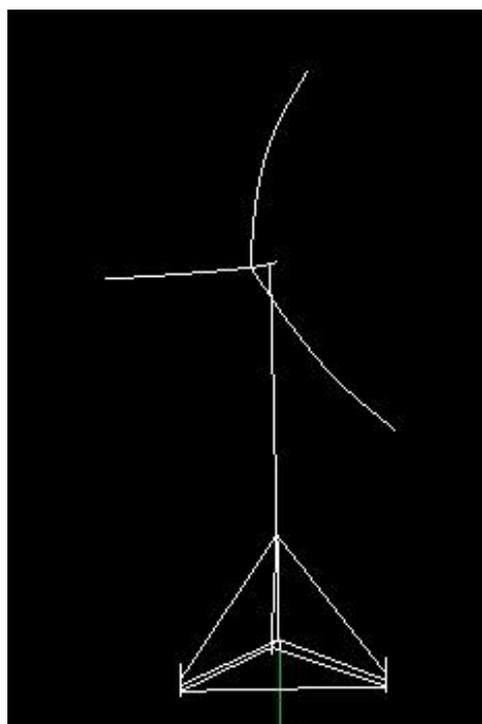


How 2 HAWC2, the user's manual

Torben Juul Larsen, Anders M. Hansen

Risø-R-1597(ver. 4-7)(EN)



Authors: Torben Juul Larsen, Anders M. Hansen
Title: How 2 HAWC2, the user's manual
Institute: Department of Wind Energy

Abstract:

The report contains the user's manual for the aeroelastic code HAWC2. The code is intended for calculating wind turbine response in time domain and has a structural formulation based on multi-body dynamics. The aerodynamic part of the code is based on the blade element momentum theory, but extended from the classic approach to handle dynamic inflow, dynamic stall, skew inflow, shear effects on the induction and effects from large deflections. It has mainly been developed within the years 2003-2006 at the aeroelastic design research programme at Risoe, National laboratory Denmark, but is continuously updated and improved.

This manual is updated for HAWC2 version 12.6 and wkin.dll version 2.4

Risø-R-1597(ver. 4-7)(EN)
May 2018
ISSN 0106-2840 ISBN
978-87-550-3583-6
Groups own reg. no.: 1110412-3

Technical University of Denmark
DTU Wind Energy
Frederiksborgvej 399
4000 Roskilde
Denmark
Telephone +45 45 46774004
bibl@risoe.dk
Fax +45 46774013

Contents

1	Preface	<i>8</i>
2	Acknowledgements	<i>9</i>
3	General input layout	<i>10</i>
3.1	Continue_in_file option	<i>10</i>
4	HAWC2 version handling	<i>11</i>
5	Coordinate systems	<i>12</i>
6	Simulation	<i>14</i>
6.1	Main command block - Simulation	<i>14</i>
6.2	Sub command block - newmark	<i>14</i>
7	Structural input	<i>15</i>
7.1	Main command block - new_htc_structure	<i>15</i>
7.2	Sub command block - main_body	<i>15</i>
7.3	Sub command - orientation	<i>25</i>
7.4	Sub command - constraint	<i>28</i>
8	DLL control	<i>34</i>
8.1	Main command block – dll	<i>34</i>
8.2	Sub command block – hawc_dll	<i>34</i>
8.3	Sub command block – type2_dll	<i>35</i>
8.4	Sub command block – init	<i>36</i>
8.5	Sub command block – output	<i>36</i>
8.6	Sub command block – actions	<i>36</i>
8.7	HAWC_DLL format example written in FORTRAN 90	<i>40</i>
8.8	HAWC_DLL format example written in Delphi / Lazarus / Pascal	<i>41</i>
8.9	HAWC_DLL format example written in C	<i>42</i>
8.10	TYPE2_dll written in Delphi / Lazarus / Delphi	<i>43</i>
8.11	TYPE2_dll written in C	<i>44</i>
8.12	TYPE2_DLL format example written in FORTRAN 90	<i>45</i>

9 Wind and Turbulence 46

- 9.1 Main command block -wind 46
- 9.2 Sub command block - mann 48
- 9.3 Sub command block - flex 50
- 9.4 File description of a user defined shear 51
- 9.5 Example of user defined shear file 51
- 9.6 Sub command block - wakes 52
- 9.7 File description of a user defined wake deficit file 53
- 9.8 Example of user defined wake deficit file 54
- 9.9 Sub command block – tower_shadow_potential 55
- 9.10 Sub command block – tower_shadow_jet 55
- 9.11 Sub command block – tower_shadow_potential_2 55
- 9.12 Sub command block – tower_shadow_jet_2 56
- 9.13 Sub command block – turb_export 56

10 Aerodynamics 58

- 10.1 Main command block - aero 58
- 10.2 Sub command block – dynstall_so 59
- 10.3 Sub command block – dynstall_mhh 59
- 10.4 Sub command block – dynstall_ateflap 60
- 10.5 Sub command block – aero_noise 61
- 10.6 Sub command block – bemwake_method 63
- 10.7 Sub command block – nearwake_method 63
- 10.8 Sub command block – vawtwake_method 64
- 10.9 Data format for the aerodynamic layout 64
- 10.10 Example of an aerodynamic blade layout file 65
- 10.11 Data format for the profile coefficients file 66
- 10.12 Example of the profile coefficients file “_pc file” 66
- 10.13 Data format for the flap steady aerodynamic input (.ds file) 67
- 10.14 Example of a .ds flap steady aerodynamic input file 68
- 10.15 Data format for the user defined a-ct relation 68
- 10.16 Main command block – blade_c2_def (for use with old_htc_structure format) 69

11 Aerodrag (for tower and nacelle drag) 70

- 11.1 Main command aerodrag 70
- 11.2 Subcommand aerodrag_element 70

12 Hydrodynamics 71

- 12.1 Main command block - hydro 71
- 12.2 Sub command block – water_properties 71
- 12.3 Sub command block – hydro_element 71
- 12.4 Description of the water_kinematics_dll format. 73
- 12.5 User manual to the standard wkin.dll version 2.4. 73
- 12.6 Main commands in the wkin.dll 74
- 12.7 Sub command reg_airy 74
- 12.8 Sub command ireg_airy 74
- 12.9 Sub command det_airy 75
- 12.10 Sub command strf 75
- 12.11 Sub command wavemods 75
- 12.12 Wkin.dll example file 77

13 Soil module 78

- 13.1 Main command block - soil 78
- 13.2 Sub command block – soil_element 78
- 13.3 Data format of the soil spring datafile 78

14 External forces through DLL 80

- 14.1 Main command block – Force 80
- 14.2 Example of a DLL interface written in fortran90 80
- 14.3 Example of a DLL interface written in Lazarus / Pascal 81

15 Output 82

- 15.1 Commands used with results file writing 82
- 15.2 File format of HAWC_ASCII files 82
- 15.3 File format of HAWC_BINARY files 83
- 15.4 File format for gtsdf and gtsdf64 files 85
- 15.5 mbody (main body output commands) 85
- 15.6 Constraint (constraint output commands) 88
- 15.7 aero (aerodynamic related commands) 89
- 15.8 wind (wind output commands) 96
- 15.9 wind_wake (wind wake output commands) 97
- 15.10 dll (DLL output commands) 97
- 15.11 hydro (hydrodynamic output commands) 98
- 15.12 general (general output commands) 98

15.13 Output_at_time (output at a given time) 99

A Example of main input file 102

B Code Version Data 114

1 Preface

The HAWC2 code is a code intended for calculating wind turbine response in time domain. It has been developed within the years 2003-2006 at the aeroelastic design research programme at Risoe, National laboratory Denmark.

The structural part of the code is based on a multibody formulation where each body is an assembly of Timoshenko beam elements. The formulation is general which means that quite complex structures can be handled and arbitrary large rotations of the bodies can be handled. The turbine is modeled by an assembly of bodies connected with constraint equations, where a constraint could be a rigid coupling, a bearing, a prescribed fixed bearing angle etc. The aerodynamic part of the code is based on the blade element momentum theory, but extended from the classic approach to handle dynamic inflow, dynamic stall, skew inflow, shear effects on the induction and effects from large deflections. Several turbulence formats can be used. Control of the turbine is performed through one or more DLL's (Dynamic Link Library). The format for these DLL's is also very general, which means that any possible output sensor normally used for data file output can also be used as a sensor to the DLL. This allows the same DLL format to be used whether a control of a bearing angle, an external force or moment is placed on the structure. The code has internally at Risoe been tested against the older validated code HAWC, the CFD code Ellipsys and numerous measurements. Further on detailed verification is performed in the IEA annex 23 and annex 30 research project regarding offshore application. Scientific papers involving the HAWC2 is normally posted on the www.hawc2.dk homepage, where the code, manual and more can be downloaded. During the programming of the code a lot of focus has been put in the input checking so hopefully meaningful error messages are written to the screen in case of lacking or obvious erroneous inputs. However since the code is still constantly improved we appreciate feedback from the users – both good and bad critics are welcome. The manual is also constantly updated and improved, but should at the moment cover the description of available input commands.

2 Acknowledgements

The code has been developed primarily by internal funds from Risø National Laboratory – Technical University of Denmark, but the research that forms the basis of the code is mainly done under contract with the Danish Energy Authority. The structural formulation of the model is written by Anders M. Hansen as well as the solver and the linking between external loads and structure. The anisotropic FPM beam model is written by Christian Pavese, Taeseong Kim and Anders M. Hansen. The aerodynamic BEM module is written by Helge A. Madsen and Torben J. Larsen. Three different stall models are implemented where the S.Ø. (Stig Øye) model is implemented by Torben J. Larsen, the mhh Beddoes model is written by Morten Hansen and Mac Gaunaa and the ateflap model used for trailing edge flaps is written by Mac Gaunaa and Peter Bjørn Andersen and has later been rewritten by Leonardo Bergami. The wind and turbulence module as well as the soil and DLL modules are written by Torben J. Larsen. The hydrodynamic module is written by Anders M. Hansen and Torben J. Larsen. The turbulence generator is written by Jacob Mann and the WASP Team and converted into a DLL by Peter Bjørn Andersen. The dynamic wake meandering module is written by Helge A. Madsen, Gunner Larsen and Torben J. Larsen. The eigenvalue solver is implemented by Anders M. Hansen and John Hansen. General maintenance is performed by Torben J. Larsen, Anders M. Hansen and Mikkel Friis-Møller. The web page www.hawc2.dk and the manual are maintained by Mikkel Friis-Møller.

3 General input layout

The HAWC 2 input format is written in a form that forces the user to write the input commands in a structured way so aerodynamic commands are kept together, structural commands the same etc.

The commands are divided into command blocks using the begin-end syntax. Each line has to be ended with a semi colon “;” which gives the possibility for writing comments and the end of each line after the semi colon. All command lines can be written with capital or small letters, but inside the code all lines are transformed into small letters. This could have importance if something case sensitive is written (e.g. the name of a subroutine within a DLL).

```
begin simulation;
  time_stop    100.0 ;
  solvertype   1 ;   (newmark)
;
  begin newmark;
    beta       0.27;
    gamma      0.51;
    deltat     0.02;
  end newmark;
end simulation;
```

In the next chapters the input commands are explained for every part of the code. The notation is main command for a begin-end command block that is not a sub part of another begin-end block, and sub command block for a begin-end block that is included within another block. In the above written example “simulation” is a main command block and “newmark” is a sub command block.

3.1 Continue_in_file option

A feature from version 6.0 and newer is the possibility of continuing reading of the main input file into another. The command word `continue_in_file` followed by a file name causes the program to open the new file and continue reading of input until the command word `exit`. When `exit` is read the reading will continue in the previous file. An infinite number of file levels can be used. The HAWC2 input format is written in a form that forces the user to write the input commands in a structured way so aerodynamic commands are kept together, structural commands the same etc.

Command name	Explanation
<code>continue_in_file</code>	1. File name (and path) to sublevel input file
<code>exit</code>	End of input file. Input reading is continued in higher level input file.

4 HAWC2 version handling

The HAWC2 code is still frequently updated and version handling is therefore of utmost importance to ensure quality control. For every new released version of the code a new version number is hard coded in the source. This number can be found by executing the HAWC2.exe file without any parameters. The version number is echoed to screen. The same version number is also written to every result file no matter whether ASCII or binary format is chosen. Hereby it is possible to reproduce all results at later stage and to dig in the source code for at previous version if special problems occur.

All information covering the different code versions has been made. These data are listed in appendix B.

5 Coordinate systems

The global coordinate system is located with the z-axis pointing vertical downwards. The x and y axes are horizontal to the side. When wind is submitted, the default direction is along the global y-axes. Within the wind system meteorological u,v,w coordinates are used, where u is the mean wind speed direction, v is horizontal and w vertical upwards. When x,y,z notation is used within the wind coo. this refers directly to the u,v,w definition. Every substructure and body (normally the same) is equipped with its own coordinate system with origo in node1 of this structure. The structure can be arbitrarily defined regarding orientation within this coordinate system. Within a body a number of structural elements are present. The orientation of coordinate systems for these elements are chosen automatically by the program. The local z axis is from node 1 to 2 on the element. The coordinate system for the blade structures must be defined with the z axis pointing from the blade root and outwards, x axis in the tangential direction of rotation and y axis from the pressure side towards the suction side of the blade profiles. This is in order to make the linkage between aerodynamics and structure function.

In order to make a quick check of the layout of the structure the small program “animation.exe” can be used (this requires that an animation file has been written using the command animation in the Simulation block). The view option in this program is handled by keyboard hotkeys:

Animation Hotkeys:

translate: (shift)+{x,y,z}
rotate: arrow keys
rotate about line-of-sight: ctrl+left/right
zoom in: ctrl+up
zoom out: ctrl+down
amplify displacement (only for animation of natural frequencies): +
decrease displacement (only for animation of natural frequencies): -

If the animation does not start, press “s”

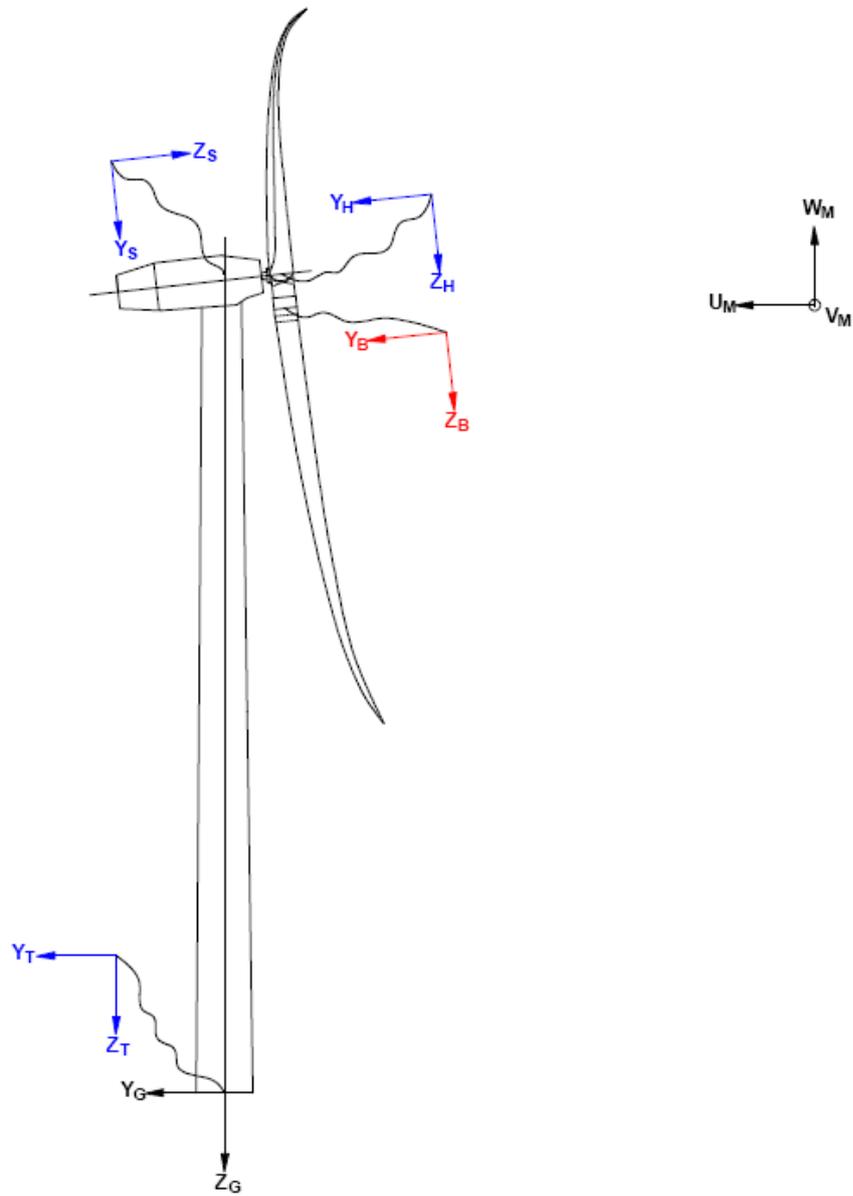


Figure 1: Illustration of coordinate system as result of user input from example in appendix A: Example of main input file. There are two coordinate systems in black which are the default coordinate systems of global reference and default wind direction. The blue coordinate systems are main body coordinate systems attached to node 1 of the substructure, the orientation of these are fully determined by the user. The red coordinate systems are also defined by the user, but in order to make the linkage between aerodynamic forces and structure work these have to have the z from root to tip, x in chordwise direction and y towards the suction side.

6 Simulation

6.1 Main command block - Simulation

Obl.	Command name	Explanation
*	time_stop	1. Simulation length [s]
	solvertype	1. Choice of available solver method (1=newmark)
	solver_relax	1. Relaxation parameter on increment within a timestep. Can be used to make difficult simulation run through solver when parameter is decreased, however on the cost of simulation speed. Default=1.0
	on_no_convergence	Parameter that informs solver of what to do if convergence is not obtained in a time step. 1. 'stop': simulation stops – default. 'continue': simulation continues, error message is written.
	convergence_limits	Convergence limits that must be obtained at every time step. 1. epsresq, residual on internal-external forces, default=10.0 2. epsresd, residual on increment, default=1.0 3. epsresg, residual on constraint equations, default=1E-7
	max_iterations	1. Number of maximum iterations within a time step.
	animation	Included if animation file is requested 1. Animation file name incl. relative path. E.g. ./animation/animation1.dat
	visualization	Included if simulation visualization file is requested 1. Visualization file name incl. relative path. E.g. ./visualization/example.hdf5;
	logfile	Included if a logfile is requested internally from the htc command file. 1. Logfile name incl. relative path. E.g. ./logfiles/log1.txt

6.2 Sub command block - newmark

Obl.	Command name	Explanation
	beta	1. beta value (default=0.27)
	gamma	1. gamma value (default=0.51)
*	deltat	1. time increment [s]
	symmetry	1. Solver assumption regarding mass, damping and stiffness matrices (1=symmetric (default), 2=assymetric (recommended for offshore structures). When hydrodynamic loading is applied this parameter will automatically change to 2.)

7 Structural input

7.1 Main command block - new_htc_structure

Obl.	Command name	Explanation
	beam_output_file_name	1. Filename incl. relative path to file where the beam data are listed (output) (example ./info/beam.dat)
	body_output_file_name	1. Filename incl. relative path to file where the body data are listed (output) (example ./info/body.dat)
	struct_inertia_output_file_name	1. Filename incl. relative path to file where the global inertia information data are listed (output) (example ./info/inertia.dat)
	body_eigenanalysis_file_name	1. Filename incl. relative path to file where the results of an eigenanalysis are written. (output) (example ./info/eigenfreq.dat) This command is still available but is NOT a recommended method.
	constraint_output_file_name	1. Filename incl. relative path to file where the constraint data are listed (output). (example ./info/constraint.dat)
	structure_eigenanalysis_file_name	1. Filename incl. relative path to file where the results of an complete turbine eigenanalysis are listed (example ./info/eigen_all.dat). Animation files of the first modes are places in same directory as the HAWC2 executable. 2. Optional parameter determining if structural damping is included in the eigenvalue calculation or not. (0=damping not included, most robust method, 1=damping included default)
	system_eigenanalysis	Eigenvalue calculation of the total system including external systems attached, eg. mooring lines. Constraint equations are also fully included in the analysis. 1. Filename incl. relative path to file where the results of an complete turbine eigenanalysis are listed (example ./info/eigen_all.dat). Animation files of the first modes are places in same directory as the HAWC2 executable. 2. (optional) Parameter determining if structural damping is included in the eigenvalue calculation or not. (0=damping not included, most robust method, 1=damping included default) 3. (optional) Number of modes outputted. 4. (optional) Time for when the eigen analysis is carried out. Eg. after a settling of a floating system.

7.2 Sub command block - main_body

This block can be repeated as many times as needed. For every block a new body is added to the structure. A main body is a collection of normal bodies which are grouped together for bookkeeping purposes related to input output. When a main body consist of several bodies the spacing the name of each body inherits the name of the master body and is given an additional name of ‘_#’, where # is the body number. An example could be a main body called ‘blade1’ which consist of two bodies. These are then called ‘blade1_1’ and ‘blade1_2’ internally in the code. The internal names are only important if (output) commands are used that refers to the specific body name and not the main body name.

From version 11.6 it is possible to attach an encrypted DLL where the blade data can be extracted. An example of how to encrypt this can be obtained by request through the www.hawc2.dk web page.

Obl.	Command name	Explanation
*	name	1. Main_body identification name (must be unique)
*	type	1. Element type used (options are: timoschenko)
*	nbodies	1. Number of bodies the main_body is divided into (especially used for blades when large deformation effects needs attention). Equal number of elements on each body, eventually extra elements are placed on the first body.
*	node_distribution	1. Distribution method of nodes and elements. Options are: "uniform" nnodes. Where uniform ensures equal element length and nnodes are the node numbers. "c2_def", which ensures a node a every station defined with the sub command block c2_def.
	damping	<p>Original damping model that can only be used when the shear center location equals the elastic center to ensure a positive definite damping matrix. It is recommended to use the damping_posdef command instead. Rayleigh damping parameters containing factors that are multiplied to the mass and stiffness matrix respectfully.</p> <p>! Pay attention, the mass proportional damping is not contributing when a mbdy consist of multiple bodies !</p> <ol style="list-style-type: none"> 1. M_x 2. M_y 3. M_z 4. K_x 5. K_y 6. K_z <p>NOTE: This damping model cannot be used with the Fully Populated Matrix ("FPM 1", see below) beam element!</p>
	damping_posdef	<p>Rayleigh damping parameters containing factors. M_x, M_y, M_z are constants multiplied on the mass matrix diagonal and inserted in the damping matrix. K_x, K_y, K_z are factors multiplied on the moment of inertia I_x, I_y, I_z in the stiffness matrix and inserted in the damping matrix. Parameters are in size approximately the same as the parameters used with the original damping model written above.</p> <p>! Pay attention, the contribution from mass proportional damping is limited when a mbdy consist of multiple bodies !</p> <ol style="list-style-type: none"> 1. M_x 2. M_y 3. M_z 4. K_x 5. K_y 6. K_z <p>NOTE: This damping model cannot be used with the Fully Populated Matrix ("FPM 1", see below) beam element!</p>

Obl.	Command name	Explanation
	damping_aniso	<p>Mixed mass/stiffness proportional and stiffness proportional damping parameters containing factors. $\eta_x^m, \eta_y^m, \eta_t^m$ are constants multiplied on a mixed mass/stiffness matrix diagonal and inserted in the damping matrix. $\eta_x^s, \eta_y^s, \eta_t^s$ are factors multiplied on the moment of inertia I_x, I_y, I_z in the stiffness matrix and inserted in the damping matrix.</p> <p>! Pay attention, the mass/stiffness proportional damping is not contributing when a mbody consist of multiple bodies !</p> <ol style="list-style-type: none"> 1. η_x^m 2. η_y^m 3. η_t^m 4. η_x^s 5. η_y^s 6. η_t^s
	copy_main_body	<p>Command that can be used if properties from a previously defined body shall be copied. The name command still have to be present, all other data are overwritten.</p> <ol style="list-style-type: none"> 1. Main_body identification name of main_body that is copied.
	gravity	<ol style="list-style-type: none"> 1. Specification of gravity (directed towards zG). <p>NB! this gravity command only affects the present main body. Default=9.81 [m/s2]</p>
	concentrated_mass	<p>Concentrated masses and inertias can be attached to the structure. The offset distance from the node to the center of mass is given in the body's coordinates system. The moments and products of inertia is given around the center of mass in the body's coordinates system.</p> <ol style="list-style-type: none"> 1. Node number to which the inertia is attached. 2. Offset distance x-direction [m] 3. Offset distance y-direction [m] 4. Offset distance z-direction [m] 5. Mass [kg] 6. $I_x x$ [kg m²] 7. $I_y y$ [kg m²] 8. $I_z z$ [kg m²] 9. $I_x y$ [kg m²] – optional 10. $I_x z$ [kg m²] – optional 11. $I_y z$ [kg m²] – optional
	external_bladedata_dll	<p>Blade structural data are found in an external encrypted dll. If this command is present only these other command lines need to be present (name, type, nbodies, node_distribution and a damping command line).</p> <ol style="list-style-type: none"> 1. Company name (that has been granted a password, eg. dtu). 2. Password for opening this specific dll, eg. test1234 3. path and filename for the dll. eg. ./data/encr_blade_data.dll

7.2.1 Sub sub command block – timoschenko_input

Block containing information about location of the file containing distributed beam property data and the data set requested.

Obl.	Command name	Explanation
*	filename	1. Filename incl. relative path to file where the distributed beam input data are listed (example ./data/hawc2_st.dat)
	FPM	Logic command for Fully Populated Mat_{ix} beam element: 1. Write "1" to read a structural input file based on the fully populated stiffness matrix. Write "0" for the original beam model If the command is neglected, HAWC2 will assume that the structural input file is based on the original beam model
*	set	1. Set number 2. Sub set number

7.2.2 Sub sub command block – c2_def

In this command block the definition of the centerline of the main_body is described (position of the half chord, when the main_body is a blade). The input data given with the sec commands below is used to define a continuous differentiable line in space using akima spline functions. This centerline is used as basis for local coordinate system definitions for sections along the structure. If two input sections are given it is assumed that all points are on a straight line. If three input sections are given points are assumed to be on the line consisted of two straight lines. If four or more input sections are given points are assumed to be on an akima interpolated spline. This spline will include a straight line if a minimum of three points on this line is defined.

Position and orientation of half chord point related to main body coo.

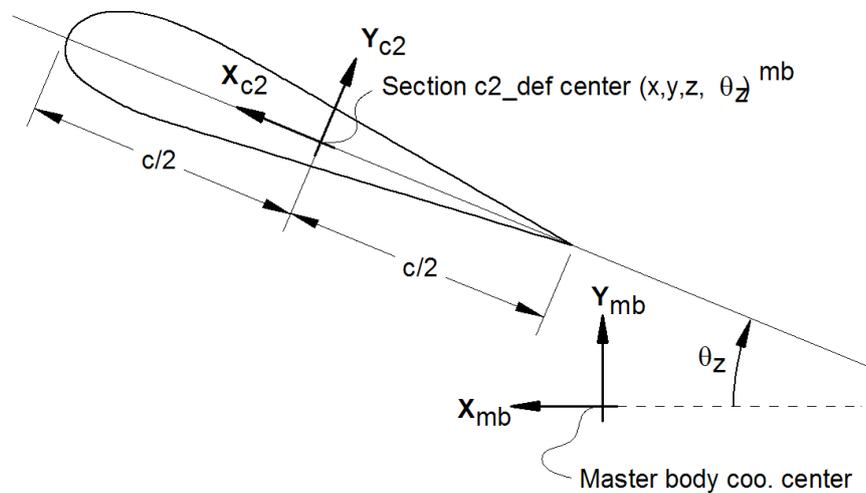
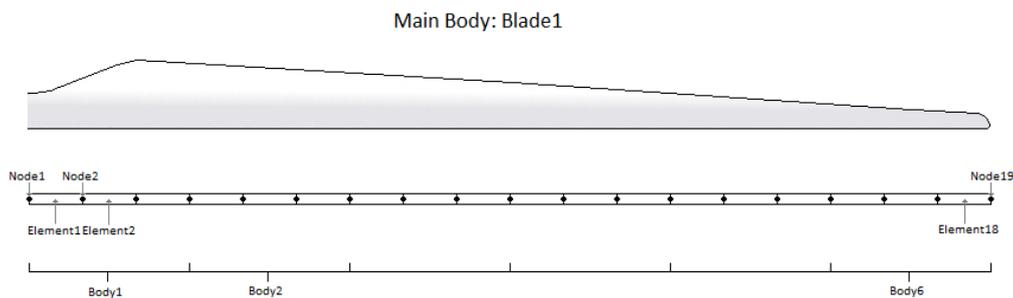


Figure 2: Illustration of c2_def coordinate system related to main body coordinates. The blade z-coordinate has to be positive from root towards the tip.

Obl.	Command name	Explanation
*	nsec	Must be the present before a “sec” command. 1. Number of section commands given below
*	sec	Command that must be repeated “nsec” times. Minimum 4 times. 1. Number 2. x-pos [m] 3. y-pos [m] 4. z-pos [m] 5. θ_z [deg]. Angle between local x-axis and main_body x-axis in the main_body x-y coordinate plane. For a straight blade this angle is the aerodynamic twist. Note that the sign is positive around the z-axis, which is opposite to traditional notation for etc. a pitch angle.

Here is an illustration of how a blade can be defined with respect to discretisation of bodies, nodes and elements.



Here is an example of this written into the htc-input file.

```
begin main_body;
name      blade1 ;
type      timoschenko ;
nbodies   6 ;
node_distribution  c2_def;
damping_posdef  1.17e-4 5.77e-5 6.6e-6 6.6e-4 5.2e-4 6.5e-4 ;
begin timoschenko_input ;
filename ./data/st_file.txt ;
  FPM 0; (optional, when parameter is 0)
  set 1 1 ;          set subset
end timoschenko_input;
begin c2_def;          Definition of centerline (main_body coordinates)
  nsec 19 ;
sec 1 -0.0000 0.0000 0.000 0.000 ;
sec 2 -0.0041 0.0010 3.278 -13.590 ;
sec 3 -0.1048 0.0250 6.556 -13.568 ;
sec 4 -0.2582 0.0492 9.833 -13.564 ;
sec 5 -0.4694 0.0587 13.111 -13.546 ;
sec 6 -0.5689 0.0957 16.389 -11.406 ;
sec 7 -0.5455 0.0883 19.667 -10.145 ;
sec 8 -0.5246 0.0732 22.944 -9.043 ;
sec 9 -0.4362 0.0669 26.222 -7.843 ;
sec 10 -0.4644 0.0554 29.500 -6.589 ;
sec 11 -0.4358 0.0449 32.778 -5.447 ;
sec 12 -0.4859 0.0347 36.056 -4.234 ;
```

```

sec 13 -0.3759 0.0265 39.333 -3.545 ;
sec 14 -0.3453 0.0130 42.611 -2.223 ;
sec 15 -0.3156 0.0084 45.889 -1.553 ;
sec 16 -0.2791 0.0044 49.167 -0.934 ;
sec 17 -0.2675 0.0017 52.444 -0.454 ;
sec 18 -0.1785 0.0003 55.722 -0.121 ;
sec 19 -0.1213 0.0000 59.000 -0.000 ;
  end c2_def ;
end main_body;

```

Format definition of file including distributed beam properties The format of this file which in the old HAWC code was known as the hawc_st file is changed slightly for the HAWC2 new_htc_structure format. In the file (which is a text file) two different datasets exist. There is a main set and a sub set. The main set is located after a “#” sign followed by the main set number. Within a main there can be as many subsets as desired. They are located after a “\$” sign followed by the local set number. The next sign of the local set number is the number of lines in the following rows that belong to this sub set.

There are two types st_file: - The st_file for the original HAWC2 beam element; input parameters for this model are reported in Table 1 HAWC2 original beam element structural data The st_file for the new anisotropic FPM beam element; input parameters reported in - Table 2 New HAWC2 anisotropic beam element structural data

In general all centers are given according to the $C_{1/2}$ center location and all other are related to the principal bending axes. For the anisotropic beam element, centers are given according to the $C_{1/2}$ center location, but the cross sectional stiffness matrix is given at the elastic center rotated along the principal bending axes.

Position of structural centers related to c2_def section coo.

