AMR-Wind Workshop

October 28, 2024

Michael Kuhn &
Lawrence Cheung

Outline

- For context: Overview of ExaWind software suite
  - Individual codes and framework
  - Applications and capabilities

- AMR-Wind
  - Features
  - Anatomy of input file
  - Tutorial
    - Installation
    - ABL
    - Actuator Disk

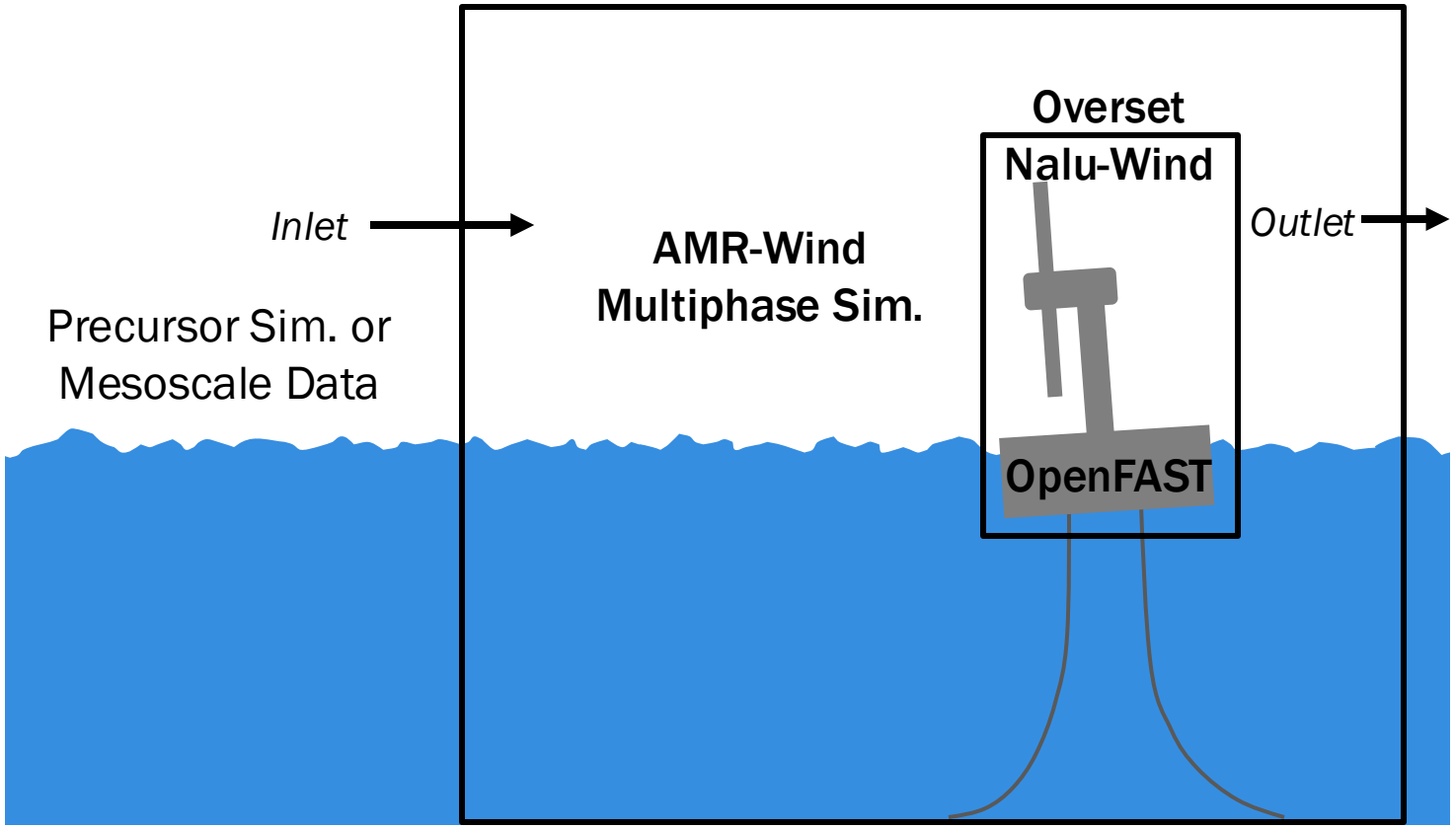
Exawind software suite

- Primary target of development: geometry-resolved floating offshore wind simulations
- In the process, provide a versatile, open-source modeling tool for wind energy researchers with high-fidelity fluid mechanics and multi-fidelity turbine modeling

AMR-Wind:
structured mesh, bulk of computational volume

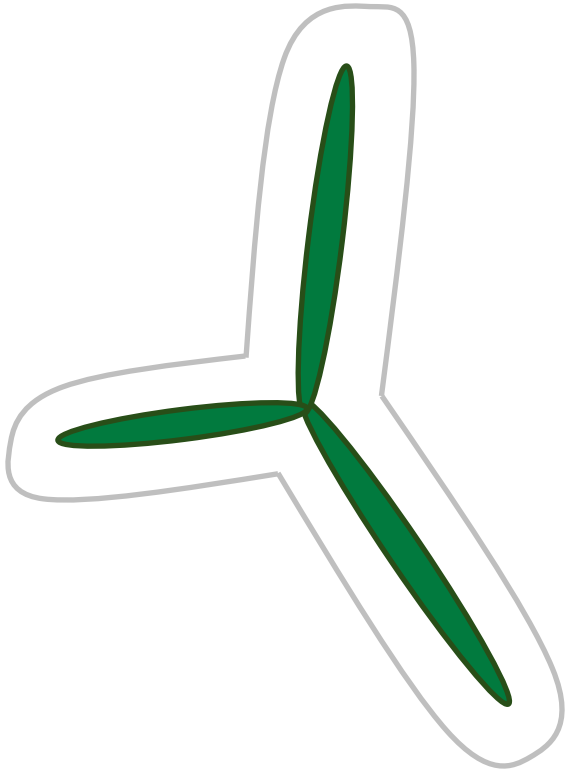
Nalu-Wind:
unstructured mesh, near-body flow dynamics

OpenFAST:
structural modeling

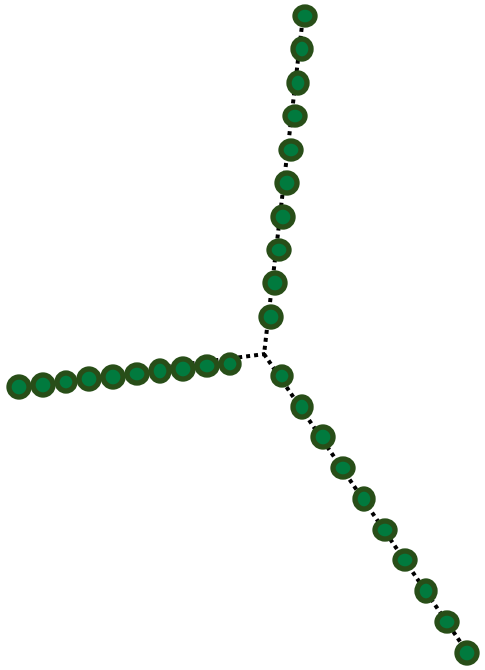


Exawind software suite

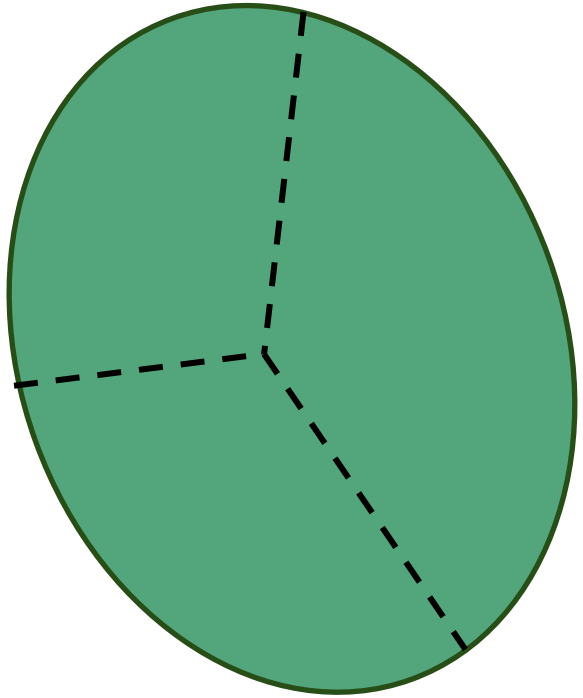
- Versatile, open-source, high-fidelity fluid mechanics and multi-fidelity turbine modeling



