

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





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

NREL's active WETO projects



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

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 Multiscale Integration of Control Systems (EMS/DMS/BMS) Wind Standards Development 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





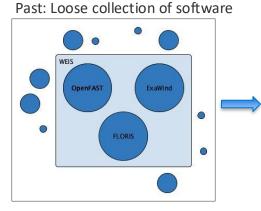


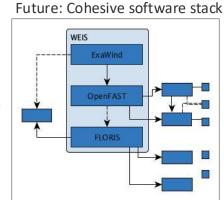




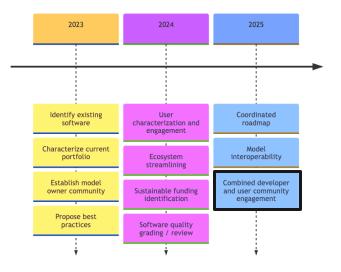
Holistic Modeling Project

Objective





Project Timeline



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



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



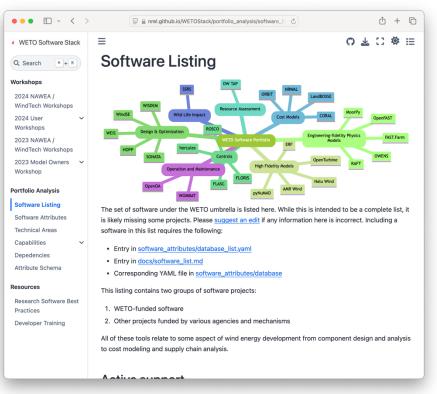
nrel.github.io/WETOStack

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
- <u>Software Listing</u>: active, inactive, and "other status" software



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
- <u>Software Listing</u>: active, inactive, and "other status" software
- <u>Software Attributes</u>: tabulated data describing each software, defined by an <u>Attribute Schema</u>

WETO Software Stack	≡				0) 7	23	ø	iΞ
Q Search + K	Tur	bine M	lodelin	g					
Workshops	► 9	Show code ce	II source						
2024 NAWEA / WindTech Workshops 2024 User ~ Workshops	ti /opt	urbine_df = t/hostedtool	turbine_df[cache/Pythor	2343.py:10: FutureWarning pd.DataFrame(turbine_d h/3.11.10/x64/lib/python3 , pd.Int64Index):	f["scope"].tolist()]	.isin(["tı	urbin	e", '
2023 NAWEA / WindTech Workshops 2023 Model Owners ~ Workshop				cells ystem location LIKE '%any%					
				,					
Software Listing									
		Run Query	Reset Data		-				
Software Attributes			Reset Data		- 				
Software Listing Software Attributes Technical Areas	F	Run Query index	Reset Data	scope	system_location		time_	_doma	in
Software Listing Software Attributes Technical Areas Capabilities	F		Reset Data		system_location ['any']		time_ dynam	2	in
Software Listing Software Attributes Technical Areas Capabilities Depedencies Attribute Schema	1 2	index	Reset Data	scope			1277	ic	in
Software Listing Software Attributes Technical Areas Capabilities Depedencies Attribute Schema Resources	1	index amr-wind	Reset Data	scope ['turbine', 'farm', 'atmos	['any']		dynam	ic ic	in
Software Listing Software Attributes Technical Areas Capabilities Depedencies Attribute Schema Resources WETO Software Best	1	index amr-wind fastfarm	Reset Data	scope ['turbine', 'farm', 'atmos ['farm', 'turbine', 'atmos	['any'] ['any']		- dynam dynam	ic ic	in
Software Listing Software Attributes Technical Areas Capabilities Depedencies Attribute Schema Resources WETO Software Best Practices	1 2 3	index amr-wind fastfarm flasc	Reset Data	scope ['turbine', 'farm', 'atmos ['farm', 'turbine', 'atmos ['turbine', 'farm', 'system']	['any'] ['any'] ['any']	•	dynam dynam steady	ic ic	in
Software Listing Software Attributes Technical Areas Capabilities Depedencies Attribute Schema Resources WETO Software Best	1 2 3 4	index amr-wind fastfarm flasc floris	Reset Data	scope ['turbine', 'farm', 'atmos ['farm', 'turbine', 'atmos ['turbine', 'farm', 'system'] ['turbine', 'farm', 'syste	['any'] ['any'] ['any'] ['any']	•	dynam dynam steady steady	ic ic ic	in
Software Listing Software Attributes Technical Areas Capabilities Depedencies Attribute Schema Resources WETO Software Best Practices	1 2 3 4 5	index amr-wind fastfarm flasc floris hercules	Reset Data	scope ['turbine', 'farm', 'atmos ['farm', 'turbine', 'atmos ['turbine', 'farm', 'system'] ['turbine', 'farm', 'syste ['turbine', 'farm', 'atmos	['any'] ['any'] ['any'] ['any'] ['any']	: : : :	dynam dynam steady steady dynam	ic ic	in
Software Listing Software Attributes Technical Areas Capabilities Depedencies Attribute Schema Resources WETO Software Best Practices	1 2 3 4 5 8	index amr-wind fastfarm flasc floris hercules landbosse	Reset Data	scope ['turbine', 'farm', 'atmos ['farm', 'turbine', 'atmos ['turbine', 'farm', 'system'] ['turbine', 'farm', 'syste ['turbine', 'farm', 'atmos ['turbine', 'system']	['any'] ['any'] ['any'] ['any'] ['any'] ['any']		dynam dynam steady steady dynam steady	ic ic ic	in

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
- <u>Software Listing</u>: active, inactive, and "other status" software
- <u>Software Attributes</u>: tabulated data describing each software, defined by an <u>Attribute Schema</u>
- <u>Best Practices</u>: guidance for creating software within the context of WETO and the research environment

