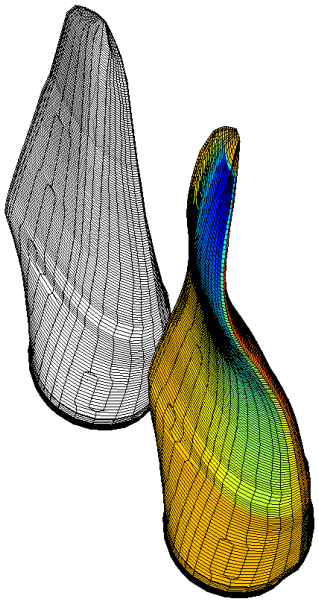
CHALMERS



Fluid Structure Interaction on Wind Turbine Blades

Master's thesis in Solid and Structural Mechanics And Fluid Dynamics

JONAS NORLIN
CHRISTOFFER JÄRPNER

Department of Applied Mechanics
Division of Dynamics And Division of Fluid Dynamics
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2012
Master's thesis 2012:24

MASTER'S THESIS IN SOLID AND STRUCTURAL MECHANICS AND FLUID DYNAMICS

Fluid Structure Interaction on Wind Turbine Blades

JONAS NORLIN
CHRISTOFFER JÄRPNER

Department of Applied Mechanics
Division of Dynamics And Division of Fluid Dynamics
CHALMERS UNIVERSITY OF TECHNOLOGY

Gothenburg, Sweden 2012

Fluid Structure Interaction
on Wind Turbine Blades
JONAS NORLIN
CHRISTOFFER JÄRPNER

Master's thesis 2012:24
ISSN 1652-8557
Department of Applied Mechanics
Division of Dynamics And Division of Fluid Dynamics
Chalmers University of Technology
SE-412 96 Gothenburg
Sweden
Telephone: +46 (0)31-772 1000

Cover:
The undeformed NREL 5 blade and a snapshot of a flutter mode, with the finite element mesh and colors for static pressure

Chalmers Reproservice
Gothenburg, Sweden 2012

Fluid Structure Interaction
on Wind Turbine Blades
Master's thesis in Solid and Structural Mechanics And Fluid Dynamics
JONAS NORLIN
CHRISTOFFER JÄRPNER
Department of Applied Mechanics
Division of Dynamics And Division of Fluid Dynamics
Chalmers University of Technology

Abstract

In the wind industry, there is a trend towards building larger and larger turbines. In order to keep increasing the power output of wind turbines, the swept area of the rotors have to be increased, i.e. the size of the turbines need to increase. This presents new challenges for structural engineers and might require blade materials that are both lighter and stiffer than the ones currently used.

Traditionally, blades have been made of fibre-glass, but a study from the University of Kalmar [1] shows great potential for significant weight reduction, and perhaps a reduction of cost, if carbon-fibre based technology is used instead. This work is aimed to aid the work of designing new blades by providing an analysis tool for wind turbine blades. It can for example be used to evaluate the pros and cons of using more lightweight material such as carbon fibre in a wind turbine blade, and to explore the possibilities the use of such a material can bring. The thesis will be a step towards a design basis for the design of a lightweight blade where the blades aeroelastic loads and certification guidelines are considered.

The trend of larger turbines and relatively softer blades as well as some other current design trends in the wind industry increases the risk of aeroelastic instabilities such as flutter. Therefore a tool to calculate the aerodynamic damping of the shell model is developed.

To create the structural model, scripts in MATLAB and the FEMAP API are developed. These allow the geometry of almost any wind turbine blade to be created and meshed in FEMAP with little to no user intervention after the designing characteristics of the turbine blade has been specified. The output is a finite element model that can be used directly in analyses of static strength, buckling, fatigue etc. The stiffness and mass matrix is also output for use in further analysis in MATLAB.

To calculate the forces from the wind three different codes are used. Most focus is put on the blade element momentum (BEM) method which is very widely used in wind turbine applications. The two other methods used are different versions of the vortex method (Helical and Free) which are possible replacements for BEM and are widely used in helicopter and propeller applications. A lot of focus is put on how to, with as little time as possible, transform the forces calculated by the BEM and vortex methods to pressure distributions along the blade.

Keywords: Fluid Structure Interaction, Blade Element Momentum, Free Vortex Method, Wind Turbines, Flutter, Aeroelasticity

Contents

1 Introduction

1.1 Background

The trend in the wind industry is towards larger turbine. This is because in order to keep increasing the performance of wind turbines, the swept area of the rotors have to be increased. This requires blade materials that are both lighter and stiffer than the ones currently used. Traditionally, blades have been made of glassfibre, but a study from the University of Kalmar [1] shows great potential for significant weight reduction, and perhaps a reduction of cost, if carbon-fibre based technology is used instead.

This work is aimed to aid the work of designing new blades by providing a sophisticated computer model for predicting aeroelastic forces and their effect on the blade.

The trend of larger turbines, as well as some current design trends in the wind industry increases the risk of aeroelastic instabilities such as flutter, so this is given special consideration.

1.2 Purpose

The purpose of this thesis is to create a tool that will aid the design of new wind turbine blades. The tool can for example be used to evaluate the pros and cons of using more lightweight material such as carbon fibre in a wind turbine blade, and to explore the possibilities the use of such a material can bring.

The behaviour at different operating conditions is modeled, and quantities like power and static displacement are calculated. An aeroelastic analysis is also carried out.

The thesis will be a step towards a design basis for the design of a lightweight blade where the blades aeroelastic loads and certification guidelines are considered.

1.3 Software

The programs developed in this thesis are mostly written in the programming environment MATLAB. To create the blade structural model, FEMAP is used as a pre-processor and NX Nastran is used to write the stiffness and mass matrix. When calculating the pressure distribution the program xfoil is used. These are used in subsequent analyses.

1.4 Limitations

This thesis will only concern the blades. The tower, drive train etc. will not be modelled or investigated in detail.

Only standalone turbines will be considered, wake effects of upwind turbines will not be simulated.

The economics of new blade design will not be considered directly. The control systems for reducing torque and power and their effect and interaction with aerodynamic forces and elastic responses of the blade will not be treated.

2 Blade Model

2.1 Blade design

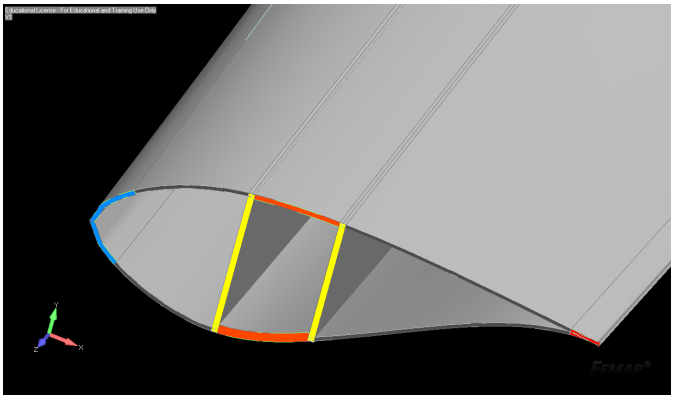


Figure 2.1.1: Cross section of a blade showing spar caps (orange), shear webs (yellow), leading edge reinforcement (blue) and trailing edge reinforcement (red).

The cross section of a typical wind turbine blade is shown in figure 2.1.1. Throughout this thesis, we have used the box spar design with two shear webs, but there are also designs with only one web. The trailing and leading edge reinforcements gives additional edgewise and torsional stiffness.

2.1.1 Spar caps

The spar caps carry the majority of bending stresses that are due to motion in the flapwise direction, which is the direction of the weak principal axis and is roughly perpendicular to the chord line. The moments in the flapwise direction are large, and the distance between the spar caps is limited by aerodynamic considerations. Therefore the sparcaps have to be relatively thick, as can be seen in figure 2.1.1 and made of a stiff fiber. As these stresses are mostly unidirectional it is efficient to use an unidirectional layup here.

It has been suggested to make the layup slightly unsymmetric and titled. This would give a strong bending- twisting coupling so that when the blade bends it also twists to decrease the angle of attack. This would serve as a passive control against too large deflections and would supplement the pitch control.

2.1.2 Shear webs

The role of the shear webs is to absorb the shear stresses associated with bending of the blade. It can be shown [2] that a good laminate layup for carrying shear stresses is obtained if it is composed of plys at $\pm 45°$. Shear webs usually have a core of foam or similar material to avoid local buckling.

2.1.3 Design for aerodynamic stability

In this section we discuss the concept of aerodynamic stability. One way to increase aerodynamic stability is to move the center of gravity closer to the leading edge. If

the center of gravity is aft of the aerodynamic center the risk of flutter is eliminated completely, see [3]. This could be accomplished by adding mass to the leading edge.

Against stall induced edgewise vibrations, better aerodynamic stability can be obtained by structural pitch, which means rotating the spar caps about the pitch axis [4]. The purpose is to rotate the principal axis, so that the blade vibrates in a direction that gives a more favorable aerodynamic damping.

2.2 Blade model

To create the structural model of the blade, a finite element approach using 2D orthotropic elements is chosen. The traditional approach for wind turbine blades is to use beam elements. A beam element model has less computational cost, does not require techniques like those in chapter 3.3.1 to be integrated with a aerodynamic code, and has been shown to give a good agreement with experiment in most situatons. However a structural model of a wind turbine blade for use in aeroelastic analysis, buckling or very extreme loading situations should include bending-torsion coupling, warping and other effects. These can be all be modeled with beam elements, but the derivation of correct parameters requires detailed information on the layout of the cross section which makes a beam element not much less complicated than a shell element model. With ever faster computers and model reduction techiques the computational cost and time of handling a 10000+ element model can be kept at an acceptable level.

2.2.1 Finite element model of wind turbine blades

The full definition of a blade is contained within a cell structure in MATLAB. The blade is defined from a number of sections, where each section has properties like chord length, twist angle, pitch axis etc. The blade cell structure and functions used are written to facilitate changing the design easily.

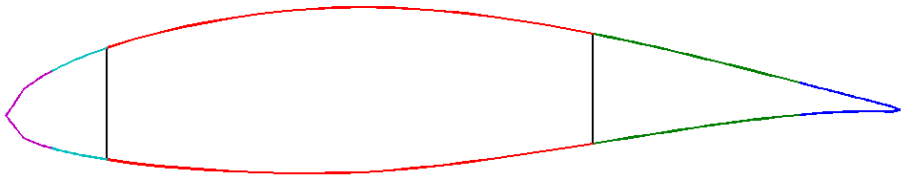


Figure 2.2.1: Blade section divided into strips

The profile type of the section defines a spline that outlines the geometry. The profile is split at certain points specified by the user, dividing the blade section into strips, see figure 2.2.1. The material of the blade is defined by assigning a material layup to each strip. A shear web or other type of reinforcement can be defined by specifying two points where the spline is split. These points will then be joined by a surface in the preprocessor.

The blade cell is input to a function in MATLAB that writes files readable to a script written in the FEMAP API. This script creates surfaces representing the external geometry of the blade, as well as surfaces representing the shear webs. These surfaces are then meshed with laminate shell elements. All geometry is thus considered thin.

The elements created on the external surfaces are offset inwards, while the shear web surfaces represent midsurfaces. Offsetting shell elements can create some problems, more on this in section 2.2.2. Most of the elements created are four node quadrilaterals, though some three node triangles have to be used as the blade narrows along the blade span. Overall the elements created have good shape in terms of Jacobians, aspect ratio, skewness etc. The most troublesome area is where the cylindrical blade section transforms into the Delft University blade shape sections. In the resulting model each node of the shell elements has six degrees of freedom, three translations and three rotations.

Table 2.2.1: Twist angle results from test run with thin walled cylinder, with and without midside nodes

elements	1281	5081	20000
Analytical solution	$3.18 \cdot 10^{-7}$	$3.18 \cdot 10^{-7}$	$3.18 \cdot 10^{-7}$
Midplane, no midside nodes	$3.26 \cdot 10^{-7}$	$3.20 \cdot 10^{-7}$	$3.18 \cdot 10^{-7}$
Offset node, no midside nodes	$3.28 \cdot 10^{-7}$	$3.20 \cdot 10^{-7}$	$3.19 \cdot 10^{-7}$
Midplane and midside nodes	$3.27 \cdot 10^{-7}$	$3.20 \cdot 10^{-7}$	$3.18 \cdot 10^{-7}$
Offset node and midside nodes	$4.41 \cdot 10^{-7}$	$6.24 \cdot 10^{-7}$	$9.18 \cdot 10^{-7}$

The component is meshed and NX Nastran is used to formulate the stiffness and mass matrix. The matrices are then written to MATLAB where they are used in the subsequent analysis.

In addition to the stiffness and mass matrix output from FEMAP, a number of pre-set analysis cases are set in FEMAP, such as modal analysis, buckling analyses under force of gravity, time- varying loads etc.

When seeding mesh points, the blade model is prepared to have nodes on the radial position of the stations on table 2.2.3. This helps the analysis in section 3.3.1 considerably.

2.2.2 Accuracy of torsion results when using offset shells

It has been shown that some solvers give erroneous results for torsion when shells with offset nodes are used. ANSYS has made some changes in their shell element formulation that appear to have solved this problem [5].

When this problem was first seen it was on a circular hollow cylinder with mid-radius $R = 1$, thickness $t = 0,1$ and length $L = 20$, fixed at one end and subjected to torsion at the other. We have modeled this in NX Nastran through FEMAP and compared the results to the analytical solution, both with the offset node approach and midplane modeling approach.

The cylinder is of an isotropic material with G = 100 GPa and subjected to a torque M = 1 kNm one end, while the other is fixed.

The resulting maximum twist angle for different element sizes can be seen in the table 2.2.1, together with the analytical solution. Evidently, the problem exists in NX Nastran as well, but only when using mid-side nodes. It is supposed that for the purposes of this thesis, sufficient accuracy is obtained with four-node elements.

2.2.3 NREL Blade

The blade that serves as a test case in this work has been the blade used in the 5MW reference turbine by NREL [6]. This is a fictitous turbine for offshore placement, and is heavily based on the real REpower 5M turbine. The blades of the turbine are defined by sectional properties, based on a blade by LM-Glasfiber Holland. There has been much work done on this blade, for example an aeroelastic analysis on a very similar blade by C.Lindenburg [4] and a stability analysis under influence by wind by Bir and Jonkman [7]. Hansen studies flutter in this blade as a function of rotor speed in [3].

A full FE model of the blade would require the full layup properties along the span of the blade. These properties were however not available. Therefore some reverse engineering has been done based on the equivalent cross sectional data that are

Table 2.2.2: Material property data used by Sandia [8]

	E-LT-5500 / EP-3	Saertex / EP-3	SNL Triax	Foam	Resin
E_L [GPa]	41.8	13.6	27.7	0.256	3.5
E_T [GPa]	14.0	13.3	11.8	0.256	2.5
G_{LT} [GPa]	2.63	11.8	7.2	0.022	1.4
ν_{LT}	0.28	0.5	0.39	0.3	0.3
ρ $[kg/m^3]$	1920	1780	1780	200	1100

provided in [6] and [4]. The blade design for this thesis has been based on Sandia's concept of a 100 meter all glass-fibre blade [8], which itself is based on the NREL blade but scaled up and modified to be stronger against buckling.

The spanwise properties of the blade, like chord and twist angle, are based on those used in the Sandia report and scaled down. These are shown in table 2.2.3. There are many stations near the root in order to put in the many ply drops here. Between station 2 and 16, the profile types are interpolated between a cylinder and the profile known as DU40. There are no aerodynamic data for these transitional profiles so we have not put any aerodynamic load at these stations. As these stations are close to the root, the load and bending moment from these are not significant. The twist is the structural twist, which is the angle the principal axis (the edgewise direction) has with the rotor plane. It happens to coincide with the twist of the chord line for this blade, but this is not true for all blades.

Laminate thicknesses in the spars, leading and trailing edge reinforcements and the root skin buildup along the blade span are shown in table 2.2.4. The same materials as in the Sandia blade has been speciefied in the layup, see table 2.2.2. There are three glass fibre - epoxy composites used, E-LT-5500, Saertex and SNL Triax. E-LT-5500 is an uni-directional laminate. In the blade it has used in the spar caps and the trailing and leading edge reinforcements. Saertex is a double bias fibreglass material and is primarily used in the shear webs. SNL Triax is used as a skin material. In additon to the fibre glass composites, there is foam used in skins and shear webs and extra resin, which make up about 6 % of the weight. The gelcoat on the outer skins that was used in the Sandia report has been neglected.

Some modifications were made to the design by Sandia. The third shear web was removed. Thicknesses were overall smaller than what they would be by a simple down-scaling. The spar was made relatively narrower and is at a constant width of 723 mm along most of the blade before it tapers towards the end. A leading-edge reinforcement was added so that the shear center were closer to the leading edge and more in line with those used by C. Lindenburg [4]. This addition also had the effect of moving the center of gravity closer to the leading edge, which might make the blade more stable.

The blade model is not supposed to exactly represent the actual blade, it just needs to be close enough overall so that it can be used as a test case and that some meaningful comparison with previous results on the same blade can be made.

Table 2.2.3: Blade properties along the blade span

Station	Radial Position [m]	Chord [m]	Twist Angle [^{o}]	Fractional Pitch	Profile type
1	0	3.5018	13.308	0.5	Cylinder
2	0.3075	3.5018	13.308	0.5	Cylinder
3	0.4305	3.5018	13.308	0.5	Interpolated
4	0.5535	3.5018	13.308	0.5	Interpolated
5	0.6765	3.5018	13.308	0.5	Interpolated
6	0.7995	3.5018	13.308	0.5	Interpolated
7	1.4760	3.5621	13.308	0.499	Interpolated
8	1.5990	3.5738	13.308	0.498	Interpolated
9	2.8905	3.7257	13.308	0.483	Interpolated
10	4.1820	3.8770	13.308	0.468	Interpolated
11	5.4735	4.0289	13.308	0.453	Interpolated
12	7.0110	4.2035	13.308	0.435	Interpolated
13	8.9790	4.4372	13.308	0.41	Interpolated
14	10.0245	4.5535	13.177	0.40	Interpolated
15	11.0085	4.6445	13.046	0.39	Interpolated
16	11.9925	4.6912	12.915	0.38	DU 40
17	13.6530	4.6648	12.133	0.378	DU 40
18	15.3135	4.6051	11.350	0.377	DU 35
19	16.9740	4.5184	10.568	0.375	DU 35
20	22.0170	4.2576	9.166	0.375	DU 30
21	26.9985	3.9538	7.688	0.375	DU 25
22	31.9800	3.6377	6.180	0.375	DU 21
23	37.0230	3.3315	4.743	0.375	DU 21
24	41.0205	3.0867	3.633	0.375	NACA 64
25	42.0045	3.0258	3.383	0.375	NACA 64
26	45.0180	2.8419	2.735	0.375	NACA 64
27	46.9860	2.7195	2.348	0.375	NACA 64
28	52.0290	2.4139	1.380	0.375	NACA 64
29	54.9810	2.2257	0.799	0.375	NACA 64
30	57.9945	1.7368	0.280	0.375	NACA 64
31	58.8555	1.4606	0.210	0.375	NACA 64
32	59.7780	1.1291	0.140	0.375	NACA 64
33	60.6390	0.7429	0.070	0.375	NACA 64
34	61.5000	0.3075	0	0.375	NACA 64

Table 2.2.4: Laminate thicknesses along the blade span

Section	Z (m)	Suction side spar (mm)	Pressure side spar (mm)	Root skin (mm)	Trailing edge (mm)	Leading edge (mm)
1	1.5			34.0		
2	1.8	0.5	0.3	34.0	0.2	0.5
3	1.9	1.0	0.5	34.0	1.8	4.2
4	2.1	1.5	0.8	34.0	1.8	4.2
5	2.2	2.0	1.1	34.0	1.8	4.2
6	2.3	4.9	2.6	34.0	1.8	4.2
7	3.0	6.4	3.4	34.0	1.8	4.2
8	3.1	6.4	3.4	34.0	2.0	4.8
9	4.4	9.8	5.3	34.0	3.0	6.9
10	5.7	14.8	7.9	10.2	4.1	9.5
11	7.0	25.1	13.5	5.7	5.7	13.2
12	8.5	33.4	18.0	1.9	7.5	17.5
13	10.5	46.2	24.9		9.1	21.2
14	11.5	54.6	29.4		11.3	26.5
15	12.5	58.5	31.5		13.6	31.8
16	13.5	66.9	36.0		13.6	31.8
17	15.2	66.9	36.0		13.6	31.8
18	16.8	66.9	36.0		13.6	31.8
19	18.5	62.9	33.9		6.8	15.9
20	23.5	58.5	31.5		6.8	15.9
21	28.5	54.6	29.4		3.4	7.9
22	33.5	50.2	27.0		1.8	4.2
23	38.5	41.8	22.5		0.9	2.1
24	42.5	33.4	18.0		0.9	2.1
25	43.5	31.5	16.9		0.9	2.1
26	46.5	23.1	12.4		0.9	2.1
27	48.5	16.7	9.0		0.9	2.1
28	53.5	8.4	4.5		0.9	2.1
29	56.5	4.4	2.4		0.9	2.1
30	59.5	2.5	1.3		0.9	2.1
31	60.4	2.5	1.3		0.9	2.1
32	61.3	2.5	1.3		0.9	2.1
33	62.1	2.5	1.3		0.9	2.1
34	63.0					

2.3 Modal reduction

Starting from the undamped equation of motion (EOM) with the mass matrix $\mathbf{M}$, stiffness matrix $\mathbf{K}$, and degrees of freedom $\mathbf{q}$:

$$\mathbf{M}\ddot{\mathbf{q}} + \mathbf{K}\mathbf{q} = \mathbf{f} \tag{2.3.1}$$

a state-space model is obtained by introducing the state vector with nodal displacements and velocities $\mathbf{z}_s$:

$$\mathbf{z}_s = \begin{bmatrix} \mathbf{q} \\ \dot{\mathbf{q}} \end{bmatrix} \tag{2.3.2}$$

then the equations of motion can be written:

$$\dot{\mathbf{z}}_s = \mathbf{A}_s\mathbf{z}_s + \mathbf{B}_s\mathbf{f} \tag{2.3.3}$$

where:

$$\mathbf{A}_s = \begin{bmatrix} \mathbf{0} & \mathbf{I} \\ -\mathbf{M}^{-1}\mathbf{K} & \mathbf{0} \end{bmatrix} \tag{2.3.4}$$

$$\mathbf{B}_s = \begin{bmatrix} \mathbf{0} \\ \mathbf{M}^{-1} \end{bmatrix} \tag{2.3.5}$$

in equations 2.3.4 and 2.3.5 we note that the inverse of $\mathbf{M}$ is required. However there was no way to make NASTRAN use the consistent mass formulation from the FEMAP Application Programming Interface (API). The mass matrix was given zero inertia in the rotational degrees of freedom and was therefore not invertible. To circumvent this problem we chose to do a modal reduction, i.e. to let the n displacements in $\mathbf{q}$, depend on a smaller number m modal degrees of freedom in $\boldsymbol{\eta}$:

$$\mathbf{q} = \mathbf{Q}\boldsymbol{\eta}, \tag{2.3.6}$$

where Q is the nxm modal matrix. The eigenmodes and eigenvalues are obtained by making the harmonic ansatz $\mathbf{q} = \hat{\mathbf{q}}e^{i\omega t}$, and inserting into 2.3.1:

$$-\omega^2\mathbf{M}\hat{\mathbf{q}} + \mathbf{K}\hat{\mathbf{q}} = 0$$

$$\Rightarrow \tag{2.3.7}$$

$$\omega^2\mathbf{M}\hat{\mathbf{q}} = \mathbf{K}\hat{\mathbf{q}} \tag{2.3.8}$$

equation 2.3.8 is the generalized eigenvalue problem, and has n solutions. We solve for the m eigenvalues and eigenvectors with the smallest eigenfrequency ω, and collect the associated eigenvectors in the modal matrix $\mathbf{Q} = \begin{bmatrix} \hat{\mathbf{q}}_1 & \hat{\mathbf{q}}_2 & \dots & \hat{\mathbf{q}}_m \end{bmatrix}$. The modal matrix $\mathbf{Q}$ diagonalizes the stiffness and mass matrix and the modes are mass diagonalized, meaning:

$$\mathbf{Q}^T\mathbf{M}\mathbf{Q} = \mathbf{I} \tag{2.3.9}$$

$$\mathbf{Q}^T\mathbf{K}\mathbf{Q} = \bar{\mathbf{K}} = diag(\omega_i^2) \tag{2.3.10}$$

this means that the undamped equation of motion become m uncoupled equations:

$$\ddot{\eta}_i + \omega_i^2 \eta_i = \bar{f}_i \tag{2.3.11}$$

on matrix form:

$$\ddot{\boldsymbol{\eta}} + diag(\omega_i^2)\boldsymbol{\eta} = \bar{\mathbf{f}} \tag{2.3.12}$$

where we introduce the modal force $\bar{\mathbf{f}} = \mathbf{Q}^T \mathbf{f}$.

