NREL
Transforming ENERGY

WETO Software Stack User Workshops
Technoeconomic Analysis and Cost Modeling

June 12, 2024

Rafael Mudafort
Pietro Bortolotti
Garrett Barter
Rob Hammond
Sophie Bredenkamp
Nick Riccobono
Owen Roberts

Agenda

Section	Duration	Time	Speaker
Intro	5’	0:00 - 0:05	Rafael Mudafort
WETO Stack Overview	10’	0:05 - 0:15	Rafael Mudafort
WETO Stack discussion	**10’**	**0:15 - 0:25**	**You**
LandBOSSE ORBIT CORAL WOMBAT WAVES NRWAL	30’	0:25 - 0:55	Rob Hammond Sophie Bredenkamp Nick Riccobono Owen Roberts
TEA / Cost Model Roadmap	5’	0:55 - 1:00	Rob Hammond
Polls / open-ended questions	**2 - 5’**	**1:00**	**You**
Community discussion	**30’ - 40’**	**1:05 - 1:40**	**You**
Wrap up	5’	1:40 - end	Rafael Mudafort

Holistic Modeling Project

WETO Software Portfolio Coordination

US DOE & Lab-based Wind Research Projects

The **Wind Energy Technologies Office** invests in wind energy research, development, demonstration, and deployment activities that enable and accelerate the innovations needed to advance offshore, land-based, and distributed wind systems; reduce the cost of wind energy; drive deployment in an environmentally conscious manner; and facilitate the integration of high levels of wind energy with the electric grid.

WETO invests in wind energy **research, development, demonstration, and deployment** activities that enable and accelerate the innovations needed to advance wind energy.

NREL's active WETO projects

- Study on the Potential Application of Additive Manufacturing in Wind Turbine Components and Tooling
- Enabling Larger Rotors Through Modular, Customizable, Inflatable Blades
- Eagle Topic Area 3 Funding Opportunity Announcement (FOA) Support
- Co-Simulation Study and Control of a Wind Farm for Conversion Services
- Continental-Scale Transmission Modeling Methods for Grid Integration Analysis
- Atmosphere to Electrons to Grid (A2e2g)
- Fusion Joining of Thermoplastic Composites Using Energy Efficient Processes (TCF)
- Automating In-Situ Grinding and Repair for Thermoplastic Blades
- Codesign and Intelligent Approaches for Cost-Effective Operation and Maintenance of Generators and Power Converters
- Modeling and Validation for Offshore Wind
- Wind Power as Virtual Synchronous Generation (WindVSG)
- Technology Development and Innovation to Address Operational Challenges
- Evaluating Deterrent Stimuli for Increasing Species-Specific Effectiveness of an Advanced Ultrasonic Acoustic Deterrent
- North American Renewable Integration Study
- High-Fidelity Modeling
- Wind Turbine Drivetrain Reliability Assessment and Remaining Useful Life Prediction (TCF)
- Enabling Autonomous Wind Plants through Consensus Control (TCF)
- North American Energy Resiliency Model (NAERM)
- Big Adaptive Rotor
- Energy Sector Modeling and Impacts Analysis
- Floating Downwind Turbines: A Conceptual System-Level Design and Feasibility Study for U.S. Waters
- Wind Standards Development
- Multiscale Integration of Control Systems (EMS/DMS/BMS)
- Advanced Modeling, Dynamic Stability Analysis, and Mitigation of Control Interactions in Wind Power Plants
- Wind Grid Integration Stakeholder Engagement
- Atmosphere to Electrons (A2e) Performance Risk, Uncertainty and Finance (PRUF) Analysis Support
- Working Together to Resolve Environmental Effects of Wind Energy (WREN)
- High-Fidelity Modeling Toolkit for Wind Farm Development

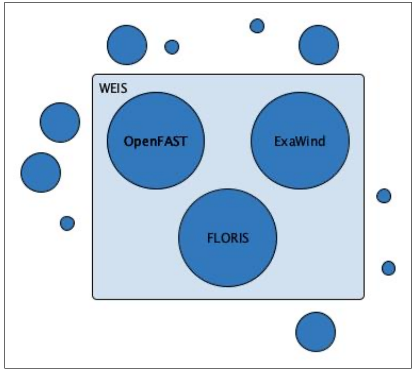
Argonne National Laboratory · Berkeley Lab – Lawrence Berkeley National Laboratory · Lawrence Livermore National Laboratory · NREL National Renewable Energy Laboratory · Pacific Northwest National Laboratory · Sandia National Laboratories

https://www.energy.gov/eere/wind/wind-energy-technologies-office-projects-map

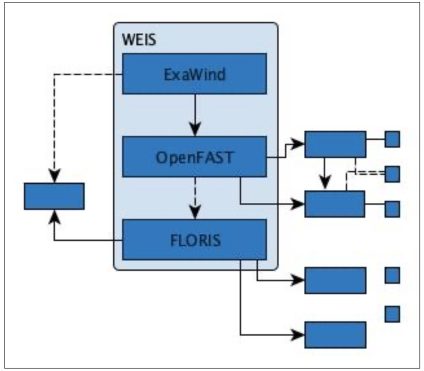
Holistic Modeling Project

Objective

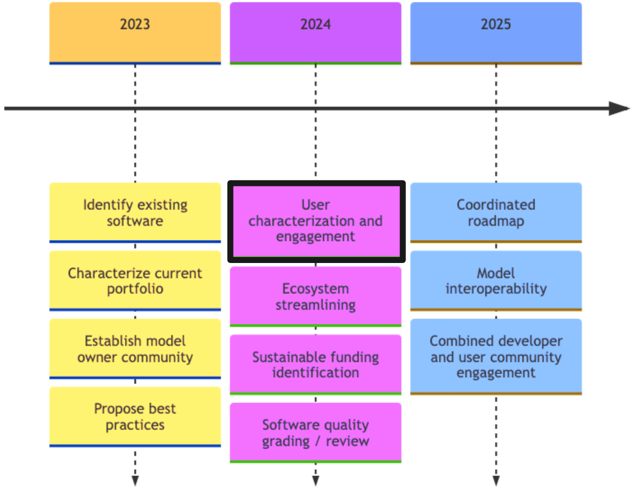
Past: Loose collection of software



Future: Cohesive software stack



Project Timeline

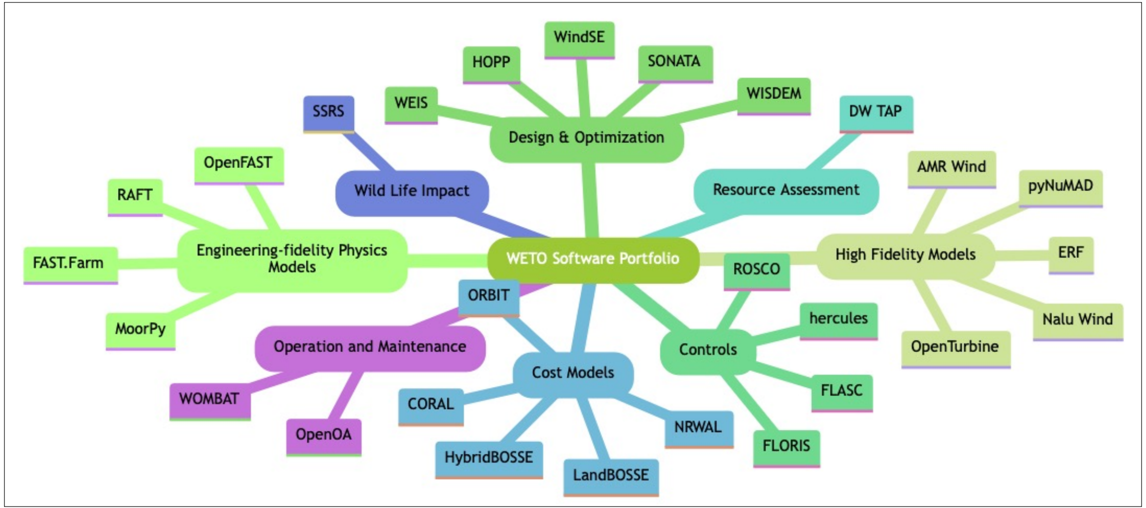


WETO Software Stack

Overview

WETO Software Stack

Grouped by what it does



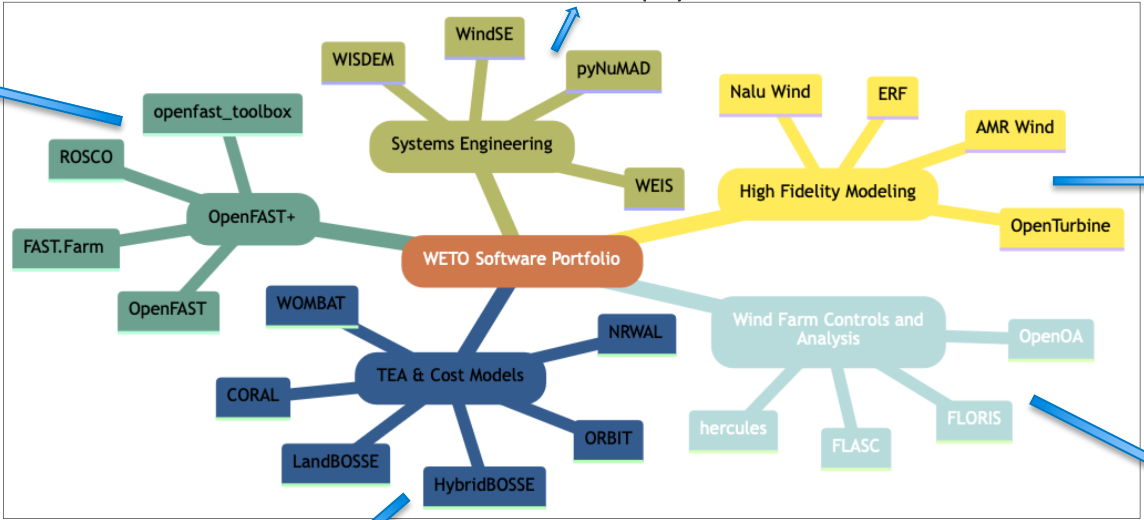
https://nrel.github.io/WETOStack/portfolio_analysis/software_list.html

WETO Software Stack

System-level optimizations with multi-physics and multi-fidelities

Grouped by how its used

Wind turbine and wind farm mid-fidelity physics



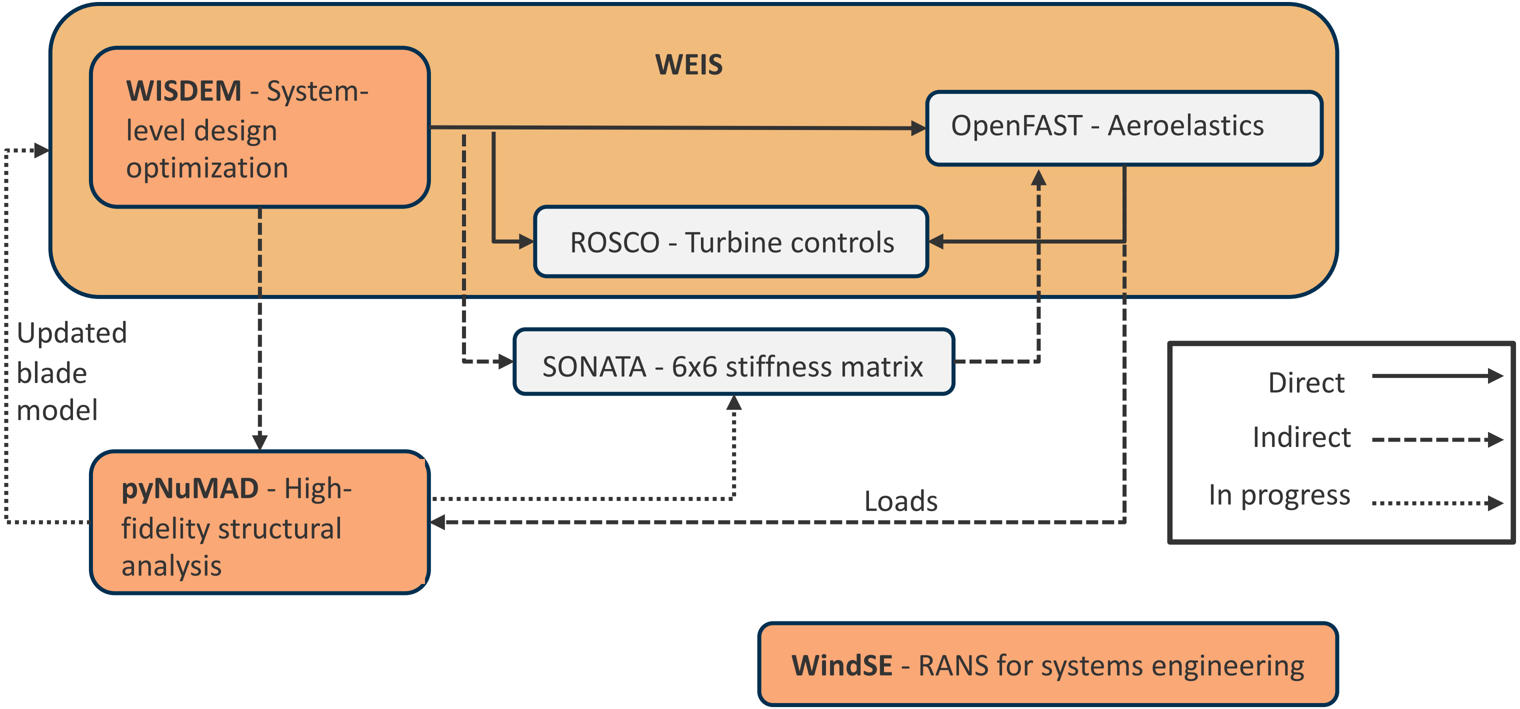
Detailed aerodynamics and structural dynamics