••• • • < >	😑 🔒 nrel.github.io/WETOStack/software_dev/best_practices 🖒 📩 📩						
WETO Software Stack	≡ 0 7 ∷ ¢ ⊞						
Q Search *+ K	Architecture and Design						
Workshops 2024 NAWEA / WindTech Workshops 2024 User Workshops 2023 NAWEA / WindTech Workshops	If you think good architecture is expensive, try bad architecture. —Brian Foote and Joseph Yoder, Clean Architecture: A Craftsman's Guide to Software Structure and Design In the development of any complex system, the design and its implementation are either explicit or implicit Emploid design implementation are either explicit or						
2023 Model Owners Workshop Portfolio Analysis Software Listing Software Attributes	implicit. Explicit design involves identifying relationships between modules, composition of data structures, and flow of data prior to writing code, whereas an implicit design evolves during the process of writing new code. In open source software, an explicit design process is critical to allowing the project to grow beyond a single developer, and the consequence of an implicit design process is the common case of technical debt.						
Technical Areas	Software Design Process						
Capabilities ~ Depedencies Attribute Schema Resources WETO Software Best Practices	 Primarily, an explicit design process involves identifying the fundamental principles of a particular design how it is expected to function in various aspects. This process should result in two statements: 1. The <u>parti</u>, a description of the fundamental, driving design intent as a brief text (one or two sentences) or a simple diagram 2. A list of requirements that the parti and its implementation should satisfy 						
Developer Training	implementation satisfies the parti. In other words, these are the tests for the design. Upon establishing this information, it should be codified into a design document and style guide that are made publicly available to all developers such as in online documentation. There are various levels of fidelity to consider when designing a software system:						

WETO Software Best Practices

nrel.github.io/WETOStack

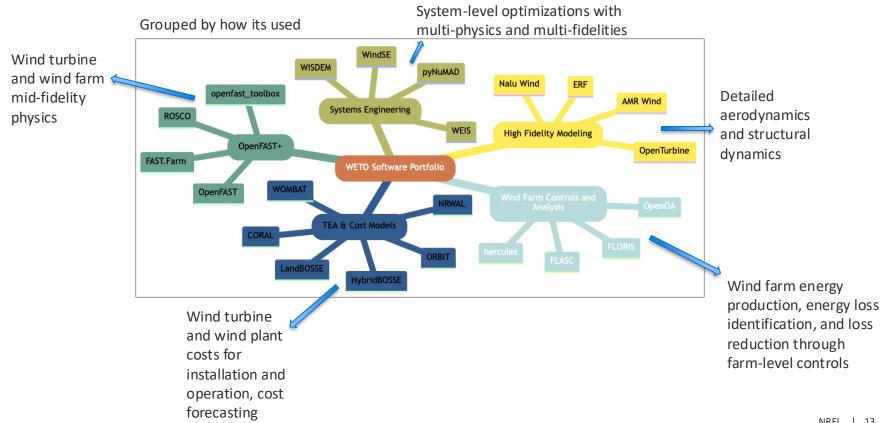
- Accessibility: How to obtain and integrate the software into your work
- Usability: How to get up to speed and become proficient at executing the software and understanding the results
- Extendability: How new features, bug fixes, and general maintenance are incorporated into the software by regular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

WETO Software Stack	≡	0	*	53	Ø	≣
Q Search *+K	Contents					
Workshops	WETO Software Best Practices					
2024 NAWEA /	Summary of Best Practices					
WindTech Workshops	<u>Accessibility</u>					
2024 User 🗸	Prerequisite Knowledge					
Workshops	<u>Distribution</u>					
2023 NAWEA / WindTech Workshops	• <u>Usability</u>					
2023 Model Owners	User Interface					
Workshop	<u>Command Line Interface</u>					
	Input and Output Files					
Portfolio Analysis	Error Messages					
Software Listing	Metadata					
Software Attributes	• Extendability					
Technical Areas	<u>Code Style</u>					
Capabilities 🗸	The Zen of Python					
Depedencies	Architecture and Design					
Attribute Schema	Software Design Process					
	Design Patterns					
Resources	Version Control					
WETO Software Best	Collaborative Workflows with GitHub					
Practices	Pull Requests					
Developer Training	Continuous Integration: Automating Tests, Compliance, and Delivery					
	Appendix - RSEs: The engineers behind research software					
	<u>RSE Value Recognition</u>					
	Career Growth and Trajectory					

WETO Software Stack

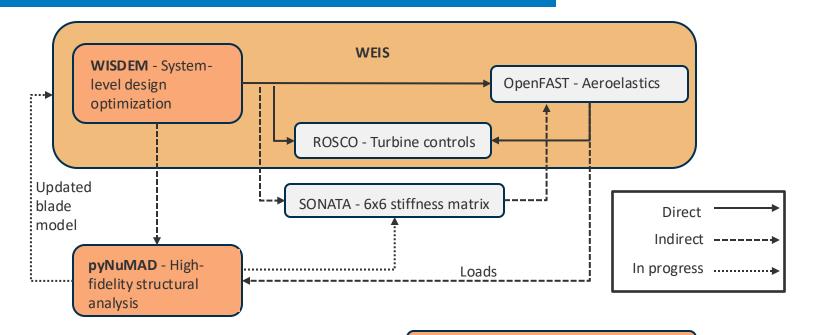
Overview

WETO Software Stack



Systems Engineering

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

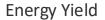


WindSE - RANS for systems engineering

Adapted from Big Adaptive Rotor (BAR) project

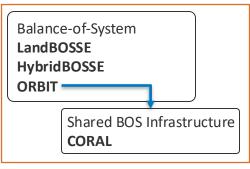
Technoeconomic Analysis / Cost Modeling

Matt.Shields@nrel.gov



Wind farm AEP estimate FLORIS

CapEx



OpEx

Operation & Maintenance **WOMBAT**

NRWAL: Offshore wind system cost and scaling model

Wind Asset Value Estimate WAVES

Wind Farm Controls and Analysis

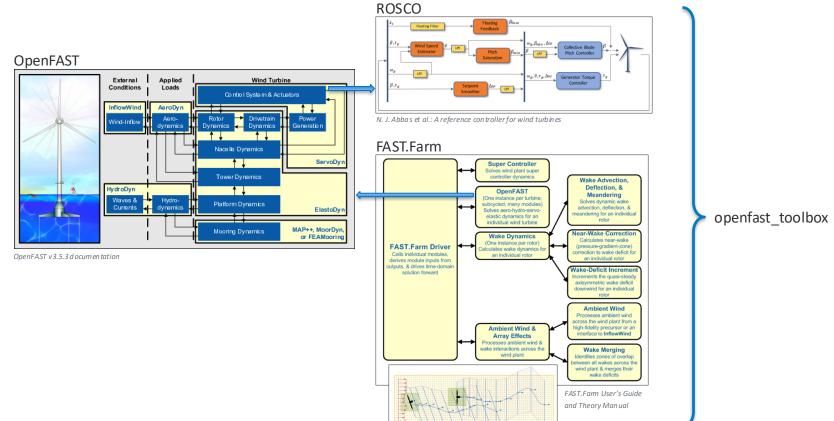
FLORIS: Steady-state FLASC: Validate FLORIS model with modeling, farm SCADA, compare control methods controls optimization 1.1 1.0 3 0.9 t 0.8 SMV1 6 0.7 Field Baseline le Bois Bouillette e 0.6 Field Controlled --- FLORIS Baseline, Measured Offsets 0.5 --- FLORIS Controlled, Measured Offsets SMV2 Sole d $\widehat{}$ 0.15 Field Gain --- FLORIS Gain, Measured Offsets SMV3 0.10 FLORIS Gain, Ideal Offsets FLORIS Gain, Expected Offsets 0.05 Sole du Bo le Chemin de Pressoir 0.00 SMV4 -0.056 -0.10 195 200 205 210 215 220 225 230 235 240 la Tombe SMV5 le de Verr Wind Direction (°) Sodar SMV6 le Bosauet Anne-Jean a Fosse à Cat SMV7 **OpenOA**: Characterize plant performance and quantify sources of operational loss 10.0 Wind Speed (m/s