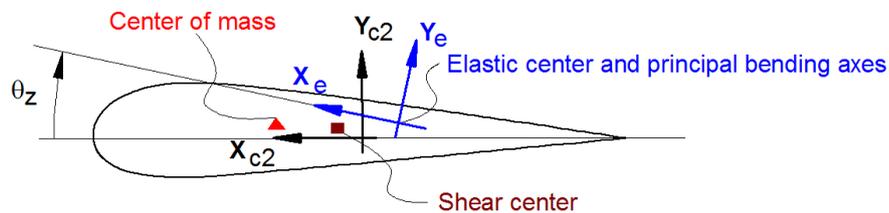


Figure 3: Illustration of structural properties that in the input files are related to the c2 coordinate system.

A small explanation about radius of gyration (also called radius of inertia) and the area moment of inertia (related to stiffness) is shown below in N.5 and N.11

An example of a st original beam formulation input file can be seen on the next page. The most important features to be aware of are colored with red.

Table 1: HAWC2 original beam element structural data

Column	Parameter
1	r, curved length distance from main_body node 1 [m]
2	m, mass per unit length [kg/m]
3	x_m , x_{c2} -coordinate from $C_{1/2}$ to mass center [m]
4	y_m , y_{c2} -coordinate from $C_{1/2}$ to mass center [m]
5	r_{ix} , radius of gyration related to elastic center. Corresponds to rotation about principal bending x_e axis [m]
6	r_{iy} , radius of gyration related to elastic center. Corresponds to rotation about principal bending y_e axis [m]
7	x_s , x_{c2} -coordinate from $C_{1/2}$ to shear center [m]. The shear center is the point where external forces only contributes to pure bending and no torsion.
8	y_s , y_{c2} -coordinate from $C_{1/2}$ to shear center [m]. The shear center is the point where external forces only contributes to pure bending and no torsion.
9	E, modulus of elasticity [N/m^2]
10	G, shear modulus of elasticity [N/m^2]
11	I_x , area moment of inertia with respect to principal bending x_e axis [m^4]. This is the principal bending axis most parallel to the x_{c2} axis
12	I_y , area moment of inertia with respect to principal bending y_e axis [m^4]
13	K, torsional stiffness constant with respect to ze axis at the shear center [m^4/rad]. For a circular section only this is identical to the polar moment of inertia.
14	k_x shear factor for force in principal bending x_e direction [-]
15	k_y , shear factor for force in principal bending y_e direction [-]
16	A, cross sectional area [m^2]
17	θ_s , structural pitch about z_{c2} axis. This is the angle between the x_{c2} -axis defined with the c2_def command and the main principal bending axis x_e . [deg]
18	x_e , x_{c2} -coordinate from $C_{1/2}$ to center of elasticity [m]. The elastic center is the point where radial force (in the z-direction) does not contribute to bending around the x or y directions.
19	y_e , y_{c2} -coordinate from $C_{1/2}$ to center of elasticity [m]. The elastic center is the point where radial force (in the z-direction) does not contribute to bending around the x or y directions.

- **N.5** r_{ix} [m] Radius of inertia. Related to the Moment of Inertia I_{xx} [$kg\ m^2$], which gives the rotation inertia, resistance to change in rotation rate:

$$I_{xx} = \int r_{ix}^2 dm \rightarrow r = \sqrt{\frac{I_{xx}}{m}} = \sqrt{\frac{I_x}{A}}$$

- **N.11** I_x [m^4] Area moment of inertia with respect to x_e . It's the second moment of area $I_x = \int y^2 dA$. Multiplied by Young's modulus E gives the flapwise bending stiffness:

$$\text{Stiffn}_{\text{flap}} = E \cdot I_x = \frac{M}{d^2w/d^2x}$$

$$\text{Stiffn}_{\text{edge}} = E \cdot I_y$$

$$\text{Stiffn}_{\text{tors}} = G \cdot K$$

1 main data sets available

Here is space for comments etc

.
.
.

#1 Main data set number 1 - an example of a shaft structure

More comments space

r	m	x_cg	y_cg	ri_x	ri_y	x_sh	y_sh	E	G	I_x	I_y	K	k_x	k_y	A	theta_s	x_e	y_e
[m]	[kg/m]	[m]	[m]	[m]	[m]	[m]	[m]	[N/m ²]	[N/m ²]	[N/m ⁴]	[N/m ⁴]	[N/m ⁴]	[-]	[-]	[m ²]	[deg]	[m]	[m]
#1 10 Sub set number 1 with 10 data rows																		
0.00	100	0	0	224.18	224.18	0	0	2.10E+11	8.10E+10	1.00E+02	1.00E+02	0.05376	0.52	0.52	0.59	0	0.0	0.0
0.10	100	0	0	224.18	224.18	0	0	2.10E+11	8.10E+10	1.00E+02	1.00E+02	0.05376	0.52	0.52	0.59	0	0.0	0.0
0.1001	1	0	0	0.2	0.2	0	0	2.10E+11	8.10E+10	1.00E+02	1.00E+02	0.05376	0.52	0.52	0.59	0	0.0	0.0
1.00	1	0	0	0.2	0.2	0	0	2.10E+11	8.10E+10	1.00E+02	1.00E+02	0.05376	0.52	0.52	0.59	0	0.0	0.0
1.90	1	0	0	0.2	0.2	0	0	2.10E+11	8.10E+10	1.00E+02	1.00E+02	0.05376	0.52	0.52	0.59	0	0.0	0.0
2.00	1	0	0	0.2	0.2	0	0	2.10E+11	8.10E+10	1.00E+02	1.00E+02	0.05376	0.52	0.52	0.59	0	0.0	0.0
3.00	1	0	0	0.2	0.2	0	0	2.10E+11	8.10E+10	1.00E+02	1.00E+02	0.05376	0.52	0.52	0.59	0	0.0	0.0
3.20	1	0	0	0.2	0.2	0	0	2.10E+11	8.10E+10	1.00E+02	1.00E+02	0.05376	0.52	0.52	0.59	0	0.0	0.0
4.00	1	0	0	0.2	0.2	0	0	2.10E+11	8.10E+10	1.00E+02	1.00E+02	0.05376	0.52	0.52	0.59	0	0.0	0.0
5.0191	1	0	0	0.2	0.2	0	0	2.10E+11	8.10E+10	1.00E+02	1.00E+02	0.05376	0.52	0.52	0.59	0	0.0	0.0

More comments space

r	m	x_cg	y_cg	ri_x	ri_y	x_sh	y_sh	E	G	I_x	I_y	K	k_x	k_y	A	theta_s	x_e	y_e
[m]	[kg/m]	[m]	[m]	[m]	[m]	[m]	[m]	[N/m ²]	[N/m ²]	[N/m ⁴]	[N/m ⁴]	[N/m ⁴]	[-]	[-]	[m ²]	[deg]	[m]	[m]
#2 10 As dataset 1, but stiff																		
0.00	100	0	0	224.18	224.18	0	0	2.10E+16	8.10E+15	1.00E+02	1.00E+02	0.05376	0.52	0.52	0.59	0	0.0	0.0
0.10	100	0	0	224.18	224.18	0	0	2.10E+16	8.10E+15	1.00E+02	1.00E+02	0.05376	0.52	0.52	0.59	0	0.0	0.0
0.1001	1	0	0	0.2	0.2	0	0	2.10E+16	8.10E+15	1.00E+02	1.00E+02	0.05376	0.52	0.52	0.59	0	0.0	0.0
1.00	1	0	0	0.2	0.2	0	0	2.10E+16	8.10E+15	1.00E+02	1.00E+02	0.05376	0.52	0.52	0.59	0	0.0	0.0
1.90	1	0	0	0.2	0.2	0	0	2.10E+16	8.10E+15	1.00E+02	1.00E+02	0.05376	0.52	0.52	0.59	0	0.0	0.0
2.00	1	0	0	0.2	0.2	0	0	2.10E+16	8.10E+15	1.00E+02	1.00E+02	0.05376	0.52	0.52	0.59	0	0.0	0.0
3.00	1	0	0	0.2	0.2	0	0	2.10E+16	8.10E+15	1.00E+02	1.00E+02	0.05376	0.52	0.52	0.59	0	0.0	0.0
3.20	1	0	0	0.2	0.2	0	0	2.10E+16	8.10E+15	1.00E+02	1.00E+02	0.05376	0.52	0.52	0.59	0	0.0	0.0
4.00	1	0	0	0.2	0.2	0	0	2.10E+16	8.10E+15	1.00E+02	1.00E+02	0.05376	0.52	0.52	0.59	0	0.0	0.0
5.0191	1	0	0	0.2	0.2	0	0	2.10E+16	8.10E+15	1.00E+02	1.00E+02	0.05376	0.52	0.52	0.59	0	0.0	0.0

More comments space

r	m	x_cg	y_cg	ri_x	ri_y	x_sh	y_sh	E	G	I_x	I_y	K	k_x	k_y	A	theta_s	x_e	y_e
[m]	[kg/m]	[m]	[m]	[m]	[m]	[m]	[m]	[N/m ²]	[N/m ²]	[N/m ⁴]	[N/m ⁴]	[N/m ⁴]	[-]	[-]	[m ²]	[deg]	[m]	[m]
#3 10 as data set 1 but changed mass properties																		
0.00	1000	0	0	2.2418	2.2418	0	0	2.10E+11	8.10E+10	1.00E+02	1.00E+02	0.05376	0.52	0.52	0.59	0	0.0	0.0
0.10	1000	0	0	2.2418	2.2418	0	0	2.10E+11	8.10E+10	1.00E+02	1.00E+02	0.05376	0.52	0.52	0.59	0	0.0	0.0
0.1001	1	0	0	0.2	0.2	0	0	2.10E+11	8.10E+10	1.00E+02	1.00E+02	0.05376	0.52	0.52	0.59	0	0.0	0.0
1.00	1	0	0	0.2	0.2	0	0	2.10E+11	8.10E+10	1.00E+02	1.00E+02	0.05376	0.52	0.52	0.59	0	0.0	0.0
1.90	1	0	0	0.2	0.2	0	0	2.10E+11	8.10E+10	1.00E+02	1.00E+02	0.05376	0.52	0.52	0.59	0	0.0	0.0
2.00	1	0	0	0.2	0.2	0	0	2.10E+11	8.10E+10	1.00E+02	1.00E+02	0.05376	0.52	0.52	0.59	0	0.0	0.0
3.00	1	0	0	0.2	0.2	0	0	2.10E+11	8.10E+10	1.00E+02	1.00E+02	0.05376	0.52	0.52	0.59	0	0.0	0.0
3.20	1	0	0	0.2	0.2	0	0	2.10E+11	8.10E+10	1.00E+02	1.00E+02	0.05376	0.52	0.52	0.59	0	0.0	0.0
4.00	1	0	0	0.2	0.2	0	0	2.10E+11	8.10E+10	1.00E+02	1.00E+02	0.05376	0.52	0.52	0.59	0	0.0	0.0
5.0191	1	0	0	0.2	0.2	0	0	2.10E+11	8.10E+10	1.00E+02	1.00E+02	0.05376	0.52	0.52	0.59	0	0.0	0.0

Table 2: New HAWC2 anisotropic beam element structural data

Column	
1.0	r , curved length distance from main_body node 1 [m]
2.0	m , mass per unit length [kg/m]
3.0	x_m, x_{c2} -coordinate from $C_{1/2}$ to mass center [m]
4.0	y_m, y_{c2} -coordinate from $C_{1/2}$ to mass center [m]
5.0	r_{ix} , radius of gyration related to elastic center. Corresponds to rotation about principal bending x_e axis [m]
6.0	r_{iy} , radius of gyration related to elastic center. Corresponds to rotation about principal bending y_e axis [m]
7.0	θ_s , structural pitch about z_{c2} axis. This is the angle between the x_{c2} -axis defined with the $c2_def$ command and the main principal bending axis x_e .
8.0	x_e, x_{c2} -coordinate from $C_{1/2}$ to center of elasticity [m]. The elastic center is the point where radial force (in the z-direction) does not contribute to bending around the x or y directions.
9.0	y_e, y_{c2} -coordinate from $C_{1/2}$ to center of elasticity [m]. The elastic center is the point where radial force (in the z-direction) does not contribute to bending around the x or y directions.
10.0	K_{11} , element 1,1 of the Cross sectional stiffness matrix [Nm^2]. REMEMBER: the cross sectional stiffness matrix is given at the elastic center rotated along the principal bending axes.
11.0	K_{12} , element 1,2 of the Cross sectional stiffness matrix [Nm^2].
12.0	K_{13} , element 1,3 of the Cross sectional stiffness matrix [Nm^2].
13.0	K_{14} , element 1,4 of the Cross sectional stiffness matrix [Nm^2].
14.0	K_{15} , element 1,5 of the Cross sectional stiffness matrix [Nm^2].
15.0	K_{16} , element 1,6 of the Cross sectional stiffness matrix [Nm^2].
16.0	K_{22} , element 2,2 of the Cross sectional stiffness matrix [Nm^2].
17.0	K_{23} , element 2,3 of the Cross sectional stiffness matrix [Nm^2].
18.0	K_{24} , element 2,4 of the Cross sectional stiffness matrix [Nm^2].
19.0	K_{25} , element 2,5 of the Cross sectional stiffness matrix [Nm^2].
20.0	K_{26} , element 2,6 of the Cross sectional stiffness matrix [Nm^2].
21.0	K_{33} , element 3,3 of the Cross sectional stiffness matrix [Nm^2].
22.0	K_{34} , element 3,4 of the Cross sectional stiffness matrix [Nm^2].
23.0	K_{35} , element 3,5 of the Cross sectional stiffness matrix [Nm^2].
24.0	K_{36} , element 3,6 of the Cross sectional stiffness matrix [Nm^2].
25.0	K_{44} , element 4,4 of the Cross sectional stiffness matrix [Nm^2].
26.0	K_{45} , element 4,5 of the Cross sectional stiffness matrix [Nm^2].
27.0	K_{46} , element 4,6 of the Cross sectional stiffness matrix [Nm^2].
28.0	K_{55} , element 5,5 of the Cross sectional stiffness matrix [Nm^2].
29.0	K_{56} , element 5,6 of the Cross sectional stiffness matrix [Nm^2].
30.0	K_{66} , element 6,6 of the Cross sectional stiffness matrix [Nm^2].

An example of a st anisotropic beam formulation input file can be seen on the next page.

7.2.3 Sub sub command - damping_distributed

In this command block, Rayleigh damping parameters can be defined as function of blade length, hence damping parameters can be different at root of tip of a blade.

Obl.	Command name	Explanation
*	nsec	Number of input lines
*	sec	This command must be repeated nsec times. 1. r/R . Non-dim distance from node 1 to node N. 2. k_x Stiffness proportional damping around x 3. k_y Stiffness proportional damping around y 4. k_z Stiffness proportional damping around z

7.2.4 Sub sub command – damping_posdef_distributed

In this command block, Rayleigh damping parameters can be defined as function of blade length, hence damping parameters can be different at root of tip of a blade.

Obl.	Command name	Explanation
*	nsec	Number of input lines
*	sec	This command must be repeated nsec times. 1. r/R . Non-dim distance from node 1 to node N. 2. k_x Stiffness proportional damping around x 3. k_y Stiffness proportional damping around y 4. k_z Stiffness proportional damping around z

7.2.5 Sub sub command – visualization_profile

This command block is used together with the command name visualization in the main command block simulation. Default profiles are:

- Blade: An aerodynamic profile where thickness <95%, otherwise a cylinder. Dimensions as specified in the aerodynamic blade layout file.
- Other bodies: Cylinder. The diameter is calculated from the mass and inertia specified in the structural data

Obl.	Command name	Explanation
*	type	Profile type. (options are: “cylinder”, “cube” and “blade”)
*	nsec	Number of visualization sections
*	sec	This command must be repeated nsec times. 1. Distance from root [m or % or any other unit of choice (scaled relative to the largest number)] 2. Diameter (cylinder), width (cube), chord (blade) [m] 3. (not needed for cylinder), height (cube) [m], thickness (blade) [%]

7.3 Sub command - orientation

In this command block the orientation (regarding position and rotation) of every main_body are specified.

7.3.1 Sub sub command - base

The orientation of a main_body to which all other bodies are linked – directly or indirectly.

Obl.	Command name	Explanation
*	mbdy (old command name body still usable)	1. Main_body name that is declared to be the base of all bodies (normally the tower or foundation)
*	inipos	Initial position in global coordinates. 1. x-pos [m] 2. y-pos [m] 3. z-pos [m]
♣	mbody_eulerang (old command name body_eulerang still usable)	Command that can be repeated as many times as needed. All following rotation are given as a sequence of euler angle rotations. All angle can be filled in (rotation order x,y,z), but it is recommended only to give a value different from zero on one of the angles and reuse the command if several rotations are needed. 1. θ_x [deg] 2. θ_y [deg] 3. θ_z [deg]
♣	mbody_eulerpar (old command name body_eulerpar still usable)	The rotation is given as euler parameters (quaternions) directly (global coo). 1. r_0 2. r_1 3. r_2 4. r_3
♣	mbody_axisangle (old command name body_axisangle still usable)	Command that can be repeated as many times as needed. A version of the euler parameters where the input is a rotation vector and the rotation angle of this vector. 1. x-value 2. y-value 3. z-value 4. angle [deg]
	mbody_ini_rotvec_d1	Initial rotation velocity of main body and all subsequent attached bodies. A rotation vector is set up and the size of vector (the rotational speed) is given. The coordinate system used is main_body coo. 1. x-value 2. y-value 3. z-value 4. Vector size (rotational speed [rad/s])

♣ One of these commands must be present.

7.3.2 Sub sub command - relative

This command block can be repeated as many times as needed. However the orientation of every main_body should be described.

Obl.	Command name	Explanation
*	mbdy1 (old command name body1 still usable)	1. Main_body name to which the next main_body is attached. 2. Node number of body1 that is used for connection. ("last" can be specified which ensures that the last node on the main_body is used).
*	mbdy2 (old command name body2 still usable)	1. Main_body name of the main_body that is positioned in space by the relative command. 2. Node number of body2 that is used for connection. ("last" can be specified which ensures that the last node on the main_body is used).
♣	mbdy2_eulerang (old command name body2_eulerang still usable)	Command that can be repeated as many times as needed. All following rotation are given as a sequence of euler angle rotations. All angle can be filled in (rotation order x,y,z), but it is recommended only to give a value different from zero on one of the angles and reuse the command if several rotations are needed. Until a rotation command is specified body2 has same coo. as body1. Rotations are performed in the present body2 coo. system. 1. θ_x [deg] 2. θ_y [deg] 3. θ_z [deg]
♣	mbdy2_eulerpar (old command name body2_eulerpar still usable)	The rotation is given as euler parameters (quaternions) directly (global coo). 1. r_0 2. r_1 3. r_2 4. r_3
♣	mbdy2_axisangle (old command name body2_axisangle still usable)	Command that can be repeated as many times as needed. A version of the euler parameters where the input is a rotation vector and the rotation angle of this vector. Until a rotation command is specified main_body2 has same coo. as main_body1. Rotations are performed in the present main_body2 coo. system. 1. x-value 2. y-value 3. z-value 4. angle [deg]

	mbdy2_ini_rotvec_d1 (old command name body2_ini_rotvec_d1 still usable)	Initial rotation velocity of main body and all subsequent attached bodies. A rotation vector is set up and the size of vector (the rotational speed) is given. The coordinate system used is main_body2 coo. 1. x-value 2. y-value 3. z-value 4. Vector size (rotational speed [rad/s])
	relpos	Vector from coupling node of mbdy 1 to coupling node of mbdy 2 in mbdy2 coo system in case a certain distance between these nodes is required. (Default for overlapping coupling nodes, this vector is (0,0,0)) 1. x-value 2. y-value 3. z-value

7.4 Sub command - constraint

In this block constraints between the main_bodies and to the global coordinate system are defined.

7.4.1 Sub sub command – fix0

This constraint fix node number 1 of a given main_body to ground.

Obl.	Command name	Explanation
*	mbdy (old command name body still usable)	Name of main body that is fixed to ground at node 1
	disable_at	Time to which constraint can be disabled 1. t_0
	enable_at	Time to which constraint can be enabled 1. t_0

7.4.2 Sub sub command – fix1

This constraint fix a given node on one main_body to another main_body's node.

Obl.	Command name	Explanation
*	mbdy1 (old command name body1 still usable)	1. Main_body name to which the next main_body is fixed. 2. Node number of main_body1 that is used for the constraint. ("last" can be specified which ensures that the last node on the main_body is used).
*	mbdy2 (old command name body2 still usable)	1. Main_body name of the main_body that is fixed to main_body1. 2. Node number of main_body2 that is used for the constraint. ("last" can be specified which ensures that the last node on the main_body is used).
	disable_at	Time to which constraint can be disabled 1. t_0
	enable_at	Time to which constraint can be enabled 1. t_0

7.4.3 Sub sub command – fix2

This constraint fix a node 1 on a main_body to ground in x,y,z direction. The direction that is free or fixed is optional.

Obl.	Command name	Explanation
*	mbdy (old command name body still usable)	1. Main_body name to which node 1 is fixed.
*	dof	Direction in global coo that is fixed in translation 1. x-direction (0=free, 1=fixed) 2. y-direction (0=free, 1=fixed) 3. z-direction (0=free, 1=fixed)

7.4.4 Sub sub command – fix3

This constraint fix a node to ground in t_x, t_y, t_z rotation direction. The rotation direction that is free or fixed is optional.

Obl.	Command name	Explanation
*	mbdy (old command name body still usable)	1. Main_body name to which node 1 is fixed. 2. Node number
*	dof	Direction in global coo that is fixed in rotation 1. tx-rot.direction (0=free, 1=fixed) 2. ty-rot.direction (0=free, 1=fixed) 3. tz-rot.direction (0=free, 1=fixed)

7.4.5 Sub sub command – fix4

Constraint that locks a node on a body to another node in translation but not rotation with a pre-stress feature. The two nodes will start at the defined positions to begin with but narrow the distance until fully attached at time T.

Obl.	Command name	Explanation
*	mbody1 (old command name body1 still usable)	1. Main_body name to which the next main_body is fixed. 2. Node number of main_body1 that is used for the constraint. ("last" can be specified which ensures that the last node on the main_body is used).
*	mbody2 (old command name body2 still usable)	1. Main_body name of the main_body that is fixed to body1. 2. Node number of main_body2 that is used for the constraint. ("last" can be specified which ensures that the last node on the main_body is used).
	time	1. Time for the pre-stress process. Default=2sec
	disable_at	Time to which constraint can be disabled 1. t_0
	enable_at	Time to which constraint can be enabled 1. t_0

7.4.6 Sub sub command – bearing1

Constraint with properties as a bearing without friction. A sensor with same identification name as the constraint is set up for output purpose.

Obl.	Command name	Explanation
*	name	1. Identification name
*	mbody1 (old command name body1 still usable)	1. Main_body name to which the next main_body is fixed with bearing1 properties. 2. Node number of main_body1 that is used for the constraint. ("last" can be specified which ensures that the last node on the main_body is used).
*	mbody2 (old command name body2 still usable)	1. Main_body name of the main_body that is fixed to body1 with bearing1 properties. 2. Node number of main_body2 that is used for the constraint. ("last" can be specified which ensures that the last node on the main_body is used).
*	bearing_vector	Vector to which the free rotation is possible. The direction of this vector also defines the coo to which the output angle is defined. 1. Coo. system used for vector definition (0=global,1=mbody1,2=mbody2) 2. x-axis 3. y-axis 4. z-axis
	disable_at	Time to which constraint can be disabled 1. t_0
	enable_at	Time to which constraint can be enabled 1. t_0

7.4.7 Sub sub command – bearing2

This constraint allows a rotation where the angle is directly specified by an external dll action command.

Obl.	Command name	Explanation
*	name	1. Identification name
*	mbdy1 (old command name body1 still usable)	1. Main_body name to which the next main_body is fixed with bearing2 properties. 2. Node number of main_body1 that is used for the constraint. (“last” can be specified which ensures that the last node on the main_body is used).
*	mbdy2 (old command name body2 still usable)	1. Main_body name of the main_body that is fixed to main_body1 with bearing1 properties. 2. Node number of main_body2 that is used for the constraint. (“last” can be specified which ensures that the last node on the main_body is used).
*	bearing_vector	Vector to which the rotation occur. The direction of this vector also defines the coo to which the output angle is defined. 1. Coor. system used for vector definition (0=global, 1=mbdy1, 2=mbdy2) 2. x-axis 3. y-axis 4. z-axis
	disable_at	Time to which constraint can be disabled 1. t_0
	enable_at	Time to which constraint can be enabled 1. t_0

7.4.8 Sub sub command – bearing3

This constraint allows a rotation where the angle velocity is kept constant throughout the simulation.

Obl.	Command name	Explanation
*	name	1. Identification name
*	mbdy1 (old command name body1 still usable)	1. Main_body name to which the next main_body is fixed with bearing3 properties. 2. Node number of main_body1 that is used for the constraint. ("last" can be specified which ensures that the last node on the main_body is used).
*	mbdy2 (old command name body2 still usable)	1. Main_body name of the main_body that is fixed to body1 with bearing3 properties. 2. Node number of main_body2 that is used for the constraint. ("last" can be specified which ensures that the last node on the main_body is used).
*	bearing_vector	Vector to which the rotation occur. The direction of this vector also defines the coo to which the output angle is defined. 1. Coor. system used for vector definition (0=global,1=body1,2=body2) 2. x-axis 3. y-axis 4. z-axis
*	omegas	1. Rotational speed [rad/sec]

7.4.9 Sub sub command – bearing4

This constraint is a cardan shaft constraint. Locked in relative translation. Locked in rotation around one vector and allows rotation about the two other directions.

Obl.	Command name	Explanation
*	name	1. Identification name
*	mbdy1 (old command name body1 still usable)	1. Main_body name to which the next main_body is fixed with bearing3 properties. 2. Node number of main_body1 that is used for the constraint. ("last" can be specified which ensures that the last node on the main_body is used).
*	mbdy2 (old command name body2 still usable)	1. Main_body name of the main_body that is fixed to body1 with bearing3 properties. 2. Node number of main_body2 that is used for the constraint. ("last" can be specified which ensures that the last node on the main_body is used).
*	bearing_vector	Vector to which the rotation is locked. The rotation angle and velocity can be outputted around the two perpendicular directions. 1. Coor. system used for vector definition (0=global,1=mbdy1, 2=mbdy2) 2. x-axis 3. y-axis 4. z-axis

7.4.10 Sub sub command – bearing5

This constraint is a spherical constraint. Locked in relative translation. Free in rotation around all three axis, but only sensor on the main rotation direction.

Obl.	Command name	Explanation
*	name	1. Identification name
*	mbdy1 (old command name body1 still usable)	1. Main_body name to which the next main_body is fixed with bearing3 properties. 2. Node number of main_body1 that is used for the constraint. ("last" can be specified which ensures that the last node on the main_body is used).
*	mbdy2 (old command name body2 still usable)	1. Main_body name of the main_body that is fixed to body1 with bearing3 properties. 2. Node number of main_body2 that is used for the constraint. ("last" can be specified which ensures that the last node on the main_body is used).
*	bearing_vector	Vector to which the rotation is locked. The rotation angle and velocity can be outputted around the two perpendicular directions. 1. Coor. system used for vector definition (0=global,1=mbdy1, 2=mbdy2) 2. x-axis 3. y-axis 4. z-axis

8 DLL control

This block contains the possible Dynamic Link Library formats accessible for the user. The DLL's are mainly used to control the turbine speed and pitch, but since the DLL format is very general, other use is possible too e.g. external loading of the turbine. Since the HAWC2 core has no information about external stiffness or inertia we have experienced some issues with the solver if the DLL includes high stiffness terms or especially large inertia terms. The new type2_dll interface is slightly more stable related to the solver than the hawc_dll interface.

8.1 Main command block – dll

So far only one DLL format is available, which is the hawc_dll format listed below.

8.2 Sub command block – hawc_dll

In the HAWC_DLL format a subroutine within an externally written DLL is setup. In this subroutine call two one-dimensional arrays are transferred between the HAWC2 core and the DLL procedure. The first contains data going from the HAWC2 core to the DLL and the other contains data going from the DLL to the core. It is very important to notice that the data are transferred between HAWC2 and the DLL in every timestep and every iteration. The user should handle the iteration inside the DLL.

Two more subroutines are called if they are present inside the dll file:

The first is an initialisation call including a text string written in the init_string in the commands below. This could be the name of a file holding local input parameters to the data transfer subroutine. This call is only performed once. The name of this subroutine is the same name as the data transfer subroutine defined with the command dll_subroutine below with the extra name '_init', hence is the data transfer subroutine is called 'test', the initialisation subroutine will be 'test_init'.

The second subroutine is a message exchange subroutine, where messages written in the DLL can be send to the HAWC2 core for logfile writing. The name of this subroutine is the same name as the data transfer subroutine defined with the command dll_subroutine below with the extra name '_message', hence is the data transfer subroutine is called 'test', the initialisation subroutine will be 'test_message'.

The command block can be repeated as many times as desired. Reference number to DLL is same order as listed, starting with number 1. However it is recommended to refer the DLL using the name feature which in many cases can avoid confusion.

Obl.	Command name	Explanation
	name	1. Reference name of this DLL (to be used with DLL output commands)
*	filename	1. Filename incl. relative path of the DLL (example ./DLL/control.dll)
*	dll_subroutine	1. Name of subroutine in DLL that is addressed (remember to specify the name in the DLL with small letters!)
*	arraysizes	1. size of array with outgoing data 2. size of array with ingoing data
	deltat	1. Time between dll calls. Must correspond to the simulation sample frequency or be a multiple of the time step size. If deltat=0.0 or the deltat command line is omitted the HAWC2 code calls the dll subroutine at every time step.
	init_string	1. Text string (max 256 characters) that will be transferred to the DLL through the subroutine 'subroutine_init'. Subroutine is the name given in in the command dll_subroutine. No blanks can be included.

8.3 Sub command block – type2_dll

This dll interface is an updated slightly modified version of the hawc_dll interface. In the TYPE2_DLL format a subroutine within an externally written DLL is setup. In this subroutine call two one-dimensional arrays are transferred between the HAWC2 core and the DLL procedure. The first contains data going from the HAWC2 core to the DLL and the other contains data going from the DLL to the core. It is very important to notice that the data are transferred between HAWC2 and the DLL in the first call of every timestep where the out-going variables are based on last iterated values from previous time step. The sub command output and actions are identical for both the hawc_dll and the type2_dll interfaces.

In the dll connected with using the type2_dll interface two subroutines should be present. An initialization routine called only once before the time simulation begins, and an update routine called in every time step. The format in the calling of these two subroutines are identical where two arrays of double precision is exchanged. The subroutine uses the cdecl calling convention.

Obl.	Command name	Explanation
	name	1. Reference name of this DLL (to be used with DLL output commands)
*	filename	1. Filename incl. relative path of the DLL (example ./DLL/control.dll)
*	dll_subroutine_init	1. Name of initialization subroutine in DLL that is addressed (remember to specify the name in the DLL with small letters!)
*	dll_subroutine_update	1. Name of subroutine in DLL that is addressed at every time step (remember to specify the name in the DLL with small letters!)
*	arraysizes_init	1. size of array with outgoing data in the initialization call 2. size of array with ingoing data in the initialization call
*	arraysizes_update	1. size of array with outgoing data in the update call 2. size of array with ingoing data in the update call
	deltat	1. Time between dll calls. Must correspond to the simulation sample frequency or be a multiple of the time step size. If deltat=0.0 or the deltat command line is omitted the HAWC2 code calls the dll subroutine at every time step.

when using the type2_dll interface the values transferred to the DLL in the initialization phase is done using a sub command block called init. The commands for this subcommand block is identical to the output subcommand explained below, but only has the option of having the constant output sensor available. An example is given for a small dll that is used for converting rotational speed between high speed and low speed side of a gearbox.:

```
begin dll;
  begin type2_dll;
    name hss_convert;
    filename ./control/hss_convert.dll ;
    arraysizes_init 3 1 ;
    arraysizes_update 2 2 ;
    begin init;
      constant 1 2.0 ;      number of used sensors - in this case only 1
      constant 2 35.110;   gearbox ratio
      constant 3 35.110;   gearbox ratio
    end init;
    begin output;
      constraint bearing1 shaft_rot 2 only 2 ;   rotor speed in rpm
      constraint bearing1 shaft_rot 3 only 2 ;   rotor speed in rad/s
    end output;
  ;
  begin actions;
  ;   rotor speed in rpm * gear_ratio
  ;   rotor speed in rad/s * gear_ratio
  end actions;
  end type2_dll;
end dll;
```

8.4 Sub command block – init

In this block type2_dlls can be initialized by passing constants to specific channels.

