



Engineering features summary

Ashes

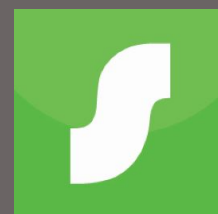
Report number 2023-1027

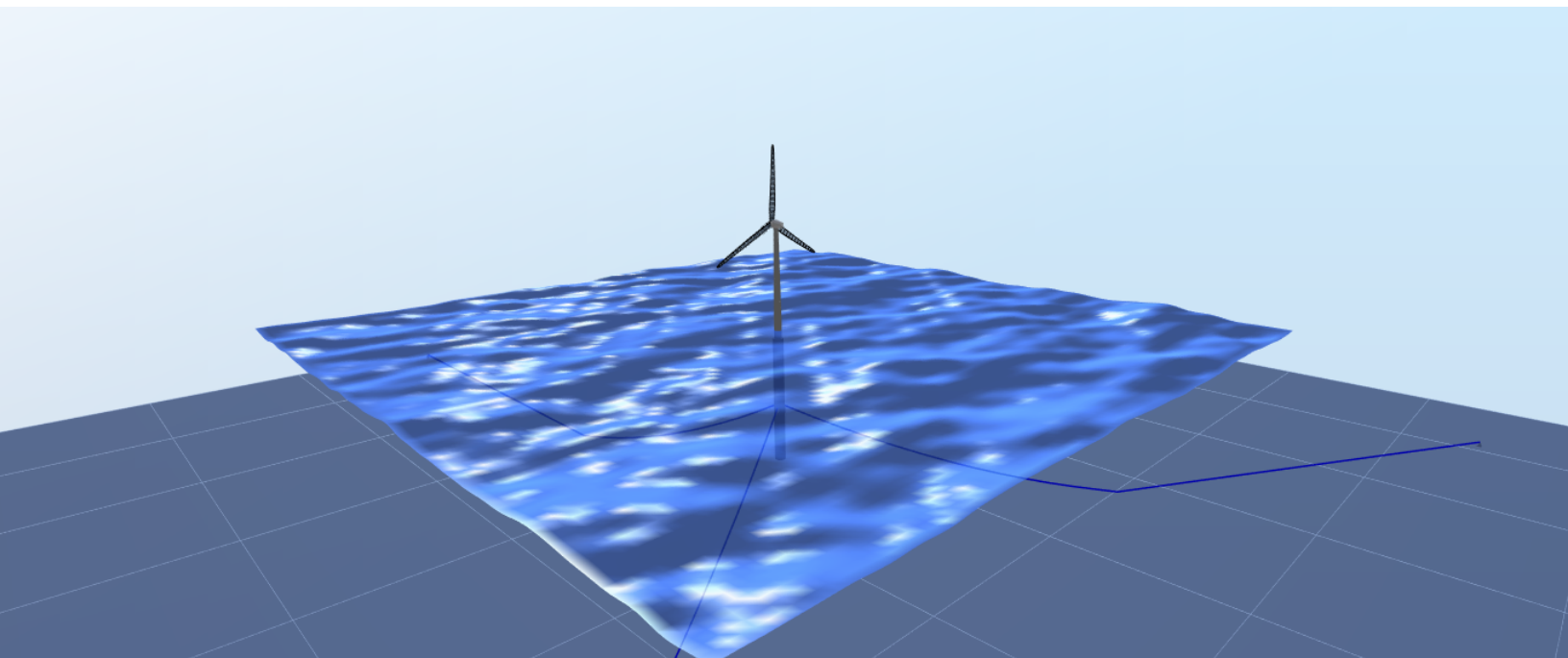
Written by Loup Suja-Thauvin

Reviewed by Paul Thomassen

Simis AS

18/07/2024





Contents

1 Introduction	2
2 Verification and Validation	3
3 Structural modeling	4
4 Fatigue analysis	6
5 Aerodynamic modeling	7
6 Vertical axis wind turbines	9
7 Hydrodynamic loads	10
8 Soil-structure interaction	12
9 Control systems	13
10 Environmental conditions	14
10.1 Marine environment	14
10.2 Wind	16
11 Parallelization and cloud computing	17
12 Command Line Interface	18



1 Introduction

Ashes is an aeroelastic software that performs highly accurate dynamic analyses of onshore, offshore bottom-fixed and offshore floating wind turbines thanks to state-of-the-art engineering models. Ashes has been used by wind turbine designers, researchers and academics for more than 10 years, and is continually being enhanced to ensure that it meets the requirements of wind turbine engineers over the world.

This document provides an overview of some of the engineering models implemented in Ashes. For a more detailed description, each section has a link to our online [Theory manual](#), where the equations are shown and the theory is explained.

If you have any question, contact us at support@simis.io. You can also download and try Ashes for free at www.simis.io.

2 Verification and Validation

Making sure that the results produced by Ashes can be trusted is of the utmost importance. In order to ensure that the implementation of new features does not introduce errors in the output from Ashes, a set of **thousands of tests** is run every night. If any of the tests fails, our engineering team is notified automatically.

For every new feature, a new test is implemented. A report with the test results and a detailed description of the benchmarks is published at each new Ashes release on [this link](#).

Benchmark MacNeal Twisted Beam	2 of 2 passed
Comparison between results produced by Ashes and results published by MacNeal (1985) See detailed description	
In-plane shear	1 of 1 passed
Out-of-plane shear	1 of 1 passed
View full report (.pdf)	
Decay test compared to analytical solution	20 of 20 passed
Decay test in Ashes compared to analytical solution. See detailed description	
Decay no damping	2 of 2 passed
Mass only	2 of 2 passed
Stiffness only	2 of 2 passed
Mass only 2nd mode	2 of 2 passed
Stiffness only 2nd mode	2 of 2 passed
Rayleigh first mode	2 of 2 passed
Rayleigh second mode	2 of 2 passed
Mass from coefficients	2 of 2 passed
Stiffness from coefficients	2 of 2 passed
Rayleigh from coefficients	2 of 2 passed
View full report (.pdf)	
DTU 10-MW steady state performance	132 of 132 passed
Comparison between aerodynamic performance of the DTU 10-MW turbine as published in the specification document and Ashes See detailed description	
4 ms wind speed	6 of 6 passed

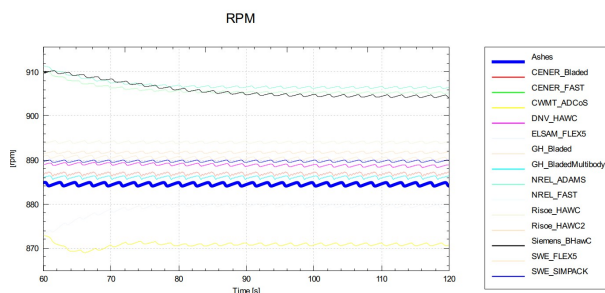
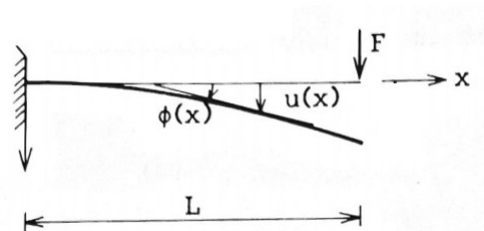
Published benchmarks for wind turbines

For several tests, models corresponding to three well-known reference wind turbines, namely the [NREL 5-MW](#), the [DTU 10-MW](#) and the [IEA 15-MW](#) are generated. The results produced by Ashes are compared to the specification documents, Jonkman et al. [2009], Bak et al. [2013] and Gaertner et al. [2020], respectively.

In addition, results from Ashes are benchmarked against the **Offshore Code Comparison Collaboration** projects, in which several aeroelastic codes use the same wind turbine specifications to generate a model and compare their results. This way we ensure that the results from Ashes match those from other aeroelastic software.

Analytical solutions

A large set of tests compares results from Ashes to **analytical solutions**. These tests generally use very simple models such as a cantilever beam or a cylindrical blade, and the results from Ashes are expected to match the analytical solution within 0.1-0.01%.



Internal comparisons

Simulations where **every parameter in Ashes** has been changed from its default value are run every night, and the results are compared to those produced the night before. If no implementation is expected to modify the results, the comparison should produce an exact match between the two sets.

3 Structural modeling

Ashes uses the **finite element method** (FEM) in conjunction with a co-rotational formulation of beam elements to determine the dynamic response of any structure. External loads (such as aerodynamic or hydrodynamic loads for example) are applied at the nodes, and the response of the structure is computed at each time step.

Co-rotational elements

The **co-rotational formulation** leads to accurate results in arbitrarily large displacements and rotations in space [Bruheim, 2012]. This is critical for wind turbines as they will by definition experience large rotation when in operation.