Paul.Fleming@nrel.gov

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

Jason.Jonkman@nrel.gov

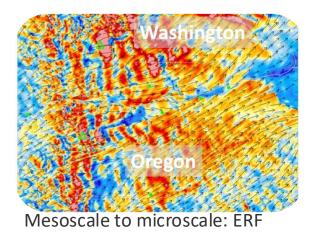
OpenFAST+



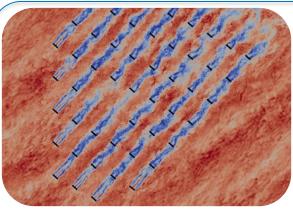
Michael.A.Sprague@nrel.gov

High Fidelity Models

ExaWind

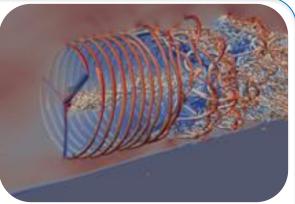


- Regional scale weather down to actuator line and actuator disc models
- Scales <1 km to 1000 km
- WRF numerics & models, built on AMReX
- GPU compatible
- Compressible, incompressible, anelastic



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

VREL |

ExaWind

Mike Sprague

What is ExaWind?

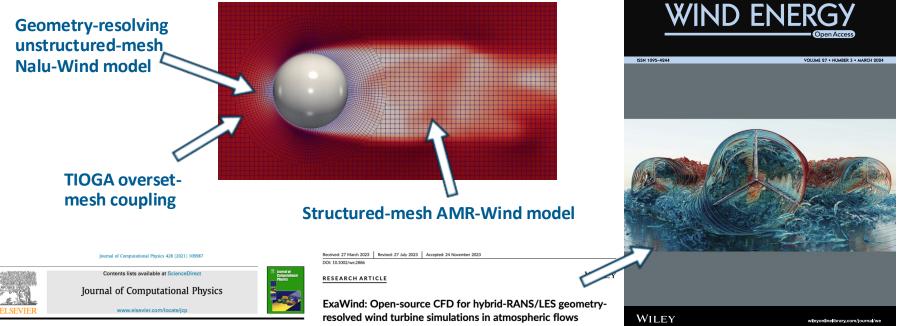
- ExaWind is an open-source suite of codes designed for highfidelity and mid-fidelity modeling and simulation of wind turbines and wind farms
 - https://github.com/Exawind/
 - Nalu-Wind, AMR-Wind, OpenFAST/OpenTurbine
- Provide predictive tool for understanding wind energy and to validate and improve 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



Ganesh Vijayakumar¹

Overset meshes for incompressible flows: On preserving Ashesh Sharma¹ | Michael J. Brazell¹ Shrevas Ananthan² | Lawrence Cheung³ | Nathaniel deVelder³ accuracy of underlying discretizations Marc T. Henry de Frahan¹ | Neil Matula³ | Paul Mullowney⁴ | Jon Rood¹ Philip Sakievich³ Ann Almgren⁵ Paul S. Crozier³ Michael Sprague¹

Ashesh Sharma^{a,*}, Shreyas Ananthan^a, Jayanarayanan Sitaraman^b, Stephen Thomas^a, Michael A. Sprague^a



AMR-Wind for actuator-line/disc simulations

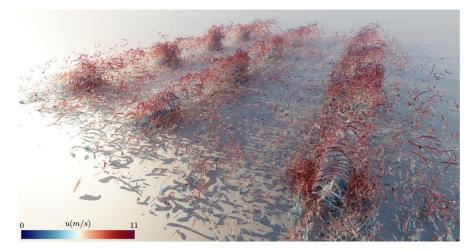
AMR-Wind methods paper submitted to Wind Energy

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

Michael B. Kuhn¹, Marc T. Henry de Frahan¹, Prakash Mohan¹, Georgios Deskos¹, Matthew Churchfield¹, Lawrence Cheung³, Ashesh Sharma¹, Ann Almgren², Shreyas Ananthan¹, Michael Brazell¹, Luis A. Martinez-Tossas¹, Regis Thedin¹, Jon Rood¹, Philip Sakievich⁴, Ganesh Vijayakumar¹, Weiqun Zhang², and Michael A. Sprague¹

¹National Renewable Energy Laboratory, Golden, CO, 80401, USA ²Lawrence Berkeley National Laboratory, Berkeley, CA, USA ³Sandia National Laboratories, Livermore, CA, 94550, USA ⁴Sandia 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 qcriterion 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)



DOE Exascale Computing Project



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





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. Vijavakumar S. Yellapantula*



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

I. Yamazaki*



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



A. Almgren W. Zhang Aaron Lattanzi



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
 - <u>https://github.com/Exawind/exawind-benchmarks</u>
 - <u>https://exawind.github.io/exawind-benchmarks/</u>
 - 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

https://github.com/Exawind/openturbine

Dev Team

Derek Slaughter Faisal Bhuiyan David Dement Paul Crozier Mike Sprague

Funding

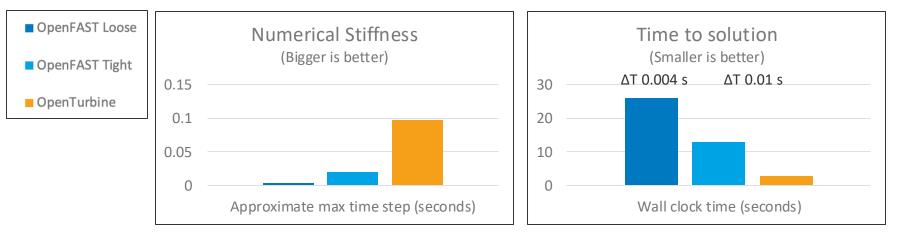
DOE EERE WETO DOE Office of Science FLOWMAS Energy Earthshot Research Center

