SAND2022-14955C

Meet the ExaWind/AMR-Wind Team

NAWEA/WindTech 2022
University of Delaware, Newark DE

September 23, 2022

Presentation by:
Georgios Deskos (NREL)
Tony Martinez (NREL)
Lawrence Cheung (SNL)

Sandia National Laboratories is a multimission laboratory managed and operated by National Technology & Engineering Solutions of Sandia, LLC, a wholly owned subsidiary of Honeywell International Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525.

ExaWind/AMR-Wind Team

ExaWind: An open-source **multi-fidelity** modeling & simulation software stack designed to run on **laptops** and **next-gen supercomputers**

AMR-Wind
- https://github.com/exawind/amr-wind
- Incompressible-flow CFD
- Structured-grid finite-volume discretization
- Built on AMReX, a framework for block-structured adaptive mesh refinement

Nalu-Wind
- https://github.com/exawind/nalu-wind
- Incompressible-flow CFD
- Unstructured-grid finite-volume discretization
- Built on Trilinos
- Leverages Trilinos and *hypre* linear-system solvers

TIOGA
- https://github.com/jsitaraman/tioga
- Library for overset-grid assembly

OpenFAST
- https://github.com/openfast
- Whole-turbine simulation code

Software introduced in NAWEA 2019 paper
https://iopscience.iop.org/article/10.1088/1742-6596/1452/1/012071/pdf

ExaWind/AMR-Wind Team

Michael A Sprague(PI)

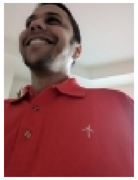
Paul Crozier (co-PI)

Ann Almgren (co-PI)

The development & HFM team

Michael Brazell (NREL)

Georgios Deskos (NREL)



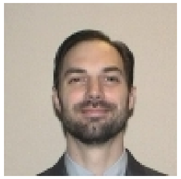
Tony Martinez (NREL)

Matt Churchfield (NREL)

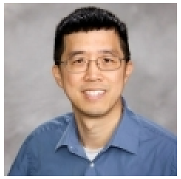
Michael Kuhn (NREL)

Jon Rood (NREL)

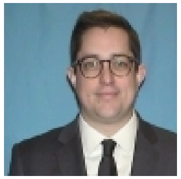
Marc Henry de Frahan (NREL)



Phil Sakievich (SNL)



Lawrence Cheung (SNL)



Nate DeVelder (SNL)

Alan Hsieh (SNL)

Neil Matula (SNL)

Shashank Yellapantula (NREL)

Shreyas Bidadi (NREL)

Ganesh Vijayakumar (NREL)

Ashesh Sharma (NREL)

Brooke Stanislawski (NREL)

Shreyas Ananthan

The basics of AMR-Wind

Georgios Deskos

Basics of AMR-Wind: The physics model

- AMR-Wind solves the unsteady, incompressible Navier-Stokes equations describing the lower atmospheric boundary layer

$$\frac{\partial u_j}{\partial x_j} = 0$$

$$\frac{\partial}{\partial t}(\rho\, u_i) + \frac{\partial}{\partial x_j}(\rho\, u_i u_j) = -\underbrace{\frac{\partial p'}{\partial x_j}\delta_{ij}}_{\textbf{I}} + \underbrace{\frac{\partial \tau_{ij}}{\partial x_j}}_{\textbf{II}} - \underbrace{2\rho\, \epsilon_{ijk}\, \Omega_j u_k}_{\textbf{III}} + \underbrace{(\rho - \rho_\circ)\, g_i}_{\textbf{IV}} + \underbrace{\rho S_i}_{\textbf{V}}$$

- Term I represents pressure gradient forces as a deviation from hydrostatic and horizontal mean gradients,
- Term II represents the contribution from viscous and sub-filter scale stresses,
- Term III represents the contribution from Coriolis forces due to earth's rotation,
- Term IV represents the effects of buoyancy
- Term V represents source terms necessary to drive the flow to a desired horizontal mean velocity

Basics of AMR-Wind: The numerical discretization

- AMR-Wind's spatial discretization is a combination of the finite-volume method and the finite-element method
- Time discretization is achieved via a fractional-step method,

Predictor

$$\frac{\rho^{n+1}-\rho^{n}}{\Delta t}+\left[\frac{\partial \rho u_j}{\partial x_j}\right]^{n+1/2}=0$$

$$\frac{c_k^{n+1}-c_k^{n}}{\Delta t}+\left[\frac{\partial c_k u_j}{\partial x_j}\right]^{n+1/2}=\frac{1}{\rho^{n+1/2}}\left((1-\alpha)\frac{\partial q_{kj}^{n}}{\partial x_j}+\alpha\frac{\partial q_{kj}^{n+1}}{\partial x_j}\right)+G_k^{n+1/2}$$

$$\frac{u_i^{*}-u_i^{n}}{\Delta t}+\left[\frac{\partial u_i u_j}{\partial x_j}\right]^{n+1/2}=\frac{1}{\rho^{n+1/2}}\left(-\frac{\partial p}{\partial x_i}^{n-1/2}+(1-\alpha)\frac{\partial \tau_{ij}^{n}}{\partial x_j}+\alpha\frac{\partial \tau_{ij}^{*}}{\partial x_j}\right)+F_i^{n+1/2}$$

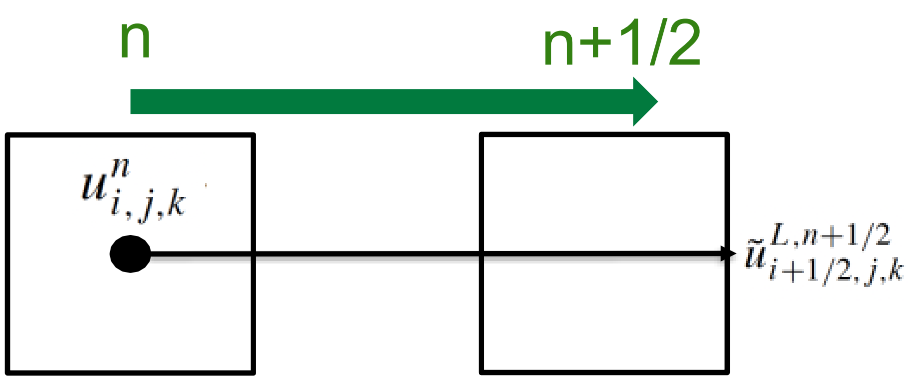
Corrector (pressure projection)

$$u^{n+1}=P(u_i^{*}) \qquad P(u_i^{*})=u_i^{*}+\frac{\Delta t}{\rho^{n+1/2}}\left(\frac{\partial p}{\partial x_i}^{n-1/2}-\frac{\partial \phi}{\partial x_i}\right).$$