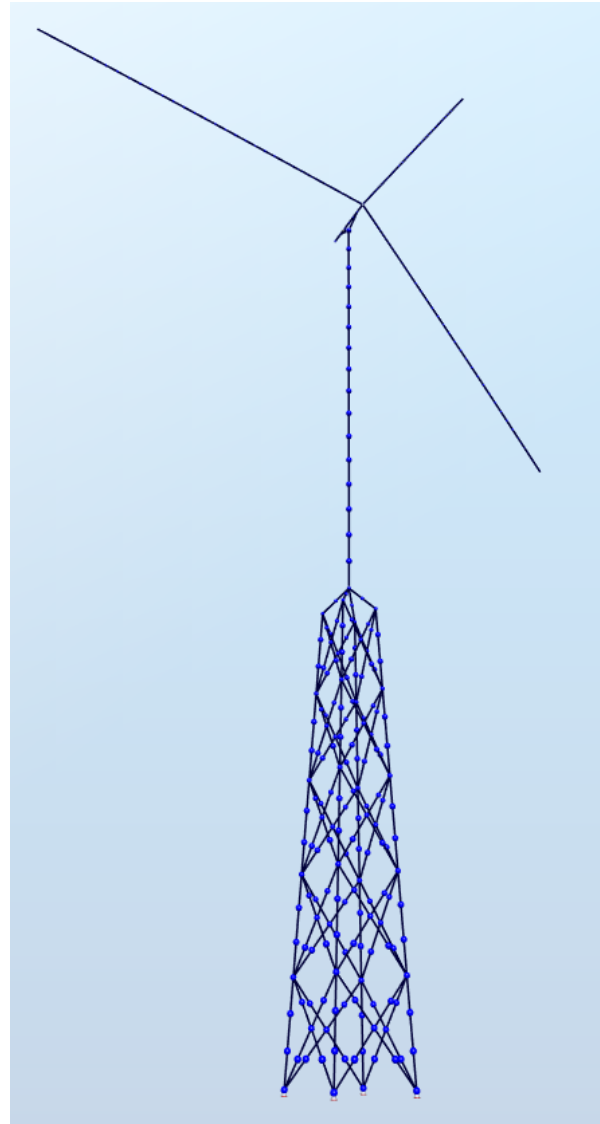
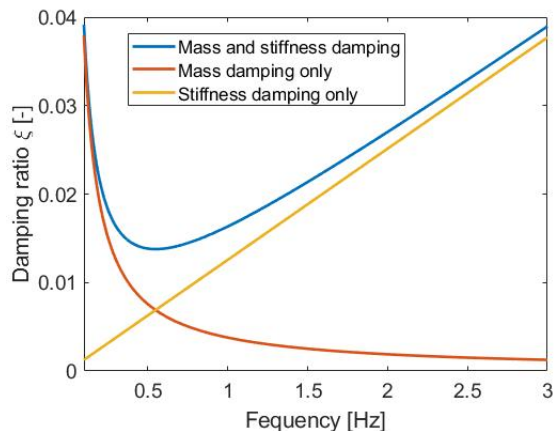
Two beam theories can be applied: the **Euler-Bernoulli** beam theory for slender structures, and the **Timoshenko** beam theory for thick beams.

Once the model is defined, the equations of motion are solved using the [Newmark-Beta](#) method.

Non-linear formulation

A linear solution of the equations of motion can be used in cases where nonlinear effects are unimportant, for example for structures that experience small deflections.

In the cases where **non-linear effects** become important, the [Newton-Raphson](#) iterative scheme is used. This implies that at each time step, the mass, stiffness and damping matrices, as well as the external loads are computed several times until the equation of motion is solved for.



Damping

[Rayleigh damping](#) is included in Ashes, with the possibility of using mass damping only, stiffness damping only or a combination of both. **Rayleigh damping** is commonly used to represent material damping.

[Numerical damping](#) is also implemented as part of the **Hilbert-Hughes-Taylor** (HHT) method (as described in A. Crisfield et al. [1997]). This is typically necessary to dampen out high frequency vibrations.

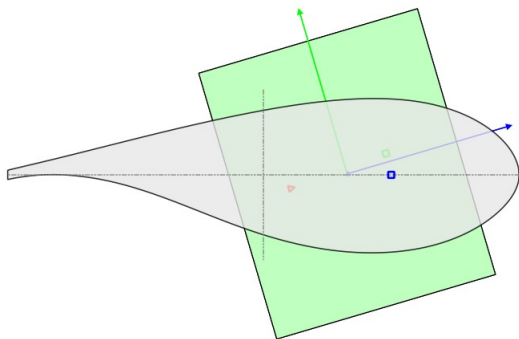
Additionally, **dashpots** can be added onto any node of the structure.

Modal analysis

Since the whole structure is modeled using the finite element approach, Ashes can calculate the **eigenfrequencies** and **mode shapes** of any structure with the **inverse iteration algorithm** [Bruheim, 2011].

This approach inherently includes any **coupling between the modes**. For instance, full coupling of blade bending and twist will be captured, during a modal analysis as well as in the time domain. This makes Ashes well-suited for analysis of soft blades.

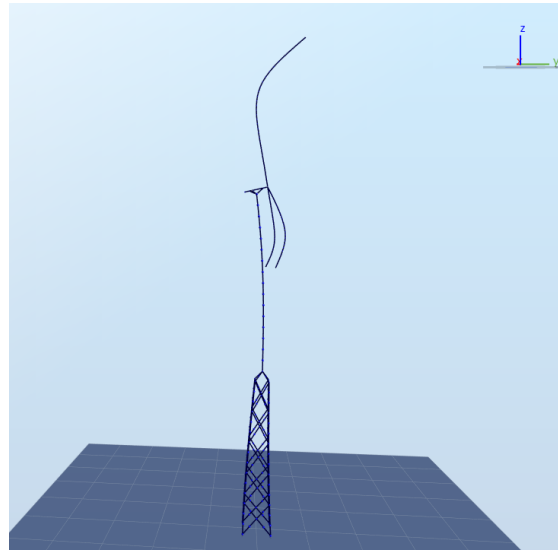
Ashes also provides the option of using **super-elements**: any part of the foundation of the turbine that has been simplified to a reduced number of mode shapes with the **Craig-Bampton substructuring method** [Klerk et al., 2008] can be imported.



Complex geometries

Since the whole structure is modeled as finite elements, any wind turbine that can be represented with beam elements can be modeled in Ashes, such as downwind configurations, monopiles, truss towers, spars or complex semi-submersibles.

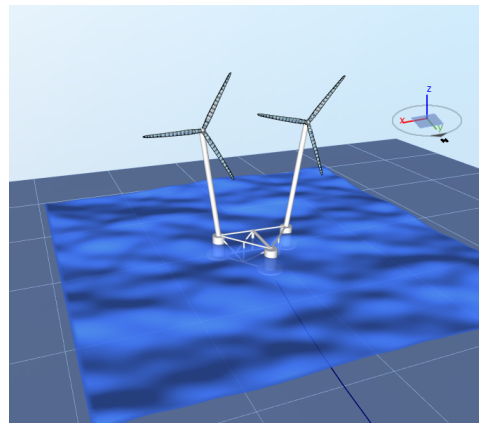
It is possible to add RNAs to any node of the structure which allows for easy generation of **multi-rotor turbines**. The rotors thus modeled are fully independent: they experience different incoming wind depending on their locations and have independent controllers.



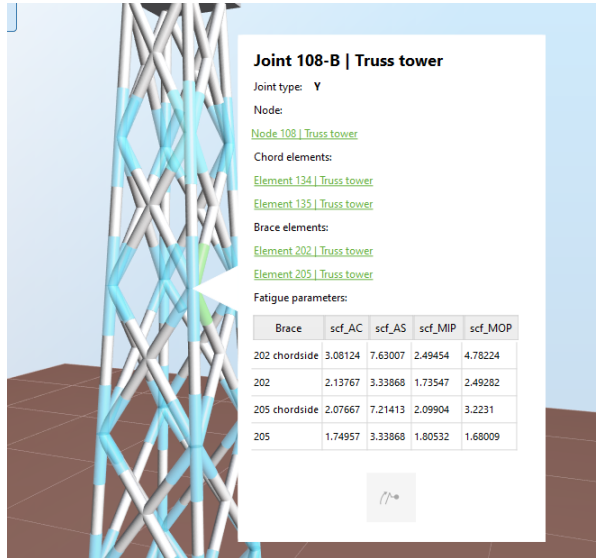
Advanced blade structural properties

Ashes gives the option to offset the center of mass, the elastic center and the shear center. This is key to accurately model the torsion of the blade, especially for large diameter wind turbines.

Ashes comes with some of the most commonly used reference blades, such as the [NREL 5-MW](#), the [DTU 10-MW](#) and the [IEA 15-MW](#), for which the structure has been verified against the corresponding specification documents.