Letting $\mathbf{z} = \begin{bmatrix} \boldsymbol{\eta} \\ \dot{\boldsymbol{\eta}} \end{bmatrix}$ we can now write:

$$\dot{\mathbf{z}} = \mathbf{A}\mathbf{z} + \mathbf{B}\mathbf{f} \tag{2.3.13}$$

where:

$$\mathbf{A} = \begin{bmatrix} \mathbf{0} & \mathbf{I} \\ -diag(\omega_i^2) & \mathbf{0} \end{bmatrix} \tag{2.3.14}$$

$$\mathbf{B} = \begin{bmatrix} \mathbf{0} \\ \mathbf{Q}^T \end{bmatrix} \tag{2.3.15}$$

with this formulation of the equations of motion, we can go ahead with the aerodynamic stability analysis in chapter 4.1.

2.3.1 Structural damping

The damping matrix is constructed in MATLAB after the modal reduction. We have used modal damping, which keeps the orthogonaliy of equation 2.3.12. To get a damping ratio ζ_i in the i:th mode, we let:

$$\bar{V}_{ii} = 2\zeta_i \omega_i \tag{2.3.16}$$

the modal equation of motion then reads:

$$\ddot{\eta}_i + 2\zeta_i \omega_i \dot{\eta}_i + \omega_i^2 \eta_i = \bar{f}_i \tag{2.3.17}$$

on matrix form, with $\bar{K} = diag(\omega_i^2)$ and $\bar{V} = diag(2\zeta_i \omega_i)$:

$$\ddot{\boldsymbol{\eta}} + \bar{\boldsymbol{V}}\dot{\boldsymbol{\eta}} + \bar{\boldsymbol{K}}\boldsymbol{\eta} = \bar{\mathbf{f}} \tag{2.3.18}$$

the state-space model then becomes:

$$\mathbf{A} = \begin{bmatrix} \mathbf{0} & \mathbf{I} \\ -\bar{\mathbf{K}} & -\bar{\mathbf{V}} \end{bmatrix} \tag{2.3.19}$$

in our case, we have chosen a damping ratio $\zeta_i = 0.477465\%$ in all modes in agreement with the NREL report, see [6].

3 Aerodynamics

In the previous chapter the method and theory behind how the Finite Element Model of the blade was created was presented. In this chapter the aerodynamic forces are calculated by various means and then applied on to the blade. In the sub-chapter 3.1 the forces are calculated by various methods, one of these methods are the so-called BEM-method which is very common for wind-turbine applications. The other two methods are two different versions of the vortex method based on the lifting line theory, which is used a lot in the helicopter industry.

These methods are very fast, but, they only calculate the sum of the forces on a section of the blade. Therefore in chapter 3.2 a method for using the forces calculated to a pressure distribution is described. By applying the pressure distribution as forces on to the blade, the deformation of the blade can easily be calculated and a one-way fluid structure interaction coupling has been achieved.

Since all the methods used for calculating the aerodynamic forces are grid-less it is very easy to go from one-way coupling to two-way, and this is described in chapter 3.3.

3.1 Methods for calculating the aerodynamic forces

As mentioned earlier the aerodynamic forces used in this thesis is calculated by three different methods. The Blade Element Momentum (BEM) method, which is described in chapter 3.1.1 is the most commonly used method for calculating aerodynamic loads in the wind-power industry.

The two other methods, the Helical Vortex Method (HVM), and the Free Vortex Method (FVM) are not much used for wind turbines yet, but is instead very used in the helicopter industry and in the propeller industry. These methods may see increasing use in the wind-power industry as well. Therefore in chapter 3.1.2 these two methods are described.

3.1.1 Classical BEM Method

Introduction

BEM is a very common tool for wind turbine applications, the benefits of BEM is the very short computational time and the fact that it is accurate, at least for the cases for which BEM is suitable for.

In short, the benefits of BEM are:

- Very fast.
- Accurate.

The negative sides are:

- No way to define the geometry in flap or edge wise direction, (for example pre-bend or a curved blade).
- Engineering models needed.

BEM can accurately be used when the blade is straight (no complicated shapes in either direction), and the analysis is done assuming a steady state. The reason for this is because of the actuator disk theorem and Prandtl's tip loss factor. Prandtl's

tip loss factor is an engineering model that have been created to deal with the fact that actuator disk model assumes "infinite" amount of blades.

The Theory

The Blade Element Momentum (BEM) theory[1] is a very widely used method for calculating the forces on a wind turbine. By using the actuator disk theory where the disk changes the pressure and the rotation of the fluid, and couple it with blade theory a very fast tool can be created.

The actuator disk theory assumes that the blade is replaced by a circular plane that changes the pressure, and creates a rotational force on the fluid, see figure 3.1.1.

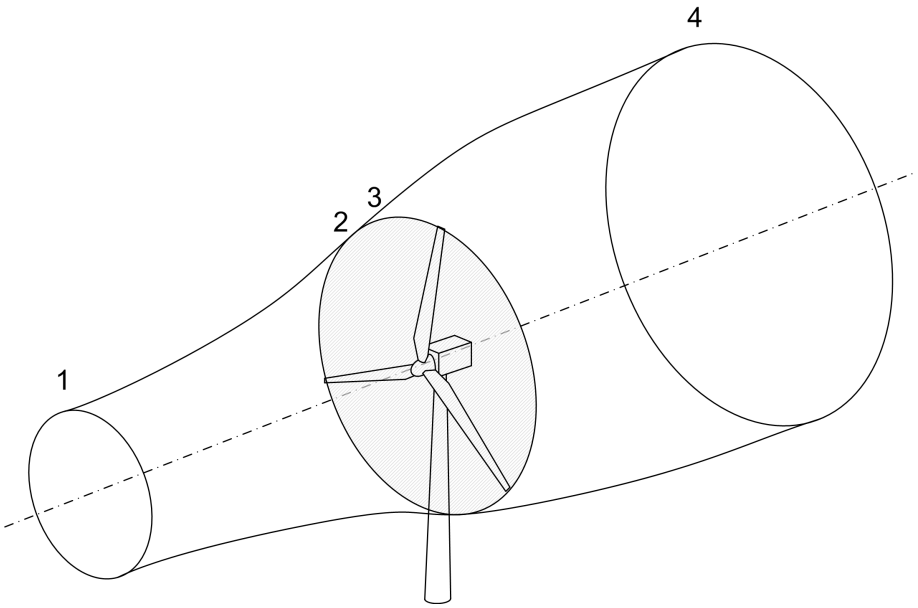


Figure 3.1.1: Actuator disk model.

By the actuator disk theory the thrust can be calculated as the pressure drop over the actuator disk.

$$T = \Delta p A \tag{3.1.1}$$

and the induced moment can be calculated as:

$$dM = \rho u \omega r^2 dA \tag{3.1.2}$$

where Δp is the pressure drop and A is the area of the disk i.e.

$$\Delta p = p_2 - p_3 \tag{3.1.3}$$

$$A = \pi R^2 \tag{3.1.4}$$

assuming that the flow is incompressible and stationary Bernoulli's equation can be used to calculate p_2 and p_3. This is done by calculating the state far upstream of the blade, and just before it (between 1 and 2) and calculating the state for far downstream of the blade and just after it (between 4 and 3).

$$H_1 = p_1 + (\rho u_1^2)/2 = p_2 + (\rho u_2^2)/2 \tag{3.1.5}$$

[1]The derivations shown in this chapter have been extracted from [9] and [10]

$$H_2 = p_3 + \rho(u_3^2 + \omega_3^2 r^2)/2 = p_4 + \rho(u_4^2 + \omega_4^2 r^2)/2 \tag{3.1.6}$$

where:

$$\Delta p = p_2 - p_3 \implies p_3 = p_2 - \Delta p$$

also

$$\Delta v = 0 = v_2 - v_3 \implies v_2 = v_3 \tag{3.1.7}$$

and

$$p_4 = p_1 \tag{3.1.8}$$

adding equations these equations and you get:

$$H1 = p_1 + \rho u_1^2/2 = p_2 + \rho u_2^2/2 \tag{3.1.9}$$

$$H2 = p_2 - \Delta p + \rho(u_2^2 + \omega_3^2 r_3^2)/2 = p_1 + \rho(u_4^2 + \omega_4^2 r_4^2)/2 \tag{3.1.10}$$

by combining these equations the head drop can be calculated as:

$$\Delta H = -\Delta p + \rho\omega_3^2 r_3^2/2 \tag{3.1.11}$$

where the total pressure head also can be calculated as:

$$0 = p_1 - p_4 = \rho(u_4^2 - u_1^2)/2 + \rho(\omega_4^2 r_4^2 - \omega_3^2 r_3^2)/2 + \Delta p \tag{3.1.12}$$

due to the fact that $\Delta H = 0$, the pressure drop over the blade can be written as:

$$\Delta p = \rho(\Omega + \omega/2)\omega r_3^2 \tag{3.1.13}$$

where Ω is the rotation of the fluid close to the blade.

By combining these equations the total pressure drop can be calculated as:

$$0 = \rho(u_4^2 - u_1^2)/2 + \rho\omega_4^2 r_4^2(\Omega + \omega/2) \tag{3.1.14}$$

since the angular velocity omega is supposed to be small, the term ω_2 can be neglected.

By applying these assumptions, on the actuator disk model the thrust and the moment can be calculated as:

$$dT = (u_1 - u_4)dm = 2\pi\rho r_3 u_1(u_1 - u_4)dr \tag{3.1.15}$$

Similarly the momentum can be calculated as:

$$dM = r\omega dm = 2\pi\rho u_1 r_3^2 \omega dr \tag{3.1.16}$$

where m is the mass of the fluid.

By substituting some variables, these equations can be written as:

$$dT = (V_0 - u_1)\, d\dot{m} = 2\pi\rho u\, (V_0 - u_1)\, dr \tag{3.1.17}$$

$$dM = rC_\Theta d\dot{m} = 2\pi r^2 \rho u C_\Theta dr \tag{3.1.18}$$

By expressing u as $u = (1-2a)V_0$ these two equations can be written as:

$$dT = 4\pi r \rho V_o^2 a\, (1-a)\, dr \tag{3.1.19}$$

$$dM = 4\pi r^3 \rho V_o \omega\, (1-a)\, a' dr \tag{3.1.20}$$

and the angle of attack can be calculated as:

$$\alpha = \phi - \Theta \tag{3.1.21}$$

where:

$$\tan(\phi) = \frac{(1-a)\, V_0}{(1+a')\, \omega r} \tag{3.1.22}$$

which makes it possible to calculated the lift and the drag:

$$L = \frac{1}{2}\rho V_{rel}^2 c C_L \tag{3.1.23}$$

$$D = \frac{1}{2}\rho V_{rel}^2 c C_D \tag{3.1.24}$$

where C_L and C_D can be gathered from tabulated data.

The lift and drag is calculated in the same direction as the flow. However a better way of dividing the forces is in the normal and tangential direction, compared to the rotor plane. By calculating the forces as:

$$P_N = L\cos(\phi) + D\sin(\phi) \tag{3.1.25}$$

$$P_T = L\sin(\phi) - D\cos(\phi) \tag{3.1.26}$$

or doing it already for the coefficients.

$$C_N = C_L\cos(\phi) + C_D\sin(\phi) \tag{3.1.27}$$

$$C_T = C_L\sin(\phi) - C_D\cos(\phi) \tag{3.1.28}$$

they can therefore be expressed as this:

$$C_N = \frac{P_N}{\frac{1}{2}\rho V_{rel}^2 c} \tag{3.1.29}$$

$$C_T = \frac{P_T}{\frac{1}{2}\rho V_{rel}^2 c} \tag{3.1.30}$$

when using the formulation as above the thrust and moment can be calculated as:

$$dT = N_B P_N dr \tag{3.1.31}$$
$$dM = r N_B P_T dr \tag{3.1.32}$$

inserting these equation in to eqs. 2.4.19 and 2.4.20 the thrust can be calculated as:

$$dT = \frac{1}{2}\rho N_B \frac{V_0^2 \left(1-a\right)^2}{\sin^2(\phi)} c C_N dr \tag{3.1.33}$$
$$dM = \frac{1}{2}\rho N_B \frac{V_0 \left(1-a\right) \omega r \left(1+a'\right)}{\sin(\phi)\cos(\phi)} c C_T r dr \tag{3.1.34}$$

which means that the indical functions a and a' can be calculated as:

$$a = \frac{1}{\frac{4\sin^2(\phi)}{\sigma C_N} + 1} \tag{3.1.35}$$
$$a' = \frac{1}{\frac{4\sin(\phi)\cos(\phi)}{\sigma C_T} - 1} \tag{3.1.36}$$

where σ can be calculated as:

$$\sigma(r) = \frac{c(r) N_B}{2\pi r} \tag{3.1.37}$$

using these equations and following the numerical procedure seen in figure 3.1.2 the forces on the blade can easily be calculated.

The models

- Prant's Tip Loss factor

  Prantl's tip loss factor is needed because the actuator disk model assumes that there is a big disk that changes the direction and pressure of the flow whereas in reality there is just three blades that have that effect. To modify this the a variable is multiplied in the a and a' equations. So instead of the equations 3.1.35 and 3.1.36, with Prantl's tip loss factor the equations 3.1.38 and 3.1.39 should be used. [10]

  $$a = \frac{1}{\frac{4F\sin^2\phi}{\sigma C_n} - 1} \tag{3.1.38}$$

  $$a' = \frac{1}{\frac{4F\sin\phi\cos\phi}{\sigma C_n} - 1} \tag{3.1.39}$$

  where the variable F is calculated as:

  $$F = \frac{2}{\pi}\cos^{-1}(\exp(-f)) \tag{3.1.40}$$

f is a part in the so-called Glauert correction factor which can be calculated as:

$$f = \frac{NB}{2}\frac{R-r}{r\sin\phi} \tag{3.1.41}$$

- Glauert Correction Factor

  At high induction factors BEM stops working well, to solve this the Glauert correction factor is used. [10]

  The Glauert correction factor is based on empirical data of the thrust, and the change can be written as:

$$\begin{cases} C_T = 4a(1-a)F & \text{if} \quad a < a_c \\ C_T = 4(a_c^2 + (1-2a_c)a)F & \text{if} \quad a > a_c \end{cases}$$

  expressing this in terms of the variable a, it looks like this:

$$\begin{cases} a = \frac{1}{\frac{4F\sin^2\phi}{\sigma C_n}-1} & \text{if} \quad a < a_c \\ a = \frac{1}{2}[2 + K(1-2a_c) - \sqrt{(K(1-2a_c)+2)^2 + 4(K(a_c^2-1)]} & \text{if} \quad a > a_c \end{cases}$$

  where K is calculated as:

$$K = \frac{4F\sin^2\phi}{\sigma C_n} \tag{3.1.42}$$

Key assumptions for BEM

- Wind is steady, and normal to the rotor plane.
- A radial element of the blade is not affected by other close-by elements.
- C_L, C_D and C_M data are used from static measurements for different angle of attacks.

Numerical procedure

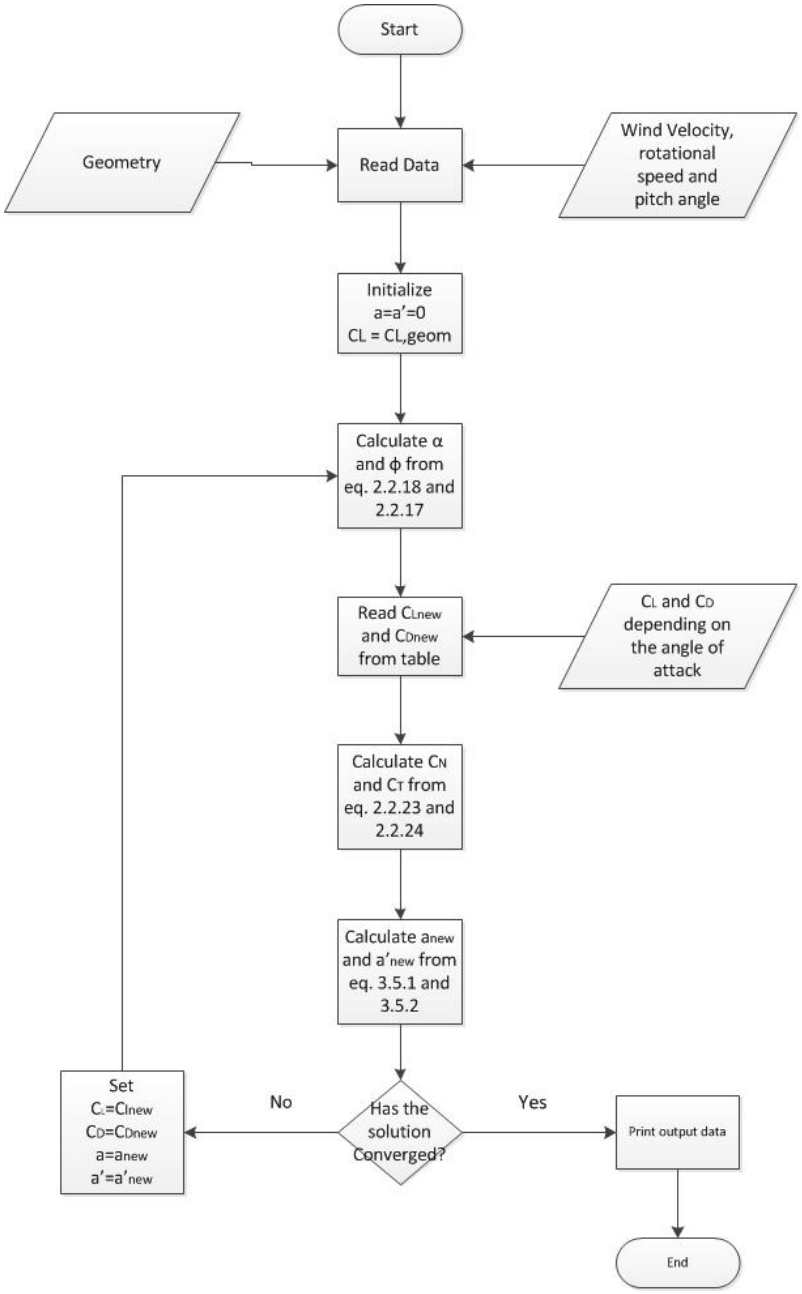


Figure 3.1.2: The Numerical Procedure when using BEM.

3.1.2 Vortex Methods

Introduction

The Vortex Methods are based on the Prandt's lifting line theory. There are a lot of different Vortex methods around. In this thesis two are focused on, these are the Helical Vortex Method (HVM) and the Free Vortex Method (FVM).

The Theory

For an infinite long thin blade the lift can be calculated by assuming that the blade is acting like a vortex, and therefore replace the geometry by a vortex line, called boundvortex, positioned at 25 % of the chord length from the leading edge, see figure 3.1.3. The strength of the vortex can be calculated by assuming that the lift generated by the blade and the vortex should be the same therefore using the Kutta–Joukowski theorem [2].

$$L = \rho \Gamma_\infty V_\infty \tag{3.1.43}$$

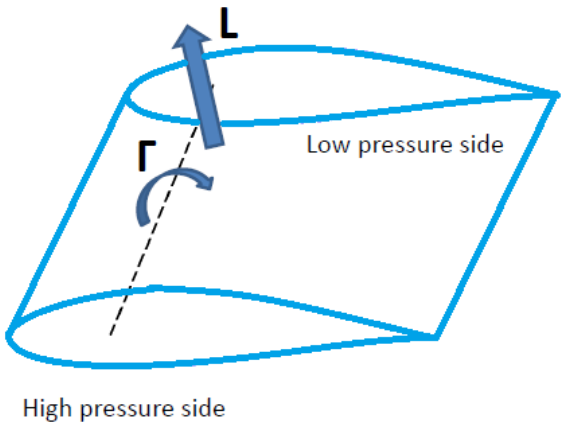


Figure 3.1.3: A blade and a bound vortex filament.

Blades generally are not infitely long however, and some tip-loss effects will occur at the ends of the blade. Since a blade, when generating lift, has a high pressure side, and a low pressure side, at the tips, the fluid at the high pressure side, wants to go to the high pressure side. This creates tip-vortices, and the tip-vortices and the bound vortex form a system of vortices called horse-shoe vortices, see 3.1.4. The strength of these vortices are as strong as that of the bound vortex.

In reality these vortices are released from all points of the blade, and not just the tip. To model this each a blade can be split in to sections of wings creating multiple systems of horse-shoe vortices, see 3.1.5.

However when the distance between the different blade sections becomes infinitely small, the strength of the wake-vortices will not be the same. The tip vortex of one blade section will rotate in a different direction from the next blade sections wake-vortex, they will if equally strong, cancel each other. In general for a wing, on for example an airplane, the bound vortex will not be equally strong over the entire length of the blade. Therefore the wake vortices will not cancel each other, they will just have the strength of the difference between the two different vortices. For

[2]The theory of the lifting line theory has been extracted from [11].

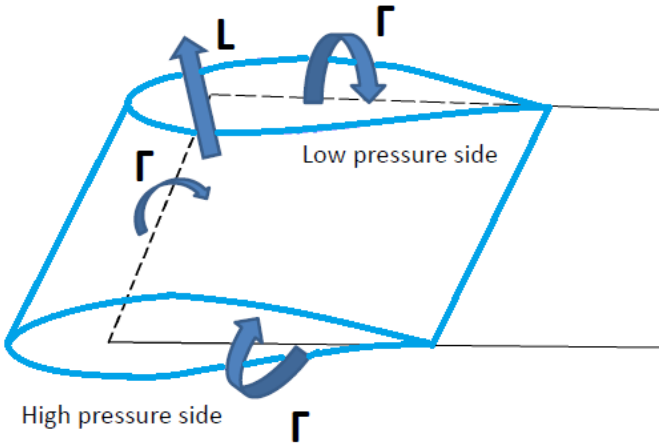


Figure 3.1.4: A finite blade with a horse shoe vortex.

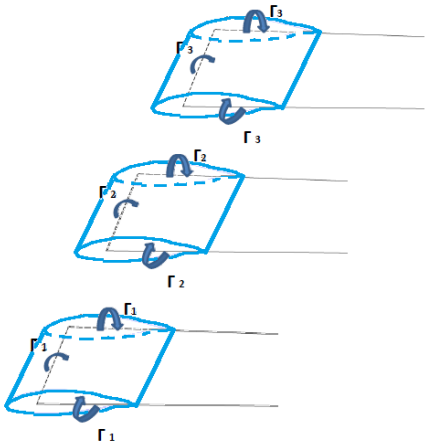


Figure 3.1.5: different segments of a blade and the horse shoe vortices they create.

a wing in straight flight, the distribution of Gammas can be approximated by a sine function and it can therefore look like in figure 3.1.6.

All these vortices, will create a flow, going downwards, called downwash. The speed of the velocity can be calculated through the Biot-Savart Law as:

$$dV = \frac{\Gamma}{4\pi} \frac{dlr}{|r|^3} \tag{3.1.44}$$

The induced velocity changes the angle of attack of the flow that the blade feels, which can be seen in figure 3.1.7

The change in angle of attack, creates a difference in the bound vortex strength, and therefore also the lift generated by the blade.

The Helical Vortex Method

In the Helical Vortex Method there are predefined positions where the so-caleld horse-shoe vortices are released. These vortex filaments are assumed to follow the free-stream velocity (which has to be normal to the rotating plane), and therefore forms a helix.

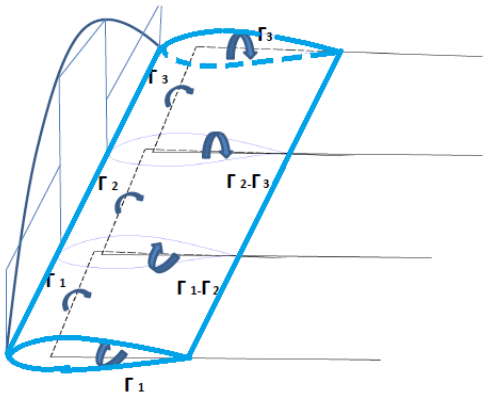


Figure 3.1.6: A blade and the sytem of horse shoe vortices.

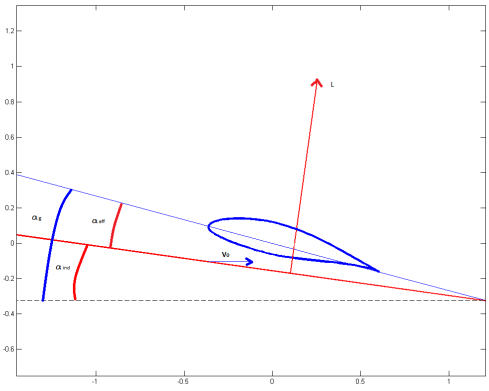


Figure 3.1.7: The angle of attack on an airfoil.