Basics of AMR-Wind: The numerical discretization

- The advection term is formed by extrapolating in space **and time** using a Godunov method

$$\tilde{u}^{L,n+1/2}_{i+1/2,j,k} \approx u^n_{i,j,k} + \frac{\Delta x}{2} u_x + \frac{\Delta t}{2} u_t$$



$$\tilde{u}^{L,n+1/2}_{i+1/2,j,k} = u^n_{i,j,k} + \left(\frac{\Delta x}{2} - u^n_{i,j,k}\frac{\Delta t}{2}\right)\left(u^{n,lim}_x\right)_{i,j,k} + \frac{\Delta t}{2}\Bigg(-(\widehat{v u}_y)_{i,j,k} - (\widehat{w u}_z)_{i,j,k} + \frac{1}{\rho^n_{i,j,k}}\left(-(G_x p)^{n-1/2}_{i,j,k} + \mu \Delta^h u^n_{i,j,k} + H^n_{U,x,i,j,k}\right)\Bigg),$$

Basics of AMR-Wind: The numerical discretization

- This intermediate velocity is NOT divergence-free!
- To make it divergence-free we apply a MAC projection

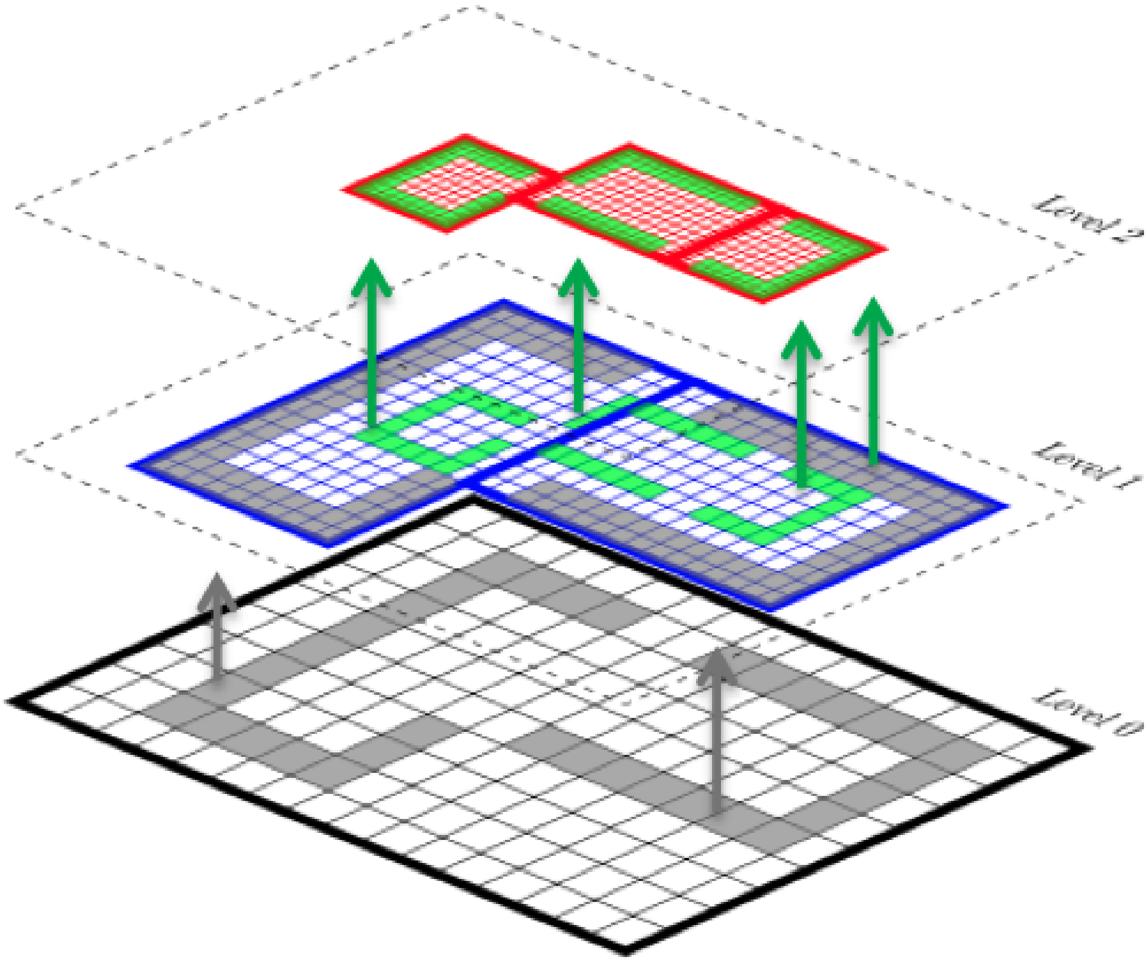
$$P^{MAC}(u_i^f) = u_i^f - \frac{1}{\rho^n}\left(\frac{\partial \psi}{\partial x_i}\right)$$

$$\frac{\partial}{\partial x_j}\left(\frac{1}{\rho^n}\frac{\partial \psi}{\partial x_j}\right) = \frac{\partial u_j^f}{\partial x_j}.$$

Subject to boundary conditions

- We can now compute the advection term by using amongst a range of schemes including PPM, PPM no lim and WENO-z!
- Finally, a solution at the cell-center is obtained after we apply a nodal projection aka pressure projection

Basics of AMR-Wind: The numerical discretization

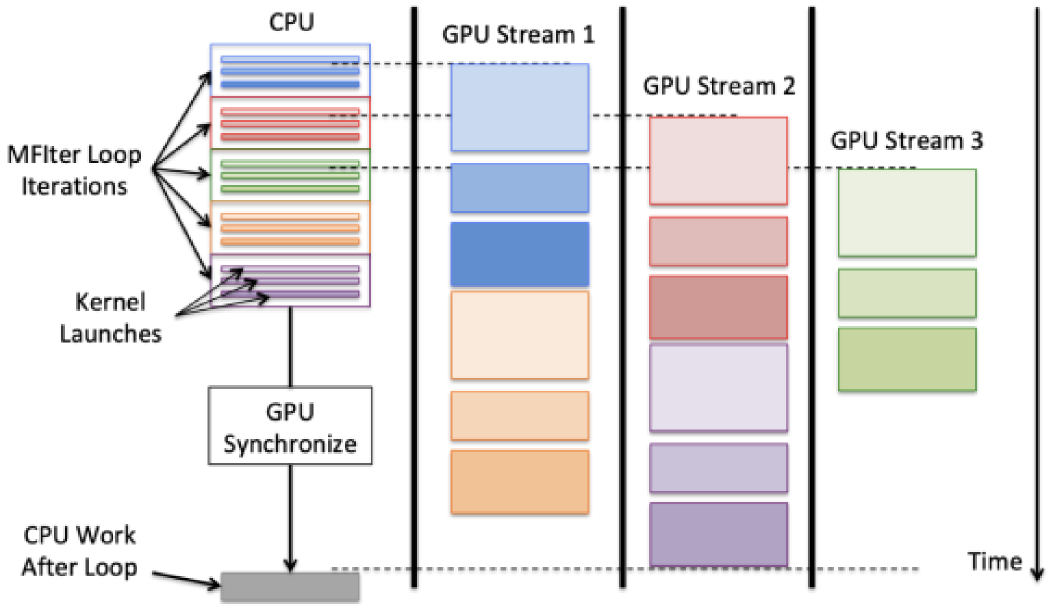


Level-based AMR organizes the grids by levels

Basics of AMR-Wind: The implementation

```
#pragma omp parallel if (amrex::Gpu::notInLaunchRegion())
for (amrex::MFIter mfi(mf,TilingIfNotGPU()); mfi.isValid(); ++mfi) {
    const amrex::Box& bx = mfi.tilebox();
    amrex::Array4<amrex::Real> const& fab = mf.array(mfi);
    amrex::ParallelFor(bx, ncomp,
    [=] AMREX_GPU_DEVICE (int i, int j, int k, int n) {
        fab(i,j,k,n) += 1.0;
    });
}
```

Figure 2: Example AMReX kernel using AMReX's C++ framework.