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

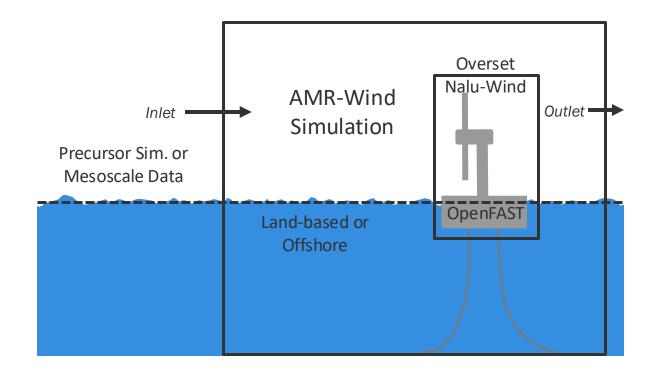
OpenTurbine Next Steps

- 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 fluidstructure-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

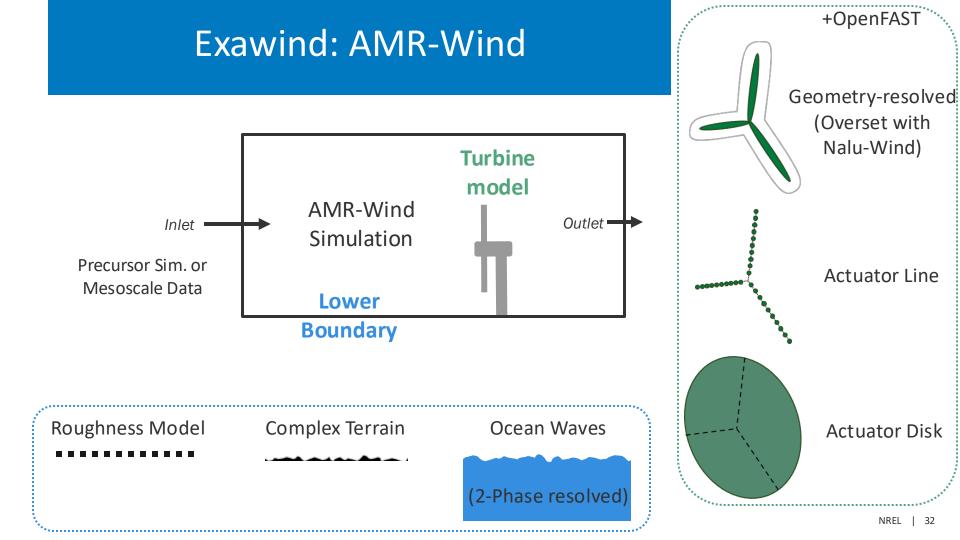
Michael Kuhn

Exawind: AMR-Wind



Far-field flow solver for turbine simulations:

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



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)

Turbulence Modeling – Boundary:

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

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

$$\frac{\partial k}{\partial t} + \frac{\partial k u_j}{\partial x_j} = \frac{\partial}{\partial x_j} \left(2\nu_t \frac{\partial k}{\partial x_j} \right) - \tau_{ij} S_{ij} + \frac{g}{\theta_{\circ}} \tau_{\theta_3} - C_{\epsilon} \frac{k^{3/2}}{l},$$

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

AMR Wind Background

Structured mesh (Cartesian); patch-based refinement

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



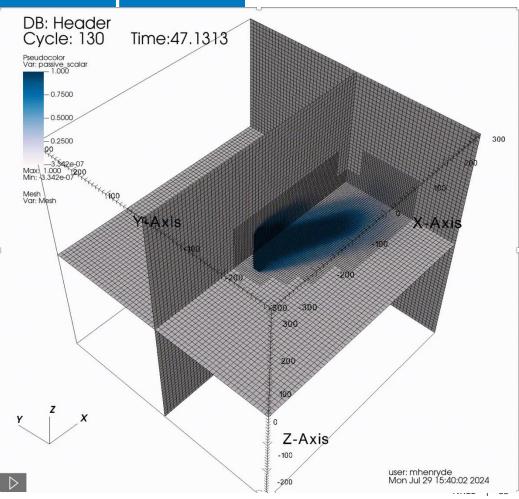
AMR Wind Back

Structured mesh (Cartesian); patch-b

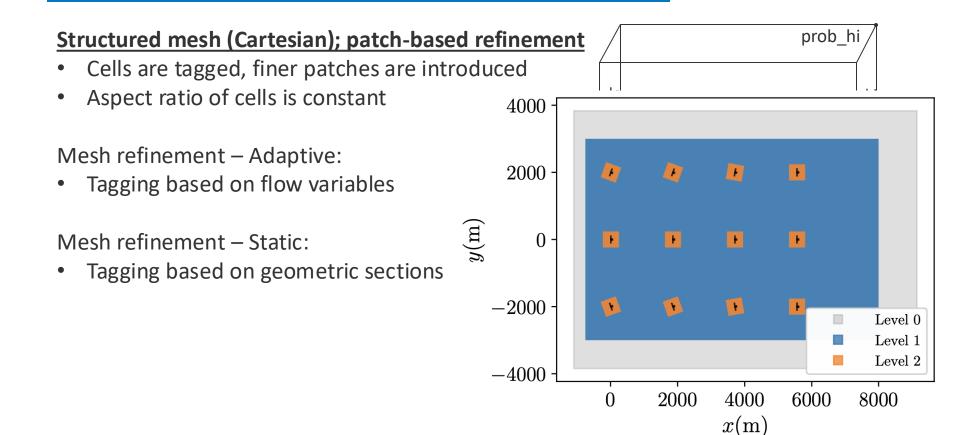
- Cells are tagged, finer patches are i
- Aspect ratio of cells is constant

Mesh refinement – Adaptive:

• Tagging based on flow variables



AMR Wind Background



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:

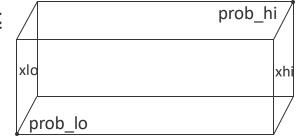
• Tagging based on flow variables

Mesh refinement – Static:

• Tagging based on geometric sections of domain

AMR-Wind Frontend:

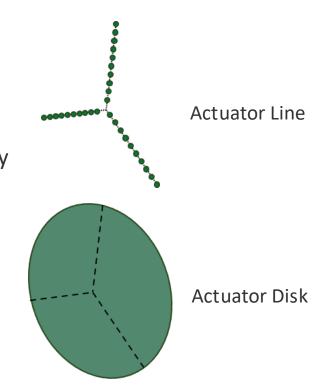
- GUI or Python interface to set up input file
- Helpful for many refinements, many turbines
- Powerful post-processing tools included



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



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

Online Documentation:

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.