The vortex lines are described by points, and since the points form the shape of a helix their positions can be expressed as a simple helix equation.

This is the beauty of the Helical Vortex Method, since the positions of the vortices are known, the effect they have on the blade through the Biot-Savart Law is known.

The Helical Vortex method can be split in to two parts, the bound vortices and the wake vortices which can be seen in figure 3.1.8. The bound-vortices are located at 25 % of the chord length from the leading edge.[3]

The strength of the bound-vortex $\Gamma_{BoundVortex}$ is dependent on the angle of attack and can be calculated from the equation:

$$\Gamma_{BoundVortex} = \frac{L}{\rho V_\infty} \tag{3.1.45}$$

where the lift L is taken from tabulated data.

The wake-vortices are spawned from the boundary between one bound-vortex and another. The strength of the near-wake vortices depends on how strong the two

[3]All the equations below have been extracted from [12].

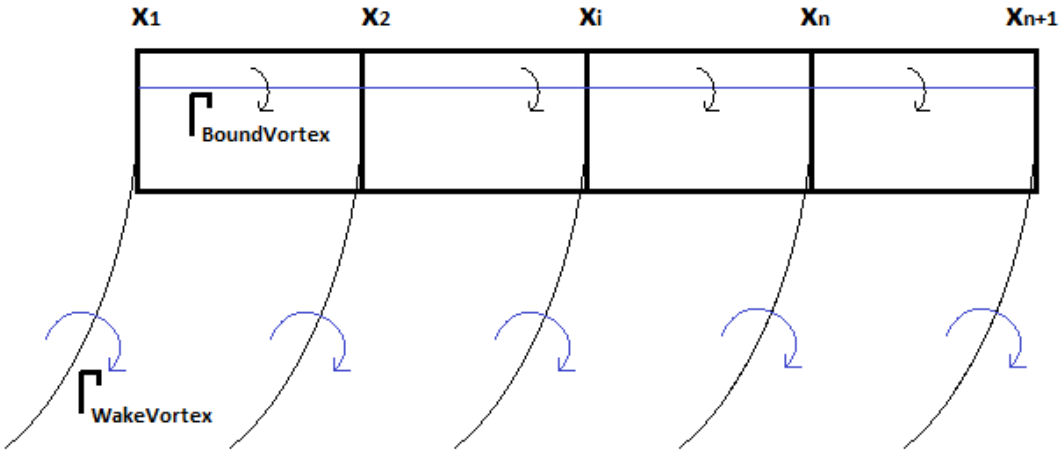


Figure 3.1.8: The vortex system for the Helical Vortex Method, with the Bound-Vortices and theWakeVortices.

closest bound-vortices are, i.e. $\Gamma_{WakeVortex}$ can be calculated as:

$$\Gamma^{i}_{WakeVortex} = \Gamma^{i}_{BoundVortex} - \Gamma^{i-1}_{BoundVortex} \tag{3.1.46}$$

the wake-vortices will, due to Biot - Savart Law induce a velocity on to the blade, which will change the angle of attack, $\alpha_{effective}$.

$$\alpha_{effective} = \alpha_{geometrical} - \alpha_{induced} \tag{3.1.47}$$

where $\alpha_{geometrical}$ is the angle of attack without any induced velocity, calculated as:

$$\alpha_{geometrical} = arctan\left(\frac{V_0}{x_i\Omega}\right) - \Theta_i \tag{3.1.48}$$

where Θ_i is the twist of the blade. And $\alpha_{induced}$ is calculated as:

$$\alpha_{induced}(i) = \frac{W_n(i)}{W_i} \tag{3.1.49}$$

where W_n is calculated as:

$$W_n(i) = \sqrt{(\overline{v}_i^2 + \overline{w}_i^2)} \tag{3.1.50}$$

and W can be calculated as:

$$W(i) = \sqrt{(V_0^2 + (x_i\Omega)^2)} \tag{3.1.51}$$

$\overline{v}_i$ and $\overline{w}_i$ is the sum of all induced velocities from the wake-vortices which can be calculated as:

$$\overline{v}_i = \sum_{j=1}^{NB} \overline{V}_{ij} \overline{\Gamma}_{ij} \tag{3.1.52}$$

$$\overline{w}_i = \sum_{j=1}^{NB} \overline{W}_{ij} \overline{\Gamma}_{ij} \tag{3.1.53}$$

where the $\overline{\Gamma}_{ij}$ is the different $\Gamma_{WakeVortex}$ calculated earlier and the $\overline{V}_{ij}$ and $\overline{V}_{ij}$ should be calculated as:

$$\overline{V}_{ij} = \frac{1}{4\pi} \int_0^{\inf} (v_{i,j+1} - v_{i,j}) d\theta \tag{3.1.54}$$

$$\overline{W}_{ij} = \frac{1}{4\pi} \int_0^{\inf} (w_{i,j+1} - w_{i,j}) d\theta \tag{3.1.55}$$

where v_{ij} and w_{ij} are calculated as:

$$v_{ij} = \sum_{n=1}^{NB} \frac{h[-\eta(\cos\theta' + \theta \sin\theta') + r]}{[h^2\theta^2 + \eta^2 + r^2 - 2r\eta\cos\theta']^{\frac{3}{2}}} \tag{3.1.56}$$

$$w_{ij} = \sum_{n=1}^{NB} \frac{\eta^2 - r\eta\cos\theta'}{[h^2\theta^2 + \eta^2 + r^2 - 2r\eta\cos\theta']^{\frac{3}{2}}} \tag{3.1.57}$$

and the variables η, r θ' and h can be calculated as:

$$\eta = \frac{x_j}{R} \tag{3.1.58}$$

$$r = \frac{x_i}{R} \tag{3.1.59}$$

$$\theta' = \theta + \frac{2\pi(n-1)}{NB} \tag{3.1.60}$$

$$h = \frac{V_0 - w_j}{R\left(\omega\left(\frac{v_j}{x_j}\right)\right)} \tag{3.1.61}$$

When a new $\alpha_{effective}$ has been calculated, a new lift coefficient can be extracted from the tabulated data. This means that a new $\Gamma_{BoundVortex}$ can be calculated. This will change the $\Gamma_{WakeVortex}$ and therefore also the induced velocities on the blade, which in turn changes the $\alpha_{effective}$ again. After some iterations the strength of $\Gamma_{BoundVortex}$ will no longer change, and the solution has then converged.

Key assumptions for HVM

- Wind is steady, and normal to the rotor plane.
- A lifting line represents the blade at 25 % of the chord length, also called the bound vortex.
- From each blade section a horse-shoe vortex is created, where the vortex filaments form perfect helices.
- C_L, C_D and C_M data are used from static measurements for different angle of attacks.

Numerical procedure for HVM

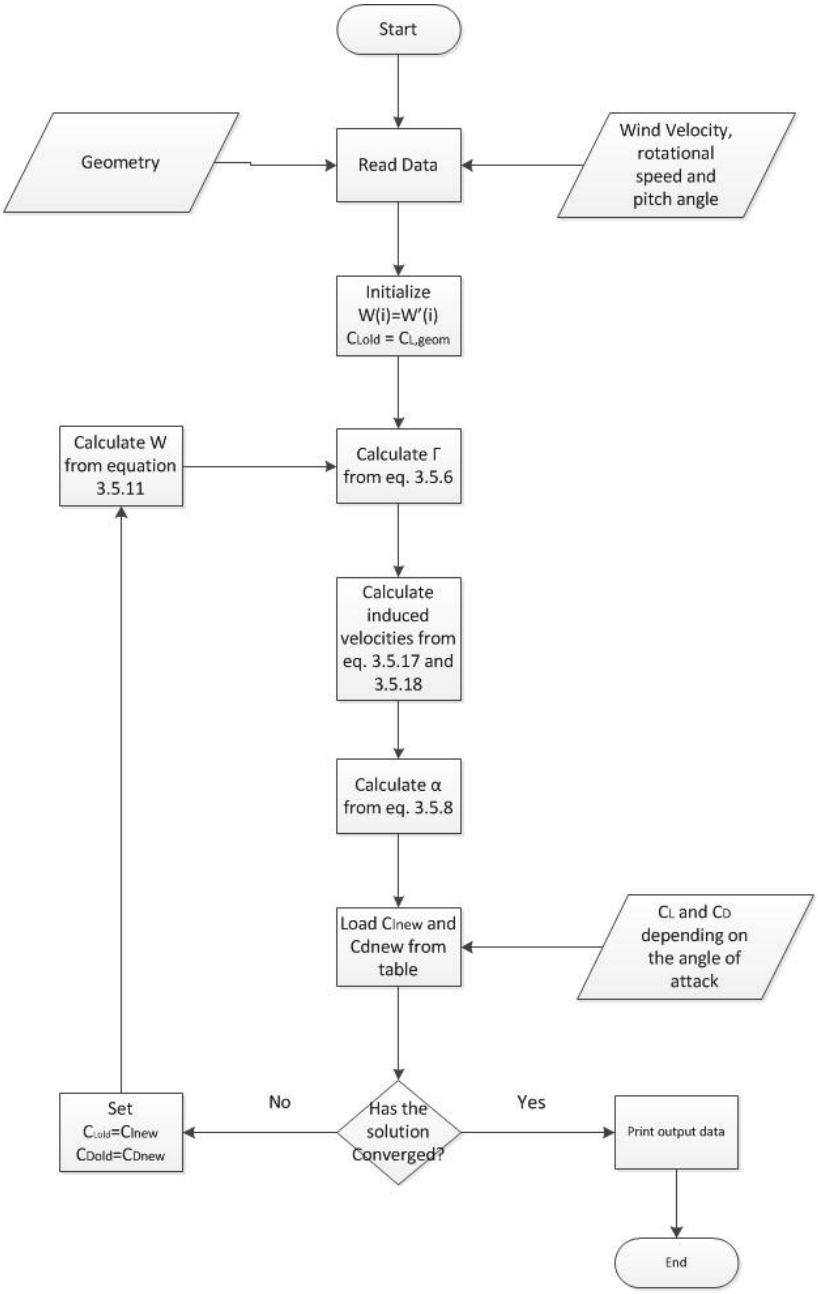


Figure 3.1.9: The Numerical Procedure when using Helical Vortex Method.

The Free Vortex Method

The Free Vortex Method used here is based on a code written by Bagai and Leishman [13] and [14] which is used for helicopter applications. A PhD student Hamidreza Abedi at Chalmers University of Technology is currently tuning the code to work for wind turbine applications. However, in writing moment, the tuning is not done and therefore the results will may be inaccurate.

Similarly to the Helical Vortex Method, the Free Vortex Method (FVM) is based on the lifting line theory. However due to the fact that the wind, in reality, never is steady or have the same wind speed over a big area, the assumption that is made for HVM (the wake forms a perfect helix) is not true. Because of this the Free Vortex Method was created. The difference is then that in the FVM the position of all the distributed vortex points are calculated from the induced velocity on each point, as well as the free stream velocity. Whereas for the HVM, the points form a perfect helix, based on the free stream velocity.

In essence, the position of each vortex point is calculated from the free stream velocity, the induced velocity of the blade and the induced velocity from all other vortex filaments. The upside of this method is the fact that the wake is more accurately predicted, and a more complex flow situation can be modelled.

The downside is the increase in calculation time. To reduce the computational time a model has been created by Bagai and Leishman, where the wake is divided in a near wake and a far wake. The near wake follows the helix equation and is very short. The far-wake is then modelled as a free wake, but with only one filament that represent all other vortex filaments.

In short, the benefits of FVM are:

- Faster than CFD.
- Pretty accurate.
- Can calculate forces on blades with pretty much any form or shape.

The negative sides are:

- Slower than BEM.
- Engineering models needed to make it computationally effective.

The Model

As previously mentioned, the Free Vortex Method can be split in to three different vortices, these vortices can be seen in figure 3.1.10.

The vortex that replaces the blade is called the bound-vortex, and is defined from the geometry of the blade, where it is positioned at 25 % of the chord length from the leading edge. From each bound-vortex a near-wake vortex is spawned. The third and last type of vortex is the far wake, which at the start is defined as having the shape of a helix similar to the helical vortex method.

The strength of the bound-vortex $\Gamma_{BoundVortex}$ is dependent on the angle of attack and can be calculated from the equation:

$$\Gamma_{BoundVortex} = \rho L V_{\infty} \tag{3.1.62}$$

where the Lift L is taken from tabulated data.

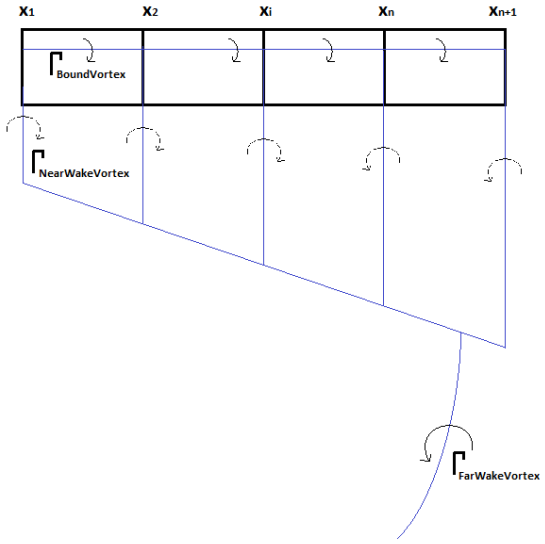


Figure 3.1.10: The vortex system for Free Vortex Method, with the BoundVortices, NearWakeVortices and the FarWakeVortex.

The strength of the near-wake vortices depends on how strong the two closest bound-vortices are, i.e. $\Gamma_{NearWakeVortex}$ can be calculated as:

$$\Gamma^i_{NearWakeVortex} = \Gamma^i_{BoundVortex} - \Gamma^{i-1}_{BoundVortex} \tag{3.1.63}$$

where i is the different vortices.

The strength of the far-wake vortex is in this model assumed to be the maximum strength of $\Gamma_{BoundVortex}$, i.e:

$$\Gamma_{FarWakeVortex} = max\{\Gamma_{BoundVortex}\} \tag{3.1.64}$$

when all this is known, the real program starts and the updating of the far-wake starts. This is done in two steps, a predictor step, and a corrector step.

First of all the induced velocity by all vortices are calculated using the Biot-Savart Law which can be seen below:

$$V_{ind}(r(\psi,\zeta), r(\psi_j,\zeta)) = \frac{h^2}{4\pi\sqrt{r_c^4 + h^4}} \int \frac{\Gamma(\psi_j,\zeta)d\zeta_j \times (r(\psi,\zeta) - r(\psi_j,\zeta))}{|r(\psi,\zeta) - r(\psi_j,\zeta)|^3} \tag{3.1.65}$$

this is done for the V_{ind} induced by the bound-vortex, the V_{ind} induced by the near-wake and also the V_{ind} induced by the other far-wake vortices.

When all the induced velocities are known the far-wake is moved using the equation below. This is called the predictor step, and it is assumed that all vortices are located where they were the last time step.

$$\overline{r}^n_{j,k} = \overline{r}^n_{j-1,k-1} + \left(\frac{\Delta\psi - \Delta\zeta}{\Delta\psi + \Delta\zeta}\right) \cdot \left(\overline{r}^n_{j,k-1} - \overline{r}^n_{j-1,k}\right) + \frac{\Delta\psi \cdot \Delta\zeta}{\Delta\psi + \Delta\zeta}\frac{2}{\Omega}\left(V_\infty + \frac{1}{4}\left(\sum_{i=1}^{N_v} V_{ind}\left(r^{n-1}_{j-1,k-1}, r^{n-1}_{i,\psi}\right) + \sum_{i=1}^{N_v} V_{ind}\left(r^{n-1}_{j-1,k}, r^{n-1}_{i,\psi}\right) + \sum_{i=1}^{N_v} V_{ind}\left(r^{n-1}_{j,k-1}, r^{n-1}_{i,\psi}\right) + \sum_{i=1}^{N_v} V_{ind}\left(r^{n-1}_{j,k}, r^{n-1}_{i,\psi}\right)\right)\right) \quad (3.1.66)$$

when the new position of the far-wake is known, the induced velocities are re-calculated for the new positions, using the same Biot-Savart Law as before.

Before the corrector step is run, a mean induced velocity is calculated, using the induced velocities from before the predictor step, as well as the ones after, using the equations below:

$$\overline{V}_{ind}\left(r_{j,k}, r_{i,\zeta}\right) = \frac{1}{2}[V_{ind}\left(r^{n-1}_{j,k}, r^{n-1}_{i,\zeta}\right) + V_{ind}\left(r^{n}_{j,k}, r^{n}_{i,\zeta}\right)] \quad (3.1.67)$$

$$\overline{V}_{ind}\left(r_{j-1,k-1}, r_{i,\zeta}\right) = \frac{1}{2}[V_{ind}\left(r^{n-1}_{j-1,k-1}, r^{n-1}_{i,\zeta}\right) + V_{ind}\left(r^{n}_{j-1,k-1}, r^{n}_{i,\zeta}\right)] \quad (3.1.68)$$

$$\overline{V}_{ind}\left(r_{j-1,k}, r_{i,\zeta}\right) = \frac{1}{2}[V_{ind}\left(r^{n-1}_{j-1,k}, r^{n-1}_{i,\zeta}\right) + V_{ind}\left(r^{n}_{j-1,k}, r^{n}_{i,\zeta}\right)] \quad (3.1.69)$$

$$\overline{V}_{ind}\left(r_{j,k-1}, r_{i,\zeta}\right) = \frac{1}{2}[V_{ind}\left(r^{n-1}_{j,k-1}, r^{n-1}_{i,\zeta}\right) + V_{ind}\left(r^{n}_{j,k-1}, r^{n}_{i,\zeta}\right)] \quad (3.1.70)$$

when these velocities have been calculated the corrector step is run, using an equation very similar to the predictor step, the new position of the far-wake is calculated.

$$r^n_{j,k} = r^n_{j-1,k-1} + \left(\frac{\Delta\psi - \Delta\zeta}{\Delta\psi + \Delta\zeta}\right) \cdot \left(\overline{r}^n_{j,k-1} - \overline{r}^n_{j-1,k}\right) + \frac{\Delta\psi \cdot \Delta\zeta}{\Delta\psi + \Delta\zeta}\frac{2}{\Omega}\left(V_\infty + \frac{1}{4}\left(\sum_{i=1}^{N_v} \overline{V}_{ind}\left(r_{j-1,k-1}, r_{i,\zeta}\right) + \sum_{i=1}^{N_v} \overline{V}_{ind}\left(r_{j-1,k}, r_{i,\zeta}\right) + \sum_{i=1}^{N_v} \overline{V}_{ind}\left(r_{j,k-1}, r_{i,\zeta}\right) + \sum_{i=1}^{N_v} \overline{V}_{ind}\left(r_{j,k}, r_{i,\zeta}\right)\right)\right) \quad (3.1.71)$$

when the new positions are updated, new induced velocities are calculated, but this time the velocity calculated is the velocity on the control points on the blade.

The Biot-Savart Law is used yet again, however now it is only the effect of the far-wake and near-wake that is taken in to account.

With the induced velocities at the blade calculated a new angle of attack can be calculated, α_E. α_E can simply be calculated as the velocity in the normal direction,

divided by the velocity in the tangential direction.

$$\alpha_E = arctan\left(\frac{U_{Normal}}{U_{Tangetial}}\right) \tag{3.1.72}$$

With a new angle of attack, the lift force can be updated from the table, changing all the vortex strengths Γ, with new strengths of the vortices the far-wake can be moved again. This procedure is repeated until the far-wake does not move anymore, and the forces are pretty much constant from one iteration to another.

Key assumptions for FVM

The strength of the far-wake is calculated as the max of the boundvortex strength.

Numerical Procedure for FVM

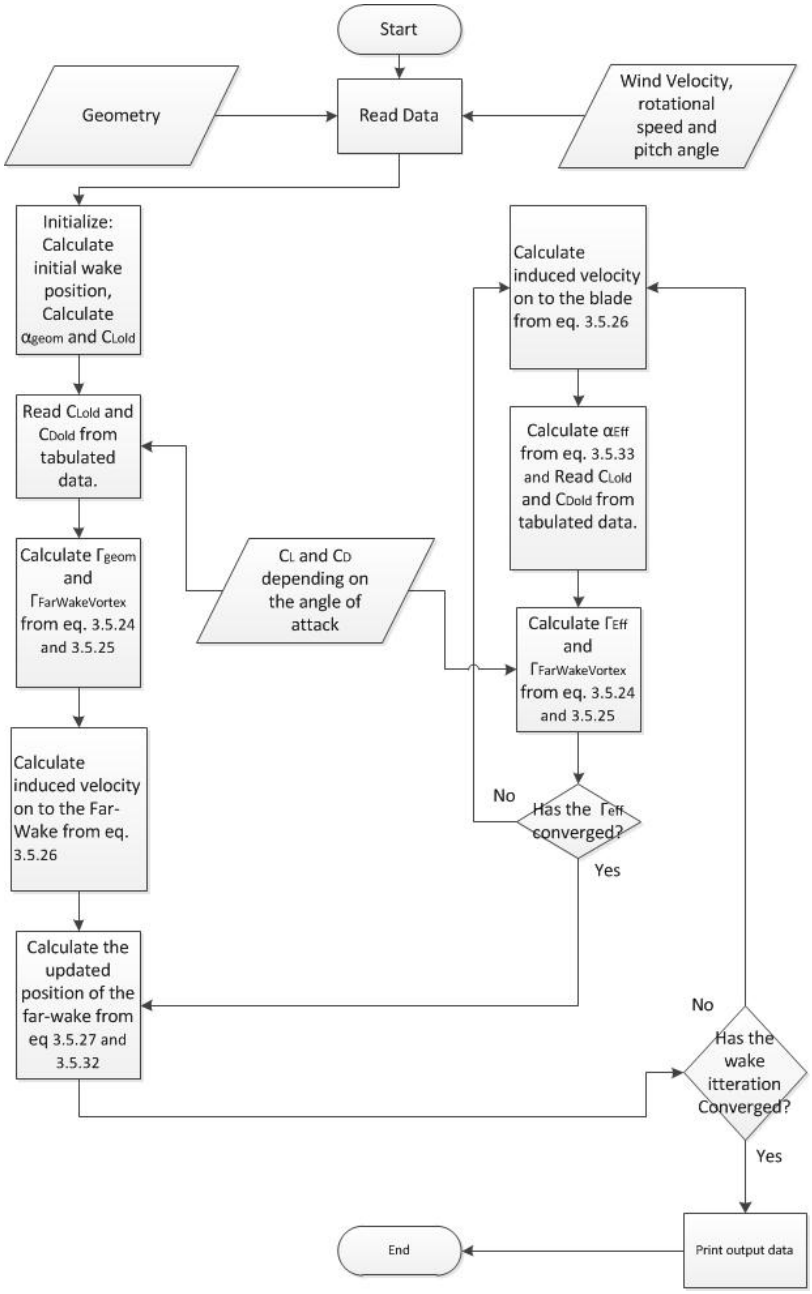


Figure 3.1.11: The Numerical Procedure when using Free Vortex Method.

3.2 Method for applying the aerodynamic forces

Since the blade calculation of the structural part is a FEM model i.e. 3-D, but the vortex methods/BEM method only give the force components on a section, some sort of engineering model is needed. Therefore to transform the 1-D data from the VM/BEM calculations to 3-D data, the OpenSource code XFoil is used. By matching the C_L, C_D and C_M data a pressure distribution can be calculated.

3.2.1 Pressure Distribution

To calculate the pressure distribution using XFoil, three different methods are used. The first and the quickest method is to calculate a lot of different pressure distributions for the different airfoils that is going to be used, before starting the simulations. Using this method around 100 different pressure distributions are calculated for each angel of attack, where the Reynolds number is changed for each case.

However, for some cases the C_L, C_D and C_M data that is used in BEM (from measurements) does not match any of the tabulated data, for these cases the tabulated data can not be used and therefore two other methods can be used, depending on the angle of attack.

By plotting the different C_L's for different angel of attacks one can see that the C_L curve can be divided in to two different sections, a roughly linearly depended part, and a non linear part, see figure 3.2.1.

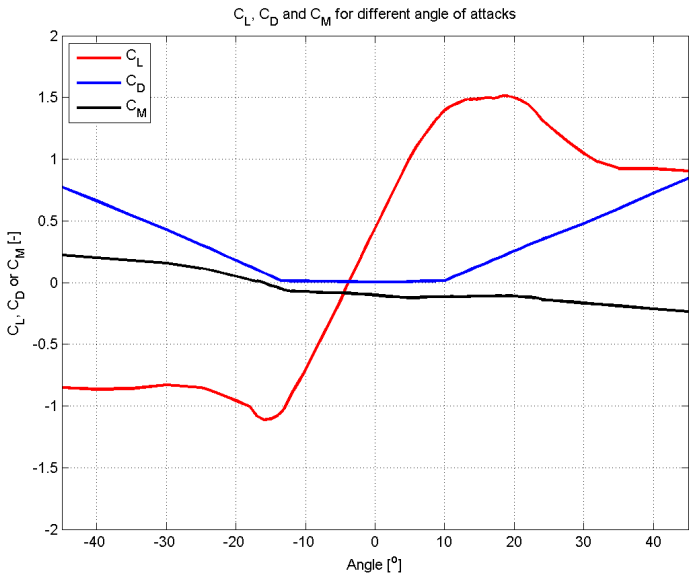


Figure 3.2.1: C_L and C_D plotted against different angle of attacks.

