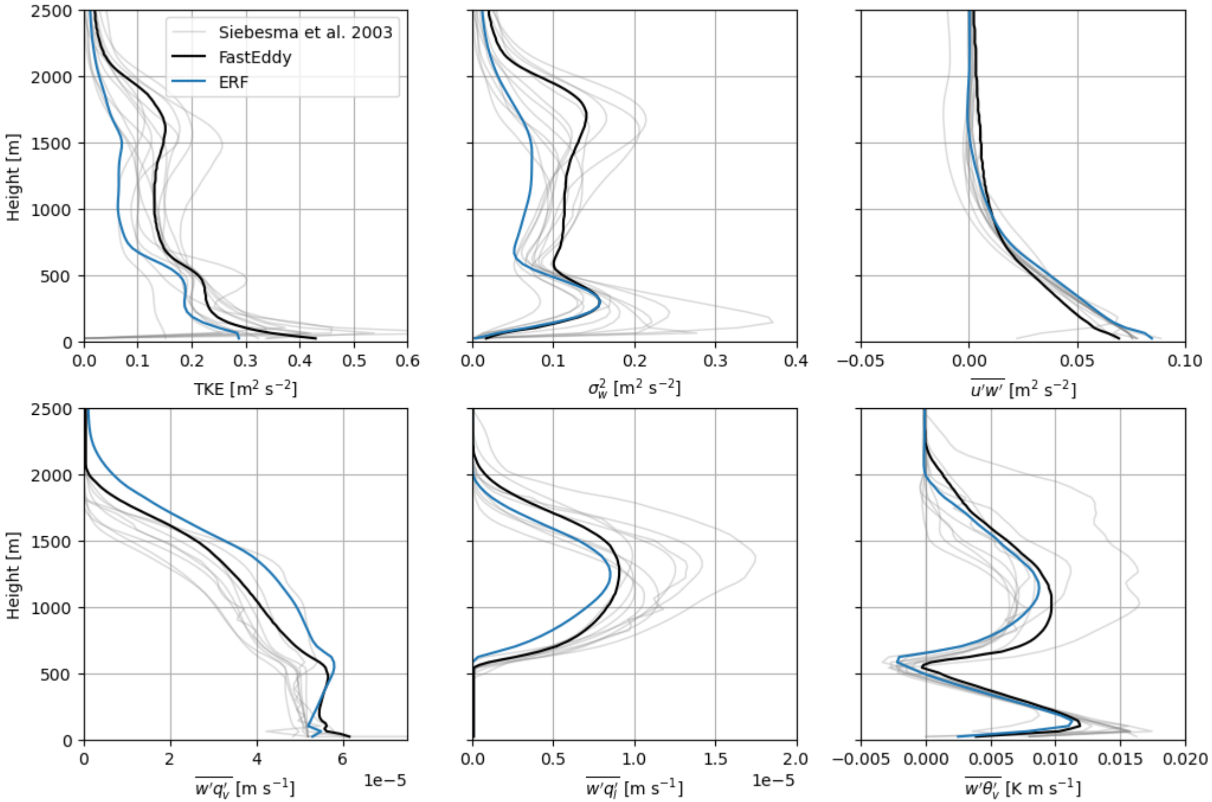
GABLS1 Stable ABL



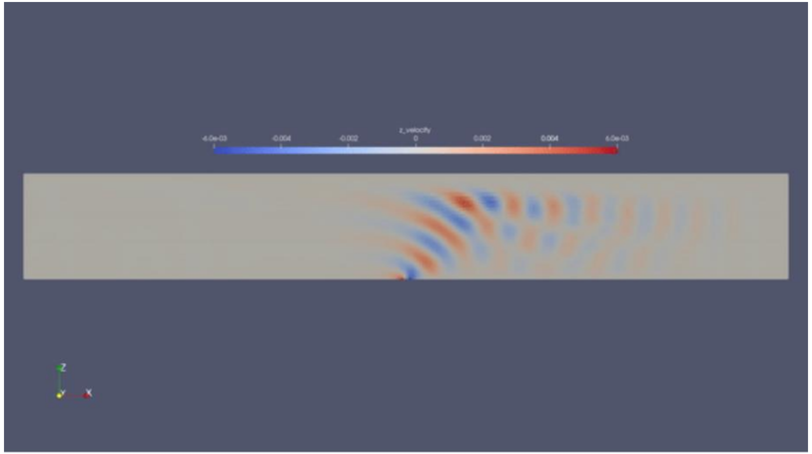
https://github.com/erf-model/validation/blob/main/GABLS1/GABLS1_postproc.ipynb