Obl.	Command name	Explanation
*	constant	Constants passed to the dll. 1. Channel number 2. Constant value

8.5 Sub command block – output

In this block the same sensors are available as when data results are written to a file with the main block command output, see section 15. The order of the sensors in the data array is continuously increased as more sensors are added.

8.6 Sub command block – actions

In this command block variables inside the HAWC2 code is changed depending of the specifications. This command block can be used for the hawc_dll interface as well as the type2_dll interface. An action commands creates a handle to the HAWC2 model to which a variable in the input array from the DLL is linked.

!NB in the command name two separate words are present.

Obl.	Command name	Explanation
	aero beta	The flap angle beta is set for a trailing edge flap section (is the mhhmagf stall model is used). The angle is positive towards the pressure side of the profile. Unit is [deg] 1. Blade number 2. Flap section number
	aero bem_grid_a	1. Number of points
	body force_ext	An external force is placed on the structure. Unit is [N]. 1. body name 2. node number 3. componet (1 = Fx, 2 = Fy, 3 = Fz)
	body moment_ext	An external moment is placed on the structure. Unit is [Nm]. 1. body name 2. node number 3. component (1 = Mx, 2 = My, 3 = Mz)
	body force_int	An external force with a reaction component is placed on the structure. Unit is [N]. 1. body name for action force 2. node number 3. component (1 = Fx, 2 = Fy, 3 = Fz) 4. body name for reaction force 5. Node number
	body moment_int	An external moment with a reaction component is placed on the structure. Unit is [N]. 1. body name for action moment 2. node number 3. component (1 = Mx, 2 = My, 3 = Mz) 4. body name for reaction moment 5. Node number
	body bearing_angle	A bearing either defined through the new structure format through bearing2 or through the old structure format (spitch1=pitch angle for blade 1, spitch2=pitch angle for blade 2,...). The angle limits are so far [0-90deg]. 1. Bearing name
	mbdy force_ext	An external force is placed on the structure. Unit is [N]. 1. main body name 2. node number on main body 3. component (1 = Fx, 2 = Fy, 3 = Fz) , if negative number the force is inserted with opposite sign. 4. coordinate system (possible options are: mbdy name,"global","local"). "local" means local element coo on the inner element (on the element indexed 1 lower that the node number). One exception if node number =1 then the element nr. also equals 1.

	mbdy moment_ext	<p>An external moment is placed on the structure. Unit is [Nm].</p> <ol style="list-style-type: none"> 1. main body name 2. node number on main body 3. component (1 = M_x, 2 = M_y, 3 = M_z) , if negative number the moment is inserted with opposite sign. 4. coordinate system (possible options are: mbdy name,"global","local"). "local" means local element coo on the inner element (on the element indexed 1 lower that the node number). One exception if node number =1 then the element nr. also equals 1.
	mbdy force_int	<p>An internal force with a reaction component is placed on the structure. Unit is [N].</p> <ol style="list-style-type: none"> 1. main body name for action force 2. node number on main body 3. component (1 = F_x, 2 = F_y, 3 = F_z) , if negative number the force is inserted with opposite sign. 4. coordinate system (possible options are: mbdy name,"global","local"). "local" means local element coo on the inner element (on the element indexed 1 lower that the node number). One exception if node number =1 then the element nr. also equals 1. 5. main body name for reaction force 6. Node number on this main body
	mbdy moment_int	<p>An internal force with a reaction component is placed on the structure. Unit is [Nm].</p> <ol style="list-style-type: none"> 1. main body name for action moment 2. node number on main body 3. component (1 = M_x, 2 = M_y, 3 = M_z) , if negative number the moment is inserted with opposite sign. 4. coordinate system (possible options are: mbdy name,"global","local"). "local" means local element coo on the inner element (on the element indexed 1 lower that the node number). One exception if node number =1 then the element nr. also equals 1. 5. main body name for reaction moment 6. Node number on this main body
	constraint bearing2 angle_deg	<p>The angle of a bearing2 constraint is set. The angle limits are so far [+/-90deg].</p> <ol style="list-style-type: none"> 1. Bearing name
	constraint bearing3 angle_deg	<p>The angle of a bearing3 constraint is set. The angle limits are so far [+/-90deg].</p> <ol style="list-style-type: none"> 1. Bearing name
	constraint bearing3 omegas	<p>The angular velocity of a bearing3 constraint is set.</p> <ol style="list-style-type: none"> 1. Bearing name
	body printvar	Variable is just echoed on the screen. No parameters.
	body ignore	1. Number of consecutive array spaces that will be ignored
	mbdy printvar	Variable is just echoed on the screen. No parameters.
	mbdy ignore	1. Number of consecutive array spaces that will be ignored
	general printvar	Variable is just echoed on the screen. No parameters.
	general ignore	1. Number of consecutive array spaces that will be ignored

	stop_simulation	Logical switch. If value is 1 the simulation will be stopped and output written.
	wind printvar	Variable is just echoed on the screen. No parameters.
	wind windspeed_u	External contribution to wind speed in u-direction [m/s]
	wind winddir	External contribution to the wind direction (turb. box is also rotated) [°]
	quake comp	1. Degree of freedom
	ext_sys control	1. Name of external system

8.7 HAWC_DLL format example written in FORTRAN 90

```
subroutine test(n1,array1,n2,array2)
implicit none
!DEC$ ATTRIBUTES DLLEXPORT, ALIAS:'test'::test
integer*4      :: n1, &    ! Dummy integer value containing the array size of array1
                n2        ! Dummy integer value containing the array size of array2
real*4,dimension(10) :: array1 ! fixed-length array, data from HAWC2 to DLL
                                ! - in this case with length 10
real*4,dimension(5)  :: array2 ! fixed-length array, data from DLL to HAWC2
                                ! - in this case with length 5

! Code is written here

end subroutine test

!-----

Subroutine test_init(string256)
Implicit none
!DEC$ ATTRIBUTES DLLEXPORT, ALIAS:'test_init'::test_init
Character*256 :: string256

! Code is written here

End subroutine test_init

!-----

Subroutine test_message(string256)
Implicit none
!DEC$ ATTRIBUTES DLLEXPORT, ALIAS:'test_message'::test_message
Character*256 :: string256

! Code is written here

End subroutine test_message
```

8.8 HAWC_DLL format example written in Delphi / Lazarus / Pascal

```
library test_dll;

type
  array_10 = array[1..10] of single;
  array_5  = array[1..5] of single;
  ts       = array[0..255] of char;

Procedure test(var n1:integer;var array1 : array_10;
              var n2:integer;var array2 : array_5);stdcall;
// n1 is a dummy integer value containing the size of array1
// n2 is a dummy integer value containing the size of array2
begin
  // Code is written here

end;

//-----

Procedure test_init(var string256:ts; length:integer);stdcall;
var
  init_str:string[255]
begin
  init_str=strpas(string256);
  // Code is written here
  writeln(init_str);
end;

//-----

Procedure test_message(var string256:ts; length:integer);stdcall;
var
  message_str:string;
begin
  // Code is written here
  message_str:='This is a test message';
  strPCopy(string256,message_str);
end;

exports test,test_init,test_message;

begin
  writeln('The DLL pitchservo.dll is loaded with succes');

  // Initialization of variables can be performed here
end;

end.
```

8.9 HAWC_DLL format example written in C

```
extern "C" void __declspec(dllexport) __cdecl test(int &size_of_Data_in,
float Data_in[], int &size_of_Data_out, float Data_out[])
{
    for (int i=0; i<size_of_Data_out; i++) Data_out[i]=0.0;
    //
    printf("size_of_Data_in  %d: \n",size_of_Data_in);
    printf("Data_in          %g: \n",Data_in[0]);
    printf("size_of_Data_out  %d: \n",size_of_Data_out);
    printf("Data_out          %g: \n",Data_out[0]);
}

extern "C" void __declspec(dllexport) __cdecl test_init(char* pString, int length)
{
    // Define buffer (make room for NULL-char)
    const int max_length = 256;
    char buffer[max_length+1];
    //
    // Print the length of pString
    printf("test_init::length = %d\n",length);
    //
    // Transfer string
    int nchar = min(max_length, length);
    memcpy(buffer, pString, nchar);
    //
    // Add NULL-char
    buffer[nchar] = '\0';
    //
    // Print it...
    printf("%s\n",buffer);
}

extern "C" void __declspec(dllexport) __cdecl test_message(char* pString, int max_length)
{
    // test message (larger than max_length)
    char pmessage[] = "This is a test message "
        "and it continues and it continues and it continues "
        "and it continues and it continues and it continues "
        "and it continues and it continues and it continues "
        "and it continues and it continues and it continues "
        "and it continues and it continues and it continues "
        "and it continues and it continues and it continues ";

    // Check max length - transfer only up to max_length number of chars
    int nchar = min((size_t)max_length, strlen(pmessage)); // nof chars to transfer
        // (<= max_length)
    memcpy(pString, pmessage, nchar);
    //
    // Add NULL-char if string space allows it (FORTRAN interprets a NULL-char as
    // the end of the string)
    if (nchar < max_length) pString[nchar] = '\0';
}
```

8.10 TYPE2_dll written in Delphi / Lazarus / Delphi

```
library hss_convert;

uses
  SysUtils,
  Classes,
  Dialogs;

Type
  array_1000 = array[0..999] of double;
Var
  factor : array of double;
  nr : integer;
  {$R *.res}

procedure initialize(var InputSignals: array_1000;var OutputSignals: array_1000); cdecl;
var
  i : integer;
begin
  nr:=trunc(inputsignals[0]);
  if nr>0 then begin
    setlength(factor,nr);
    for i:=1 to nr do
      factor[i-1]:=Inputsignals[i];
    outputsignals[0]:=1.0;
  end else outputsignals[0]:=0.0;
end;

procedure update(var InputSignals: array_1000;var OutputSignals: array_1000); cdecl;
var
  i : integer;
begin
  for i:=0 to nr-1 do begin
    OutputSignals[i] := InputSignals[i]*factor[i];
  end;
end;

exports Initialize,Update;

begin
  // Main body

end.
```

8.11 TYPE2_dll written in C

```
extern "C" void __declspec(dllexport) __cdecl initialize(dfloat *Data_in, dfloat *Data_out)
{ for (int i=0; i<8; i++) Data_out[0]+=Data_in[i];
}
```

```
extern "C" void __declspec(dllexport) __cdecl update(dfloat *Data_in, dfloat *Data_out)
{ for (int i=0; i<25; i++) Data_out[0]+=Data_in[i];
  Data_out[8]=123;
}
```

8.12 TYPE2_DLL format example written in FORTRAN 90

```
subroutine initialize(array1,array2)
implicit none
!DEC$ ATTRIBUTES DLLEXPORT, C, ALIAS:'initialize'::initialize
real*8,dimension(1000) :: array1 ! fixed-length array, data from HAWC2 to DLL
! - in this case with length 1000
real*8,dimension(1)    :: array2 ! fixed-length array, data from DLL to HAWC2
! - in this case with length 1

! Code is written here

end subroutine initialize

!-----

subroutine update(array1,array2)
implicit none
!DEC$ ATTRIBUTES DLLEXPORT, C, ALIAS:'update'::update
real*8,dimension(1000) :: array1 ! fixed-length array, data from HAWC2 to DLL
! - in this case with length 1000
real*8,dimension(100)  :: array2 ! fixed-length array, data from DLL to HAWC2
! - in this case with length 100

! Code is written here

end subroutine initialize
```

9 Wind and Turbulence

9.1 Main command block -wind

Obl.	Command name	Explanation
*	wsp	1. Mean wind speed in center [m/s]
*	density	1. Density of the wind [kg/m ³]
*	tint	Turbulence intensity [-].
*	horizontal_input	This command determines whether the commands above should be understood as defined in the global coordinate system (with horizontal axes) or the meteorological coordinates system (u,v,w) witch can be tilted etc. 1. (0=meteorological – default, 1=horizontal)
*	center_pos0	Global coordinates for the center start point of the turbulence box, meteorological coordinate system etc. (default should the hub center) 1. x_G [m] 2. y_G [m] 3. z_G [m]
*	windfield_rotations	Orientation of the wind field. The rotations of the field are performed as a series of 3 rotations in the order yaw, tilt and roll. When all angles are zero the flow direction is identical to the global y direction. 1. Wind yaw angle [deg], positive when the wind comes from the right, seen from the turbine looking in the $-y_G$ direction. 2. Terrain slope angle [deg], positive when the wind comes from below. 3. Roll of wind field [deg], positive when the wind field is rotated according to the turbulence u-component.
*	shear_format	Definition of the mean wind shear 1. Shear type 0=none. !This option sets the mean wind speed to zero ! $\bar{u}(z) = 0$ 1=constant $\bar{u}(z) = c$, 2=logarithmic $\bar{u}(z) = u_0 \frac{\log \frac{-z_0^G + z^M}{r_0}}{\log \frac{-z_0^G}{r_0}}$ 3=power law $\bar{u}(z) = u_0 \left(\frac{-z_0^G + z^M}{-z_0^G} \right)^\alpha$ 4=linear $\bar{u}(z) = u_0 \frac{\partial u}{\partial z}$ 2. Parameter used together with shear type (case of shear type: 0=dummy, 1=c, 2= r_0 , 3= a, 4= d_u/d_z at center)
*	turb_format	1. Turbulence format (0=none, 1=mann, 2=flex)
*	tower_shadow_method	1. Tower shadow model (0=none, 1=potential flow – default, 2=jet model, 3=potential_2 (flow where shadow source is moved and rotated with tower coordinates system). Please see section, page 68 for sub block commands.

Obl.	Command name	Explanation
	scale_time_start	1. Starting time for turbulence scaling [s]. Stop time is determined by simulation length.
	wind_ramp_factor	Command that can be repeated as many times as needed. The wind_ramp_factor is used to calculate a factor that is multiplied to the wind speed vectors. Can be used to make troublefree cut-in situations. Linear interpolation is performed between t_0 and t_{stop} . 1. time start, t_0 2. time stop, t_{stop} 3. factor at t_0 4. factor at t_{stop}
	wind_ramp_abs	Command that can be repeated as many times as needed. The wind_ramp_abs is used to calculate a wind speed that is added to the wind speed u-component. Can be used to make wind steps etc. Linear interpolation is performed between t_0 and t_{stop} . 1. time start, t_0 2. time stop, t_{stop} 3. wind speed at t_0 4. wind speed at t_{stop}
	user_defined_shear	1. Filename incl. relative path to file containing user defined shear factors (example ./data/shear.dat)
	user_defined_shear_turbulence	1. Filename incl. relative path to file containing user defined shear turbulence factors (example ./data/shearturb.dat)
	met_mast_wind	1. Filename incl. relative path to file containing time series of wind components in meteorological coordinates. The file should have four columns of data: time, v_u , v_v and v_w .
	iec_gust	Gust generator according to IEC 61400-1 1. Gust type 'eog' = extreme operating gust $u(z, t) = u(z, t) - 0.37A \sin\left(\frac{3\pi(t-t_0)}{T}\right) \left(1 - \cos\frac{2\pi(t-t_0)}{T}\right)$ 'edc' = extreme direction change $\theta(t) = 0.5\phi_0 \left(1 - \cos\left(\frac{\pi(t-t_0)}{T}\right)\right)$ 'ecg' = extreme coherent gust $u(z, t) = u(z, t) + 0.5A \left(1 - \cos\left(\frac{\pi(t-t_0)}{T}\right)\right)$ 'ecd' = extreme coherent gust with dir. change $u(z, t) = u(z, t) + 0.5A \left(1 - \cos\left(\frac{\pi(t-t_0)}{T}\right)\right)$ $\theta(t) = 0.5\phi_0 \left(1 - \cos\left(\frac{\pi(t-t_0)}{T}\right)\right)$ 'ews' = extreme wind shear $d = \sqrt{y_M^2 + z_M^2}$

Obl.	Command name	Explanation
		$u(z, t) = u(z, t) + d A \left(1 - \cos \left(\frac{2\pi(t-t_0)}{T} \right) \right) \cos \left(\arctan 2 \left(y^M, -z^M \right) - \phi_0 \right)$ <p>even though the 'ews' expressions do not match the expressions in the standard completely, it gives identical results provided a mutual power law shear is used and the A parameter is set to</p> $A = \frac{2.5 + 0.2\beta\sigma_1 \left(\frac{D}{\Lambda_1} \right)^{\frac{1}{4}}}{D}$ <p>and the parameter ϕ_0 is set to 0, 90, 180, 270 [deg] respectively.</p> <p>2. Amplitude A [m/s]. For the 'eog', 'edc', 'ecd' this corresponds to the parameter 'V_{gust}', '0', 'V_{cg}' respectively, in the IEC61400-1 standard.</p> <p>3. Angle ϕ_0 [deg]</p> <p>4. Time start, t_0 [s]</p> <p>5. Duration T [s]</p>

9.2 Sub command block - mann

Block that must be included if the mann turbulence format is chosen. Normal practice is to use all three turbulence components (u,v,w) but only the specified components are used. In 2008 the turbulence generator was linked to the code so mannturbulence can be created without using external software. The command create_turb_parameters will search for turbulence files with names given below, but if these are not found the turbulence will be created.

A short explanation of the parameters L and $\alpha\varepsilon^{\frac{2}{3}}$ and its relation to the IEC61400-1 ed. 3 standard is given:

The fundamentals of the Mann model is isotropic turbulence in neutral atmospheric conditions. The energy spectrum is given based on the Von Karman spectrum (1). In isotropic turbulence, the properties of turbulence like variance and turbulent length scale is identical for all three direction corresponding to vortex structures being circular.

$$E(k) = \alpha\varepsilon^{\frac{2}{3}} L^{\frac{5}{3}} \frac{(Lk)^4}{\left(1 + (Lk)^2\right)^{\frac{17}{6}}} \quad (1)$$

The relation between wave number k and frequency f is related through the mean wind speed \bar{U} .

$$k = \frac{2\pi f}{\bar{U}} \quad (2)$$

However, atmospheric conditions are not isotropic and the vortex structures become more elliptic in shape with longer length scale and higher variance level in the u direction. In the Mann model, this is accounted for using rapid distortion theory quantified through a shear blocking factor Γ . A Γ parameter of 0 corresponds to isotropic turbulence, whereas a higher Γ value is used for non-isotropic turbulence. The relation between non-isotropic and isotropic properties as function of Γ can be seen in Figure 5. It is normally recommended to use $\Gamma = 3.9$ for normal atmospheric conditions. A length scale of $L = 0.7\Lambda_1$ is recommended for normal conditions. Λ_1 is defined as the wavelength where the longitudinal power spectral density is equal to 0.05. According to the IEC61400-1 the wavelength Λ_1 shall be considered as a constant of 42m (above a height of 60m). In the Mann generation of turbulence a length scale L has to be used. This is the length scale of the Von Karman spectrum (1) and therefore different than the

length scale used in the Kaimal formulation (3). The energy spectrum of Kaimal is formulated

$$E(f) = \sigma^2 \frac{4L/\bar{U}}{(1 + 6fL/\bar{U})^{5/3}} \quad (3)$$

where the input parameters are given based on the table values in

	Velocity component index (k)		
	1	2	3
Standard deviation σ_k	σ_1	$0,8 \sigma_1$	$0,5 \sigma_1$
Integral scale, L_k	$8,1 A_1$	$2,7 A_1$	$0,66 A_1$

Figure 4: Information about Kaimal length scales and standard deviation ratio from the IEC61400-1

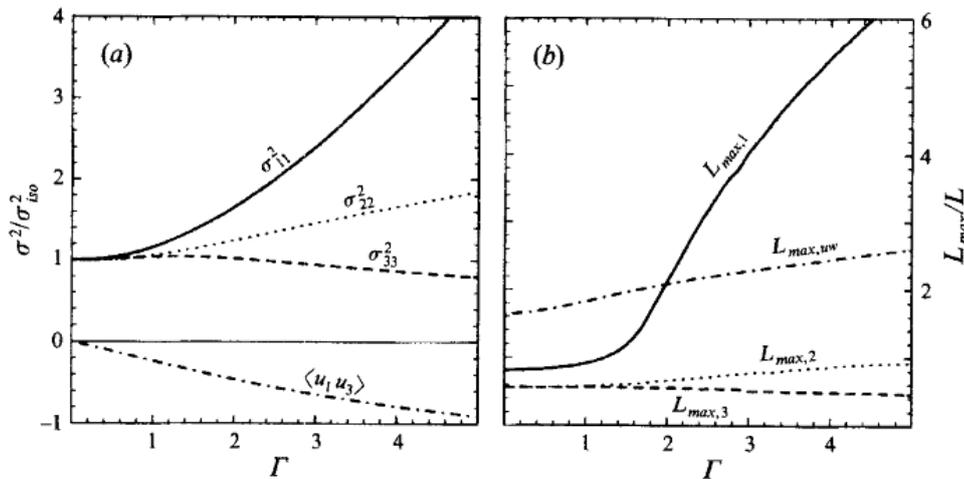


Figure 5: Turbulence characteristics compared to isotropic conditions as function of gamma parameter, Mann.. Left: Relation between variance is changed for higher shear distortions. Right: The relation between length scales are also changed for non-isotropic turbulence. It is recommended to use $\Gamma = 3.9$ for normal atmospheric conditions. This is also the requirement in the IEC61400-1 standard. Isotropic conditions are obtained using $\Gamma=0$.

The result of using $\Gamma = 3.9$ is that the structure of the turbulence corresponds to the normal atmospheric conditions, but the actual level of turbulence is also affected as seen in Figure 4. It is not straight forward to give the exact analytical relationship between the input parameter $\alpha \varepsilon^{2/3}$ and the final longitudinal variance and it is therefore very practical to introduce a turbulence scaling factor SF. This turbulence scaling factor is calculated based on the actual variance level in the box (normally extracted in the center of the box of longitudinal turbulence) and the target variance σ_{target}^2 based on the requested turbulence intensity $\sigma = Ti \bar{U}$. In this case of rescaling, which is the normal usage, the input value for $\alpha \varepsilon^{2/3}$ can be any arbitrary value except for zero.

$$SF = \sqrt{\frac{\sigma_{\text{target}}^2}{\sigma^2}} \quad (4)$$

The scale factor is to be multiplied to every values in the turbulence box for all the u,v and w directions. This is done automatically inside HAWC2.

Obl.	Command name	Explanation
	create_turb_parameters	With this command, the code will search for turbulence files with names given below, but if these are not found the turbulence will be created based on the given parameters. 1. Length scale L (normally L=29.4) 2. $\alpha\varepsilon^{2/3}$ (when rescaling applied, 1.0 is normal practice) 3. γ (3.9 for neutral atmospheric conditions) 4. Seed number (any integer will do) 5. High frequency compensation (1=point velocity only represent local value which is closest to anemometer measurements, recommended in most cases, 0=point velocity represents average velocity in grid volume)
	filename_u	1. Filename incl. relative path to file containing mann turbulence u-component (example ./turb/mann-u.bin)
	filename_v	1. Filename incl. relative path to file containing mann turbulence v-component (example ./turb/mann-v.bin)
	filename_w	1. Filename incl. relative path to file containing mann turbulence w-component (example ./turb/mann-w.bin)
*	box_dim_u	1. Number of grid points in u-direction 2. Length between grid points in u-direction [m]
*	box_dim_v	1. Number of grid points in v-direction 2. Length between grid points in v-direction [m]
*	box_dim_w	1. Number of grid points in w-direction 2. Length between grid points in w-direction [m]
	std_scaling	Ratio between standard deviation for specified component related to turbulence intensity input specified in main wind command block. If the std_scaling command is omitted, the SF is determined based on the u-variance, the SF for v and w direction are kept equal to u-direction (recommended) 1. Ratio to u-direction (default=1.0) 2. Ratio to v-direction (default=0.8) 3. Ratio to w-direction (default=0.5)
	dont_scale	If this command is used the normal scaling to ensure the specified turbulence intensity is bypassed. 1. (0=scaling according to specified inputs – default, 1=raw turbulence field used without any scaling)
	factor_scaling	If this command is used constant, scaling factors are applied. 1. Scaling factor in u-direction, F_u 2. Scaling factor in v-direction, F_v 3. Scaling factor in w-direction, F_w

9.3 Sub command block - flex

Block that must be included if the flex turbulence format is chosen.

Obl.	Command name	Explanation
*	filename_u	1. Filename incl. relative path to file containing flex turbulence u-component (example ./turb/flex-u.int)

Obl.	Command name	Explanation
*	filename_v	1. Filename incl. relative path to file containing flex turbulence v-component (example ./turb/flex-v.int)
*	filename_w	1. Filename incl. relative path to file containing flex turbulence w-component (example ./turb/flex-w.int)
	std_scaling	Ratio between standard deviation for specified component related to turbulence intensity input specified in main wind command block. 1. Ratio to u-direction (default=1.0) 2. Ratio to v-direction (default=0.7) 3. Ratio to w-direction (default=0.5)

9.4 File description of a user defined shear

In this file a user defined shear used instead, or in combination with one of the default shear types (logarithmic, exponential...). When the user defined shear is used the name and location of the datafile must be specified with the *wind – user_defined_shear* command. This command specifies the location of the file and activates the user defined shear. If this shear is replacing the original default shear the command *wind – shear_format* must be set to zero!

Only one shear can be present in a single file. The shear describes the mean wind profile of the u, v and w component of a vertical cross section at the rotor. The wind speeds are normalized with the mean wind speed defined with the command *wind – wsp*.

Line number	Description
1	Headline (not used by HAWC2)
2	Information of shear v-component. #1 is the number of columns, NC #2 is the number of rows, NR
3	Headline (not used by HAWC2)
4..+NR	Wind speed in v-direction, normalized with u-mean. # NC columns
1	Headline (not used by HAWC2)
+1..+NR	Wind speed in u-direction, normalized with u-mean. # NC columns.
1	Headline (not used by HAWC2)
+1..+NR	Wind speed in w-direction, normalized with u-mean. # NC columns
1	Headline (not used by HAWC2)
+1..+NC	Horizontal position of grid points (meteorological coo)
1	Headline (not used by HAWC2)
+1..+NR	Vertical position of grid points (meteorological coo)

9.5 Example of user defined shear file

```
# User defined shear file
3 4 # nr_v, nr_w      array sizes
# shear_v component, normalized with U_mean
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
```

```

0.0 0.0 0.0
# shear_u component, normalized with U_mean
1.0 1.0 1.0
1.0 1.0 1.0
1.0 1.0 1.0
1.0 1.0 1.0
# shear_w component, normalized with U_mean
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
0.0 0.0 0.0
# v coordinates
-50.0
0.0
50.0
# w coordinates (zero is at ground level)
0.0
60.0
100.0
200.0

```

9.6 Sub command block - wakes

Block that must be included if the Dynamic Wake Meandering model is used to model the wind flow from one or more upstream turbines. In order to make the model function, two Mann turbulence boxes must be used. One for the meandering turbulence – which is a box containing atmospheric turbulence, but generated with a coarse resolution in the v,w plane (grid size of 1 rotor diameter). It is important that the turbulence vectors at the individual grid points represent a mean value covering a grid cube. It is also important that the total size of the box is large enough to cover the different wake sources including their meandering path. The resolution in the u-direction should be as fine as possible. The used length scale should correspond to normal turbulence condition. The other turbulence box that is needed is a box representing the micro scale turbulence from the wake of the upstream turbine itself. The resolution of this box should be fine (e.g. 128x128 points) in the v,w plane which should only cover 1 rotor diameter. The resolution in the u direction should also be fine, but a short length of the box (e.g. 2.5Diameter) is OK, since the turbulence box is reused. The length scale for this turbulence is significantly shorter than for the other boxes since it represents turbulence from tip and root vortices mainly. A length scale of 1/16 rotor diameter seems appropriate.

The two turbulence boxes are included by the following sub commands

```

begin mann_meanderturb;
    (parameters are identical to the normal Mann turbulence box, see above)
end mann_meanderturb;

begin mann_microturb;
    (parameters are identical to the normal Mann turbulence box, see above)
end mann_microturb;

```

The rest of the wake commands are given in the following table.

Obl.	Command name	Explanation
*	nsource	1. Number of wake sources. If 0 is used the wake module is by-passed (no source positions can be given in this case).

Obl.	Command name	Explanation
*	source_pos	Command that must be repeated nsource times. This gives the position of the wake source (hub position) in global coordinates. Wake source position given for down stream turbines are however not used in the simulations since they don't affect the target turbine. 1. x-pos [m] 2. y-pos [m] 3. z-pos [m]
*	op_data	Operational conditions for the wake sources. 1. Rotational speed [rad/s] 2. Collective pitch angle [deg]. Defined positive according to the blade root coo, with z-axis from root towards tip.
	ble_parameters	Parameters used for the BLE model used for developing the wake deficit due to turbulent mixing. 1. k_1 [-], default=0.10 2. k_2 [-], default=0.008 3. clean-up parameter (0=intermediate files are kept, 1=intermediate files are deleted), default=1
	microturb_factors	Parameters used for scaling the added wake turbulence according to the deficit depth and depth derivative. 1. k_{m1} [-], factor on deficit depth, default=0.60 2. k_{m2} [-], factor on depth derivative, default=0.25
	multiple_deficit_method	Command that is used for choosing the best approach for handling multiple deficit 1. method (1=MAX operator (default), 2=Direct summation) In general it is recommended to use the MAX operator when the ambient free wind speed is below rated and the direct summation approach above rated wind speed.
	tint_meander	Turbulence intensity of the meander turbulence box. If this command is not used then the default turbulence intensity from the general wind commands is used (normal use) 1. Turbulence intensity [-]
	use_specific_deficit_file	File with the deficits used in the correct downstream distance is used instead of the build in deficit generator. The wind speed deficits are non-dim with the mean wind speed. 1. Filename incl. path (e.g. ./data/deficit.data)
	write_ct_cq_file	File including the local axial and tangential forces (non-dim) as function of blade radius is written. 1. Filename incl. path (e.g. ./res/ct_cq.data)
	write_final_deficits	File with the deficits used in the correct downstream distance is written. The windspeed deficits are non-dim with the mean wind speed. 1. Filename incl. path (e.g. ./res/ct_cq.data)

9.7 File description of a user defined wake deficit file

When another flow solve has been used to find the non-dim turbulence deficit, eg. using an actuator disc approach, this can replace the deficit otherwise calculated internally. This method cannot be used together with multiple deficits as only one deficit can be read.

Line number	Description
1	#1 is a char (not used). Can eg be a '#'

Line number	Description
	#2 is the number of rows, NR
2..+NR	Deficit non-dim with ambient free mean wind speed. #1 Radius (non-dim with rotor radius) #2 Deficit (non-dim with free mean wind speed). In the free

9.8 Example of user defined wake deficit file

```
# 121
0.000000000E+00 8.276891200E-01
2.500000000E-02 8.486243600E-01
5.000000000E-02 8.809613720E-01
7.500000000E-02 9.007844070E-01
1.000000000E-01 8.957724550E-01
1.250000000E-01 8.660702830E-01
1.500000000E-01 8.303410890E-01
1.750000000E-01 8.044380440E-01
2.000000000E-01 7.895593800E-01
2.250000000E-01 7.786515560E-01
2.500000000E-01 7.691674220E-01
2.750000000E-01 7.618372330E-01
3.000000000E-01 7.572012850E-01
3.250000000E-01 7.550918200E-01
3.500000000E-01 7.542137030E-01
3.750000000E-01 7.518827010E-01
4.000000000E-01 7.456746090E-01
4.250000000E-01 7.357259740E-01
4.500000000E-01 7.250309980E-01
4.750000000E-01 7.168460970E-01
5.000000000E-01 7.119492260E-01
5.250000000E-01 7.088296670E-01
5.500000000E-01 7.057605130E-01
5.750000000E-01 7.021459650E-01
6.000000000E-01 6.983228280E-01
6.250000000E-01 6.947171830E-01
6.500000000E-01 6.913423360E-01
6.750000000E-01 6.879199230E-01
7.000000000E-01 6.842943230E-01
7.250000000E-01 6.806519720E-01
7.500000000E-01 6.773263690E-01
7.750000000E-01 6.744196220E-01
8.000000000E-01 6.716445590E-01
8.250000000E-01 6.684818930E-01
8.500000000E-01 6.644046880E-01
8.750000000E-01 6.592242170E-01
9.000000000E-01 6.529686490E-01
9.250000000E-01 6.445576730E-01
9.500000000E-01 6.324201240E-01
9.750000000E-01 6.173566910E-01
1.000000000E+00 5.982423590E-01
1.028634580E+00 5.679249380E-01
1.058116050E+00 5.982195030E-01
1.088469450E+00 7.292761710E-01
1.119720570E+00 9.095984580E-01
```

1.151895960E+00 1.014958390E+00
 1.185022960E+00 1.022114240E+00
 1.219129700E+00 1.017341600E+00
 ...
 8.903031630E+00 1.000285950E+00
 9.165402860E+00 1.000213540E+00
 9.435533870E+00 1.000143160E+00
 9.713654150E+00 1.000066170E+00
 1.000000000E+01 1.000018010E+00

9.9 Sub command block – tower_shadow_potential

Block that must be included if the potential flow tower shadow model is chosen.