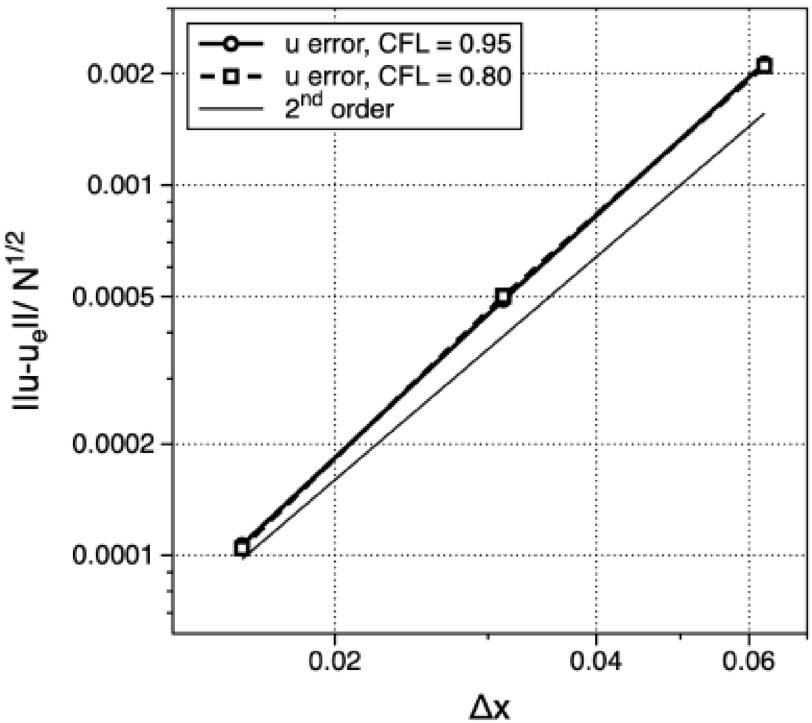


To obtain concurrent kernel execution on a device, "streams" are employed. Streams allow concurrency in execution for multiple kernels and guarantee that the order of execution for kernels within a stream are preserved.

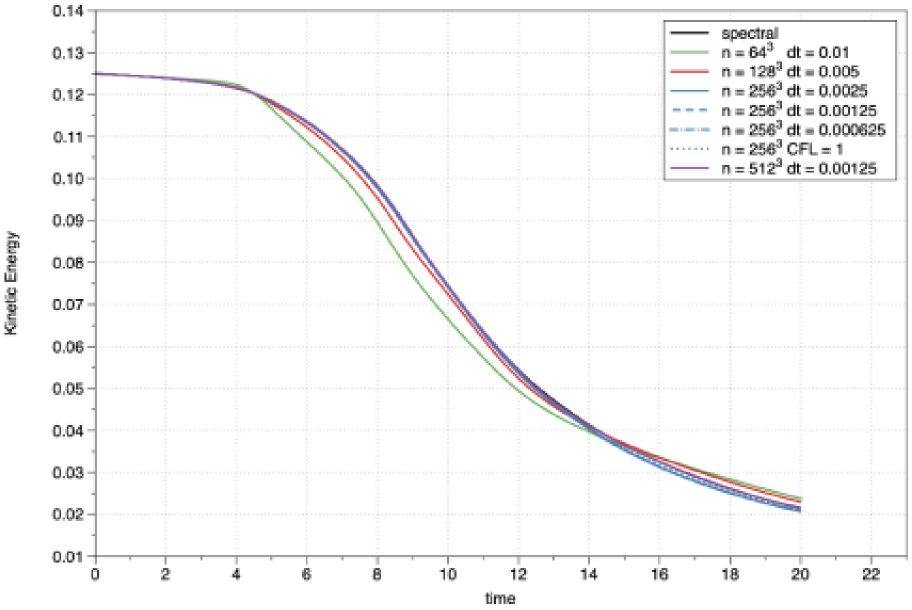
Basics of AMR-Wind: V&V

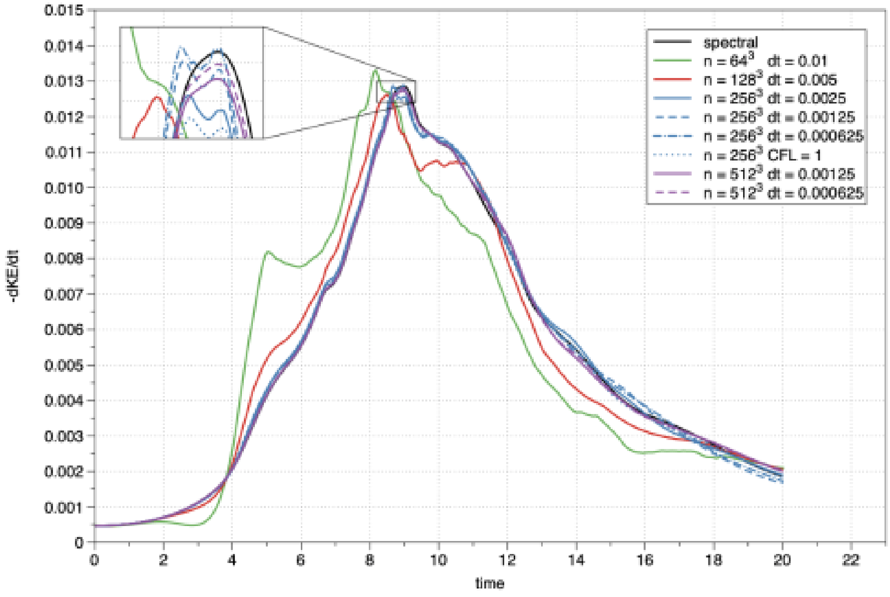
The two-dimensional Convecting Taylor Vortex problem is a useful case for verifying the code

$$u(x,y,t) = u_0 - \cos(\pi(x - u_0 t))\sin(\pi(y - v_0 t))e^{-2\omega t}$$
$$v(x,y,t) = v_0 - \sin(\pi(x - u_0 t))\cos(\pi(y - v_0 t))e^{-2\omega t}$$