Geometry-resolved (Overset)



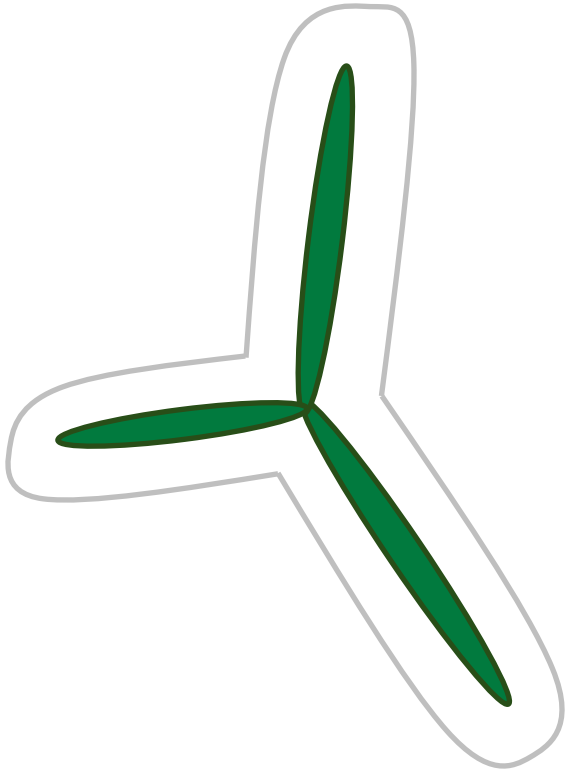
Actuator Line (ALM)



Actuator Disk (ADM)

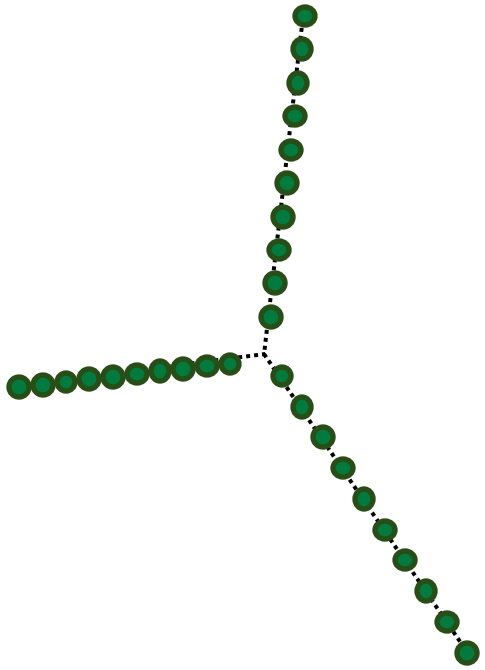
Exawind software suite

- Versatile, open-source, high-fidelity fluid mechanics and multi-fidelity turbine modeling

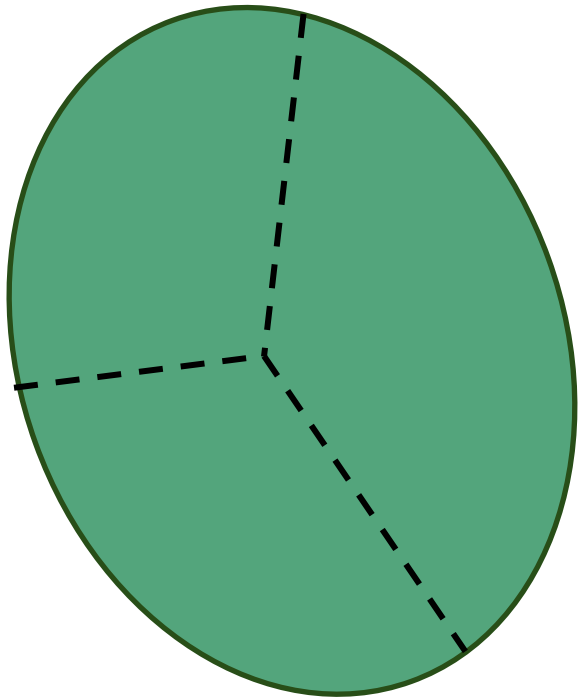


Geometry-resolved
(Overset)

Available in AMR-Wind (+ OpenFAST)



Actuator Line
(ALM)



Actuator Disk
(ADM)

AMR-Wind

- Repository
  - https://github.com/Exawind/amr-wind

- Documentation
  - https://exawind.github.io/amr-wind/
  - Walkthrough
  - User Manual
    - Capabilities and Roadmap
    - Input file reference
  - Theory Manual
  - Developer Documentation

- Regression tests can serve as example input files
  - test/test_files

AMR-Wind documentation intro

- Documentation
  - https://exawind.github.io/amr-wind/
  - User Manual
    - Capabilities and Roadmap
    - Input file reference
    - Post-processing examples

Anatomy of AMR-Wind input file

- File: “case_name.inp” or “case_name.i”

- Syntax (in general):

```
Category          =   Entry1 Entry2 Entry3
Entry1.option1    =   Value1
Entry2.option2    =   Value2
```

- Spacing and indentation do not matter
- Comment with ‘#’

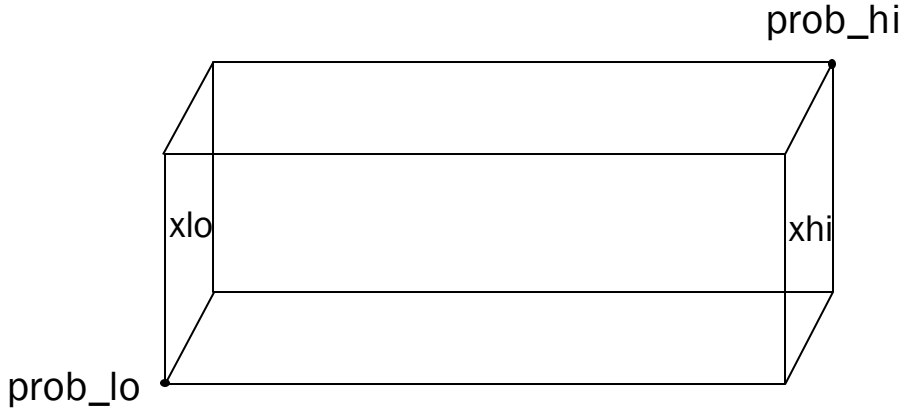
Anatomy of AMR-Wind input file

- Physics
  - Flow initialization and changing environment during simulation
  - Can connect distinct parts of the code, like forcing and boundary conditions
  - This is the “case” that you are running; but also, can specify more than one because they can address different aspects of the simulation
  - E.g., FreeStream, ABL, Actuator, TaylorGreenVortex

- ICNS.source_terms
  - ICNS = InCompressible Navier-Stokes, i.e., the momentum equation
  - E.g., BoussinesqBuoyancy, CoriolisForcing, ABLForcing

Anatomy of AMR-Wind input file

- Domain
  - Due to block-structured nature, all domains are rectangular
  - Define bottom corner position, top corner position, number of cells in each direction for base resolution
  - geometry.prob_lo, geometry.prob_hi, amr.n_cell



- Boundary conditions
  - Specify type for each: xlo, xhi, ylo, etc.
  - Additional values depending on type
  - Should not be specified in periodic direction: geometry.is_periodic

- Time stepping
  - time.fixed_dt, time.initial_dt, time.cfl
  - Set negative value to make inactive