Obl.	Command name	Explanation
*	tower_offset	The tower shadow has its source at the global coordinate z axis. The offset is the base point for section 1 1. Offset value (default=0.0)
*	nsec	Command that needs to present before the radius commands. 1. Number of datasets specified by the radius command.
*	radius	Command that needs to be listed nsec times. 1. z coordinate [m] 2. Tower radius at z coordinate [m]

9.10 Sub command block – tower_shadow_jet

Block that must be included if the model based on the boundary layer equations for a jet is chosen. This model is especially suited for downwind simulations.

Obl.	Command name	Explanation
*	tower_offset	The tower shadow has its source at the global coordinate z axis. The offset is the base point for section 1 1. Offset value (default=0.0)
*	nsec	Command that needs to present before the radius commands. 1. Number of datasets specified by the radius command.
*	radius	Command that needs to be listed nsec times. 1. z coordinate [m] 2. Tower radius at z coordinate [m] 3. Cd drag coefficient of tower section (normally 1.0 for circular section, but this depends heavily on the reynold number)

9.11 Sub command block – tower_shadow_potential_2

Block that must be included if the tower shadow method 3 is chosen. This potential model is principally similar to the potential flow model described previously but differs in the way that the shadow source is moved and rotated in space as the tower coordinate system is moving and rotating. It is also possible to define several tower sources e.g. if the tower is a kind of tripod or quattropod. Just include more tower_shadow_potential_2 blocks if more sources are required.

The coordinate system that the shadow method is linked to is specified by the user, e.g. the mbdy coordinate from the tower main body. To make sure that the tower source model is always linked in the same way as the tower (could be tricky since the tower is fully free to be specified

along the x,y or z axis or a combination) the base coordinate system for the shadow model is identical to the coordinates system obtained by the local element coordinates, where the z axis is always pointing from node 1 towards node 2. This is the reason that the tower radius input has to be specified with positive z-values, see below.

Obl.	Command name	Explanation
*	tower_mbody_link	Name of the main body to which the shadow source is linked. 1. mbody name
*	nsec	Command that needs to present before the radius commands. 1. Number of datasets specified by the radius command.
*	radius	Command that needs to be listed nsec times. 1. z coordinate [m] (always positive!) 2. Tower radius at z coordinate [m]

9.12 Sub command block – tower_shadow_jet_2

Block that must be included if the tower shadow method 4 is chosen. This jet model is principally similar to the jet model described previously but differs in the way that the shadow source is moved and rotated in space as the tower coordinate system is moving and rotating. It is also possible to define several tower sources e.g. if the tower is a kind of tripod or quattropod. Just include more tower_shadow_jet_2 blocks if more sources are required.

The coordinate system that the shadow method is linked to is specified by the user, e.g. the mbody coordinate from the tower main body. To make sure that the tower source model is always linked in the same way as the tower (could be tricky since the tower is fully free to be specified along the x,y or z axis or a combination) the base coordinate system for the shadow model is identical to the coordinates system obtained by the local element coordinates, where the z axis is always pointing from node 1 towards node 2. This is the reason that the tower radius input has to be specified with positive z-values, see below.

Obl.	Command name	Explanation
*	tower_mbody_link	Name of the main body to which the shadow source is linked. 1. mbody name
*	nsec	Command that needs to present before the radius commands. 1. Number of datasets specified by the radius command.
*	radius	Command that needs to be listed nsec times. 1. z coordinate [m] 2. Tower radius at z coordinate [m] 3. C_d drag coefficient of tower section (normally 1.0 for circular section, but this depends heavily on the reynold number)

9.13 Sub command block – turb_export

With this sub command block, a mann format turbulence box including information from shear, wakes, tower shadow etc. is written. Same data point positions are used as specified in the turbulence module including the parameters specified for the originally used mann turbulence box.

Obl.	Command name	Explanation
*	filename_u	Filename of turbulence box with axial turbulence 1. File name
*	filename_v	Filename of turbulence box with lateral turbulence 1. File name

Obl.	Command name	Explanation
*	filename_w	Filename of turbulence box with vertical turbulence 1. File name
	samplefrq	1. Sample frequency
	time_start	1. Time at which the the turbulence recording will start
	nsteps	1. Number of steps between output
	box_dim_v	1. Number of points in v-direction 2. Distance between points in v-direction
	box_dim_w	1. Number of points in w-direction 2. Distance between points in w-direction

10 Aerodynamics

10.1 Main command block - aero

This module set up parameters for the aerodynamic specification of the rotor. It is also possible to submit aerodynamic forces to other structures as example the tower or nacelle, but see chapter (Aerodrag) regarding this. The module can be added as many times as requested if multiple aerodynamic rotors are needed.

Obl.	Command name	Explanation
(*)	name	Name of rotor (in case of multiple rotors defined this is obligatory.)
*	nblades	Must be the first line in aero commands! 1. Number of blades
*	hub_vec	Link to main-body vector that points downwind from the rotor under normal conditions. This corresponds to the direction from the pressure side of the rotor towards the suction side where the coordinate system is normally taken from the main shaft system.. 1. mbdy name or 'old_input' if old_htc_structure format is applied. 2. mbdy coo. component (1=x, 2=y, 3=z). If negative the opposite direction used. Not used together with old_htc_structure input (specify a dummy number). 3. Node number (optional). Node number on mbdy where rotor center is located. 'last' can also be used (default if no value is present).
*	link	Linker between structural blades and aerodynamic blades. There must be same number of link commands as nblades! 1. blade number 2. link chooser – options are - mbdy_c2_def (used with new structure format) - blade_c2_def (used with old structure format, see description below in this chapter) 3. mbdy name (with new structure format), not used to anything with old structure format.
*	ae_filename	1. Filename incl. relative path to file containing aerodynamic layout data (example ./data/hawc2_ae.dat)
*	pc_filename	1. Filename incl. relative path to file containing profile coefficients (example ./data/hawc2_pc.dat)
*	induction_method	1. Choice between which induction method that shall be used (0=none, 1=normal BEM dynamic induction, 2= Near Wake induction method, 3= VAWT)
*	aerocalc_method	1. Choice between which aerodynamic load calculation method that shall be used. (0=none, 1=normal)
	aerosections	Number of aerodynamic calculation points at a blade. The distribution is performed automatically using a cosine transformation which gives closest spacing at root and tip. 1. Number of points at each blade.
	aero_distribution	1. Distribution method of aerodynamic calculation points. Options are: - “default” number. The distribution is performed automatically using npoints position with a cosine transformation which gives closest spacing at root and tip.

Obl.	Command name	Explanation
		- "ae_file" set. The distribution is given with same spacing as values in the ae_file with set number set..
*	ae_sets	Set number from ae_filename that is linked to blade 1,2,...,nblades 1. set for blade number 1 2. set for blade number 2 . . . nblades. set for blade number nblades
*	tiploss_method	1. Choice between which tip-loss model that shall be used (0=none, 1=prandtl (default))
*	dynstall_method	1. Choice between which dynamic stall model that shall be used (0=none, 1=Stig Øye method, 2=MHH Beddoes method, 3=Gaunaa-Andersen method with Deformable Trailing Edge Flap's)
	3d_correct_method	Airfoil Cl values from the pc_file is modified for 3D effects. 1. Correction method (1=Snel method for correction of Cl values)
	external_bladedata_dll	Blade structural data are found in an external encrypted dll. If this command is present the following command lines shall not be present (ae_filename, pc_filename and ae_sets). 1. Company name (that has been granted a password, eg. dtu). 2. Password for opening this specific dll, eg. test1234 3. path and filename for the dll. eg. ./data/encr_blade_data.dll
	output_profile_coef_filename	Interpolated profile coefficients at all aerodynamic calculation points are written into a data file. This command can not be used in combination with encrypted_profile_coef_filename. 1. path and filename for the dll. eg. ./res/aero_profiles.dat

10.2 Sub command block – dynstall_so

Block that may be included if the Stig Øye dynamic stall method is chosen. If not included defaults parameters are automatically used.

Obl.	Command name	Explanation
	dclda	1. Linear slope coefficient for unseparated flow (default=6.28)
	dcldas	1. Linear slope coefficient for fully separated flow (default=3.14)
	alfs	1. Angle of attack [deg] where profile flow is fully separated. (default=40)
	alrund	1. Factor used to generate synthetic separated flow Cl values (default=40)
	taufak	1. Time constant factor in first order filter for F function (default=10.0). Internally used as tau=taufak*chord*vrel

10.3 Sub command block – dynstall_mhh

Block that may be included if the MHH Beddoes dynamic stall method is chosen (see Risø report 1354(en)). If not included defaults parameters are automatically used.

Obl.	Command name	Explanation
	a1	1. Coefficients of the exponential potential flow step response approximation: $\Phi(s)=1-A1*\exp(-b1*s)-A2*\exp(-b2*s)$. (default=0.165)

Obl.	Command name	Explanation
	a2	1. Coefficients of the exponential potential flow step response approximation: $\Phi(s)=1-A1*\exp(-b1*s)-A2*\exp(-b2*s)$. (default=0.335)
	b1	1. Coefficients of the exponential potential flow step response approximation: $\Phi(s)=1-A1*\exp(-b1*s)-A2*\exp(-b2*s)$. (default=0.0455)
	b2	1. Coefficients of the exponential potential flow step response approximation: $\Phi(s)=1-A1*\exp(-b1*s)-A2*\exp(-b2*s)$. (default=b2=0.300)
	update	Choice between update methods: 1. 1 (default)=>update aerodynamics all iterations all timesteps; 0=>only update aerodynamics first iteration each new timestep
	taupre	1. Non-dimensional time-lag parameters modeling pressure time-lag. Default value =1.5
	taubly	1. Non-dimensional time-lag parameters modeling boundary layer time-lag. Default value=6.0
	only_potential_model	1. 0(default)=>run full MHH beddoes model; 1=>Potential flow model dynamics superposed to steady force coefficients;
	max_cl_attached	1. Maximum value of lift coefficient for attached flow.

10.4 Sub command block – dynstall_ateflap

This sub-block should be included if the ATEFlap dynamic stall model is chosen (dynstall_method number 3). The dynamic stall model is similar to the MHH model, expanded to account for steady and dynamic effects of trailing edge flap deflections; the model is described in L. Bergami and M. Gaunaa, *ATEFlap Aerodynamic model, a dynamic stall model including the effects of trailing edge flap deflection* (Risoe-R-1792(EN), Risoe DTU, February 2012). The model requires a .ds input file containing pre-processed steady aerodynamic data for the blade sections containing a flap (see following paragraphs for the file specifications). Sections without any flap are attributed steady input data according to the aerodynamic layout specified in the ae_filename.

Obl.	Command name	Explanation
*	flap	Mandatory command to define a flap section. The flap is defined on all the blades of the rotor. Command syntax: 1. Radius r_start [in m]. Starting point of flap section. 2. Radius r_end [in m]. Ending point of flap section (should be > r_start). 3. Filename incl. relative path to .ds file containing pre-processed aerodynamic steady input data. See .ds file specifications in the following paragraph. N.B. The location along the blade refer to the 'stretched' blade, distances are given along the half-chord line (as the layout in ae_file). A maximum of 99 flap sections can be defined.
	ais	Coefficients for the indicial response exponential function: 1. A1 (default= 0.1784) 2. A2 (default=0.07549) 3. A3 (default=0.3933) Default coefficients describe the step response of a NACA 64-418 profile, where t/c=0.18.
	bis	Coefficients of the exponential potential flow step response approximation: 1. B1 (default= 0.8000)

Obl.	Command name	Explanation
		2. B2 (default= 0.01815) 3. B3 (default= 0.1390) Default coefficients describe the step response of a NACA 64-418 profile, where $t/c=0.18$.
	taupre	1. Non-dimensional time-lag parameters modelling pressure time-lag. Default value =1.5
	taubly	1. Non-dimensional time-lag parameters modelling boundary layer time-lag. Default value=6.0
	only_potential_model	1. 0(default)=>run full ATEFlap model; 1=>Potential flow model dynamics superposed to steady force coefficients;
	update	Choice between update methods: 1. 1 (default)=>update aerodynamics all iterations all timesteps; 0=>only update aerodynamics first iteration each new timestep
	hystar	1. Camberline coef. (default= -4.675844E-003)
	fylestar	1. Camberline coef. (default= +4.155446E-004)
	fdydxle	1. Camberline coef. (default= +7.236104E-003)
	gdydxle	1. Camberline coef. (default= +3.309147E-003)

The camber line coefficients describe the camber line deformation shape induced by the flap; they are computed according to the thin-airfoil model described in Gaunaa's Wind Energy journal article *Unsteady two-dimensional potential-flow model for thin variable geometry airfoils*. Hystar and fylestar are dimensionless parameters corresponding to the shape integrals Hy and FyLE normalized by the half-chord length. The default coefficients refer to a 10% chord length flap with a continuous deformation shape, describing a circular arc, whose chord forms an angle of 1 degree with the horizontal axis.

10.5 Sub command block – aero_noise

If this command block is used, aero-acoustic calculations are performed. The blade is discretized spanwise into elementary blade sections corresponding to the aerodynamic calculation points of the main command block – aero, i.e. as defined by the command 'aerosections'. Aerodynamic noise is calculated for each of these blade sections and subsequently added at the observer location(s) assuming incoherent noise sources. Only geometrical spreading is considered for the noise propagation between blade sections and observer. Details of the implementation for the turbulent inflow, trailing edge and stall noise models can be found in Bertagnolio et al, *A combined aeroelastic-aeroacoustic model for wind turbine noise: verification and analysis of field measurements*, Wind Energy (20), 2017. As for the loading-thickness noise model, the implementation is described in Bertagnolio et al, *A temporal wind turbine model for low-frequency noise*, InterNoise (Conf. Proc.), 2017.

Obl.	Command name	Explanation
	noise_mode	1. Noise mode (0=no noise calculation, 1=compute noise at each time-step on the fly, 2=store aerodynamic data for later noise calculation as post-processing (using option 3 or 4), 3=compute noise at each time-step using stored data, 4=compute steady-state noise using stored data and rotor disk azimuthal sector averaging yielding large time-saving) (default=0)
	noise_start_end_time	Start and end time for noise computation. 1. Start time, t_0 [s] 2. End time, t_1 [s] (default: at all time)

Obl.	Command name	Explanation
	noise_deltat	1. Time-step for noise calculation (default: at each HAWC2 time-step)
	noise_azimuth_sectors	1. Number of rotor disk azimuthal sectors when running noise_mode=4 (default=16)
	atmospheric_pressure	1. Atmospheric pressure [Pa] (default=101325.)
	temperature	1. Temperature [deg. Celsius] (default=20.)
	octave_bandwidth	1. Octave band frequency centers used for defining noise spectra. Options are: 1, 3, 12 and 24 (default=3)
	spl_min_max_frq	Minimum and maximum computed frequency for integrated sound pressure level calculations. 1. Minimum frequency, $f_{r_{min}}$ [Hz] 2. Maximum frequency, $f_{r_{max}}$ [Hz] (default: all octave band frequency centers are used)
	turbulent_inflow_noise	1. Turbulent inflow noise model (0=using Von Karman turbulence spectra, 1=using Mann atmospheric turbulence model) (default=0)
	turbulent_inflow-_thickness_correction	1. Turbulent inflow thickness correction (0=none, 1=correction is added to turbulent inflow noise) (default=0)
	mann_turbulence_parameters	Mann turbulence parameters. 1. L: turbulent integral length (default=29.7m) 2. $\alpha\varepsilon^{2/3}$: energy level (default=1.0) 3. γ : anisotropy factor (default=3.7) If any value is negative, then its default value is assumed.
	surface_roughness	1. Surface roughness, z_0 (If specified, it is used to re-define the Mann turbulence parameters)
	trailing_edge_noise	1. Trailing edge model (0=none, 5=TNO 'frba' model, 31=Amiet 'frba' model, 41=Amiet 'asfi' model) (default=0)
*	bldata_filename	1. Filename incl. relative path defining tabulated input data for trailing edge noise model.
	trailing_edge_serration	Trailing edge serration model parameters. 1. R_1 Inboard radius [m] 2. R_2 Outboard radius [m] 3. L_{ser} Serration periodic span length [m] 3. H_{ser} Serration crest to trough height [m]
	stall_noise	1. Stall noise model (0=none, 1=Amiet based model, 2=Full formulation) (default=0)
	stall_separation	Stall separation definition. 1. Stall separation (1=tabulated and given in bldata_filename, 2=use dynamic stall model, 3=forced separation location) (default=1) 2. Forced separation location ($x/C[-]$): if positive on suction side, if negative on pressure side)
	tip_noise	1. Tip noise model (0=none, 1=not implemented yet!!!) (default=0)
	loading_noise	1. Loading-thickness noise model (0=none, 1=based on tabulated Cl, 2=based on Cp distribution from tabulated data, 3=based on Cl from HAWC2 aerodynamics) (default=0) This model does not work with noise_mode=4.
	loading_data_filename	1. Filename incl. relative path defining tabulated input data for loading-thickness noise.
*	xyz_observer	Position of observer in global reference system. 1. x [m] 2. y [m]

Obl.	Command name	Explanation
		3. z [m] More than one observer is allowed (but must be <256).
	output_filename	1. Filename incl. relative path for output log file.

10.6 Sub command block – bemwake_method

Dynamic inflow settings used to calculate the dynamic induction. If not included defaults parameters are automatically used.

Obl.	Command name	Explanation
	nazi	1. Number of azimuthal points in the induction grid. A high number increased accuracy but slow down the simulation time. Default is 16.
	fw	Dynamic time constants and mixing ratio contribution for the far wake part of the induction. 1. Mixing ratio, default is 0.4 2. k_3 (poly. coef. for r/R sensitivity) default=0.0 3. k_2 (poly. coef. for r/R sensitivity) default=-0.4751 4. k_1 (poly. coef. for r/R sensitivity) default=0.4101 5. k_0 (poly. coef. for r/R sensitivity) default=1.921
	nw	Dynamic time constants and mixing ratio contribution for the near wake part of the induction. 1. Mixing ratio, default is 0.6 2. k_3 (poly. coef. for r/R sensitivity) default=0.0 3. k_2 (poly. coef. for r/R sensitivity) default=-0.4783 4. k_1 (poly. coef. for r/R sensitivity) default=0.1025 5. k_0 (poly. coef. for r/R sensitivity) default=0.6125
	a-ct-filename	Filename for a user defined relation between a and ct.

10.7 Sub command block – nearwake_method

The near wake model implementation in HAWC2 couples the lifting line theory based near wake model for trailed vorticity with the modified HAWC2 BEM as a far wake model. Inherently included in the trailed vorticity computations are the influences of the tip and root vortices; a ‘root-loss’ model is otherwise not included in HAWC2. The model is described in [1,2] and has been shown to improve the dynamic blade loading in the presence of turbulence, blade vibrations and flap actuations.

In case of strong load gradients on the blade due to for example flaps at fixed angle or other aerodynamic devices activating the near wake model leads to an improved steady state load distribution. When used in this case with a prescribed point distribution along the blade (defined in the ae-file) then sudden changes in the point density (for example close to the flap) should be avoided as they can lead to numerical instability of the model. As with any vortex model, care should be taken when operating in deep stall conditions, such as extreme yaw conditions in standstill.

[1] Pirrung, G., Riziotis, V., Madsen, H., Hansen, M., and Kim, T.: Comparison of a coupled near- and far-wake model with a free-wake vortex code, *Wind Energ. Sci.*, 2, 15-33, <https://doi.org/10.5194/wes-2-15-2017>, 2017.

[2] Pirrung, G. R., Madsen, H. A., Kim, T., and Heinz, J.: A coupled near and far wake model for wind turbine aerodynamics, *Wind Energy*, doi:10.1002/we.1969, 2016.

Obl.	Command name	Explanation
	only_one_nw_function	Dynamic accuracy, see Section 6 in [1] for details. (0=2 exponential functions used; 1=default, 1 exponential function used: minimally lower accuracy but almost twice as fast)
	only_axial_nw	(0=default, near wake model used for both axial and tangential induction; 1= near wake model used for axial induction only)
	fast_nwm	(0=full iteration loop of the near wake model; 1= default, helix angle and vortex filament length fixed during iteration loop, almost identical results, much faster)
	fixed_kfw	kfw (0<kfw<1). This coupling factor will be used and is fixed during the computation. Not using this command means the coupling factor will be computed automatically and dynamically updated each time step (default, see Section 5 in [2] for details)
	r_core	r_core determines the vortex core radius (default=0: no vortex core is used). The implementation is in beta version and not validated.

10.8 Sub command block – vawtwake_method

VAWT wake method parameters.

Obl.	Command name	Explanation
	nazi	1. Number of azimuthal points in the induction grid. A high number increased accuracy but slow down the simulation time.
	fw	Dynamic time constants and mixing ratio contribution for the far wake part of the induction. 1. Mixing ratio, default is 0.4 2. k_3 (poly. coef. for r/R sensitivity) default=0.0 3. k_2 (poly. coef. for r/R sensitivity) default=-0.4751 4. k_1 (poly. coef. for r/R sensitivity) default=0.4101 5. k_0 (poly. coef. for r/R sensitivity) default=1.921
	nw	Dynamic time constants and mixing ratio contribution for the near wake part of the induction. 1. Mixing ratio, default is 0.6 2. k_3 (poly. coef. for r/R sensitivity) default=0.0 3. k_2 (poly. coef. for r/R sensitivity) default=-0.4783 4. k_1 (poly. coef. for r/R sensitivity) default=0.1025 5. k_0 (poly. coef. for r/R sensitivity) default=0.6125

10.9 Data format for the aerodynamic layout

The format of this file which in the old HAWC code was known as the hawc_ae file is changed slightly for the HAWC2 input format. The position of the aerodynamic center is no longer an input value, since the definition is that the center is located in $C_{1/4}$ with calculated velocities in $C_{3/4}$.

Position of aerodynamic centers related to c2_def section coo.

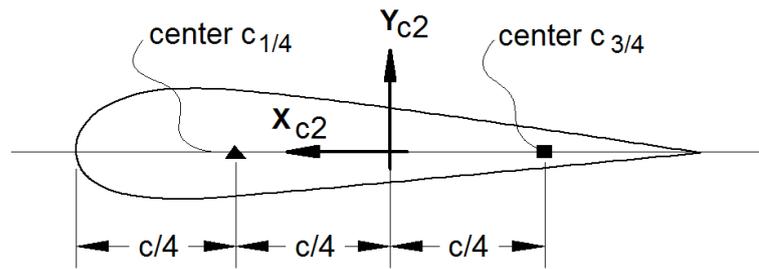


Figure 6: Illustration of aerodynamic centers $C_{1/4}$ and $C_{3/4}$

The format of the file is specified in the following two tables

Line number	Description
1	#1: Nset, Number of datasets present in the file. The format of each data set can be read below. The datasets are repeated without blank lines etc.
2	#1: Set number. #2: Nrows, Number of data rows for this set
3..2+Nrows	Data row according to Table 5

Table 22: Format of main data structure for the aerodynamic “_ae” blade layout file

The content of the columns in a data row is specified in the table below.

Column	Parameter
1	r, distance from main_body node 1 along z-coordinate [m]
2	chord length [m]
3	thickness ratio between profile height and chord [%]
4	Profile coefficient set number
(5)	Optional column. When present, it includes a dynamic stall model selector. It is then possible to bypass or change dynamic stall model for different part of the blade. Numbers are identical to the one used in the command “aero dynstall_method”

Table 23: Format of the data rows for the aerodynamic “_ae” blade layout file

10.10 Example of an aerodynamic blade layout file

```

1 Number of datasets in the file.
1 25 Set nr, nrows.
0 2.42 100 1 Radius [m] chord[m] thick[%] PC [-]
1.239 2.42 100 1
1.24 2.42 99.9 1
3.12 2.48 96.4 1
5.24 2.65 80.5 1
7.24 2.81 65.0 1
9.24 2.98 51.6 1
11.24 3.14 40.3 1
13.24 3.17 32.5 1
15.24 2.99 28.4 1

```

```

17.24 2.79 25.6 1
19.24 2.58 23.7 1
20.44 2.46 22.8 1
23.24 2.21 20.9 1
25.24 2.06 20.0 1
27.24 1.92 19.4 1
29.24 1.8 19.0 1
31.24 1.68 18.7 1
33.24 1.55 18.6 1
35.24 1.41 18.3 1
37.24 1.18 17.9 1
38.24 0.98 17.3 1
39.24 0.62 16.3 1
39.64 0.48 15.7 1
40.00 0.07 14.8 1

```

10.11 Data format for the profile coefficients file

The format of this file which in the old HAWC code was known as the hawc_pc file has not been changed for the HAWC2 code.

The format of the file is specified in the following two tables

Line number	Description
1	#1: Nset, Number of datasets present in the file. The format of each data set can be read below. The datasets are repeated without blank lines etc.
2	#1: Nprofiles. Number of profiles included in the data set. There must be more than 1 Nprofiles. First profile is the thinnest, last profile is the thickest (continuously increasing order).
3	#1: Set number. #2: Nrows. #3: Thickness in percent of chord length
4..3+Nrows	Data row according to Table 7

Table 24: Format of main data structure for the profile coefficients file

The content of the columns in a data row is specified in table below.

Column	Parameter
1	α , angle of attack [deg]. Starting with -180.0, ending with +180.0
2	C_l lift coefficient [-]
3	C_d drag coefficient [-]
4	C_m moment coefficient [-]

Table 25: Format of the data rows for the profile coefficients file

10.12 Example of the profile coefficients file “_pc file”

```

1 Airfoil data for the nrel 5 mw turbine
8
1 127 17 DU17 airfoil with an aspect ratio of 17. Original -180 to 180deg
-180.00 0.000 0.0198 0.0000

```

-175.00	0.374	0.0341	0.1880
-170.00	0.749	0.0955	0.3770
-160.00	0.659	0.2807	0.2747
-155.00	0.736	0.3919	0.3130
-150.00	0.783	0.5086	0.3428
-145.00	0.803	0.6267	0.3654
-140.00	0.798	0.7427	0.3820
-135.00	0.771	0.8537	0.3935
-130.00	0.724	0.9574	0.4007
-125.00	0.660	1.0519	0.4042
-120.00	0.581	1.1355	0.4047
-115.00	0.491	1.2070	0.4025
-110.00	0.390	1.2656	0.3981
-105.00	0.282	1.3104	0.3918
-100.00	0.169	1.3410	0.3838
-95.00	0.052	1.3572	0.3743
-90.00	-0.067	1.3587	0.3636
-85.00	-0.184	1.3456	0.3517
-80.00	-0.299	1.3181	0.3388
-75.00	-0.409	1.2765	0.3248
-70.00	-0.512	1.2212	0.3099
-65.00	-0.606	1.1532	0.2940
-60.00	-0.689	1.0731	0.2772
-55.00	-0.759	0.9822	0.2595

10.13 Data format for the flap steady aerodynamic input (.ds file)

This file contains the pre-processed steady data required by the ATEFlap dynamic stall model. Steady lift, drag and moment coefficients are given as function of angle of attack and flap deflection, together with the fully separated and fully attached lift, and the separation function values required by the Beddoes-Leishmann dynamic stall model. The input file can be generated automatically through an external pre-processing application, as for instance the “Preprocessor for ATEFlap Dynamic Stall Model, v.2.04”. Please refer to the application documentation for further details.

The format of the file is specified in the following two tables:

Line number	Description
1	Free for comments
2	Free for comments
3	#1: Aoa0 [rad]. Angle of attack returning a null steady lift
4	Free for comments
5	#1: dCl/dAoa [1/rad]. Gradient of the steady lift function with respect to angle of attack variations
6	Free for comments
7	#1: dCl/dBeta [-]. Gradient of the steady lift function with respect to flap deflection variations
8	Free for comments
9	#1: Nrows. Total number of the following data-rows.
10...9+Nrows	Data rows, as specified in following table.

Table 26: Format of main data structure for the .ds flap steady aerodynamic input file

The content of the columns in a data row is specified in table below.

Column	Parameter
1	α , Angle Of Attack [deg]. Starting with -180.0, ending with +180.0. External loop (changes value after going through all the beta flap deflection values, i.e. 100 rows)
2	Beta, flap deflection. Starting from -49 to +50. Internal loop (changes at every data row)
3	C_l st. Steady lift coefficient [-]
4	C_l att. Fully attached lift coefficient [-]
5	C_l fs. Fully separated lift coefficient [-]
6	C_d drag coefficient [-]
7	C_m moment coefficient [-]
8	f . Steady value of the separation function [-]

Table 27: Format of the data rows for the .ds flap steady aerodynamic input file

10.14 Example of a .ds flap steady aerodynamic input file

Input file for Flap dyn.stall model. Generated with Delphi preprocessor

.Linear Region: Aoa Cl0 [rad]:

-0.06523855

.Linear Region: dCl / dAoa [1/rad]:

6.60081861

.Linear Region: dCl / dBeta [1/deg]:

0.0435375

. Polars: 1.Aoa | 2.Beta | 3.Clst | 4.Cl Att | 5.Cl fs | 6.Cd | 7.Cm | 8.F
36100

```

-180 -49 -0.22013 -20.5241432 -0.22013 0.0199118108 0.0451649986 0
-180 -48 -0.22013 -20.5241432 -0.22013 0.0199118108 0.0451649986 0
... ...
-180 +50 0.21096 -20.088768 0.21096 0.0199443996 -0.0431930013 0
-179 -49 ...
-179 -48 ...
... ...
+180 +50 ...

```

10.15 Data format for the user defined a-ct relation

The format of the file is specified in the following two tables

Line number	Description
1. nrad interp	Nrad interpolation. Interpolation method can either be "linear" or "akima"
2. nazi	Data row according to Table 11

Table 28: Format of main data structure for the profile coefficients file

The content of the columns in a data row is specified in table below.

Column	Parameter
1	non-dim radius r/R

Column	Parameter
2	k_1 polynomial coef
3	k_2 polynomial coef
4	k_3 polynomial coef
5	k_4 polynomial coef

Table 29: Format of the data rows for the profile coefficients file

10.16 Main command block – blade_c2_def (for use with old_htc_structure format)

In this command block the definition of the centerline of the main_body is described (position of the half chord). This command shall be used as a main command even though it is only used together with the aerodynamic module. The reason for this is that it used to submit information that is usually given in the new_htc_structure format, which is also a main command block. The input data given with the sec commands below is used to define a continuous differentiable line in space using akima spline functions. This centerline is used as basis for local coordinate system definitions for sections along the structure. If a straight line is requested a minimum of three points of this line must be present.

Obl.	Command name	Explanation
*	nsec	Must be the present before a “sec” command. 1. Number of section commands given below
*	sec	Command that must be repeated “nsec” times 1. Number 2. x-pos [m] 3. y-pos [m] 4. z-pos [m] 5. θ_z [deg]. Angle between local x-axis and main_body x-axis in the main_body x-y coordinate plane. For a straight blade this angle is the aerodynamic twist. Note that the sign is positive around the z-axis, which is opposite to traditional notation for etc. a pitch angle.

11 Aerodrag (for tower and nacelle drag)

11.1 Main command aerodrag

With this module it is possible to apply aerodynamic drag forces at a given number of structures.

11.2 Subcommand aerodrag_element

Command block that can be repeated as many times as needed. In this command block aerodynamic drag calculation points are set up for a given main body.

Obl.	Command name	Explanation
*	body_name or mbody_name	1. Main_body name to which the hydrodynamic calculation points are linked.
*	aerodrag_sections	1. Distribution method: ("uniform" only possibility) 2. Number of calculation points (min. 2).
	nsec	This command must be present before the sec commands 1. Number of sections given below
	sec	This command must be repeated nsec times 1. Distance in [m] along the main_body c2_def line. Positive directed from node 1 to node "last". 2. C_d drag coefficient (default=1.0) 3. Width of structure (diameter)
	update_states	Logical parameter that determines whethe the movement of the structure is included or not. 1. parameter (1=states are updated (default), 0=not updated)

*) Input commands that must be present

12 Hydrodynamics

12.1 Main command block - hydro

In this command block hydrodynamic forces calculated using Morison's formula is set up.