Wind turbine and wind plant costs for installation and operation, cost forecasting

Wind farm energy production, energy loss identification, and loss reduction through farm-level controls

Systems Engineering

Workshop: June 10



Adapted from Big Adaptive Rotor (BAR) project

Technoeconomic Analysis / Cost Modeling

Workshop: June 12

Energy Yield

Wind farm AEP estimate
FLORIS

CapEx

Balance-of-System
LandBOSSE
HybridBOSSE
ORBIT

Shared BOS Infrastructure
CORAL

OpEx

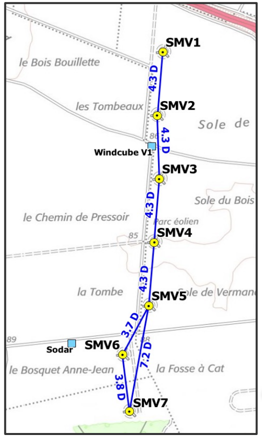
Operation & Maintenance
WOMBAT

Wind Asset Value Estimate
WAVES

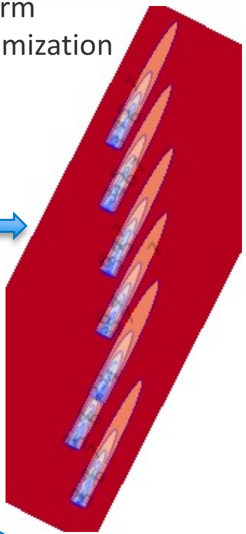
NRWAL: Offshore wind
system cost and scaling model

Wind Farm Controls and Analysis

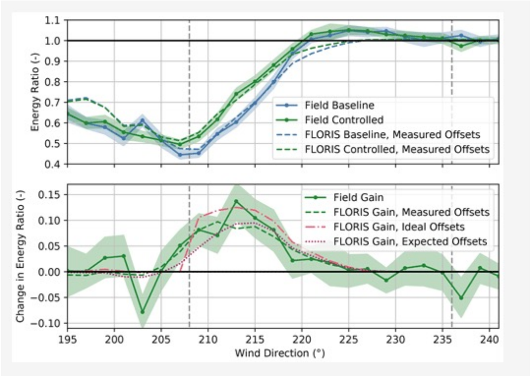
Workshop: June 18



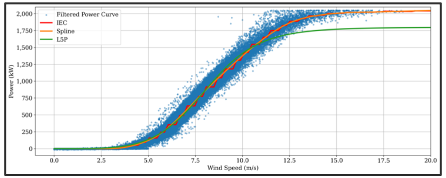
FLORIS: Steady-state modeling, farm controls optimization



FLASC: Validate FLORIS model with SCADA, compare control methods



Hercules: Realtime high-fidelity simulator for hybrid power plants with a specific focus on wind farm controls.

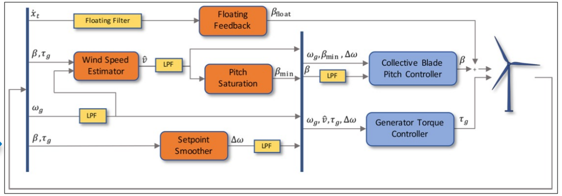


OpenOA: Characterize plant performance and quantify sources of operational loss

OpenFAST+

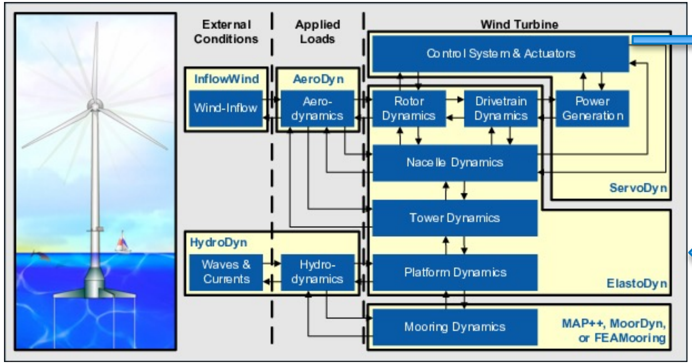
Workshop: June 20

ROSCO



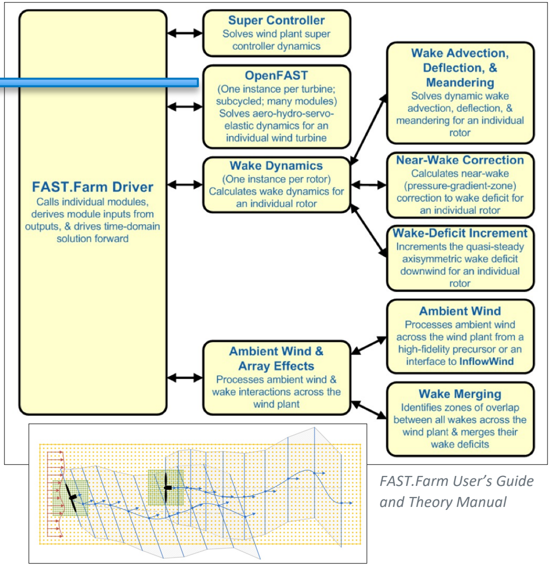
N. J. Abbas et al.: A reference controller for wind turbines

OpenFAST



OpenFAST v3.5.3 documentation

FAST.Farm

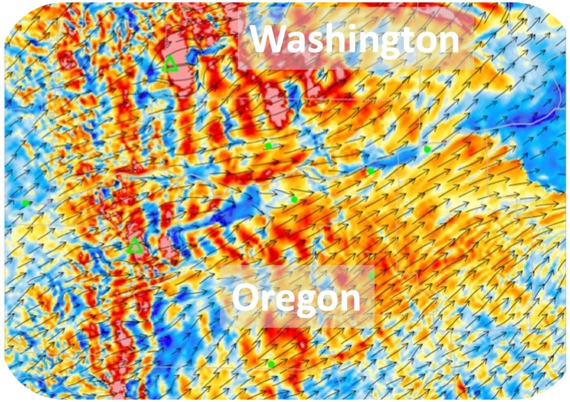


FAST.Farm User's Guide and Theory Manual

```
openfast_toolbox
```

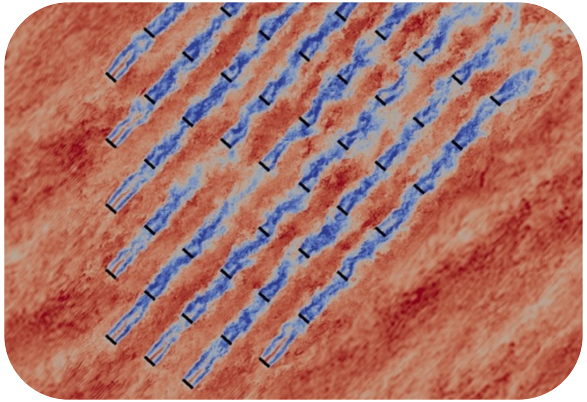
High Fidelity Models

Workshop: TBD



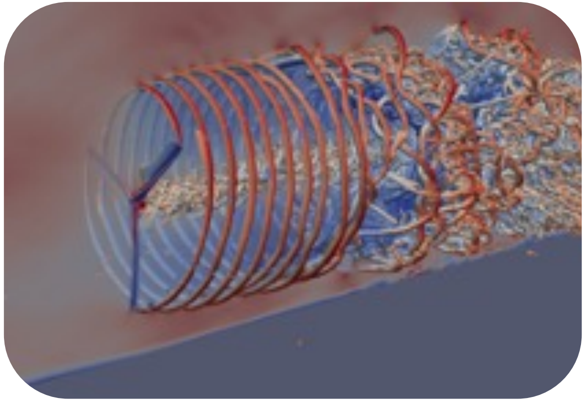
Mesoscale: ERF

- Regional scale weather
- **Scales 10 km to 1000 km**
- WRF numerics & models, built on AMReX
- GPU compatible
- Compressible



Microscale: AMR-Wind

- Atmospheric boundary layer
- **Scales less than 10 km**
- Large Eddy Simulation built on AMReX
- GPU compatible
- Structured grid with refinement zones
- Incompressible



Turbine scale: NALU-Wind

- Turbine, rotor, tower, nacelle
- **Scales less than 1 km**
- Unsteady Reynolds Averaged Navier Stokes
- GPU compatible
- Unstructured grid, geometry resolving
- Incompressible

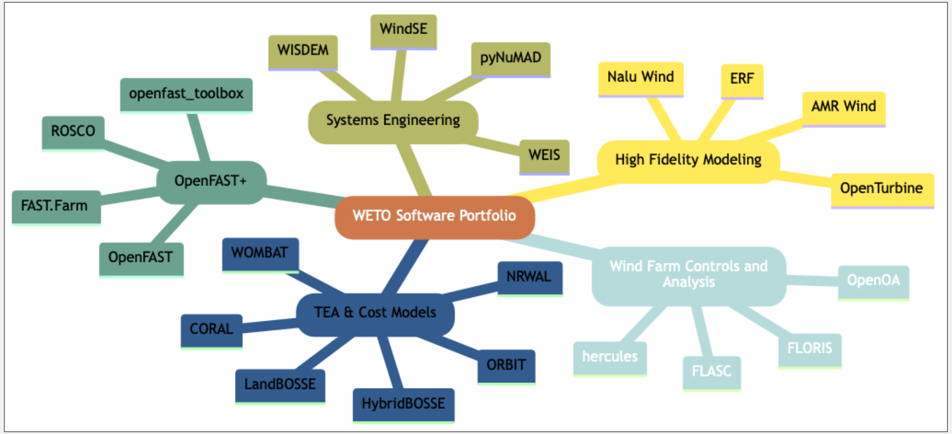
WETO Software Stack

Open Discussion

WETO Software Stack

Raise your “hand” and we’ll call your name to ask your question.

- **Discussion topics**
  - Prospective / new users:
    - What is your experience in the learning or onboarding process?
  - Experienced users:
    - What have been your primary pain points or bottlenecks?
    - What has or has not worked in integrating WETO software into your workflows?
    - How thoroughly do you feel you understand the capability of the tools available in the WETO Software Stack?
    - What has helped or hindered your open-source contribution to the WETO Software Stack?



TEA & Cost Models Overview

Rob Hammond

LandBOSSE

Owen Roberts

ORBIT

Nick Riccobono

ORBIT: Offshore Renewables Balance-of-system Installation Tool

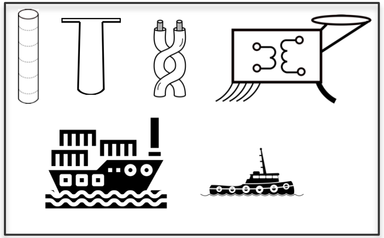
A simulation model to evaluate the impact of technology and process innovations on BOS costs

Objective:

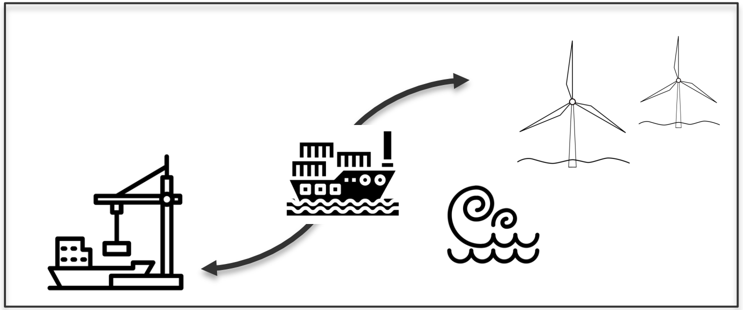
Provide a flexible, process-based simulation, incorporating novel technologies or installation methodologies.

Approach:

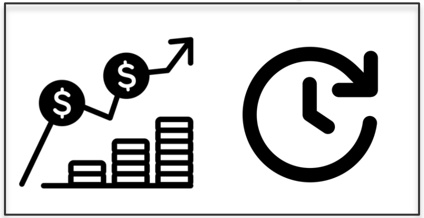
1. Define Inputs



2. Simulate installation



3. Evaluate Impact



Benefits:

Compares strategies and design choices to evaluate most impactful cost reduction opportunities.