4 Fatigue analysis



Fatigue life estimation

Ashes can perform **Rainflow counting** on stress time series, which will give a stress histogram of number of cycles for a given stress range. Based on the **SN curves** (also known as **Wöhler curves**) selected by the user (available from the Eurocode-EN-1993 [2005] and DNV-RP-C203 [2016] standards), a fatigue damage is computed for each of the stress ranges.

The **Palmgren-Miner** rule is then applied to compute the total fatigue damage for the stress time series.

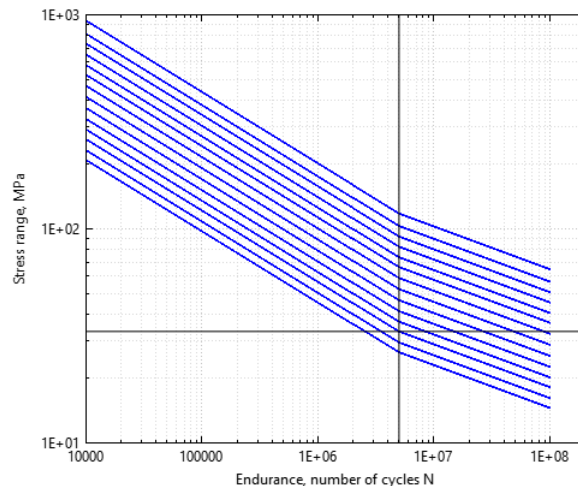
The stress time series will take into account the **Stress Concentration Factor** and stress filter. The user can specify how many points along the cross section should be checked for fatigue damage.

Automated fatigue report

Fatigue sensors on joints and the corresponding Stress Concentration Factors are automatically generated according to the specifications in DNV-RP-C203 [2016].

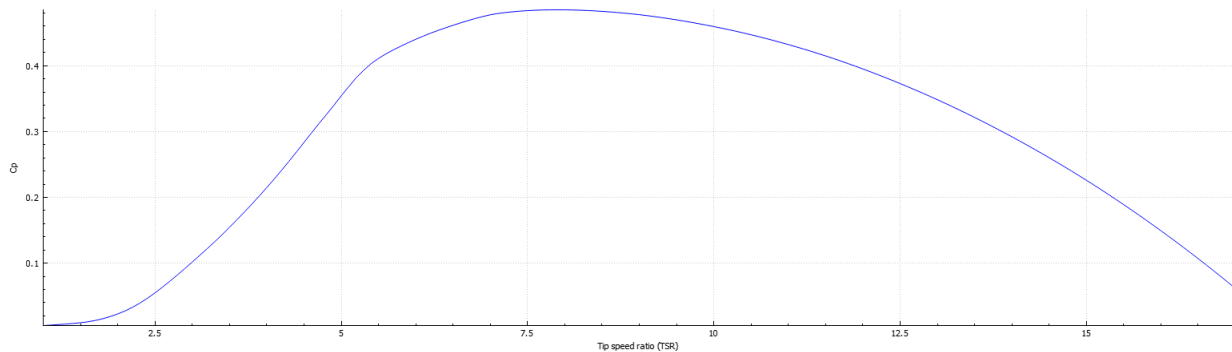
The **Design Load Case generator** can then be used to generate a batch based on either a **Wind rose**, a **scatter diagram** or a combination of both. The probability of occurrence of each environmental condition is then taken into account to compute the total fatigue life.

A fatigue report summing up the fatigue analysis is automatically generated in pdf at the end of each fatigue analysis.



5 Aerodynamic modeling

The aerodynamic loads in Ashes are calculated following the [Blade Element Momentum \(BEM\)](#) theory, as described in Hansen [2008]. This method is widely used in the industry to compute the **drag** and **lift** forces on elements of the blade, and our implementation in Ashes has been thoroughly verified and validated against measurements as well as other aeroelastic software (see for example Popko et al. [2018], Krogstad and Eriksen [2013])



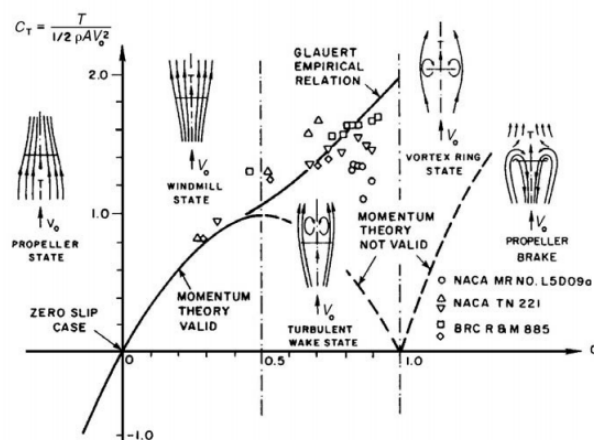
In addition, several corrections and extensions to the theory have been implemented to best account for the complex aerodynamic behavior of wind turbine blades, such as

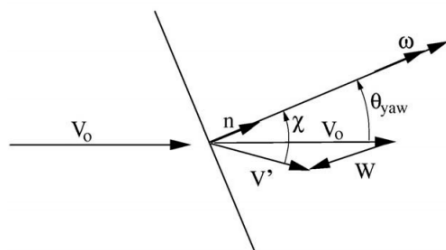
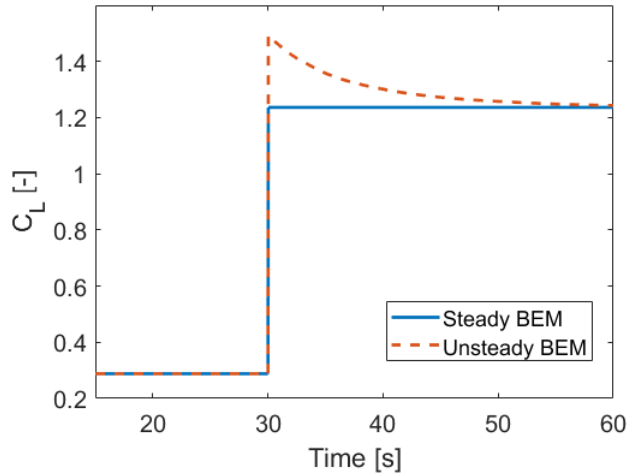
Heavily loaded rotors

The simple momentum theory used to derive the BEM breaks down for high induction factors, typically happening at low TSRs. [Glauert's correction](#) is an empirical method that ensures that under these conditions the theory will match experimental data.

Tip and hub losses

Ashes includes [Prandtl's correction](#) factor applied to the root and the tip of the blade to account for the BEM assumption of an infinite number of blades. This correction is derived from modeling the wake using vortex theory.





Dynamic wake model

To account for the time delay before equilibrium in the system is reached, a **dynamic wake model** is added. This is essentially a time filter for the induced velocities and is necessary to capture the time behavior of the aerodynamic loads when the incoming wind is not constant.

Dynamic stall model

Similarly, the effect of changing angles of attack on the blade do not affect the aerodynamic load instantaneously but rather experience a time delay. This is accounted for by the **dynamic stall model** described by Øye [1991] and implemented in Ashes.

Skewed rotor model

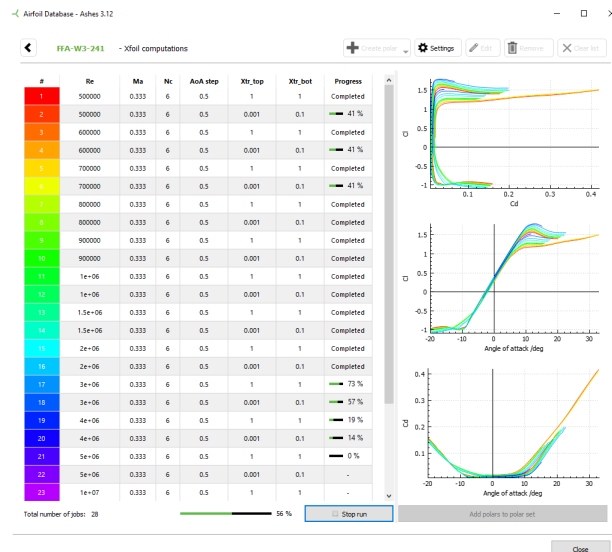
Whenever there is an azimuthal variation of the induced velocity (for example when the rotor is yawed or tilted), a **skewed rotor model** is necessary to correctly capture yawing moments on the rotor. Ashes includes the **Yaw/Tilt** model described in Snel et al. [1995].

Polar calculations

Ashes comes with **XFOil** [Drela, 2013], a program to design and analyse airfoils. Given the geometry of any airfoil and for your relevant Reynolds numbers, this will enable you to compute the **aerodynamic coefficients** that are used by the BEM to calculate the aerodynamic loads.