Basics of AMR-Wind: V&V



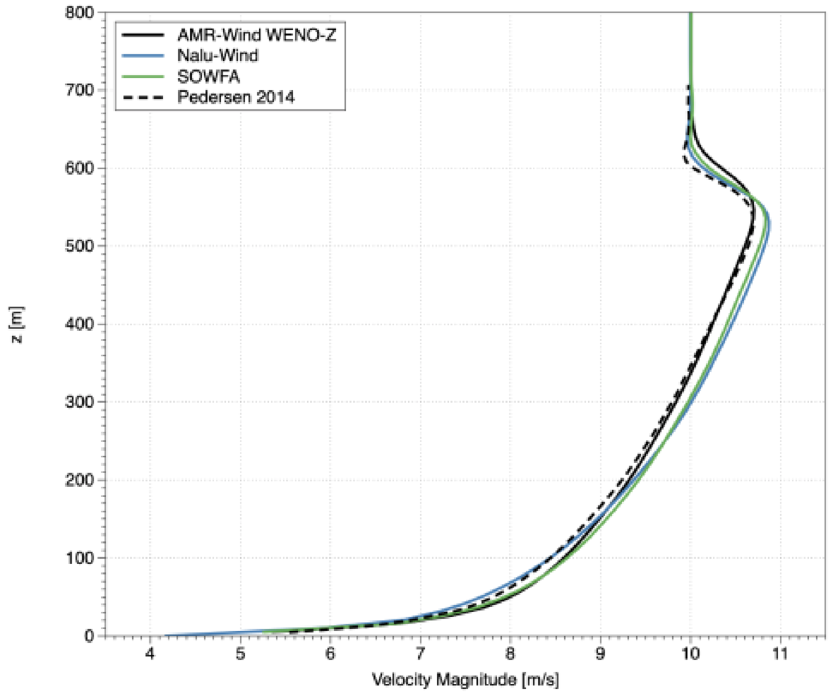


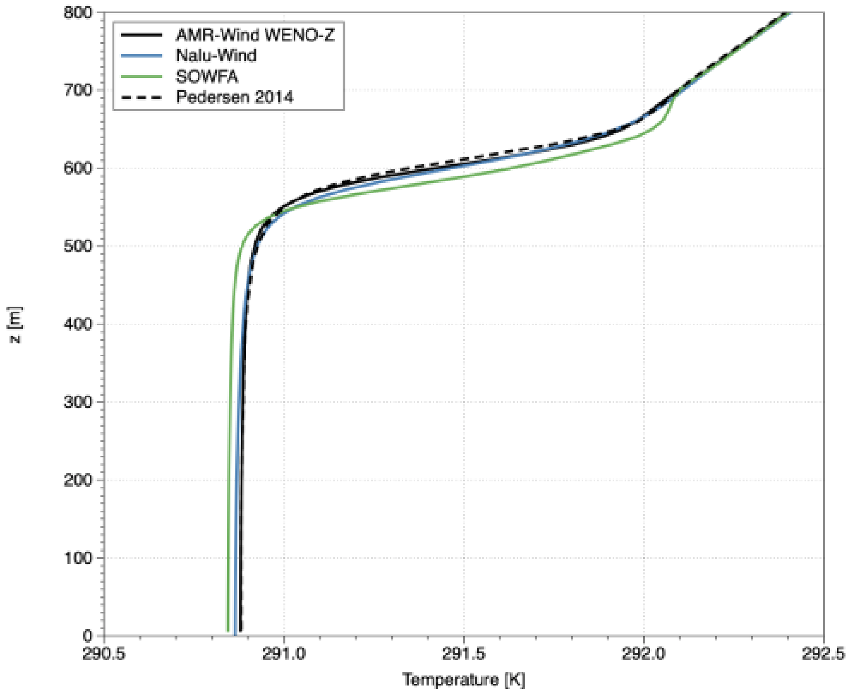
AMR-Wind simulation of Taylor Green Vortex problem at Re = 1600, showing computed kinetic energy, energy dissipation, and enstrophy over time.

Various mesh sizes and time steps are investigated and compared to a spectral DNS simulation.

Basics of AMR-Wind: V&V

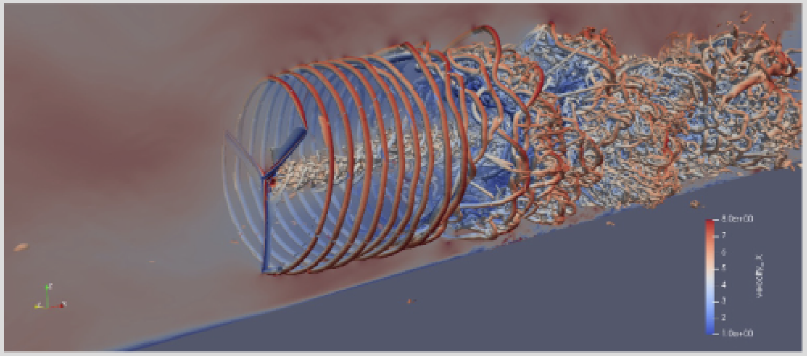
- AMR-Wind simulation of Pedersen N02 neutral ABL, velocity magnitude and potential temperature profiles, averaged from 31-32 hours
- Also included are Nalu-Wind, SOWFA, and Pedersen results

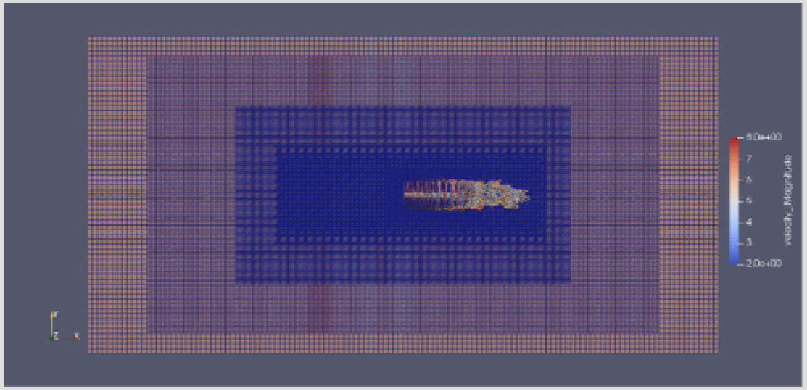




Capability overview

New hybrid solver enables validation-quality blade-resolved turbine simulations (in collaboration with DOE Wind Energy Technologies Office)

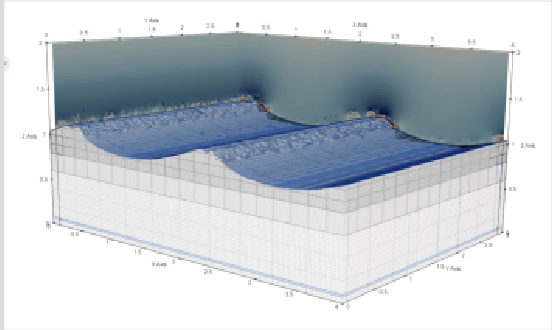




AMR-Wind/Nalu-Wind simulation of the NM-80 DanAero wind turbine in turbulent flow with 122M grid points. Simulations were performed as part of the IEA Task 29 validation campaign. Simulation performed on NREL's Eagle computer.