The linear part appears at low angle of attacks (angle of attacks smaller than 11). This is important because in the linear section a rather easy method for calculating the pressure distribution can be utilized. Whereas for higher angle of attacks a more complicated and less accurate method needs to be used.

More detailed information can be found in the sub-chapters below.

Tabulated Pressure fields

The tabulated data is as previously mentioned stored for different angle of attacks. For a given angle of attack. The pressure distribution is calculated in XFoil for different Reynolds numbers, by changing the Reynolds number the C_L, C_D and C_M will change. How C_L, C_D and C_M changes for the angle of attack of 7^o can be seen in figure 3.2.2.

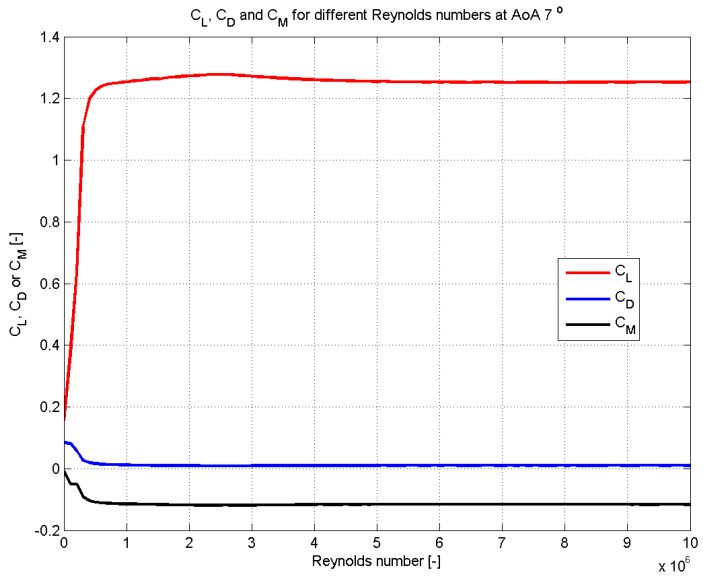


Figure 3.2.2: How the C_L, C_D and C_M changes with different Reynolds numbers for the angle of attack $\alpha = 7\ ^o$.

All these values, the C_L, C_D, C_M and the pressure distribution are stored in .mat files in a specific folder. When the fluid code is run a function will then look at the effective angle of attack and try to find two pressure distribution that has the smallest difference between the tabulated $\frac{C_L}{C_D}$ value and the $\frac{C_L}{C_D}$ thats calculated from the fluid code.

Sometimes this does not work however. For example as seen in figure 3.2.2 the airfoil has a maximum $\frac{C_L}{C_D}$ value whereas the $\frac{C_L}{C_D}$ value calculated from the fluid code can be higher than this. In that case the code would not work and an iterative method needs to be used.

Small angle of attacks

For small angle of attacks (roughly in the linear part of the C_L curve) XFoil has a function that lets the user specify the C_L value which then calculates the pressure distribution and the C_D and C_M. By inputting the C_L value received from the fluid code, the pressure can be iterated, changing the the Reynolds number to get a C_D that matches the one calculated by the fluid code.

High angle of attacks

When the angle of attacks are so high that the method used for small angle of attacks does not work, the angle of attack is given as an input to XFoil. By changing the Reynolds number a $\frac{C_L}{C_D}$ can be found that matches the $\frac{C_L}{C_D}$ thats calculated from the fluid code.

Numerical Procedure for calculating the pressure distribution

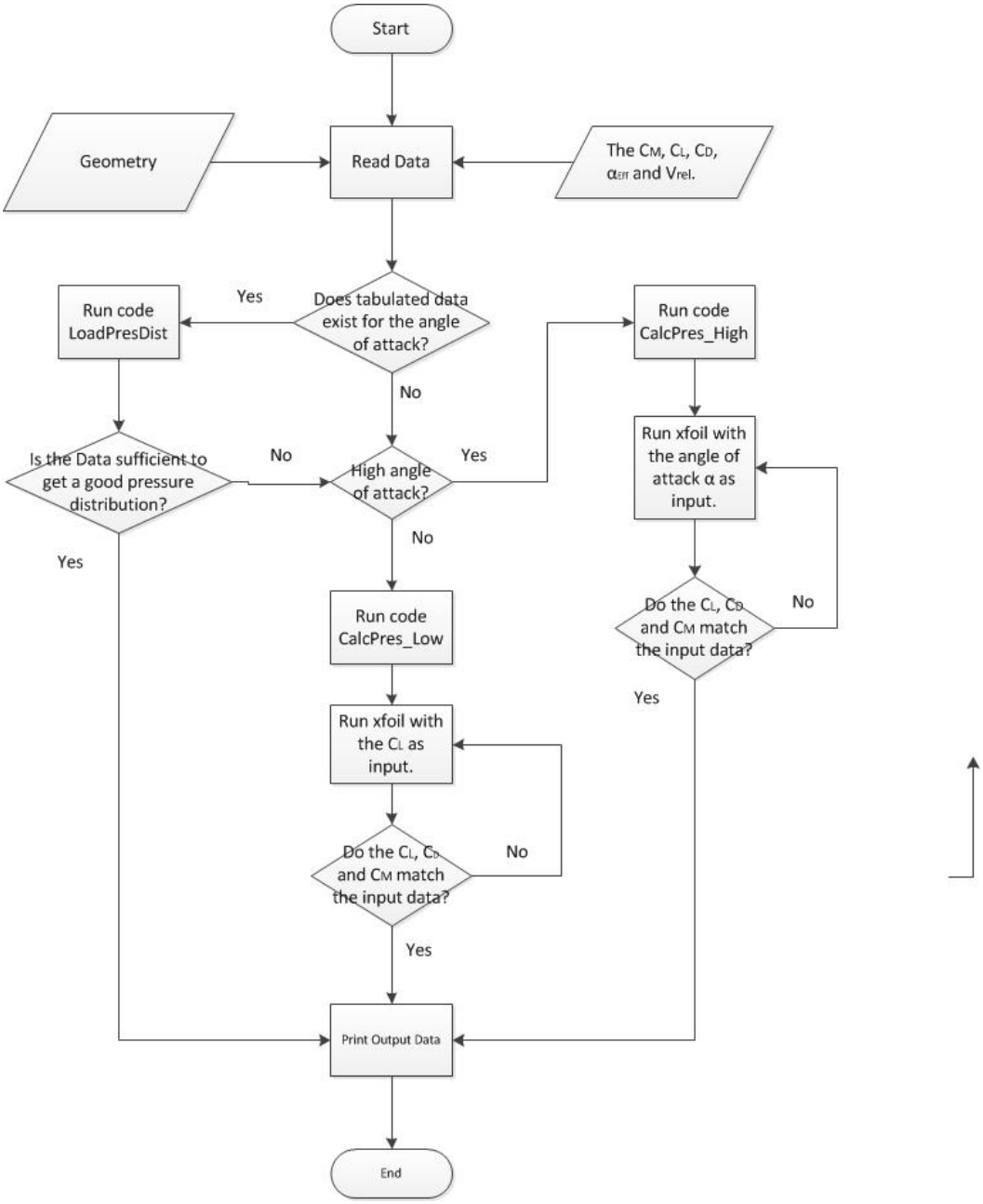


Figure 3.2.3: The Numerical Procedure when calculating the pressure distribtuin.

3.2.2 XFoil

Introduction

Xfoil is a program based on he 2D-panel method and has been created by Mark Drela from MIT [15].

The program can be run either from a text file with commands in it, or "live" where the user inputs commands on the fly. XFoil in itself has a lot of different functions, and how it works and more in-depth of the theory can be read in [15]. In figure 3.2.4 a graphical result from XFoil is shown.

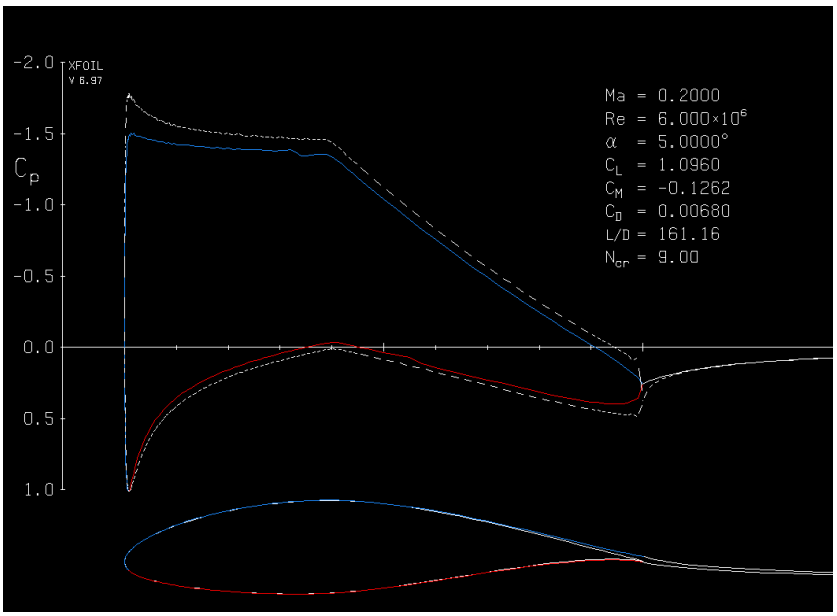


Figure 3.2.4: In the figure a plot generated by XFoil can be seen.

Brief Theory Description

This report will not cover much of the panel method, because it is not used much for the thesis. However to get a complete picture it will be introduced. For more information [16] is a good start.

The panel method is pretty similar to the lifting line theory. The main difference is the fact that instead of a line, the geometry is replaced by several square panels. On all the panels, at 25 percent of the chord length (of the panel), a lifting line is creating a lift force (similar to lifting line theory). From the panel, other wake panels are spawned, these panels model the wake, and follows the induced plus the free stream velocity.

How XFoil is used

In this thesis, XFoil is used to calculate the pressure distribution for a given angle of attack and airfoil. To do this a Matlab script creates a text file with the commands that XFoil should run. Typical inputs for XFoil is either the lift coefficient C_L or the angle of attack α. Other inputs are the profile coordinates for the specified airfoil, along with the Mach number the airfoil is travelling with, as well as the Reynolds number.

With the data described above XFoil starts to do some calculations, calculating the pressure distribution and the C_L, C_D and C_M that the pressure creates. This is saved in two text files, the pressure distribution in one file, called PressureBlade1.txt, and the C_L, C_D and C_M in another file, called Polar1.txt.

A typical command file created by the Matlab script can be seen in the appendix A 2.

3.3 Two-way Fluid Structure Interaction

With one-way coupling the fluid forces are calculated and then applied to the structure. When switching to two-way coupling the forces on the structure creates a deformation, which at the same time changes the forces that is applied by the fluid. To create two-way coupling with these methods, the only thing needed is to calculate the forces, calculate the deformation, update the geometry, calculate new forces, and then iterate this until the residual becomes small (deformation does not change).

Going from one-way to two-way is quite simple. All that is needed is a way to update the geometry. This is done in two steps, since the structural model is a 3d FEM model there is a lot of data of how the blade moves. However the fluid only uses one point per section.

The first thing that is needed is to calculate how each section of the blade behaves. This is done by the least square fitting method. More can be read in the chapter below.

The second thing that needs to be done is to update the geometry for the fluid forces calculator. Since the geometry is defined slightly different depending on what solver is used there will be some small differences.

For Blade Element Momentum (BEM)

BEM uses very few parameters to define the geometry therefore BEM probably will not give extremely good results for a deformed blade. The only parameters that can be changed is the radial position of the section, and the twist angle (which is the most important parameter).

For Helical Vortex Method (HVM)

For Helical Vortex Method the deformation in the flap-wise and edge-wise direction is taken in to account for the as well as the twist of the blade, and the deformation in the radial direcion.

However, when taking into account the deformation, the equations presented in chapter 3.1.2 has to be modified slightly. This is because the Helical Vortex Method presented in this thesis do not have the possibility to change the geometry in the flap-wise and edge-wise directions.

To be able to take this is to account, the influence coefficient equations needs to be modified in the following way:

Instead of:

$$v_{ij} = \sum_{n=1}^{NB} \frac{h[-\eta(\cos\theta' + \theta\sin\theta') + r]}{[h^2\theta^2 + \eta^2 + r^2 - 2r\eta\cos\theta']^{\frac{3}{2}}} \tag{3.3.1}$$

$$w_{ij} = \sum_{n=1}^{NB} \frac{\eta^2 - r\eta\cos\theta'}{[h^2\theta^2 + \eta^2 + r^2 - 2r\eta\cos\theta']^{\frac{3}{2}}} \tag{3.3.2}$$

The equations should look like this:

$$v_{ij} = \sum_{n=1}^{NB} \frac{h[-\eta \cdot (\cos(\theta' + \eta_e) + (\theta + \eta_f) \cdot \sin(\theta' + \eta_e)) + r + r_f]}{[h^2 \cdot (\theta + \eta_f)^2 + \eta^2 + r^2 + r_f^2 + r_e^2 - 2\eta(r\cos(\theta' + \eta_e) + r_e \sin(\theta' + \eta_e)) - 2r_f h \cdot (\theta + \eta_f)]^{\frac{3}{2}}} \tag{3.3.3}$$

$$w_{ij} = \sum_{n=1}^{NB} \frac{\eta^2 - r\eta\cos(\theta' + \eta_e) - r_e\eta\sin(\theta' + \eta_e)}{[h^2 \cdot (\theta + \eta_f)^2 + \eta^2 + r^2 + r_f^2 + r_e^2 - 2\eta(r\cos(\theta' + \eta_e) + r_e \sin(\theta' + \eta_e)) - 2r_f h \cdot (\theta + \eta_f)]^{\frac{3}{2}}} \tag{3.3.4}$$

Where r_f is the distance the control point moves in flap-wise direction, η_f is the distance the vortices move in the flap-wise direction. Similarly for r_e and η_e but in the edge-wise direction. All the distances are normalized by dividing them with the tip radius R_B.

By changing the equations in this way, it is possible to use the Helical Vortex Method to calculate the forces on blade which are not perfectly straight. So if one wants to calculate the forces on a blade thats pre-bent, or curved, then these equations should be used as well.

For Free Vortex Method (FVM)

FVM works in a similar way to the the Helical Vortex Method, i.e. it is possible to calculated the forces on blades which are either curved or bent. The parameters thats used are the radial position, the edge-wise position, the flap-wise position aswell as the twist angle for all sections.

Iterating

With the new geometry the fluid code can be run again and will create a different force compared to the first iteration, after around 5-12 iterations there will not be much, if any difference in the force, between the current iteration and the previous one, the solution has then converged and the two-way fluid structure interaction calculation is complete.

3.3.1 Derivation of equivalent sectional displacements

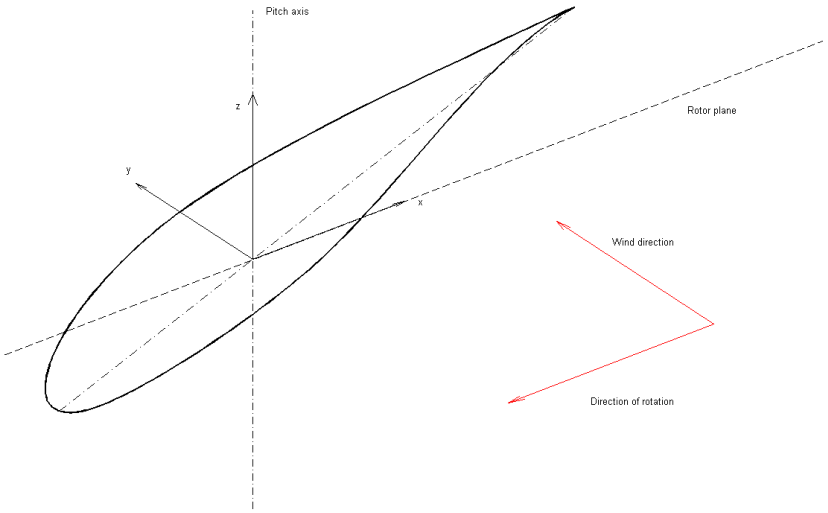


Figure 3.3.1: Definition of cordinate system used throughout this thesis

The equations for the aerodynamic coefficients depend on sectional displacements such as pitch angle, plunge velocity etc, but when using an FE - model, there are no direct quantities that corresponds to these. In this paper, in order to approximate these, we have used a least square fit. We choose k nodes from the i:th section, where node j has the coordinates $(x^{i,j}, y^{i,j})$, the elastic center of the section i is at (x_e^i, y_e^i) and the aerodynamic center of section i is at (x_c^i, y_c^i). See figure 3.3.1 for a definition of the coordinate system. If we write the equation for the nodal displacements as if they deform as a rigid section, we get an overdetermined system of equations for the equivalent sectional displacements:

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & -(y^{i,1}-y_c^i) \\ 0 & 1 & 0 & 0 & 0 & (x^{i,1}-x_c^i) \\ 0 & 0 & 1 & y^{i,1}-y_e^i & -x^{i,1}+x_e^i & 0 \\ 1 & 0 & 0 & 0 & 0 & -(y^{i,2}-y_c^i) \\ 0 & 1 & 0 & 0 & 0 & (x^{i,2}-x_c^i) \\ 0 & 0 & 1 & y^{i,2}-y_e^i & -x^{i,2}+x_e^i & 0 \\ & & & & \dots & \\ 1 & 0 & 0 & 0 & 0 & -(y^{i,k}-y_c^i) \\ 0 & 1 & 0 & 0 & 0 & (x^{i,k}-x_c^i) \\ 0 & 0 & 1 & y^{i,k}-y_e^i & -x^{i,k}+x_e^i & 0 \\ & & & & & \end{bmatrix} \begin{bmatrix} \hat{u}_x^i \\ \hat{u}_y^i \\ \hat{u}_z^i \\ \hat{\theta}_x^i \\ \hat{\theta}_y^i \\ \hat{\theta}_z^i \end{bmatrix} = \begin{bmatrix} u_x^{i,1} \\ u_y^{i,1} \\ u_z^{i,1} \\ u_x^{i,2} \\ u_y^{i,2} \\ u_z^{i,2} \\ \dots \\ u_x^{i,k} \\ u_y^{i,k} \\ u_z^{i,k} \end{bmatrix} \tag{3.3.5}$$

Or

$$\mathbf{T}_i \hat{\mathbf{u}}_i = \mathbf{u}_i \tag{3.3.6}$$

By a least square fit, the equivalent sectional displacements become:

$$\hat{\mathbf{u}}_i = (\mathbf{T}_i^T \mathbf{T}_i)^{-1} \mathbf{T}_i^T \mathbf{u}_i = \mathbf{T}^+{}_i \mathbf{u} \tag{3.3.7}$$

Doing this for all sections we get an equation for the full system:

$$\begin{bmatrix} \hat{\mathbf{u}}^1 \\ \hat{\mathbf{u}}^2 \\ \dots \\ \hat{\mathbf{u}}^N \end{bmatrix} = \begin{bmatrix} \mathbf{T}_1^+ & \mathbf{0} & \dots & \mathbf{0} \\ \mathbf{0} & \mathbf{T}_2^+ & & \mathbf{0} \\ & \dots & & \\ \mathbf{0} & \mathbf{0} & \dots & \mathbf{T}_N^+ \end{bmatrix} \mathbf{u} \tag{3.3.8}$$

$$\hat{\mathbf{u}} = \mathbf{T}^+ \mathbf{u} \tag{3.3.9}$$

Where $\hat{\mathbf{u}}$ holds the sectional displacements of the blade. The same procedure of course applies to respective velocities. As an example, the out-of-plane displacements under gravity is shown in figure 3.3.2.

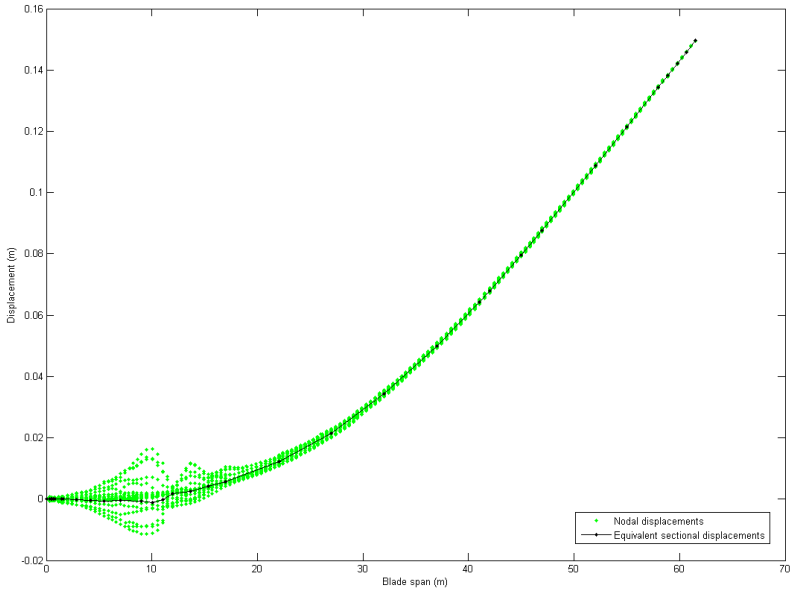


Figure 3.3.2: Static flap-wise displacement under gravity when the blade is in the horizontal postion. Each point represents a node, and the black line is the sectional displacements calculated by equation (3.3.9)

4 Aeroelasticity

4.1 Aerodynamic stability

Under certain conditions, the equilibrium of an elastic structure subjected to aerodynamic forces can become unstable. When this happens, the vibrations become self-feeding and destructive. Physically, an instability implies that the elastic, inertial, and aerodynamic forces line up so that a mode of the structure gains energy from the surrounding airstream. There are different types of instabilities that can occur, two being stall induced vibrations and classical flutter. Stall induced vibrations, or stall flutter happens at stalled conditions, i.e. for blunt objects, or, in our case for high angles of attack. Classical flutter always involves more than one degree of freedom [17], and typically involves strongly coupled torsion and bending. Aerodynamic stability has, with some exceptions, not been a large issue for wind turbines. An exception was in 1997 when several 19 m blades by LM Glasfiber suffered cracks near the trailing edge by the root due to large edgewise oscillations.[18] Classical flutter has not been seen in wind turbine blades, though the increasing size of turbine blades and strong bending-torsion coupling can increase the risk for flutter.

The stability problem is adressed in the following way. For given operating conditions, let the the state vector $\mathbf{z}$ be:

$$\mathbf{z}(t) = \mathbf{z}_0 + \mathbf{z}^*(t) \tag{4.1.1}$$

Where the constant $\mathbf{z}_0$ is the state at equilibrium and $\mathbf{z}^*(t)$ a small pertubation. Lets assume that the aerodynamic force $\mathbf{f}$ is dependent only on the state vector $\mathbf{z}$, and linearize the force around $\mathbf{z}_0$:

$$\mathbf{f} = \mathbf{f}_0 + \frac{\partial \mathbf{f}}{\partial \mathbf{z}}_{|\mathbf{z}=\mathbf{z}_0}(\mathbf{z} - \mathbf{z}_0) = \mathbf{f}_0 + \frac{\partial \mathbf{f}}{\partial \mathbf{z}}_{|\mathbf{z}=\mathbf{z}_0}\mathbf{z}^* \tag{4.1.2}$$

where, if $\mathbf{f}$ is a $nx1$ vector, and we have m states, then $\frac{\partial \mathbf{f}}{\partial \mathbf{z}}_{|\mathbf{z}=\mathbf{z}_0}$ is a nxm matrix. Introduce

$$\mathbf{A}_f = \begin{bmatrix} \mathbf{0} \\ \mathbf{Q}^T \\ \mathbf{0} \end{bmatrix} \frac{\partial \mathbf{f}}{\partial \mathbf{z}}_{|\mathbf{z}=\mathbf{z}_0} \tag{4.1.3}$$

then, as $\dot{\mathbf{z}} = \dot{\mathbf{z}}^*$, equation (4.1.1) can be written:

$$\dot{\mathbf{z}}^* = (\mathbf{A} + \mathbf{A_f})\mathbf{z}^* + \mathbf{B}\mathbf{f}_0 = \tilde{\mathbf{A}}\mathbf{z}^* + \mathbf{B}\mathbf{f}_0 \tag{4.1.4}$$

where we introduced $\tilde{\mathbf{A}} = (\mathbf{A} + \mathbf{A_f})$. Ignoring the constant force $\mathbf{f}_0$, and using the ansatz $\mathbf{z}^* = \hat{\mathbf{z}}^* e^{\lambda t}$ we get the eigenvalue problem for the eigenvalues λ and the eigenvectors $\hat{\mathbf{z}}$ of the matrix $\tilde{\mathbf{A}}$:

$$\lambda \hat{\mathbf{z}}^* = \tilde{\mathbf{A}}\hat{\mathbf{z}}^* \tag{4.1.5}$$

As $\tilde{\mathbf{A}}$ is not symmetric, there is now the possibility that the real part of λ is positive, and the system is unstable. The matrix $\mathbf{A}_f$ has terms proportional to displacement and velocity that can be thought of as forms of stiffness and damping. Through those terms, the eigenmodes $\boldsymbol{\eta}$ of the free structure become coupled, so orthogonality is lost and the eigenmodes $\hat{\mathbf{z}}^*$ become combinations of the eigenmodes of the free structure.