Viterna extrapolation

The aerodynamic coefficients for a given airfoil generally cover a reduced range of angles of attack, as for larger angles they are dependent on the blade characteristics as well. Ashes automatically applies the **Viterna method** to extrapolate the coefficients to larger angles of attack.

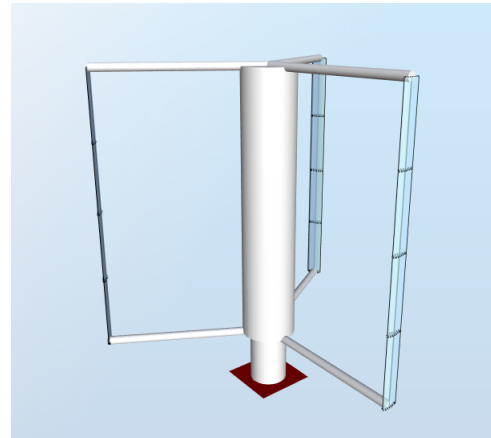


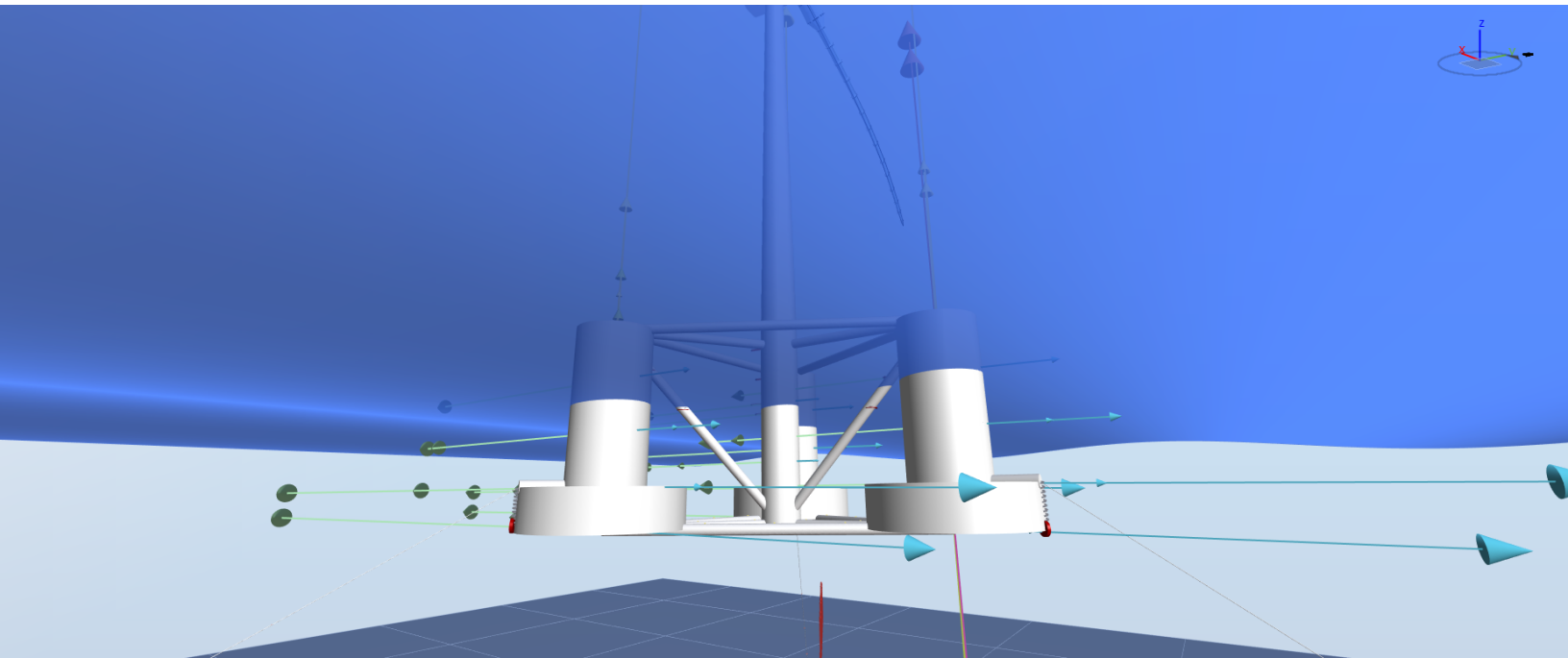
6 Vertical axis wind turbines

Aerodynamic loads on vertical axis wind turbines (VAWT) can be modelled using the [DMST algorithm](#) (i.e. Double Multiple StreamTube algorithm). This method allows for fast computation of steady aerodynamics on VAWT and accounts for the influence of the blades in the upstream half of the rotor on the downstream blades.

The struts are modelled as aerodynamic and structural elements with drag loads. Different configurations of struts can be simulated, such as struts on both ends or middle of the blades.

This allows structural analyses of VAWT, including support structure, blades and struts.





7 Hydrodynamic loads

Morison equation

Hydrodynamic loads on the elements defining the structure in Ashes are calculated with the well-known [Morison equation](#) (see for example Faltinsen [1990]). The finite element solver then computes the response of the structure to these loads, enabling to calculate shear forces, bending moments and stress distribution in e. g. floaters, truss towers or monopiles.

Potential flow theory

For large floaters, loads can be computed considering potential flow theory with the [Cummins equation](#). These loads will include linear and second order wave excitation loads (using Newman's approximation), added mass and radiation memory-effect force.

MacCamy-Fuchs correction

For large diameter bottom-fixed cylinders, the [MacCamy Fuchs](#) theory adds loads from the wave diffracted by the structure. Disregarding diffraction effect will produce too conservative loads.

Buoyancy loads

Two contributions of [buoyancy](#) are included in Ashes:

- stiffness contribution from the waterplane area
- variation in submergence of the structure

Buoyancy loads also take into account the inclination of the elements they apply to.

Python API

More advanced hydrodynamic models, including loads from breaking waves, can be included in Ashes with the [Application Programming Interface \(API\)](#) for custom made Python scripts.

The API gives you access to the position, velocity and acceleration of all nodes of the structure, as well as the characteristics of all elements. At each time step, loads can thus be calculated within the Python code and sent back to Ashes, with virtually no time-cost.

Super-element

Ashes also gives you the option to import **Craig-Bampton** super-elements to model the substructure of the turbine. These elements can be generated with an external software, together with the corresponding hydrodynamic loads. The finite element solver then includes the behavior of the super-element in the computation of the structural response.

```

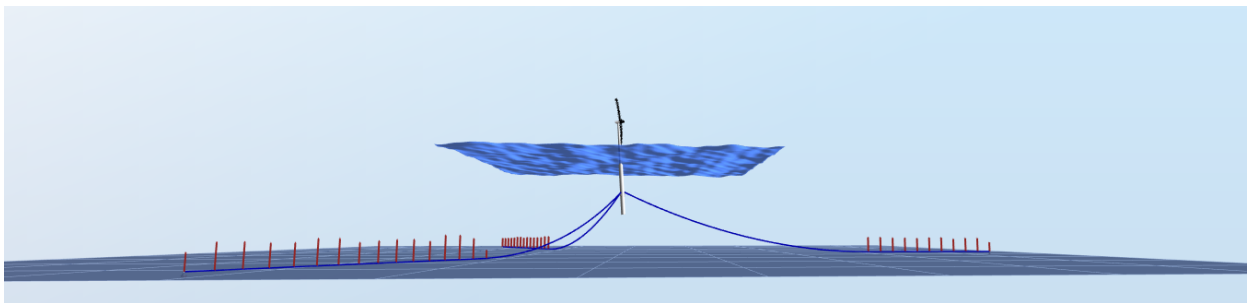
1 from Simis.Ashes import Ashes
2 import math
3
4 # this script calculates the loads from the Morison equation for regular waves in inifini
5 # it is meant to be used as a template for writing your own scripts
6 # loup, 2019-11-11
7
8 # Define a callback function in which we can modify the model given by Ashes
9 def update(model):
10     Ca = 2 # inertia coefficient
11     Cd = 1 # drag coefficient
12
13     H = 2 # wave height
14     T = 5 # wave period
15
16     g = 9.81 # acceleration of gravity
17     rho = 1025 # water density
18
19     D = 6 # pile diameter
20     dz = 5 # element length
21
22     w = 2*math.pi/T # wave frequency
23     k = math.pow(w,2)/g # wave number
24
25     t = model.time # get the time from the model
26
27     # this function calculates the horizontal wave particle acceleration, according to Se
28     def calc_particle_acc(t, z, x, H, w):
29         return math.pow(w,2)*H/2*math.exp(k*(z))*math.cos(w*t-k*x)
30
31     # this function calculates the horizontal wave particle velocity, according to Sea L
32     def calc_particle_vel(t, z, x, H, w):
33         return w*H/2*math.exp(k*(z))*math.sin(w*t-k*x)
34
35     #print(t) # prints the current time step
36
37     # loop through all children in the model
38     for key in model.child:
39         # only take action if the child is a node
40         if "Node" in key:
41
42             z_init = model.child[key].initialposition.z # vertical coordinate of the node
43
44             # we only apply loads to nodes under the sea level, i.e. only nodes which ini
45             if z_init<=0:
46                 x = model.child[key].initialposition.x - model.child[key].displacement.x
47                 z = model.child[key].initialposition.z - model.child[key].displacement.z
48                 u = model.child[key].velocity.x # current horizontal velocity of the node
49                 a = model.child[key].acceleration.x # current horizontal acceleration of