This simulation demonstrates:
- Coupling AMR-Wind and Nalu-Wind
- Coupling hybrid-RANS/LES models
- Moving meshes
- Wake generation and evolution in turbulent inflow

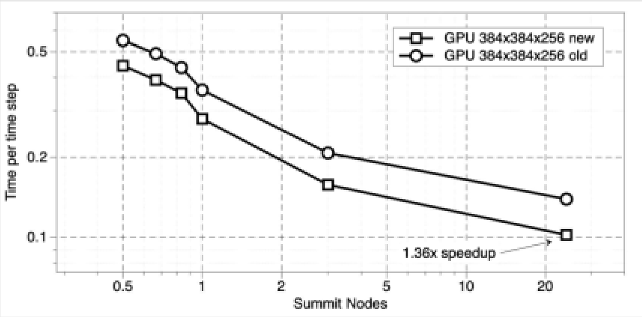
ExaWind Is Going Offshore



Multi-phase flow simulation of waves in AMR-Wind (Figure: Georgios Deskos)

Through funding for the DOE EERE Wind Energy Technologies Office, the team is equipping ExaWind for multi-phase fluid dynamics which is the first step to simulating floating offshore wind turbines.

AMR-Wind Speed Up (on Summit GPUs)



Strong scaling results on Summit GPUs for an atmospheric boundary layer simulation; "old" is from AMR-Wind in April 2021, whereas "new" is AMR-Wind in April 2022.

Through, in part, a collaboration with the ECP CEED and AMReX teams, the ExaWind team has significantly reduced the time per timestep in AMR-Wind by using the switching to the BiCG solver for momentum, solving segregated momentum systems, and using the latest version of AMReX.

AMR-Wind Capabilities

Tony Martinez

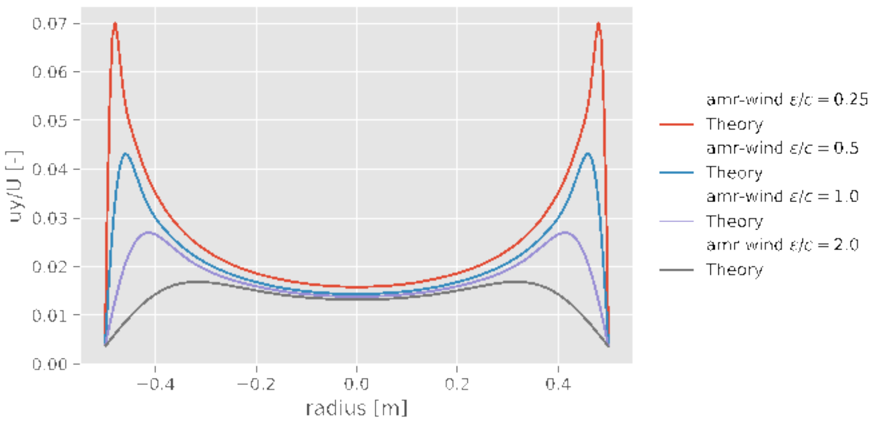
AMR-Wind Capabilities

- ABL structured
- ADM/ALM
- Local refinement
- Only one input file!
- Stability (neutral, unstable, stable)

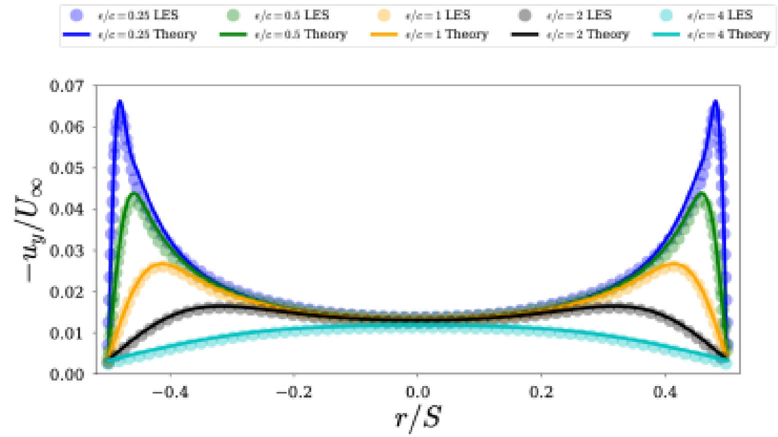
- No blade resolved
- No terrain

Quantities Along the Blade

AMR-Wind Milestone Results



Published Results

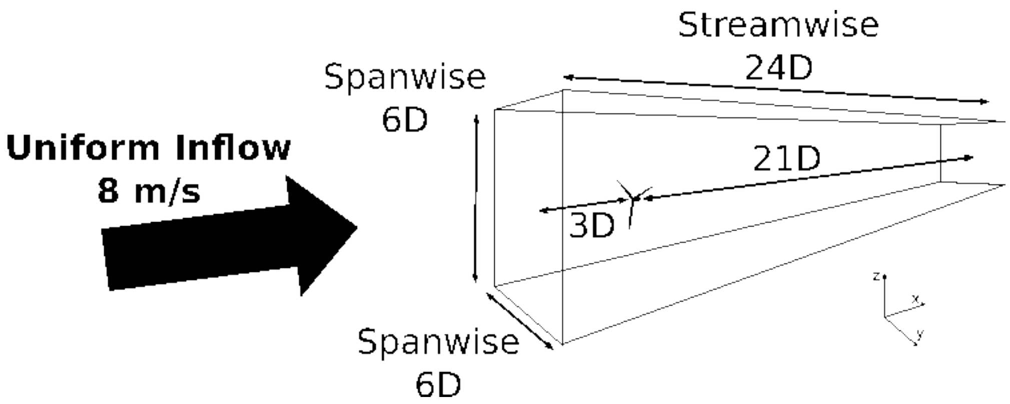


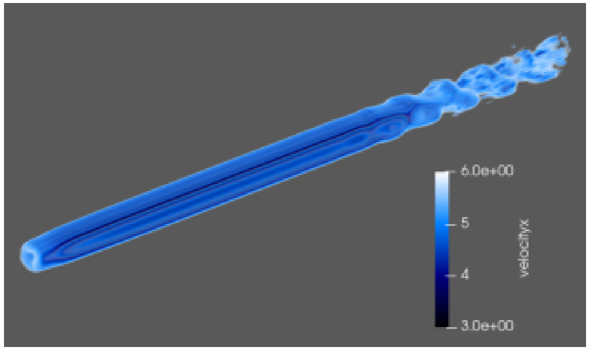
- Excellent agreement between theory/literature and AMR-Wind

Martínez-Tossas, L., & Meneveau, C. (2019). Filtered lifting line theory and application to the actuator line model. Journal of Fluid Mechanics

Verification: Flow Over NREL 5MW ALM

- Benchmark case: Actuator Line Model NREL 5MW in Uniform Inflow
- Compared 4 codes: SOWFA, Johns Hopkins, KU Leuven
- Compare 1-Quantities along the Blade
- Compare 2-Wake Profiles