\Box Walkthrough

Compiling using Spack, with Exawind-Manager

Precursor (ABL) walkthrough

Turbine simulation walkthrough

Compiling:

- 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
- \$ spack install
 \$ spack load amr-wind

□ 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	<pre>= bndry_file.native</pre>
ABL.bndry_io_mode	= 0
ABL.bndry_planes	= xlo
ABL.bndry_output_start_time	= 7200.0
ABL.bndry_var_names	= velocity temperature tke

₹EL

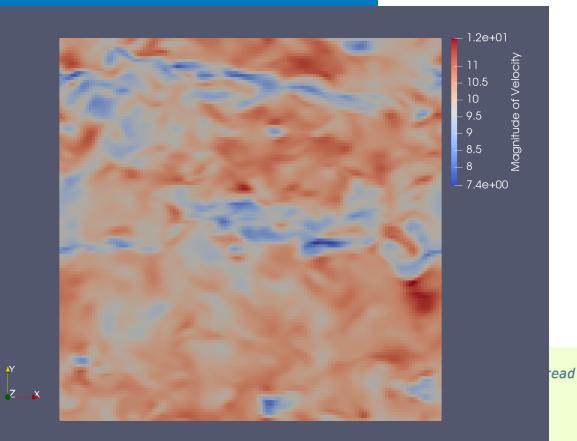
43

□ Walkthrough

Compiling using Spack, with Exawind-Manager

Precursor (ABL) walkthrough

Turbine simulation walkthrough



□ Walkthrough

Compiling using Spack, with Exawind-Manager

Precursor (ABL) walkthrough

Turbine simulation walkthrough

Running turbine simulation:

• Place turbines in domain

Actuator.T0.type Actuator.T0.openfast_input_file Actuator.T0.base_position Actuator.T0.rotor_diameter = TurbineFastDisk

- = T0_0penFAST/NREL-2p8-127.fst
- = 640.0 1280.0 0.0
- = 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

<pre>io.restart_file</pre>	=	/spinup/
ABL.bndry_file	=	/precurs
ABL.bndry_io_mode	=	1
ABL.bndry_planes	=	xlo
ABL.bndry_output_start_time	=	7200.0
ABL.bndry_var_names	=	velocity t

../spinup/chk14400 ../precursor/bndry_file.native 1 # 0 = write, 1 = read xlo 7200.0 velocity temperature tke

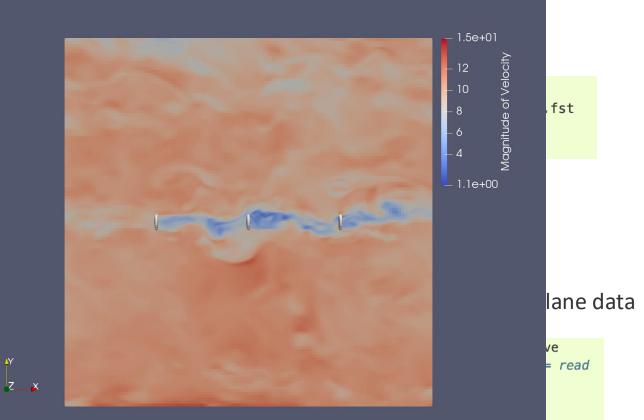
45

□ Walkthrough

Compiling using Spack, with Exawind-Manager

Precursor (ABL) walkthrough

Turbine simulation walkthrough



AMR Wind Roadmap

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

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!

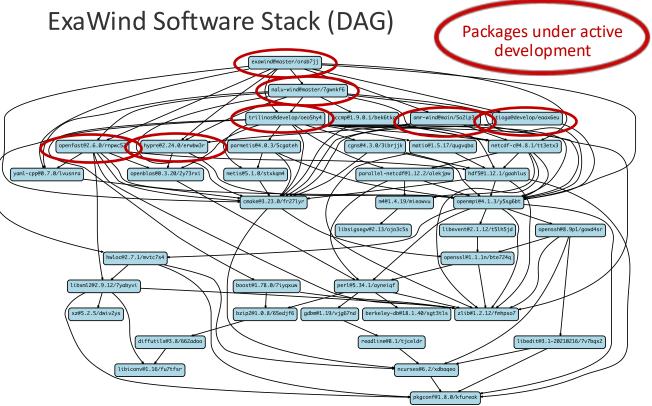
To get on our user mailing list, send an email to *amr-wind-maintainers@groups.nrel.gov* GitHub is preferred for most inquiries, but this email can also be used for direct correspondence.

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 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

Package

Some stats on problem sizes

- Main logic program is:
 - ~250

Lawrence Livermore

- 20 or
- 933 I
- Proble
- Com num
- gmał most
- gnuc

0 rules		dependencies		
optimization criteria	gnuconfig	1	150	
lines of ASP code	zlib	527	30,095	4
em instances can vary quite a bit	gmake	527	30,160	
nmon dependencies get us some magic	openmpi	527	109,021	
nbers	qt	527	109,029	
ake's optional dependency on guile makes	trilinos	694	224,142	
st solves consider at least 527 packages	root	699	146,372	
config is notably very simple ©	mfem	714	273,078	
	r-condop	774	142,212	
	pro-	919	515,574	
	exawind	820	322,535	
National Laboratory			NNS 13	

Possible

Facts

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

🛐 exawind-manager (Public)	😒 Edit Pins 🚽	⊙ Unwatch 5	→ ²⁹ Fork 8 → ☆ Star 2 →
문 main ▾ 우오 Branches ▷ 0 Tags	Q Go to file t +	<> Code -	About 🕸
🗊 jrood-nrel Update Spack. (#180)	c443edc · 2 days ago	174 Commits	Dev-Ops tooling and configuration management for ExaWind development
Configs	Enable wind-utils and waves2amr. (#179)	last week	🛱 Readme
repos/exawind	Update Spack. (#173)	last week	 ✓ Activity E Custom properties