Publicly available: https://github.com/WISDEM/ORBIT

Model architecture

Project Manager

Design Phases

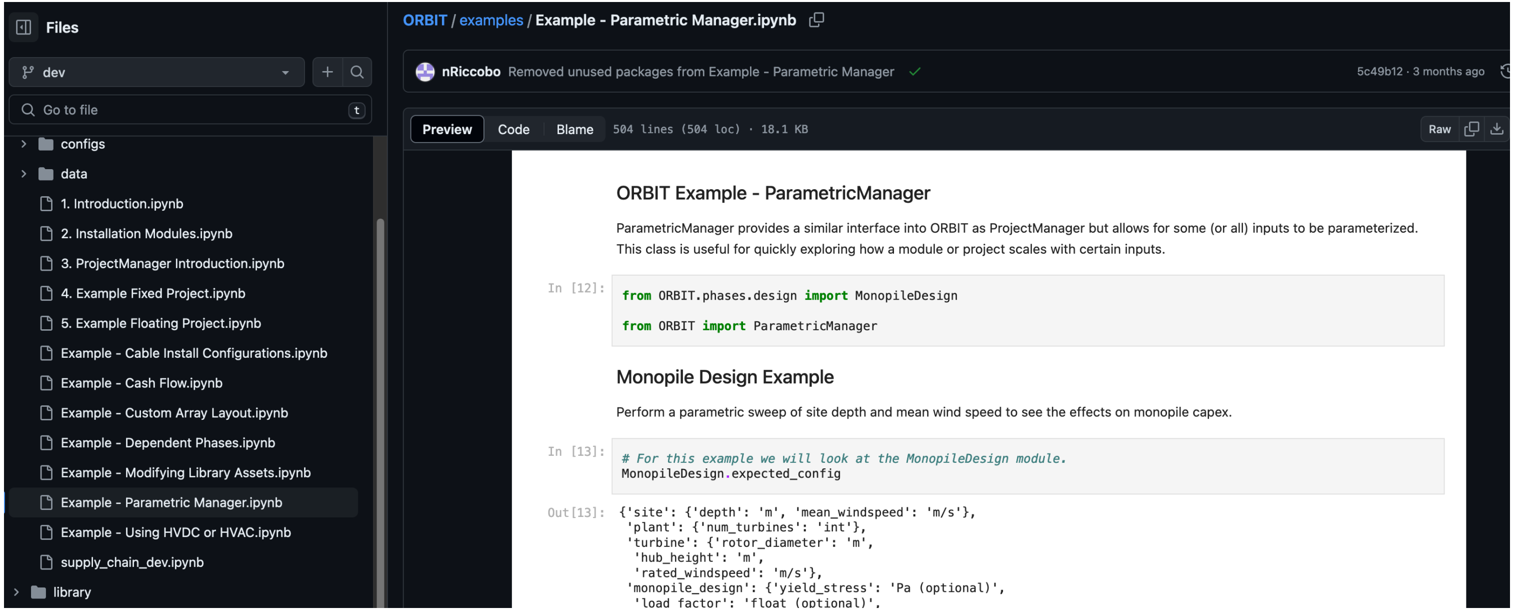
- Array cables
- Export system (cables + OSS)
- GBF
- Jacket
- Monopile
- Mooring system
- Scour protection
- Semisubmersible
- Spar
- Novel design system

Installation Phases

- Cables
- Jacket
- Monopile
- Mooring system
- OSS
- Floating platform + turbine
- Scour protection
- Turbine
- Novel installation

Examples

- ORBIT is written in Python 3.10
- Examples: https://github.com/WISDEM/ORBIT/tree/dev/examples



Example – Parametric Manager

<u>Monopile CapEx:</u>

Parameters: site depth, wind speed, soil coefficient

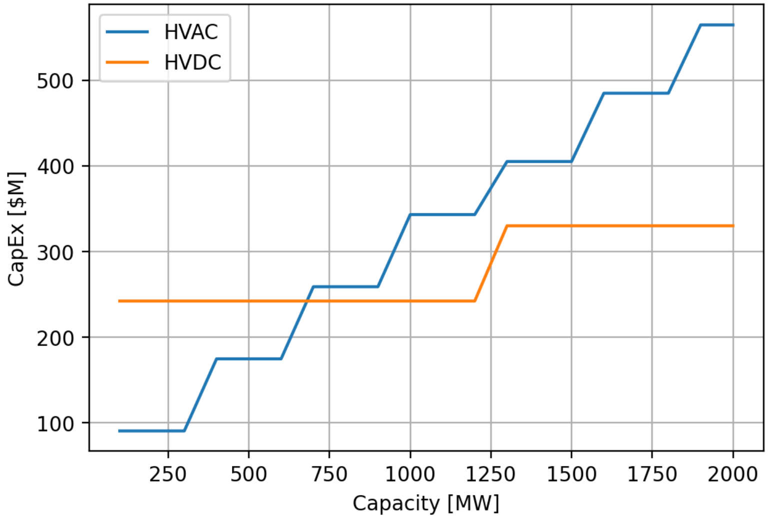
```
10 runs elapsed time: 0.02s
27 runs estimated time: 0.06s
```

	site.depth	site.mean_windspeed	monopile_design.soil_coefficient	capex
0	40	9	4000000	3.261239e+08
1	60	9	4000000	4.041444e+08
2	60	8	4000000	3.798072e+08
3	20	8	4500000	2.371472e+08
4	20	9	4500000	2.526935e+08
5	60	8	5000000	3.758376e+08
6	40	10	5000000	3.421996e+08
7	40	8	5000000	3.027043e+08
8	40	8	4000000	3.062888e+08
9	60	9	5000000	3.998461e+08

Example – Parametric Manager

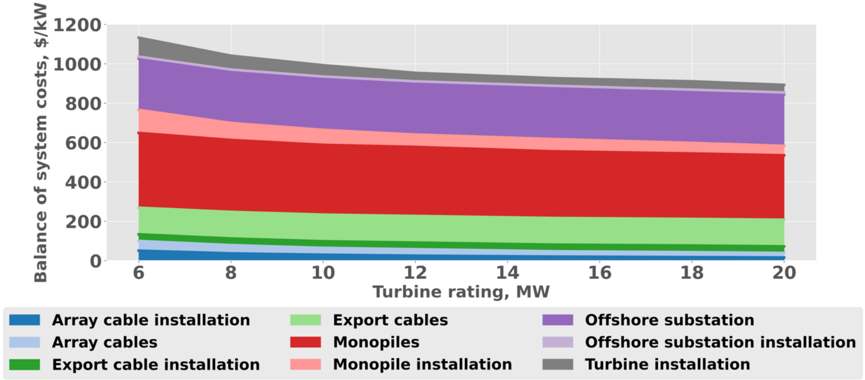
Substation CapEx:

Parameters: Farm capacity, Transmission type

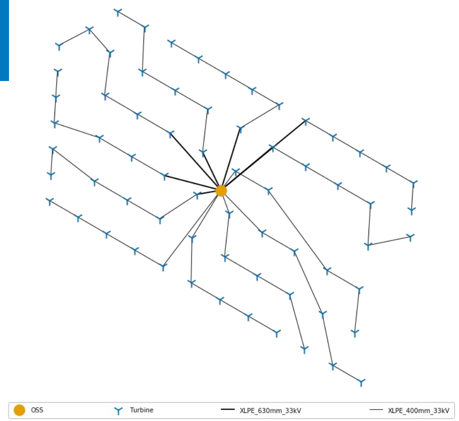


Example – Using HVDC or HVAC

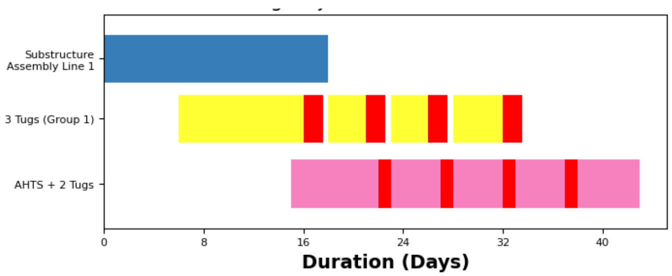
Other Examples & Results



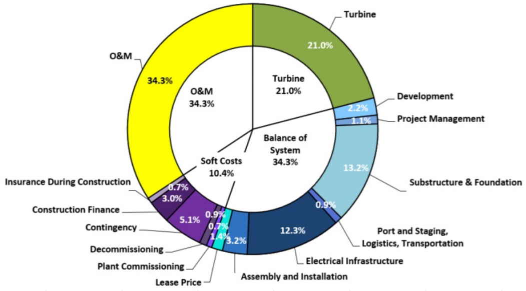
Impacts of turbine upsizing on balance of system cost
Shields, et al, Applied Energy (2021)



Example – Custom Array Layout



Installation schedule concept plot



Cost of Energy 2022: Cost breakdown for fixed-bottom offshore wind projects.
Source: Stehly, Beiter, and Duffy, 2022.

Reference

Technical Report:

Nunemaker, Jacob, Matthew Shields, Robert Hammond, and Patrick Duffy. 2020. "ORBIT: Offshore Renewables Balance-of-System and Installation Tool." NREL/TP-5000-77081, 1660132, MainId:26027. https://doi.org/10.2172/1660132.

Publicly Available:

https://github.com/WISDEM/ORBIT

Examples:

https://github.com/WISDEM/ORBIT/tree/dev/examples

Tutorials:

https://wisdem.github.io/ORBIT/source/tutorial/index.html

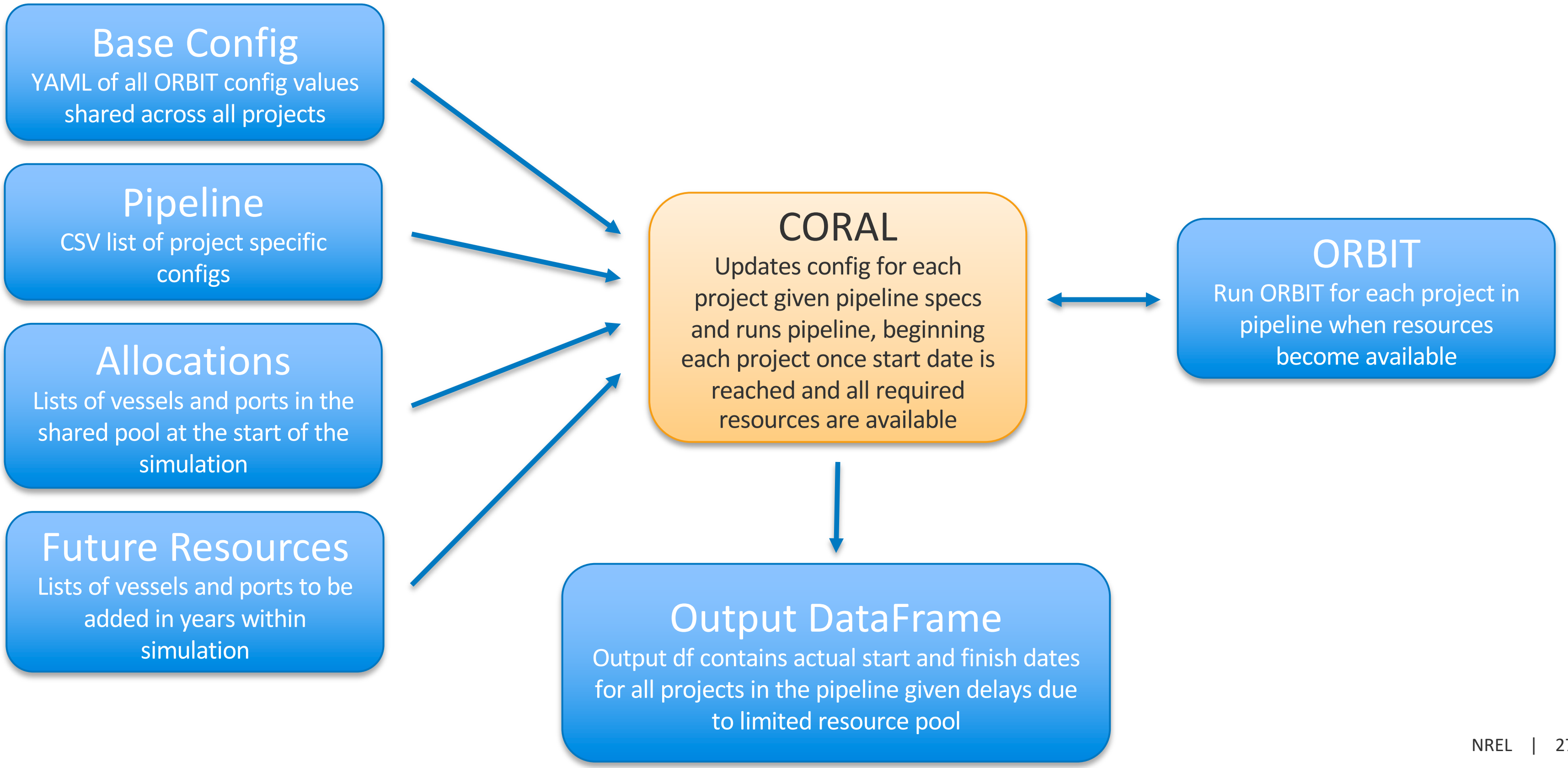
ORBIT Future Features

- Incorporate commodity prices (Steel, concrete, labor, etc.)
- Evaluate supply chain and logistical constraints
- Improve data visualization

