<table>
<thead>
<tr>
<th>Tool</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>FLORIS</td>
<td>- Control-oriented model &lt;br&gt; - Runs in fractions of seconds &lt;br&gt; - Can be used to find optimal control settings and analyze across wind rose to estimate AEP</td>
</tr>
<tr>
<td>FAST.FARM</td>
<td>- New code which overlays DWM wakes &lt;br&gt; - Includes embedded FAST models of turbines &lt;br&gt; - Runs on few cores, near real time, allowing load suite analysis</td>
</tr>
<tr>
<td>WindSE</td>
<td>- Solves the steady/unsteady 2D/3D RANS equations &lt;br&gt; - Adjoints included for large-scale optimizations &lt;br&gt; - Runs in serial or in parallel, in minutes</td>
</tr>
<tr>
<td>SOWFA</td>
<td>- Wind farm simulator based on large-eddy simulation &lt;br&gt; - Allows detailed investigation of wake physics, but requires many cores and time to run simulations</td>
</tr>
</tbody>
</table>
Modeling Tools at NREL

**FLORIS**
- Control-oriented model
- **Runs in fractions of seconds**
- Can be used to find optimal control settings and analyze across wind rose to estimate AEP

**FAST.FARM**
- Overlays DWM wakes
- Includes embedded FAST models of turbines
- **Runs on few cores**, near real time, allowing load suite analysis

**WindSE**
- Solves the steady/unsteady 2D/3D RANS equations
- Adjoint included for large-scale optimizations
- Runs in serial or in parallel, in minutes

**SOWFA**
- Wind farm simulator based on large-eddy simulation
- Allows detailed investigation of wake physics, but **requires many cores and time to run simulations**

**Increasing Flow Physics**
Modeling Tools at NREL

**FLORIS**
- Control-oriented model
- Runs in fractions of seconds
- Can be used to find optimal control settings and analyze across wind rose to estimate AEP

**FAST.FARM**
- Overlays DWM wakes
- Includes embedded FAST models of turbines
- Runs on few cores, near real time, allowing load suite analysis

**WindSE**
- Steady/unsteady 2D/3D RANS equations
- Adjoint included for large-scale optimizations
- Runs in serial or in parallel, in minutes

**SOWFA**
- Simulator based on large-eddy simulation
- Allows detailed investigation of wake physics, but requires many cores and time to run simulations

All are (will be) available open source on www.github.com
FLORIS: Controls-oriented wind farm model

- Computationally inexpensive (<1s for 100 turbines)
- [https://github.com/NREL/floris](https://github.com/NREL/floris)

**Models**
- Wake models
- "Turbulence Models"
- Turbine models

**Tools**
- Visualization
- Optimization
- Analysis
Wake Models

• Jensen (Park) Model – 0.0018 s

• Multi-zone wake model – 0.0019 s

• Gaussian wake model – 0.0025 s

• Curl model – 1.6 s
Wake Models - Gaussian

- Analytical solution to the simplified linearized Navier-Stokes equations
- Dependent on physical parameters that can be measured in the field
  - Ambient turbulence intensity
  - Shear
  - Veer
- Only 4 tuning parameters
- Good for normal turbine operation
Wake Models - Curl

• Solves the linearized Navier-Stokes equations in time marching fashion

• Dependent on physical parameters that can be measured in the field
  • Ambient turbulence intensity
  • Shear
  • Veer

• Only 2 tuning parameters

• Good for wake steering analysis
Deflection Models

- Gaussian model deflection
  - Wake is offset from centerline

- Curl model deflection
  - Counter-rotating vortices
  - Wake rotation
  - Secondary steering
Gaussian Model
Gaussian Model

Diagram showing a wind turbine with a label indicating "Yaw" and a Gaussian distribution.
Aerodynamics of Wake Steering

Curl

Yaw
Both the Gaussian and Curl model capture 2 turbine effects very well.
Aerodynamics of Wake Steering