NRFI

Continuous testing, monitoring, updating

		Con	figure		alid .		Test		
Site	Build Name	Error	Warn V	Error	Warn	Not Run	Fail	Pass	Start Time ¥
filis.hpc.nrel.gov	O Monoterfigce@ir12.3.0+am_wind_gpu=asan-cdash_submit+cude=gpu=eware-mpi-ipo+nalu_wind_gpu=ninja=room=syci build_system=omake build_type=Release cdst_suppi=None_cdst_submits=inde=relemanc_golds=indext	0		0	0	0	0	12	4 hours ago
lis.hpc.nrel.gov	A Gmasterficiang@+17.0.6-amr_wind_gou-asan-odash_submit-cuda-gou-aware-mpi-too-nalu_wind_gou-ninja-norm-sycl build_system-omaixe build_type=Debug cest_angl=None_ponention-maixe reference_golds-clickaut	o	0	o	0	٥	0	12	2 hours ago
lis.hpc.nrel.gov	A BinasterligccB-12.3.0-am, wind, gpu-asan-colabl, submit-cuda-gpu-aware-mpi-ipo-nalu, wind, gpu-ninja-ncom-syci build, system-cmake build, type-Release cetst, ange-Nore generator-make reference, golds-indexut	٥	0	o	D	٥	0	12	4 hours ago
lis.hpc.nrel.gov	Comparison of the second	0	0	0	0	0	0	12	4 hours ago

Nalu-Wind # builds

		Con	tigure	B	uild		Test		
Site	Build Name	Error	Warn	Error	Warn	Not Run	Fail ¥	Pass	Start Time ¥
ellis.hpc.nrel.gov	0 master/spc8=12.2.0-man-boost-catalyst-cataly, submit-cade-the-thi-gou assee mpin/spc=-to-inigi-spenfast-joi-nom-shand-tests-loga-triinos- asters-majes-unit-lasts-wind-utila abs, tol-15 build, system-canale build, type-filasee cleat, arge-tone cude, ar Comparison of the test-set of the test of tes	0	0	0	50	0	1	10	4 hours ago
ellis.hpc.nrel.gov	A maturingcole 12.2.0-asses-boost-catalyst-cataly, submit-caste-files-thi-gos-asses-reprint/pre-too-reprint-generate-pro-recenvelwared-tests-togge-trilinos- solvers-umpire-unit-tests-vend utila abs, tol=15 build, system-cmaile build, type-filesee ctest, args-Nore generat	0	0	0		0	art.	14-14	4 hours ago
eliis.hpc.nrei.gov	manter fickurgiti 17.0.8-aan-boost-catayst-catan, submt-cuta-filter thi-gou aware mpi-hyper-joo-mija-opentat-joo-room-shared-tests-toga-trilinoa- solvera-unpievunit-tests-wind-uits abs, toi-te-15 build, system-cmake build, type-Reisse ctest, ags-Nore gener	0	0	o		0	a.	14-14	4 hours ago
ilis.hpc.nrei.gov	0. matter https://doi.org/bit/12.0.8-aaan-boot-catalysti-clash, submt-cuta-fflw-thi-gau aware repii-hyper-too-mija-opentati-jic-mom-shared-tests-tioga-triinos- notwar-unpre-unit-tests-wind-site abs_toi-te-15 baid_system-cmake buid_type-Debug ctest_arga-hone generat ####################################	0	0	a		o		14	4 hours ago

Vind 4 builds

		Con	ligure		uiki		Test		
Sito	Build Name	Error	Warn	Error	Wern	Not Run	Fail V	Pass	Start Time
ellis.hpc.nrel.gov	3. Marshipped-12.30-asan-assert-icitati, submit-clargiday-coda-gou assert- main inter-field-anglese-po-main-equivated-ingia-operature-opera-shared-aych-testa-tay, polite-ungrise-waved/arm build, system-cmain build, type-Resaus and upper type and upper dynamic and upper type-Resaus and upper type and upper type and upper type-Resaus and upper type and up	o	1	D	0	O	•**	124****	4 hour ago
lis.hpc.nrel.gov	3 Amanticiangite 17.6 &-sam-accent-citaty, submit-citagridg-cuta-gou asseement indication processing and the submit procesing and the submit processing and the submit proc	0	0	0	0	0		127''*	12 hou ago
ilis.hpc.nrel.gov	A manifolding/bi-17.6.6-sour-secont-clash, submt-clash(s)-code-gou asser- mant/scharb) pro-pointain-reprinted-reprinted-reprinted-reprinted-systematic-tory, profile-unpire resultant build, system-cinala build, spec-filesas and specific growthere provided reprinted-reprinted-reprinted-systematic-tory, profile-unpire resultant build, system-cinala build, spec-filesas and specific growthere provided reprinted-reprinted-reprinted-systematic-tory, profile-unpire resultant build, system-cinala build, spec-filesas and specific growthere reprinted reprint	(0	0	0	0	0		132***	12 hou ago
ilis.hpc.nrel.gov	A mark/spcd+12.3.0-sax+-sacet-colar), sizeri-clargidg-code-gou exere- mpi-idd-bics-hype-boinasa-mpi-neted-wrigi-opertait-opermp-nom-shared-syci-tests-tiny_profile-unpire-exeres2am built_system-cmains.built_system- exercise_sam-hore exeremity-main r	0	0	D		0	0	133***	4 hour ago

🙃 Summary	Triggered via push 2 days ago	Status	Total duration
Jobs	🚱 marchdf pushed 🔶 8a9e821 main	Success	32m 4s
Formatting			
Save-PR-Number	ci.yml		
🥝 CPU (ubuntu-latest, Release, N	on: push		
🥝 CPU (ubuntu-latest, Debug, Op			
🥝 CPU (ubuntu-latest, Debug, No	Second Se	📀 gpu-hip	22m 41s
🤣 CPU (macos-latest, Release, No		SPU-SYCL	31m 36s
🤣 GPU-CUDA (11.4)		Lint-clang-tidy	15s
gru-hip	Save-PR-Number 0s	Lint-codeql	30m 1s
GPU-SYCL		Lint-codespell	8s
Lint-clang-tidy		 Docker 	14m 2s
🥝 Lint-codeql		DUCKEI	14111 23
Lint-codespell		latrix: CPU	
🥝 Docker		4 jobs completed	
Run details		Show all jobs	
Ö Usage			
ා Workflow file		latrix: GPU-CUDA	
		1 job completed Show all jobs	

ExaWind: Programming models for performance portability

Nalu-Wind

Kokkos abstraction layer

TIOGA

Currently restricted to CPUs

AMR-Wind	OpenFAST
AMReX abstraction layer	 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	Image: A start of the start	Image: A start of the start		
Intel GPUs		In progress		

ExaWind Binary Device Modes

Device	AMR-Wind	Nalu-Wind
CPU	Image: A start of the start	 Image: A set of the set of the
GPU	Image: A start and a start	