Anatomy of AMR-Wind input file

- Checkpoints: output with time.checkpoint_interval, use with io.restart_file
- Plotfiles: output with time.plot_interval, can specify non-default and derived variables

- Postprocessing
  - Targeted tools to extract specific quantities and output to data files
  - E.g., ABLStats, Sampling, Averaging

- Mesh refinement
  - Add levels by specifying amr.max_level > 0
  - 2 primary methods for static refinement:
    - CartBoxRefinement: define inset rectangular domains
    - GeometryRefinement: define shapes (cylinders, boxes) with parameters

AMR-Wind: Walkthrough

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

- Typical workflow
  - Compiling
  - Precursor (ABL)
  - Turbine (Actuator disk)

AMR-Wind: Walkthrough

- Pause for questions, discussion

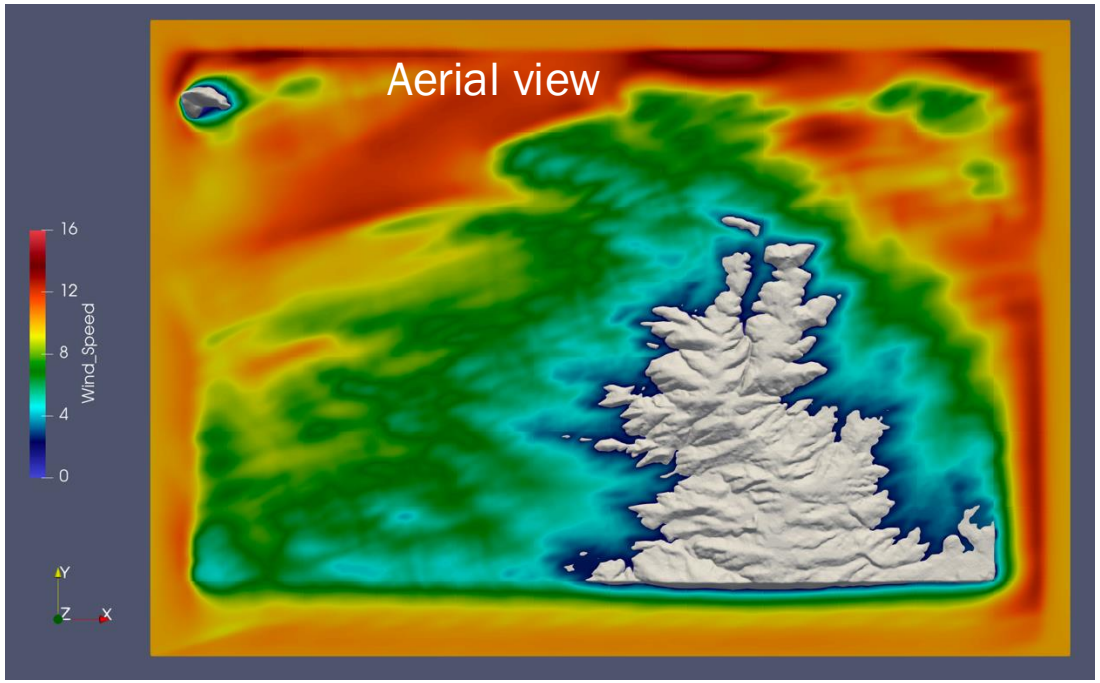
AMR-Wind: Walkthrough

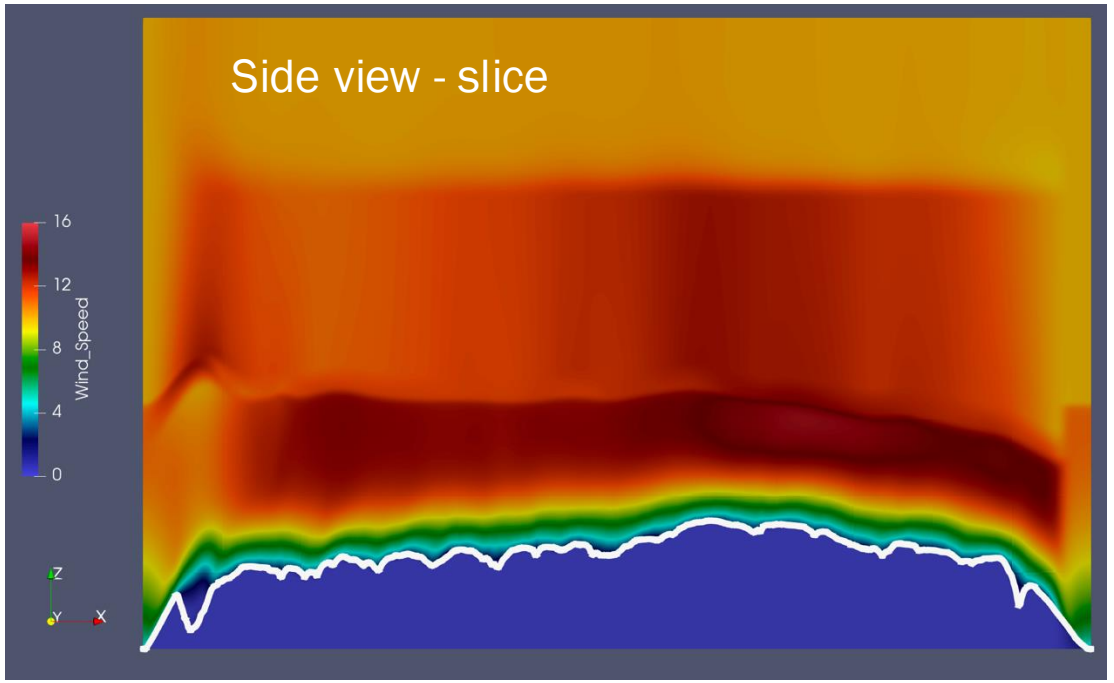
- https://exawind.github.io/amr-wind/walkthrough/

- Other walkthrough components
  - Complex Terrain
  - Actuator-line method (ALM) calibration

AMR-Wind: New features

- Complex terrain capabilities – Immersed Boundary Model
  - Forces velocity toward zero within terrain boundary
  - Turns on wall model in proximity to terrain boundary