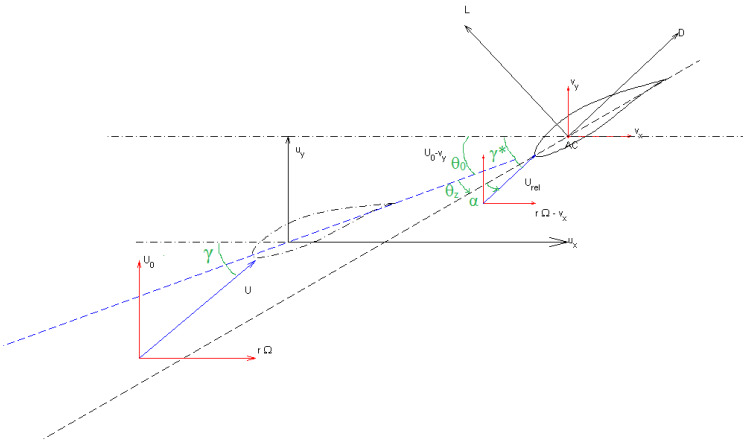


Figure 4.1.1: Flow over a section of the blade, showing relative flow direction when the blade moves

4.1.1 Aerodynamic damping on one section

To get the matrix $\mathbf{A}_f$, we begin by looking at a single section that, superimposed onto its rigid motion in the x-direction due to the blades rotation, is free to translate in the x- and y-direction with displacements $\hat{u}_x, \hat{u}_y$, velocities $\hat{v}_x, \hat{v}_y$ and rotate about the aerodynamic center with the angle θ_z, see figure 4.1.1. The free stream velocity is V_0 and is perpendicular to the rotor plane. Then the flow angle ϕ relative to the rotor plane is given by $\phi = \theta_{twist} + \theta_{pitch} + \alpha = tan^{-1}(\frac{V_0}{r\Omega})$.

Due to the movement of the blade, we have the relative flow angle $\phi^* = tan^{-1}(\frac{V_0 - \hat{v}_y}{r\Omega - \hat{v}_x})$, and the lift L and drag D are defined with respect to this. The forces in the x- and y-direction and the pitching monent becomes:

$$f_x = Dcos(\phi^*) - Lsin(\phi^*) \tag{4.1.6}$$
$$f_y = Dsin(\phi^*) + Lcos(\phi^*) \tag{4.1.7}$$
$$M_z = M \tag{4.1.8}$$

We now take the derivative with respect to some state z_j. These states will later be the eigenfrequencies of the structure. As both the magnitude of the aerodynamic forces as well as the angle of attack changes due to the displacement of the section the chain rule gives:

$$\frac{\partial f_x}{\partial z_j} = \frac{\partial D}{\partial z_j}cos(\phi^*) - \frac{\partial L}{\partial z_j}sin(\phi^*) - (Dsin(\phi^*) + Lcos(\phi^*))\frac{\partial \phi^*}{\partial z_j} \tag{4.1.9}$$
$$\frac{\partial f_y}{\partial z_j} = \frac{\partial D}{\partial z_j}sin(\phi^*) + \frac{\partial L}{\partial z_j}cos(\phi^*) + (Dcos(\phi^*) - Lsin(\phi^*))\frac{\partial \phi^*}{\partial z_j} \tag{4.1.10}$$
$$\frac{\partial M_z}{\partial z_j} = \frac{\partial M}{\partial z_j} \tag{4.1.11}$$

the parts $\frac{\partial D}{\partial z_j}$ and $\frac{\partial L}{\partial z_j}$ have a dependence both on the relative flow velocity and on the aerodynamic coefficients:

$$D = \frac{1}{2}\rho c V_{rel}^2 C_D \tag{4.1.12}$$

$$L = \frac{1}{2}\rho c V_{rel}^2 C_L \tag{4.1.13}$$

$$M = \frac{1}{2}\rho c^2 V_{rel}^2 C_L \tag{4.1.14}$$

$$\begin{aligned}\frac{\partial D}{\partial z_j} &= \rho c V_{rel} C_D \frac{\partial V_{rel}}{\partial z_j} + \frac{1}{2}\rho c V_{rel}^2 \frac{\partial C_D}{\partial z_j} \\ &\approx \frac{1}{2}\rho c U^2 (\frac{2}{U} C_D \frac{\partial V_{rel}}{\partial z_j} + \frac{\partial C_D}{\partial z_j}) \end{aligned} \tag{4.1.15}$$

$$\begin{aligned}\frac{\partial L}{\partial z_j} &= \rho c V_{rel} C_L \frac{\partial V_{rel}}{\partial z_j} + \frac{1}{2}\rho c V_{rel}^2 \frac{\partial C_L}{\partial z_j} \\ &\approx \frac{1}{2}\rho c U^2 (\frac{2}{U} C_L \frac{\partial V_{rel}}{\partial z_j} + \frac{\partial C_D}{\partial z_j}) \end{aligned} \tag{4.1.16}$$

$$\begin{aligned}\frac{\partial M}{\partial z_j} &= \rho c^2 V_{rel} C_M \frac{\partial V_{rel}}{\partial z_j} + \frac{1}{2}\rho c^2 V_{rel}^2 \frac{\partial C_M}{\partial z_j} \\ &\approx \frac{1}{2}\rho c^2 U^2 (\frac{2}{U} C_M \frac{\partial V_{rel}}{\partial z_j} + \frac{\partial C_M}{\partial z_j}) \end{aligned} \tag{4.1.17}$$

as the velocity relative to the blade V_{rel} is

$$V_{rel} = \sqrt{(r\Omega - \hat{v}_x)^2 + (V_0 - \hat{v}_y)^2} \tag{4.1.18}$$

$$\Rightarrow$$

$$\begin{aligned}\frac{\partial V_{rel}}{\partial z_j} &= -\frac{r\Omega - \hat{v}_x}{V_{rel}} \frac{\partial \hat{v}_x}{\partial z_j} - \frac{V_0 - \hat{v}_y}{V_{rel}} \frac{\partial \hat{v}_y}{\partial z_j} \approx -\frac{r\Omega}{U} \frac{\partial \hat{v}_x}{\partial z_j} - \frac{V_0}{U} \frac{\partial \hat{v}_y}{\partial z_j} \\ &= -cos(\phi) \frac{\partial \hat{v}_x}{\partial z_j} - sin(\phi) \frac{\partial \hat{v}_y}{\partial z_j} \end{aligned} \tag{4.1.19}$$

insert into 4.1.15 to 4.1.17:

$$\frac{\partial D}{\partial z_j} = \frac{1}{2}\rho c U^2 \{-\frac{2}{U} C_D [cos(\phi) \frac{\partial \hat{v}_x}{\partial z_j} + sin(\phi) \frac{\partial \hat{v}_y}{\partial z_j}] + \frac{\partial C_D}{\partial z_j}\} \tag{4.1.20}$$

$$\frac{\partial L}{\partial z_j} = \frac{1}{2}\rho c U^2 \{-\frac{2}{U} C_L [cos(\phi) \frac{\partial \hat{v}_x}{\partial z_j} + sin(\phi) \frac{\partial \hat{v}_y}{\partial z_j}] + \frac{\partial C_L}{\partial z_j}\} \tag{4.1.21}$$

$$\frac{\partial M}{\partial z_j} = \frac{1}{2}\rho c^2 U^2 \{-\frac{2}{U} C_M [cos(\phi) \frac{\partial \hat{v}_x}{\partial z_j} + sin(\phi) \frac{\partial \hat{v}_y}{\partial z_j}] + \frac{\partial C_M}{\partial z_j}\} \tag{4.1.22}$$

using that $\phi^* = tan^{-1}(\frac{V_0 - \hat{v}_y}{R\Omega - \hat{v}_x})$, it can be shown that

$$\frac{\partial \phi^*}{\partial z_j} \approx \frac{1}{U} sin(\phi) \frac{\partial \hat{v}_x}{\partial z_j} - \frac{1}{U} cos(\phi) \frac{\partial \hat{v}_y}{\partial z_j} \tag{4.1.23}$$

therefore

$$[Dsin(\phi)+Lcos(\phi)]\frac{\partial\phi^*}{\partial z_j} = -\frac{1}{2}\rho cU^2[C_Dsin(\phi)+C_Lcos(\phi)][-\frac{1}{U}sin(\phi)\frac{\partial\hat{v}_x}{\partial z_j}+\frac{1}{U}cos(\phi)\frac{\partial\hat{v}_y}{\partial z_j}]$$
$$= -\frac{1}{2}\rho cU^2\{-\frac{1}{U}[C_Dsin(\phi)+C_Lcos(\phi)]sin(\phi)\frac{\partial\hat{v}_x}{\partial z_j}+$$
$$+\frac{1}{U}[C_Dsin(\phi)+C_Lcos(\phi)]cos(\phi)\frac{\partial\hat{v}_y}{\partial z_j}\} \quad (4.1.24)$$
$$[Dcos(\phi)-Lsin(\phi)]\frac{\partial\phi^*}{\partial z_j} = -\frac{1}{2}\rho cU^2[C_Dcos(\phi)-C_Lsin(\phi)][-\frac{1}{U}sin(\phi)\frac{\partial\hat{v}_x}{\partial z_j}+\frac{1}{U}cos(\phi)\frac{\partial\hat{v}_y}{\partial z_j}]$$
$$= -\frac{1}{2}\rho cU^2\{-\frac{1}{U}[C_Dcos(\phi)-C_Lsin(\phi)]sin(\phi)\frac{\partial\hat{v}_x}{\partial z_j}+$$
$$+\frac{1}{U}[C_Dcos(\phi)-C_Lsin(\phi)]cos(\phi)\frac{\partial\hat{v}_y}{\partial z_j}\} \quad (4.1.25)$$

use eqns 4.1.20 - 4.1.22 and 4.1.24 - 4.1.25 in eqns 4.1.9 to 4.1.11 :

$$\frac{\partial f_x}{\partial z_j} = -\frac{1}{2}\rho cU^2\{\frac{1}{U}[C_D(2cos^2(\phi)+sin^2(\phi))-C_Lcos(\phi)sin(\phi)]\frac{\partial\hat{v}_x}{\partial z_j}$$
$$+\frac{1}{U}[C_Dsin(\phi)cos(\phi)-C_L(2sin^2(\phi)+cos^2(\phi))]\frac{\partial\hat{v}_y}{\partial z_j}$$
$$-\frac{\partial C_D}{\partial z_j}cos(\phi)+\frac{\partial C_L}{\partial z_j}sin(\phi)\} \quad (4.1.26)$$
$$\frac{\partial f_y}{\partial z_j} = -\frac{1}{2}\rho cU^2\{[\frac{1}{U}C_Dcos(\phi)sin(\phi)+\frac{1}{U}C_L(2cos^2(\phi)+sin^2(\phi))]\frac{\partial\hat{v}_x}{\partial z_j}+$$
$$+[\frac{1}{U}C_D(2sin^2(\phi)+cos^2(\phi))+\frac{1}{U}sin(\phi)cos(\phi)C_L]\frac{\partial\hat{v}_y}{\partial z_j}$$
$$-\frac{\partial C_D}{\partial z_j}sin(\phi)-\frac{\partial C_L}{\partial z_j}cos(\phi)\} \quad (4.1.27)$$
$$\frac{\partial M}{\partial z_j} = -\frac{1}{2}\rho c^2U^2\{\frac{2}{U}C_M[cos(\phi)\frac{\partial\hat{v}_x}{\partial z_j}+sin(\phi)\frac{\partial\hat{v}_y}{\partial z_j}]-\frac{\partial C_M}{\partial z_j}\} \quad (4.1.28)$$

from these equations, we see that for small ϕ, and when C_D is smaller than C_L the motion in the x-direction is less damped than in the y-direction.

There is no direct dependence on the twist angle θ_z in these equations, though classical flutter always includes bending and twist. The dependence on the twist angle comes in through the terms $\frac{\partial C_L}{\partial z_j}$ and less importantly $\frac{\partial C_D}{\partial z_j}$ and $\frac{\partial C_M}{\partial z_j}$.

As we have that

$$\begin{bmatrix}\hat{v}_x\\ \hat{v}_y\end{bmatrix} = \mathbf{T}_i\dot{\mathbf{q}} = \mathbf{T}_i\mathbf{Q}\dot{\boldsymbol{\eta}} \quad (4.1.29)$$

where $\mathbf{T}$ is from equation (3.3.9) from chapter 3.3.1 and $\mathbf{Q}$ is the modal matrix, we have the derivatives $\frac{\partial\hat{v}_x}{\partial z_j}$ and $\frac{\partial\hat{v}_y}{\partial z_j}$ as rows of the matrix $\mathbf{TQ}$ The derivatives $\frac{\partial C_L}{\partial z_j}$, $\frac{\partial C_M}{\partial z_j}$ and $\frac{\partial C_D}{\partial z_j}$ is the subject of chapter 4.2.

4.1.2 Stall induced vibrations

By inspection, equations 4.1.26 to 4.1.28 are equivalent to those used in the Risø report 4.2.1 if one ignores the pitching moment component and sets $z_1 = \hat{v}_x$ and

$z_2 = \hat{v}_y$. Further, if one makes the quasi-steady assumtion that the $\frac{\partial C_L}{\partial z_j}$ only components is due to changes in angle of attack α, i.e:

$$\frac{\partial C_L}{\partial z_j} = \frac{\partial C_L}{\partial \alpha}\frac{\partial \alpha}{\partial z_j} \tag{4.1.30}$$

the derivative $\frac{\partial C_L}{\partial \alpha}$ is taken from tabulated data of steady conditions. Then one gets the following equations for aeroelastic damping coefficients:

$$\begin{aligned}
\frac{\partial f_x}{\partial \hat{v}_x} &= -c_{xx} = -\frac{1}{2}\rho c U\{C_D(2cos^2(\phi) + sin^2(\phi)) - C_L cos(\phi)sin(\phi) - \\
&\quad \frac{\partial C_D}{\partial \alpha}cos(\phi)sin(\phi) + \frac{\partial C_L}{\partial \alpha}sin^2(\phi)\} && (4.1.31)\\
\frac{\partial f_x}{\partial \hat{v}_y} &= -c_{xy} = -\frac{1}{2}\rho c U\{C_D sin(\phi)cos(\phi) - C_L(2sin^2(\phi) + cos^2(\phi)) + \\
&\quad \frac{\partial C_D}{\partial \alpha}cos^2(\phi) - \frac{\partial C_L}{\partial \alpha}sin(\phi)cos(\phi)\} && (4.1.32)\\
\frac{\partial f_y}{\partial \hat{v}_x} &= -c_{yx} = -\frac{1}{2}\rho c U\{C_D cos(\phi)sin(\phi) + C_L(2cos^2(\phi) + sin^2(\phi)) - \\
&\quad \frac{\partial C_D}{\partial \alpha}sin^2(\phi) - \frac{\partial C_L}{\partial \alpha}cos(\phi)sin(\phi)\} && (4.1.33)\\
\frac{\partial f_y}{\partial \hat{v}_y} &= -c_{yy} = -\frac{1}{2}\rho c U\{C_D(2sin^2(\phi) + cos^2(\phi)) + sin(\phi)cos(\phi)C_L + \\
&\quad \frac{\partial C_D}{\partial \alpha}sin(\phi)cos(\phi) + \frac{\partial C_L}{\partial \alpha}cos^2(\phi)\} && (4.1.34)
\end{aligned}$$

One can use these damping coefficients and constrain the section to move in one direction. If the damping is plotted as a function of direction of vibration, it can be seen that for certain directions of motion, the damping becomes negative. When the angle of attack becomes high, i.e, at stalled conditions, either the edgewise or the flapwise mode, typically the edgewise, will have vibration motion that falls into a range where the damping is negative. This instability is stall induced vibrations or stall flutter. The difference to classical flutter is that it occurs under stalled conditions and involves little coupling between modes. In the actual program, we have not made the quasi-steady assumption, as it has been shown that one must take into account the effects of dynamic stall.

The NREL reference turbine is pitch-regulated, which means that it does not operate at high angles of attack under normal circumstances. Stall flutter is not thus not a large concern for it.

4.1.3 Flutter

Flutter is another phenomenom of instablity. It occurs due to an unfavorable coupling between the torsion and bending of a blade or wing. When the blade moves in a torsional mode, the angle of attack changes. Under attached flow conditions, a change in angle of attack means that the lift force increases. This change in lift force excites the bending modes. Thus the bending and torsional modes become coupled.

4.1.4 Aeroelastic stiffness and damping matrix

By linearizing the aerodynamic forces through equation 4.1.26 to 4.1.28 for each section, a force vector can be constructed as:

$$\frac{\partial \hat{\mathbf{f}}}{\partial z_j} = \begin{bmatrix} \frac{\partial f_x^1}{\partial z_j} \\ \frac{\partial f_y^1}{\partial z_j} \\ 0 \\ 0 \\ 0 \\ \frac{\partial m_z^1}{\partial z_j} \\ \dots \\ \frac{\partial f_x^N}{\partial z_j} \\ \frac{\partial f_y^N}{\partial z_j} \\ 0 \\ 0 \\ 0 \\ \frac{\partial m_z^N}{\partial z_j} \end{bmatrix} \tag{4.1.35}$$

where zeros have been put in for the axial force and distributed bending moments. By doing this for all our modes $z_j, j = 1, 2..., m$ we get the derivative of sectional forces $\hat{\mathbf{f}}$ with respect to the eigenmodes $\mathbf{z}$:

$$\frac{\partial \hat{\mathbf{f}}}{\partial \mathbf{z}} = \begin{bmatrix} \frac{\partial \hat{\mathbf{f}}}{\partial z_1} & \frac{\partial \hat{\mathbf{f}}}{\partial z_2} & \dots & \frac{\partial \hat{\mathbf{f}}}{\partial z_m} \end{bmatrix} \tag{4.1.36}$$

in equation 4.1.4 there is the nodal forces $\mathbf{f}$, that one would have to find from the forces on the sections. However as $\mathbf{A}_f$ can be rewritten to:

$$\mathbf{A}_f = \begin{bmatrix} \mathbf{0} \\ \mathbf{Q}^T \\ \mathbf{0} \end{bmatrix} \frac{\partial \mathbf{f}}{\partial \mathbf{z}}_{|\mathbf{z}=\mathbf{z}_0} = \begin{bmatrix} \mathbf{0} \\ \frac{\partial \bar{\mathbf{f}}}{\partial \mathbf{z}}_{|\mathbf{z}=\mathbf{z}_0} \\ \mathbf{0} \end{bmatrix} \tag{4.1.37}$$

where $\bar{\mathbf{f}}$ are the modal forces. The modal forces can be got by constructing a modal matrix $\hat{\mathbf{Q}} = \mathbf{TQ}$, of sectional modes. The modal forces are then calculated from that matrix:

$$\frac{\partial \bar{\mathbf{f}}}{\partial \mathbf{z}} = \hat{\mathbf{Q}}^T \frac{\partial \hat{\mathbf{f}}}{\partial \mathbf{z}} \tag{4.1.38}$$

to summarize:

$$\mathbf{A}_f = \begin{bmatrix} \mathbf{0} \\ \hat{\mathbf{Q}}^T \\ \mathbf{0} \end{bmatrix} \frac{\partial \hat{\mathbf{f}}}{\partial \mathbf{z}}_{|\mathbf{z}=\mathbf{z}_0} \tag{4.1.39}$$

4.1.5 Including stall variables in eigenvalue analysis

The "stall variables" x_1 to x_4 are included by collecting them in the vector x:

$$\mathbf{x} = \begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \\ \mathbf{x}_3 \\ \mathbf{x}_4 \end{bmatrix} \tag{4.1.40}$$

equations 4.2.23 can be written on this form:

$$\begin{bmatrix} \dot{\mathbf{x}}_1 \\ \dot{\mathbf{x}}_2 \\ \dot{\mathbf{x}}_3 \\ \dot{\mathbf{x}}_4 \end{bmatrix} = \mathbf{A}_{xx}\mathbf{x} + \mathbf{A}_{xu}\mathbf{z} \tag{4.1.41}$$

all dependence on geometrical terms like the three quarter point downwash angle etc. are expressed through the term $\mathbf{A}_{xu}\mathbf{z}$. Introduce the new state vector $\mathbf{z}_x$:

$$\mathbf{z}_x = \begin{bmatrix} \mathbf{z} \\ \mathbf{x}_1 \\ \mathbf{x}_2 \\ \mathbf{x}_3 \\ \mathbf{x}_4 \end{bmatrix} = \begin{bmatrix} \mathbf{z} \\ \mathbf{x} \end{bmatrix} \tag{4.1.42}$$

the state space equation for this vector then becomes:

$$\dot{\mathbf{z}}_x = \begin{bmatrix} \tilde{\mathbf{A}} & \tilde{\mathbf{A}}_{zx} \\ \mathbf{A}_{xz} & \mathbf{A}_{xx} \end{bmatrix} \mathbf{z}_x \tag{4.1.43}$$

Where the modification of lift, drag, and pitch moment due to the stall variables enters into the term $\tilde{\mathbf{A}}_{zx}\mathbf{x}$

4.2 Dynamic Stall

Introduction

All methods for calculating the forces in this thesis (BEM, HVM and FVM) use tabulated data for C_L, C_D and C_M for a given angle of attack. These data are created mainly from experiments where a profile is exposed to flow at a certain Reynolds number for different angles of attack. However these data are only true during steady conditions, so if the purpose is to calculate anything unsteady there would be errors. The reason is that there is a time lag and an unsteady effect in the force and the pressures when unsteady effects appear.

To take the unsteadiness in account a model called the dynamic stall model has been created. Nowadays there are many different types of dynamic stall models (Onera, Leishman-Beddoes are two commonly used models). Depending on the application different dynamic stall models should be used, for wind turbines the Leishman-Beddoes is very popular.

Leishman-Beddoes was written mainly for helicopter applications and thus there are some parts that is not really necessary. Therefore Risø has created a version specifically for wind turbines, it can be found in [19].

Physical explanation of Dynamic Stall

Physically what is happening when there is a change in the flow field is described below:

Assume first that an airfoil is exposed to a steady flow which can be seen in figure 4.2.1 [1]. The left figure shows the flow field and the right figure shows the Cp curve for this flow field (a Cp curve is the pressure distribution over the airfoil, where the pressure has been normalized by dividing it with the stagnation pressure).

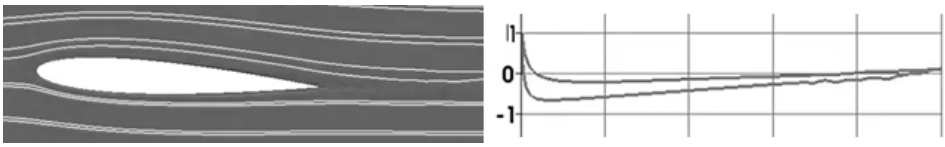


Figure 4.2.1: The left figure shows the flow around an airfoil in steady conditions. And the right figure shows the Cp curve for the flow field.

The airfoil is then rotated a few degrees, and what happens is that due to a delay in the flow field a vortex is created at the leading edge (hence called leading edge vortex). This vortex creates suction at the suction side of the leading edge, creating a lift force, a decrese in the drag and a moment that wants to lift the leading edge. This can be seen in figure 4.2.2.

After a small amount of time, the leading edge vortex is realeased from the blade, which removes the suction at the leading edge. However, as an aftermath of the leading edge vortex releasing a trailing edge vortex is created (not so surprisingly located at the trailing edge of the airfoil). This vortex creates a suction at the suctions side of the airfoil at the trailing edge. This can be seen in figure 4.2.3 where the suction increases the lift and the drag force, and creates a moment that wants to lift the trailing edge of the airfoil.

[1] The four figures in this chapter are snapshots from a video on YouTube uploaded by kimyusiks on Jul 5, 2011. The video can be seen either by using the link: http://www.youtube.com/watch?v=MHuQIWfD9dk , or by searching on YouTube for Oscillating airfoil.