12.2 Sub command block – water_properties

Obl.	Command name	Explanation
*	gravity	1. Gravity acceleration (used for calculation of buoyancy forces). Default = 9.81 m/s ²
*	mudlevel	1. Mud level [m] in global z coordinates.
*	mwl	1. Mean water level [m] in global z coordinates.
*	rho	1. Density of the water [kg/m ³]. Default=1027
	wave_direction	1. Wave direction [deg]. Direction is positive when the waves come forward from the right when looking towards the wind at default conditions.
	current	1. Current type (0=none (default), 1=constant, 2=power law $U(z) = U_0((z + mudlevel - mwl)/(mudlevel - mwl))^\alpha$ 2. Current velocity at mwl, u_0 3. type parameter. If type=2 then parameter is alfa 4. Current direction relative to wave direction [deg]. Positive direction if current comes from the right looking towards the incoming waves.
	water_kinematics_dll	1. Filename incl. relative path to file containing water kinematics dll (example ./hydro/water_kin.dll) 2. String sent to initialization of dll. This is typical the name of a local inputfile of the dll.

12.3 Sub command block – hydro_element

Command block that can be repeated as many times as needed. This command block set up hydrodynamic calculation points and link them to a main_body.

Obl.	Command name	Explanation
*	body_name or mbody_name	1. Main_body name to which the hydrodynamic calculation points are linked.
*	hydrosections	1. Distribution method of hydrodynamic calculation points. Options are: “uniform” nnodes. Where uniform ensures equal distance of the calculation points. nnodes are number of calculation points. “auto” nint. Here calculations points are chosen as the postions of the structural nodes and the hydro dynamic input section given by the sec command. The parameter nint is a refinement parameter given nint extra calculation points in between the other points.
*	nsec	This command must be present before the sec commands 1. Number of sections given below
*	sec_type	Type of cross section (1=circular, 2=general)
*	sec	This command must be repeated nsec times and is different for each section type. Section type 1 – circular:

Obl.	Command name	Explanation
		<p>1. Relative distance along the main_body c2_def line. Positive directed from node 1 to node "last".</p> <p>2. C_m inertia coefficient (default=1.0)</p> <p>3. C_d drag coefficient (default=1.0)</p> <p>4. Cross sectional area [m2]</p> <p>5. Cross sectional area to which C_m is related. (default=area for circular sections) [m2]</p> <p>6. Width of construction perpendicular to flow direction [m]</p> <p>7. drdz gradient(optional). For calculating the buoyancy also for conical sections the gradient expressing the change in radius with change of distance along the main_body c2_def line. Only important when buoyancy forces are included.</p> <p>8. Axial drag C_d coefficient for concentrated force contribution (optional). Drag area is circular area defined by the local width. Contribution is quadratic regarding water velocity.</p> <p>9. Axial inertia C_m coefficient for concentrated force contribution (optional). Inertia volume is a sphere defined by the local width as diameter.</p> <p>10. Axial drag C_d coefficient for concentrated force contribution (optional). Drag area is circular area defined by the local width. Contribution is linear regarding water velocity.</p> <p>11. Internal cross sectional area for flooded members [m2] (optional). 0=member is not flooded.</p> <p>12. Torque friction coefficient C_f (optional). For rotating cylinders around local z-direction.</p> $M_z = \frac{1}{16} \rho D^4 \omega^2 C_f$
	buoyancy	1. Specification whether buoyancy forces are included or not. 0=off (default), 1=on (remember to define the 7th parameter in the sec input line).
	update_states	1. Specification whether the hydrodynamic sections are updated in time with respect to pos, vel, acc and orientations, or simply considered to remain fixed. 0=not updated, 1=updated (default)
	update_kinematics	1. Specification whether the water kinematics are updated during iterations or only once per time step. 0=only updated once per time step, 1=full update (default).

Here is an example of this written into the htc-input file.

```

begin HYDRO_ELEMENT ;
  mbdy_name cylinder ;
  buoyancy 1 ;
  update_states 1 ; (0: no dynamic interaction, 1: fully coupled solution
  hydrosections auto 4 ; dist, of hydro calculation points from 1 to nsec
  nsec 2; z    Cm Cd A      Aref  width dr/dz Cd_a_(quad) Cm_a Cd_a_lin Aif
  sec    0.0 1 1 3.404 3.404 2.082 0.0 0.0      0.0 0.0 3.023;
  sec    5.0 1 1 3.404 3.404 2.082 0.0 0.0      0.0 0.0 3.023;
end HYDRO_ELEMENT ;

```

This example shows a flooded cylindrical element (l=5 m, d= 2,082 m and t=60mm).

12.4 Description of the water_kinematics_dll format.

```
subroutine init(inputfile,t0,t1,dt) implicit none
character*(*) :: inputfile
real*8 :: t0 ! start time for simulation
real*8 :: t1 ! stop time for simulation
real*8 :: dt ! time increment
!DEC$ ATTRIBUTES DLLEXPORT, ALIAS:'init'::init
end subroutine init

!-----
subroutine set_new_time(time)
implicit none
!DEC$ ATTRIBUTES DLLEXPORT, ALIAS:'set_new_time'::set_new_time
real*8          :: time

end subroutine set_new_time

!-----
subroutine get_sea_elevation(posxy_h,elevation)
implicit none
!DEC$ ATTRIBUTES DLLEXPORT, ALIAS:'get_sea_elevation'::get_sea_elevation
real*8,dimension(2) :: posxy_h          ! horizontal position coordinates
real*8              :: elevation ! water height above mean water          ! level, positive u

end subroutine get_sea_elevation

!-----
!DEC$ ATTRIBUTES DLLEXPORT, ALIAS:'get_kinematics'::get_kinematics
real*8,dimension(3) :: pos_h,& vel_h,&
acc_h
real*8 :: pres
end subroutine get_kinematics
```

12.5 User manual to the standard wkin.dll version 2.4.

The wkin.dll which is delivered along with the HAWC2 code needs a separate inputfile. The format for these inputs are the same as the HAWC2 main inputfile with usage of begin..end clauses, semi colon separators, exit command etc. Command words are described below.

All command words written below has to be included in an begin .. end clause called wkin_input:

```
begin wkin_input;
...
end wkin_input;
exit;
```

Version info:

- 1-0 TJUL Basic edition by TJUL
- 1-1 ANMH Wave field can be read by file and used directly through fft conversion
- 1-2 TJUL Directional spreading included
- 1-3 ANMH Bug corrected regarding read on seed number using iredular waves

1-4 TJUL Pierson-Moscowitz spectrum added as option
Stream function wave added
Possible pre processing of wave field to speed up simulation time and enable many more coefficients
1-5 TJUL Bug in stream function wave. Static pressure was included - now removed
1-6 TJUL Bug in stream function wave. lateral position was applied instead of vertical in kinematics look-up!!!
1-7 TKIM New wave format for precalculated (high order) wave fields
1-8 ANMH Update in deterministic irregular waves+bugfix
1-9 TJUL New option for white noise wave excitation
2-0 TJUL Bug fix of version 1-9. Version 1-9 had some debug statements included that could mess up the time.
2-1 ANMH Ported to intel
ANMH Correction for high wave numbers in deterministic irregular waves
TJUL Embedded stream function wave, phase velocity used instead of group velocity with respect to pregenerated waves
2-2 TJUL Bug fix. Tightened criteria for jonswap spectrum min-max. Use of real*8 in all internal memory related variables.

12.6 Main commands in the wkin.dll

Obl.	Command name	Explanation
*	wavetype	1. Type of wave used. (0=regular airy, 1=irregular airy, 2=deterministic irregular airy, 3=regular stream function, 4=general wavemode format)
*	wdepth	1. Water depth [m]. Positive value.

12.7 Sub command reg_airy

Command that need to be present if the wavetype equals 0 in the main command.

Obl.	Command name	Explanation
*	stretching	1. Wheeler stretching of waves. (0=off, 1=on)
*	wave	1. Significant wave height H_s [m] 2. Wave period T [s]

12.8 Sub command ireg_airy

Command that need to be present if the wavetype equals 1 in the main command.

Obl.	Command name	Explanation
*	stretching	1. Wheeler stretching of waves. (0=off, 1=on)
*	spectrum	1. Base spectrum used. (1=jonswap, 2= Pierson-Moscowitz)
	jonswap	Jonswap spectrum formulation 1. Significant wave height H_s [m] 2. Wave period T_p [s] 3. γ parameter [-]. A typical value is 3.3
	pm	Pierson-Moscowitz spectrum 1. Significant wave height H_s [m] 2. Wave period T_p [s]

Obl.	Command name	Explanation
	wn	White noise. 1. Target variance level [m^2] 2. f_0 , minimum frequency 3. f_1 , maximum frequency
*	coef	1. Number of coefficients. Normally 200 are used even though higher values are recommended in general. A speed issue... 2. Seed number. A positive integer value.
	spreading	1. Spreading model. (0=none, 1= K_{2s} model also referred to as K_n model) 2. Spreading parameter. If model=1 the parameter is s, a positive integer. The higher value, the less spreading.
	pregen	Pre-generation of a wave field (default is on). Using this option the irregular wave field is calculated during initialization phase and only table look-up is done during the time simulation phase. Very fast and still accurate. 1. Pregen option. (0=traditional approach (slow), 1=pregenerated wave field used (default))
	Embed_strf	Embed stream function wave in time series at the time when the otherwise largest wave occurs. The wave kinematics is blended into the irregular waves before and after. Wave time period equals T_p 1. Wave height H [m]

12.9 Sub command det_airy

Command that need to be present if the wavetype equals 2 in the main command. This command is used when water kinematics needs to be calculated based on a measured elevation time series.

Obl.	Command name	Explanation
*	file	1. File name for measured wave elevation.
*	nsamples	1. Number of lines present in wave elevation file
*	nskip	1. Number of lines to skip before reading of wave elevation file
*	columns	1. Column number for time sensor in file. 2. Column number for wave elevation in file.
	stretching	1. Wheeler stretching of waves. (0=off, 1=on (default))
*	cutoff_frac	1. Fraction of total energy which is discarded in the low and high frequency ranges. Default 1E-5

12.10 Sub command strf

Stream function wave input.

Obl.	Command name	Explanation
*	wave	1. Significant wave height H_s [m] 2. Wave period T [s]

12.11 Sub command wavemods

Command that need to be present if the wavetype equals 4 in the main command. This command is used when water kinematics needs to be calculated based on a measured elevation time series.

Obl.	Command name	Explanation
*	datafile_y	1. Name of datafile where wave kinematic data is present for the horizontal (wave) direction
*	datafile_z	1. Name of datafile where wave kinematic data is present for the vertical direction
*	datafile_nd	1. Number of depth locations
*	datafile_depth	1. Minimum water depth (m)
*	datafile_nt	1. Number of time steps in datafile
*	datafile_t0	1. Time for when wave data is extracted in the datafiles
*	ncol_y	1. Number of columns in datafile1 (time+eta+vel+acc)
*	ncol_z	2. Number of columns in datafile2 (time+vel+acc)

An example of input files with wave kinematics data for the wavemods option is given below. Please note the following:

- The first 9 lines are general comment lines
- Line 10 lists the relative depths, and the number of relative depths must match datafile_Nd in the wavemods subcommand
- Each row starting at Line 12 corresponds to a single time step, and there should be at least datafile_Nt rows before the end of the file
- The datafile columns correspond to time, eta (the distance between the wave height and the MSL; not present in the vertical-component input file), datafile_Nd velocities, and then datafile_Nd accelerations

Example of datafile_y (horizontal wave component):

```
Wave kinematics input to Flex5 Monopile ver. 2.1
General comment line
Wave load program "WaveKin" ver. 1.0
Echo file : Outfile.dat
Name of Case
Wave Description
slope 1:25
50 water depth
3 No rel. depths N
0.000 0.500 1.000
T eta u[1]..u[N] a[1]..a[N]
0.000 -0.645 -0.022 -0.027 -0.047 -0.018 -0.022 -0.035
0.063 -0.659 -0.023 -0.029 -0.049 -0.017 -0.021 -0.032
0.126 -0.671 -0.025 -0.030 -0.051 -0.016 -0.020 -0.030
(etc)
```

Example of datafile_z (vertical wave component):

```
Wave kinematics input to Flex5 Monopile ver. 2.1
General comment line
Wave load program "WaveKin" ver. 1.0
Echo file : Outfile.dat
Name of Case
Wave Description
slope 1:25
```

```

50 water depth
  3 No rel. depths N
0.000 0.500 1.000
  T      u[1]..u[N]      a[1]..a[N]
0.000 -0.022 -0.027 -0.047 -0.018 -0.022 -0.035
0.063 -0.023 -0.029 -0.049 -0.017 -0.021 -0.032
0.126 -0.025 -0.030 -0.051 -0.016 -0.020 -0.030
(etc.)

```

12.12 Wkin.dll example file

```

begin wkin_input ;
  wavetype 1 ;      0=regular, 1=irregular, 2=deterministic
  wdepth 220.0 ;
;
  begin reg_airy ;
    stretching 0;    0=none, 1=wheeler
    wave 9 12.6;    Hs,T
  end;
;
  begin ireg_airy ;
    stretching 0;    0=none, 1=wheeler
    spectrum 1;      (1=jonswap)
    jonswap 9 12.6 3.3 ; (Hs, Tp, gamma)
    coef 200 1 ;      (coefnr, seed)
    spreading 1 2;    (type(0=off 1=on), s parameter (pos. integer min 1))
  end;
;
  begin det_airy ;
    stretching 0;    0=none, 1=wheeler
    file ..\waves\elevation.dat ;
    nsamples 32768 ;
    nskip 1 ;
    columns 1 5 ;    time column, elevation column
  end;
;
  begin wavemods;
    datafile_y ./wavedata/wavekin_y.dat;
    datafile_z ./wavedata/wavekin_z.dat;
    datafile_nt 900; number of time steps in file
    datafile_nd 3; number of relative water depths
    datafile_t0 50; start time for data extraction
    datafile_depth 50 ; minimum water depth
    ncol_y 8; Number of data columns in file
    ncol_z 7; Number of data columns in file
  end;
end;
;
exit ;

```

13 Soil module

13.1 Main command block - soil

In this command block soil spring/damper forces can be attached to a main body. The formulation is performed so it can be used for other external distributed spring/damper systems than soil.

13.2 Sub command block – soil_element

Command block that can be repeated as many times as needed. In this command block the distributed soil spring/damper system is set up for a given main body.

Obl.	Command name	Explanation
*	mbdy_name	1. Main_body name to which the soil calculation points are linked.
*	datafile	1. Filename incl. relative path to file containing soil spring properties (example ./soil/soildata.dat)
*	soilsections	1. Distribution method: (“uniform” only possibility) 2. Number of section (min. 2).
	damping_k_factor	1. Rayleigh kind of damping. Factor the linear stiffness coefficients are multiplied with to obtain the damping coefficients. When the factor is 1.0 the vibration is critically damped for the rigid mainbody connected to the spring and dampers.
♣	set	1. Set number in datafile that is used.

*) Input commands that must be present

♣) Command can be repeated as many times as desired.

13.3 Data format of the soil spring datafile

In the file (which is a text file) different distributed springs can be defined. Each set is located after the “#” sign followed by the set number. Within a set the following data needs to be present.

line 1	“spring type”	(can be “axial”, “lateral” or “rotation_z”)
line 2	“nrow ndefl”	(nrow is number of rows, ndefl is number of deflections (columns))
line 3.. 3+nrow	“z_global F(1) F(2),..., F(ndefl)”	First colum is the spring location (global z coordinate). The following colums are Force/length at the different deflection stations. First deflection must be zero. The forces are assumed symmetrical around the zero deflection.

An example is given below:

This is a nonlinear soil spring demonstration file

```
#1
lateral          (axial/lateral)
5 4             nrow ndefl
0.0 0.1 0.2 1.0  x1 x2 x3 ..... [m]
0.0 0 15 20 500  Z_G F_1 F_2 F_3 .... F_ndefl [kN/m]
10.0 0 15 20 500
```

```

20.0 0 15 20 500
30.0 0 15 20 500
40.0 0 15 20 500
#2
axial (axial/lateral)
5 4 nrow ndefl
0.0 0.1 0.2 1.0 x1 x2 x3 ..... [m]
0.0 0 150 200 5000 Z_G F_1 F_2 F_3 .... F_ndefl [kN/m]
10.0 0 150 200 5000
20.0 0 150 200 5000
30.0 0 150 200 5000
40.0 0 150 200 5000
#3
rotation_z (axial/lateral/rotation_z)
5 4 nrow ndefl
0.0 0.1 0.2 1.0 x1 x2 x3 ..... [rad]
0.0 0 150 200 5000 Z_G M_1 M_2 M_3 .... M_ndefl [kNm/m]
10.0 0 150 200 5000
20.0 0 150 200 5000
30.0 0 150 200 5000
40.0 0 150 200 5000

```

14 External forces through DLL

14.1 Main command block – Force

14.1.1 Sub command - DLL

This command block can be used when a user defined external force is applied to the structure. The main difference between this DLL format and the normal DLL control interface (used with external controllers) is that added stiffness is calculated initially leading to a more robust a fast solution of the coupled system. This force module can with good results be applied for external equivalent soil-springs or hydrodynamic forces for floating constructions or mooring lines.

Obl.	Command name	Explanation and parameters
	dll	1. Filename incl. relative path to the external DLL (example ./dll/force.dll)
	update	1. Name of subroutine in the DLL.
	mbdy	1. Name of main body to which force dll is coupled.
	node	1. Node number of main body to which force dll is couple

14.2 Example of a DLL interface written in fortran90

```
!  
! Demonstration of force DLL  
!  
SUBROUTINE DemoForceDLL(time,x,xdot,xdot2,amat,omega,omegadot,F,M)  
!DEC$ ATTRIBUTES DLLEXPORT::DemoForceDLL  
!DEC$ ATTRIBUTES ALIAS:'demoforcedll' :: DemoForceDLL  
! input  
DOUBLE PRECISION          :: time      ! time  
DOUBLE PRECISION ,DIMENSION(3)  :: x      ! global pos. of reference node  
DOUBLE PRECISION ,DIMENSION(3)  :: xdot   ! global vel. of reference node  
DOUBLE PRECISION ,DIMENSION(3)  :: xdot2  ! global acc. of reference node  
DOUBLE PRECISION ,DIMENSION(3)  :: omega  ! angular vel. of ref. node  
      ! (global base)  
DOUBLE PRECISION ,DIMENSION(3)  :: omegadot ! angular acc. of ref. node  
      ! (global base)  
DOUBLE PRECISION ,DIMENSION(3,3) :: amat   ! rotation matrix (body ->  
      ! global)  
! output  
DOUBLE PRECISION ,DIMENSION(3)  :: F      ! External force in reference  
      ! node (global base)  
DOUBLE PRECISION ,DIMENSION(3)  :: M      ! External moment in reference  
      ! node (global base)  
! locals  
LOGICAL, SAVE              :: bInit = .FALSE. ! Initialization flag  
DOUBLE PRECISION          :: mass = 0.d0    ! Point mass  
!  
! Initialise on first call  
IF (.NOT.bInit) THEN  
  bInit = .TRUE.  
  ! Open file and read mass  
  OPEN(10,FILE="DemoForceDLL_mass.dat")  
  READ(10,*) mass
```

```

    CLOSE(10)
ENDIF
!
! Calc. force
F = mass*((/0.d0,0.d0,9.81d0/) - xdot2)
M = 0.d0
!
END SUBROUTINE DemoForceDLL

```

14.3 Example of a DLL interface written in Lazarus / Pascal

```
library force_dll;
```

```
Type
```

```

    vect = array[0..2] of double;
    mat  = array[0..2,0..2] of double;

```

```

procedure update( var time:double;var x:vect;var xdot:vect;var xdot2:vect;
var amat:mat;var omega:mat;var omegadot:vect;
var F,M:vect);stdcall;

```

```
// Example of applying a step up force in the x-direction:
```

```

begin
if time < 10 then
    F[0] := 0.0;
if time >= 10 then
    F[0] := 20000.0;
if time >= 20 then
    F[0] := 40000.0;
end;

```

```
exports update;
```

```
begin
```

```

    writeln('The DLL force_dll.dll is loaded with succes');
end.

```

15 Output

This command output can either be a main command block or a sub command block within the hawc_dll command block. In the tables below two special columns are introduced. One is only option and the other label option. When the check mark is ‘yes’ in only option it is possible to use only one of the fields if more than one sensor was defined through the command. The sensor that is used is determined by the number following the only command word, see example below.

```
constraint bearing1 shaft_rot 2 only 2;
```

If the only command (and the following number) was omitted two sensors was defined; one for the angle and one for the velocity. With the only command only the velocity sensor is used in the output since the following number is 2.

With the label option it is possible to make a user defined label of the sensor which is written in the sensor list file. The label command is the # symbol. Everything after the # symbol is used as a label. An example of this could be

```
dll inpvec 1 1 # This is a dummy label ;
```

15.1 Commands used with results file writing

When the output command is used for output files (the most normal purpose) some information regarding file name and format needs to be given.

Obl	Command	Explanation
*	filename	1. Filename incl. relative path to outputfile without extension (example ./res/output)
	data_format	ASCII or compressed binary output can be chosen. Default is the ASCII format if nothing is specified. 1. format (‘hawc_ascii’=ASCII format, ‘hawc_binary’=compressed binary format, ‘flex_int’=compressed binary format, ‘gtsdf’=General time series data format (hdf5 based compressed binary), ‘gtsdf64’=General time series data format (hdf5 based binary))
	buffer	Buffer size in terms of time steps. When the buffer is full the data are written to data file. Only used together with the ‘hawc_ascii’, ‘gtsdf’ and ‘gtsdf64’ formats. Default is 3000 time steps 1. 1. buffer size
	deltat	Number of time steps in between outputs. (1=all, 2=every second)
	time	Time start t_0 and stop t_1 for output is defined. Default is the entire simulation length if nothing is specified. 2. t_0 3. t_1

15.2 File format of HAWC_ASCII files

Results are written to an ascii formatted data file with the name assigned to the filename variable (eg. filename ./res/resfil). The data file will have the extension .dat as a standard. The description

of the sensors in the data file is given in another textfile with same filename as the data file but the extension .sel. An example could be: ./res/resfil.dat and ./res/resfil.sel.

In the .sel-file, line number 9 specifies the following parameters: Number of scans, Number of sensors, Duration of output file, Data format (ASCII/BINARY). Example:

```
10 96 20.000 ASCII
```

From line number 13 and onwards, the sensors are specified with the following information: Sensor number, Variable description, unit, Long description. Example:

```
5      bea1 angle_speed          rad/s      pitch1 angle speed
```

Full example of the .sel file:

```
-----
Version ID : HAWC2MB 4.3w
Time : 14:23:28
Date : 22:11.2006
-----
Result file : ./res2_rev0/case41c_nohydro.dat
-----
Scans   Channels   Time [sec]   Format
4500    199           90.000      ASCII

Channel  Variable Description
1        Time                s           Time
2        bea1 angle          deg         shaft_rot angle
3        bea1 angle_speed   rpm         shaft_rot angle speed
4        bea1 angle          deg         pitch1 angle
5        bea1 angle_speed   rad/s      pitch1 angle speed
6        bea1 angle          deg         pitch2 angle
7        bea1 angle_speed   rad/s      pitch2 angle speed
8        bea1 angle          deg         pitch3 angle
9        bea1 angle_speed   rad/s      pitch3 angle speed
-----
```

15.3 File format of HAWC_BINARY files

In this file format results are written to a binary unformatted data file with the name assigned to the filename variable (eg. filename ./res/resfil). The data file will have the extension .dat as a standard. The description of the sensors in the data file is given in another textfile with same filename as the data file but the extension .sel. An example could be: ./res/resfil.dat and ./res/resfil.sel.

The data are scaled to standard 2-byte integers, with a range of 32000 using a scalefactor. The scalefactor is determined for each output sensor

$$s = \frac{MAX(abs(max),abs(min))}{32000}$$

where max and min are the largest and lowest number in the original data for the sensor. These scale factors are written in the end of the accompanying .sel file. When converting a binary

number to the actual number its just a matter of multiplying the binary numbers of a sensor with the corresponding scalefactor.

In the accompanying text file, which has the extension .sel-file, information of the content in the datafile is stored. In line number 9 the following parameters are specified: Number of scans, Number of sensors, Duration of output file, Data format (ASCII/BINARY). Example:

```
10 96 20.000 ASCII
```

From line number 13 and onwards, the sensors are specified with the following information: Sensor number, Variable description, unit, Long description. Example:

```
5      bea1 angle_speed          rad/s      pitch1 angle speed
```

From line number 9+nensors+5 and upwards the scalefactors are written.

Full example of the .sel file:

```
-----
Version ID : HAWC2MB 4.3
Time : 14:23:28
Date : 22:11.2006
-----
Result file : ./res2_rev0/case41c_nohydro.dat
-----
Scans   Channels   Time [sec]   Format
 4500      9       90.000     ASCII

Channel  Variable Description
-----
 1      Time                s           Time
 2      bea1 angle           deg         shaft_rot angle
 3      bea1 angle_speed   rpm        shaft_rot angle speed
 4      bea1 angle           deg         pitch1 angle
 5      bea1 angle_speed   rad/s      pitch1 angle speed
 6      bea1 angle           deg         pitch2 angle
 7      bea1 angle_speed   rad/s      pitch2 angle speed
 8      bea1 angle           deg         pitch3 angle
 9      bea1 angle_speed   rad/s      pitch3 angle speed
-----
Scale factors:
1.56250E-04
5.61731E-03
4.41991E-04
1.00000E+00
1.00000E+00
1.00000E+00
1.00000E+00
1.00000E+00
1.00000E+00
1.00000E+00
```

An important thing to notice is that in the binary data file all sensors are stored sequentially, i.e. all data for sensor 1, all data for sensor 2, etc. This way of storing the data makes later reading of a sensor extra fast since all data for a sensor can be read without reading any data for the other sensor.

15.4 File format for gtsdf and gtsdf64 files

The file formats and reading and writing examples of the gtsdf and gtsdf64 file types and are described here: <https://gitlab.windenergy.dtu.dk/toolbox/WindEnergyToolbox/blob/master/wetb/gtsdf/General%20Time%20Series%20Data%20Format.pdf>

A small matlab code for reading the binary HAWC2 format can be seen below.

```
function sig = ReadHawc2Bin(FileName,path);
% Reads binary HAWC2 results file
% -----
% [t,sig] = ReadFlex4(FileName,Ch);
% filename should be without extension
% -----
% BSKA 26/2-2008
% -----
ThisPath = pwd; cd(path(1,:))

% reading scale factors from *.sel file
fid = fopen([FileName,'.sel'], 'r'); fgets(fid); fgets(fid);
fgets(fid); fgets(fid); fgets(fid); fgets(fid); fgets(fid);
fgets(fid);
tline = fscanf(fid,'%d');
N = tline(1); Nch = tline(2); Time = tline(3); fclose(fid);
ScaleFactor = dlmread([FileName,'.sel'],'',[9+Nch+5 0 9+2*Nch+4
0]);

% reading binary data file
fid = fopen([FileName,'.dat'], 'r'); sig =
fread(fid,[N,Nch],'int16')*diag(ScaleFactor); fclose(fid);

cd(ThisPath)
```

15.5 mbdy (main body output commands)

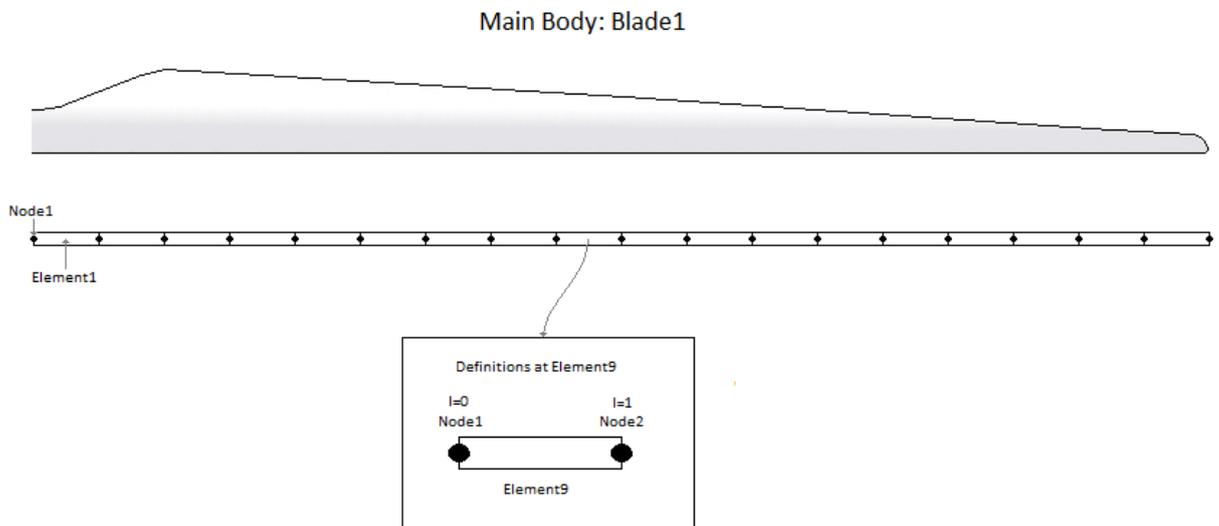
Command 1	Command 2	Explanation	Only option	Label option
mbdy	forcevec	Fx, Fy, Fz shear force vector defined to output. 1. Main_body name 2. Element number 3. Node number on element 4. Main_body name of which coordinate system is used for output. "global" and "local" can also be used. Local is around local beam main bending directions.	yes	yes
mbdy	momentvec	Mx, My, Mz moment vector defined to output. 1. Main_body name 2. Element number 3. Node number on element	yes	yes

Command 1	Command 2	Explanation	Only option	Label option
		4. Main_body name of which coordinate system is used for output. “global” and “local” can also be used. Local is around local beam main bending directions.		
mbdy	forcemomentvec_interp	Fx, Fy, Fz, Mx, My, Mz interpolated shear force and moment vector defined to output. This sensor can write out an interpolated set of cross sectional forces and moments independent of the node discretization. It can also write out in local deformed c2_def coordinates and therefore breaks the limit of using element coordinates. 1. Main_body name 2. Position of location outputted: ‘c2def’ or ‘default’ (default = elastic center). 3. Name of mbdy used for output coordinate system: mbdy_name, ‘global’, ‘local_aero’ or ‘local_element’ 4. Distance along c2_def to output location 5. Sign multiplied to output: 1.0 or -1.0	yes	yes
mbdy	state	Vector with 3 components of either position, velocity or acceleration of a point on an element defined to output. If ‘acg’ is used, the acceleration including the gravity contribution is written. 1. State: ‘pos’, ‘vel’, ‘acc’, ‘acg’ (“pos”=position, “vel”=velocity, “acc”=acceleration) 2. Main_body name 3. Element number 4. Relative distance from node 1 to node 2 on element 5. Main_body name of which coordinate system is used for output. “global” can also be used.	yes	yes
mbdy	state_at	Vector with 3 components of either position, velocity or acceleration of a point on an element defined to output. The point is offset from the element z axis by an x and y distance in element coordinates. 1. State: ‘pos’, ‘vel’ or ‘acc’ 2. Main_body name 3. Element number 4. Relative distance from node 1 to node 2 on element 5. Main_body name of which coordinate system is used for output. “global” can also be used. 6. x-coordinate offset [m] 7. y-coordinate offset [m]	yes	Yes
mbdy	state_at2	Vector with 3 components of either position, velocity or acceleration of a point on an element defined to output. The point is offset from the c2_def centerline z axis by an x and y distance in local c2def centerline coordinates. 1. State: ‘pos’, ‘vel’ or ‘acc’	yes	Yes

Command 1	Command 2	Explanation	Only option	Label option
		2. Main_body name 3. Element number 4. Relative distance from node 1 to node 2 on element 5. Main_body name of which coordinate system is used for output. "global" can also be used. 6. x-coordinate offset [m] 7. y-coordinate offset [m]		
mbody	state_rot	Vector with components of either axis and angle (angle [rad], r_1, r_2, r_3), euler parameters (quaternions r_0, r_1, r_2, r_3), euler angles, rotation velocity (-vector) or rotation acceleration (-vector) of a point on an element defined to output. For the sensor eulerang_xyx a set of euler angles are created based on the orientation matrix. Be aware that the method used is only valid for rotations in the intervals ($\theta_x \pm 180^\circ$, $\theta_y \pm 90^\circ$, $\theta_z \pm 180^\circ$). The method proj_ang can be used to see how much a blade tip rotates around the pitch axis, but be aware that the angles are how the element is oriented and not necessarily how the local chord is rotated. With the command proj_ang the angles are obtained from the local element orientation 3x3 matrix T_e , seen from the chosen coordinate system using the Atan2 functions (rot_x=atan2[$T_e(2,3), T_e(3,3)$], rot_y=atan2[$T_e(3,1), T_e(1,1)$], rot_z=atan2[$T_e(1,2), T_e(2,2)$]). 1. State : 'axisangle', 'eulerp', 'eulerang_xyz', 'omega', 'omegadot' or proj_ang 2. Main_body name 3. Element number 4. Relative distance from node 1 to node 2 on element 5. Main_body name of which coordinate system is used for output. "global" can also be used.	yes	Yes
mbody	statevec_new	This sensor writes out the position vector and orientation vector for a point on the structure. The orientation vector is a direction vector to which the structure is rotated and the vector length is the size of this rotation. There is a direct relation between this vector and the 3x3-orientation matrix, but it is easier to overview as each single element corresponds to a 2D projected rotation (rot_x, rot_y, rot_z). Furthermore it can write out the orientation of the local deformed c2_def coordinates system and therefore breaks the limit of only looking at element orientations. 1. Main_body name	yes	Yes

Command 1	Command 2	Explanation	Only option	Label option
		2. Position of location outputted: 'c2def' or 'default' (default = elastic center). 3. Name of mbdy used for output coordinate system: mbdy_name, 'global', 'local_aero' or 'local_element' 4. State: 'elastic' or 'absolute'. Elastic means that initial location is subtracted results 5. Distance along c2_def to output location 6. Sign multiplied to output: 1.0 or -1.0 7. x-coordinate offset from center to a point where location is outputted (local c2def coo) [m] 8. y-coordinate offset from center to a point where location is outputted (local c2def coo) [m]		

This illustration shows how the sensors are placed on an element in terms of local nodes and relative distance.