V&V: BOMEX Shallow Cumulus Convection

- Prescribed surface sensible & latent heat fluxes
- Prescribed mesoscale tendencies with subsidence
- Results sensitive to numerics

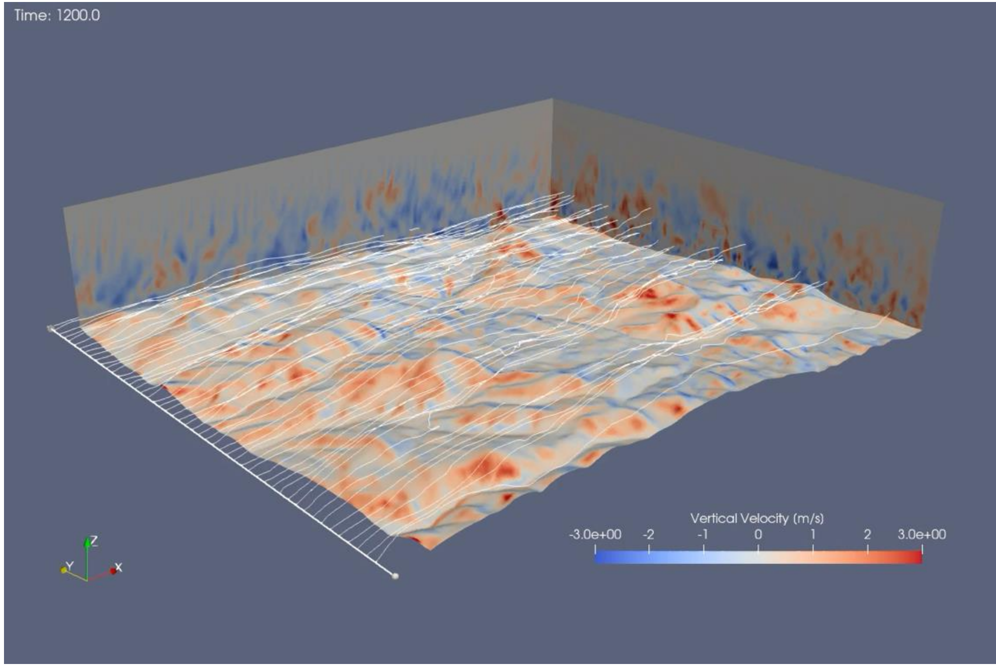


Terrain Flows



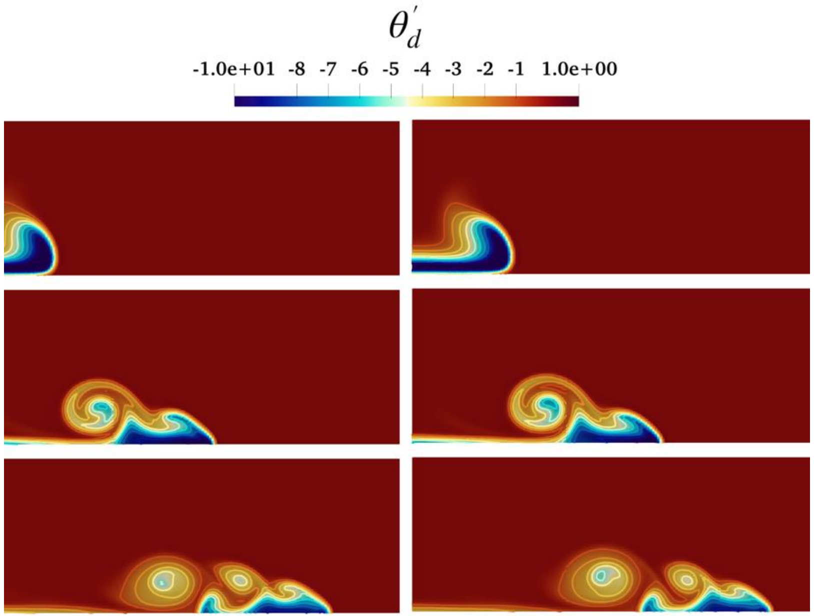
Gravity wave propagation over a Witch of Agnesi hill geometry

