Ensemble of Python-based Steady-state Analytical Wind Farm Wake Model Implementations

Keywords: wind farm flow control, analytical wake modeling

1. Introduction

Wind farm flow control presents an opportunity to improve the cost of energy and energy yield of wind farms in specific environments. A class of software tools that model the effects of wake losses and opportunities for wake loss mitigation via wind farm flow control has emerged in the past decade. Many of these tools share key characteristics [1,2,3]:

* Written in the Python programming language
* Implement steady-state, analytical wake models
* Target workstation use
* Integrate with optimization software ecosystems
* Implement multiple mathematical models

However, software design decisions and implementation details can lead to discrepancies and errors across tools that reference corresponding models. While the implicit uncertainty in this class of software is not yet characterized, the bankability of wind farm controls depends on the accuracy of the analytical tools available in the industry [4].

Building on prior efforts in the wind farm flow control community, this work characterizes the ensemble of results across this class of software for a set of characteristic test cases. In addition to the comparison itself, an open-source, online dashboard is established as a living document to enhance collaboration across research organizations and coordinate the presentation and accessibility of the data.

1. Method

Research software often requires satisfying competing constraints and incentives from funding agencies and community needs. The myriad of constraints requires developers to settle on implementation details that can influence accuracy of results such as:

* Order of accuracy of numerical approximations
* Order of accuracy of discretization schemes
* Practicality and visibility of default configurations
* Considerations for optimization of computational performance

A set of implementation-specific test cases are developed to characterize the impact of implementation details including:

* Grid dependency
* Velocity averaging methods
* Wake superposition methods
* Partial wake modeling

Additionally, characteristic test cases are developed using reference wind turbines[5] in layouts designed to capture modern wind farm phenomenon including[6]:

* High turbine density
* Low ground clearance rotors
* Floating offshore wind farms with variable turbine locations due to slack in mooring lines
* Neighbouring wind farms
* Active flow control methods

*wcomp*[7], a recently published software framework,allows any Python-based steady-state wake modeling software conforming to the IEA Task 37 windIO[8] common ontology to compare results alongside other conforming software via a common set of output data structures and plotting methods. All test cases developed here are expressed in the windIO ontology and results compared within the *wcomp* framework to establish the spread of results for realistic and artificial exercises.

1. Expected Results

Since this study is ongoing, this section provides a general overview of the types of comparisons to be included in the final results.

* 1. Implementation details – Grid dependence

Diagram

Description automatically generatedChart, line chart

Description automatically generated

**Figure 1.** Left – A successively coarsened grid on the rotor swept area; Right – The rate of convergence of the integrated values on the grid points as a function of point-count (M).

The effect of grid-point density will be evaluated and an order of convergence established. In Figure 1, grid points on the rotor swept area are increased and a quantity of interest is integrated. This comparison highlights the grid dependency and rate of convergence across implementations.

* 1. Profile plots

Chart, line chart

Description automatically generated

**Figure 2.** Left – Vertical profile of streamwise velocity through a many-turbine wind farm; Right –Cross section of streamwise velocity across a many-turbine wind farm.

The wake recovery and wake deflection captured in mathematical formulations and their implementations can be analysed by plotting 1D profiles of the velocity in a wind farm. Figure 2 highlights the differences in implementation of the same Jensen / top-hat wake model across three distinct software tools. The rates of wake recovery and wake spreading are notable differences.

1. Conclusion

This study establishes a basis for characterizing the ensemble of results from a set of Python-based steady-state wind farm wake modelling software tools. It develops a set of physically realistic wind farm flow control test cases as well as implementation-specific test cases to systematically exercise the software systems. The spread of results across the software tools are reported in a consistent form in an open-source, public facing dashboard intended to be a living database where new results can be submitted on an ongoing basis and community dialogue, including discussion and analysis of results, is supported. This study is expected to be an ongoing, community-focused activity that strengthens the community’s understanding of the existing tools and provides a mechanism for identifying the role of new models and implementations.

References

1. Mads M. Pedersen, Alexander Meyer Forsting, Paul van der Laan, Riccardo Riva, Leonardo A. Alcayaga Romàn, Javier Criado Risco, Mikkel Friis-Møller, Julian Quick, Jens Peter Schøler Christiansen, Rafael Valotta Rodrigues, Bjarke Tobias Olsen and Pierre-Elouan Réthoré. (2023, February). PyWake 2.5.0: An open-source wind farm simulation tool. DTU Wind, Technical University of Denmark.
2. Schmidt et al., (2023). FOXES: Farm Optimization and eXtended yield Evaluation Software. Journal of Open Source Software, 8(86), 5464, <https://doi.org/10.21105/joss.05464>
3. FLORIS. Version 3.4 (2023). Available at <https://github.com/NREL/floris>.
4. "Wind farm control: The route to bankability" position paper 2021 <https://www.dnv.com/Publications/wind-farm-control-198162>
5. Bortolotti, Pietro, Tarres, Helena C, Dykes, Katherine L, Merz, Karl, Sethuraman, Latha, Verelst, David, and Zahle, Frederik. *IEA Wind TCP Task 37: Systems Engineering in Wind Energy - WP2.1 Reference Wind Turbines*. United States: N. p., 2019. Web. doi:10.2172/1529216.
6. Meyers, Johan & Bottasso, Carlo & Dykes, Katherine & Fleming, Paul & Gebraad, Pieter & Giebel, Gregor & Göçmen, Tuhfe & Wingerden, J. W.. (2022). Wind farm flow control: prospects and challenges. 10.5194/wes-2022-24.
7. wcomp. Version 0.1 (2023). Available at <https://github.com/rafmudaf/wcomp>.
8. Bortolotti, Pietro, Bay, Christopher, Barter, Garrett, Gaertner, Evan, Dykes, Katherine, McWilliam, Michael, Friis-Moller, Mikkel, Molgaard Pedersen, Mads, & Zahle, Frederik. System Modeling Frameworks for Wind Turbines and Plants: Review and Requirements Specifications. United States. <https://doi.org/10.2172/1868328>