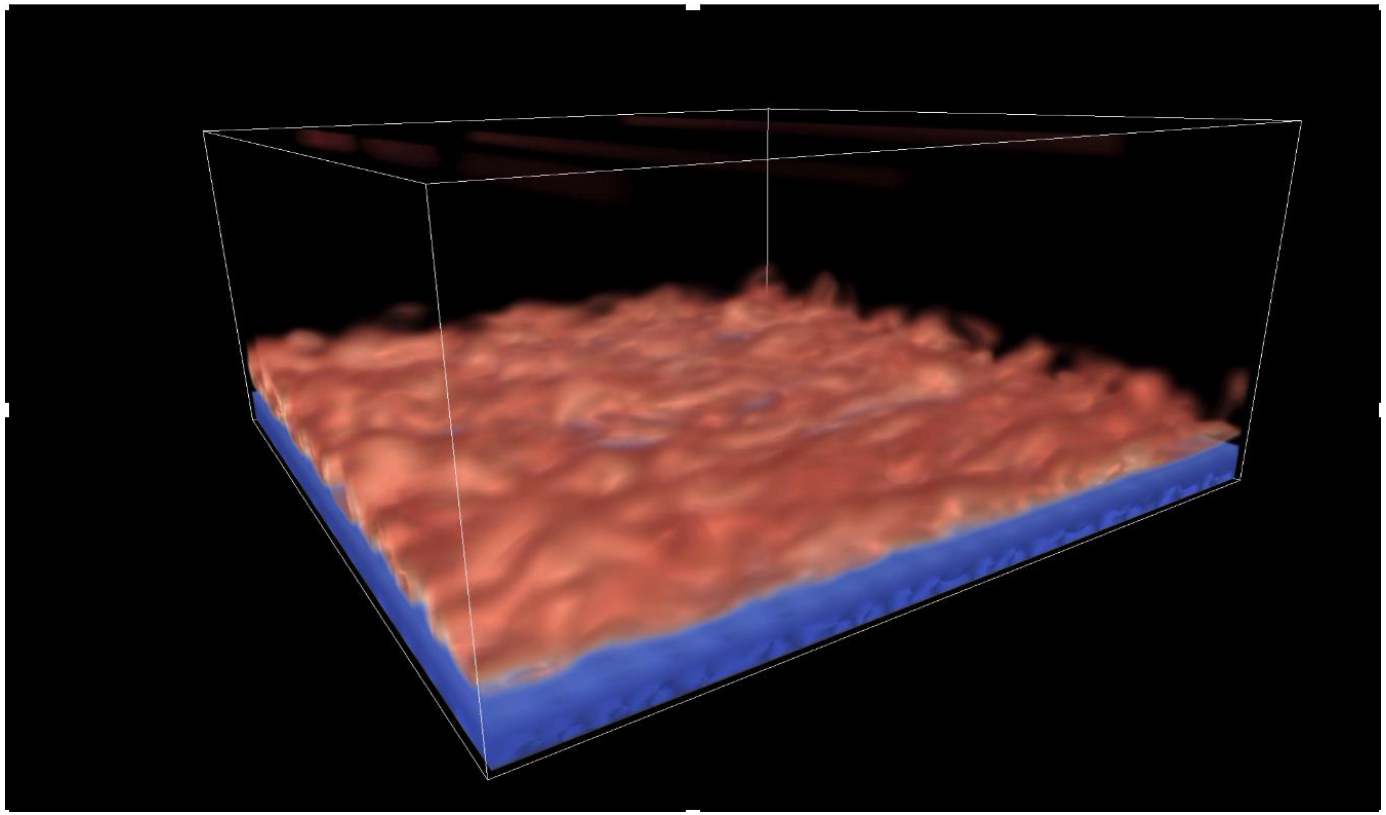




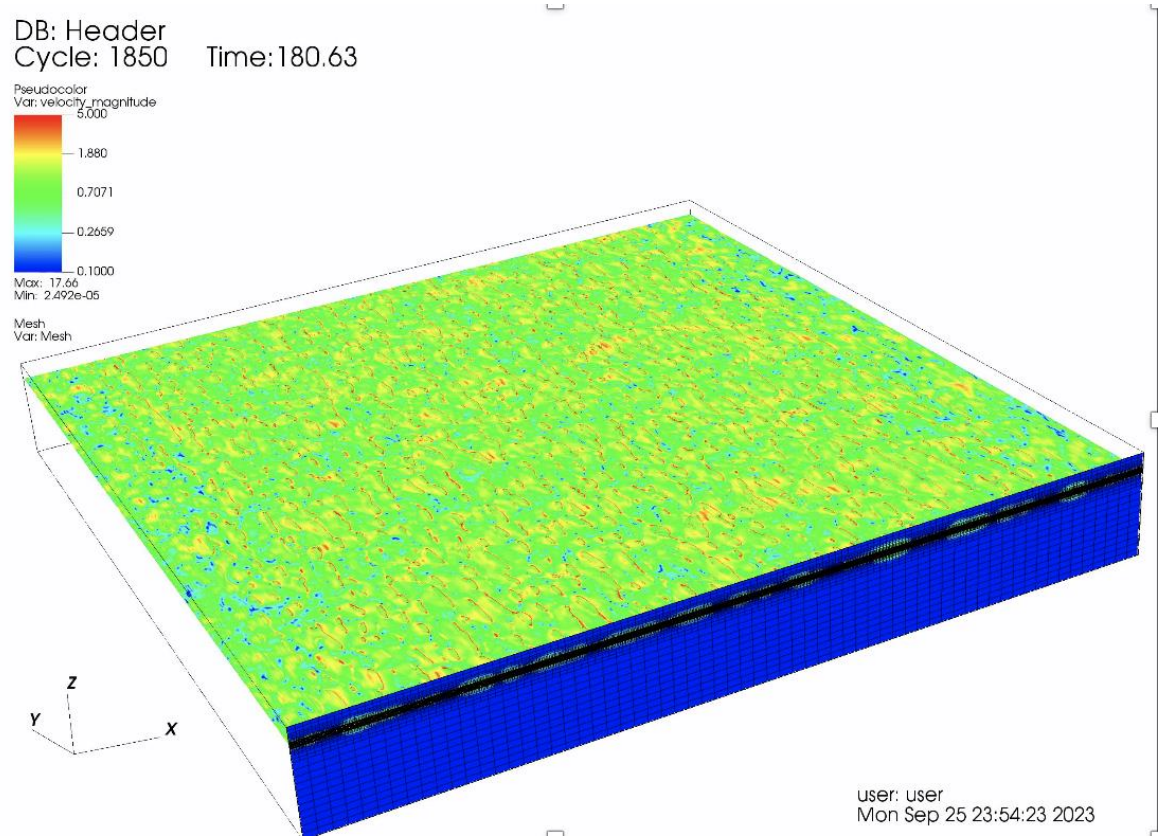
Harish Gopalan

AMR-Wind: New features

- ABL + waves
  - Wave forcing and dissipation zones



Monochromatic linear waves with ABL flow



Irregular waves

AMR-Wind: New features

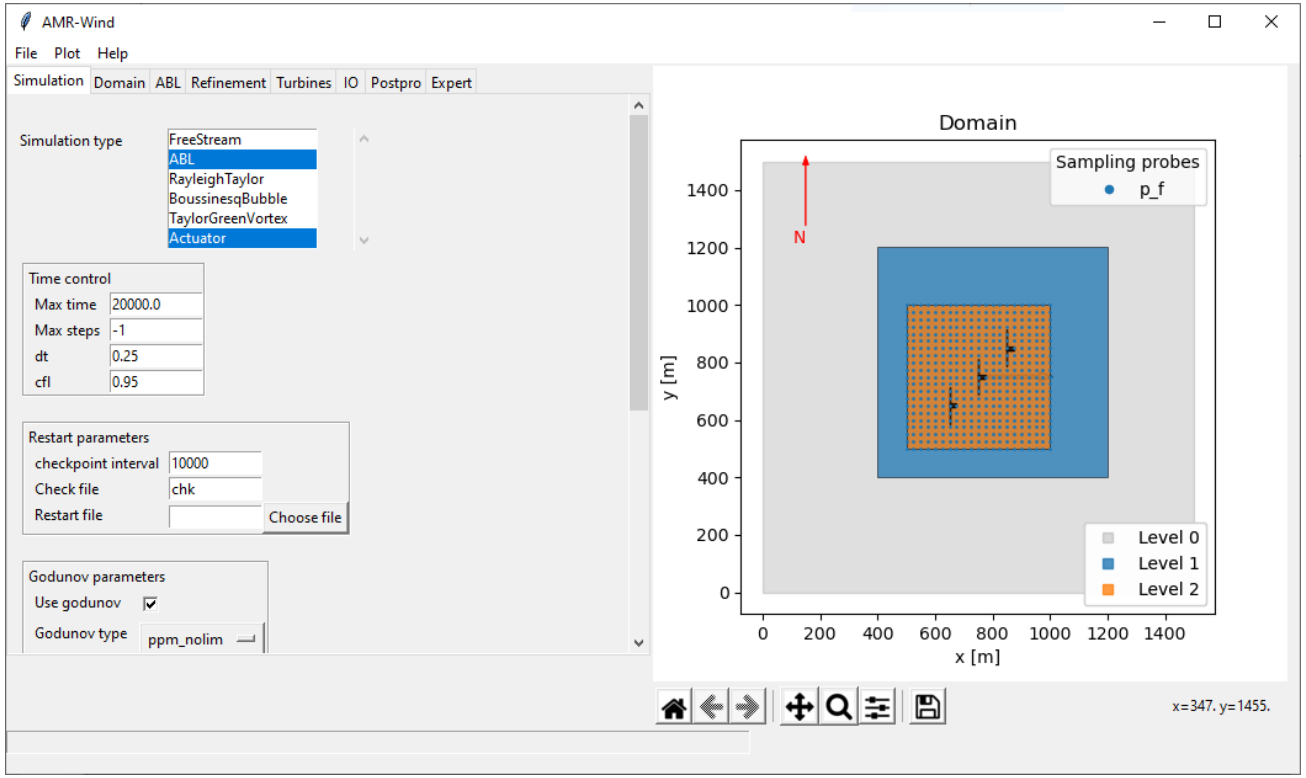
- To be invited to quarterly meetings, join our mailing list!
  - Send a request to amr-wind-maintainers@groups.nrel.gov
  - (this email is also listed in the github README)