15.6 Constraint (constraint output commands)

15.6.1 bearing1

Command 1	Command 2	Explanation	Only option	Label option
constraint	bearing1	Bearing angle and angle velocity defined to output 1. bearing1 name 2. unit of output (1:angle [unit=rad, range $-\pi:\pi$], vel [rad/s]; 2:angle [unit=deg, range 0:360], vel [rpm]; 3:angle [unit=deg, range 0:360], vel [rad/s]; 4:angle [unit=deg, range -180:180], vel [rad/s]; 5:angle [unit=deg, range -180:180], vel [deg/s])	Yes	No

15.6.2 bearing2

Command 1	Command 2	Explanation	Only option	Label option
constraint	bearing2	Bearing angle and angle velocity defined to output 1. bearing2 name 2. unit of output (1:angle [unit=rad, range $-\pi:\pi$], vel [rad/s]; 2:angle [unit=deg, range 0:360], vel [rpm]; 3:angle [unit=deg, range 0:360], vel [rad/s]; 4:angle [unit=deg, range -180:180], vel [rad/s]; 5:angle [unit=deg, range -180:180], vel [deg/s])	Yes	No

15.6.3 bearing3

Command 1	Command 2	Explanation	Only option	Label option
constraint	bearing3	Bearing angle and angle velocity defined to output 1. bearing3 name 2. unit of output (1:angle [unit=rad, range $-\pi:\pi$], vel [rad/s]; 2:angle [unit=deg, range 0:360], vel [rpm]; 3:angle [unit=deg, range 0:360], vel [rad/s]; 4:angle [unit=deg, range -180:180], vel [rad/s]; 5:angle [unit=deg, range -180:180], vel [deg/s])	Yes	No

15.6.4 bearing4

Rotation angle and velocity of the two axis perpendicular to the cardan shaft torsion axis are outputted.

Command 1	Command 2	Explanation	Only option	Label option
constraint	bearing4	Bearing angle and angle velocity defined to output 1. bearing4 name 2. unit of output (1:angle [unit=rad, range $-\pi:\pi$], vel [rad/s]; 2:angle [unit=deg, range 0:360], vel [rpm]; 3:angle [unit=deg, range 0:360], vel [rad/s]; 4:angle [unit=deg, range -180:180], vel [rad/s]; 5:angle [unit=deg, range -180:180], vel [deg/s])	Yes	No

15.7 aero (aerodynamic related commands)

Command 1	Command 2	Explanation	Label option
aero	time	Simulation time to output. No parameters.	No
aero	azimuth	Azimuth angle of selected blade. Zero is vertical downwards. Positive clockwise around blade root y-axis. Unit [deg] 1. Blade number	No
aero	omega	Rotational speed of rotor. Unit [rad/s]	No

Command 1	Command 2	Explanation	Label option
aero	vrel	Relative velocity in x-y local aerodynamic plane. Unit [m/s] 1. Blade number 2. Radius [m] (nearest inner calculation point is used)	No
aero	vrel_3d	Relative velocity in x-y-z local aerodynamic space. Unit [m/s] 3. Blade number 4. Radius [m] (nearest inner calculation point is used)	No
aero	alfa	Angle of attack in x-y local aerodynamic plane. Unit [deg] 1. Blade number 2. Radius [m] (nearest inner calculation point is used)	No
aero	alfadot	Pitch rate term (z-axis rotation) in local aerodynamic plane, as used for non-circulatory contributions. Unit [rad/s] 1. Blade number 2. Radius [m] (nearest inner calculation point is used)	No
aero	sideslip	Side slip angle (from radial flow of BEM expansion) 1. Blade number 2. Radius [m] (nearest inner calculation point is used)	No
aero	beta	Flap deflection angle (matching the deflection specified by the flap control .dll): 1. Blade number 2. Flap number, according to the order defined in the dynstall_ateflap sub-command block.	No
aero	cl	Instantaneous lift coefficient. Unit [-] 1. Blade number 2. Radius [m] (nearest inner calculation point is used)	No
aero	cd	Instantaneous drag coefficient. Unit [-] 1. Blade number 2. Radius [m] (nearest inner calculation point is used)	No
aero	cm	Instantaneous moment coefficient. Unit [-] 1. Blade number 2. Radius [m] (nearest inner calculation point is used)	No
aero	lift	Lift force at calculation point. Unit [kN/m] 1. Blade number 2. Radius [m] (nearest inner calculation point is used)	No
aero	drag	Drag force at calculation point. Unit [kN/m] 1. Blade number 2. Radius [m] (nearest inner calculation point is used)	No

Command 1	Command 2	Explanation	Label option
aero	moment	Aerodynamic moment at calculation point. Unit [kNm/m] 1. Blade number 2. Radius [m] (nearest inner calculation point is used)	No
aero	secforce	Aerodynamic force at calculation point. Local aero coo. Unit [kN/m] 1. Blade number 2. Dof number (1= F_x , 2= F_y , 3= F_z) 3. Radius [m] (nearest inner calculation point is used)	No
aero	secmoment	Aerodynamic moment at calculation point. Local aero coo. Unit [kN/m] 1. Blade number 2. Dof number (1= M_x , 2= M_y , 3= M_z) 3. Radius [m] (nearest inner calculation point is used)	No
aero	int_force	Integrated aerodynamic forces from tip to calculational point. NB the integration is performed around the $C_{3/4}$ location. Unit [kN] 1. Coordinates system (1=local aero coo, 2=blade ref. system, 3=global, 4=rotor polar) 2. Blade number 3. Dof number (1= M_x , 2= M_y , 3= M_z) 4. Radius [m] (nearest inner calculation point is used)	No
aero	int_moment	Integrated aerodynamic moment from tip to calculational point. NB the integration is performed around the $C_{3/4}$ location. Unit [kN] 1. Coordinates system (1=local aero coo, 2=blade ref. system, 3=global, 4=rotor polar) 2. Blade number 3. Dof number (1= M_x , 2= M_y , 3= M_z) 4. Radius [m] (nearest inner calculation point is used)	No
aero	torque	Integrated aerodynamic forces of all blades to rotor torsion. Unit [kNm]. No parameters	No
aero	thrust	Integrated aerodynamic forces of all blades to rotor thrust. Unit [kN]. No parameters	No
aero	position	Position of calculation point. Unit [m]. Please be aware that if the blade ref system is used, the orientation is in the blade coo, but the origo is still in the hub center. 1. Coordinates system (1=local aero coo, 2=blade ref. system, 3=global, 4=rotor polar) 2. Blade number 3. Dof number (1= M_x , 2= M_y , 3= M_z) 4. Radius [m] (nearest inner calculation point is used)	No

Command 1	Command 2	Explanation	Label option
aero	power	Integrated aerodynamic forces of all blades to rotor torsion multiplied by the rotor speed. Unit [kNm]. No parameters	No
aero	rotation	Orientation of calculation point. Unit [deg]. 1. Blade number 2. Dof number (1= θ_x , 2= θ_y , 3= θ_z) 3. Radius [m] (nearest inner calculation point is used) 4. Coordinates system (1=blade_ref. coo, 2=rotor polar coo.)	No
aero	rotation_e	Orientation of calculation point. Unit [deg]. 1. Blade number 2. Dof number (1= θ_x , 2= θ_y , 3= θ_z) 3. Radius [m] (nearest inner calculation point is used) 4. Coordinates system (1=blade_ref. coo, 2=rotor polar coo.)	No
aero	velocity	Velocity of calculation point. Unit [m/s]. 1. Coordinates system (1=local aero coo, 2=blade ref. system, 3=global, 4=rotor polar) 2. Blade number 3. Dof number (1= V_x , 2= V_y , 3= V_z) 4. Radius [m] (nearest inner calculation point is used)	No
aero	acceleration	Acceleration of calculation point. Unit [m/s ²]. 1. Coordinates system (1=local aero coo, 2=blade ref. system, 3=global, 4=rotor polar) 2. Blade number 3. Dof number (1= V_x , 2= V_y , 3= V_z) 4. Radius [m] (nearest inner calculation point is used)	No
aero	tors_e	Aeroelastic torsional twist minus initial static twist of a blade section. 1. Blade number 2. Radius [m] (nearest inner calculation point is used)	No
aero	windspeed	Free wind speed seen from the blade. Unit [m/s] 1. Coordinates system (1=local aero coo, 2=blade ref. system, 3=global, 4=rotor polar) 2. Blade number 3. Dof number (1= V_x , 2= V_y , 3= V_z) 4. Radius [m] (nearest inner calculation point is used)	No
aero	spinner_lidar	Sensor emulating a spinner mounted lidar 1. Measurement type (1=single point, 3=volume average) 2. Scan type (1=circular scan, 2=horizontal line (sine sweep), 3=horizontal line (linear sweep), 4=circular 2D scan) 3. Focus length [m] 4. Measurement angle [deg] 5. Scanning velocity [rev/sec]	No

Command 1	Command 2	Explanation	Label option
		6. Velocity fraction (2D scan) 7. Aperture radius (volume) [m] 8. Number of points in volume scan 9. Wavelength (Volumen) [m]	
aero	induc	Local induced velocity at calculation point. Unit [m/s] 1. Coordinates system (1=local aero coo, 2=blade ref. system, 3=global, 4=rotor polar) 2. Blade number 3. Dof number (1= V_x , 2= V_y , 3= V_z) 4. Radius [m] (nearest inner calculation point is used)	No
aero	induc_sector_ct	Thrust coefficient at a position on the rotor. Unit [-] 1. Radius [m] 2. Azimuth angle (zero downwards) [deg]	No
aero	induc_sector_cq	Torque coefficient at a position on the rotor. Unit [-] 1. Radius [m] 2. Azimuth angle (zero downwards) [deg]	No
aero	induc_sector_a	Axial induction coefficient at a position on the rotor. Unit [-] 1. Radius [m] 2. Azimuth angle (zero downwards) [deg]	No
aero	induc_sector_am	Tangential induction coefficient at a position on the rotor. Unit [-] 1. Radius [m] 2. Azimuth angle (zero downwards) [deg]	No
aero	induc_a_norm	Axial velocity used in normalization expression of rotor thrust coefficients. The average axial wind velocity incl. induction. Unit [m/s]. No parameters.	No
aero	induc_am_norm	Tangential velocity used in normalization expression of torque coefficient. Average tangential velocity at a given radius. Unit [m/s]. 1. Radius [m]	No
aero	inflow_angle	Angle of attack + rotation angle of profile related to polar coordinates (not pitching). Unit [deg] 1. Blade number 2. Radius [m] (nearest inner calculation point is used)	No
aero	dclalpha	Gradient $dCl/d\alpha$. Unit [deg^{-1}] 1. Blade number 2. Radius [m] (nearest inner calculation point is used)	No
aero	dcddalpha	Gradient $dCd/d\alpha$. Unit [deg^{-1}] 1. Blade number 2. Radius [m] (nearest inner calculation point is used)	No
aero	gamma	Circulation strength at calculation point. Unit [m ² /s]	No

Command 1	Command 2	Explanation	Label option
		1. Blade number 2. Radius [m] (nearest inner calculation point is used)	
aero	lambda	Tip speed ratio, Unit []	No
aero	windspeed_boom	Free wind speed seen by a boom mounted on a blade section. Coordinate system used “blade ref. system”. Unit [m/s]. 1. Blade number 2. Radius [m] (nearest inner calculation point is used) 3. Boom-length X, measured from half chord point positive towards LE [m] 4. Boom-length Y, measured from half chord point positive towards pressureside [m]	No
aero	actuatordiskload	Actuator disk load provide normalized load export for the Actuator Disk Model. 1. DOF (1=Ft, 2=Fa, 3=Fr) 2. Radius [m] (nearest inner calculation point is used)	No
aero	grid_radius_nd	Aerodynamic calculation point non-dim radius r/R 1. Number of radial stations outputted (should normally correspond to number of aerodynamic calculation points on a blade)	No
aero	vawt_induc_x	Induction for a VAWT outputted in tangential polar coordinates 1. disc number 2. azimuth number	No
aero	vawt_induc_y	Induction for a VAWT outputted in radial polar coordinates 1. disc number 2. azimuth number	No
aero	nacelle_lidar	Model of a single-beam CW nacelle-mounted lidar. Laser beam is approximated by a line with a Lorentzian shaped weighting function. Outputs are: 1. Line-of-sight velocity [m/s] 2. Doppler spectrum variance [m2/s2] 3. Global x position of focus point 4. Global y position of focus point 5. Global z position of focus point Input are: 1. Mounting distance from rotor center in global x coordinates [m] 2. Mounting distance from rotor center in global y coordinates [m] 3. Mounting distance from rotor center in global z coordinates [m] 4. Half-cone opening angle of beam [deg] 5. Azimuth angle of beam measured (clockwise as seen from turbine) from vertical up position [deg] 6. Focus length measured from rotor center (along the beam) [m]	No

Command 1	Command 2	Explanation	Label option
		7. Rayleigh length of beam (Gamma) [m] 8. half-width of integration interval over probe volume [Gamma] 9. Number of integration points (recommendation: 100)[-] 10. Beam identifier number [-]	
aero	effective_wind_speed	Estimation of rotor effective wind speed as a (weighted) average of longitudinal wind speeds within the rotor area: $v_{eff} = \sqrt{\frac{n \int_0^{2\pi} \int_0^R v_u^n(r, \varphi) w(r, \varphi) r dr d\theta}{\int_0^{2\pi} \int_0^R w(r, \varphi) r dr d\theta}}. \text{ Unit [m/s]}$ Inputs are: 1. Number of blades [-] 2. Rotor radius (R) [m] 3. Tip speed ratio at rated wind speed [-] (only used when input 9 is equal to 3) 4. Exclusion of root part [R] (only used when input 9 is equal to 3) 5. Normal measurement distance from rotor plane [m] 6. Width of turbulence box [m] (use the values from the turbulence box block) 7. Number of integration points along width/height [-] 8. power to weight wind speed with (n) [-] 9. Weighting method: 1: arithmetic mean 2: dCpdr weight w/o losses 3: dCpdr weight with (tip and root) losses 10. Optimum axial induction factor (only used when input 9 is equal to 2 or 3)	No

For multi-rotor simulation, three commands are used: - Command 1: aero_mr - Command 2: name of rotor given in main command block; aero - Command 3: as command 2 from above table

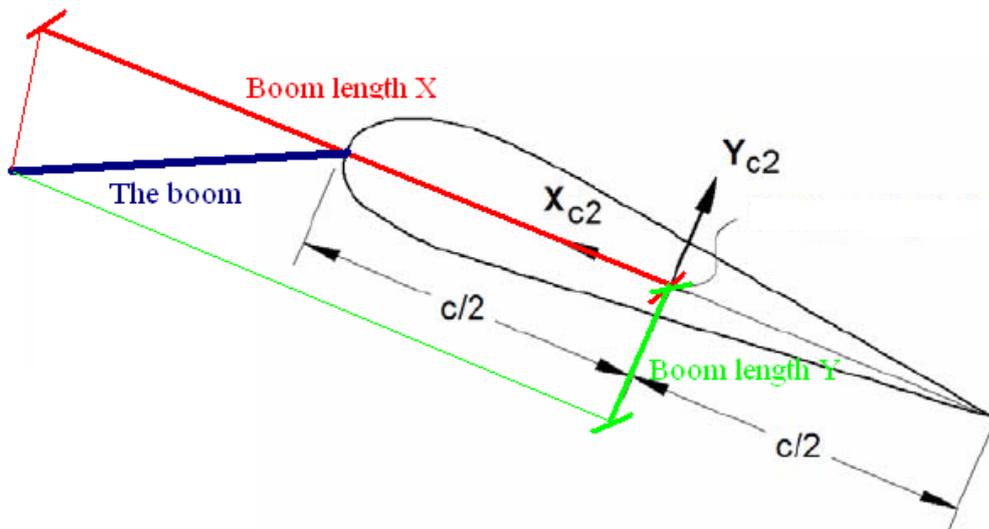


Figure 7: Illustration of the boom coordinates used by the “windspeed_boom” command.

15.8 wind (wind output commands)

Command 1	Command 2	Explanation	Only option	Label option
wind	free_wind	Wind vector V_x, V_y, V_z , (wind as if the turbine didn't exist). 1. Coordinate system (1=global, 2=non rotating rotor coordinates (x always horizontal, y always out-of-plane)) 2. x-pos (global coo) 3. y-pos (global coo) 4. z-pos (global coo)	Yes	Yes
wind	free_wind_center_pos0	Wind vector V_x, V_y, V_z , (wind as if the turbine didn't exist). 1. Coordinate system (1=global, 2=non rotating rotor coordinates (x always horizontal, y always out-of-plane)) _center_pos0	Yes	Yes
wind	free_wind_hor	Horizontal wind component velocity [m/s] and direction [deg] defined to output. Dir=0 when wind equals y-dir. 1. Coordinate system (1=global, 2=non rotating rotor coordinates (x always horizontal, y always out-of-plane)) 2. x-pos (global coo) 3. y-pos (global coo) 4. z-pos (global coo)	Yes	Yes
wind	free_wind_-hor_center_pos0	Horizontal wind component velocity [m/s] and direction [deg] defined to output. Dir=0 when wind equals y-dir. 1. Coordinate system (1=global, 2=non rotating rotor coordinates (x always horizontal, y always out-of-plane))	Yes	Yes
wind	free_wind_shadow	As sensor “free_wind”, but with tower shadow included.	Yes	Yes

Command 1	Command 2	Explanation	Only option	Label option
		1. Coordinate system (1=global, 2=non rotating rotor coordinates (x always horizontal, y always out-of-plane)) 2. x-pos (global coo) 3. y-pos (global coo) z-pos (global coo)		

15.9 wind_wake (wind wake output commands)

Command 1	Command 2	Explanation	Only option	Label option
wind_wake	wake_pos	Position of the wake deficit center after the meandering proces to the downstream end position. x,y and z position is written in meteorological coordinates $(x, y, z)_M = (u, v, w)$ with origo in the position defined with center_pos0 in the general wind commands. 1. wake source number	Yes	Yes

15.10 dll (DLL output commands)

Command 1	Command 2	Explanation	Label option
dll	inpvec	Value from DLL input vector is defined to output 1. DLL number 2. array index number	yes
dll	outvec	Value from DLL output vector is defined to output 1. DLL number 2. array index number	yes
dll	hawc_dll	Special output commands for the "hawc_dll" format. With this command the dll name can be used in the output definitions 1. string. Reference name of the dll given in the begin – end hawc_dll input definitions. 2. string. "outvec" or "inpvec" can be used. Same definition as previously written above. 3. Channel number in the in or out going array.	yes
dll	type2_dll	Special output commands for the "type2_dll" format. With this command the dll name can be used in the output definitions 1. string. Reference name of the dll given in the begin – end hawc_dll input definitions. 2. string. "outvec" or "inpvec" can be used. Same definition as previously written above. 3. Channel number in the in or out going array.	yes
dll	sensor_id	Name of sensor_id defined for other output sensor 1. Sensor number if sensor id refers to a vector	

15.11 hydro (hydrodynamic output commands)

Command 1	Command 2	Explanation	Only option	Label option
hydro	water_surface	Water surface level at a given horizontal location is defined to output (global coordinates). Unit [m] 1. x-pos 2. y-pos	No	No
hydro	water_vel_acc	Water velocity V_x, V_y, V_z , and acceleration A_x, A_y, A_z vectors defined to output. Unit [m/s] and [m/s ²]. 1. x-pos 2. y-pos 3. z-pos	Yes	No
hydro	fm	Inertia force F_x, F_y, F_z contribution from Morisons formula in a given calculation point. Unit [kN] 1. hydro element number 2. radius 3. coordinate system (1=global, 2=local hydro sec coo)	Yes	No
hydro	fd	Drag force F_x, F_y, F_z contribution from Morisons formula in a given calculation point. Unit [kN] 1. hydro element number 2. sec number 3. coordinate system (1=global, 2=local hydro sec coo)	Yes	No

15.12 general (general output commands)

Command 1	Command 2	Explanation	Label option
general	constant	A constant value is send to output 1. constant value	No
general	step	A step function is created. This function changes from f_0 to f_1 at time t_0 . 1. t_0 [sec] 2. f_0 3. f_1	No
general	step2	A step function is created. This function changes from f_0 to f_1 between time t_0 and t_1 using linear interpolation. 1. t_0 [sec] 2. t_1 [sec] 3. f_0 4. f_1	No
general	step3	A step function is created. This function changes from f_0 to f_1 between time t_0 and t_1 using a continous sinus2 interpolation function. 1. t_0 [sec] 2. t_1 [sec] 3. f_0 4. f_1	No
general	time	The time is send to output. No parameters	No

Command 1	Command 2	Explanation	Label option
general	deltat	The time increment is send to output. No parameters	No
general	harmonic	A harmonic function is send to output $F(t) = A \sin(2\pi f_0 t) + k$ 1. A 2. f_0 3. k	No
general	harmonic2	A harmonic function is send to output $F(t) = \begin{cases} 0 & t < t_0 \\ A \sin(2\pi f_0(t - t_0)) + k & t_0 \leq t \leq t_1 \\ 0 & t > t_1 \end{cases}$ 1. A 2. f_0 3. k 4. t_0 5. t_1	No
general	stairs	A series of steps resulting in a staircase signal is created. 1. t_0 time for first step change [s] 2. f_0 start value of function 3. Step size 4. Step duration [s] 5. Number of steps	No
general	status	A status flag (mainly for controller purpose) is written. A first time step and first iteration the output value is 0. During the rest of the simulation the value is 1 until last time step where the value is -1.	No
general	random	A randon (uniform distribution) is written 1. lower limit 2. upper limit 3. seed number	No
general	impulse	A step function which return to zero after a certain duration 1. t_0 time for impulse start [s] 2. Impulse duration [s] 3. f_0 impulse level	No
general	sensor_id	Sensor name. 1. Sensor number if sensor_id refers to a vector	No

15.13 Output_at_time (output at a given time)

This command is especially usefull if a snapshot of loads or other properties are required at a specific time. This is mostly used for writing calculated aerodynamic properties as function of

blade location. The command block can be repeated as many times as needed (e.g. if outputs at more than one time is needed)

The command must be written with the following syntax

`output_at_time keyword time`

where *keyword* is a command listed in the subsections below. So far only the command `aero` is present. The last command word `time` is the time in seconds from simulation start to which the output are written.

15.13.1 `aero` (aerodynamic output commands)

The first line in the `output_at` block must be the information regarding which file the outputs are written (the filename command listed in the table below)

Command	Explanation	Label option
<code>filename</code>	Filename incl. relative path to output file (example <code>./output/output_at.dat</code>). 1. filename	No
<code>alfa</code>	Angle of attack [deg]. 1. Blade number	No
<code>alfadot</code>	Pitch rate term (z-axis rotation) in local aerodynamic plane, as used for non-circulatory contributions. Unit [rad/s]. 1. Blade number	No
<code>vrel</code>	Relative velocity [m/s] 1. Blade number	No
<code>cl</code>	Lift coefficient [-] 1. Blade number	No
<code>cd</code>	Drag coefficient [-] 1. Blade number	No
<code>cm</code>	Moment coefficient [-] 1. Blade number	No
<code>lift</code>	Lift force L [N] 1. Blade number	No
<code>drag</code>	Drag force D [N] 1. Blade number	No
<code>moment</code>	Moment force M [Nm] 1. Blade number	No
<code>secforce</code>	Aerodynamic forces [N] 1. Blade number 2. DOF number (1=x,2=y,3=z) 3. Coordinate system (1=aero, 2=blade, 3=global, 4=rotor polar)	No
<code>secmoment</code>	Aerodynamic moments [Nm] 1. Blade number 2. DOF number (1=x,2=y,3=z) 3. Coordinate system (1=aero, 2=blade, 3=global, 4=rotor polar)	No
<code>int_force</code>	Aerodynamic forces integrated from tip to given radius [N] 1. Blade number 2. DOF number (1=x,2=y,3=z) 3. Coordinate system (1=aero, 2=blade, 3=global, 4=rotor polar)	No
<code>int_moment</code>	Aerodynamic moment integrated from tip to given radius [N] 1. Blade number	No

Command 1	Explanation	Label option
	2. DOF number (1=x,2=y,3=z) 3. Coordinate system (1=aero, 2=blade, 3=global, 4=rotor polar)	
inipos	Initial position of sections in blade coo [m] 1. Blade number 2. DOF number (1=x,2=y,3=z)	No
position	Actual position of section [m] 1. Blade number 2. DOF number (1=x,2=y,3=z) 3. Coordinate system (1=aero, 2=blade, 3=global, 4=rotor polar)	No
velocity	Actual velocity of section [m/s] 1. Blade number 2. DOF number (1=x,2=y,3=z) 3. Coordinate system (1=aero, 2=blade, 3=global, 4=rotor polar)	No
acceleration	Actual acceleration of section [m/s] 1. Blade number 2. DOF number (1=x,2=y,3=z) 3. Coordinate system (1=aero, 2=blade, 3=global, 4=rotor polar)	No
ct_local	Local thrust coefficient [-]. Calculated based on the expression $C_t = \frac{F_{axial} B}{\frac{1}{2} \rho 2\pi r V_{inf}^2}$ 1. Blade number	No
cq_local	Local tangential force coefficient [-]. Calculated based on the expression $C_q = \frac{F_{tan} B}{\frac{1}{2} \rho 2\pi r V_{inf}^2}$ 1. Blade number	No
chord	Chord length [m] 1. Blade number	No
induc	Induced velocity [m/s] 1. Blade number 2. DOF number (1=x,2=y,3=z) 3. Coordinate system (1=aero, 2=blade, 3=global, 4=rotor polar)	No
windspeed	Free windspeed (without induction but incl. tower shadow effects if used) [m/s] 1. Blade number 2. DOF number (1=x,2=y,3=z) 3. Coordinate system (1=aero, 2=blade, 3=global, 4=rotor polar)	No
inflow_angle	Angle of attack + rotation angle of profile related to polar coordinates (not pitching). Unit [deg] 1. Blade number	No
dcldalfa	Gradient $dC_l/d\alpha$. Unit [deg^{-1}] 1. Blade number	No
dcddalfa	Gradient $dC_d/d\alpha$. Unit [deg^{-1}] 1. Blade number	No
tiploss_f	The local Prandtl tiploss factor f is written 1. Blade number	No

A Example of main input file

```
begin Simulation;
  time_stop 100;
  solvetype 1 ; (newmark)
  on_no_convergence continue ;
  logfile ./log/oc3_monopile_phase_1.log ;
  animation ./animation/oc3_monopile_phase_1.dat;
;
begin newmark;
  deltat 0.02;
end newmark;
end simulation;
;
begin new_htc_structure;
  ; Optional - Calculated beam properties of the bodies are written to file:
  beam_output_file_name ./log/oc3_monopile_phase_1_beam.dat;
  ; Optional - Body initial position and orientation are written to file:
  body_output_file_name ./log/oc3_monopile_phase_1_body.dat;
; body_eigenanalysis_file_name ./eigenfrq/oc3_monopile_phase_1_body_eigen.dat;
; structure_eigenanalysis_file_name ./eigenfrq/oc3_monopile_phase_1_strc_eigen.dat ;
;-----
;
begin main_body;          monopile 30m
  name      monopile ;
  type      timoschenko ;
  nbodies   1 ;
  node_distribution  c2_def ;
  damping   4.5E-02 4.5E-02 8.0E-01 1.2E-03 1.2E-03 4.5E-04 ;
  begin timoschenko_input;
    filename ./data/Monopile.txt ;
    set 1 1 ;          set subset 1=flexible,2=stiff
  end timoschenko_input;
  begin c2_def;          Definition of centerline (main_body coordinates)
    nsec 7;
    sec 1 0.0 0.0 0.0 0.0 ; x,y,z,twist Mudline
    sec 2 0.0 0.0 -0.1 0.0 ; x,y,z,twist
    sec 3 0.0 0.0 -10.0 0.0 ; x,y,z,twist 50% between mudline and MSL
    sec 4 0.0 0.0 -15.0 0.0 ; x,y,z,twist
    sec 5 0.0 0.0 -20.0 0.0 ; x,y,z,twist MWL
    sec 6 0.0 0.0 -25.0 0.0 ;
    sec 7 0.0 0.0 -30.0 0.0 ; Monopile flange
  end c2_def ;
end main_body;
;
begin main_body;          tower 80m
  name      tower ;
  type      timoschenko ;
  nbodies   1 ;
  node_distribution  c2_def ;
  damping_posdef 6.456E-4 6.45E-4 1.25E-3 1.4E-3 1.4E-3 1.25E-3 ;
  ;damping_posdef Mx My Mz Kx Ky Kz , M's raises overall level, K's raises high frequency level
  ;
  begin timoschenko_input;
```

```

filename ./data/NREL_5MW_st.txt ;
set 1 1 ;
end timoschenko_input;
begin c2_def;          Definition of centerline (main_body coordinates)
nsec 8;
sec 1 0.0 0.0 0.0    0.0 ; x,y,z,twist
sec 2 0.0 0.0 -10.0  0.0 ;
sec 3 0.0 0.0 -20.0  0.0 ;
sec 4 0.0 0.0 -30.0  0.0 ;
sec 5 0.0 0.0 -40.0  0.0 ;
sec 6 0.0 0.0 -50.0  0.0 ;
sec 7 0.0 0.0 -60.0  0.0 ;
sec 8 0.0 0.0 -77.6  0.0 ;
end c2_def ;
end main_body;
;
begin main_body;
name      towertop ;
type      timoschenko ;
nbodies   1 ;
node_distribution    c2_def ;
; damping_posdef  9.025E-06 9.025E-06 8.0E-05 8.3E-06 8.3E-06 8.5E-05 ;
damping  2.50E-04  1.40E-04  2.00E-03  3.00E-05  3.00E-05  2.00E-04 ;
;
;Nacelle mass and inertia:
concentrated_mass 2 0.0  1.9 0.21256 2.4E5 1741490.0  1.7E5 1741490.0 ;
begin timoschenko_input;
filename ./data/NREL_5MW_st.txt ;
set 2 1 ;
end timoschenko_input;
begin c2_def;          Definition of centerline (main_body coordinates)
nsec 2;
sec 1 0.0 0.0 0.0    0.0 ; x,y,z,twist
sec 2 0.0 0.0 -1.96256  0.0 ;
end c2_def ;
end main_body;
;
begin main_body;
name      shaft ;
type      timoschenko ;
nbodies   1 ;
node_distribution    c2_def ;
; damping_posdef  7.00E-3  7.00E-03  7.00E-02  3.48E-04  3.48E-04  1.156E-03 ;
damping_posdef  7.00E-3  7.00E-03  7.00E-02  6.5E-04  6.5E-04  1.84E-02 ;
concentrated_mass 1 0.0 0.0 0.0 0.0 0.0 0.0 5025497.444 ;generator equivalent slow shaft
concentrated_mass 5 0.0 0.0 0.0 56780 0.0 0.0 115926 ; hub mass and inertia;
begin timoschenko_input;
filename ./data/NREL_5MW_st.txt ;
set 3 1 ;
end timoschenko_input;
begin c2_def;          Definition of centerline (main_body coordinates)
nsec 5;
sec 1 0.0 0.0 0.0    0.0 ; Tower top x,y,z,twist
sec 2 0.0 0.0 1.0    0.0 ;