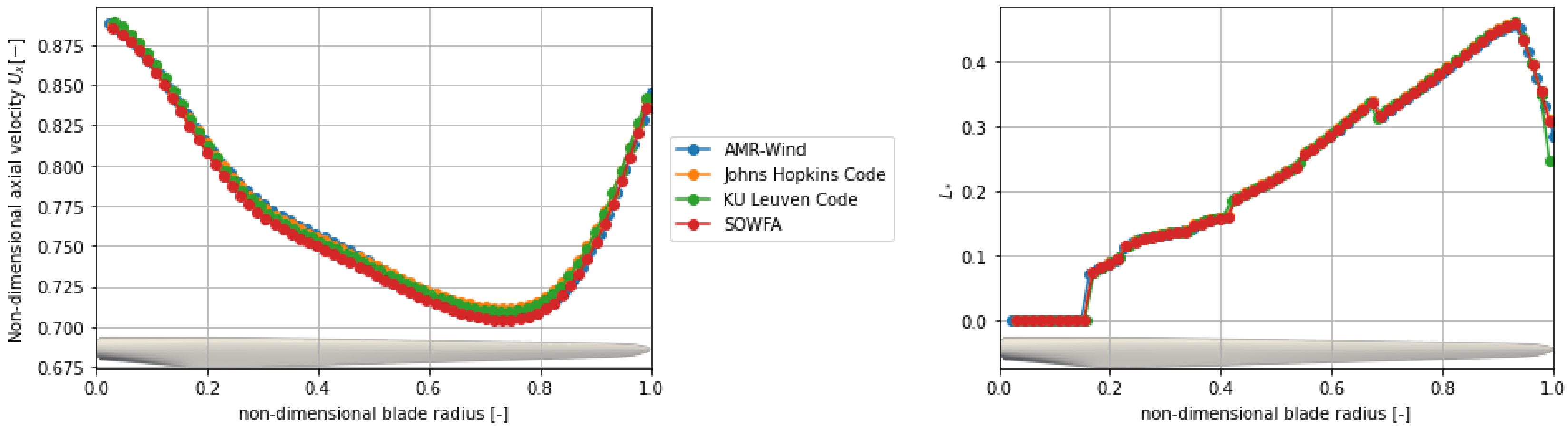


Volume rendering of streamwise velocity from simulation in AMR-Wind

Martínez-Tossas et al. **Comparison of four large-eddy simulation research codes and effects of model coefficient and inflow turbulence in actuator-line-based wind turbine modeling.** JRSE 2018

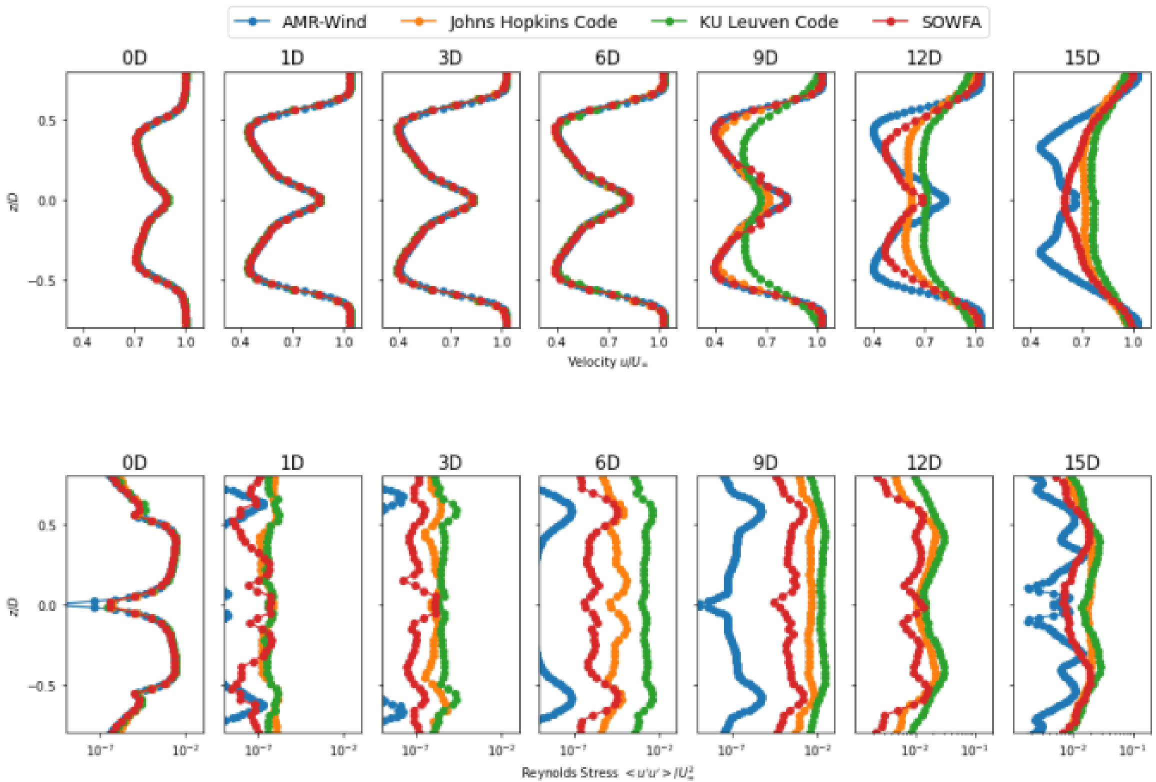
Quantities Along the Blade

- Excellent agreement between AMR-Wind and other codes
- Implementation of ALM verified
- Small differences in implementation would show in these quantities