CORAL

Sophie Bredenkamp

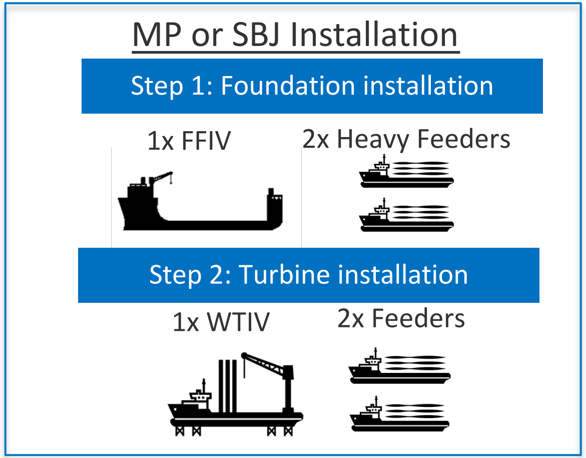
Concurrent ORBIT for shared Resource Analysis Library

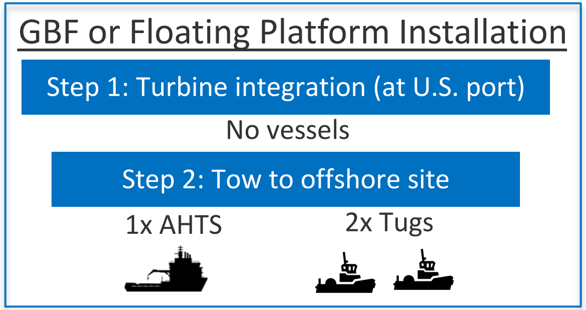


CORAL Capabilities

User has control over:

- Feedering vs Shuttling installation method
  - Choose to enforce feedering for entire pipeline or for specific projects
- Regional Ports
  - Choose to assign port regions rather than assigning a specific port to each project
- US vs foreign vessels
  - Distinguish between US and foreign flagged vessels in the allocations
- Foundation type
  - Assign different foundation types to each project in the pipeline to influence installation methods

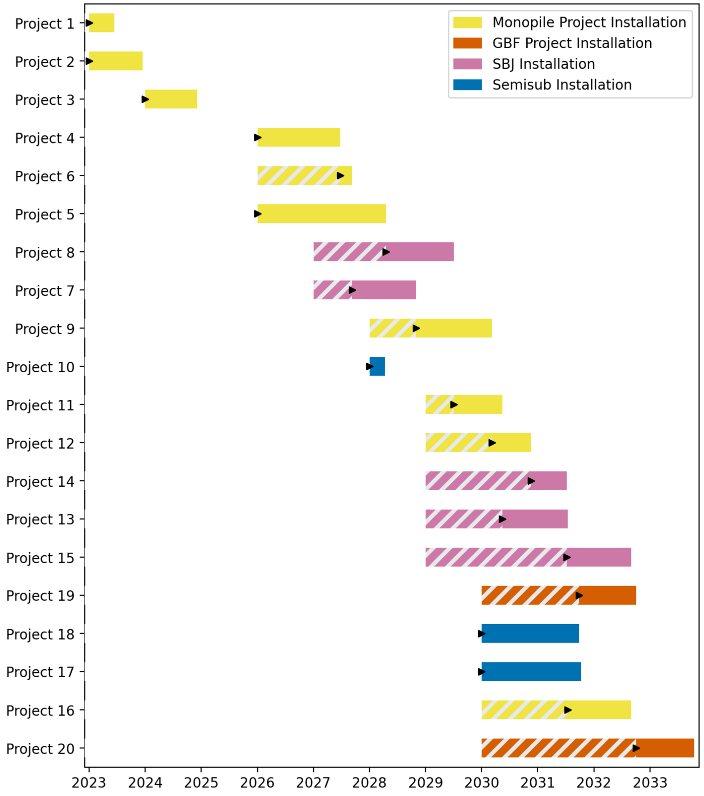




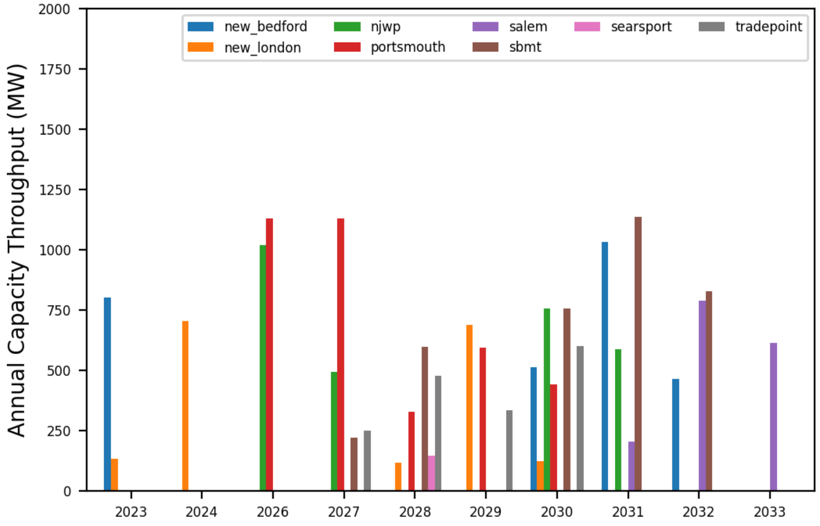
Demand versus supply in net vessel years for turbine installation, by installation year, excl. China
Image from Spinergie

Pipeline Results

Full Pipeline Gantt

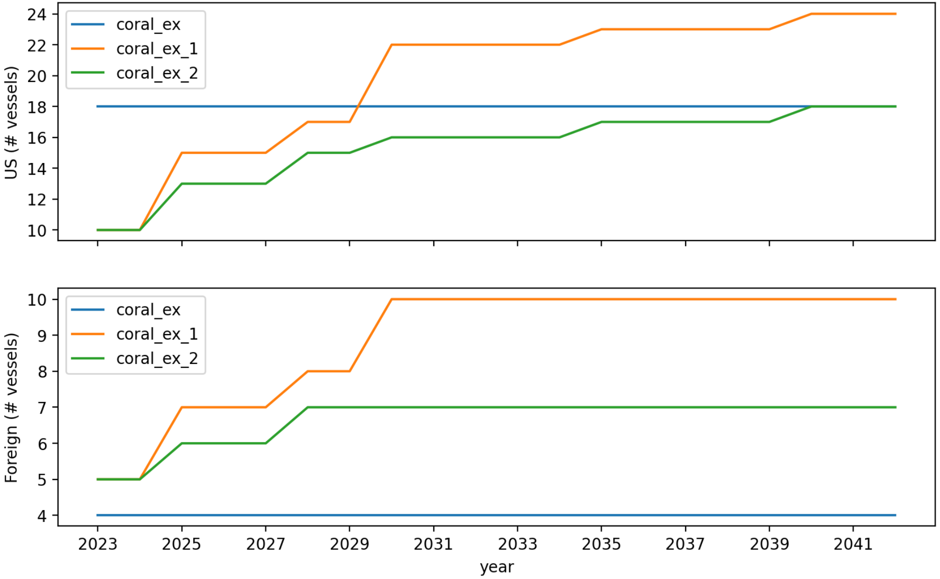


Annual Port Throughput

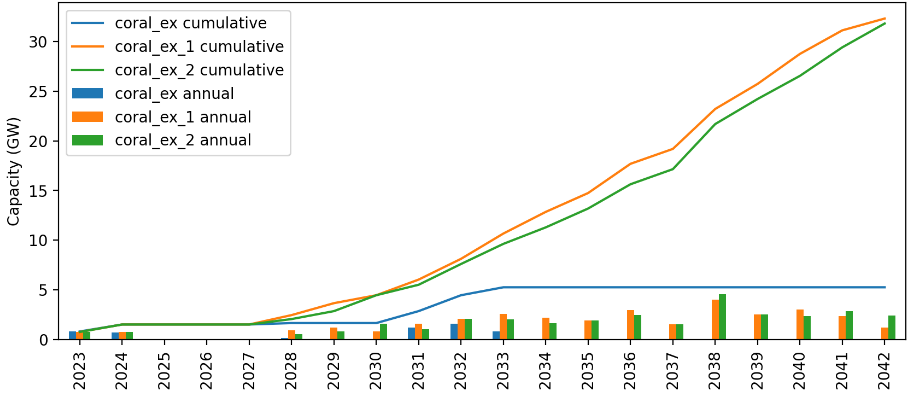


Summary Statistics

Vessel Count over Time



Installed Capacity



Set up Example

Define filepaths for configuration files

```
base = os.path.join(os.getcwd(), "analysis", "configs", "base.yaml")
base_float = os.path.join(os.getcwd(), "analysis", "configs", "base_float.yaml")
library = os.path.join(os.getcwd(), "analysis", "library")
base_baseline = os.path.join(os.getcwd(), "analysis", "pipelines", "base_baseline_pipeline.csv")
```

Define vessel and port allocations for the start of the simulation

```
allocations = {
    "wtiv": [('example_heavy_lift_vessel', 2),('example_wtiv', 2)],
    "feeder": ('example_heavy_feeder', 6),
    "port": [('new_london', 1), ('new_bedford', 1), ('sbmt', 1), ('njwp', 1), ],
    "ahts_vessel": ('example_ahts_vessel', 2),
    "towing_vessel": ('example_towing_vessel', 2),
}
```

Define future resources

```
wtiv_dates = [dt.date(2025,1,1), dt.date(2030,1,1)]
```

Run pipeline to update configs and manager to run the simulation

```
pipeline = Pipeline(base_baseline, base, base_float, enforce_feeders=True)
manager = GlobalManager(pipeline.configs, allocations, library_path=library)
manager.add_future_resources('wtiv', 'example_wtiv', wtiv_dates)
manager.add_future_resources('wtiv', 'example_heavy_lift_vessel', wtiv_dates)
manager.run()
```

Create the DataFrame with all project installation periods

```
df = pd.DataFrame(manager.logs).iloc[::-1]
```

Potential Future Work

- Build out US vs foreign flagged fleet flexibility
- Add tier 1 manufacturing tracking
    - Be able to set constraints on manufacturing throughput
    - Track tier 1 component demand throughout pipeline

- **Prioritize usability and documentation/examples**

WOMBAT

Rob Hammond

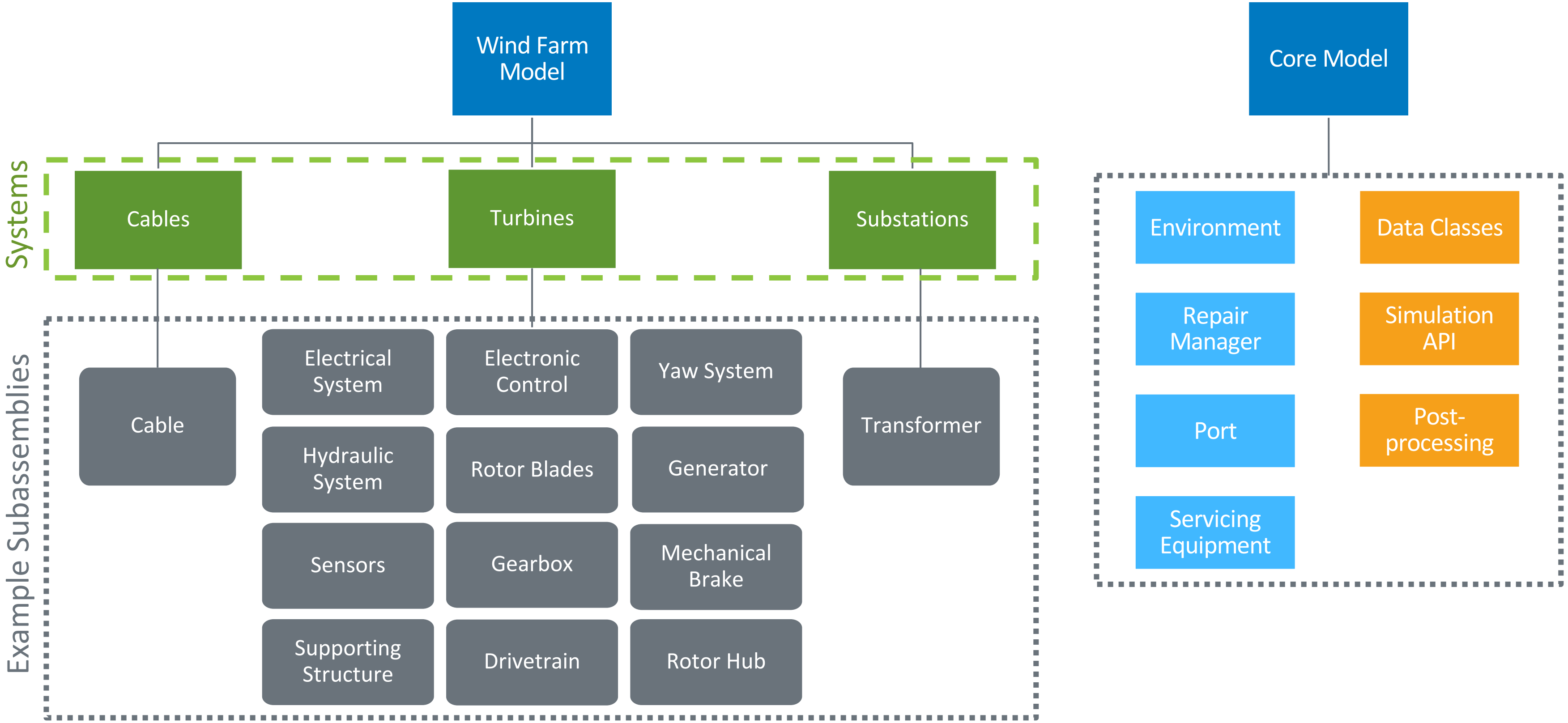
At a Glance

Thesis: How might maintenance strategies, technological innovations, and site conditions influence wind power plant operational expenditures (OpEx) and, ultimately, levelized cost of energy (LCOE)?