  - Summary of software updates
    - Usability
    - New features
    - Bug fixes
  - Provide feedback
  - Discuss with other users

AMR-Wind-frontend overview

Handy GUI & python interface to help setup complex cases

- 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



AMR-Wind frontend: Installation & Documentation

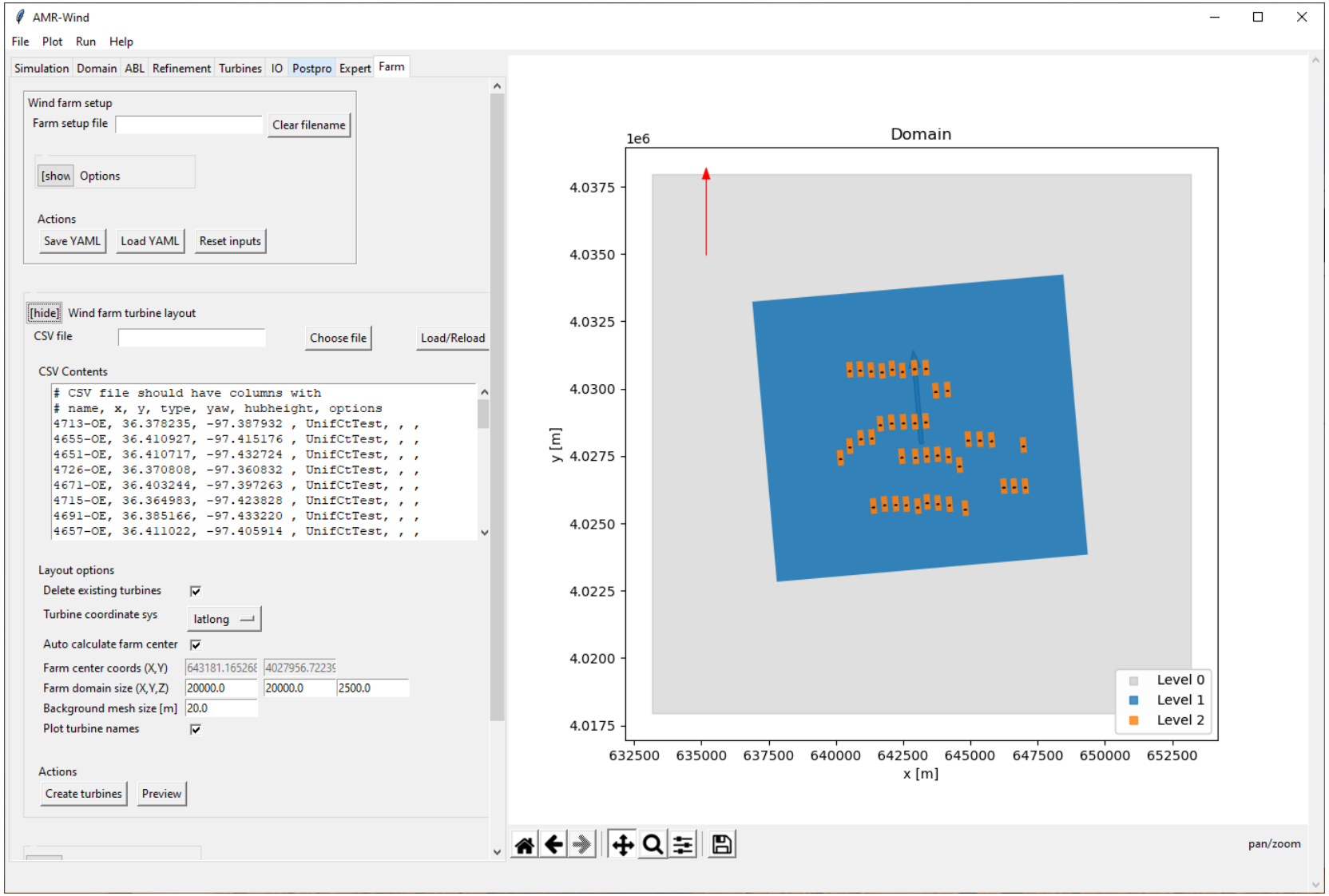
AMR-Wind frontend documentation available at
https://github.com/Exawind/amr-wind-frontend/tree/main/docs

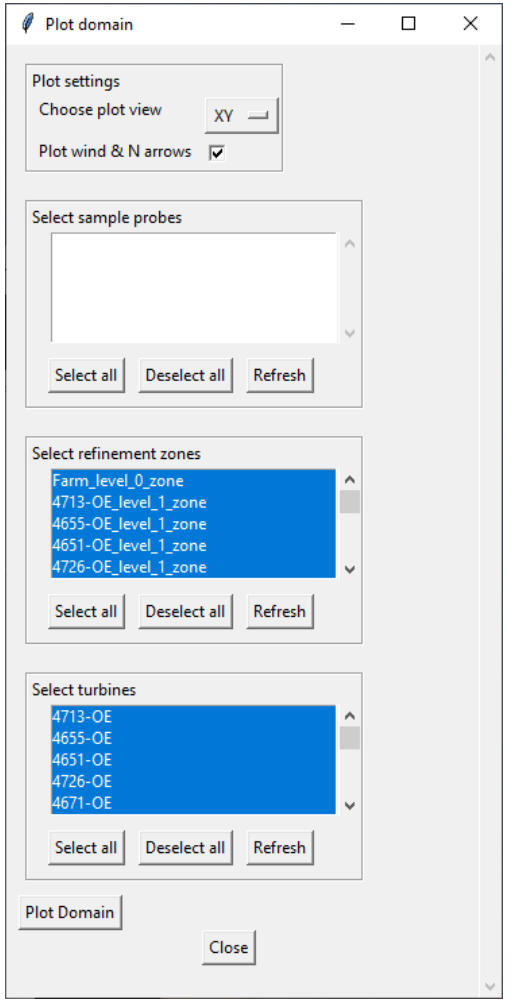
Contains installation, usage, and customization instructions

- 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

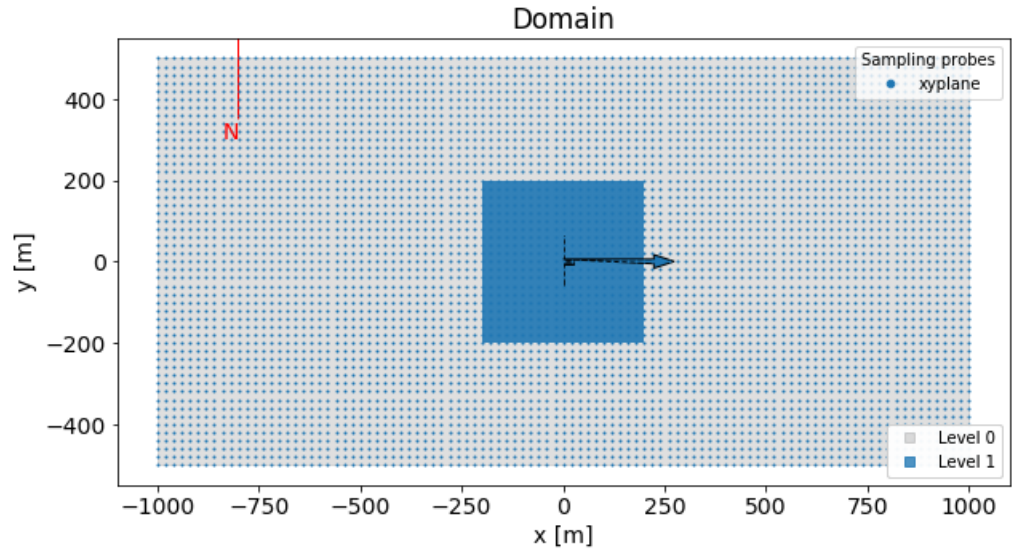
```
# Load the module
import amrwind_frontend as amrwind

# 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)
```



