Comparison of steady-state analytical wake models implemented in wind farm analysis software

Keywords: wind farm wake modeling, software testing, wake model validation

Rafael Mudafort1  
1 Researcher, National Renewable Energy Laboratory, Golden, CO, US email: Rafael.mudafort@nrel.gov

Abstract. A common set of wind turbine wake mathematical models are implemented in a few, well-adopted computational tools for wind farm wake modeling. While the referenced mathematical formulations are common, implementation details may lead to differences in results. This study presents methods for systematically comparing the implementation of mathematical wake models in Python-based software projects, and a set of the models are directly compared.

1. Introduction

With growth in the size and number of installed wind farms, the study of mitigating loss in energy production due to turbine-turbine interactions through their wakes has become a major field of study with significant potential for increased efficiency and profitability of wind farms [1]. Because of the physical scales of the phenomenon that characterize wind turbine wakes, the possibility to understand and mitigate these effects directly depends on the accuracy of analytical tools [2]. To that effect, the decades of development of steady-state, analytical wake models for general purpose and specialized application, including wind farm flow control methods, has resulted in a wide field of models available [3]. This set of mathematical models balance low computational cost with low fidelity in the physics, and their implementation into software tools has enabled the ability to further study methods for reducing wind farm losses. This study presents a systematic comparison of mature research software for wind farm wake modelling, the common mathematical models implemented in them, and the unique implementation details in each.

Several commercial and open-source software projects aimed at both the research and industrial communities implement a subset of the published mathematical wake models. Though the number of active software projects related to this field is broad, this study focuses on three well known software within the research community: FLORIS from the National Renewable Energy Laboratory [4], FOXES from Fraunhofer Institute for Wind Energy Systems [5], and PyWake from the Technical University of Denmark [6]. Each of these software projects are in active development, as shown in Figure 1. They are all within a category of software that are developed in the Python programming language and in the object-oriented programming paradigm. All three projects are relevant within the domains of wind energy yield assessment, wind farm layout design, wind farm controls design, and similar topic areas. While each has its own overarching design objective, all three software prioritize time-to-solution performance and leverage numerical and data processing libraries typical within the Python ecosystem. All three software are free, open-source software with permissive licenses, and they're available on any computer system. A cross-listing of the published models included in each software project is given in Figure 2 where the common models across software projects are connected by lines. This scope of this study extends only to the models common across projects.

Timeline

Description automatically generated

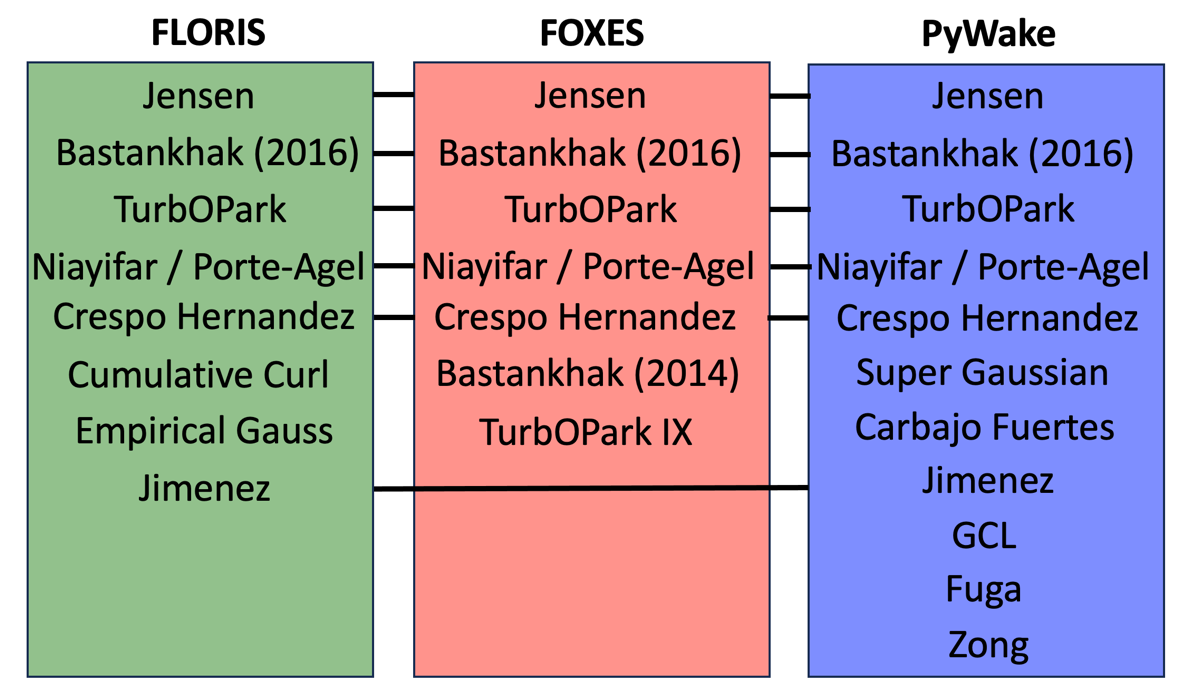
**Figure 1.** Release timeline for FLORIS, FOXES, and PyWake obtained from their respective source code repository release listings.

FLOw Redirection in Steady-state (FLORIS) has been in development at the National Renewable Energy Laboratory (NREL) in various forms since 2014, and the current iteration began in 2017. FLORIS is designed to support the development and implementation of new wake models, as well as wind farm energy yield calculation and wind farm layout and controls design. The software architecture is designed with explicit interfaces where new models can plug in to integrate into the existing framework.

Farm Optimization and eXtended yield Evaluation Software (FOXES) has been in development since 2022 at the Fraunhofer Institute for Wind Energy System (IWES). It is designed as a modular tool for modelling aspects of wind farm calculations allowing users to combine existing models and easily add new models. FOXES directly integrates with optimization libraries that allow for any variable to be optimized.