Flow over part of the Altamont Pass Wind Resource Area, CA

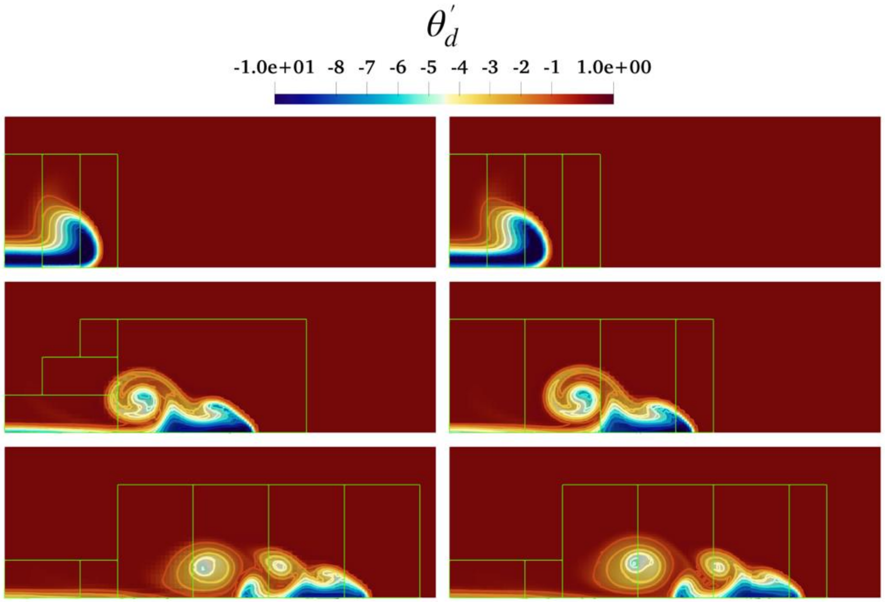


Adaptive Mesh Refinement

Compressible vs Anelastic: **single resolution**



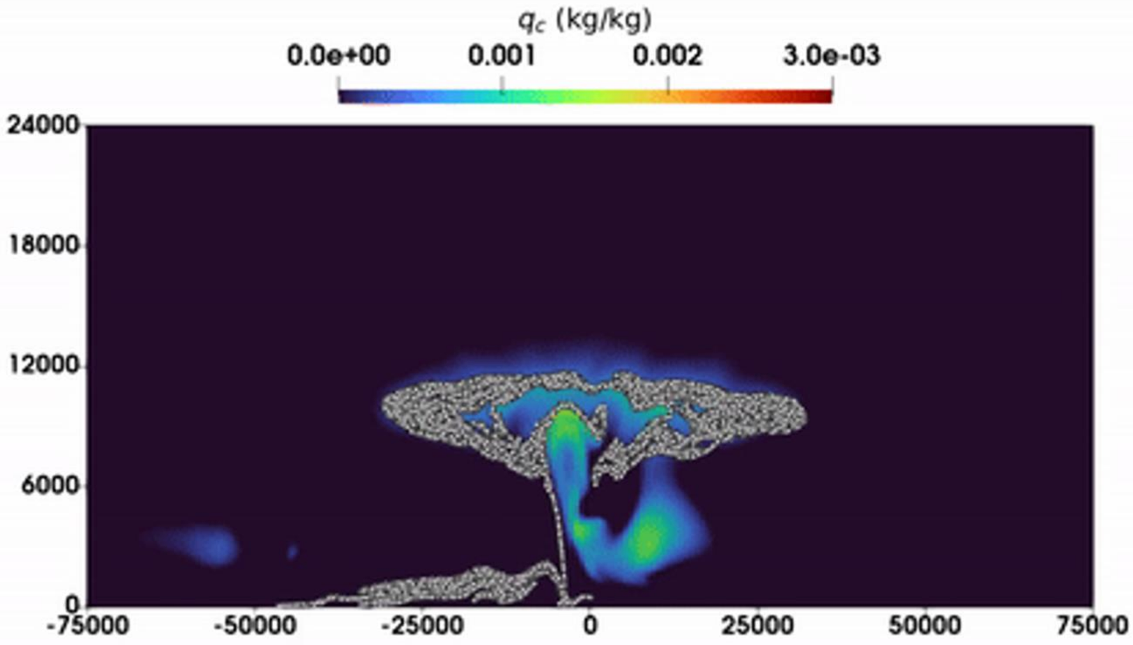
Compressible vs Anelastic: **adaptive mesh refinement**



Density current simulation: Perturbational potential temperature (top to bottom) at t= 300, 600, 900 [s] for (left) compressible and (right) anelastic modes with single level. Contour lines are spaced every 1K.

Same as left figure but with green outline denoting the regions of refinement.