AMR-Wind frontend

Postprocessing engine

- Uniform, centralized way to postprocess AMR-Wind runs
- Acts on output sample planes
- All analyses use common YAML interface
- Run from command line or in Jupyter notebook

Example (Input)

```yaml
task1:
  key1: value1
  key2: value2
  action1:
    param1: val1
  action2:
    param1: val1

task2:
  - name: postprocessing1
    key1: value1
    key2: value2
  - name: postprocessing2
    key1: valueA
    key2: valueB
```

Available analyses

- Instantaneous plots + 2D movie creation
- Time averaging
- Reynolds stress averaging
- 2 point correlations
- Single point spectra in time
- Wake meandering statistics
- SPOD analysis
- OpenFAST postprocessing
- + more to come…

More documentation!

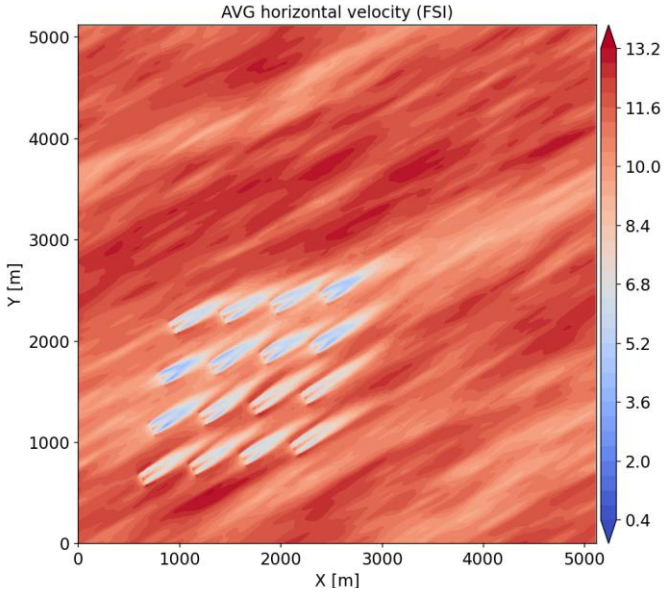
https://github.com/Exawind/amr-wind-frontend/tree/main/postproengine/doc

Contact: lcheung@sandia.gov, gyalla@sandia.gov

AMR-Wind frontend

Example (time averaged planes)

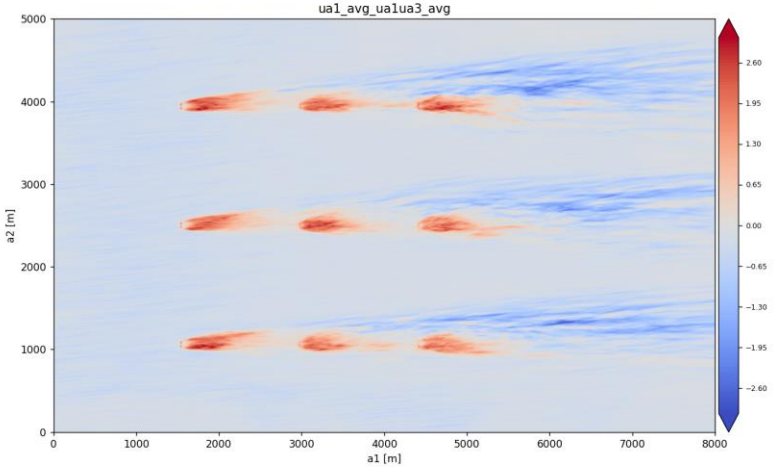
```
avgplanes:
- name: avg_domain_fsi
  ncfile:
  - /lustre/orion/cfd162/proj-shared/lcheung/ALCC_Frontier_WindFarm/windfarm/16_turb_abl_fsi_FY24Q3/run_prod/post_processing/turbdomain_*.nc
  tavg:  [15065, 15125]
  contourplot:
    plotfunc: "lambda db: np.sqrt(db['velocityx_avg']**2 + db['velocityy_avg']**2)"
    title: 'AVG horizontal velocity (FSI)'
    xaxis: x          # Which axis to use on the abscissa
    yaxis: y          # Which axis to use on the ordinate
    iplane: [0]
```



Example (Flow-aligned reynolds stress & turbulent fluxes)

```
reynoldsstress:
  name: Wake YZ plane
  ncfile: /ascldap/users/gyalla/GPFS/Advanced_Controls/AdvancedControlsWakes/post_processing/baseline_YZcoarse_*.nc
  tavg: [28400,28450]
  group: T0_YZdomain
  varnames: ['velocitya1','velocitya2','velocitya3']

  turbulent_fluxes:
```