Under the hood:

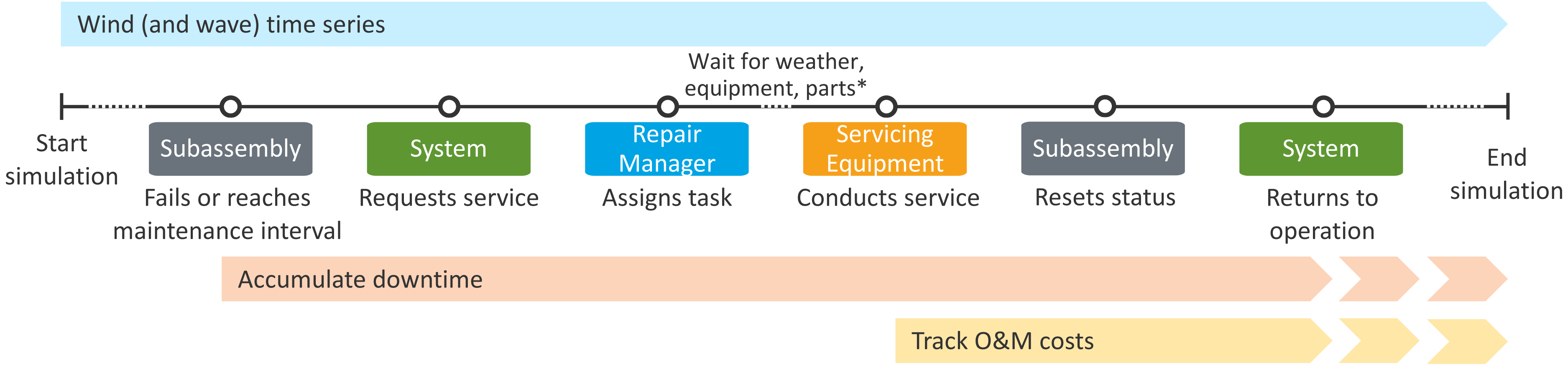
- Low code interface powered by YAML configurations.
- Prescriptive modeling via discrete event simulation to run what-if scenarios for decision making.
- Modular and flexible code base to enable:
  - Land-based, distributed, and offshore wind with little change in inputs for users.
  - Arbitrarily simple or complex failure models to understand technology tradeoffs as needed.
- Documentation driven development to ensure all aspects of the code are documented for a user to follow along.

High-Level Software Architecture



High-Level Simulation Architecture

- Model evaluates O&M costs using discrete event simulation (series of events in sequential order where no changes occur between events):
  - Allows for detailed documentation of a system and its processes.
  - Allows for a prescriptive approach for exploring specific impacts compared to an optimization with a “best choice.”



*Parts availability not yet implemented

Configuring WOMBAT

Baseline Inputs

Subassemblies

- Failure rate(s)
- Maintenance tasks
- Equipment requirements
- Cost and time to complete repairs
- Power curve (IEC power curve)

Servicing Equipment

- Maintenance strategy and related definitions
- Port (if required)
- Capabilities
- Labor rates
- Equipment rates
- Operational limits

Miscellaneous

- Weather profile
  - Hourly wind speed and/or wave height
- Wind farm layout
- Site working hours
- Port distance

Repairs and Maintenance

- Primary definitions for cables, substations, and turbines
- Maintenance (scheduled)
  - Fixed frequency model (number of days) to align with regular servicing intervals

- Failures (unscheduled maintenance)
  - Weibull distribution (exponential if using 1 for the shape parameter)
    - Scale = 1 / (# Failures/system/year) = Mean Time Between Failures (MTBF, in years)
  - Random sampling for the next time a failure occurs

- Common parameterizations
  - Materials cost (whole cost or proportional to system CapEx)
  - Repair time
  - Service equipment requirement (maps to vessel/crane definitions)
    - CTV, DRN, RMT, SCN, LCN, CAB, DSV, TOW, AHV (more details on the next slide)
  - Operating reduction
  - Severity level (priority designation)

Servicing Equipment

- Strategy
  - Scheduled
  - Unscheduled: downtime
  - Unscheduled: requests
  - Unscheduled: tow (offshore only)
- Capabilities
- Working hours
- Mobilization
- Port distance
- Crew
- Weather constraints
  - Wind speed
  - Wave height
  - Speed (including during inclement weather)
  - Nonoperational periods
  - Slowdown periods

CTV: crew transfer vessel (onsite truck for land-based)

DRN: drone

RMT: remotely operated vessel/vehicle

SCN: small crane (e.g., field support vessel)

LCN: large crane (e.g., WTIV, jackup vessel)

CAB: cable laying vessel (or similar for land-based)

DSV: diving support vessel

TOW: tugboat (requires a port)

AHV: anchor handling vessel (can be based out of a port, or operate like other vessels)

Setting Up the COREWIND Example

```
corewind
  cables
    __init__.py
    corewind_array.yaml
    corewind_export.yaml
  project
    config
      __init__.py
      morro_bay_in_situ.yaml
      morro_bay_tow_to_port.yaml
    plant
      __init__.py
      morro_bay_9D_layout_ttp.csv
      morro_bay_9D_layout.csv
      morro_bay_9D_layout.png
      morro_bay_9D_layout.svg
    port
      __init__.py
      morro_bay_port.yaml
    __init__.py
  results
  substations
    __init__.py
    corewind_substation.yaml
  turbines
    __init__.py
    2020ATB_NREL_Reference_15MW_240.csv
    corewind_15MW_ttp.yaml
    corewind_15MW.yaml
  vessels
  weather
    __init__.py
    central_ca.csv
```

morro_bay_in_situ.yaml M ×

library > corewind > project > config > morro_bay_in_situ.yaml >

```yaml
name: COREWIND Morro Bay In Situ
weather: central_ca.csv
service_equipment:
  - ctv1.yaml
  - ctv2.yaml
  - ctv3.yaml
  - ctv4.yaml
  - ctv5.yaml
  - ctv6.yaml
  - ctv7.yaml
  - cab.yaml
  - dsv.yaml
  - ahv.yaml
  - hlv1.yaml
layout: morro_bay_9D_layout.csv
port_distance: 60  # 20/100 for West of Barra Islan
inflation_rate: 0
workday_start: 6
workday_end: 22
start_year: 2002
end_year: 2021
project_capacity: 1200
```

hlv1.yaml ×

library > corewind > vessels > hlv1.yaml > {} crew

Rob Hammond, 7 months ago | 1 author (Rob Hammond)

```yaml
name: Heavy Lift Vessel
equipment_rate: 290000
capability: LCN
speed: 20.37  # 11 knots
strategy: requests
strategy_threshold: 1
max_windspeed_transport: 10
max_windspeed_repair: 10
max_waveheight_transport: 2
max_waveheight_repair: 2
onsite: false
mobilization_cost: 325000
mobilization_days: 10
workday_start: 0
workday_end: 24
charter_days: 20
crew_transfer_time: 0.25
n_crews: 1
crew:
  day_rate: 219.18  # 80K/yr / 365 days
  n_day_rate: 5  # Most tasks are 3 or 4,
  hourly_rate: 0
  n_hourly_rate: 0
```

COREWIND Configuration: https://github.com/WISDEM/WOMBAT/tree/main/library/corewind
COREWIND Example: https://github.com/WISDEM/WOMBAT/blob/main/examples/NAWEA_interactive_walkthrough.ipynb

Setting Up the COREWIND Example

Preview 'central_ca.csv'

Datetime	Windspeed	Waveheight
1/1/2002 0:00	1.78	2.86
1/1/2002 1:00	1.47	2.88
1/1/2002 2:00	1.05	2.88
1/1/2002 3:00	2.61	2.88
1/1/2002 4:00	5.14	2.88
1/1/2002 5:00	4.47	2.86
1/1/2002 6:00	1.74	2.84
1/1/2002 7:00	1.21	2.81
1/1/2002 8:00	2.43	2.77
1/1/2002 9:00	3.11	2.73
1/1/2002 10:00	1.13	2.69
1/1/2002 11:00	2.2	2.64

corewind_15MW.yaml

library > corewind > turbines > corewind_15MW.yaml > {} electr


```yaml
capacity_kw: 15000
capex_kw: 1300
power_curve:
  file: 2020ATB_NREL_Reference_15MW_240.csv
  bin_width: 0.5
electrical_system:
  name: electrical_system
  maintenance:
  - description: n/a
    time: 0
    materials: 0
    service_equipment: CTV
    frequency: 0
  failures:
    1:
      scale: 1.859
      shape: 1
      time: 14
      materials: 1000
      service_equipment: CTV
      operation_reduction: 0.0
      level: 1
      description: power converter minor repair
      # n_technicians: 2
```


corewind_array.yaml

library > corewind > cables > corewind_array.yaml > {}

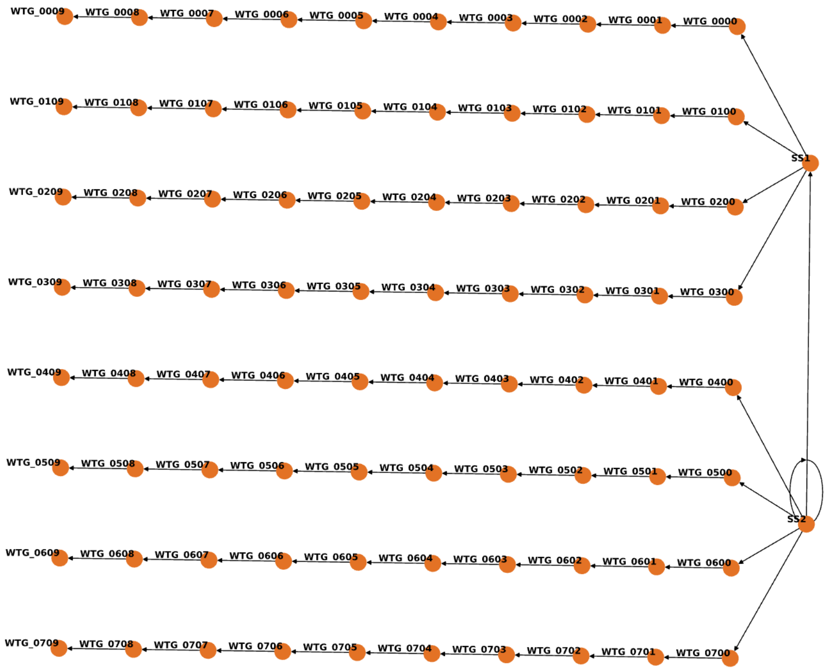
Rob Hammond, 7 months ago | 1 author (Rob Hammond)


```yaml
name: array cable
maintenance:
  -
    description: n/a
    time: 0
    materials: 0
    service_equipment: CTV
    frequency: 0
failures:
  4:
    scale: 40
    shape: 1
    time: 240
    materials: 30000
    operation_reduction: 0
    service_equipment: [CAB]
    replacement: false
    level: 4
    description: array cable major repair
    # n_technicians: 10
  6:
    scale: 62.5
    shape: 1
    time: 360
    materials: 220000
    operation_reduction: 0
    service_equipment: [CAB]
    replacement: true
    level: 6
    description: array cable replacement
    # n_technicians: 10
```


COREWIND Configuration: https://github.com/WISDEM/WOMBAT/tree/main/library/corewind
COREWIND Example: https://github.com/WISDEM/WOMBAT/blob/main/examples/NAWEA_interactive_walkthrough.ipynb

Setting Up the COREWIND Example

id	substation_id	name	type	longitude	latitude	string	order	subassembly	upstream_cable
SS1	SS2	SS1	substation	-121.6445116	35.40616396	9	0	corewind_substation.yaml	corewind_export.yam
SS2	SS2	SS2	substation	-121.6458132	35.32827984	9	1	corewind_substation.yaml	corewind_export.yam
WTG_0000	SS1	WTG_0000	turbine	-121.6678132	35.4356351	0	0	corewind_15MW.yaml	corewind_array.yaml
WTG_0001	SS1	WTG_0001	turbine	-121.6916045	35.43589529	0	1	corewind_15MW.yaml	corewind_array.yaml
WTG_0002	SS1	WTG_0002	turbine	-121.7153961	35.43615079	0	2	corewind_15MW.yaml	corewind_array.yaml
WTG_0003	SS1	WTG_0003	turbine	-121.739188	35.43640161	0	3	corewind_15MW.yaml	corewind_array.yaml



COREWIND Configuration: https://github.com/WISDEM/WOMBAT/tree/main/library/corewind
COREWIND Example: https://github.com/WISDEM/WOMBAT/blob/main/examples/NAWEA_interactive_walkthrough.ipynb

Results and Workflows

Running a Simulation

Configuring is 90% of the work!