```

Mooring lines

By default, mooring lines in Ashes are modeled as [catenary lines](#) with finite elements, which enables to correctly capture their non-linear stiffness behavior, but they can also be modeled as linear or non-linear springs. Mooring line elements are also subjected to Morison loads, and the interaction with the seabed is accounted for by including

- seabed stiffness
- seabed shear stiffness
- seabed friction coefficient



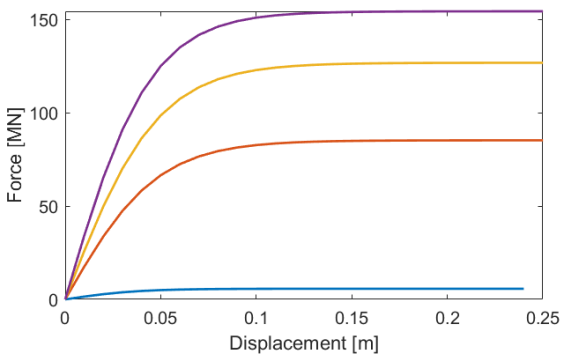
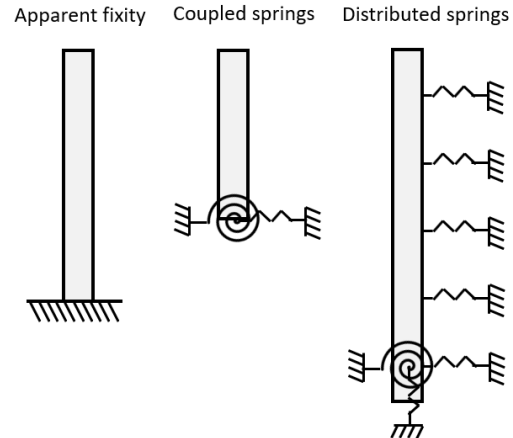
8 Soil-structure interaction

Simple geotechnical models

The simplest way to model a wind turbine foundation in Ashes is to constrain one of its nodes, thus allowing for an **apparent fixity** implementation.

It is also possible to apply **dashpots** or **linear springs** in all six degrees of freedom (i.e. translational or rotational springs) at one node of the structure.

Any combination of springs can be applied to several nodes along the structure to obtain a distributed stiffness model, where each spring has its own stiffness.



Non-linear P-y curves

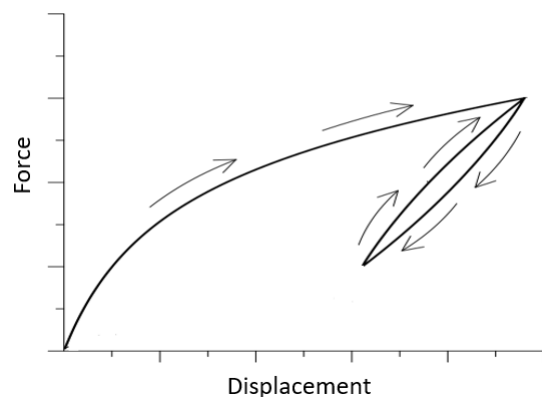
Non-linear springs can be applied to any node of the model. This enables to model the geotechnic-structural interaction with **P-y** curves as suggested in the API standards [API, 2011].

Non-linear springs can be applied in all three translational degrees of freedom and combined with linear springs, which enables the implementation of the **PISA** theory [Byrne et al., 2019]

Python API

More advanced soil-structure interaction models can be included in Ashes with the **API** for custom made Python scripts. This enables to calculate geotechnical loads depending on the kinematics of a node as well as its history.

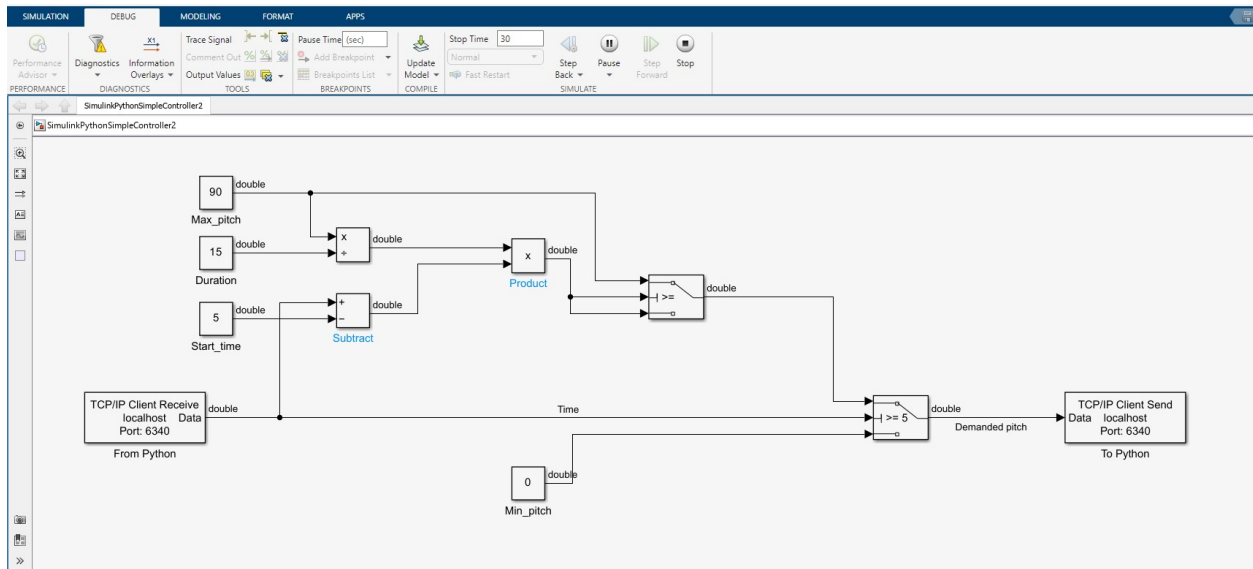
This can be used to compute loads based on the kinematics of a node, for example to include **hysteretic damping** for sequences of loading-unloading of the foundation.



9 Control systems

Parameterized controller

Ashes comes with an internal controller based on the **NREL 5-MW controller** [Jonkman et al., 2009]. The parameters of this controller can easily be changed to scale the control system to turbines of different rated powers. This is especially relevant for initial design of control systems. It is also possible to control the generator torque or the collective blade pitch with a common **PID controller**.



External controllers

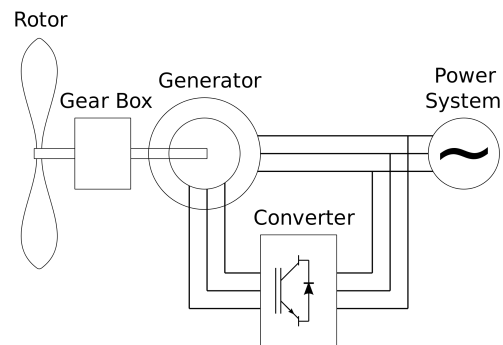
For more advanced control strategies, it is possible to define controllers as a [DLL](#) (dynamic-linked library). DLLs that can be imported into Ashes follow the Bladed format. Several DLLs, for bottom-fixed as well as floating wind turbines are provided with Ashes.

Control systems can also be written in Python and connected to Ashes through the [API](#). Similarly, any framework that can communicate over TCP, such as [Simulink](#) or [LabView](#) can be connected to the Python API. This enables real-time debugging of the control system and design using other programming languages.