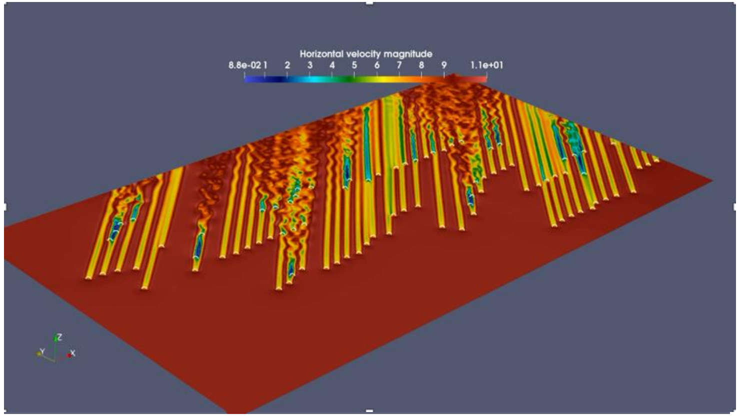
Thunderstorm Outflow



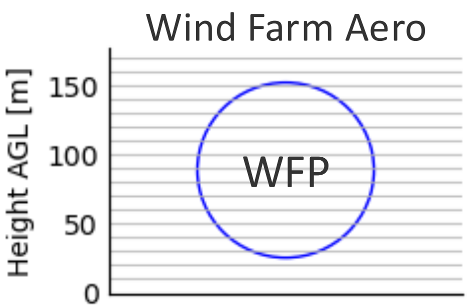
- Squall line simulation, arising from an initial warm, moist bubble
- Particles seeded in initial bubble, advected as passive tracers

Wind Farm & Turbine Modeling in ERF

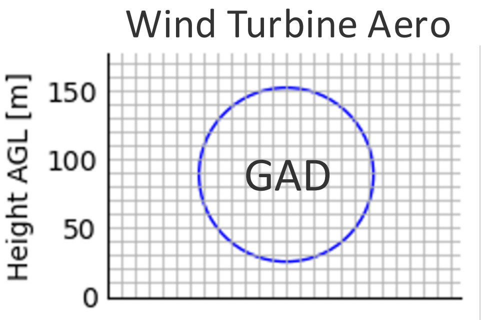
- **Wind Farm Parameterization** (WFP; Fitch et al. 2012, Volker et al. 2015)
- **Generalized Actuator Disk** with BEMT (GAD; Mirocha et al. 2014)
- ERF can be **online coupled** with AMR-Wind (+ OpenFAST)



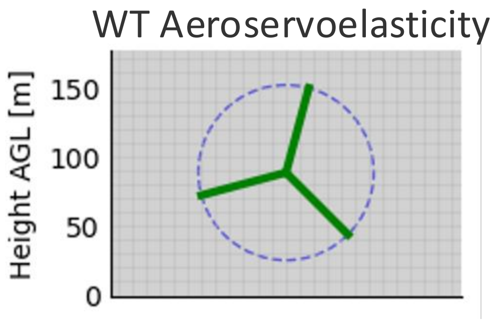
*Velocity magnitude contours at hub height for a wind farm (88 turbines) simulated with **GAD**.*



Horizontal Grid Spacing ~ 1–3 km
Vertical Grid Spacing ~ 10 m



Uniform Grid Spacing ~ 10 m



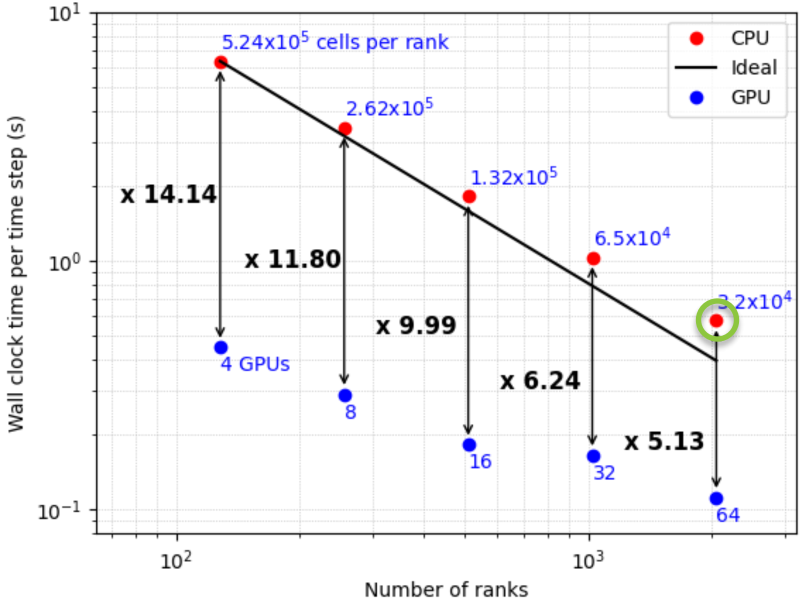
Uniform Grid Spacing ~ 1 m

ERF is scalable on CPUs and GPUs

<u>NERSC Perlmutter</u>
CPU node: 128 core AMD EPYC 7763 (Milan)
GPU node: 4x NVIDIA A100 (Ampere)