```

```

        sec 3 0.0 0.0 2.0    0.0 ;
        sec 4 0.0 0.0 3.1071 0.0 ; Main bearing
        sec 5 0.0 0.0 5.0191 0.0 ; Rotor centre
    end c2_def ;
end main_body;
;
begin main_body;
    name      hub1 ;
    type      timoschenko ;
    nbodies   1 ;
    node_distribution    c2_def ;
    damping_posdef 2.00E-05 2.00E-05 2.00E-04 3.00E-06 3.00E-06 2.00E-05;
begin timoschenko_input;
    filename ./data/NREL_5MW_st.txt ;
    set 4 1 ;
end timoschenko_input;
begin c2_def;          Definition of centerline (main_body coordinates)
    nsec 2;
    sec 1 0.0 0.0 0.0    0.0 ; x,y,z,twist
    sec 2 0.0 0.0 1.5    0.0 ;
end c2_def ;
end main_body;
;
begin main_body;
    name      hub2 ;
    copy_main_body hub1;
end main_body;
;
begin main_body;
    name      hub3 ;
    copy_main_body hub1 ;
end main_body;
;
begin main_body;
    name      blade1 ;
    type      timoschenko ;
    nbodies   9 ;
    node_distribution    c2_def;
; damping 3.5e-2 5.5e-4 5.0e-4 3.0e-4 0.5e-3 5.5e-3 ;
damping_posdef 1.16e-4 5.75e-5 6.1e-6 6.5e-4 5.1e-4 6.4e-4 ;
begin timoschenko_input ;
    filename ./data/NREL_5MW_st.txt ;
    set 5 1 ;          set subset
end timoschenko_input;
begin c2_def;          Definition of centerline (main_body coordinates)
    nsec 19 ;
sec 1 0.0000 0.0000 0.000 0.000 ; x.y.z. twist
sec 2 -0.0041 0.0010 1.367 -13.308 ;
sec 3 -0.1058 0.0250 4.100 -13.308 ;
sec 4 -0.2502 0.0592 6.833 -13.308 ;
sec 5 -0.4594 0.1087 10.250 -13.308 ;
sec 6 -0.5699 0.1157 14.350 -11.480 ;
sec 7 -0.5485 0.0983 18.450 -10.162 ;
sec 8 -0.5246 0.0832 22.550 -9.011 ;

```

```

sec 9 -0.4962 0.0679 26.650 -7.795 ;
sec 10 -0.4654 0.0534 30.750 -6.544 ; 50% blade radius
sec 11 -0.4358 0.0409 34.850 -5.361 ;
sec 12 -0.4059 0.0297 38.950 -4.188 ;
sec 13 -0.3757 0.0205 43.050 -3.125 ;
sec 14 -0.3452 0.0140 47.150 -2.319 ;
sec 15 -0.3146 0.0084 51.250 -1.526 ;
sec 16 -0.2891 0.0044 54.667 -0.863 ;
sec 17 -0.2607 0.0017 57.400 -0.370 ;
sec 18 -0.1774 0.0003 60.133 -0.106 ;
sec 19 -0.1201 0.0000 61.500 -0.000 ;
  end c2_def ;
  end main_body;
;
  begin main_body;
    name          blade2 ;
    copy_main_body blade1;
  end main_body;
;
  begin main_body;
    name          blade3 ;
    copy_main_body blade1 ;
  end main_body;
;-----
;
  begin orientation;
  begin base;
    body  monopile;
    inipos      0.0 0.0 20.0 ;          initial position of node 1
    body_eulerang 0.0 0.0 0.0;
  end base;
;
  begin relative;
    body1 monopile last;          indtil videre antages der internt i programmet at der
    ;                               altid kobles mellen sidste knude body1 og første
    ;                               knude body 2
    body2 tower 1;
    body2_eulerang 0.0 0.0 0.0;
  end relative;
;
  begin relative;
    body1 tower last;
    body2 towertop 1;
    body2_eulerang 0.0 0.0 0.0;
  end relative;
;
  begin relative;
    body1 towertop last;
    body2 shaft 1;
    body2_eulerang 90.0 0.0 0.0;
    body2_eulerang 5.0 0.0 0.0;    5 deg tilt angle
    ;body initial rotation velocity x.y.z.angle velocity[rad/s] (body 2 coordinates):
    body2_ini_rotvec_d1 0.0 0.0 -1.0 0.5 ;
  end relative;

```

```

;
begin relative;
  body1 shaft last;
  body2 hub1 1;
  body2_eulerang -90.0 0.0 0.0;
  body2_eulerang 0.0 180.0 0.0;
  body2_eulerang 2.5 0.0 0.0;      2.5deg cone angle
end relative;
;
begin relative;
  body1 shaft last;
  body2 hub2 1;
  body2_eulerang -90.0 0.0 0.0;
  body2_eulerang 0.0 60.0 0.0;
  body2_eulerang 2.5 0.0 0.0;      2.5deg cone angle
end relative;
;
begin relative;
  body1 shaft last;
  body2 hub3 1;
  body2_eulerang -90.0 0.0 0.0;
  body2_eulerang 0.0 -60.0 0.0;
  body2_eulerang 2.5 0.0 0.0;      2.5deg cone angle
end relative;
;
begin relative;
  body1 hub1 last;
  body2 blade1 1;
  body2_eulerang 0.0 0.0 0.0;
end relative;
;
begin relative;
  body1 hub2 last;
  body2 blade2 1;
  body2_eulerang 0.0 0.0 0.0;
end relative;
;
begin relative;
  body1 hub3 last;
  body2 blade3 1;
  body2_eulerang 0.0 0.0 0.0;
end relative;
;
end orientation;
;-----
begin constraint;
;
begin fix0; fixed to ground in translation and rotation of node 1
  body monopile;
end fix0;
;
begin fix1; fixed relative to other body in translation and rotation
  body1 monopile last;
  body2 tower 1;

```

```

end fix1;
;
begin fix1;
  body1 tower last ;
  body2 towertop 1;
end fix1;
;
begin bearing1;                                free bearing
  name shaft_rot;
  body1 towertop last;
  body2 shaft 1;
  bearing_vector 2 0.0 0.0 -1.0;                x=coo (0=global.1=body1.2=body2) vector in body2
  ;                                              coordinates where the free rotation is present
end bearing1;
;
begin fix1;
  body1 shaft last ;
  body2 hub1 1;
end fix1;
;
begin fix1;
  body1 shaft last ;
  body2 hub2 1;
end fix1;
;
begin fix1;
  body1 shaft last ;
  body2 hub3 1;
end fix1;
;
begin bearing2;
  name pitch1;
  body1 hub1 last;
  body2 blade1 1;
bearing_vector 2 0.0 0.0 -1.0;
end bearing2;
;
begin bearing2;
  name pitch2;
  body1 hub2 last;
  body2 blade2 1;
bearing_vector 2 0.0 0.0 -1.0;
end bearing2;
;
begin bearing2;
  name pitch3;
  body1 hub3 last;
  body2 blade3 1;
bearing_vector 2 0.0 0.0 -1.0;
end bearing2;
end constraint;
;
end new_htc_structure;
;-----

```

```

begin wind ;
  density          1.25;
  wsp              8 ;
  horizontal_input 1;
  windfield_rotations 0.0 0.0 0.0 ;    yaw, tilt, rotation
  center_pos0      0.0 0.0 -90.00;    hub_height
  shear_format     3 0.12;
  turb_format      1 ;    0=none, 1=mann,2=flex
  tower_shadow_method 1;
  tint             0.06 ;
  scale_time_start 200;
  wind_ramp_factor 0.0 200 0.5 1.0 ;
;-----
begin tower_shadow_potential;
  tower_offset 0.0;
  nsec 2;
  radius      0.0 2.10;
  radius      -68.10 1.15;
end tower_shadow_potential;
;-----
; This next part is only to be include in case of wake effects being studied
begin wakes;
  nsource 35;
  source_pos 2548 -2900 -90 ;
  source_pos 2123 -2417 -90 ;
  source_pos 1706 -1942 -90 ;
  source_pos 1281 -1458 -90 ;
  source_pos 857 975 -90 ; WT5
  source_pos 432 491 -90 ; WT6
  source_pos -425 -484 -90 ; WT8
  source_pos -850 -968 -90 ; WT9
  source_pos -1267 1458 -90 ;
  source_pos -1700 1935 -90 ;
  source_pos -2125 2419 -90 ;
  source_pos 3556 -2533 -90 ;
  source_pos 3131 -2049 -90 ;
  source_pos 2706 -1565 -90 ;
  source_pos 2281 1081 -90 ; WT16
  source_pos 1602 308 -90 ; WT17
  source_pos 1176 -176 -90 ; WT18
  source_pos 751 -660 -90 ; WT19
  source_pos 326 -1144 -90 ; WT20
  source_pos -99 -1627 -90 ; WT21
  source_pos 3915 -1427 -90 ;
  source_pos 3486 -943 -90 ;
  source_pos 3062 -455 -90 ;
  source_pos 2405 -292 -90 ; WT25
  source_pos 1927 -836 -90 ; WT26
  source_pos 1502 -1319 -90 ; WT27
  source_pos 1077 -1803 -90 ; WT28
  source_pos 652 -2287 -90 ; WT29
  source_pos 4235 -283 -90 ;
  source_pos 3813 205 -90 ;
  source_pos 3163 944 -90 ;

```

```

source_pos 2679 1495 -90 ;
source_pos 2254 1979 -90 ;
source_pos 1829 2463 -90 ;
source_pos 1404 2947 -90 ;
op_data      1.4252392 2 ; 1.8 -23.1 ;1.87 0.0 rad/sec, pitch [grader] opstrøms;
ble_parameters 0.10 0.008 0;
begin mann_meanderturb ;
    create_turb_parameters 33.6 1 3.7 508 0.0 ;      L, alfaeps,gamma,seed, highfrq compensation
    filename_v  ./free_sector_monopile/wake-meander/wake_meand_turb_wsp8_s508_t1800v.bin ;
    filename_w  ./free_sector_monopile/wake-meander/wake_meand_turb_wsp8_s508_t1800w.bin ;
    box_dim_u   16384 1.7578125 ;
    box_dim_v   32 90 ;
    box_dim_w   32 90 ;
end mann_meanderturb;
;
begin mann_microturb ;
    create_turb_parameters 8.0 1.0 1.0 508 1.0 ;      L, alfaeps,gamma,seed, highfrq compensation
    filename_u  ./free_sector_monopile/wake-micro/wake_turb_wsp8_s508_t1800u.bin ;
    filename_v  ./free_sector_monopile/wake-micro/wake_turb_wsp8_s508_t1800v.bin ;
    filename_w  ./free_sector_monopile/wake-micro/wake_turb_wsp8_s508_t1800w.bin ;
    box_dim_u   128 1.0 ;
    box_dim_v   128 1.0 ;
    box_dim_w   128 1.0 ;
end mann_microturb;
end wakes;
;-----
begin mann;
    create_turb_parameters 33.6 1 3.7 508 1.0 ;      L, alfaeps,gamma,seed, highfrq compensation
    filename_u  ./free_sector_monopile/turb/turb_wsp8_s508_t1800u.bin ;
    filename_v  ./free_sector_monopile/turb/turb_wsp8_s508_t1800v.bin ;
    filename_w  ./free_sector_monopile/turb/turb_wsp8_s508_t1800w.bin ;
    box_dim_u   16384 1.7578125 ;
    box_dim_v   32 3.75;
    box_dim_w   32 3.75;
end mann;
end wind;;
begin aero ;
    nblades 3;
    hub_vec shaft -3 ;      rotor rotation vector (normally shaft component directed from
    ;                          pressure to suction side)
    link 1 mbdy_c2_def blade1;
    link 2 mbdy_c2_def blade2;
    link 3 mbdy_c2_def blade3;
    ae_filename      ./data/NREL_5MW_ae.txt;
    pc_filename      ./data/NREL_5MW_pc.txt;
    induction_method 1 ;      0=none, 1=normal
    aerocalc_method  1 ;      0=ingen aerodynamic, 1=med aerodynamic
    arosections      30 ;
    ae_sets          1 1 1;
    tiploss_method   1 ;      0=none, 1=prandtl
    dynstall_method  2 ;      0=none, 1=stig øye method,2=mhh method
end aero ;
;
;-----

```

```

begin hydro;
  begin water_properties;
    rho 1027 ; kg/m^3
    gravity 9.81 ; m/s^2
    mwl 0.0 ;
    mudlevel 20.0 ;
    water_kinematics_dll ./wkin_dll.dll ./htc_hydro/reg_airy_h6_t10.inp ;
end water_properties;
;
begin hydro_element;
  body_name monopile ;
  hydrosections uniform 50 ; distribution of hydro calculation points from sec 1 to nsec
  nsec 2;
  sec 0.0 1.0 1.0 28.27 28.27 6.0 ; nr z Cm Cd V Vr width
  sec 30.0 1.0 1.0 28.27 28.27 6.0 ; nr z Cm Cd V Vr width
end hydro_element;
end hydro;
;
;-----
begin dll;
  begin hawc_dll;
    filename ./control/bladed2hawc.dll ;
    dll_subroutine regulation ;
    arraysizes 15 15 ;
    deltat 0.02;
    begin output;
      general time ;
      constraint bearing2 pitch1 1; angle and angle velocity written to dll
      constraint bearing2 pitch2 1; angle and angle velocity written to dll
      constraint bearing2 pitch3 1; angle and angle velocity written to dll
      constraint bearing2 shaft_rot 1; angle and angle velocity written to dll (slow speed shaft)
      wind free_wind 1 0.0 0.0 -90.55; local wind at fixed position: coo
      general constant 97.0 ; generator exchange ratio
    end output;
;
  begin actions;
    body moment_int shaft 1 3 towertop 2 ;
  end actions;
end hawc_dll;
;
begin hawc_dll;
  filename ./control/pitchservo_pos.dll ;
  dll_subroutine servo ;
  arraysizes 15 15 ;
  deltat 0.02 ;
  begin output;
    general time ;
    dll invec 1 2;
    dll invec 1 3;
    dll invec 1 4;
    constraint bearing2 pitch1 1; angle and angle velocity written to dll
    constraint bearing2 pitch2 1; angle and angle velocity written to dll
    constraint bearing2 pitch3 1; angle and angle velocity written to dll
  end output;

```

```

;
  begin actions;
    body bearing_angle pitch1;
    body bearing_angle pitch2;
    body bearing_angle pitch3;
  end actions;
end hawc_dll;
;
begin hawc_dll;
  filename ./control/damper.dll ;
  dll_subroutine damp ;
  arraysizes 15 15 ;
  begin output;
    general time ;
    general constant 5.0;
    general constant 10.0;
    general constant -1.0E1 ;
    mbdy state vel towertop 1 1.0 tower;
  end output;
;
  begin actions;
    mbdy force_ext towertop 2 1 towertop;
    mbdy force_ext towertop 2 2 towertop;
  end actions;
end hawc_dll;
end dll;
;
;-----
;
begin output;
  filename ./res/oc3_monopile_phase_1 ;
; time 390.0 450.0 ;
  buffer 1 ;
  general time;
  data_format hawc_binary;
;
  constraint bearing1 shaft_rot 2; angle and angle velocity
  constraint bearing2 pitch1 5;   angle and angle velocity
  constraint bearing2 pitch2 5;   angle and angle velocity
  constraint bearing2 pitch3 5;   angle and angle velocity
  aero omega ;
  aero torque;
  aero power;
  aero thrust;
  wind free_wind 1 0.0 0.0 -90.0; local wind at fixed position: coo
  hydro water_surface 0.0 0.0 ;      x,y gl. pos
  mbdy momentvec towertop 1 2 towertop # yaw bearing ;
  mbdy forcevec towertop 1 2 towertop # yaw bearing ;
  mbdy momentvec shaft 4 1 shaft # main bearing ;
  mbdy momentvec blade1 3 1 blade1 # blade 1 root ;
  mbdy momentvec blade1 10 1 local # blade 1 50% local e coo ;
  mbdy momentvec hub1 1 2 hub1 # blade 1 root ;
  mbdy momentvec hub2 1 2 hub2 # blade 2 root ;
  mbdy momentvec hub3 1 2 hub3 # blade 3 root ;

```

```

mbdy state pos towertop 1 1.0 global # tower top flange position ;
mbdy state pos tower 1 0.0 global # tower MSL position ;
mbdy state pos blade1 18 1.0 blade1 # blade 1 tip pos ;
mbdy state pos blade2 18 1.0 blade2 # blade 2 tip pos ;
mbdy state pos blade3 18 1.0 blade3 # blade 3 tip pos ;
mbdy state pos blade1 18 1.0 global # blade 1 tip pos ;
aero windspeed 3 1 1 63.0; wind seen from the blade:
; coo(1=local ae,2=blade,3=global,4=rotor polar),
aero windspeed 3 1 2 63.0;
aero windspeed 3 1 3 63.0;
aero alfa 1 45.0;
aero alfa 2 45.0;
aero alfa 3 45.0;
mbdy momentvec towertop 1 1 tower # tower top -1: below top mass ;
mbdy forcevec towertop 1 1 tower # tower top -1: below top mass ;
mbdy momentvec tower 1 1 tower # tower MSL ;
mbdy forcevec tower 1 1 tower # tower MSL ;

; mbdy statevec_new mbdyname center coo elastic/absolute r sign xy_vector:
mbdy statevec_new blade1 c2def blade1 elastic 88.0 1.d0 0.0 0.0
mbdy statevec_new blade1 default blade1 elastic 88.0 1.d0 0.0 0.0 ;
mbdy statevec_new blade1 c2def blade1 absolute 88.0 1.d0 0.0 0.0 ;
mbdy statevec_new blade1 default blade1 absolute 88.0 1.d0 0.0 0.0 ;
mbdy statevec_new blade1 default global absolute 88.0 1.d0 0.0 0.0 ;

; mbdy forcemomentvec_interp mbdy_name center coo_mbdy curved_distance_from_orig sign
mbdy forcemomentvec_interp blade1 default blade1 5 1.0 # blade1 R= 5 ;
mbdy forcemomentvec_interp blade1 default blade1 55 1.0 # blade1 R=55 ;
mbdy forcemomentvec_interp blade1 c2def local_aero 35 1.0 # blade1 R=35 ;
mbdy forcemomentvec_interp blade1 c2def local_aero 60 1.0 # blade1 R=60 ;
mbdy forcemomentvec_interp blade1 c2def local_element 50 1.0 # blade1 R=50 ;
; an example where the forces and moments are extracted at the c2def instead of the actual node:
mbdy forcemomentvec_interp blade1 c2def blade1 5 1.0 # blade1 R= 5 ; ()
;

dll outvec 1 1 # time;
dll outvec 1 2 # pitch angle 1;
dll outvec 1 3 # pitch vel 1;
dll outvec 1 4 # pitch angle 2;
dll outvec 1 5 # pitch vel 2;
dll outvec 1 6 # pitch angle 3;
dll outvec 1 7 # pitch vel 3;
dll outvec 1 8 # gen. azi slow;
dll outvec 1 9 # gen. speed slow;
dll outvec 1 10 # free wind x;
dll outvec 1 11 # free wind y;
dll outvec 1 12 # free wind z;
dll outvec 1 13 # gear ratio;
dll inpvec 1 1 # Mgen slow;
dll inpvec 1 2 # pitchref 1;
dll inpvec 1 3 # pitchref 2;
dll inpvec 1 4 # pitchref 3;
dll inpvec 1 7 # F;
dll inpvec 1 8 # Mechanical power generator [kW];
dll inpvec 1 10 # Pitch rate [rad/s];

```

```
dll invec 2 1 # pitch 1;
dll invec 2 2 # pitch 2;
dll invec 2 3 # pitch 3;
dll outvec 2 1 # time;
dll outvec 2 2 # pitchref 1;
dll outvec 2 3 # pitchref 2;
dll outvec 2 4 # pitchref 3;
dll outvec 2 5 # pitch angle 1;
dll outvec 2 6 # pitch speed 1;
dll outvec 2 7 # pitch angle 2;
dll outvec 2 8 # pitch speed 2;
dll outvec 2 9 # pitch angle 3;
dll outvec 2 10 # pitch speed 3;
end output;
;
exit;
```

B Code Version Data

Version information:

Version name

Date Resp Info

!-----
global%version='HAWC2MB 1.0'
20.04.2006 TJUL Version system started. Changes in so_dyn_stall model performed.
24.04.2006 TJUL/ANMH Bearing3 in topology - slight modification still needed, but now mhha
needs a version
global%version='HAWC2MB 1.1'
25.04.2006 TJUL mhha laptop in MAC check, integer overflow negletec in compiler
settings
global%version='HAWC2MB 1.1work'
26.04.2006 TJUL tjul stationairy pc in MAC check
New check regarding thicknesses in aeodynamic files
28.04.2006 TJUL ktho stationairy pc in MAC check
global%version='HAWC2MB 1.2'
28.04.2006 TJUL Radius non-dim in structural _st input data and aerodynamic _ae data
global%version='HAWC2MB 1.3'
01.05.2006 TJUL Extra check in structural files reading procedures
Tab characters can now be used in htc files and other input files
Check that c2_def structure length larger than eps
global%version='HAWC2MB 1.4'
02.05.2006 TJUL New check in hawc_file output that time_stop>time_start
Topologi_timoschenko.f90 updated related to changes in version 1.3
global%version='HAWC2MB 1.5'
03.05.2006 TJUL ktho laptop in MAC check
Get_state_rot function in body.f90
New mbdy state_rot output command in topologi_mainbody_output
Rotation velocity and acceleration in aerodynamic blade section
variables
MACQ/TJUL Dynamic_stall_mhh included
global%version='HAWC2MB 1.6'
04.05.2006 TJUL Extension of bladelink criteria for execution stop
global%version='HAWC2MB 1.7'
09.05.2006 TJUL New error message in windturb_mann.f90
New error messages regarding matrix not definite problems
New MAC checks (Niels Kjølstad + students)
global%version='HAWC2MB 1.8'
09.05.2006 TJUL New MAC check
global%version='HAWC2MB 1.9'
16.05.2006 TJUL New MAC check
global%version='HAWC2MB 2.0'
18.05.2006 TJUL New MAC check
global%version='HAWC2MB 2.1'
19.05.2006 TJUL Error messages corrected in mbdy state_rot command
MHHA/TJUL New MAC check procedure (loop over all adresses instead of only one)
global%version='HAWC2MB 2.2'
22.05.2006 TJUL New ignore function in body actions
30.05.2006 TJUL Old MAC check procedure reimplemented since troubles occured with the
new version
global%version='HAWC2MB 2.3'
30.05.2006 TJUL Replacement of procedure that calculates euler parameters based on

transformation matrix
(only important for cases with eulerp output used)
global%version='HAWC2MB 2.4'
31.05.2006 TJUL General cleanup in multibodyproto.f90 file (simple generator model excluded, now tmp_gen_speed output command is excluded)
01.06.2006 TJUL New MAC checks
global%version='HAWC2MB 2.5'
04.06.2006 TJUL External Licence manager DLL used. Avoids new versions of the HAWC2 code to be build at every new MAC number and also works when the computer is not connected to a LAN
global%version='HAWC2MB 2.6'
13.06.2006 TJUL Newmark variables reorganized
Hydrodynamic loads cut-in at 2secs, as for the aero loads. To reduce initial transients
New acceptance criteria from License manager
New input check in topologi_mainbody
Order of radius of gyration input shifted for the new_htc_structure input. Now: 1st column (Rix) is the one affected if mass center position changes on the chord line
global%version='HAWC2MB 2.7'
23.06.2006 ANMH Normalisation of vectors in utils funtions get_two_plane_vectors. Used for better accuracy in bearing1 and bearing2 definitions
global%version='HAWC2MB 2.8'
17.07.2006 TJUL/FRBA Correction of bug in get_ae_data procedure in aeroload_calcfoces unit. Profile sets higher than one is now also usable.
global%version='HAWC2MB 2.9'
17.07.2006 TJUL Gravity loads cut-in at 0.5secs, same method as for the aero loads. To reduce initial transients
Harmonic2 function in general output (time limitid harmonic function)
global%version='HAWC2MB 3.0'
24.07.2006 TJUL topologi_mainbody_actions module added. New features to the actions list.
global%version='HAWC2MB 3.1'
26.07.2006 TJUL Mann turbulence is reused if simulation time is longer than included in turbulence box
global%version='HAWC2MB 3.2'
28.07.2006 TJUL Correction of bug in aerodynamic moment integration procedure (only related to aerodynamic file output)
31.07.2006 TJUL Change of error message criteria regarding allowable number of bodies within a mainbody (<n elements)
01.08.2006 TJUL Correction of bug in dynstall_mhh model so no division by zero occurs when a zerolift profile is used.
01.08.2006 ANMH Correction of bug related to torsion of blade in the blade linker
global%version='HAWC2MB 3.3'
04.08.2006 TJUL Check applied on exp expressions in dynamic stall mhh model to avoid underflow errors
global%version='HAWC2MB 3.4'
09.08.2006 TJUL New check applied in mann turbulence unit to avoid array out of bounds during bizar startup transients

11.08.2006 TJUL Correction of exp check in dynstall_mhh model just created in version 3.2
global%version='HAWC2MB 3.5'

28.08.2006 TJUL Generator_rotation sensor setup for old_htc_structure format - replaces the older tmp_gen_speed sensor. Updates in hawcstructure.f90 and body_output.f90
New error message in body_output
Improvement of general command reader in genout_tools in order to accept tabulator spacings
General shine up of aerodynamic calculations regarding induction and tiploss calculations rechecked against IEA rev 3 calculations
Number of radial point in the induction calculation is default set to the name number as number of aero sections. Previous default of 30 stations
Linear interpolation in aeroload_tools updated so no division by zero occurs when x0=x1, used in cases where extrapolation is not wanted

29.08.2006 ANMH/TJUL Fix1 constraints updated in topologi_constraints_fix1.f90 and hawcstructure.f90. Ensures e.g. that constraint properties are identical for blades. Ensures that blades performs identically.
global%version='HAWC2MB 3.6'

14.09.2006 TJUL New acceptance criteria from license manager
global%version='HAWC2MB 3.7'

15.09.2006 TJUL New general load linker that replaces bladelink.f90 and wavelink.f90
ANMH/TJUL Pitchsensors (bearing sensor) updated during iterations too. Especially important for DLL controllers
global%version='HAWC2MB 3.8'

06.10.2006 TJUL Correction of bug related to aero int_force and int_moment sensors
Correction of bug in DLL actions. On nodes different from nr. 1, in- and external forces and moments were placed on the node 1 number lower.
TJUL Pitch sensor modified. Now pitch velocity is calculated based on numerical differentiation of calculated angle. Should be less sensitive to solver inaccuracies.
TJUL In output of bearing sensor new options are added. (-180:180 deg output etc.)
ANMH Correction of bug in loadlinker. It turned out that loadfunction were only correct if an even number of calculation points were used (aero or hydro). Now OK also for odd numbers
global%version='HAWC2MB 3.9'

06.10.2006 TJUL Soil spring module added (soil stuff from hydro module removed)
global%version='HAWC2MB 4.0'

02.11.2006 TJUL Extra output commands in aero output_at
global%version='HAWC2MB 4.1'

02.11.2006 ANMH/TJUL Replacement of added stiffness method for soil springs. Much better and faster than previous. Still not perfect.
10.11.2006 ANMH/TJUL Update of bearing3. Now it is general.
10.11.2006 TJUL Output variables rearranged. Only command included in bearing outputs
10.11.2006 TJUL Topologi input modified so many bases are allowable.
15.11.2006 ANMH/TJUL Files synchronized with Anders. Slight update in dll_calls,dll_types and windturb_mann.
global%version='HAWC2MB 4.2'

16.11.2006 TJUL Rearrangement of output/action sensor allocation. Reduces .exe size from 23MB to 2.3MB
TJUL/HAMA Correction of sensors induc and windspeed in output_at aero. They were previously in a wrong coordinate system when written to output_at.

global%version='HAWC2MB 4.3'

17.11.2006 TJUL New fix3 constraint. Locks a node to ground in a given rotation direction.

23.11.2006 ANMH Update of loadlinker with respect to procedures for numerical update of stiffness, damping and mass terms. Improves solutions of soil spring systems significant.

27.11.2006 TJUL Same procedure used for the hydrodynamic part => faster convergence.

global%version='HAWC2MB 4.4'

27.11.2006 TJUL Bug fixed related to input for action sensor: mbdy moment_int

04.12.2006 ANMH index+7 -> index+6 in body_get_state_rot subroutine. Affects orientation of all local load elements in load linker.

global%version='HAWC2MB 4.5'

06.12.2006 TJUL New sensors in aero module.

Out of bounds bug in aero output_at corrected

07.12.2006 ANMH Files used in topologi_tools are closed after use.

12.12.2006 TJUL In make output command, outputs are bypassed if global time > output stoptime

13.12.2006 TJUL In mann turbulence a new command (dont_scale) is made.

21.12.2006 TJUL Update of HAWC_mann module. Important only if turbulence outside box is used.

04.01.2007 TJUL omega vector for aerodynamic module in rotor reference coordinates. In aero files only rotation speed around y-axis is used. Eliminates influence from e.g. pitch velocity

04.01.2007 MHHA/TJUL New optional relaxation parameter for solver. Extra command in simulation input.

global%version='HAWC2MB 4.6'

12.01.2007 ANMH Change of sign in forcedll.f90. Important only if an external force dll as coupled springs are used. Not important for hawc_dll

global%version='HAWC2MB 4.7'

06.02.2007 ANMH/TJUL Update of code structure, multibodyproto split into several subroutines

New logical variables related to simulation_input

New state_at in mbdy output

global%version='HAWC2MB 4.8'

08.02.2007 TJUL mbdy actions force/moment commands updated with sign possibility on force component

global%version='HAWC2MB 4.9'

12.02.2007 TJUL new error message in turbulence input reader

19.02.2007 TJUL New potential flow tower shadow model where source is linked to tower motion

global%version='HAWC2MB 5.0'

26.02.2007 TJUL New mbdy state_rot output option: orientation in euler angles defined through the rotation order xyz

global%version='HAWC2MB 5.1'

27.02.2007 TJUL Correction of method used to calculate mbdy state_rot rotation in general

New mbdy state_rot output option: orientation in euler angles defined through the rotation order yxz

Small adjustments in DLL_output to avoid array out of bounds when long mbdy names are used

global%version='HAWC2MB 5.2'

02.03.2007 ANMH/TJUL Bug fixed related to continue_on_no_convergence criteria

TJUL HAWC2MB version echoed to screen before input is read.