Generator models

In addition to the **variable speed generator** modeled by the NREL controller, Ashes comes with some simple generator models:

- Induction generator
- Permanent Magnet Synchronous Generator
- Live generator, where you set the torque in real time



10 Environmental conditions

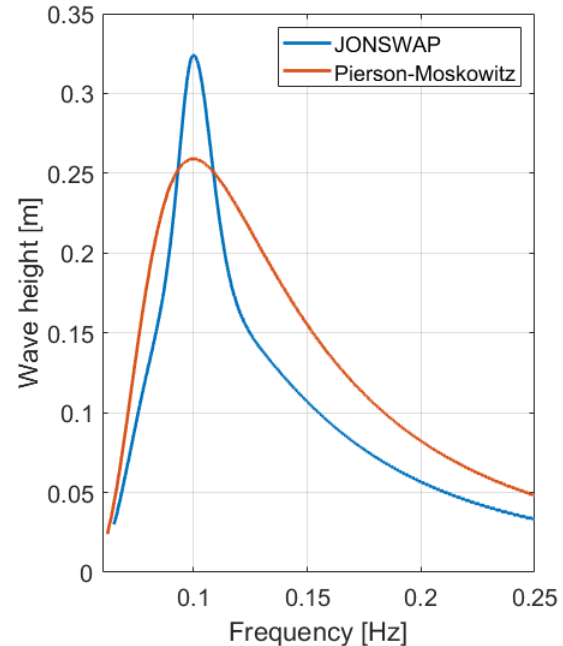
10.1 Marine environment

Linear waves

The kinematics for regular waves are computed using [Airy wave theory](#). By default, a deep water approximation is applied when the wavelength is longer than half the water depth. This can be changed in the **Sea** parameters.

[Irregular waves](#) are modeled as a linear superposition of regular waves following a JONSWAP spectrum. This also enables to model the **Pierson-Moskowitz**, which is a simpler version of the JONSWAP spectrum. In addition, two **JONSWAP spectra** can be added onto each other. The components of a wave spectrum can be defined as having **equal delta frequency** or **equal wave energy**.

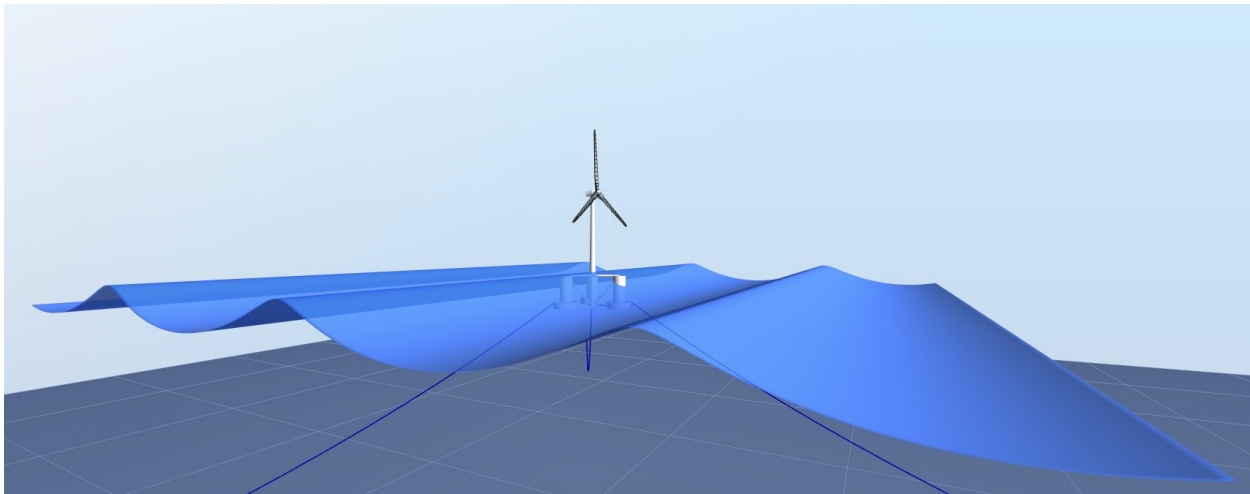
Wave spreading can be added according to the specifications in DNV-RP-C205 [2017]. It is also possible to import wave spectra defined externally.

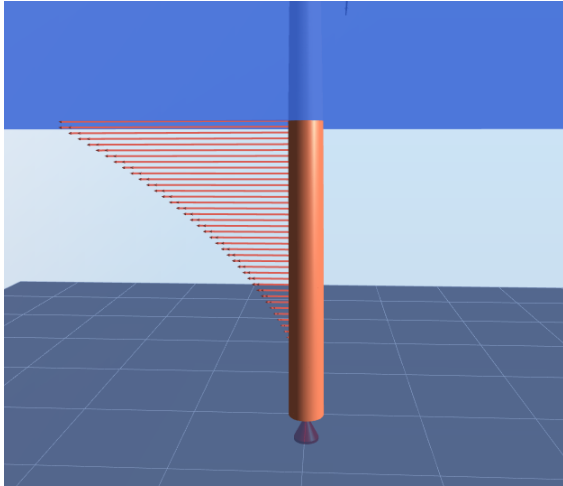


Nonlinear waves

The water particle kinematics can be modeled using the [stream function wave theory](#), as suggested by DNV-RP-C205 [2017] for modeling steep waves such as those encountered in extreme weather events. The algorithm implemented in Ashes is based on the work by [Fenton](#) and as such the order of the stream function can be set by the user.

Solving the stream function wave equations is seamlessly embedded in the simulation so that the nonlinear wave kinematics can automatically be used as input for the **Morison equation**.





Marine current

It is possible to simulate [marine currents](#) as water particles with a constant speed. Marine currents can be defined either following a power law in Ashes or having a user-defined vertical distribution externally. The loads from marine currents are added to that of waves.

Marine growth

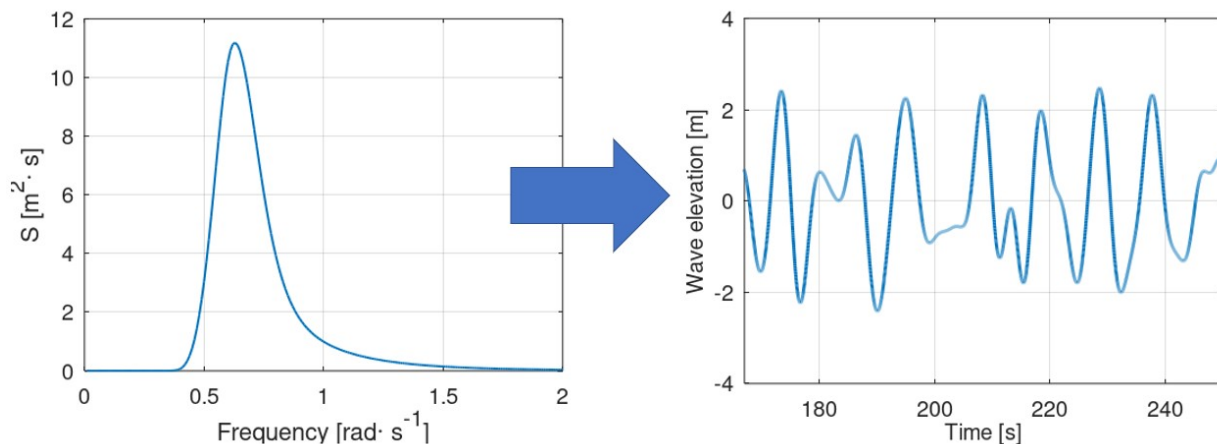
Marine growth is simulated as an increase in the diameter of selected elements with a custom density. This affects the weight of the element and the hydrodynamic load on the element.

Wheeler stretching

This method stretches the vertical coordinates to account for the fact that linear theory does not provide any particle velocity above the still water surface. Without this correction, wave loads are significantly underestimated.

Custom wave time series

Custom wave elevation time series can also be input into Ashes and used as linear wave kinematics for the **Morison equation**. This makes it possible to run a simulation with data generated by another software or measured in an experiment. The input has to be a [wave spectrum](#) which Ashes translates into a wave elevation time series.