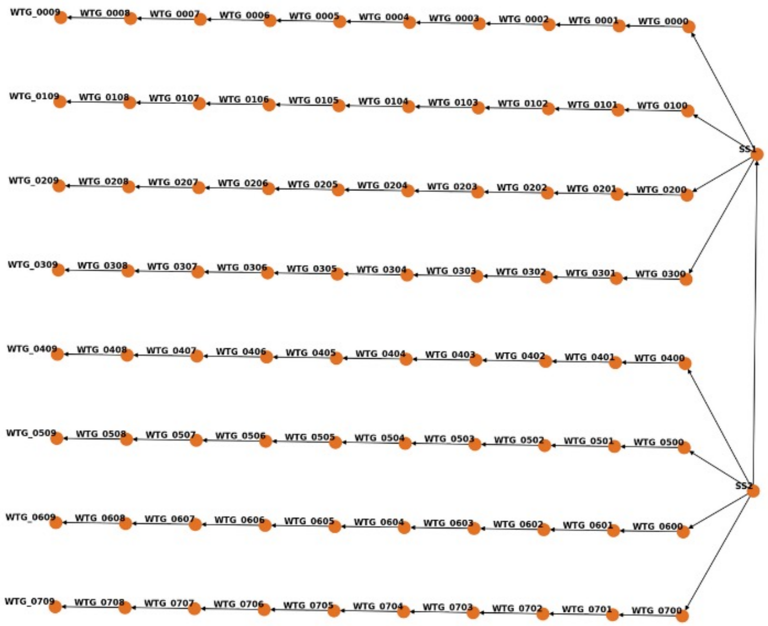
[1]:
```python
from wombat import Simulation

sim = Simulation(
    library_path="../library/corewind/",
    config="morro_bay_in_situ.yaml",
)
sim.run()
print("Done!")
```

```
Done!
```

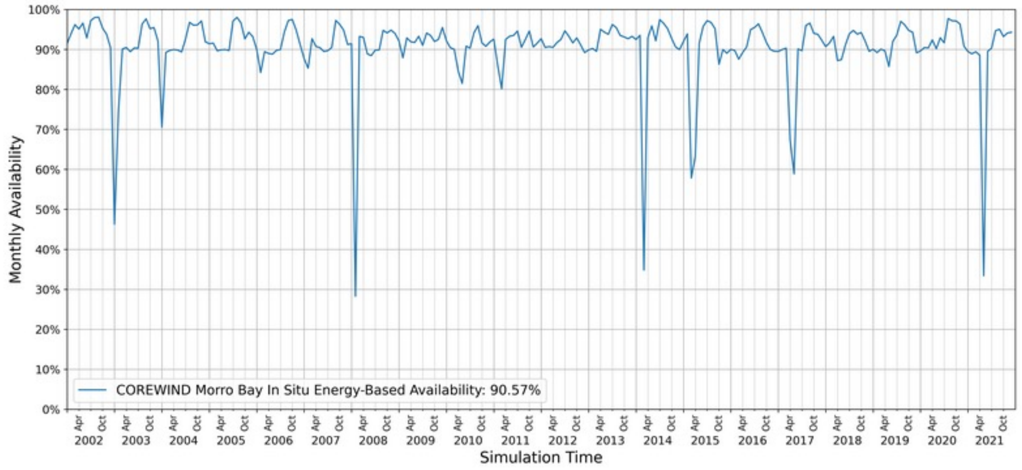
The next set of examples comes from a more in-depth explanation on understanding simulation results and comparing results between scenarios: https://github.com/WISDEM/WOMBAT/blob/main/examples/NAWEA_interactive_walkthrough.ipynb

Inspecting Results

```python
from wombat.utilities import plot
plot.plot_farm_layout(sim_in_situ.windfarm, plot_kwargs={"node_size": 200})
```




```python
plot.plot_farm_availability(sim_in_situ, which="energy")
```




```python
project_mw = sim_in_situ.windfarm.capacity / 1000
```

```python
opex = sim_in_situ.metrics.opex("project")
opex.index = ["In Situ"]
opex.loc["Tow-to-Port", "OpEx"] = sim_ttp.metrics.opex("project").values[0]
opex["OpEx (€/MW/yr)"] = opex.OpEx / project_mw / 20
opex = opex.rename(columns={"OpEx": "OpEx (€)"})
opex.style.format("{:,.2f}")
```

	OpEx (€)	OpEx (€/MW/yr)
In Situ	2,374,449,536.67	98,935.40
Tow-to-Port	1,638,630,272.56	68,276.26

Inspecting Results

```python
print("In Situ")
scheduled = sim_in_situ.metrics.task_completion_rate(which="scheduled", frequency="project").values[0][0]
unscheduled = sim_in_situ.metrics.task_completion_rate(which="unscheduled", frequency="project").values[0][0]
combined = sim_in_situ.metrics.task_completion_rate(which="both", frequency="project").values[0][0]
print(f"  Scheduled Task Completion Rate: {scheduled:.2%}")
print(f"Unscheduled Task Completion Rate: {unscheduled:.2%}")
print(f"    Overall Task Completion Rate: {combined:.2%}")
print()
print("Tow-to-Port")
scheduled = sim_ttp.metrics.task_completion_rate(which="scheduled", frequency="project").values[0][0]
unscheduled = sim_ttp.metrics.task_completion_rate(which="unscheduled", frequency="project").values[0][0]
combined = sim_ttp.metrics.task_completion_rate(which="both", frequency="project").values[0][0]
print(f"  Scheduled Task Completion Rate: {scheduled:.2%}")
print(f"Unscheduled Task Completion Rate: {unscheduled:.2%}")
print(f"    Overall Task Completion Rate: {combined:.2%}")
```

```
In Situ
  Scheduled Task Completion Rate: 96.46%
Unscheduled Task Completion Rate: 96.11%
    Overall Task Completion Rate: 96.24%

Tow-to-Port
  Scheduled Task Completion Rate: 71.72%
Unscheduled Task Completion Rate: 76.93%
    Overall Task Completion Rate: 75.01%
```

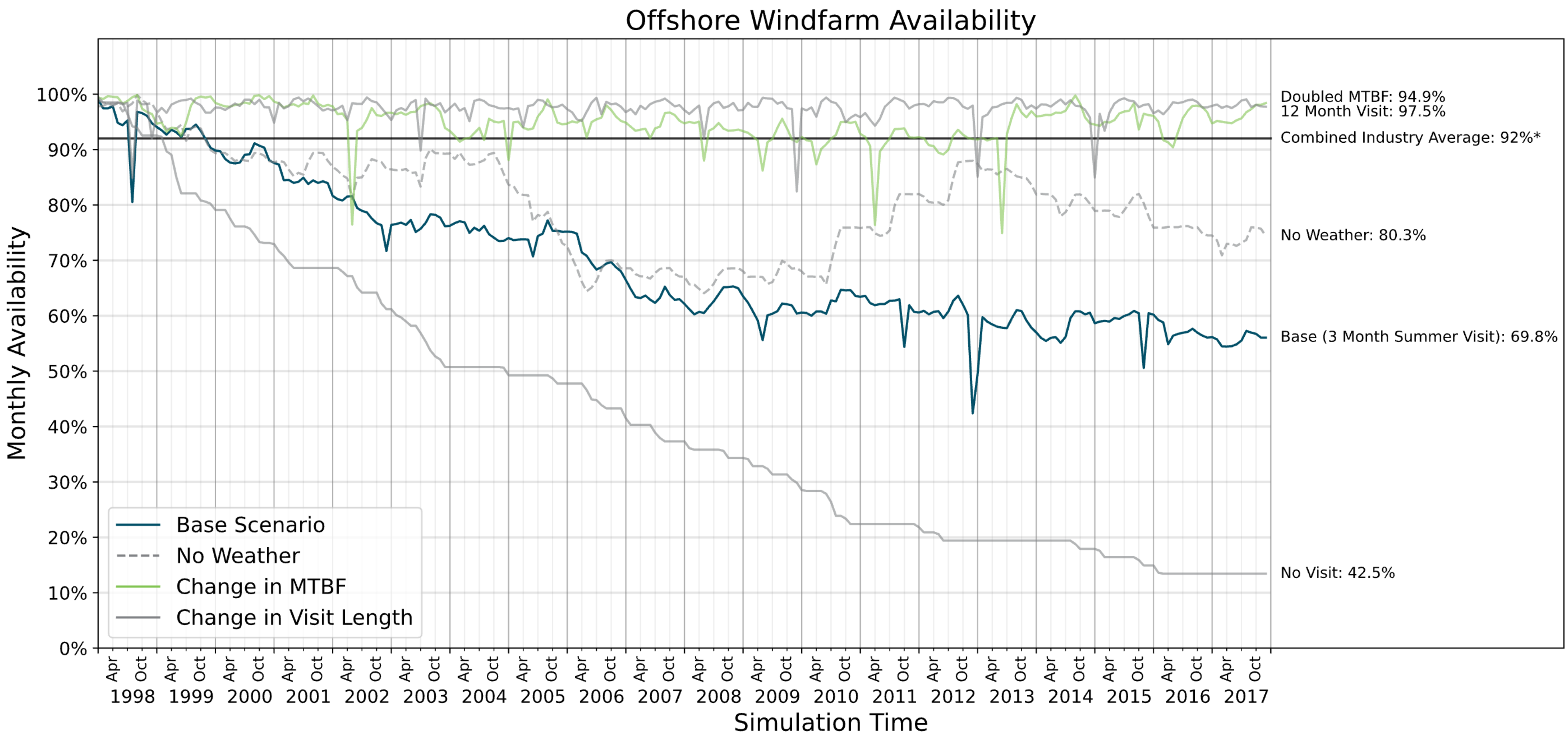
```python
opex_breakdown = pd.concat(
    [
        sim_in_situ.metrics.equipment_costs("project"),
        sim_in_situ.metrics.labor_costs("project"),
        sim_in_situ.metrics.project_fixed_costs("project", "low"),
        sim_in_situ.metrics.port_fees("project"),
    ],
    axis=1
)
opex_breakdown.index = ["In Situ"]
opex_breakdown.loc["Tow-to-Port"] = pd.concat(
    [
        sim_ttp.metrics.equipment_costs("project"),
        sim_ttp.metrics.labor_costs("project"),
        sim_ttp.metrics.project_fixed_costs("project", "low"),
        sim_ttp.metrics.port_fees("project"),
    ],
    axis=1
).values[0]
opex_breakdown.style.format("{:,.2f}")
```

	equipment_cost	total_labor_cost	operations	port_fees
In Situ	2,274,924,948.23	48,935,360.44	0.00	0.00
Tow-to-Port	1,027,098,070.73	72,228,657.83	0.00	480,000,000.00

```python
process_times = sim_ttp.metrics.process_times()

# Normalize the times for hours per failure, to understand the average waiting and repair time
time_columns = ["time_to_completion", "process_time", "downtime", "time_to_start"]
process_times.loc[:, time_columns] = process_times[time_columns].values / process_times.N.values.reshape(-1, 1)

# Sort and make it look nice
process_times.sort_values("time_to_completion", ascending=False).style.format("{:,.0f}")
```

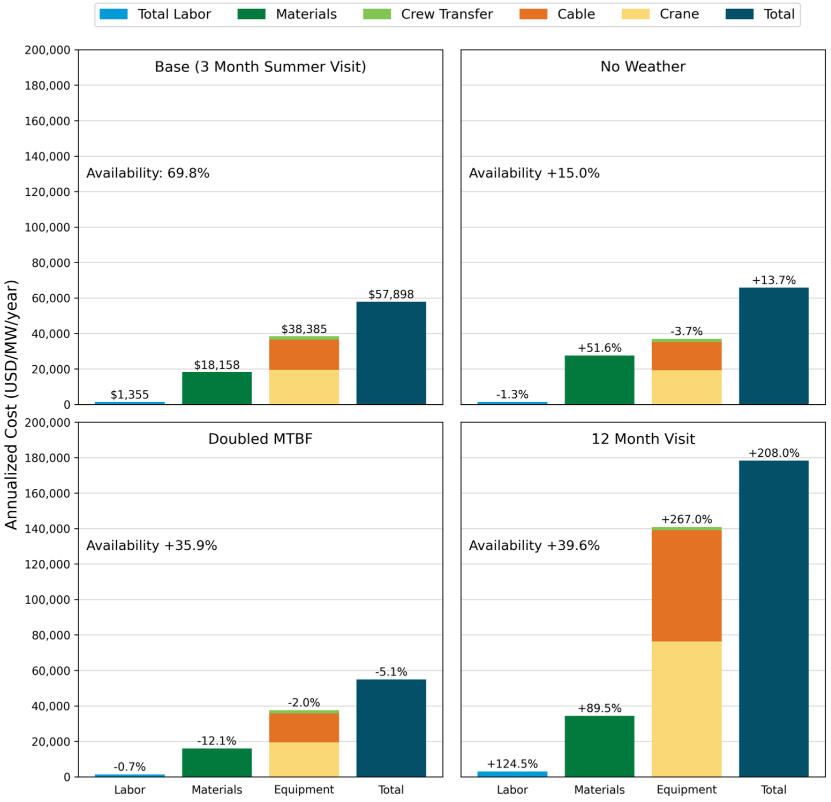
category	time_to_completion	process_time	downtime	time_to_start	N
yaw system major replacement	20,250	110	110	9,882	2
minor ballast pump repair	16,370	23	81	13,933	30
direct drive generator major repair	15,323	49	709	12,760	151
direct drive generator major replacement	14,774	190	202	11,442	35
annual turbine inspection	12,666	34	148	10,243	4,765
main shaft major repair	12,637	33	1,770	11,289	59
main shaft replacement	12,269	132	273	11,039	25
direct drive generator minor repair	11,877	23	55	9,591	2,732
yaw system major repair	10,833	40	43	10,793	12