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

Problem description

- ABL, neutral atmospheric boundary layer problem
- 5 x 5 x 1 km³
- 10 m grid spacing

<u>Summit</u>

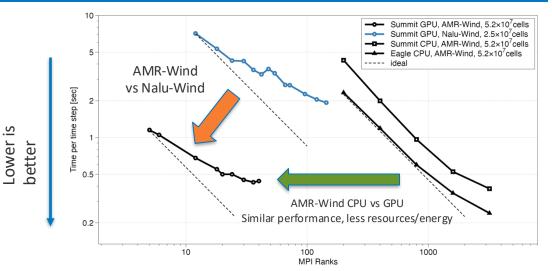
- 4608 compute nodes, each with:
 - 2x IBM POWER9 CPUs w/ 42 cores total
 - o 6x Nvidia V100 GPUs

<u>Eagle</u>

- 2618 compute nodes, each with:
 - $\circ~$ 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

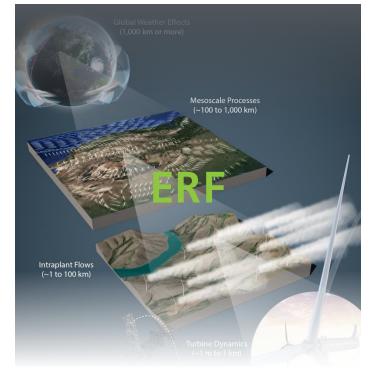
Energy Research & Forecasting

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

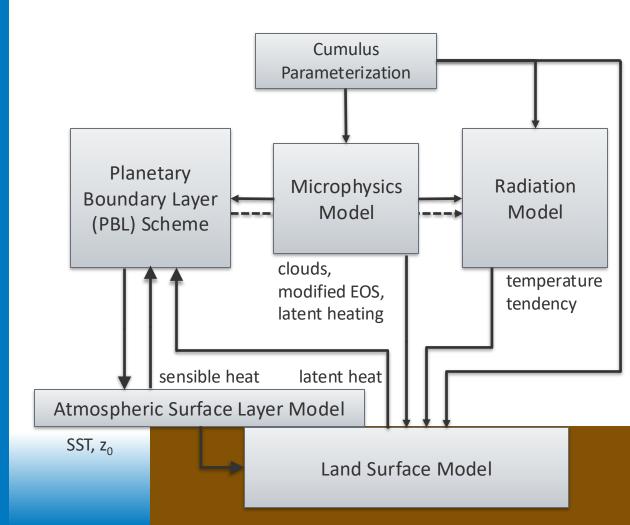
Design Choices

- 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)
 - $2^{nd} 6^{th}$ 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



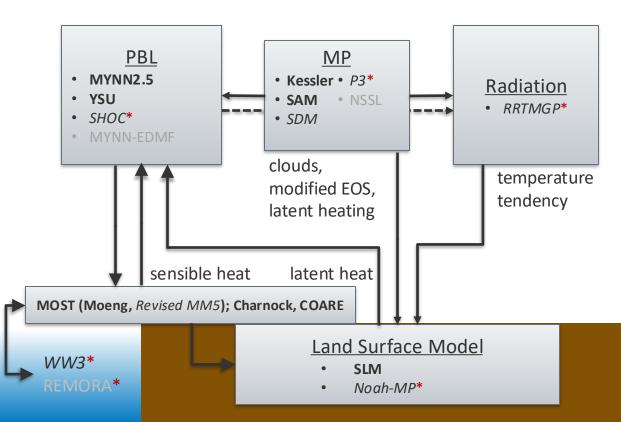
Atmospheric Physics Modeling in ERF

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

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

Turbulence closure

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



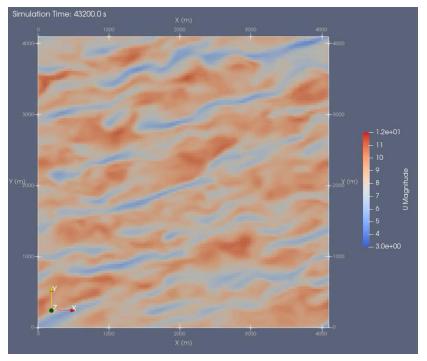
We will support a variety of real-data simulation workflows

	Large-scale data (reanalysis, HRRR,)	Intermediate Processing	Mesoscale/microscale simulation
WRF	Manual download	WPS tools + real.exe	wrf.exe
WRF \rightarrow ERF	Manual download	WPS tools + real.exe -or- ndown.exe	erf_abl
WPS \rightarrow ERF	Manual download	WPS tools	erf_abl
E3SM \rightarrow ERF	Run E3SM	See below \checkmark	erf_abl
ERF standalone	Python tools : HRRR \rightarrow E (others under development)	erf_abl	

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

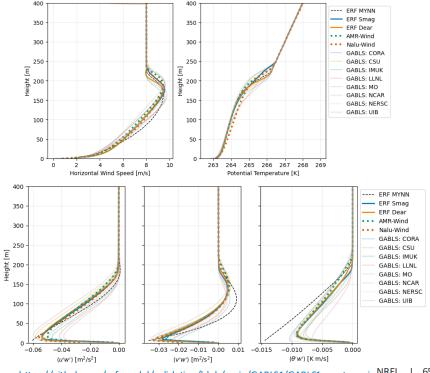
V&V: Dry Atmospheric Boundary Layer

Neutral ABL, WRF-like configuration



Contours of horizontal wind speed at hub height

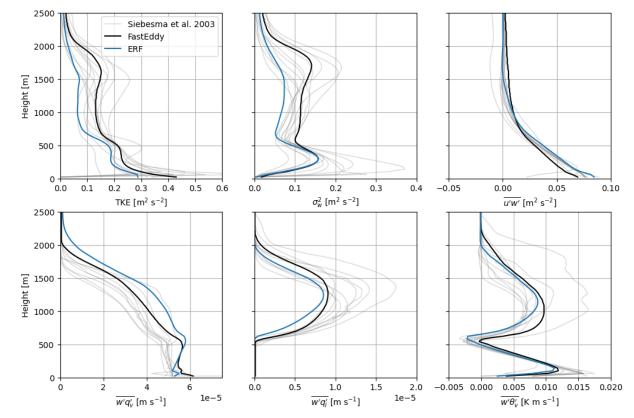
GABLS1 Stable ABL



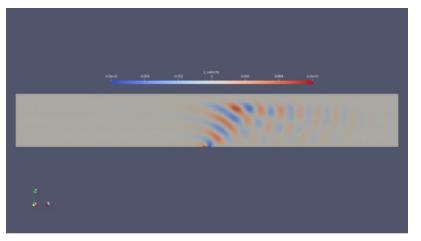
https://github.com/erf-model/validation/blob/main/GABLS1/GABLS1_postproc.ip/https//github.com/erf-model/validation/blob/main/GABLS1/GABLS1_postproc.ip/https//github.com/erf-model/validation/blob/main/GABLS1/GABLS1_postproc.ip/https//github.com/erf-model/validation/blob/main/GABLS1/GABLS1_postproc.ip/https//github.com/erf-model/validation/blob/main/GABLS1/GABLS1_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