- Curl
- Gauss
- No Yaw
- Yaw

- Curl
Overview of the Curl Model

Find $V, W$ analytically

\[
U \frac{\partial u'}{\partial x} + V \frac{\partial u'}{\partial y} + W \frac{\partial (U + u')}{\partial z} = v_{\text{eff}} \left( \frac{\partial^2 u'}{\partial x^2} + \frac{\partial^2 u'}{\partial y^2} + \frac{\partial^2 u'}{\partial z^2} \right).
\]

Turbines can work together to help build larger vortex-structures, developing flow control strategies throughout the farm
Compare Wake Steering

Relative power difference 6D downstream of 2nd turbine

- SOWFA
- FLORIS GAUSS
- FLORIS CURL

![Graph showing relative power difference](image)
“Turbulence” Models

- **Wake expansion** dependent on ambient turbulence intensity
- **Added turbulence** due to turbine operation
  - As $C_t$ increases, wake expansion increases
- Very important for investigating **deep array effects** (ongoing work)
Turbine Model - Cp/Ct Tables

- Turbine represented as **Actuator Disks**

- Generate **Cp/Ct tables** by:
  - *FAST* – Aeroelastic code
  - *CCBlade* – steady/state BEM coupled to FLORIS
Code Examples
FLORIS: Open-source and Collaborative

Available at: https://github.com/NREL/floris

Divided into two packages:
• simulation:
  • Contains code for FLORIS models
• tools:
  • Modules for interacting with FLORIS models and data

Documentation and examples available at: https://floris.readthedocs.io/en/develop/index.html
Example 0000: Open and Visualize FLORIS

Tools module allows for easy and intuitive interaction with FLORIS models.

All in python using open-source python modules.

• Line 15: import the FLORIS tools module
• Line 18: create FLORIS interface
• Line 21: calculate the wake
• Line 24: capture a horizontal cut-plane of the flow
• Line 31: use visualization module plot horizontal cut-plane
Example 0005: Changing Locations/Wind Direction

Programmatically change turbine and environmental parameters without re-loading the input file.

• Line 28: change turbine layout
• Line 63: change wind speed and direction
Example 0015: Compare with SOWFA

Module to load and interact with SOWFA data for analysis and comparison

- Line 27: use SOWFA interface to load SOWFA data
- Lines 55 & 61: set the relevant FLORIS model parameters to be equal to the SOWFA conditions

```python
26 # Load the SOWFA case in
27 s1 = wfct.sowfa_utilities.SowfaInterface('sowfa_example')

54 # Set the relevant FLORIS parameters to equal the SOWFA case
55 fi.reinitialize_flow_field(wind_speed=s1.precon_wind_speed,
56       wind_direction=s1.precon_wind_dir,
57       layout_array=(s1.layout_x, s1.layout_y))
59
60 # Set the yaw angles
61 fi.calculate_wake(yaw_angles=s1.yaw_angles)
```
Example 0010: Optimization

Perform yaw optimizations to investigate wake steering power gains.

- Line 59: create optimization object with min. and max. yaw angles
- Line 64: perform yaw optimization

```
Optimizing wake redirection control...
Number of parameters to optimize = 3

yaw angles =
Turbine 0 = 24.892413743776544 deg
Turbine 1 = 24.77131815924147 deg
Turbine 2 = 1.869530430117275e-05 deg

Total Power Gain = 8.2%
```
Ongoing Developments of FLORIS

- Incorporate **local effects** (currently: one wind speed/direction)
- **Deep-array effects** through better turbulence modeling
- Blade/Rotor **loads** calculations – using CCBlade from WISDEM
- Analytic **gradients** for large-scale optimizations (many turbines)
- Combinations of **optimizations** – Layout/Yaw/Thrust/Loads
- **FLORIS is a living code** – please let us know any suggestions on how we can address critical research questions in FLORIS to benefit the wind energy community.