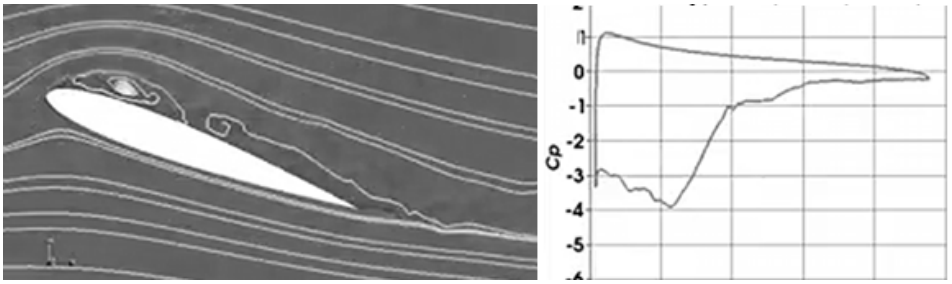


Figure 4.2.2: The left figure shows the flow around an airfoil shortly after a change in the angle of attack. And the right figure shows the Cp curve for the flow field.

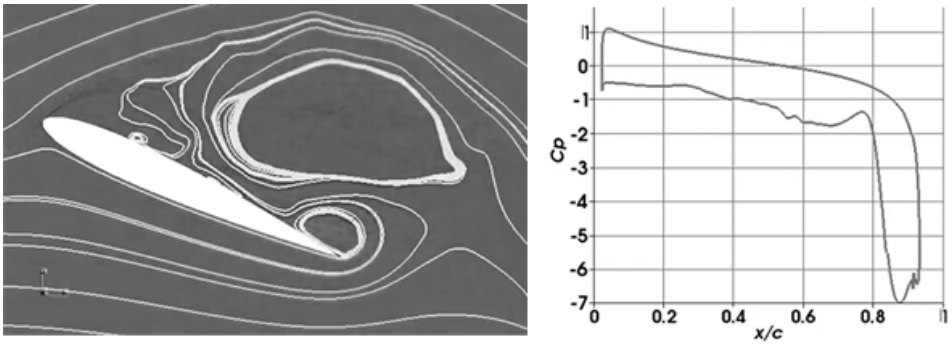


Figure 4.2.3: The left figure shows the flow around an airfoil shortly after the trailing edge vortex is released. And the right figure shows the Cp curve for the flow field.

After some time the trailing edge vortex will release aswell and the flow will go back to normal, as can be seen in figure 4.2.4.

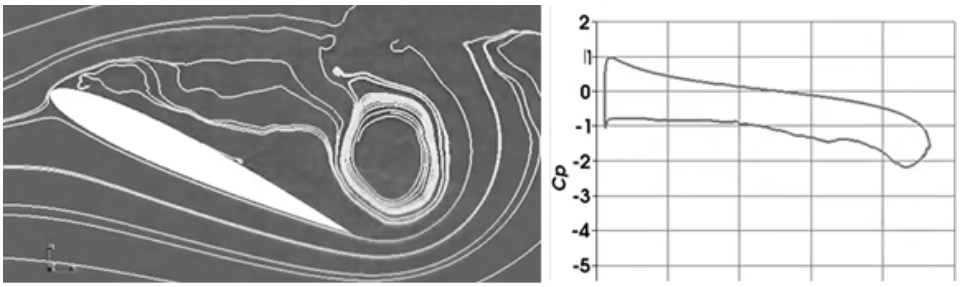


Figure 4.2.4: Flow around an airfoil in steady conditions at the new angle of attack.

The description above was for a general airfoil. However for wind turbines, due to the relatively low wind speeds and the relatively thick airfoils used, the leading edge vortex is not that strong [19]. Therefore in the Risø model the trailing edge vortex has been neglected and only the trailing edge vortex is accounted for.

More detailed physical eplanation can be found in [11] and [19].

Mathematical model

The dynamic stall model[2] calculates the lift, drag and momentum forces when a change in angle of attack or inlet velocity happens. To calculate this the input needed is the tabulated data for the static C_L, C_D and C_M for all angle of attacks.

[2]The derivations shown in this chapter have been extracted from [19].

The other things that are needed is the velocity U, the angle of attack α for both the previous times-step, and the current time-step. The acceleration $\dot{U}$ and the angular velocity $\dot{\alpha}$ are also needed, and the size of the time-step dt is needed as well.

The tabulated data for the lift, drag and momentum are denoted C_L^{st}, C_D^{st} and C_M^{st}, respectively which are measured at the different angle of attacks α. There are six different constants used for this model, these can be seen in table 4.2.1.

Table 4.2.1: Constants used in the Risø model

b_1	0.0455
b_2	0.3
A_1	0.165
A_2	0.335
ω_7	0.4125
ω_3	0.0875

For all airfoils there are some characteristic data: the angle at which the lift is zero C_{L_0} and the slope of the linear part of the lift curve called $C_{L,\alpha}$. Other interesting data are the drag and the momentum at the angle of attack where the lift is zero C_{D_0} and C_{M_0}.

$C_{L,\alpha}$ is calculated as:

$$C_{L,\alpha} = \max\{\frac{C_L^{st}(\alpha)}{(\alpha - \alpha_0)}\} \tag{4.2.1}$$

When calculating near stall conditions it is very important is where the separation begins, or where the flow close to the wall is flowing against the free stream velocity. This point is expressed as f^{st} which is the procentual distance from the leading edge. f^{st} can be approximated from the tabulated data as:

$$f^{st} = \frac{C_L^{st}}{C_{L,\alpha} \cdot (\alpha - \alpha_0)} \tag{4.2.2}$$

The lift for a fully stalled airfoil at a certain angle of attack can be calculated as:

$$C_L^{fs} = \frac{C_L^{st} - C_{L,\alpha} \cdot (\alpha - \alpha_0) \cdot f^{st}}{1 - f^{st}} \tag{4.2.3}$$

The above equation does not work if f^{st} =1, so when f^{st} =1 then C_L^{fs} should be calculated as:

$$C_L^{fs} = \frac{C_L^{st}}{2} \tag{4.2.4}$$

For the moment there is another interesting point, which is the position of the pressure centre a^{st}, which can be calculated as:

$$a^{st} = \frac{C_M^{st} - C_{M_0}}{C_L^{st}} \tag{4.2.5}$$

If one approximates the impulse response function $\Phi(t)$ as:

$$\Phi(t) = 1 - A_1 e^{-\omega_1 t} - A_2 e^{-\omega_2 t} \tag{4.2.6}$$

Then the state-space variables x_i can be calculated as:

$$\dot{x}_1 + T_u^{-1}\frac{b_1 + c\dot{U}}{2U^2}x_1 = b_1 A_1 T_u^{-1}\alpha_{3/4} \tag{4.2.7}$$

$$\dot{x}_2 + T_u^{-1}\frac{b_2 + c\dot{U}}{2U^2}x_2 = b_2 A_2 T_u^{-1}\alpha_{3/4} \tag{4.2.8}$$

$$\dot{x}_3 + T_p^{-1}x_3 = T_p^{-1}\cdot(C_{L,\alpha}\cdot(\alpha_E - \alpha_0) + \pi T_u\dot{\alpha}) \tag{4.2.9}$$

$$\dot{x}_4 + T_f^{-1}x_4 = T_f^{-1}f^{st}(\frac{x_3}{C_{L,\alpha} + \alpha_0}) \tag{4.2.10}$$

where α_E is calculated as:

$$\alpha_E = \alpha_{3/4}\cdot(1 - A_1 + A_2) + x_1 + x_2 \tag{4.2.11}$$

And T_u, T_f and T_p are calculated as:

$$T_u = \frac{c}{2U} \tag{4.2.12}$$

$$T_p = \frac{\omega_7 c}{2U} \tag{4.2.13}$$

$$T_f = \frac{\omega_3 c}{2U} \tag{4.2.14}$$

Where c is the chord length.

When the differential equations are solved the only thing that is left to do is to calculate the dynamic variables as:

$$C_L^{dyn} = C_{L,\alpha}\cdot(\alpha_E - \alpha_0)\cdot x_4 + C_L^{fs}(\alpha_E)\cdot(1 - x_4) + \pi T_u\dot{\alpha} \tag{4.2.15}$$

$$C_D^{dyn} = C_D^{st}(\alpha_E) + (\alpha - \alpha_E)\cdot C_L^{dyn} + (C_D^{st}(\alpha_E) - C_{D_0})\cdot(\frac{\sqrt{f^{st}(\alpha_E)} - \sqrt{x_4}}{2} - \frac{f^{st}(\alpha_E) - x_4}{4}) \tag{4.2.16}$$

$$C_M^{dyn} = C_M^{st}(\alpha_E) + C_L^{dyn}\cdot(a^{st}(x_4) - a^{st}(f^{st}(\alpha_E))) - \frac{\pi}{2}T_u\dot{\alpha} \tag{4.2.17}$$

4.2.1 Using dynamic stall for flutter analys

When the blade vibrates and changes the angle of attack, dynamic stall effects as described above starts to show. For most frequencies the fluid is acting as a viscous dampener, reducing the amplitude of the vibrations. However at certain frequencies, the vibration of the blade matches with the Reynolds number of the flow in such a way that the fluid, instead of dampening the vibrations, are increasing the amplitude and thus flutter problems have arisen.

When doing a flutter analysis, which is described in chapter 4.1, the eigenvalues are calculate, through the equation 4.1.4.

The equation is linear, where the forces from the fluid is called $\mathbf{A_f}$. However the dynamic stall equations are very non-linear, therefore to be able to do a flutter analysis the dynamic stall equations needs to be linearised[3]. This can rather safely be done when linearising around a steady state, and assuming that the changes in α and U and so on are small, i.e. the variables can be expressed as:

$\alpha = \alpha^0 + \alpha^1$, $\alpha_{3/4} = \alpha^0 + \alpha^1_{3/4}$ and $U = U_0 + U_1$ where U_1, α^1 and $\alpha^1_{3/4}$ are small. Which means that x^1_i in the equation below is small aswell:

$$x_i = x^0_i + x^1_i \tag{4.2.18}$$

The above assumptions gives that the state-space variables for the steady state x^0_i can be expressed as below:

$$x^0_1 = A_1\alpha^0 \tag{4.2.19}$$
$$x^0_2 = A_2\alpha^0 \tag{4.2.20}$$
$$x^0_3 = C_{L,\alpha} \cdot \left(\alpha^0 - \alpha_0\right) \tag{4.2.21}$$
$$x^0_4 = f^{st}(\alpha^0) \tag{4.2.22}$$

This means that the differential equations can be expressed as:

$$\dot{x}^1_1 + T_1^{-1}x^1_1 = A_1T_1^{-1}\alpha^1_{3/4} - \frac{A_1\alpha^0}{U_0}\dot{U}_1 \tag{4.2.23}$$
$$\dot{x}^1_2 + T_2^{-1}x^1_2 = A_2T_2^{-1}\alpha^1_{3/4} - \frac{A_2\alpha^0}{U_0}\dot{U}_1 \tag{4.2.24}$$
$$\dot{x}^1_3 + T_p^{-1}x^1_3 = T_p^{-1} \cdot (C_{L,\alpha}\alpha^1_E + \pi T_0\dot{\alpha}^1) \tag{4.2.25}$$
$$\dot{x}^1_4 + T_f^{-1}x^1_4 = T_f^{-1}\frac{df^{st}}{d\alpha}\Big|_{\alpha=\alpha^0}\frac{x^1_3}{C_{L.\alpha}} \tag{4.2.26}$$

Where, as before α^1_E can be calculated as:

$$\alpha^1_E = \alpha^1_{3/4} \cdot (1 - A_1 + A_2) + x^1_1 + x^1_2 \tag{4.2.27}$$

And T_1 and T_2 is calculated as:

$$T_1 = \frac{c}{2U_0b_1} \tag{4.2.28}$$
$$T_2 = \frac{c}{2U_0b_2} \tag{4.2.29}$$

The variable $\frac{df^{st}}{d\alpha}$ is the equation for x_4 when it has been Taylor expanded giving this expression:

$$\frac{df^{st}}{d\alpha} = \frac{2}{C_{L,\alpha} \cdot (\alpha - \alpha_0)} \cdot \left(\frac{dC^{st}_L}{d\alpha} - \frac{C^{st}_L(\alpha)}{\alpha - \alpha_0}\right) \cdot \left(2 - \sqrt{\frac{C_{L,\alpha} \cdot (\alpha - \alpha_0)}{C_L(\alpha)}}\right) \tag{4.2.30}$$

When linearising the dynamic variables, the lift, drag and moment depends on a couple of variables some of them are common to all variables, some are not, however

[3]The derivations shown in this chapter have been extracted from [19].

all of them depends on the effective angle of attack α_E and the position where the flow becomes stalled x_4. By Taylor expanding the dynamic lift, drag and moment the equations below are given:

$$C_L^{lin} = C_L^0 + c_{l,\alpha} \cdot \alpha_E^1 + c_{l,f} \cdot x_4^1 + \pi \frac{c}{2U_0} \dot{\alpha}^1 \tag{4.2.31}$$

$$C_D^{lin} = C_D^0 + c_{d,\alpha} \cdot \alpha_E^1 + c_{d,f} \cdot x_4^1 + C_L^0 \cdot \left(\alpha^1 - \alpha_E^1\right) \tag{4.2.32}$$

$$C_M^{lin} = C_M^0 + c_{m,\alpha} \cdot \alpha_E^1 + c_{m,f} \cdot x_4^1 - \pi \frac{c}{2U_0} \dot{\alpha}^1 \tag{4.2.33}$$

Where the linearised constants $c_{l,\alpha}$, $c_{l,f}$ etc. is calculated as below:

$$c_{l,\alpha} = C_{L,\alpha} \cdot f_0 + \frac{dC_L^{fs}}{d\alpha}\Big|_{\alpha=\alpha^0} (1 - f_0) \tag{4.2.34}$$

$$c_{l,f} = C_{L,\alpha} \cdot \left(\alpha^0 - \alpha_0\right) - C_L^{fs}(\alpha^0) \tag{4.2.35}$$

$$c_{d,\alpha} = \frac{dC_D^{st}}{d\alpha}\Big|_{\alpha=\alpha^0} - \frac{df^{st}}{d\alpha}\Big|_{\alpha=\alpha^0} \left(C_{D_0} - C_D^0\right) \cdot \frac{1 - \sqrt{f_0}}{4\sqrt{f_0}} \tag{4.2.36}$$

$$c_{d,f} = \left(C_{D_0}^{st} - C_D^0\right) \cdot \frac{1 - \sqrt{f_0}}{4\sqrt{f_0}} \tag{4.2.37}$$

$$c_{m,\alpha} = \frac{dC_M^{st}}{d\alpha}\Big|_{\alpha=\alpha^0} - C_L^0 \cdot \frac{df^{st}}{d\alpha}\Big|_{\alpha=\alpha^0} \cdot \frac{da^{st}}{df}\Big|_{\alpha=\alpha^0} \tag{4.2.38}$$

$$c_{m,f} = C_L^0 \cdot \frac{da^{st}}{df}\Big|_{\alpha=\alpha^0} \tag{4.2.39}$$

$\frac{dC_L^{fs}}{d\alpha}$ can be calculated as:

$$\frac{dC_L^{fs}}{d\alpha} = \frac{\left(\frac{dC_L^{st}}{d\alpha} - C_{L,\alpha} \cdot f^{st}(\alpha)\right)(1 - f^{st}(\alpha)) + (C_L^{st} - C_{L,\alpha} \cdot (\alpha - \alpha_0))\frac{df^{st}}{d\alpha}}{(1 - f^{st}(\alpha))^2} \tag{4.2.40}$$

And $\frac{dC_L^{st}}{d\alpha}$, $\frac{dC_D^{st}}{d\alpha}$, $\frac{dC_M^{st}}{d\alpha}$ and $\frac{da^{st}}{df}$ is simply calculated as the difference between C_L^{st} / C_D^{st} or C_M^{st} divided by the difference between α for the tabulated data (or difference between da^{st} divided by the difference between α for $\frac{da^{st}}{df}$).

The degrees of freedom like the effective angle of attack, downwash angle on 3/4 chord in equations 4.2.31 - 4.2.33 can be calculated from modal degrees of freedom in the following way. Lets collect small pertubations of them in the vector $d\mathbf{u}_d$:

$$d\mathbf{u}_d = \begin{bmatrix} d\boldsymbol{\alpha}_E \\ d\boldsymbol{\alpha}_{3/4} \\ d\dot{\boldsymbol{\alpha}} \\ d\mathbf{x}_1 \\ d\mathbf{x}_2 \\ d\mathbf{x}_3 \\ d\mathbf{x}_4 \\ d\boldsymbol{\alpha} \end{bmatrix} \tag{4.2.41}$$

lets introduce the state vector $\mathbf{z}_{s,x}$ as:

$$\mathbf{z}_{s,x} = \begin{bmatrix} \mathbf{u} \\ \dot{\mathbf{u}} \\ \mathbf{x} \end{bmatrix} \tag{4.2.42}$$

lets further presume that $d\mathbf{u}_d$ are to be found from the state vector $d\mathbf{z}_{sx}$ by the matrix $\mathbf{T}_{ds}$:

$$d\mathbf{u}_d = \mathbf{T}_{ds} d\mathbf{z}_{sx} \tag{4.2.43}$$

if we take the derivative of for example for the lift coefficient with respect to the state vector $\mathbf{z}_x$ using the chain rule:

$$\frac{\partial \mathbf{C}_L}{\partial \mathbf{z}_x} = \frac{\partial \mathbf{C}_L}{\partial \mathbf{u}_d} \frac{\partial \mathbf{u}_d}{\partial \mathbf{z}_{s,x}} \frac{\partial \mathbf{z}_{s,x}}{\partial \mathbf{z}_x} \tag{4.2.44}$$

$$\frac{\partial C_L^i}{\partial u_d^i} = \begin{bmatrix} c_{l,\alpha}^i & 0 & \pi T_0^i & 0 & 0 & 0 & c_{l,f}^i & 0 \end{bmatrix} \tag{4.2.45}$$

The first part of (4.2.44) is taken from [19] and is shown, for one section as an example, in (4.2.45). For the second part of (4.2.44), we start with the sectional displacements and velocities, obtained in chapter 3.3.1, and the stall variables collected in the vector $\hat{\mathbf{z}}$:

$$d\hat{\mathbf{z}}_x = \begin{bmatrix} d\hat{\mathbf{u}} \\ d\dot{\hat{\mathbf{u}}} \\ d\mathbf{x} \end{bmatrix} = \begin{bmatrix} \mathbf{T} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{T} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{I} \end{bmatrix} \begin{bmatrix} d\mathbf{q} \\ d\dot{\mathbf{q}} \\ d\mathbf{x} \end{bmatrix} = \mathbf{T}_{hs} d\mathbf{z}_{s,x} \tag{4.2.46}$$

the quantities $\mathbf{u}_d$ are easily found from the sectional displacements and their velocities together with the stall variables via the matrix $\mathbf{T}_{dh}$.

$$d\mathbf{u}_d = \mathbf{T}_{dh} d\hat{\mathbf{z}}_x = \mathbf{T}_{dh}\mathbf{T}_{hs} d\mathbf{z}_{s,x} = \mathbf{T}_{ds} d\mathbf{z}_{s,x} \tag{4.2.47}$$

$$\begin{bmatrix} d\alpha_E \\ d\alpha_{3/4} \\ d\alpha \\ d\dot{\alpha} \\ dx_1 \\ dx_2 \\ dx_3 \\ dx_4 \end{bmatrix} = \begin{bmatrix} \phi(0)\frac{1}{U} & \phi(0) & \phi(0)\frac{c}{2U} & 1 & 1 & 0 & 0 \\ \frac{1}{U} & 1 & \frac{c}{2U} & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} d\dot{h} \\ d\alpha \\ d\dot{\alpha} \\ dx_1 \\ dx_2 \\ dx_3 \\ dx_4 \end{bmatrix} \tag{4.2.48}$$

here $d\dot{h} = cos(\phi)dv_y - sin(\phi)dv_x$, $d\alpha = -d\hat{\theta}_z$, and $d\alpha = -d\dot{\hat{\theta}}_z$. $\mathbf{T}_{ds}$ is the second part of (4.2.44), i.e

$$\mathbf{T}_{ds} = \frac{\partial \mathbf{u}_d}{\partial \mathbf{z}_{s,x}} \tag{4.2.49}$$

rembering that

$$d\mathbf{z}_x = \begin{bmatrix} d\boldsymbol{\eta} \\ d\dot{\boldsymbol{\eta}} \\ \mathbf{x} \end{bmatrix} \tag{4.2.50}$$

to get $\frac{\partial \mathbf{z}_{s,x}}{\partial \mathbf{z}_x}$ we use that the state vector $\mathbf{z}_{s,x}$ is found from the state vector $\mathbf{z}_x$ by:

$$\mathbf{z}_{s,x} = \begin{bmatrix} d\mathbf{q} \\ d\dot{\mathbf{q}} \\ d\mathbf{x} \end{bmatrix} = \begin{bmatrix} \mathbf{Q} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{Q} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{I} \end{bmatrix} \mathbf{z}_x \tag{4.2.51}$$

$$\frac{\partial \mathbf{z}_{s,x}}{\partial \mathbf{z}_x} = \begin{bmatrix} \mathbf{Q} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{Q} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{I} \end{bmatrix} \tag{4.2.52}$$

$$\mathbf{u}_d = \mathbf{T}_{ds} \begin{bmatrix} \mathbf{q} \\ \dot{\mathbf{q}} \\ \mathbf{x} \end{bmatrix} = \mathbf{T}_{ds} \begin{bmatrix} \mathbf{Q} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{Q} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{I} \end{bmatrix} \mathbf{z} = \mathbf{T}_{d\eta} \mathbf{z} \tag{4.2.53}$$

$$\frac{\partial \mathbf{C}_i}{\partial \mathbf{z}} = \frac{\partial \mathbf{C}_i}{\partial \mathbf{u}_d} \frac{\partial \mathbf{u}_d}{\partial \mathbf{z}} = \frac{\partial \mathbf{C}_i}{\partial \mathbf{u}_d} \mathbf{T}_{d\eta} \tag{4.2.54}$$

5 Model verification

5.1 CFD calculation

Introduction

To be able to compare the results from the BEM, HVM and FVM two CFD calculations are done on a 3-D wind-turbine blade. The first CFD calculation is done on a non-deformed blade, whereas the second calculation is done on a deformed blade. The results are not important in themselves and therefore this chapter will not be very detailed.

Geometry

Both blades are created from the same profiles as the structural model, see figure 5.1.1.

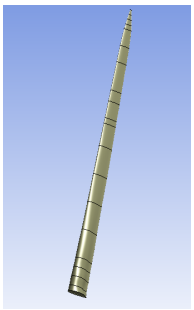


Figure 5.1.1: Flow around an airfoil in steady conditions.

For the deformed blade, the profiles are moved and rotated to form the shape of a deformed blade, which can be seen in figure 5.1.2. How much they are moved and rotated are calculated from the displacement of each section, using the method for calculating the two-way FSI as described above.

Figure 5.1.2: Flow around an airfoil in steady conditions.

The fluid volume is defined as a third of a cylinder, see figure 5.1.3.

Meshing

The meshing has been done in Ansys Mesher where the mesh itself contains six different zones, see figure 5.1.4.

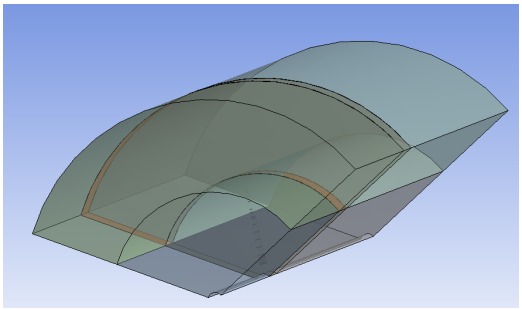


Figure 5.1.3: Flow around an airfoil in steady conditions.

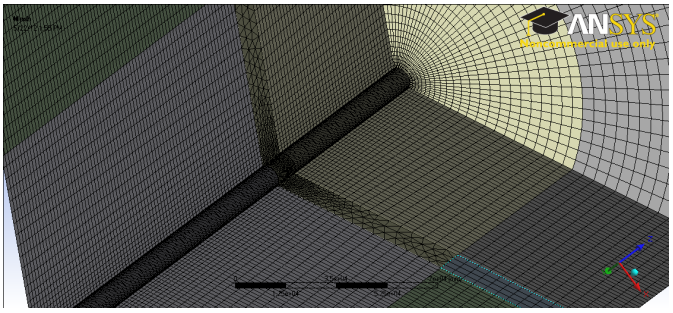


Figure 5.1.4: Flow around an airfoil in steady conditions.

The mesh has around 18 million cells, where the most part of the mesh is hexagonal, except for the part which rotates (the part with the blade in it) which have mostly tetrahedral cells.

Settings

The boundary conditions of the mesh can be seen in figure 5.1.5. To the left the inlet can be seen with an inlet velocity of 11.4 m/s. The centre part of the geometry, the part with the blade in it, has a frozen rotor boundary condition, which rotates with the speed of $\Omega = 12.1$ RPM. All the sides in the cylinder have a periodical boundary condition specified.