TJUL New licence manager compiler option

global%version='HAWC2MB 5.3'

13.03.2007 ANMH/TJUL Bug fixed related to bearing3. Somehow the coupling nodes was not defined since version 4.0 It affects the transfer of loads from bearing3 and further dow the tower.
global%version='HAWC2MB 5.4'

21.03.2007 TJUL/ANMH Eigenfrequency analysis feature added. Performs analysis on every individual body
TJUL Some pointer nullify's are changed to deallocate(pointer).
global%version='HAWC2MB 5.4'

21.03.2007 TJUL/ANMH Eigenfrequency analysis feature added. Performs analysis on every individual body
TJUL Some pointer nullify's are changed to deallocate(pointer).
global%version='HAWC2MB 5.5'

29.03.2007 TJUL Small change in constraint bearing2 action input. Now only 4 parameters nessecairy as was allways the idea.
global%version='HAWC2MB 5.6'

10.04.2007 TJUL Bug fix related to number of output sensors in DLL output
ANMH Change in external force module force_dll.f90. Update sequence of affected body changed.
TJUL body_update_T is called in the end of post_init in order to allow for added stiffness, damping etc. by the rest of the initialization subroutines.
global%version='HAWC2MB 5.7'

16.04.2007 TJUL Small update of continue on no convergence
global%version='HAWC2MB 5.8'

18.04.2007 TJUL Mann turbulence files is closed after every buffer read. To allow several simulations acces to the same turbulence files.
New initial buffer read so out of x-bounds errors are avoided.
Uses periodicity of turbulence boxes.
In principal this allows for infinitely large simulations.
global%version='HAWC2MB 5.9'

23.04.2007 TJUL Opening of mann turbulence boxed with loops and waits so several simulations can acces the same turbulence.
26.04.2007 TJUL Only option in mbody output, wind output, hydro output
SO dynamic stall input parameters put in as default. No need for parameter input if not changed.
23.05.2007 TJUL Check that turbulence scale_time_start is less the total simulation length
Correction of bug related to "only" option for output for main_body, wind and hydro output commands
global%version='HAWC2MB 6.0'

01.06.2007 TJUL New error check that animation can be written to. Error message if not.
08.06.2007 TJUL New possibility of continuing read in masterfile in a new file with the command:'continue_in_file'. Infinite number of level can be made.
Filename also written to logfile when line number is written.
15.06.2007 TJUL Logfile_name command option in simulation_input. Enables file written logfiles. Error messages more clear with *** ERROR *** as key word
08.08.2007 TJUL Aerodynamic drag forces on structures enables with the new module aerodrag.
global%version='HAWC2MB 6.1'

03.09.2007 TJUL Corrections made in continue_in_file option. End of file check removed - replaced with exit command.
03.09.2007 ANMH New unitnumber used when turbulence files are reopened. To avoid unit mismatch especiall
05.09.2007 TJUL Bug fixed in hydroload module. Only important when more than one hydro

element are used.

06.09.2007 TJUL Bug fixed in hydroload module. Important if hydroelements have different coo than global.

07.09.2007 TJUL Bug fixed in hydroload module. Important if relative z_distances has been used as hydro element input

global%version='HAWC2MB 6.2'

20.09.2007 PBJA/TJUL Dynamic stall module that combines the mhh Beddoes stall model with the MACflap model. Coded by PBJA, implemented by TJUL.

global%version='HAWC2MB 6.3'

10.10.2007 TJUL New general output command "general stairs" for a series of step functions.

global%version='HAWC2MB 6.4'

29.10.2007 TJUL Some files synchronized with HAWC2aero regarding !IFDEF compiler directives

12.11.2007 TJUL Torque and power output sensor in aero module modified to give correct results also with use of hub extenders

27.11.2007 TJUL Wake meandering model implemented, rearrangement of aero files to avoid compiler linker (circulation) errors

29.11.2007 ANMH Eigenvalue solver for complete turbine at standstill, initialisation of aerodrag element number!

global%version='HAWC2MB 6.5'

04.01.2008 TJUL User defined turbulence scaling implemented. Similar in principle to user defined shear.

17.01.2008 TJUL Bearing3 omegaS action command implemented to enable rotor speed control directly from external DLL

04.02.2008 ANMH Bouyancy forces calculated based on external pressures

TJUL Prestress constraint fix4

DLL call to external wake kinematics dll changed. E.g. dynamic pressure added

global%version='HAWC2MB 6.6'

04.02.2008 TJUL bearing4. Cardan shaft constraint. Locked in relative translation. Locked in rotation around one vector

08.02.2008 TJUL Bug fixed in turbulence module affecting version 5.5.

08.02.2008 TJUL Bug fixed regarding turbulence scaling factors. In version 6.5 the turbulence was excluded for normal use - corrected.

08.02.2008 TJUL Previous .dat file deleted when hawc_binary output files are written.

11.02.2008 TJUL In mann and flex turbulence module: std scaling factors default to $v=0.8$ $u=1.0$ $w=0.5$

11.02.2008 TJUL Bug fixed regarding IEC-gust EWS

13.02.2008 TJUL New Auto distribution of hydrodynamic calculation points possible

13.02.2008 TJUL/ANMH Bug fixed regarding hydrodynamic boyancy. Axial force on conical members changed from distributed forces to constant force contributions instead (to decrease sensitivity to number of hydro points)

F function only on external kinematics in hydro module.

Dynamic pressure contribution include. Also in wkin_dll calling format.

Coordinates in wkin_dll call changed from global to local hydro coo (origo in 0,0,MSL Z-dir vertical upwards, X-dir in wave direction)

Change in hydro output command "fm" and "fd"

15.02.2008 TJUL trim commands inserted in reading of master input "begin" and "exit" commands.

15.02.2008 TJUL Bug fixed regarding output of "free_wind_hor" command.

19.02.2008 TJUL S.O. dynamic stall parameters included as default

global%version='HAWC2MB 6.7'

26.02.2008 TJUL Bug fixed regarding mbdy action command with "local" coordinates

27.02.2008 TJUL Extra error messages for errors during aero read routines

29.02.2008 TJUL Small modifications in eigenvalue solver so large eigenvalue problem can be solved without very large stack size

06.03.2008 TJUL Dynamic pressure on conical sections also in hydroload

09.03.2008 TJUL Wordlength increased to 100 chars in general input reading.

11.03.2008 TJUL Error handling for infinity cases in hawc_binary output

11.03.2008 ANMH/TJUL Dynamic pressure on conical hydro sections

11.03.2008 TJUL Update of mann turb reading routines for boxes where $N_y < N_z$

11.03.2008 TJUL Small modifications in the wake module for robustness

13.03.2008 TJUL Update of mann turb reading so buffer is updated also when requested point is before buffer start pos (especially important for wake sim. with several wake sources)

global%version='HAWC2MB 6.8'

14.03.2008 TJUL Update of tower shadow pot2 and jet2 models, so they can handle multiple sources.

global%version='HAWC2MB 6.9'

21.03.2008 TJUL Increase of maxloops in mann turbulence reading.

global%version='HAWC2MB 7.0'

09.04.2008 TJUL New check in license_manager

global%version='HAWC2MB 7.1'

21.05.2008 TJUL Bug fixed in command line interpreter (if too many command words were present)

26.05.2008 ANMH/TJUL Concentrated masses option in main_body (no coriolis effects etc. so far)

11.06.2008 TJUL Extra acceleration sensor including gravity

13.06.2008 TJUL Minimum values of rotational speed and free wind speed in the induction module.

global%version='HAWC2MB 7.2'

15.06.2008 TJUL F startup function and relative motion in aerodrag included extra check on shear power law expression in wind module to avoid NAN's

global%version='HAWC2MB 7.3'

22.07.2008 TJUL Concentrated mass in modal calculation

01.08.2008 TJUL Bug in calculation procedure of aerodynamic torque and power corrected. Bug in tower shadow pot2 and jet2 models corrected. Important only if rotation of tower legs were present.

global%version='HAWC2MB 7.4'

05.08.2008 ANMH Change in output of forces/moments in general. More correct when long elements are used.

Distributed external loads, inertial loads included on top of elastic part. Previously only elastic part used.

ANMH Hydrodynamic axial drag possible

Bearing 2 updated to allow for +-180deg rotation

06.08.2008 TJUL/HAMA Update of tower shadow 2 models. Factors multiplied instead of deficits added. Better when several tower shadow sources are used.

global%version='HAWC2MB 7.5'

08.08.2008 TJUL Correction of matrix conditioning during eigenvalue calculations. Version 7.3 and 7.4 was not correct regarding this!

global%version='HAWC2MB 7.6'

24.09.2008 TJUL Bug fixed in tower pot2 model.

Old LIB files for old HAWC input format read, removed from project

global%version='HAWC2MB 7.7'

01.10.2008 TJUL Extra logfile output regarding load linking.

NEED EXTRA ATTENTION -NOT COMPLETELY FIXED YET- WORK ONLY when body structure is defined along the body z coordinates

Bug fixed in aerodrag module (important if aerodrag is linked to a structure where local element and body coo doesn't coincide)

08.10.2008 PBJA/TJUL Mann turbulence generator DLL call added

08.10.2008 TJUL Warning written if a comma "," is written within a command line
global%version='HAWC2MB 7.8'

10.10.2008 TJUL Animation files for structure eigenvalues calc placed in same directory as eigenvalue list

10.10.2008 TJUL Limitations in orientation_relative removed. Any coupling node can now be chosen. Eigenvalue solver however not updated for this option yet.
global%version='HAWC2MB 7.9'

17.11.2008 TJUL Extra subroutines in normal DLL hawc_dll call. Subroutines added: init and message.

18.11.2008 TJUL Directories needed are now automatically created if they do not exist

19.11.2008 TJUL A status sensor is added in the general outputs.

19.11.2008 TJUL Solvetype is default set to 1=newmark

20.11.2008 TJUL In dynamic wake model, downstream distance without offset, makes better agreement with measurements and FIDAP

20.11.2008 TJUL In Dynamic Wake Model: Possibility of writing file with Ct and Cq data

21.11.2008 TJUL Change in force DLL module. Now bodyname refers to a main_body

21.11.2008 ANMH Update of initial hydrodynamic loads for added mass/stiffness calculation

21.11.2008 ANMH Update of loadlinker and solver wrt. calculation of added mass/stiffness/damping.

Asymmetric solver implemented - to improved convergence for hydrodyn. problems(not active in version 7.9)

global%version='HAWC2MB 8.0'

28.11.2008 TJUL In wake meander model. User calculated deficits can be read.

02.12.2008 TJUL Change in error message of tower shadow jet and jet2 model - when points requested is inside tower.

02.12.2008 TJUL General output sensor "status" is set to -1 in last time step.

03.12.2008 PBJA Near wake induction model implemented

03.12.2008 PBJA Possibility of exporting wind field including shear, tower shadow, wake etc.

03.12.2008 PBJA In normal induction model. First order time filter on induced velocities replaced with two indicial functions - modified filter approach. Better agreement with NASA AIMES experiment.

18.12.2008 ANMH/TJUL Bug correction of concentrated mass indexing in eigenvalue calculation. Important (only) if mass is connected to body node 1

18.12.2008 ANMH/TJUL Possibility of calculating structural natural frequencies without damping contribution. More robust calculation

global%version='HAWC2MB 8.1'

09.01.2009 TJUL In mann model. Auto generation of missing turbulence in more general form.

09.01.2009 TJUL In hydro module. Currents included, wave direction included.

16.01.2009 ANMH Assymmetric solver option, which decreases number of iterations for offshore simulations considerable. Newmark-symmetric option

global%version='HAWC2MB 8.2'

20.01.2009 TJUL Bug found in version 8.0 regarding Dynamic wake meander model. Input deficits to Aislie model with wrong value in last radius point.

global%version='HAWC2MB 8.3'

21.01.2009 TJUL Rearrangement of write procedure for final deficit in Dynamic wake meander model. Array-out-of-bound could occur in special cases

02.02.2009 TJUL New twist angle sensor in output_at aero commands

27.02.2009 TJUL Small correction of tip loss model. sin(phi) instead of phi.

11.03.2009 ANMH/TJUL Update of modal solver. Now also usable for floating systems.

18.03.2009 HAMA/TJUL Update of Dynamic wake meander model. Deficit are now more narrow than previous. Default parameters k1,k2 are changed.

05.05.2009 ANMH Bug fix in mass matrix and orthogonality of local orientation matrices. Important (only) with prebend and mass center offset from elastic axis.

05.05.2009 TJUL Small updates regarding mbody commands instead/supplementary to old body commands in new_htc_structure inputs

05.05.2009 ANMH Extra parameter in hydro element regarding linear axial drag contribution.

06.05.2009 TJUL More residual information outputted in case of no convergence
global%version='HAWC2MB 8.4'

11.05.2009 TJUL New input check on number of mann box points. power of 2 criteria. Mode shape animation files written in appropriate directories.

12.05.2009 ANMH Initialization of timosection properties

13.05.2009 TJUL No double eigenvalue sets are written in table of structural frequencies
global%version='HAWC2MB 8.5'

14.05.2009 ANMH Bug corrected in eigenvalue solver related to version 8.3 and 8.4
global%version='HAWC2MB 8.6'

08.07.2009 TJUL Bug fix related to mann turbulence look-up indexes for points just outside the turbulence box.

08.07.2009 TJUL New updates of DWM wake model. New ainslie-15.exe and modification of default parameters.
global%version='HAWC2MB 8.7'

24.08.2009 TJUL In main_body input limitation of 4 c2def points lower to 2. If less than 4 points, linear interpolation is used.

30.08.2009 TJUL Element coordinates can now be used without limitations. Local coordinate system written in beam_output_file.

04.09.2009 ANMH Loadlinker updated so arbitrary body coordinations systems can be used. Linker now follows local curved beam direction

04.09.2009 TJUL Positive definite damping model originally formulated by Morten H. Hansen is included in HAWC2. Makes it possible to utilize the shear center position away from the elastic axis without problems with damping model.

05.09.2009 TJUL Small bugfix related to aerodrag module.
global%version='HAWC2MB 8.8'

08.10.2009 TJUL Bug related to damping model changes in version 8.7 corrected.
global%version='HAWC2MB 8.9'

09.10.2009 ANMH/TJUL New structure output: Structure_inertia_file_name
global%version='HAWC2MB 9.0'

26.11.2009 TJUL New check for input errors regarding negative diameters in aerodrag module

03.12.2009 TJUL In wake model. Ainslie-15.exe replaced by ainslie-15.dll, to enable execution on linux platforms using WINE

21.12.2009 TJUL In wake model. Bug fixed dealing with several neighbouring wind turbines. Change in turbine order for ainslie15.dll input/output

30.12.2009 TJUL Flex integer format option in output.

04.01.2010 TJUL Change in variables in eigenvalue module to avoid stack errors

05.01.2010 TJUL Change in loadlinker merge criteria to avoid error with closely spaced nodes

31.01.2010 TJUL New control DLL module names type2_dll

31.01.2010 TJUL Possibility to input case_sensitive words using ' symbols. Especially related to control subroutine names.

15.02.2010 TJUL Change in wind shear logarithmic format to ensure a shear of zero below

global zero.

global%version='HAWC2MB 9.1'

29.03.2010 TJUL New parameter possible in mbdy moment_int actions command

31.03.2010 ANMH Possibility of external systems solved together with HAWC2, this goes for bodies and constraints

01.04.2010 TJUL Small update of result file sensorlist output for hawc_ascii and hawc_binary

15.04.2010 TJUL Updates in FORCE DLL module. New initialization option, label option.

17.05.2010 TJUL Update of wind ramps to speed up simulation time

29.06.2010 TJUL DLL module, type2_dll updated regarding first outputs in dll calls

16.07.2010 TJUL Small change in compiler settings.

global%version='HAWC2MB 9.2'

13.08.2010 TJUL Possibility for aerodynamic sections positioned according to ae_file input.

25.08.2010 TJUL Updated procedure for hub coo which defines coo with arbitrary rotor orientation

09.09.2010 TJUL Simulation stop enabled by external dll action.

13.09.2010 JOMH Updated eigenvalue solution procedure

13.09.2010 JOMH Zero-termination of all strings used for dll's

28.09.2010 TJUL Bug correction from version 9.1 regarding wind steps.

global%version='HAWC2MB 9.3w'

06.10.2010 TJUL Bug correction in interpolation of profile coef when several pc sets are used.

20.10.2010 TJUL Bug correction of predictor in newmark solver. Version 9.0 and 9.1 gave problems coupling in controllers

21.10.2010 TJUL Eigenvalue solver from version 9.2 removed and old 9.1 version included instead. Not correct solutions in all cases.

21.10.2010 TJUL Turbulence buffer not updated when buffer contains the full turbulence box.

global%version='HAWC2MB 9.4'

21.10.2010 TJUL Code updated for handling multiple aerodynamic rotors

26.10.2010 TJUL Be aware that the near wake induction model is not working in this edition. The normal induction model works fine though.

26.10.2010 JOMH Eigenvalue solver updated again, so now it should be both correct and robust

global%version='HAWC2MB 9.5'

12.11.2010 TJUL Bug fixed when outputting wake pos when no wake defined - code crash occurred.

17.11.2010 TJUL BEM rewritten in more structured way and to make sure local properties in grid points are all local. More correct for non-uniform loading

19.11.2010 TJUL IN BEM, time filters are now on induced velocities instead of factors, same goes with azimuthal yaw correction in induction.

21.11.2010 TJUL WSP lookup only performed for first iteration to save simulation time.

24.11.2010 TJUL Final adjustments of BEM induction

global%version='HAWC2MB 9.6'

27.11.2010 TJUL Improvements of yaw correction in BEM calculation, comparison with FIDAP

02.12.2010 TJUL Bugfix related to output of DWM wake position

global%version='HAWC2MB 9.7'

03.12.2010 TJUL Improvements of yaw correction in BEM calculation, comparison with FIDAP

03.12.2010 TJUL BEM restructured for more correct local CT prediction, time constants on induced velocities, yaw correction on induced axial velocities instead of factors

03.12.2010 TJUL BEM improved regarding aerodynamic yaw correction, compared with actuatordisc results

03.12.2010 TJUL BEM improved regarding dynamic time constants, compared with actuatordisc results

03.12.2010 TJUL Dyn. time constants reduced for first 10seconds to reduce time for initial equilibrium

global%version='HAWC2MB 9.8'

17.12.2010 TJUL Small bug correction. Crash occurred when aero output requested when rotor was not defined

28.12.2010 MHHA Bug fix regarding omega vector for aero module. Version9-2 to 9.7 had problems with rotor orientations azimuthally angle more than +-90deg.

global%version='HAWC2MB 9.9'

05.01.2011 TJUL Update regarding wake meandering, lowpass filter in time instead of area based filter

global%version='HAWC2MB 10.0'

17.01.2011 TJUL New calibration in DWM model. Ainslie_16.dll used.

global%version='HAWC2MB 10.1'

01.02.2011 TJUL BEM bug correction regarding induction for two bladed turbines. + DWM model deficit k1 parameter changed to 0.20 as default

global%version='HAWC2MB 10.2'

01.02.2011 TJUL User defined a-ct relation in BEM included

global%version='HAWC2MB 10.3'

08.02.2011 TJUL DLL action command aero bem_grid_a inserted.

global%version='HAWC2MB 10.4'

18.03.2011 TJUL In DWM model, k1 parameter default value changed back to 0.13 (good match with Egmond aa Zee measurements)

13.05.2011 TJUL/LEOB Bug corrections in the maghmh dynstall model

16.05.2011 ANMH Update and general improvement of hydrodynamic mass so is it coded in body coordinates (time varying)

17.05.2011 TJUL Small updates of modal_new.f90 to minimize stack usage

global%version='HAWC2MB 10.5'

17.05.2011 ANMH Concentrated hydrodynamic added masses also included in new procedure, which makes the solver converge faster and more stable than previous

global%version='HAWC2MB 10.6'

15.06.2011 TJUL Update of mbdy state_rot output

12.08.2011 TJUL Small update in variable check to fullfill requirements from new compiler

30.08.2011 TJUL Updates in hydrodynamics related to flooded members.

31.08.2011 HAMA/TJUL Limitation in tangential induction changed (tangential induction was not correct for high loading Ct)

31.08.2011 LEOB New updated model for dynamic stall of trailing edge flaps

05.09.2011 TJUL Updates in buoyancy regarding point change in diameters

05.09.2011 TJUL Update in structure output. Hydro added mass outputted if defined.

13.09.2011 LEOB New alfadot sensor in aero output

19.09.2011 TJUL Hydro added mass solver module updated for inner flooded area

26.09.2011 KNKR New spinner output sensor in aeroload module

global%version='HAWC2MB 10.7'

04.10.2011 TJUL External forces enabled for external modules

18.10.2011 TJUL In DWM model, Multiple wakes handled by choosing the deficit with largest wsp reduction. Previously a summation of deficits was performed

24.10.2011 TJUL In buoyancy part. Correction with eps at mudlevel, to improve stability when structure ends at mudlevel.

global%version='HAWC2MB 10.8'

02.11.2011 TJUL Fourier based low pass filter in DWM model instead of second order

filter

09.11.2011 LEOB/TJUL Update in dynamic induction to include the influence of trailing edge flaps

18.11.2011 TJUL New random output sensor added to general output commands.

22.11.2011 ANMH/TJUL Update in topology initial rotation velocities. Should decrease initial transients

22.11.2011 TJUL Aerodynamic load is removed for first 5 second and then gradually put on structure (fully applied at 10s) to avoid convergence problems during initial transients.

05.12.2011 TJUL Close dll's inserted to increase robustness on especially windows 7 platforms

05.12.2011 ANMH Eigenvalue solver update to account for non-coinciding nodes and free relative rotation

02.11.2011 ANMH Orientation of node in call to FORCE_DLL is changed (back) to global reference

02.11.2011 ANMH Disabling of constraints can now be handled - new command "disable_at" is introduced for constraints FIX0, FIX1, BEARING1 and BEARING2

02.11.2011 ANMH "INPUT" subroutine introduced in EXTSYS which enables HAWCDLL to send output to EXTSYS

06.11.2011 LEOB Update of dynamic stall models MHH and ATEF for correct handling of +-pi angle of attack crossing.

07.11.2011 TJUL New aero command 'output_profile_coef_filename', enables output of static interpolated profile coefficients

07.11.2011 ANMH/TJUL Update of initial structural velocity conditions, to reduce the initial transients in general

08.11.2011 TJUL New aero sensor, where is is possible to ouput the local elastic torsion

global%version='HAWC2MB 10.9'

15.12.2011 TJUL New procedure for calculating non-rotating hub coordinate system. important for free-yawing rotors.

global%version='HAWC2MB 11.0'

21.12.2011 TJUL New dll action commands (wind windspeed_u, wind winddir)

21.12.2011 TJUL For DWM wake model, Ainslie_17.dll replaces Ainslie_16.dll. Updated regarding number of neighbor turbines in the flow solver.

30.01.2012 TJUL Bug fix related to input of using different aerodynamic _ae sets.

13.01.2012 TJUL Bug fix related to naming of 'output_profile_coef_filename' in the aero block

13.01.2012 LEOB Update of deflection shape integrals in ATEF dyn. stall model. Based on UPWIND test cases.

21.03.2012 TJUL Update in DWM wake model. Turbulence intensity for kinematic viscosity is now from ambient tint and not meander turb tint.

23.03.2012 TJUL New sensor mbdy state_at2, where the xy offset is defined in local c2def coordinates instead of elastic axis coo.

global%version='HAWC2MB 11.1'

21.12.2011 TJUL New dll action commands (wind windspeed_u, wind winddir)

24.04.2012 TJUL Bug fix in IEC gust EWS. Wrong rotor diameter used in normalization - manual updated too.

global%version='HAWC2MB 11.2'

09.05.2012 TJUL Bug fix related to aerodynamic induction of one-bladed turbines.

global%version='HAWC2MB 11.3'

24.05.2012 TJUL In tower shadow JET model, distance criteria changed to match physical

units. Only of importance if tower radius < 1m

25.05.2012 TJUL Bug fix in tower shadow JET model. x-offset normalized with tower diameter instead of radius.

10.07.2012 TJUL Bug fix in S.O. dyn. stall model

10.07.2012 ANMH External output sensors in ESYS (external systems) enabled

10.07.2012 ANMH In hydro dynamics. Added mass and drag defined in two dimensions for new general hydro section

10.07.2012 ANMH DLL call for ESYS (external systems) changed to MEFF,CEFF,KEFF instead of newmark matrices

10.07.2012 ANMH Bug fix in output of mass centers

10.07.2012 ANMH Eigenvectors output together with eigenfrequencies for local bodies

20.07.2012 TJUL New wind sensor "wind free_wind_shadow" including effects of tower shadow

global%version='HAWC2MB 11.4'

20.07.2012 TJUL/HAMA Induction model for VAWTC's implemented.

15.08.2012 LEOB/TJUL/GEORG Update of induction model for high CT loading

11.09.2012 TJUL New aero output sensor "aero grid_all_induc_u"

27.09.2012 ANMH/TJUL In hawc ascii outout, limit of 500 channels removed

27.11.2012 TJUL Correction of bug regarding more _ae data sets than 2.

global%version='HAWC2MB 11.5'

06.12.2012 TJUL in orientation relative, new "relpos" command enabled

10.12.2012 TJUL Taylors hypothesis included in EOG gust

21.03.2013 GEPIR Speedup optimization modification of aeroload_induction_bem module

22.03.2013 TJUL New general output impulse function added

22.04.2013 TJUL Mann turbulence DLL is closed after turb generation, to remove fault seen on linux clusters with multiple turbulence set created

03.05.2013 TJUL New Flex int writing loop to minimize writing time on network drives

14.05.2013 TJUL Wave direction reversed to match manual definition

26.06.2013 TJUL Bug fix in VAWT module regarding aerodynamics on turbines with more blades than 2

27.06.2013 TJUL Reordering of output related to command "body_eigenanalysis_file_name". Frequencies first, later eigenvectors.

global%version='HAWC2MB 11.6'

25.09.2013 TJUL VAWT update for handling non vertical blade configs

30.09.2013 MACG Update of MHHA dyn. stall model to account for extra term important for VAWT operation

30.09.2013 TJUL In aero output. "tors_e" options inserted parallel to "tors_ang" to match explanantion in the manual

01.10.2013 TJUL Bug fix in aero drag, sign issue on force direction for negative force directions.

11.10.2013 TJUL With VAWT's aero center is moved to C_1/2 instead of C_3/4

03.01.2014 TJUL Encrypted blade data

04.01.2014 TJUL Move of update time call to external water kinematics dll. To ensure it is also called even if no hydro elements are used.

16.01.2014 TJUL Aero output parameters ct_local, cq_local and tiploss_f enabled

17.01.2014 TJUL Aero BEM induction. Small change in procedure of how to find induction grid loading. Now in line with VAWT approach.

27.02.2014 TJUL Change in shears, so size of shears change if windsteps are included.

06.03.2014 EBRA General prepare for intel compiler

13.03.2014 TJUL new aero option: 3D profile correction using Snel's approach

19.03.2014 LARH New stiffness proportional damping model

19.03.2014 ANMH New eigenvalue option including contributions from external systems

19.03.2014 ANMH Super element approach to condensate degrees of freedom for sub systems - eg jackets. Still not a perfect match with full DOF solution for

internal cross sectional members of a static undertermined structure!

20.03.2014 TJUL Turbulence export option re-implemented

14.05.2014 TJUL Encrypted password lib updated

14.05.2014 TJUL Procedure to ensure more robust auto shut-off of bem for idling/standstill conditions

global%version='HAWC2MB 11.7'

21.05.2014 TJUL Added stiffness contribution from version 11.6 is switched off again - caused increase in iterations especially for jackets

Memory use related to output sensors minimized

global%version='HAWC2MB 11.8'

16.07.2014 TJUL Create directory function increased for "../.." paths

22.07.2014 TJUL Dyn stall MHH model rolled back to version 11.5. The modification in 11.6 wrt. VAWT affected HAWT operation. A better dyn stall model for VAWT must be coded

22.07.2014 TJUL/ANYD/MMPE Comprehensive testprogram established. Used on this version and future.

global%version='HAWC2MB 11.9'

TJUL Read and write status declared when working with files. Should prevent access errors on net work drives especially

TJUL Stop code defined when code is stopped by internal criteria

12.09.2014 TJUL In BEM model with "use_mean_average" switched on, the inducted velocities are now found based on a consistent use of average wind speed in annular element.

19.09.2014 TJUL Update to BEM "use_mean_average" stuff from 12.09.2014. The local grid velocities was still present a few places

22.10.2014 TJUL/ANMH Code sync with svn version. Prepared for intel compilation.

23.10.2014 ANMH Contributions from inertial/gyroscopic effect of concentrated masses added in a general way. It is now OK to use concentrated massed on the rotor part.

global%version='HAWC2MB 12.0'

03.09.2014 TJUL Bug fix, userdefined number of azimuthal stations was not active. Default value of 16 stations were always present until now.

13.11.2014 GEORG Buxfix in BEM module, relevant when negative windspeeds occurs

08.12.2014 TJUL Update in SNEL 3D profile correction method

16.12.2014 GEORG Bugfix of mhh dyn stall model wrt rel velocity

16.01.2014 TJUL/MHHA/ANMH Bugfix wrt position of rotor center in aero variables

10.02.2015 MMPE Small update in call of output function (if aerodyn rotor not defined)

20.02.2015 TJUL Update in BEM "use_mean_average" approach + approach to find load in local BEM grid

global%version='HAWC2MB 12.1'

17.03.2015 TJUL Option to specify dyn stall model manually in the _ae file depending on radial location.

27.10.2014 CPAV TKIM anisotropic beam element added. CPAV damping introduced (the damping matrix is the same than MHH, but it's generalized to support 6x6 stiffness matrix)

20.03.2015 TJUL in wind speed module. factor used for ramps etc is also applied on wake deficits and turbulence to prevent negative wind speed in simulation start up.

20.03.2015 TJUL Update in output file write updated in order to allow more than 999 sensors

25.03.2015 TJUL Bugfix relatede to _ae dyn stall choice from 17.03.2015

30.04.2015 TJUL Type2_dll. New option to include "initstring" and "message"

05.05.2015 TJUL New option in DWM method (multiple_deficit_method) - summation or max deficit operator

27.05.2015 TJUL Update in Mann turbulence dll call. Now the mann_win32.dll and ressource_win32.dll must be present in same folder as HAWC2

28.05.2015 TJUL This version is compiled with intel visual fortran

global%version='HAWC2MB 12.2'

09.07.2015 TJUL Additional read specification when opening files. To avoid network multiple access errors

21.07.2015 ALSTA Compiler flags needed for HAWCStab2 compilation inserted.

24.07.2015 TJUL Update in create directories procedure - needed for special situations where the intel version was blocked from creating certain directories.

24.07.2015 TJUL Mann turbulence generator call moved from mann module directly. No need for the mann.dll or mann_win32.dll anymore

global%version='HAWC2MB 12.4 upd 1'

13.10.2015 TJUL Adjustments of export wind field output function, new commands possible.

14.12.2015 GEPIR Near wake model implemented

29.01.2016 tjul manual factor scaling of turbulence added as option

09.09.2016 tjul Output with "deltaT" intervals is now enabled

04.01.2017 tjul labels added to bearing outputs

06.01.2017 tjul Version handling system for water kinematics dlls

18.01.2017 tjul New met_mast_wind input option (time variable shear)

global%version='HAWC2MB 12.5'

09.05.2017 ANMH 6 call getfreewind from ext system. enable/disable constraint update

06.07.2017 ANMH 10 Updates regarding output of eigenvectors related to body_eigenanalysis. Bugfix with respect to critical damped modes

global%version='HAWC2MB 12.6 '

08.08.2017 anmh 1 Constraint enable/disable made active for $t \geq 0$ (previous $t > 0$)

24.08.2017 tjul 2 azimuthal user dependent rotor induction a-ct enabled. Previous userdefined (radial) a-ct dependent induction limited to linear interpolation

29.08.2017 tjul/dominique 3 Nacelle lidar sensor + effective wind speed sensor. Written by Dominique Philipp Held domhel@dtu.dk

19.09.2017 tjul 4 New mbdy sensor forcemoment_interp. This gives interpolated forces and moments independent on model discretisation (interpolated between nodal results)

25.09.2017 tjul 5 Bugfix in type2.dll initstring

06.10.2017 tjul 6 New sensor id reference option

06.10.2017 tjul 6 New wind free_wind_center_pos0 and free_wind_hor_center_pos0 option

06.10.2017 tjul 6 Bugfix in aero induction outputs when induction model not used.

06.10.2017 tjul 6 New mbdy sensor statevec_new. This gives positions and rotations based on local chord orientation, either absolut of elastic deformation

20.10.2017 tjul 7 Label option on all outputs enabled

25.10.2017 mmpe 8 Start/stop in visualization

25.10.2017 mmpe 8 Crash prevented when scale_time_start > time_stop for no-turbulent simulation

26.10.2017 tjul 9 Wind sensors wind_free_center_pos0 and wind_free_hor_center_pos0 added

27.10.2017 frbe 10 Aerodynamic noise model included

13.11.2017 tjul 11 Bugfix related to label option and dll sensor linking i version 12.6 rev 7-10, noise module bypassed (aeronoise bypassed)

21.11.2017 tjul 12 Bugfix in tower pot2 method to avoid strange situations causing crashes. (aeronoise bypassed)

27.11.2017 tjul 13 Aeronoise included again

14.12.2017 tjul 14 Bugfix in flex binary output format using reduced timestep outputs

22.12.2017 tjul 15 Bugfix in output with dll reference system (affected only versions from 12.6 rev 6). This is now the final version 12.6

Risø's research is aimed at solving concrete problems in the society.

Research targets are set through continuous dialogue with business, the political system and researchers.

The effects of our research are sustainable energy supply and new technology for the health sector.