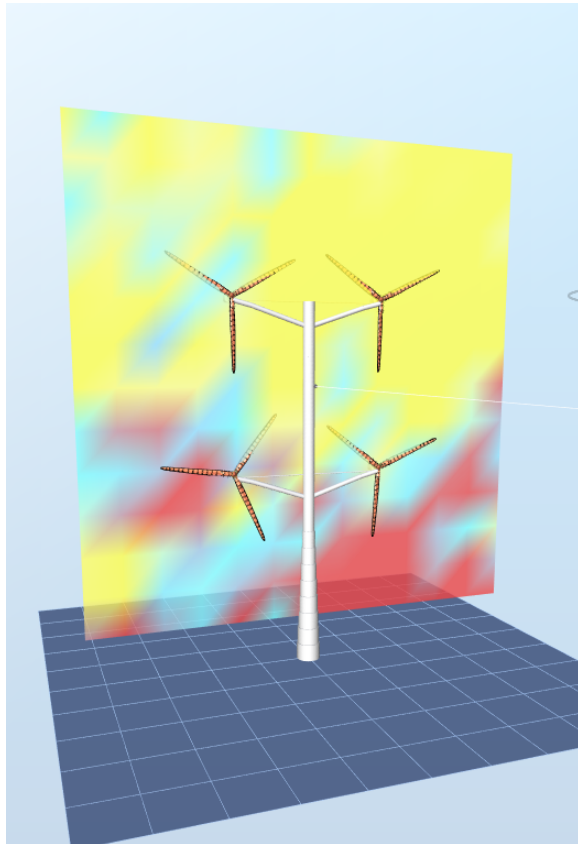


10.2 Wind

Steady wind

Ashes comes with several steady wind models. The simplest wind model in Ashes is the uniform wind, i.e. time-invariant wind. Several other options are available to account for different phenomena:

- vertical shear, following a [power law](#)
- linearly increasing wind
- wind direction variation
- user defined wind



$$V(t, z) = V_{hub} \left(\frac{z}{z_{hub}} \right)^\alpha + \left(\frac{z - z_{hub}}{D} \right) \left[2.5 + 0.2\beta\sigma_1 \left(\frac{D}{\lambda_1} \right)^{0.25} \right] (1 - \cos(2\pi t/T)) \text{ for } 0 \leq t \leq T$$

$$V(t, z) = V_{hub} \left(\frac{z}{z_{hub}} \right)^\alpha \text{ for } t > T$$

and the horizontal extreme wind shear is defined by

$$V(t, y, z) = V_{hub} \left(\frac{z}{z_{hub}} \right)^\alpha + \left(\frac{y}{D} \right) \left[2.5 + 0.2\beta\sigma_1 \left(\frac{D}{\lambda_1} \right)^{0.25} \right] (1 - \cos(2\pi t/T)) \text{ for } 0 \leq t \leq T$$

$$V(t, y, z) = V_{hub} \left(\frac{z}{z_{hub}} \right)^\alpha \text{ for } t > T$$

with

- $\alpha = 0.2$ the power law exponent (as defined in [Wind profile power law](#))
- $\beta = 6.4$

Tower shadow

The **wake deficit** due to the presence of the tower can be modeled using either a [potential flow theory](#) model for upwind turbines or the [Powles model](#) (see Powles [1983]) for downwind configurations.

IEC extreme winds

The [IEC extreme wind events](#) have been implemented as described in IEC-61400-1 [2019]. These can be used to easily generate load cases that follow the main standards in the wind energy industry.

Turbulent wind

Ashes provides a Graphical User Interface to both the TurbSim and the IEC turbulence generators. This means that Ashes can read and generate turbulent wind fields based on the **Kaimal**, **Von Karman** or **Mann** wind spectra.

The batch manager allows for seamless generation of thousands of turbulent wind fields that are automatically applied to the relevant simulation.

11 Parallelization and cloud computing

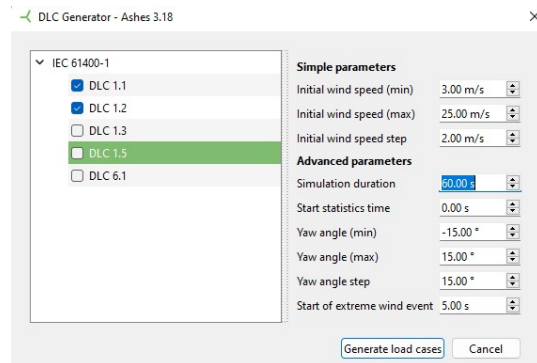
Batch analyses

The **Batch manager** is designed to enable users to easily program multiple simulations to run simultaneously. When running a batch locally, Ashes will automatically use all the available cores in your computer to run the simulations concurrently.

The batch import and export features enable the user to easily script and generate batches with thousands of load cases.

No additional license is required to run batches.

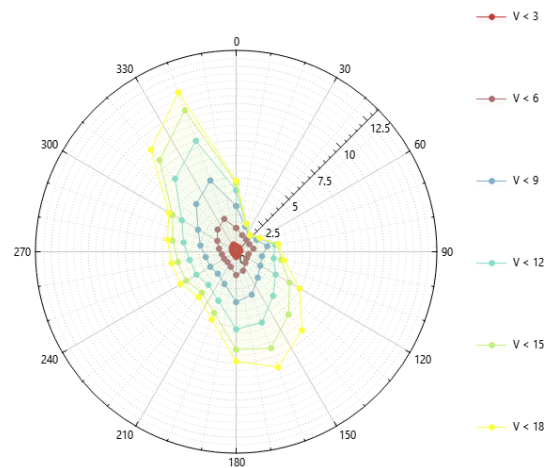
Name	Enabled	Duration	Rec. time steps	Sensors	Jobs	Atmosphere	Wind general		
							Type	Average wind speed	Shear
DLC 6.1 (3)									
DLC 6.1-1	<input checked="" type="checkbox"/>	600	1						
DLC 6.1-2	<input checked="" type="checkbox"/>			4	1	Turbulent	50	<input checked="" type="checkbox"/>	0.11
DLC 6.1-3	<input checked="" type="checkbox"/>			4	1	Turbulent	50	<input checked="" type="checkbox"/>	0.11
- Add load cases									
DLC 12 (36)									
DLC 12-1	<input checked="" type="checkbox"/>			5	1	Turbulent	3	<input checked="" type="checkbox"/>	0.2
DLC 12-2	<input checked="" type="checkbox"/>			5	1	Turbulent	3	<input checked="" type="checkbox"/>	0.2
DLC 12-3	<input checked="" type="checkbox"/>			5	1	Turbulent	3	<input checked="" type="checkbox"/>	0.2
DLC 12-4	<input checked="" type="checkbox"/>			5	1	Turbulent	5	<input checked="" type="checkbox"/>	0.2
DLC 12-5	<input checked="" type="checkbox"/>			5	1	Turbulent	5	<input checked="" type="checkbox"/>	0.2
DLC 12-6	<input checked="" type="checkbox"/>			5	1	Turbulent	5	<input checked="" type="checkbox"/>	0.2
DLC 12-7	<input checked="" type="checkbox"/>			5	1	Turbulent	7	<input checked="" type="checkbox"/>	0.2
DLC 12-8	<input checked="" type="checkbox"/>			5	1	Turbulent	7	<input checked="" type="checkbox"/>	0.2
DLC 12-9	<input checked="" type="checkbox"/>			5	1	Turbulent	7	<input checked="" type="checkbox"/>	0.2
DLC 12-10	<input checked="" type="checkbox"/>			5	1	Turbulent	9	<input checked="" type="checkbox"/>	0.2
DLC 12-11	<input checked="" type="checkbox"/>			5	1	Turbulent	9	<input checked="" type="checkbox"/>	0.2
DLC 12-12	<input checked="" type="checkbox"/>			5	1	Turbulent	9	<input checked="" type="checkbox"/>	0.2



Load case generator

The **Load Case generator** enables fast generation of the most common DLCs from IEC-61400-1 [2019], and we are adding more DLCs from more standards at every new release.

The design load cases have been predefined in the LC generator so that only a limited amount of data needs to be input. Extra parameters can also be added once the standard batch has been created.



It is also possible to generate load cases directly from meteocean conditions. A text file defining a **Scatter diagram** can be imported into Ashes, and different strategies can be used to reduce the number of load cases that have to be run.

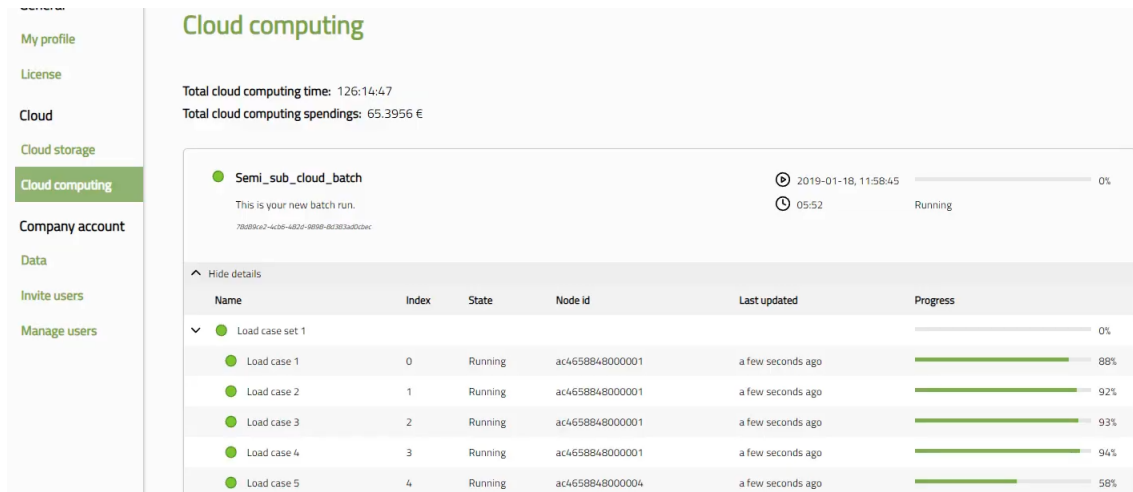
Similarly, a **wind rose file** containing information about wind directions and speeds can be imported and load cases can be generated accordingly.