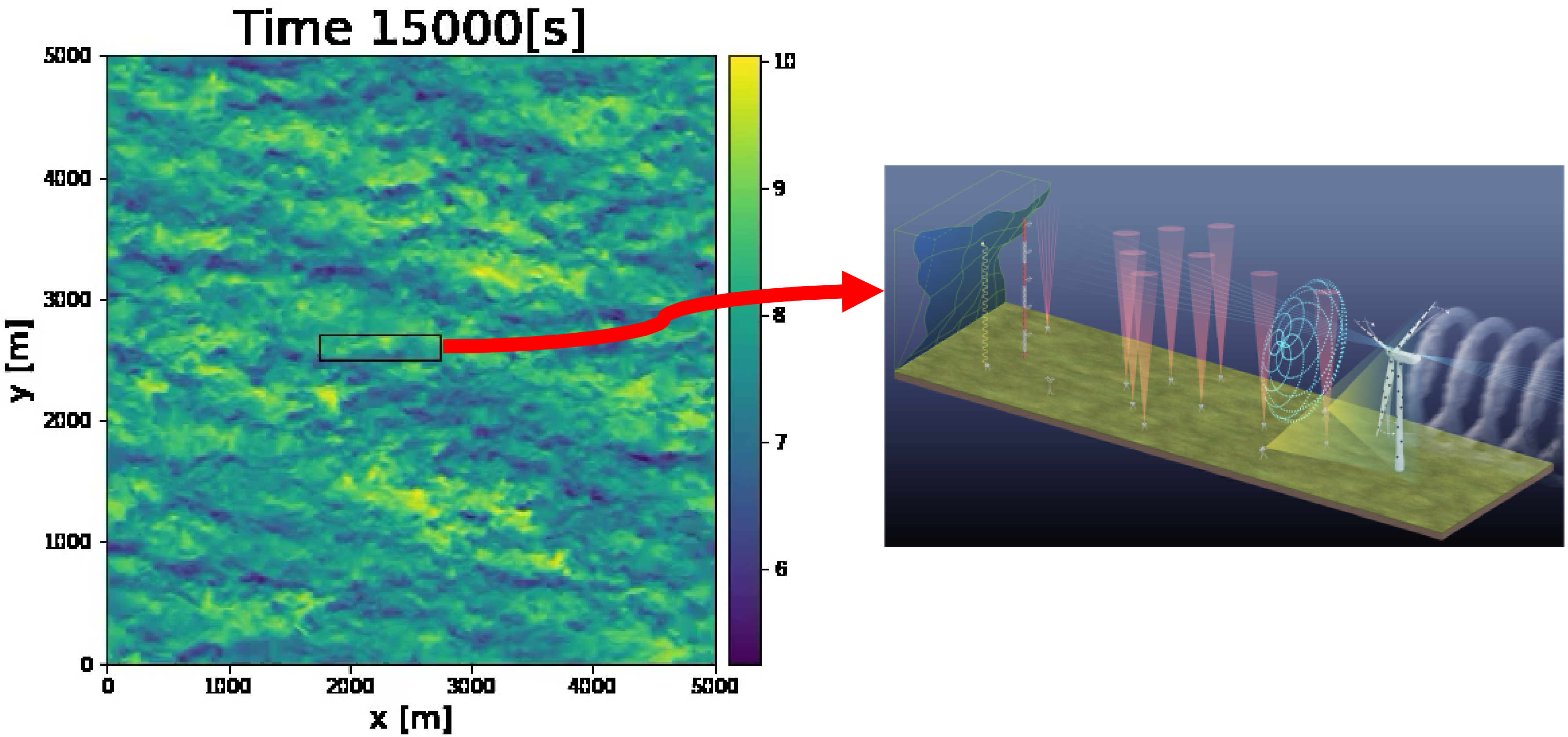
Wake Profiles

- Excellent agreement in mean velocity
- Correct implementation of turbine forces yields consistent mean velocity profiles



- Differences in Reynolds Stresses
- Further work required to understand turbulence in AMR-Wind

RAAW Experiment –Data Assimilation



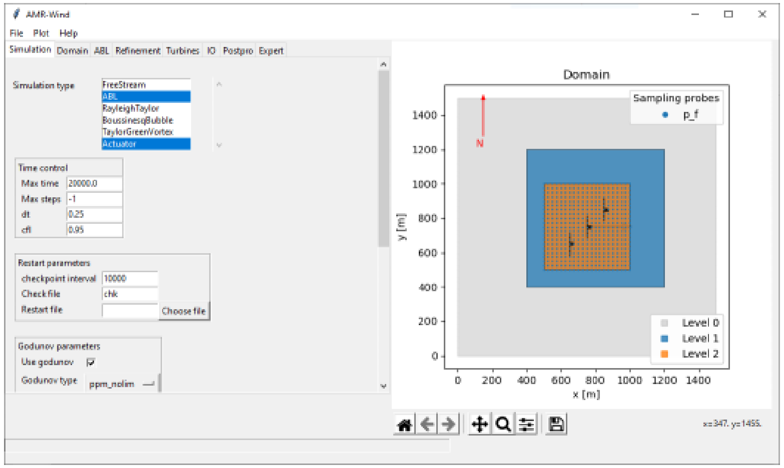
Helpful tools, websites, and documentation

Lawrence Cheung

AMR-Wind-frontend overview

Handy GUI & python interface to help

- Load an AMR-Wind input file and change parameters interactively
- Plot the simulation domain, including refinement zones and sampling probes/planes
- Set up complex wind farm configurations
- Validate AMR-Wind inputs before job submissions
- Submit jobs to a cluster
- Visualize the sampling outputs (probes, lines, and planes)
- Postprocess ABL statistics files
- Use it in Jupyter notebooks or python scripts to automate processing



Download on github at https://github.com/Exawind/amr-wind-frontend

AMR-Wind frontend: Caveats and limitations

The frontend implements a *subset* of all features in AMR-Wind

- Most commonly used ABL/turbine options included
- Many detailed parameters excluded in interface, but they are still compatible with frontend

AMR-Wind frontend is still under active development

- Currently in beta stage – expect bugs & missing documentation
- Some interface and API features may still evolve
- May choose to have tighter integration with AMR-Wind solver in the future

Features that are currently not included, but planned for development:

- RANS ABL and mesh mapping capability

AMR-Wind frontend: Installation & Documentation

AMR-Wind front-end works on Python 2.7/3.X

- Built using TkInter GUI libraries (works on Windows, Linux, Mac)
- Requires NumPy, SciPy, Matplotlib, NetCDF, pyyaml, pandas
- Optional libraries: ruamel, xvfbwrapper, utm
- Can integrate with an AMR-Wind executable & HPC queues directly, or use in stand-alone mode

Quick install:

```
$ git clone --recursive https://github.com/Exawind/amr-wind-frontend.git
```

AMR-Wind frontend can also be customized

- Add organization specific defaults, turbine models, etc.
- Create custom input forms for specific workflows