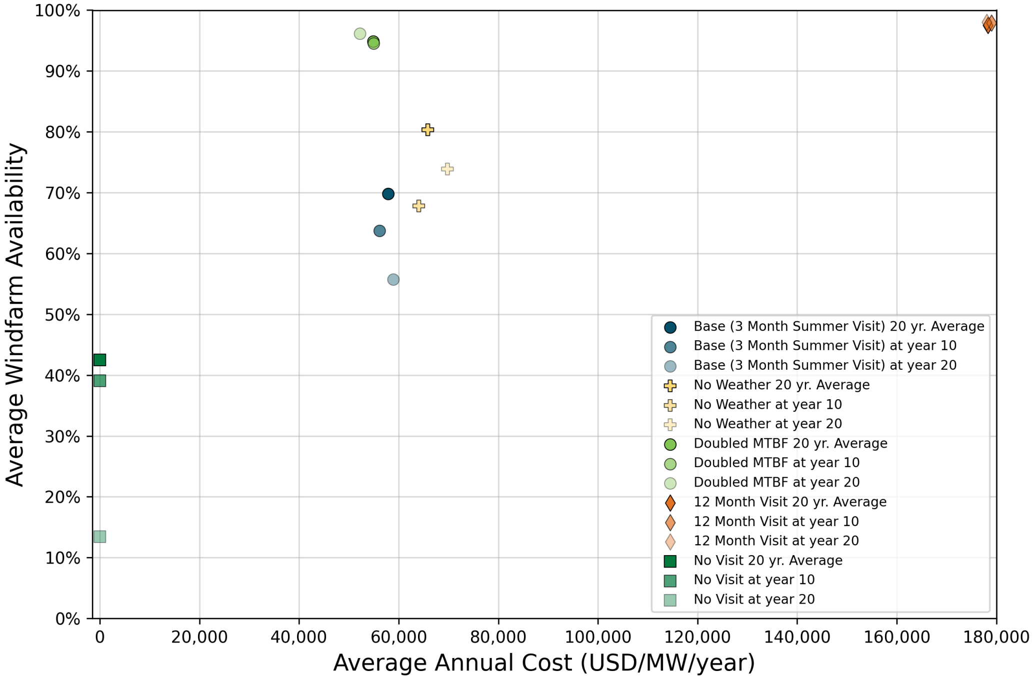
Comparing Many Scenarios



* Pfaffel, S., S. Faulstich, and K. Rohrig. "Performance and Reliability of Wind Turbines: A Review." *Energies*, 10(11), 2017.

Relationships Between Cost & Availability





Future of WOMBAT

What’s on the horizon for WOMBAT
over the next few years?

Potential Future Work

Model Development

- Model tow-to-port simulations at a higher resolution
- Support Operation Vessels (SOV)
- Date-based maintenance
- Single file/dictionary configuration like ORBIT and FLORIS
- Multicrew handoff
- Other ideas? Add them to the Issues board: https://github.com/WISDEM/WOMBAT/issues

Validation and Review

- Create a land-based example
- Develop a baseline land-based, fixed offshore, and floating offshore wind data set for easier modeling
- Publish the results of a model comparison exercise

WAVES

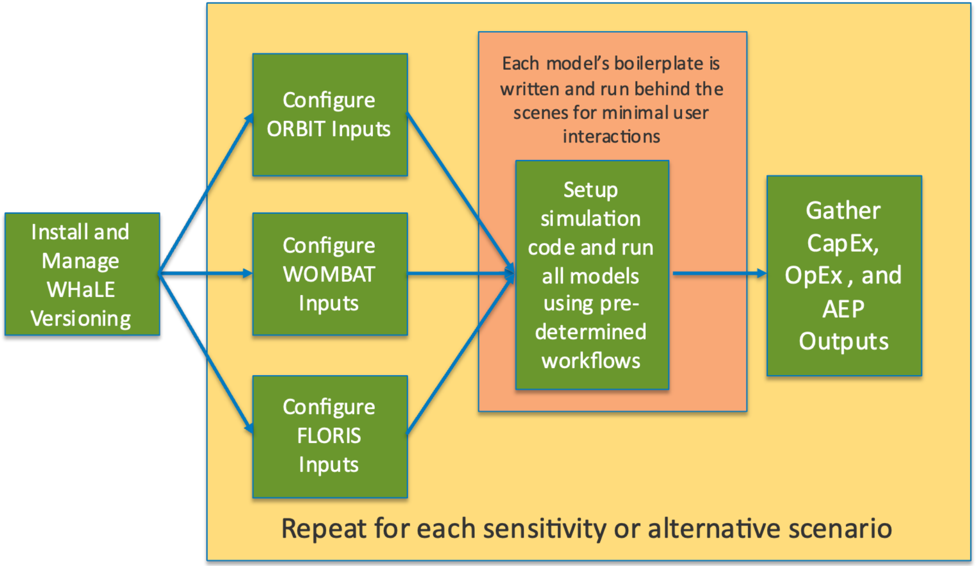
Rob Hammond

At a Glance

Thesis: How might maintenance strategies, technological innovations, and site conditions influence wind power plant operational expenditures (OpEx) and, ultimately, levelized cost of energy (LCOE)?

Under the hood:

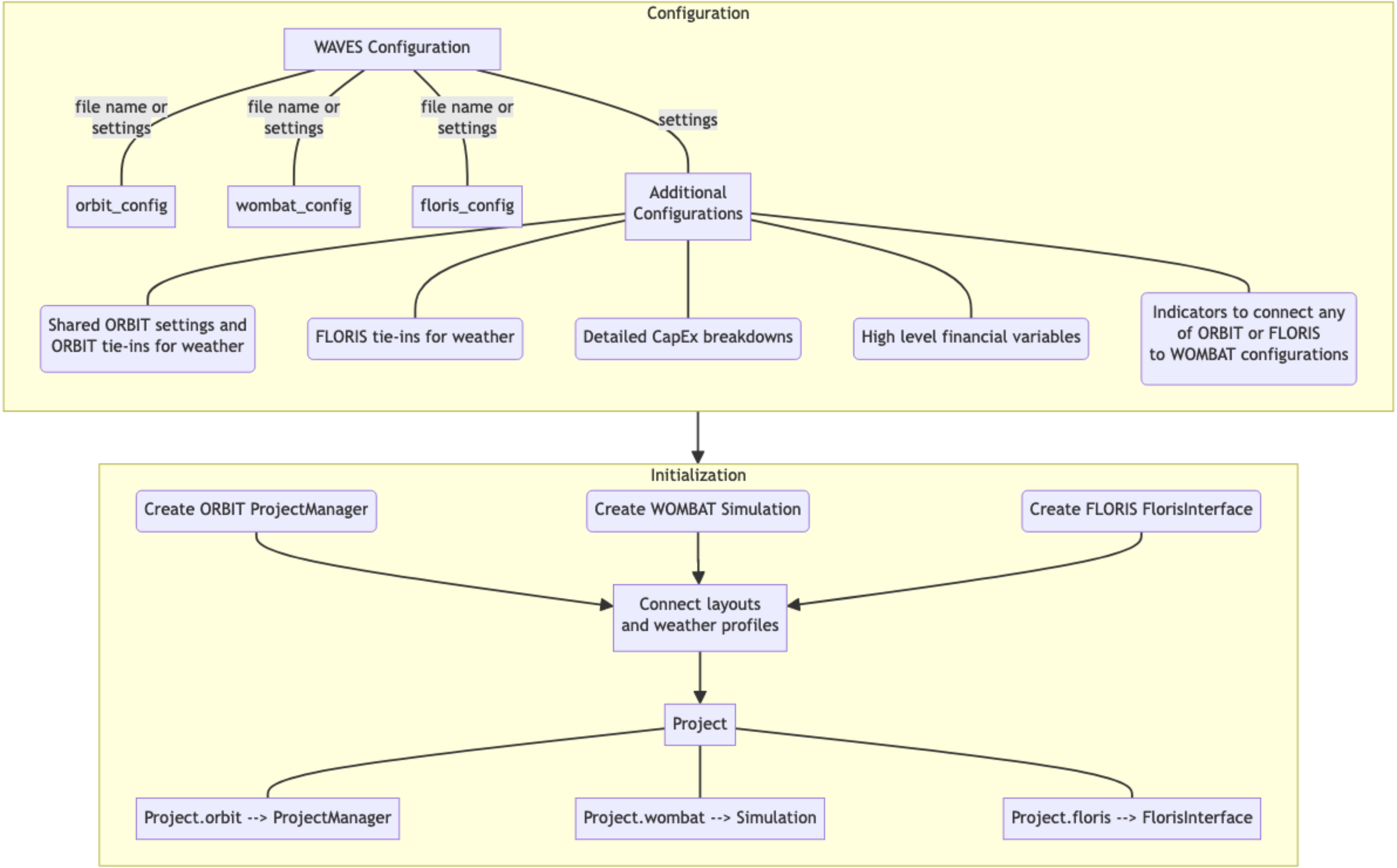
- Thin wrapper for the offshore wind modeling libraries to get a holistic look at LCOE
- Provides the key missing connections between ORBIT, WOMBAT, and FLORIS inputs and outputs that would pose a significant burden to analysts



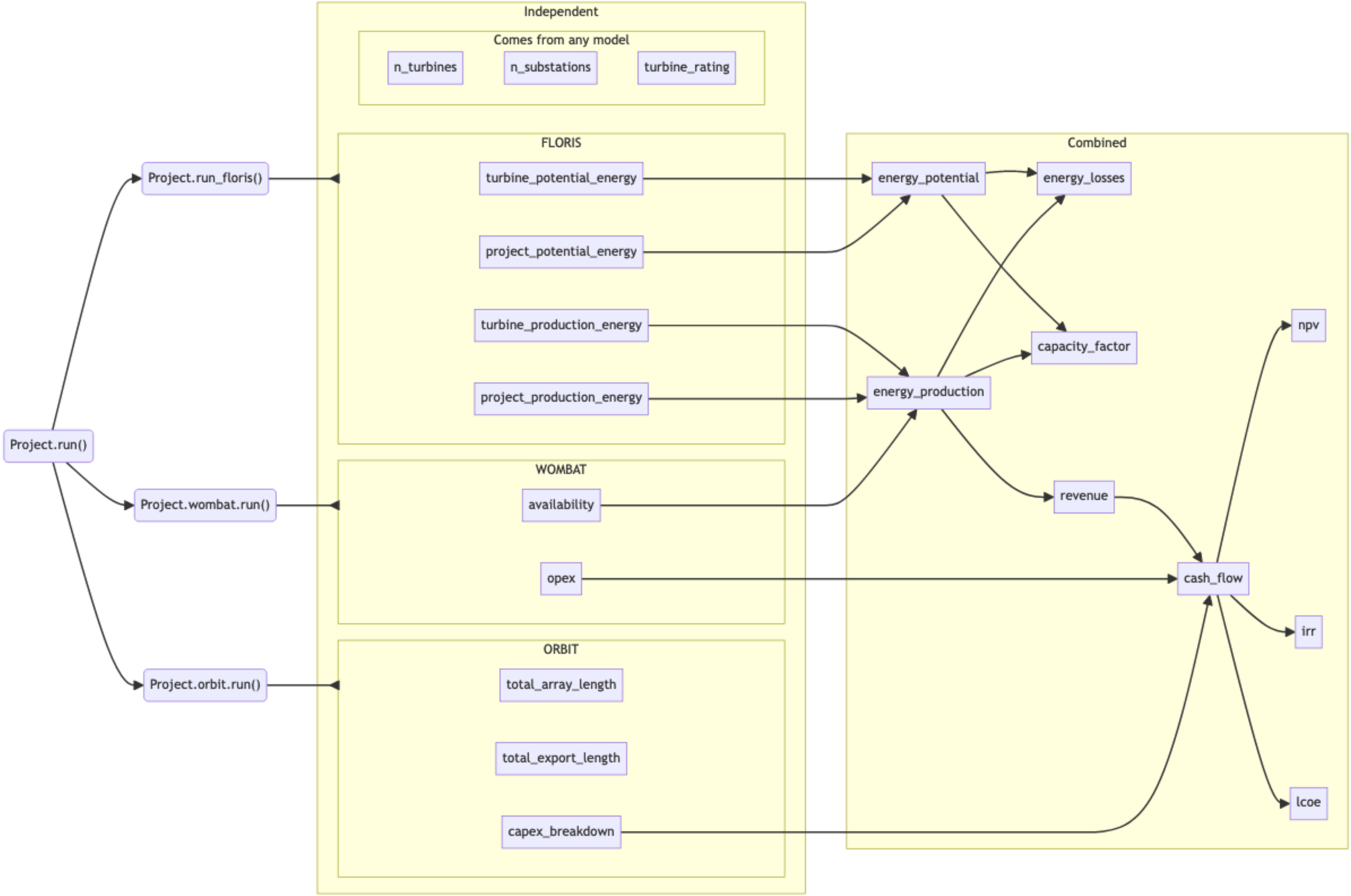
https://github.com/NREL/WAVES

https://nrel.github.io/WAVES/

Input Model


Configuration
WAVES Configuration
file name or settings
file name or settings
file name or settings
settings
orbit_config
wombat_config
floris_config
Additional Configurations
Shared ORBIT settings and ORBIT tie-ins for weather
FLORIS tie-ins for weather
Detailed CapEx breakdowns
High level financial variables
Indicators to connect any of ORBIT or FLORIS to WOMBAT configurations
Initialization
Create ORBIT ProjectManager
Create WOMBAT Simulation
Create FLORIS FlorisInterface
Connect layouts and weather profiles
Project
Project.orbit --> ProjectManager
Project.wombat --> Simulation
Project.floris --> FlorisInterface

Results Model



Setting Up a Scenario

base_fixed_bottom_2022.yaml 9+ ×

library > base_2022 > project > config > base_fixed_bottom_2022.yaml > ...