There is also the option to generate the wave data as a function of the wind speed and direction, from a wind rose file.

Cloud computing

For batches with large numbers of simulations, the [cloud capability](#) is available on a pay-per-simulation basis. This feature is seamlessly integrated in Ashes and does not require any cloud competence on the user end.

The simulations are run on the cloud from the cheapest provider among Amazon, IBM or Oracle, and the results are safely stored on Microsoft Azure, where the time series can be downloaded from.



12 Command Line Interface

The [command-line interface](#) (or **CLI**) provides an easy way to run an Ashes simulation from an external coding framework. It has been tested with **Python**, **Matlab** and **C#** but can be used with any language that can call an executable file.

This enables to run Ashes without opening the graphical user interface and thus increases the speed of simulations by reducing the amount of human interaction required to do pre- and post-processing around an Ashes simulation.

It is also a very powerful tool for optimization as it makes it possible to use the output of a simulation as the input of another within a given script.

```

1 pkg load io % Load the IO package to use 'csvread' function
2
3 % Change these paths
4 ashesExePath = "C:/Program Files/Ashes 3.19/ashes-cli.exe";
5 projectFilePath = "C:/Users/Loup/Documents/Ashes 3.20/onshore.ash";
6 originalCsvFilePath = "C:/Users/Loup/Documents/Ashes 3.20/Batch.csv";
7 batchResultsDir = "C:/Users/Loup/Documents/Ashes 3.20/Batch runs";
8
9 initialThickness = 0.001; % 1 millimeter as initial tower thickness
10 thickness = initialThickness;
11 deltaThickness = 0.0005; % Step 0.5 millimeter every iteration
12 yieldStrength = 355.0/1.25; % Yield strength of steel in MPa divided by
13 i = 1;
14
15 while true
16 % Make a copy of the csv file that we will modify
17 outDir = fileparts(originalCsvFilePath);
18 batchSize = sprintf("Optimization batch, t=%f", thickness);
19 modifiedCsvFilePath = [outDir, "/", batchSize, ".csv"];
20 copyfile(originalCsvFilePath, modifiedCsvFilePath);
21 % Read csv using pandas
22 csv = csv2cell(modifiedCsvFilePath, ':');
23 csv(cellfun(@isempty, csv)) = {};
24 % Modify the tower thickness value and save file
25 csv(5,11) = num2str(thickness);
26 cell2csv(modifiedCsvFilePath, csv, ':');
27 % Run ashes-cli
28 cmd = sprintf("%s" -runbatch -projectfile "%s" -batchcsvfile "%s",
29 [status, cmdout] = system(cmd);
30 if status == 0
31     fprintf('Return code was %d, check console output for more inform
32     break; % Something went wrong, check console output.

```

References

- M A. Crisfield, U Galvanetto, and Gordan Jelenić. Dynamics of 3-D co-rotational beams. *Computational Mechanics*, 20:507–519, January 1997. doi: 10.1007/s004660050271.
- API. Geotechnical and Foundation Design Considerations. Technical report, API, April 2011.
- Christian Bak, Frederik Zahle, Robert Bitsche, Taesong Kim, Anders Yde, Lars Christian Henriksen, Anand Natarajan, and Morten Hansen. Description of the DTU 10 MW Reference Wind Turbine. Technical Report I-0092, DTU Vindenergi, July 2013.
- Per Ivar Bruheim. Implementation of a modified inverse iteration algorithm in an object-oriented finite element framework. Technical report, NTNU, Trondheim, Norway, 2011.
- Per Ivar Bruheim. Development and validation of a finite element software facilitating large-displacement aeroelastic analysis of wind turbines. 83, 2012. URL <https://brage.bibsys.no/xmlui/handle/11250/232200>.
- Byron W. Byrne, Harvey J. Burd, Lidija Zdravković, Ross A. McAdam, David M. G. Taborda, Guy T. Houlsby, Richard J. Jardine, Christopher M. Martin, David M. Potts, and Kenneth G. Gavin. PISA: new design methods for offshore wind turbine monopiles. *Revue Française de Géotechnique*, (158):3, 2019. ISSN 0181-0529, 2493-8653. doi: 10.1051/geotech/2019009. URL <https://www.geotechnique-journal.org/articles/geotech/abs/2019/01/geotech190009s/geotech190009s.html>.
- DNV-RP-C203. DNV RP-C203 Fatigue design for offshore steel structures. Technical report, DNV, April 2016.
- DNV-RP-C205. DNV-RP-C205 Environmental Conditions and Environmental Loads. Technical report, DNV, August 2017.
- Mark Drela. XFOIL, December 2013. URL <http://web.mit.edu/drela/Public/web/xfoil/>.
- Eurocode-EN-1993. Eurocode 3: Design of steel structures - Part 1-9: Fatigue, May 2005. URL <https://www.phd.eng.br/wp-content/uploads/2015/12/en.1993.1.9.2005-1.pdf>.
- O.M. Faltinsen. *Sea loads on ships and offshore structures*. Ocean Technology. Cambridge, 1990. ISBN 0-521-45870-6.
- Evan Gaertner, 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. Definition of the IEA Wind 15-Megawatt Offshore Reference Wind Turbine. Technical Report NREL/TP-5000-75698, National Renewable Energy Lab. (NREL), Golden, CO (United States), March 2020.
- Martin O. L. Hansen. *Aerodynamics of wind turbines*. Earthscan, February 2008. ISBN 978-1-84407-438-9.
- IEC-61400-1. IEC 61400-1: Wind Turbines—Part 1: Design Requirements. Technical report, February 2019.
- Jason M. Jonkman, S. Butterfield, Walt Musial, and G. Scott. Definition of a 5-MW Reference Wind Turbine for Offshore System Development. February 2009.
- D. De Klerk, D. J. Rixen, and S. N. Voormeeren. General Framework for Dynamic Substructuring: History, Review and Classification of Techniques. *AIAA Journal*, 46(5):1169–1181, 2008. ISSN 0001-1452. doi: 10.2514/1.33274. URL <https://doi.org/10.2514/1.33274>.

- Per-Åge Krogstad and Pål Egil Eriksen. "Blind test" calculations of the performance and wake development for a model wind turbine. *Renewable Energy*, 50:325–333, February 2013. ISSN 0960-1481. doi: 10.1016/j.renene.2012.06.044. URL <http://www.sciencedirect.com/science/article/pii/S0960148112003953>.
- Wojciech Popko, Matthias L. Huhn, Amy Robertson, Jason Jonkman, Fabian Wendt, Kolja Müller, Matthias Kretschmer, Fabian Vorpahl, Torbjørn Ruud Hagen, Christos Galinos, Jean-Baptiste Le Dreff, Philippe Gilbert, Bertrand Auriac, Francisco Navarro Villora, Paul Schünemann, Ilmas Bayati, Marco Belloli, Sho Oh, Yoshitaka Totsuka, Jacob Qvist, Erin Bachynski, Stian Høegh Sørum, Paul E. Thomassen, Hyunkyung Shin, Felipe Vittori, Josean Galván, Climent Molins, Paul Bonnet, Tjeerd van der Zee, Roger Bergua, Kai Wang, Pengcheng Fu, and Jifeng Cai. Verification of a Numerical Model of the Offshore Wind Turbine From the Alpha Ventus Wind Farm Within OC5 Phase III. pages V010T09A056–V010T09A056. American Society of Mechanical Engineers, June 2018. doi: 10.1115/OMAE2018-77589. URL <http://proceedings.asmedigitalcollection.asme.org/proceeding.aspx?articleid=2704884>.
- S.R.J. Powles. The Effects of Tower Shadow on the Dynamics of a Horizontal-Axis Wind Turbine. *Wind Engineering*, 7(1):26–42, 1983. ISSN 0309-524X. URL <https://www.jstor.org/stable/43749009>. Publisher: Sage Publications, Ltd.
- H. Snel, J. G. Schepers, and S. E.C Nederland. *Joint investigation of dynamic inflow effects and implementation of an engineering method*. Netherlands Energy Research Foundation ECN, 1995.
- Stig Øye. Dynamic stall, simulated as a time lag of separation. *Proceedings of the 4th IEA Symposium on the Aerodynamics of Wind Turbines*, 1991. URL http://scholar.google.no/scholar?hl=no&q=%C3%98ye%2C+S.+%281991%29+%E2%80%98Dynamic+stall%2C+simulated+as+a+time+lag+of+separation%E2%80%99%2C&btnG=S%C3%B8k&as_ylo=&as_vis=0.