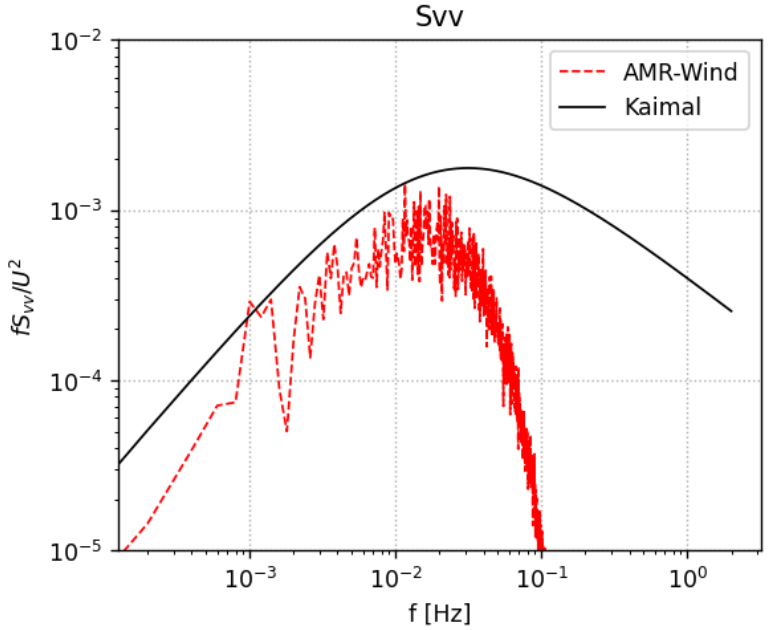


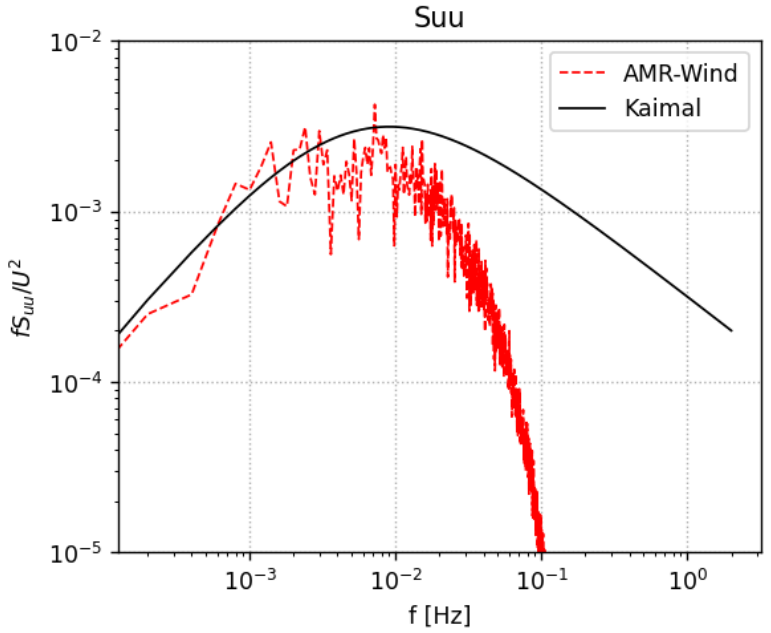
AMR-Wind frontend

Example (spectra)

```
windspectra:
- name: spectra1
  ncfile: /lustre/orion/cfd162/scratch/lcheung/sampling_80000.nc
  group: p_bot
  pointlocationfunction: spectrapoints.getptlist
  csvfile: spectra1.csv
  kaimal:
    ustarsource: ablstatsfile
    ablstatsfile:  /lustre/orion/cfd162/scratch/lcheung/abl_statistics80000.nc
    avgt: [20000, 25000]
    #ustar: 0.289809
    csvfile: kaimal1.csv
    z: 26.5

plotcsv:
  - name: plotSuu
    xlabel: 'f [Hz]'
    ylabel: '$f S_{uu}/U^2$'
    xscale: log
    yscale: log
    title: 'Suu'
    figsize: [5,4]
    legendopts: {'loc':'upper right'}
    postplotfunc: spectrapoints.formatplot
    csvfiles:
    - {'file':'spectra1.csv', 'xcol':'f', 'ycol':'Suu',
      'lineopts':{'color':'r', 'lw':1, 'linestyle':'--', 'label':'AMR-Wind'}}
    - {'file':'kaimal1.csv',  'xcol':'f', 'ycol':'Suu',
      'lineopts':{'color':'k', 'lw':1, 'linestyle':'-', 'label':'Kaimal'}}
```





AMR-Wind frontend

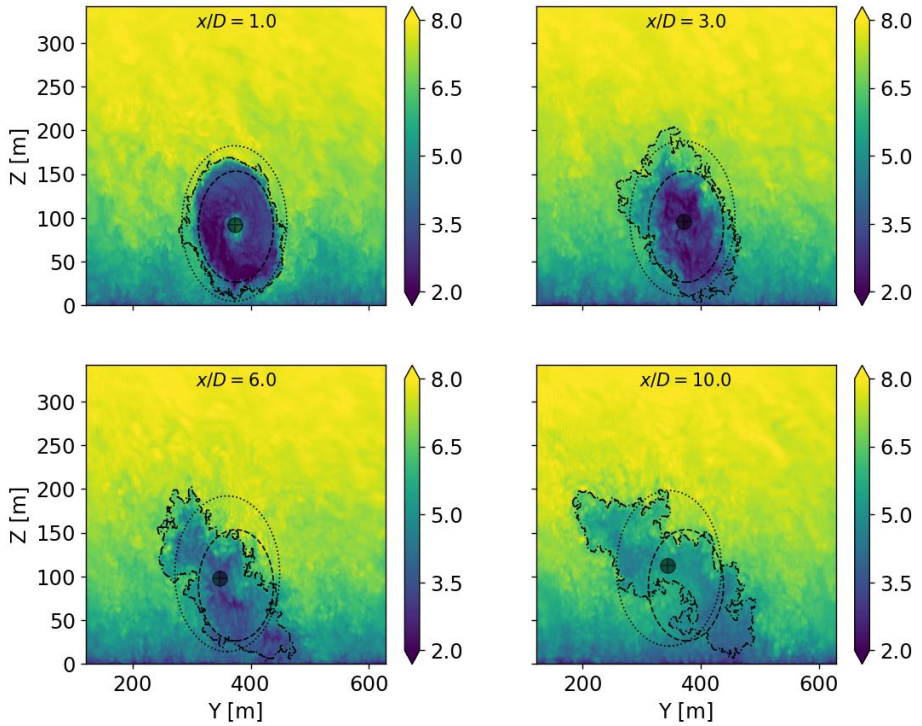
Example (Wake Tracking & Meandering Statistics)

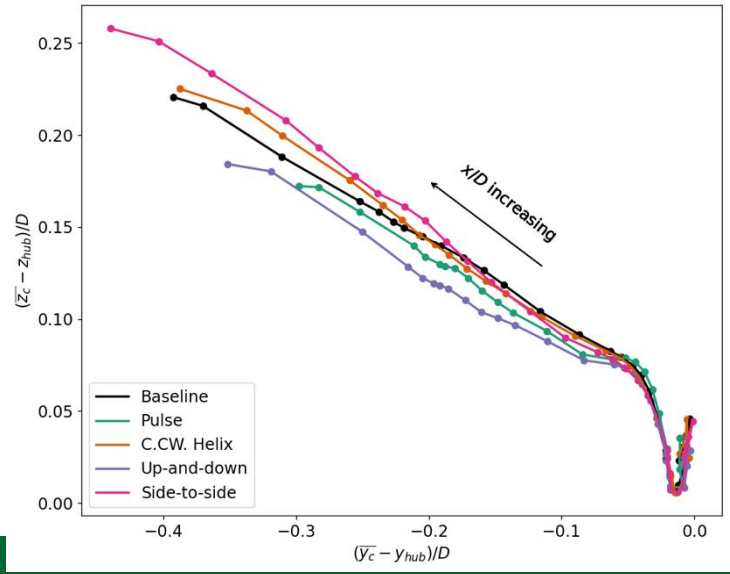
```yaml
wake_meander:
  name: Wake YZ plane
  ncfile: ../PA_1p25_2/YZslice_05.00D_456.00s_1556.00s_AWCOFF.nc
  group: xslice
  method: ConstantArea
  yhub: 375.0
  zhub: 87.617700
  diam: 177.8
  savefile: wake_center_05.00D.csv
  output_dir: ./wake_meandering/

  plot:
    xlabel: 'Y [m]'
    ylabel: 'Z [m]'
    iter: 0
    savefile: wake_center_05.00D.png

  statistics:
    savefile: wake_stats_05.00D.csv
    mean: True
    std: True
    anisotropy: True
```

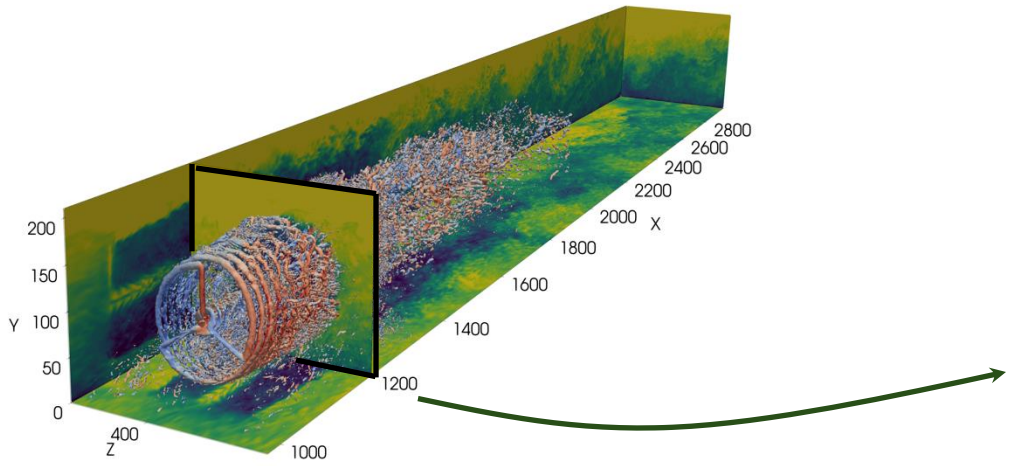
→ Integration with the SAMWICH package to compute wake boundary and center