AMR-Wind frontend: Installation & Documentation

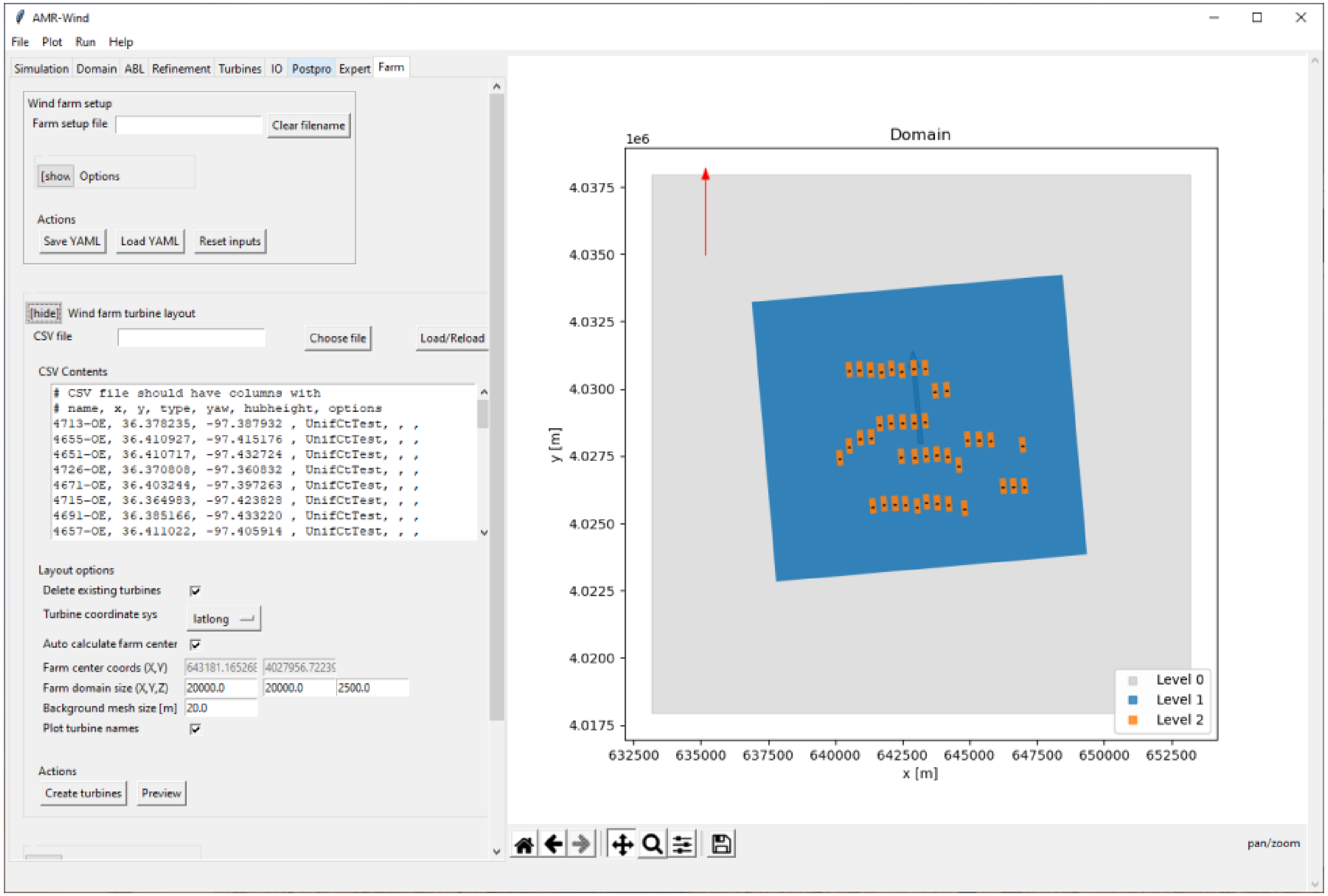
AMR-Wind frontend documentation available at
https://github.com/Exawind/amr-wind-frontend/tree/main/docs

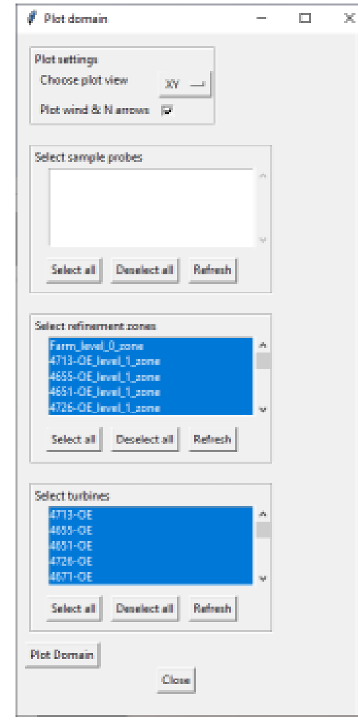
Contains installation, usage, and customization instructions

- Three tutorials available:
  1. An actuator disk model in uniform flow
  2. Running an unstable ABL LES case
  3. Setting up a wind farm configuration

- Two case studies
  1. SWIFT ABL test case
  2. ADM turbine model run

Example of using GUI to set up wind farm





Python/Jupyter notebook interface

Example of using the frontend via python interface

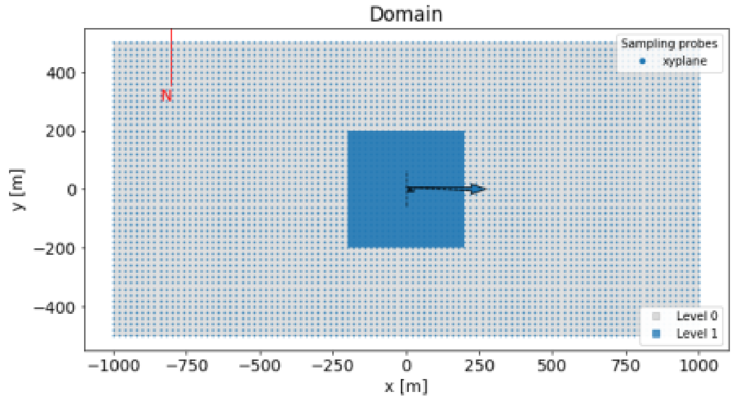
```
# Load the module
import amrwind_frontend as amrwind

# Start the amrwind_frontend app
tutorial1 = amrwind.MyApp.init_nogui()

# Set some parameters
tutorial1.setAMRWindInput('time_control',['const dt'])
tutorial1.setAMRWindInput('time.stop_time',100)
tutorial1.setAMRWindInput('time.fixed_dt', 0.1)
tutorial1.setAMRWindInput('incflo.physics', ['FreeStream', 'Actuator'])

# Do some other stuff here
...

# Plot the figure
tutorial1.plotDomain(ax=ax)
```



→ Any action through the GUI can be scripted in python

Your feedback is requested!

Would greatly appreciate any feedback on AMR-Wind frontend tool

- Report any bugs, missing inputs, etc.
- Any items that need more documentation or additional clarification
- Feature requests

Can report issues on github (https://github.com/Exawind/amr-wind-frontend/issues), or email me (lcheung@sandia.gov) directly

Additional Links and Documentation

Official AMR-Wind user documentation:

https://exawind.github.io/amr-wind/

- Has usual manual, theory manual, and developer documentation

AMReX documentation:

https://amrex-codes.github.io/amrex/

- Information on underlying numerical libraries and additional input parameters

Publications

- Sprague, M. A., Ananthan, S., Vijayakumar, G., & Robinson, M. (2020). ExaWind: A multifidelity modeling and simulation environment for wind energy. In Journal of Physics: Conference Series (Vol. 1452, No. 1, p. 012071). IOP Publishing. https://iopscience.iop.org/article/10.1088/1742-6596/1452/1/012071/pdf
- Lawrence Cheung, Michael J. Brazell, Alan Hsieh, Shreyas Ananthan, Ganesh Vijayakumar and Nathaniel deVelder, "Computation and comparison of the stable Northeastern US marine boundary layer," AIAA Scitech 2021 Forum. AIAA 2021-0454. January 2021, https://doi.org/10.2514/6.2021-0454

Question and Answer Section

Question and answer session

<u>Our questions for the community</u>

- What would users like to see in AMR-Wind?
- How can we help users in their workflow?
- Are people interested in developing for AMR-Wind?
- How do users envision using AMR-Wind?
  - Examples: validation work, Exascale simulations, design studies
  - Is there interest in using AMR-Wind with the full ExaWind stack?