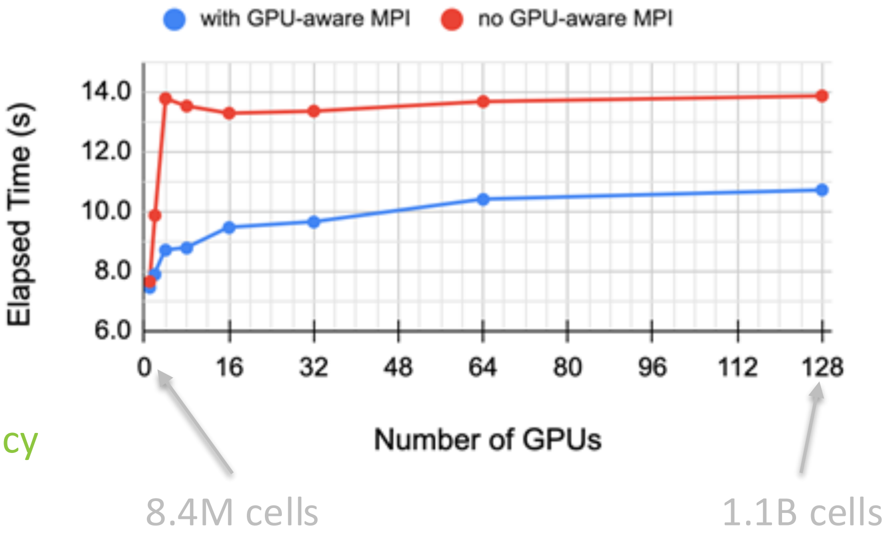
with 5–10x GPU speedup

Strong Scaling (67M cells)



86% parallel efficiency

Weak Scaling

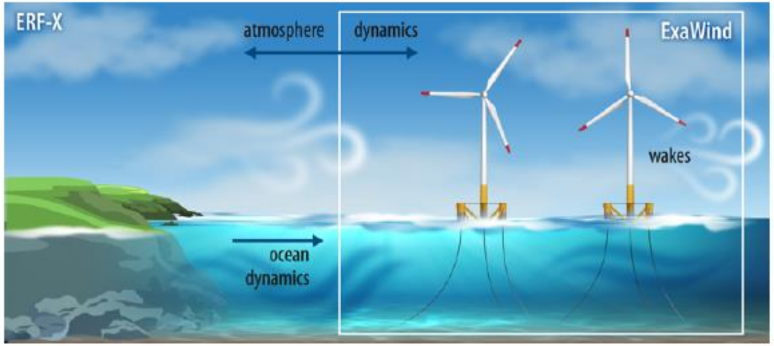


Outlook

- Model description and V&V in paper under submission
  - Email list
- FY25 focus areas:
  - Onshore validation over complex terrain (Tall Towers)
  - Offshore (ORACLE)
  - Extreme Weather (TREXO/STORM)
  - Incorporating PINACLES capabilities
- Working toward ("ERF-X"):
  - Coupling with wave model (WW3)
  - Coupling with ocean model (REMORA, also built on AMReX)
  - Ingesting data from global simulations (e.g., E3SM)

Getting started:

- erf.readthedocs.io
- github.com/erf-model/ERF



High Fidelity Modeling

Polls

Open Discussion

Open Discussion HFM

HFM Software

- ExaWind
  - AMR Wind
  - Nalu Wind
  - OpenTurbine / OpenFAST
  - (+ Tioga, Exawind-manager, amr-wind-frontend)
- ERF

General questions

- What remains unanswered about the HFM area of the WETO Stack?
- What are pain points?
- Where would you like to see focused attention?
- What works well?

Thank you for your time today!

- Have feedback or suggestions for user workshops?
  - Send feedback to Rafael.Mudafort@nrel.gov
  - Anonymous feedback form to be sent as a follow up

- Software repositories:
  - AMR Wind: https://github.com/exawind/amr-wind
  - Nalu Wind: https://github.com/exawind/nalu-wind
  - OpenTurbine: https://github.com/exawind/openturbine
  - ERF: https://github.com/erf-model/ERF

- WETO Stack Site: https://nrel.github.io/WETOStack
  - Workshop recordings (including this one!)