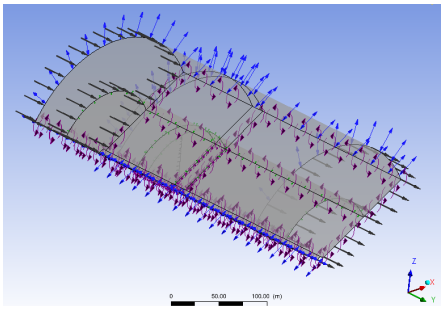


Figure 5.1.5: Flow around an airfoil in steady conditions.

The solver used is Ansys CFX, with the turbulence model k-ω - SST.

Gathering the results

When the simulation have been run, the forces on all sections are calculated from the pressures using the mathematical function with the CFX equation.

$$force\_y()@F244.240 \tag{5.1.1}$$

5.2 Beam model

In order to test the aeroelastic code and how much the results depend on the finite element model, a beam model was created from the properties in the NREL report. This was done with the FE-package CALFEM for MATLAB. The distributed polar moment of inertias were not specified and was taken to a value proportional of the extensional stiffness, and scaled so that the first torsional eigenfrequency was about 8 Hz.

The only modifications of the aeroelastic program necessary to analyze a beam model was to change from the modal matrix of the shell model to the modal matrix of the beam model, and change the matrix $\mathbf{T}$ to the identity matrix.

6 Results

6.1 Blade model results

The finite element model of the blade is shown in figure 6.1.1.

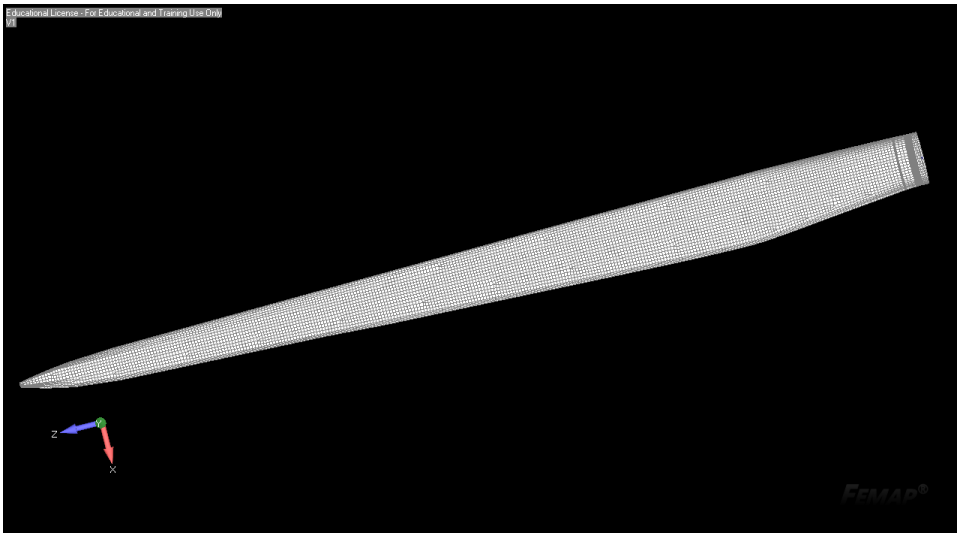


Figure 6.1.1: Finite Element Model of the blade.

The overall integrated results are calculated in FEMAP and shown in table 6.1.1 and compared to those reported by Risø.

Table 6.1.1: Integrated blade results compared to NREL

	Our	NREL
Integrated mass	17 510 kg	17 740 kg
CM location (w.r.t. Root)	20.07 m	20.475 m
Mass moment of inertia (w.r.t. Root)	11 129 312 kg m^2	11 776 047 kg m^2

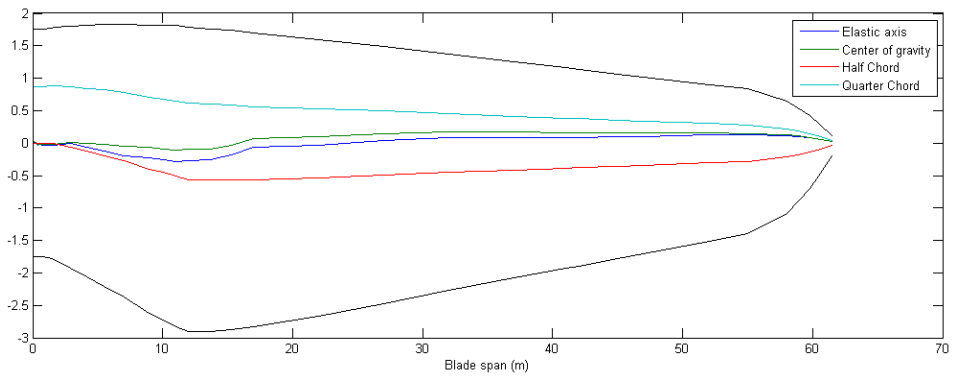


Figure 6.1.2: Blade planform showing center of gravity, elastic axis, half and quarter chord

The sectional properties were calculated in the program crosec by Risø and are shown together with the sectional properites given by NREL in the figure below. The elastic axis and center of gravity have also been calcuated with the same

program and are shown in figure 6.1.2.

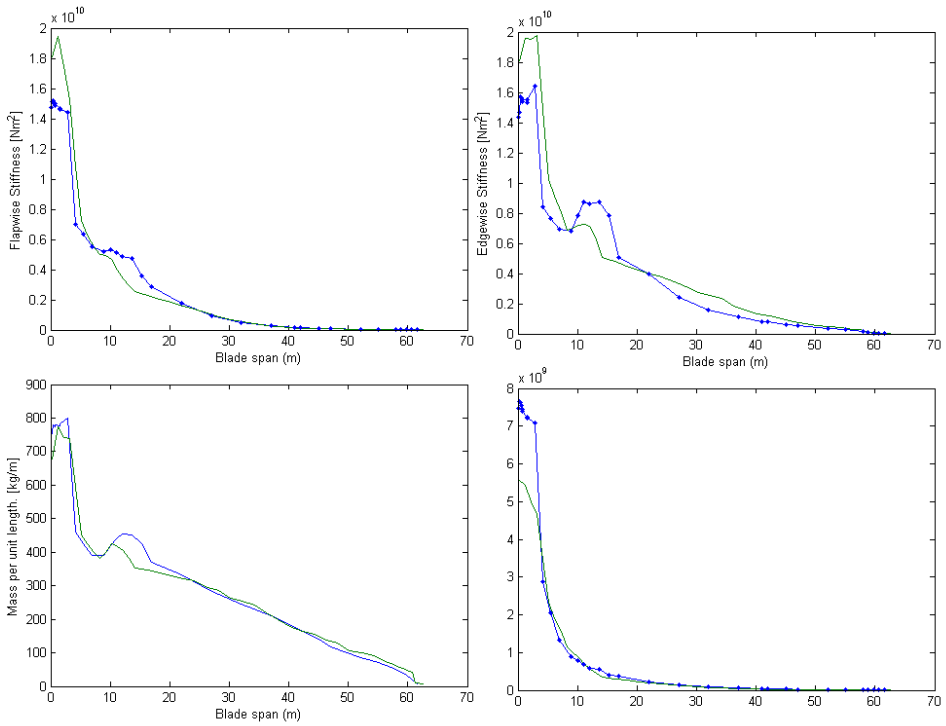


Figure 6.1.3: Cross sectional properties

The cross sectional properties of the blade are compared to those in [4] in figure 6.1.3. These have been calculated with the program CROSEC by Risø. The values calculated have not been used for anything other than validation of the model and when tweaking the model parameters. The cross sectional properties are not exactly equal, and there is especially a large discrepancy near the root of the blade. Overall the blade design has been a compromise betwen getting the right cross sectional propeties, correct eigenfrequencies, and the same integrated properties like total blade mass. One could set up some kind of iterative procedure to get a design that satifies all of these, but this was considered too cumbersome asnd not worthwhile. It is considered sufficient that the total blade mass, and the eigenfrequencies are close to the original, and that the cross sectional properties are reasonably close. If the blade mass, and the eigenfrequencies are correct, the overall stiffness and mass properties are probably correct as well.

One possible weakness is that the first edgewise mode has a fairly large component in the out-of-plane direction. At the tip, the blade moves with an angle 17.7° relative to the rotor plane in the first edgewise mode, and 22.4° in the second. Possibly this reflects wrong principal directions. As shown in chapter 4.1, flapwise motion has higher aerodynamic damping than edgewise motion, so getting too much flapwise motion can stabilize the blade and make the stability analysis unconservative.

6.1.1 Eigenfrequencies and eigenmodes of the blade

An eigenvalue is done in MATLAB to get the eigenfrequencies and eigenvalues. A modal analysis in FEMAP gives the same result. The first six eigenmodes are shown in figure 6.1.4, where the method of calculating equivalent beam displacements from chapter 3.3.1 has been used. Modes 1, 3 and 5 are flapwise modes, 2 and 4 as edgewise modes and mode 6 a torsional mode.

Table 6.1.2: Eigenfrequencies of the finite element blade model.

Mode	Frequency	Note	Dowec
1	0.6581 Hz	1st flapwise	0.6542 Hz *
2	1.1089 Hz	1st edgewise	1.0911 Hz **
3	1.8836 Hz	2nd flapwise	1.9026 Hz**
4	3.4400 Hz	2nd edgewise	3.7741 Hz
5	3.9847 Hz	3rd flapwise	
6	6.1312 Hz	1 st torsional	5.5883 Hz **
7	6.6586 Hz	4 th flapwise	

* From [4] with PHATAS.
** From [4] at 11.844 rpm with BLADMODE.

The resulting eigenfrequencies are shown in table 6.1.2. Note that some of the frequencies included in the comparison from [4] are calculated with a non-zero rotational speed. In those cases the centrifugal force tends to stiffen the structure, though with the low rotational speeds of a wind turbine, the difference is probably not more than 10 %.

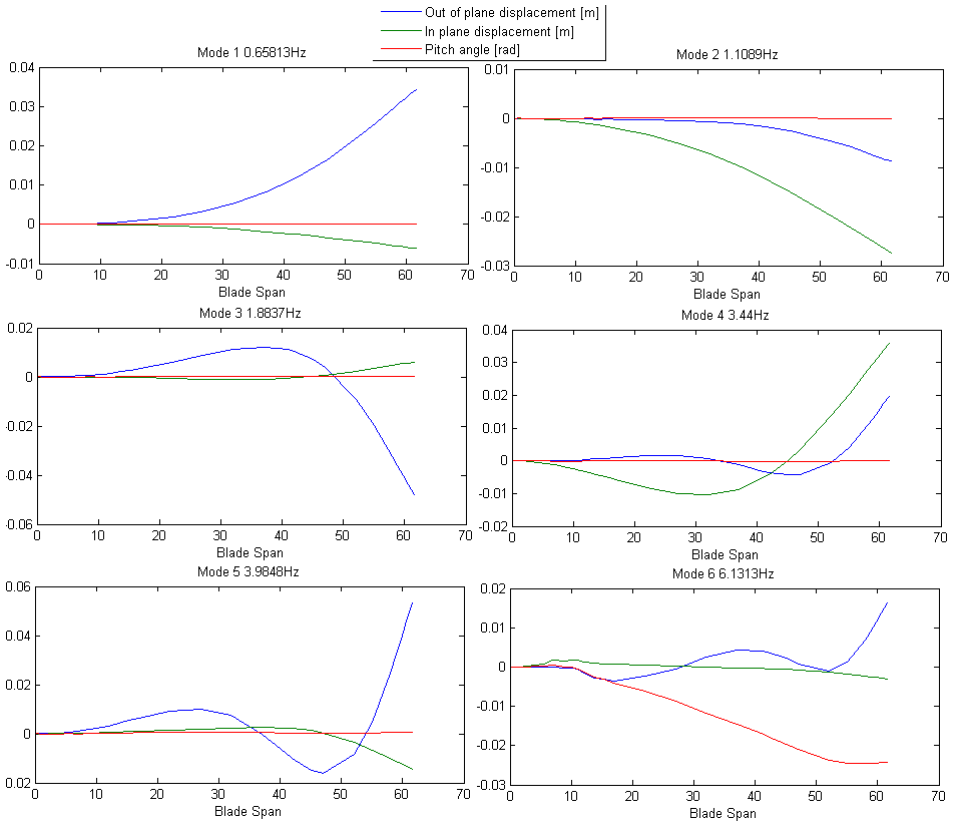


Figure 6.1.4: The first six eigenmodes of the blade.

6.2 Aerodynamic forces

By calculating the forces on the blade using BEM, HVM and FVM. The results from these can be compared to the forces on the blade calculated by a 3-D CFD model.

For all the below cases the 5 MW reference wind turbine has been used. The calculations are done on the steady case where the wind blows at 11.4 m/s and the rotor speed of 12.1 RPM.

6.2.1 Blade Element Momentum (BEM)

In figure 6.2.1 the forces calculated by BEM can be seen.

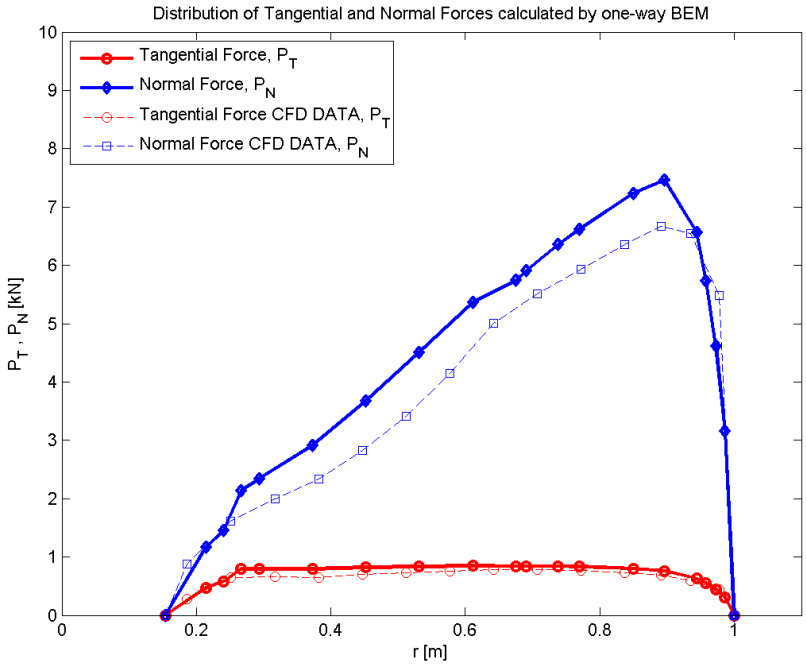


Figure 6.2.1: Forces calculated by BEM compared to forces from 3D CFD data.

As can be seen the BEM results are pretty similar to the CFD results. This means that for a static simulation BEM predicts the forces with almost as accurate as a CFD calculation. And the big difference is ofcourse that BEM takes less than one second on an ordinary PC, whereas the CFD calculation takes around 5 hours on 16 CPUs on a cluster.

6.2.2 Helical Vortex Method (HVM)

In figure 6.2.2 the forces calculated by the Helical Vortex Method is shown.

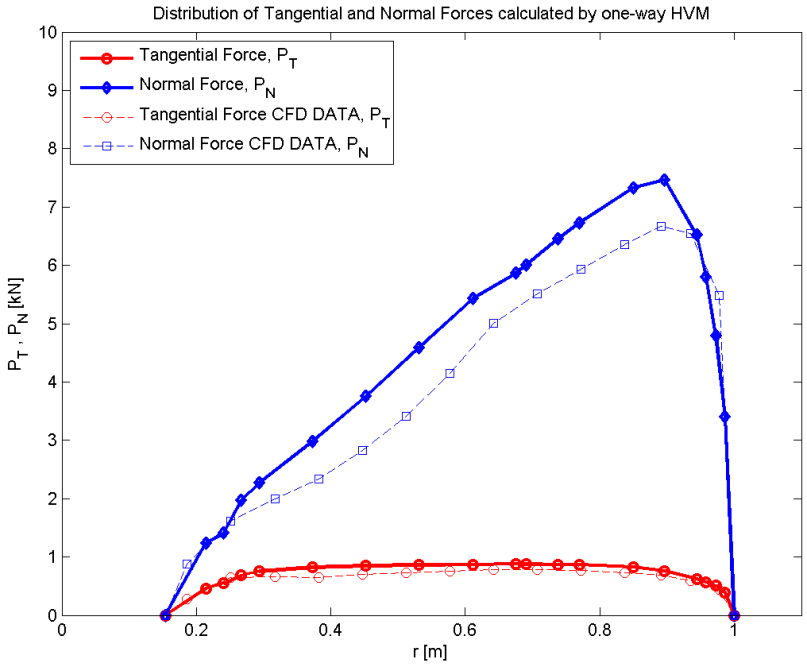


Figure 6.2.2: Forces calculated by Helical Vortex Method compared to 3D CFD data.

As can be seen the forces calculated by are also very close to the CFD results. The simulations take around 5-10 seconds on an ordinary PC.

6.2.3 Free Vortex Method (FVM)

As mentioned before the Free Vortex Method code still needs to be tuned for wind turbine applications. Therefore the FVM results are not nearly as good as they should and could be.

In figure 6.2.3 the forces calculated by the Free Vortex Method are shown.

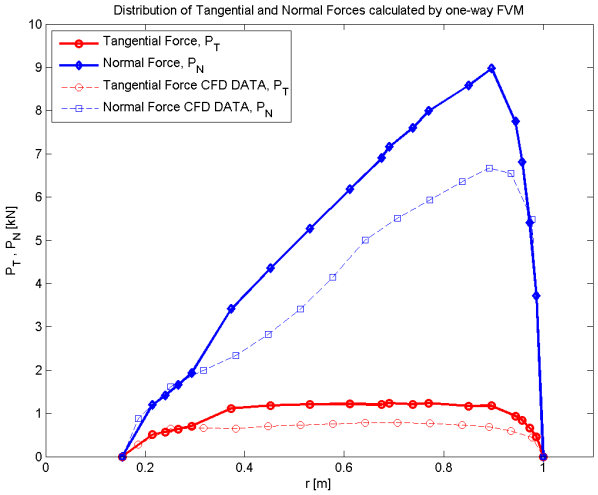


Figure 6.2.3: Forces calculated by FVM compared to forces from 3D CFD data.

Especially the over-prediction of the forces near the rotor tip are serious, since those forces have the most impact on predicted power and tip deflection.

The result above most likely have something to do with the model of only one free tip vortex, instead of several vortices. In figure 6.3.7 the Free Vortex code used here is compared to another vortex code from National Technical University of Athens. As can be seen, the Genuvp code does not over-predict the forces at the tip, however that code has a very long computational time.

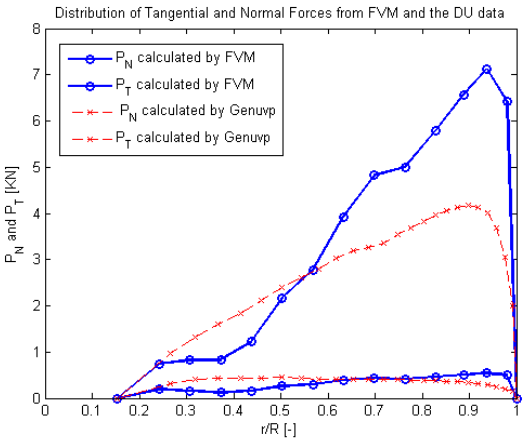


Figure 6.2.4: Forces calculated by FVM compared to forces from the Genuvp vortex particle method code from National Technical University of Athens.

In the above comparison the wind speed is 8 m/s and the rotating speed is 1.0032 rad/s.

6.2.4 Comparison between the methods

In figure 6.2.5 the different methods for calculating forces are compared to each other for a wind speed of 11.4 m/s.

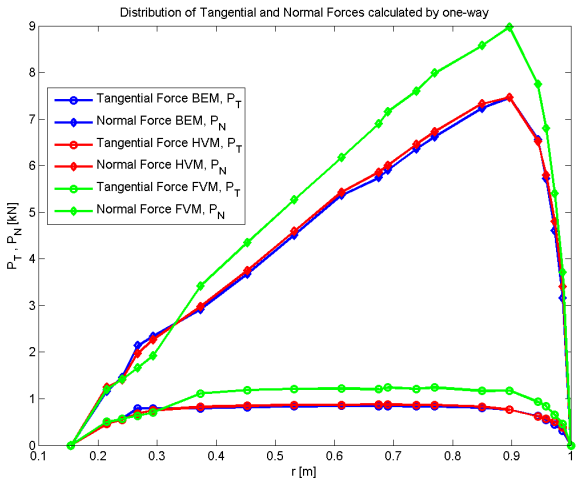


Figure 6.2.5: Forces calculated by BEM, compared to forces calculated by HVM and FVM.

As can be seen the forces calculated by BEM and HVM are very similar, HVM predicts a slightly higher force than the BEM at this wind speed. The FVM give much higher forces, which is probably because it is still tuned for helicopters.

In figure 6.2.6 the different methods for calculating forces are compared to each other for a wind speed of 19 m/s.

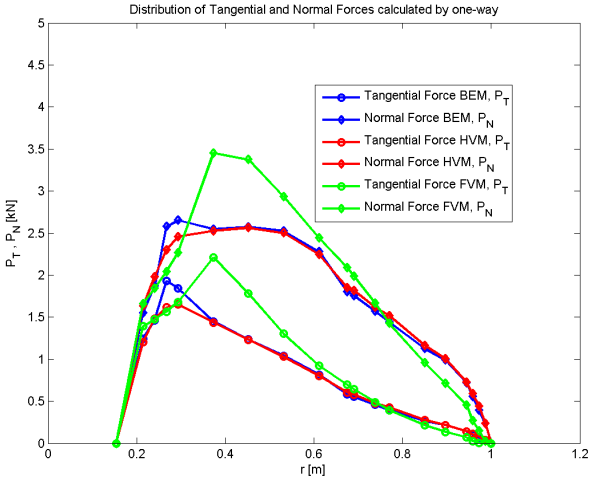


Figure 6.2.6: Forces calculated by BEM, compared to forces calculated by HVM and FVM.

For 19 m/s, at the tip all models predict similarly, where BEM and HVM predict almost the same forces, whereas FVM has a slightly lower force. At the root however some differences starts to show. The force calculated by FVM is much higher at the root then for the other BEM which in turn has quite a bit higher force then HVM.

6.3 Forces for Two Way Fluid Structure Interaction

6.3.1 Blade Element Momentum (BEM)

In figure 6.3.1 the forces calculated by the BEM, using both the two-way FSI method and the one-way FSI method.

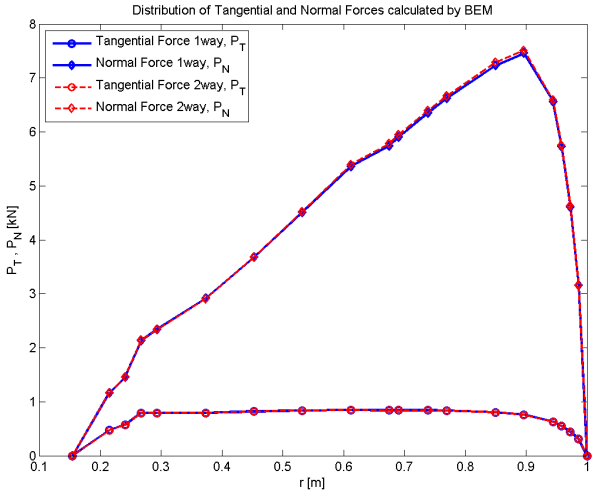


Figure 6.3.1: Forces calculated by BEM using the two-way FSI method compared to forces calculated using the one-way FSI method for a wind speed of 11.4 m/s.

As can be seen there is not much difference at all between the two-way and one-way FSI method. The difference comes mainly at the tip (which is not a surprise). However something that is a surpise is that the forces in the flap-wise direction, are higher for the two-way method, than they are for the one-way method.

In figure 6.3.2, the wind speed has been increased from 11.4 m/s in the previous figure, to 19 m/s.

At the higher wind speed it can be seen that the difference between the forces calculated by one-way and two-way FSI, are much bigger then previously.

The reason that there is a much bigger difference between the forces at higher wind speeds is most likely due to the increased power of the wind which creates a higher pitching force. The pitching force increases the twist and therefore the forces become smaller.

In figure 6.3.3 the forces calculated by two-way FSI is compared to the CFD simulation on a deformed blade.

As can be seen the CFD calculations have a much flatter curve at the tip of the blade, compared to the one-way FSI forces shown above. However nothing of the flattness can be seen in the BEM results. For this angle of attack, the difference does not matter as much as the difference is not that big. However, for higher angle of attacks this might play an important role.