PyWake has been in development at the Technical University of Denmark since 2018 with the mission of “empowering users” guiding the ongoing development. It is used to study the interaction between turbines within a wind farm and its influence on the farm’s flow field and power production. It has a highly modular architecture providing predefined modelling blocks for calculating AEP.

Section 2 of this paper describes the methods and tools used to systematically compare the software projects. Section 3 details the comparisons of the wake models and implementation details. Section 4 summarizes the comparison and provides some key conclusions.



**Figure 2.** A listing of the published models in each software project included in this study.

1. Methods

The objective of this comparison is to evaluate the software tools relative to each other. The validity of a particular mathematical model to a real-world application is not considered here, and readers should review the literature around each model for that analysis. This study considers the software projects as two primary components: the mathematical formulation and the implementation in computer code, as shown in Figure 3. Approaching with the perspective of software testing, the “grey box testing” technique [7], common in software engineering, is applied here to both primary components of the software projects. The individual mathematical and computational models are isolated within the testing environment, and the relevant results from the test suite are compared across software.

Diagram

Description automatically generated

**Figure 3.** The two primary components of the studied set of wake modelling software: mathematical models and their implementation as computational models.

This study study leverages the wake model comparison framework *wcomp* [8] to express the tests and visualize the results. *wcomp* was made available by NREL in 2023, and it provides a consistent API for Python-based wake modelling projects to interface to the windIO ontology developed by the IEA Wind Task 37 [9], as shown in Figure 4.

Graphical user interface, diagram

Description automatically generated

**Figure 4.** Architecture of the Python-based wake modelling software comparison framework *wcomp* used to integrate the software compared in this study.

1. Models compared

Referring to the two primary components outlined in Section 2, this section begins the comparison of software projects with the mathematical models. Here, the wake behind a single turbine is profiled to evaluate its magnitude and direction using four one-dimensional profiles of the wake:

* A streamwise profile at hub height starting one rotor diameter upstream and extending 20 rotor diameters downstream
* Three horizontal profiles extending two rotor diameters from the rotor center and positioned at one, five, and 10 rotor diameters downstream.

The turbine definition is the IEA Task 37 15MW offshore reference turbine [10] and the operating wind speed is 9.8 m/s, turbulence intensity is 7.5%, and air density is 1.225 kg/m3. The turbine configuration is normal (no yaw misalignment) except in the wake deflection test cases where the yaw is 10°.

Chart, line chart

Description automatically generated

**Figure 5.** Common velocity deficit models implemented in FLORIS, FOXES, and PyWake compared with 1-dimensional profiles in the streamwise and spanwise directions.

Chart

Description automatically generated

**Figure 6.** Common deflection models implemented in FLORIS, FOXES, and PyWake compared with 1-dimensional profiles in the streamwise and spanwise directions. Contour plots of the FLORIS and PyWake wake fields are also compared direction in the right pane where the lower plot is a difference of the upper plots.

1. Results and Conclusions

This paper presents a methodology to systematically compare the common computational implementation of mathematical wind turbine wake models across a set of Python-based wind farm wake modeling software projects. In evaluating the results in Section 3, it can be concluded that the Jensen and TurbOPark implements are very consistent across software projects. The Niayifar / Porte-Agel deficit model has variation in implementation for the near wake region, which is undefined in the mathematical model, but all implementations see reasonable agreement in the far wake. The Jimenez deflection model also shows consistency across implementations. However, the results included here are limited in breadth and depth, and immediate future work will extend these studies to include a comprehensive range of computational model components within the included software projects.

This study presents the initial framework for what the authors intend to be a regular component of the wind turbine wake modeling research community. Imminent future work in this regard will include close collaborations between the primary developers of each software project to add additional parameterized studies. The extension of these test cases to real world data is also anticipated. Further, this study compares the results of the software to each other, and this comparison would be strengthened by including published reference data sets such as those included with the formulations for the mathematical models, high fidelity modeling data, and field data. Finally, this method and results are planned to be hosted on an online, open-source dashboard in which anyone in the wind farm wake modeling community can submit new test cases for comparison across the included projects, and new projects can register their software to be included in the comparison.

References

1. Meyers, Johan, et al. "Wind farm flow control: prospects and challenges." Wind Energy Science Discussions 2022 (2022): 1-56.
2. Boccolini, M., et al. Wind Farm Control: The Route to Bankability. Tech. rep., DNV, 2021.
3. Porté-Agel, Fernando, Majid Bastankhah, and Sina Shamsoddin. "Wind-turbine and wind-farm flows: A review." Boundary-layer meteorology 174.1 (2020): 1-59.
4. NREL. FLORIS. Version 3.5. https://github.com/NREL/floris (2023).
5. 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>
6. 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.
7. Lewis, William E., and Gunasekaran Veerapillai. Software testing and continuous quality improvement. Auerbach publications, 2004.
8. NREL. wcomp. Version 0.1. <https://github.com/NREL/wcomp> (2023).
9. 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>
10. Gaertner, Evan, Jennifer Rinker, Latha Sethuraman, Frederik Zahle, Benjamin Anderson, Garrett Barter, Nikhar Abbas, Fanzhong Meng, Pietro Bortolotti, Witold Skrzypinski, George Scott, Roland Feil, Henrik Bredmose, Katherine Dykes, Matt Shields, Christopher Allen, and Anthony Viselli. 2020. *Definition of the IEA 15-Megawatt Offshore Reference Wind*. Golden, CO: National Renewable Energy Laboratory. NREL/TP-5000-75698. https://www.nrel.gov/docs/fy20osti/75698.pdf