Rob Hammond, 6 months ago | 1 author (Rob Hammond)

```yaml
# Primary model configurations
orbit_config: base_fixed_bottom_2022_install.yaml
wombat_config: base_fixed_bottom_2022_operations.yaml
floris_config: base_fixed_bottom_2022_floris_jensen.yaml
weather_profile: era5_40.0N_72.5W_1990_2020.csv

# Shared input connections
orbit_start_date: 1/1/1998
orbit_weather_cols:
- windspeed_100m
- windspeed_10m
- waveheight
floris_wind_direction: wind_direction_100m
floris_windspeed: windspeed_100m
floris_x_col: floris_x
floris_y_col: floris_y

# Create the necessary connections
# NOTE: these are default values, but worth highlighting for an example
connect_floris_to_layout: true
conenct_orbit_array_design: true
```

base_fixed_bottom_2022.yaml 9+ ×

library > base_2022 > project > config > base_fixed_bottom_2022.yaml > ...

```yaml
# High-level project financials
discount_rate: 0.025
fixed_charge_rate: 0.0648 # real FCR from national LCOE study, 25 y
loss_ratio: 0.1
offtake_price: 83.30

# Cash flow settings
project_capex_date:
- !!python/tuple
  - 1996
  - 1
- !!python/tuple
  - 1996
  - 7
- !!python/tuple
  - 1997
  - 1
- !!python/tuple
  - 1997
  - 7
soft_capex_date: !!python/tuple ...
system_capex_date: ...
turbine_capex_date: ...
```

https://github.com/NREL/WAVES/blob/main/library/base_2022/project/config/base_fixed_bottom_2022.yaml

Running WAVES

> waves library/base_2022 base_fixed_bottom_2022.yaml

base_fixed_bottom_2022.yaml 9+ ×

library > base_2022 > project > config > base_fixed_bottom_2022.yaml > {} repor

```yaml

# CLI Arguments
run:
  which_floris: wind_rose  # month-based wind rose wake analysis
  full_wind_rose: False  # use the WOMBAT date range
  floris_reinitialize_kwargs:
    cut_in_wind_speed: 3.0
    cut_out_wind_speed: 25.0  # standard ws range
report_config:
  name: Base Fixed Bottom 2022
  "# Turbines":
    metric: n_turbines
  Turbine Rating (MW):
    metric: turbine_rating
  Project Capacity (MW):
    metric: capacity        Rob Hammond, 6 months ago • First Releas
    kwargs:
      units: mw
```

[1]:
```python
from pathlib import Path
from waves import Project
from waves.utilities import load_yaml
```

[2]:
```python
# Handle lingering path issues for FLORIS turbine library
library_path = Path("../library/base_2022/")
config = load_yaml(library_path / "project/config", "base_floating_2022.yaml")

config.update({"library_path": library_path,})
config["floris_config"] = load_yaml(library_path / "project/config", config["floris_config"])
config["floris_config"]["farm"]["turbine_library_path"] = library_path / "turbines"
```

[3]:
```python
project = Project.from_dict(config)
```

FutureWarning: /Users/rhammond/GitHub_Public/WAVES/waves/project.py:114 •••

[4]:
```python
project.run(
    which_floris="wind_rose",  # month-based wind rose wake analysis
    full_wind_rose=False,  # use the WOMBAT date range
    floris_reinitialize_kwargs={"cut_in_wind_speed": 3.0, "cut_out_wind_speed": 25.0}  # standard ws range
)
```

UserWarning: /Users/rhammond/GitHub_Public/ORBIT/ORBIT/phases/design/array_system_design.py:906 •••

https://github.com/NREL/WAVES/blob/main/library/base_2022/project/config/base_fixed_bottom_2022.yaml

Inspecting Results

```
project_fixed.cash_flow("annual", breakdown=True).T
```

year	1995	1996	1997	1998	1999	2000
CapEx_Soft	-325,896,000.00	0.00	0.00	0.00	0.00	0.00
CapEx_Project	0.00	-75,625,000.00	-75,625,000.00	0.00	0.00	0.00
CapEx_Turbine	0.00	-255,000,000.00	-255,000,000.00	-255,000,000.00	-255,000,000.00	0.00
CapEx_ArrayCableSystem	0.00	-27,798,308.93	-27,798,308.93	-27,798,308.93	-27,798,308.93	0.00
CapEx_ExportCableSystem	0.00	-41,914,140.00	-41,914,140.00	-41,914,140.00	-41,914,140.00	0.00
CapEx_OffshoreSubstationSystem	0.00	-22,909,293.75	-22,909,293.75	-22,909,293.75	-22,909,293.75	0.00
CapEx_ScourProtectionSystem	0.00	-2,560,500.00	-2,560,500.00	-2,560,500.00	-2,560,500.00	0.00
CapEx_MonopileSystem	0.00	-123,603,041.22	-123,603,041.22	-123,603,041.22	-123,603,041.22	0.00
CapEx_ArrayCableInstallation	0.00	0.00	0.00	-129,854,883.03	0.00	0.00
CapEx_ExportCableInstallation	0.00	0.00	0.00	-31,809,412.69	0.00	0.00
CapEx_MonopileInstallation	0.00	0.00	0.00	-44,655,354.55	0.00	0.00
CapEx_OffshoreSubstationInstallation	0.00	0.00	0.00	-5,540,460.12	0.00	0.00
CapEx_ScourProtectionInstallation	0.00	0.00	0.00	-44,131,310.50	0.00	0.00
CapEx_TurbineInstallation	0.00	0.00	0.00	-58,007,701.61	0.00	0.00
OpEx	0.00	0.00	0.00	0.00	0.00	-37,767,273.29
Revenue	0.00	0.00	0.00	0.00	0.00	209,106,520.29
cash_flow	-325,896,000.00	-549,410,283.90	-549,410,283.90	-787,784,406.41	-473,785,283.90	171,339,247.00

Inspecting Results

```python
# Capture the CapEx breakdown from each scenario
df_capex_fixed = pd.DataFrame(
    project_fixed.orbit.capex_breakdown.items(),
    columns=["Component", "CapEx ($) - Fixed"]
)
df_capex_floating = pd.DataFrame(
    project_floating.orbit.capex_breakdown.items(),
    columns=["Component", "CapEx ($) - Floating"]
)

# Compute the normalized CapEx for each scenario
df_capex_fixed["CapEx ($/kW) - Fixed"] = df_capex_fixed["CapEx ($) - Fixed"] / project_fixed.capacity("kw")
df_capex_floating["CapEx ($/kW) - Floating"] = df_capex_floating["CapEx ($) - Floating"] / project_floating.capacity("kw")

# Combine the results into one, easy to view dataframe
df_capex = df_capex_fixed.merge(
    df_capex_floating,
    on="Component",
    how="outer",
).fillna(0.0).set_index("Component")
df_capex = df_capex.iloc[pd.Categorical(df_capex.index, capex_order).argsort()]
df_capex
```

Component	CapEx ($) - Fixed	CapEx ($/kW) - Fixed	CapEx ($) - Floating	CapEx ($/kW) - Floating
Array System	111,193,235.71	185.32	133,234,144.17	222.06
Export System	100,357,800.00	167.26	75,794,538.61	126.32
Offshore Substation	99,479,100.00	165.80	99,479,100.00	165.80
Substructure	307,153,308.59	511.92	630,709,636.60	1,051.18
Scour Protection	10,242,000.00	17.07	0.00	0.00
Mooring System	0.00	0.00	275,612,740.33	459.35
Turbine	1,020,000,000.00	1,700.00	1,020,000,000.00	1,700.00
Array System Installation	108,761,352.72	181.27	167,028,845.71	278.38
Export System Installation	135,777,423.05	226.30	144,141,926.89	240.24
Offshore Substation Installation	7,424,892.50	12.37	15,152,770.85	25.25
Substructure Installation	44,655,354.55	74.43	88,975,886.55	148.29
Scour Protection Installation	44,131,310.50	73.55	0.00	0.00
Mooring System Installation	0.00	0.00	69,384,372.60	115.64
Turbine Installation	58,007,701.61	96.68	0.00	0.00
Soft	325,896,000.00	543.16	325,896,000.00	543.16
Project	151,250,000.00	252.08	151,250,000.00	252.08

```python
report_df_fixed = project_fixed.generate_report(metrics_configuration, project_name_fixed).T
...
report_df
```

Metrics	COE 2022 - Fixed	COE 2022 - Floating
# Turbines	50.00	50.00
Turbine Rating (MW)	12.00	12.00
Project Capacity (MW)	600.00	600.00
# OSS	1.00	1.00
Total Export Cable Length (km)	118.07	89.17
Total Array Cable Length (km)	277.98	333.09
FCR (%)	6.48	6.48
Offtake Price ($/MWh)	83.30	83.30
CapEx ($)	2,524,329,479.24	3,196,659,962.31
CapEx per kW ($/kW)	4,207.22	5,327.77
OpEx ($)	1,060,154,836.28	972,561,379.04
OpEx per kW ($/kW)	1,766.92	1,620.94
Annual OpEx per kW ($/kW)	84.14	77.19
Energy Availability (%)	94.58	90.34
Gross Capacity Factor (%)	52.91	59.35
Net Capacity Factor With Wake Losses (%)	46.75	51.04
Net Capacity Factor With All Losses (%)	41.46	45.11
AEP (MWh)	2,181,019.94	2,372,504.25
AEP per kW (MWh/kW)	3.64	3.95
LCOE ($/MWh)	98.15	106.83

https://github.com/NREL/WAVES/blob/main/examples/waves_example.ipynb

Future Work

- Adopt FLORIS v4.0
- Incorporate LandBOSSE for land-based CapEx
- Parameter sweeps to better inform decision making
- Other ideas? Add them to the Issues board: https://github.com/NREL/WAVES/issues

NRWAL

Owen Roberts

Roadmap

Rob Hammond

TEA Software Roadmap

Short Term

- Repository cleanup
- Documentation improvements (README and docs sites)
- Ensure examples are informative and useful
- PyPI listings for easy “pip install xx”

Long Term

- Link up adjacent models that work well together (like WAVES did)
- Inflate costs and document sources adequately
- Host more workshops or user meetings (if helpful)
- Identify new and relevant pathways (hybrid plants, hydrogen, new fixed/floating/vessel technologies)

Technoeconomic Analysis and Cost Models

Polls

Open Discussion

TEA & Cost Models

Raise your hand and we'll call your name to ask a question.

- **Discussion topics**
  - Prospective / new users:
    - What are your thoughts on the learning or onboarding process?
  - Experienced users:
    - What have been your primary pain points or bottlenecks?
    - What has worked or not worked in helping to integrate these software into your workflows?
    - How thoroughly do you understand the capability of these tools?
    - What has helped or hindered your open-source contribution to these software?

Thank you for your time today!

- Need help with a particular problem?
  - GitHub Issues or Discussions pages for any of the models
  - NREL User Forum (for NREL models): forums.nrel.gov

- Have further thoughts that you want to share? Send feedback to Rafael.Mudafort@nrel.gov
- How could we have done better? Send feedback to Rafael.Mudafort@nrel.gov
- Software repositories:
  - LandBOSSE: https://github.com/WISDEM/LandBosse
  - ORBIT: https://github.com/WISDEM/ORBIT
  - CORAL: https://github.com/NREL/CORAL
  - WOMBAT: https://github.com/WISDEM/WOMBAT
  - WAVES: https://github.com/NREL/WAVES
  - NRWAL: https://github.com/NREL/NRWAL