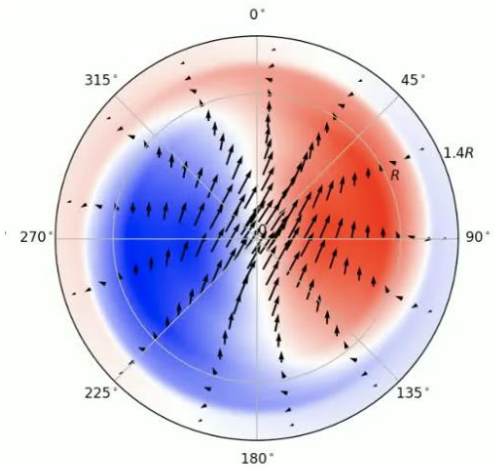


AMR-Wind frontend

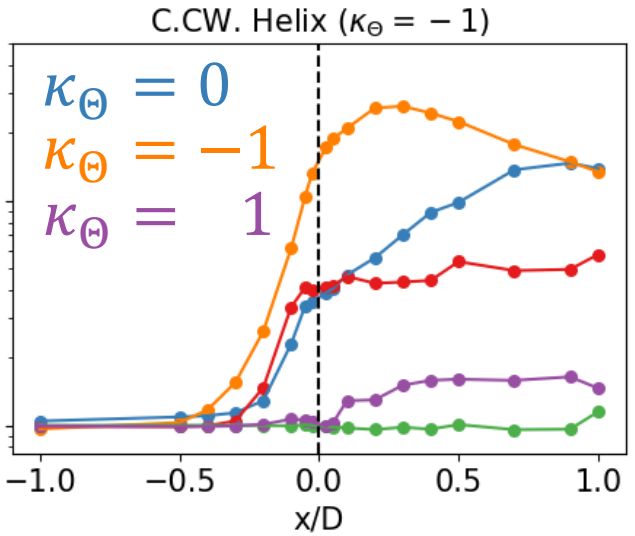
$$\int_{\Omega} \boldsymbol{S}(x, x', f')\boldsymbol{W}(x')\boldsymbol{\psi}(x', f')\,\mathrm{d}x' = \lambda(f')\boldsymbol{\psi}(x, f').$$



Eigenvectors



Eigenvalues



Example (SPOD Analysis of Cross Flow Planes – Work in progress)

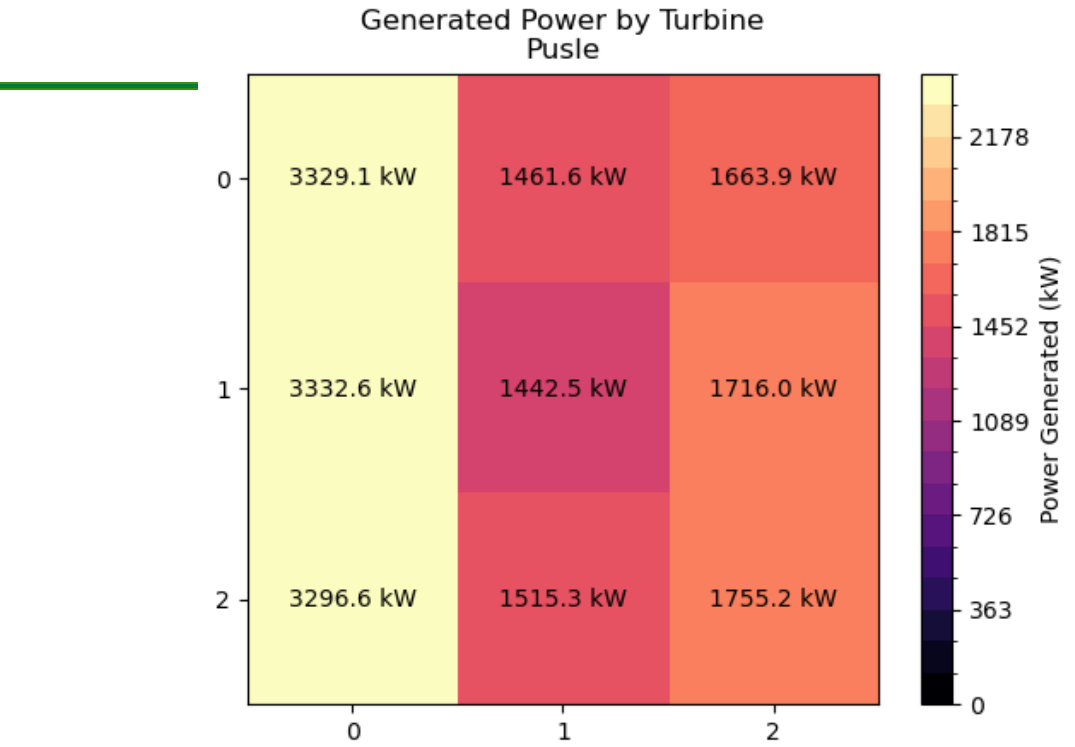
```
spod:
  name: Wake YZ plane
  ncfile: ../../data_converter/PA_1p25_2/YZslice_05.00D_456.00s_1556.00s_n1m.nc
  group: xslice
  nperseg: 256
  diam: 127
  yc: 375.0
  zc: 90
  wake_meandering_stats_file: ../../data_converter/wake_meandering_baseline_area_1p4R/wake_meandering/wake_stats_05.00D.csv
  correlations:
    - U-V-W
    - U
  output_dir: ./
  savepklfile: spod_05.00D.pkl
  save_num_modes: 100
```

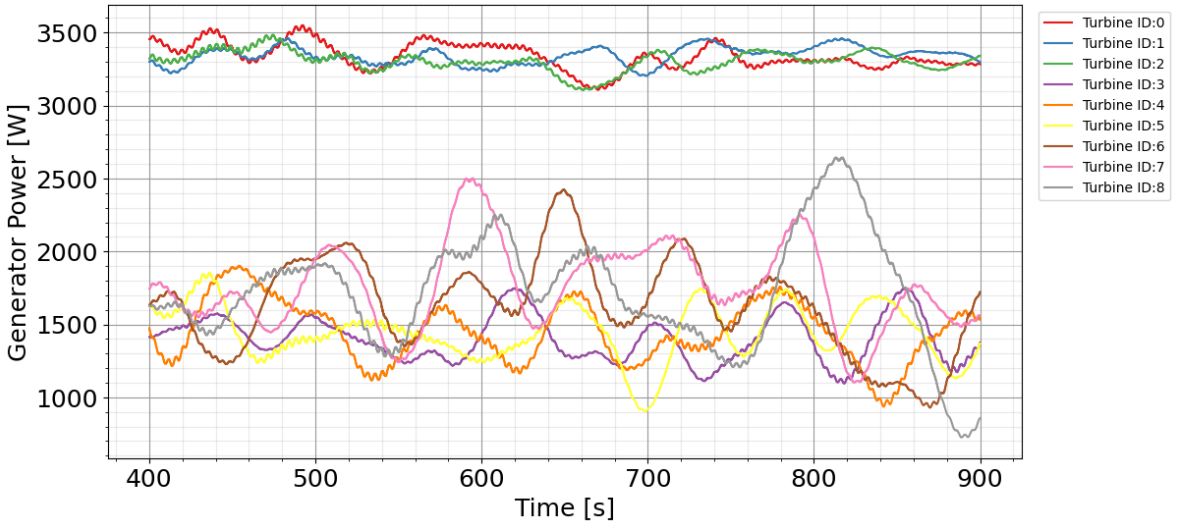
AMR-Wind frontend

Example (OpenFAST postprocessing)

```yaml
openfast:
  name:
  - T1_6D
  filename:
  - ./T1_OpenFAST3p5_IEA15MW/IEA-15-240-RWT-Monopile/IEA-15-240-RWT-Monopile.out
  extension: .csv
  output_dir: ./OpenFAST/
  vars:
    - Time
    - BldPitch1
    - RotSpeed
    - RotThrust
    - GenPwr
    - RootMyb1
    - RootMxb1
    - GenTq
    - RotTorq
    - AB1N038Alpha
    - TwrBsFxt
    - TwrBsMxt
    - YawBrMxp
    - LSSTipMys
    - LSSTipMzs
  csv:
    individual_files: True

  operate:
     operations: ['mean','std','DEL']
     trange: [120,600]
     awc_period: False
     awc: baseline
     St: 0.3
     diam: 240
     U_st: 6.5
```





AMR-Wind + frontend

- Questions, discussion