V&V: BOMEX Shallow Cumulus Convection

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

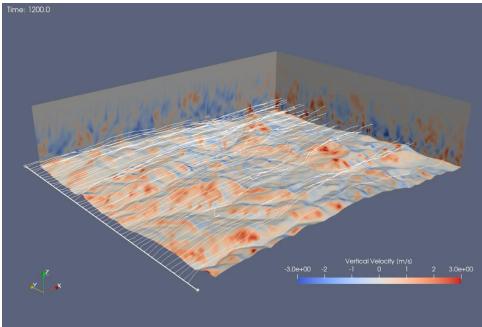


Terrain Flows

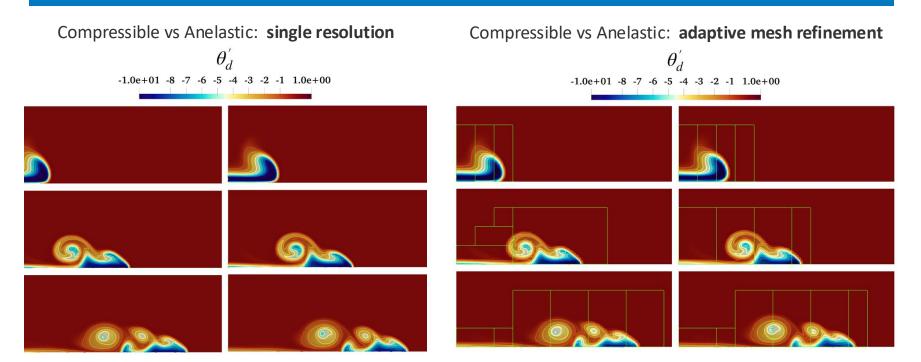


Gravity wave propagation over a Witch of Agnesi hill geometry

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



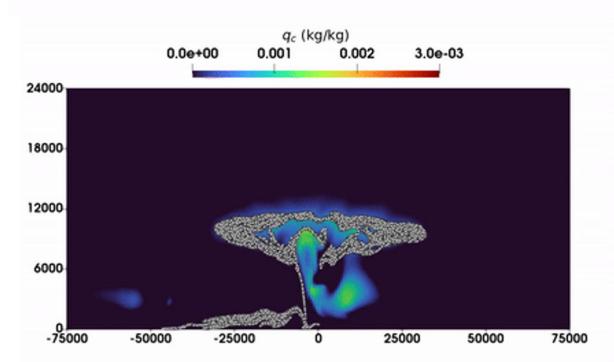
Adaptive Mesh Refinement



<u>Density current simulation</u>: 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.

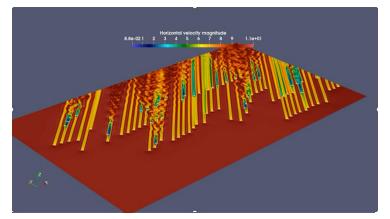
Thunderstorm Outflow



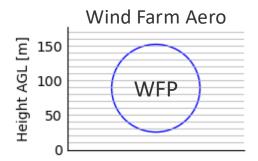
- Squall line simulation, arising from an initial warm, moist bubble
- Particles seeded in initial bubble, advected as passive tracers

Wind Farm & Turbine Modeling in ERF

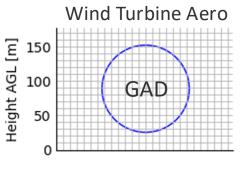
- Wind Farm Parameterization (WFP; Fitch et al. 2012, Volker et al. 2015)
- Generalized Actuator Disk with BEMT (GAD; Mirocha et al. 2014)
- ERF can be online coupled with AMR-Wind (+ OpenFAST)



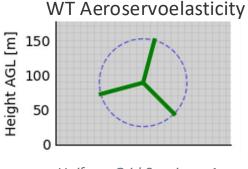
Velocity magnitude contours at hub height for a wind farm (88 turbines) simulated with GAD.



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



Uniform Grid Spacing ~10 m

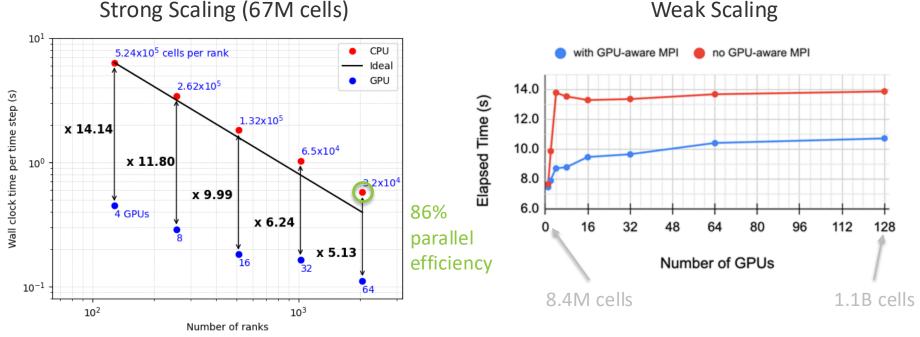


Uniform Grid Spacing ~1 m

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

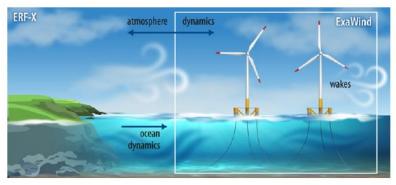


Outlook

- 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 <u>Rafael.Mudafort@nrel.gov</u>
 - Anonymous feedback form to be sent as a follow up
- Software repositories:
 - AMR Wind: <u>https://github.com/exawind/amr-wind</u>
 - Nalu Wind: <u>https://github.com/exawind/nalu-wind</u>
 - OpenTurbine: <u>https://github.com/exawind/openturbine</u>
 - ERF: <u>https://github.com/erf-model/ERF</u>
- WETO Stack Site: <u>https://nrel.github.io/WETOStack</u>
 - Workshop recordings (including this one!)