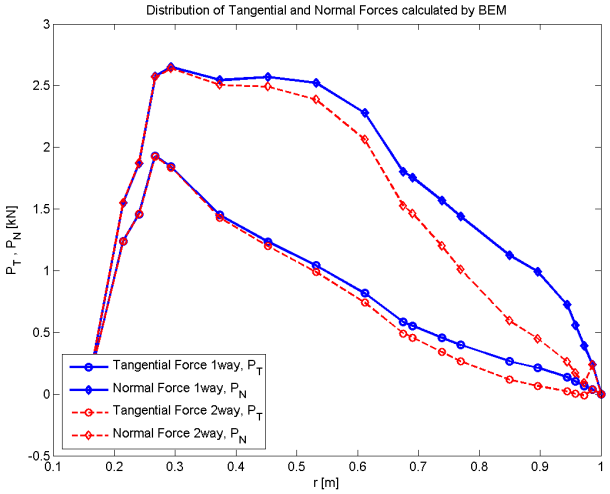


Figure 6.3.2: Forces calculated by BEM using the two-way FSI method compared to forces calculated using the one-way FSI method for a wind speed of 19 m/s.

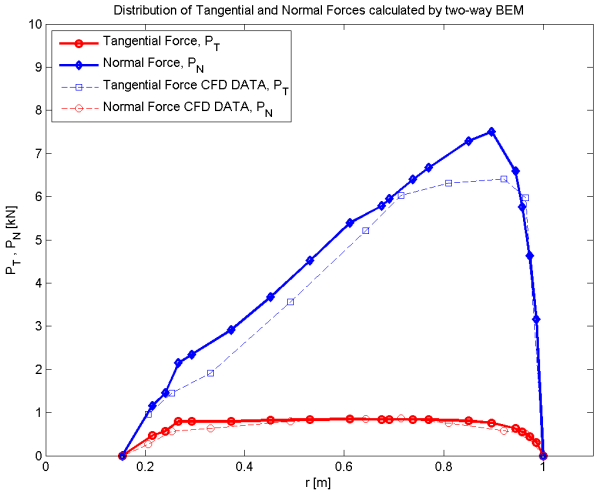


Figure 6.3.3: Forces calculated by BEM compared to forces from 3D CFD data.

6.3.2 Helical Vortex Method (HVM)

In figure 6.3.4 the forces calculated by the HVM, using both the two-way FSI method and the one-way FSI method.

As can be seen in the figure there is not much difference at all between the two-way and one-way FSI method. The difference comes mainly at the tip (which is not a surprise). For the HVM method the forces in the flap-wise direction are lower for the two-way method than they are for the one-way method. This was expected, however, the BEM results showed the opposite.

In figure 6.3.5, the wind speed has been increased from 11.4 m/s in the previous figure, to 19 m/s.

At the higher wind speed it can be seen that the difference between the forces calculated by one-way and two-way FSI, are much bigger then previous.

The reason that there is a much bigger difference between the forces at higher

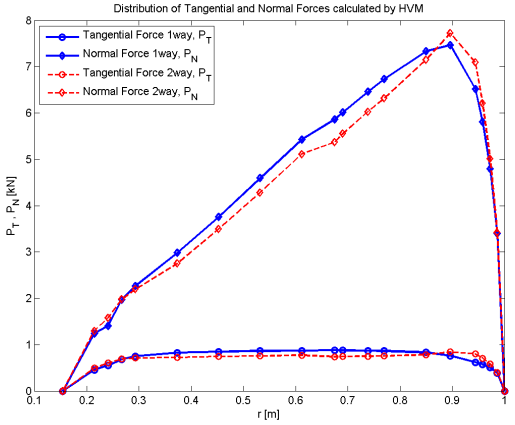


Figure 6.3.4: Forces calculated by HVM using the two-way FSI method compared to forces calculated using the one-way FSI method for a wind speed of 11.4 m/s.

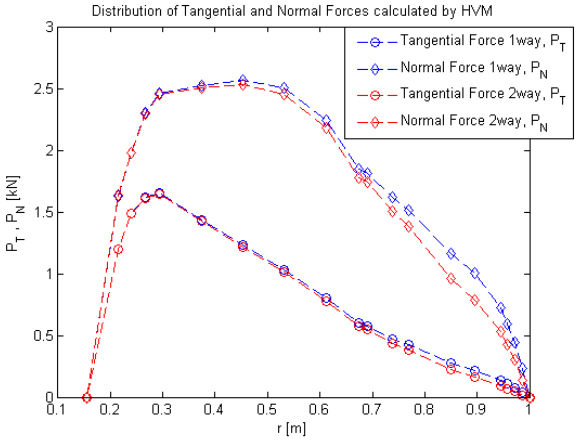


Figure 6.3.5: Forces calculated by HVM using the two-way FSI method compared to forces calculated using the one-way FSI method for a wind speed of 19 m/s.

wind speeds is most likely due to the increased power of the wind which creates a higher pitching force. The pitching force increases the twist and therefore the forces become smaller.

In the figure 6.3.9 the forces calculated by two-way FSI is compared to the CFD simulation on a deformed blade.

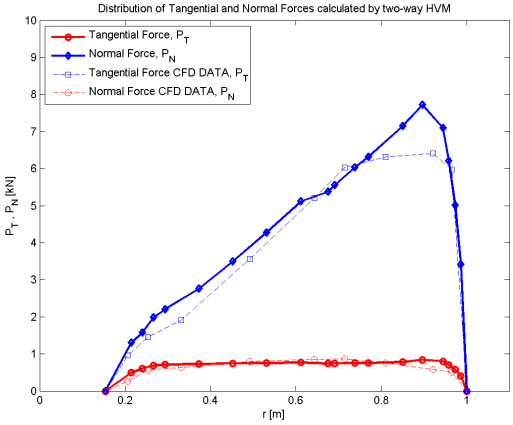


Figure 6.3.6: Forces calculated by HVM compared to forces from 3D CFD data.

6.3.3 Free Vortex Method (FVM)

In figure 6.3.7 below the forces calculated by the FVM, using both the two-way FSI method and the one-way FSI method.

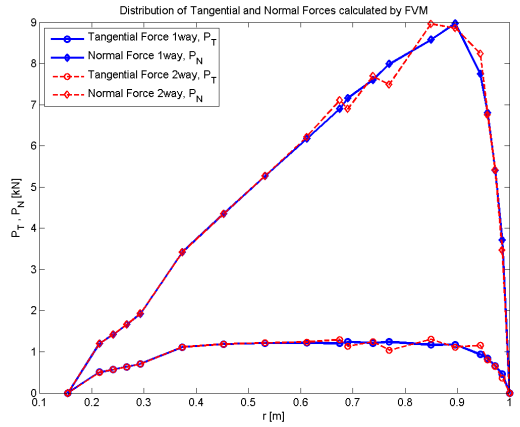


Figure 6.3.7: Forces calculated by FVM using the two-way FSI method compared to forces calculated using the one-way FSI method for a wind speed of 11.4 m/s.

As can be seen in the figure the difference between the two-way and one-way FSI method are very small. The difference comes mainly at the tip where the graph is very jagged for the two-way method.

In figure 6.3.8, the wind speed has been increased from 11.4 m/s in the previous figure, to 19 m/s.

As can be seen in the figure the difference between the forces calculated by one-way and two-way FSI, are much bigger at higher wind speeds.

In figure 6.3.9 the forces calculated by two-way FSI is compared to the CFD simulation on a deformed blade.

As can be seen the CFD calculations have, at the tip of the blade a much flatter curve, compared to the one-way FSI forces shown above. The results from FVM seems to try and flatten out at the tip but something is missing. For this angle of attack, the difference does not matter as much as the difference is not that big,

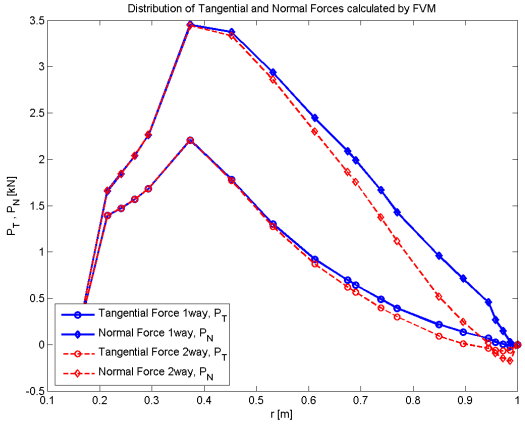


Figure 6.3.8: Forces calculated by HVM using the two-way FSI method compared to forces calculated using the one-way FSI method for a wind speed of 19 m/s.

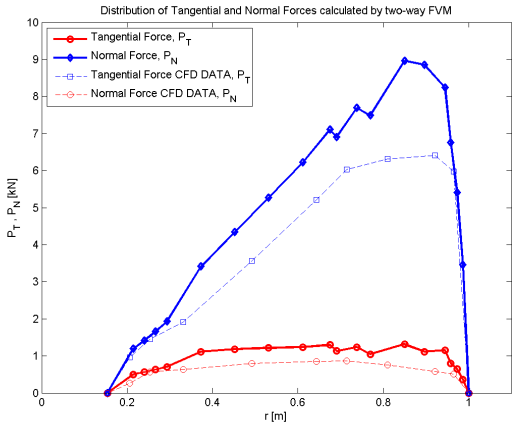


Figure 6.3.9: Forces calculated by FVM compared to forces from 3D CFD data.

However for higher angle of attacks this migth play an important role.

6.3.4 Comparison between the methods

In figure 6.3.10 the different methods for calculating forces are compared to each other for a wind speed of 11.4 m/s.

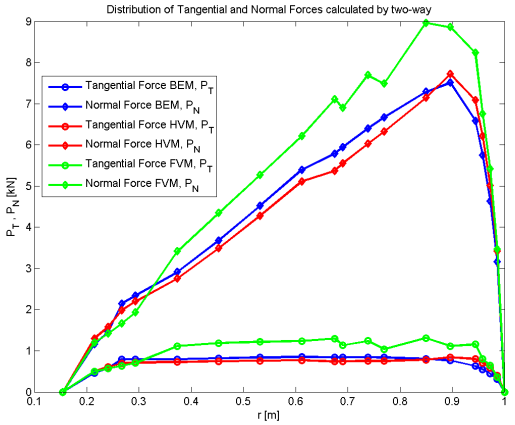


Figure 6.3.10: Forces calculated by BEM, compared to forces calculated by HVM and FVM.

For the two-way method there are some differences between the models at 11 m/s, in general HVM has a lower force than BEM, which is most likely due to the fact that the deformed geometry decreases the forces a bit. However HVM have a slightly higher maximum force. The FVM forces are still much higher than the BEM and HVM forces.

In figure 6.3.11 the different methods for calculating forces are compared to each other for a wind speed of 19 m/s.

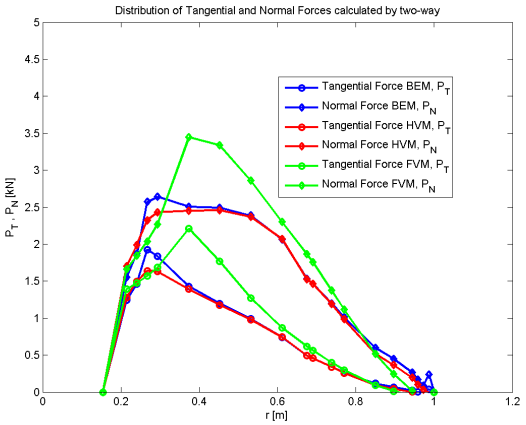


Figure 6.3.11: Forces calculated by BEM, compared to forces calculated by HVM and FVM.

For 19 m/s, at the tip there are some small differences. BEM predicts a slightly higher force than HVM, which in turn has a slightly higher force than FVM. At the root the models behave the same way as they did in one-way, i.e. FVM gives much higher forces than BEM which gives higher than HVM.

6.4 Pressure distribution

The pressure distribution that have been calculated using the method described in chapter 3.2.1 is compared to the pressure distribution calculated by the 3-D CFD model.

The results are calculated at the different radial positions where the blade is defined. In figure 6.4.1 the pressure distributions for the four sections which are closest to the centre are shown at the radial positions $r = 11.75, 15.85, 19.95, 24.05$.

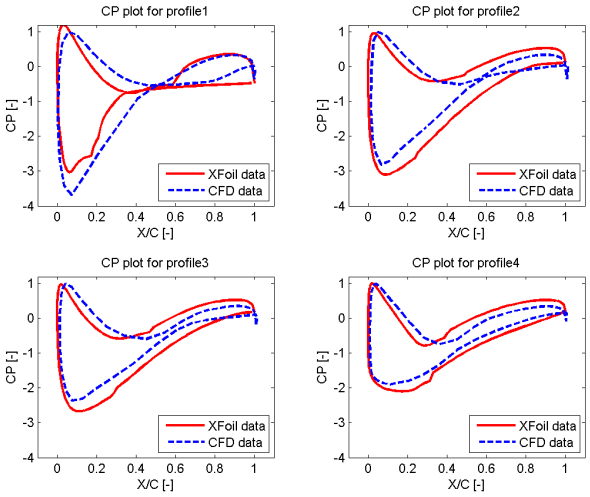


Figure 6.4.1: Pressures Calculated by Xfoil compared to 3D CFD pressures.

As can be seen the pressure distribution closest to the root is quite different from the CFD solution. However the further up the blade the better the results become.

In figure 6.4.2 the pressure distribution is shown for the radial positions $r = 28.15, 32.25, 36.35, 40.45$.

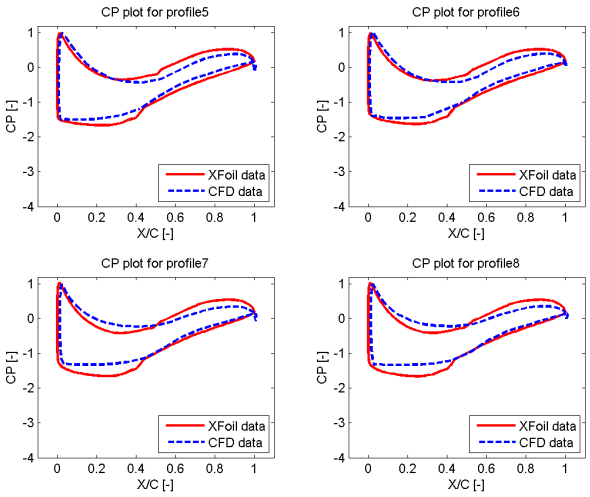


Figure 6.4.2: Pressures Calculated by Xfoil compared to 3D CFD pressures.

As can be seen in the figure above, the pressure at these radial position compares

quite well to the CFD results.

All the above pressure distributions have been for the different DU profiles, the DU profiles are harder to calculate on for both xfoil and CFD (especially the thicker ones at the root). In the figure 6.4.3 below the pressure distribution of the first NACA profiles can be seen, this is at the radial positions $r = 44.55, 48.65, 52.75, 56.16$.

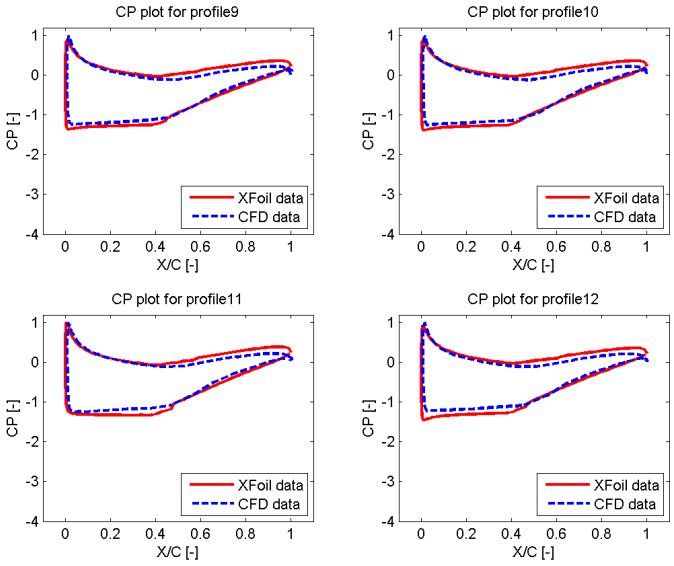


Figure 6.4.3: Pressures Calculated by Xfoil compared to 3D CFD pressures.

As can be seen now with the NACA profiles the different pressure distributions match almost perfectly with the CFD results.

However most likely due to the slightly bigger difference between the forces from the CFD result and the BEM result, the last two profiles does not have as good agreement between the different results. This can be seen in figure 6.4.4.

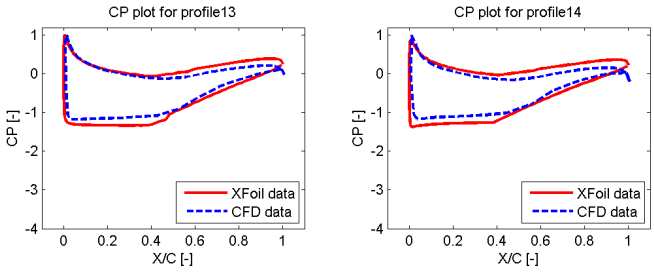


Figure 6.4.4: Pressures Calculated by Xfoil compared to 3D CFD pressures.

This means that in general the pressure distributions get better and better results, the further out on the blade they are calculated. This is a very good thing, because the further out on the blade, the higher the forces, and also the higher impact the force will have on the deflection, as well as on the power output.

All these pressure distributions are also applied to the structural model, which can be seen in figure 6.4.5 and figure 6.4.6.

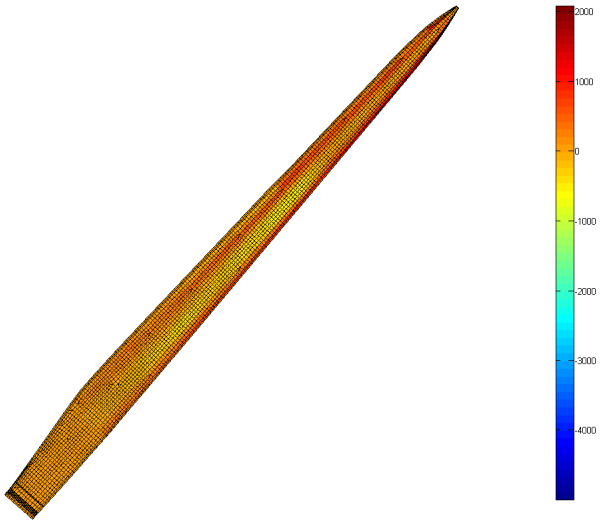


Figure 6.4.5: Pressures Calculated by Xfoil applied to the structural model.

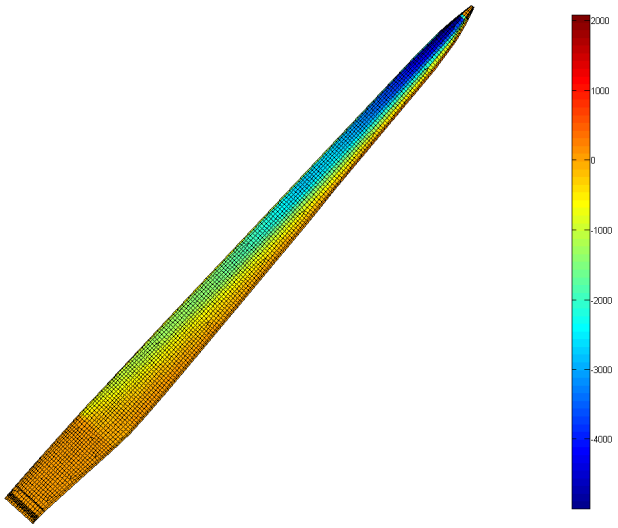


Figure 6.4.6: Pressures Calculated by Xfoil applied to the structural model.

6.5 Tip Deflection

When the pressure distribution in chapter 4.3 has been applied to the structural model, it is quite simple to calculate the deformation of the blade. The deformation of the blade when the incoming wind velocity is 11.4 m/s and the rotational speed Ω is 12.1 RPM can be seen in figure 6.5.1

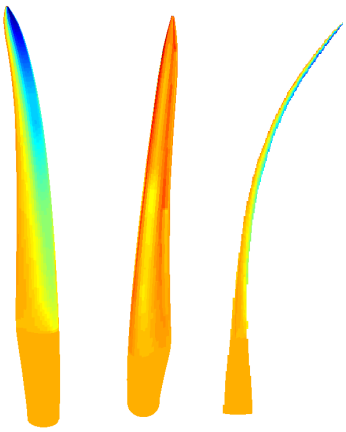


Figure 6.5.1: Pressures applied to the structural model creating a deformed blade.

Some interesting values that describe the deformation is the tip deflection in the edge-wise (DiED) and flap-wise direction (DiFD).

For the blade shown above, these values are DiED=-1.4 and DiFD=6.1.

The tip deflection has been calculated, not only for this specific wind speed, but for a range of different wind speeds. Similarly to the report [6], In that report the blade is pitched and has a varying rotational speed Ω depending on the wind velocity. To be able to compare the results the same values for pitch and Ω is used. These values can be seen in figure 6.5.2.

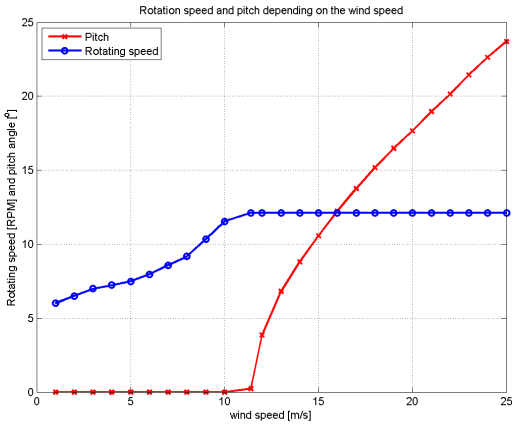


Figure 6.5.2: The pitch and the rotational speed of the blade (Ω for different incoming wind velocities.

These tip deflections have been calculated by the three different methods for calculating forces. For all three methods both one-way and two-way Fluid Structure Interaction have been used.

6.5.1 Blade Element Momentum (BEM)

The figure 6.5.3 shows how the tip deflection varies with the different wind velocities as described above. The forces are calculated using BEM.

In the figure there are six different lines, the three at the top describe the flap-wise tip deflection, and the three at the bottom describe the edgewise tip deflection.

The black lines shows the data from the Jonkman et. al. report [6]. The red lines show the deflection when calculated by one-way FSI. The blue lines show the deflection when using two-way FSI.

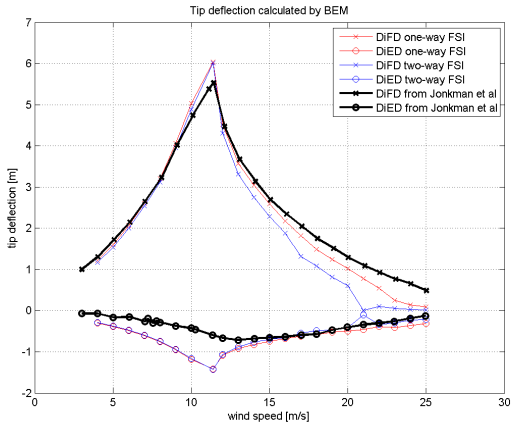


Figure 6.5.3: The tip deflection using our model and calculating forces with BEM compared to Jonkman et al report DiFD = Deflection in Flapwise Direction, and DiED = Deflection in Edgewise Direction.

As can be seen in the figure the flap-wise tip deflection follows the Jonkman results quite well until, at a wind speed of about 15 m/s. The edgewise deflection is way too high at the low wind speeds and becomes good first after 15 m/s.

The reason for the low tip deflection in the flap-wise direction at high wind speeds probably comes from the fact that the torque and thrust generated at these velocities are different than what Jonkman had. The torque and the thrust for the different wind velocities can be seen in the figure 6.5.4 below.

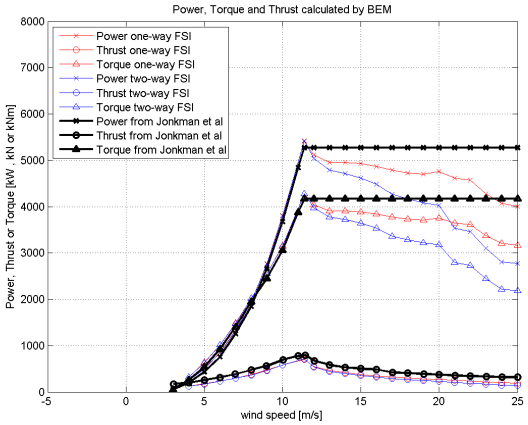


Figure 6.5.4: The torque and thrust calculated by BEM, compared to the Jonkman et al report.

The reason for the high edgewise tip deflection can most likely be found in the structural model of the blade. Where compared to the Jonkman blade [6], the strength toward deformation in the edgewise direction is smaller. As expected the tip deflection when using two-way FSI is smaller than when using one-way FSI, this is especially true for the high wind speeds.

6.5.2 Helical Vortex Method (HVM)

The figure 6.5.5 below shows how the tip deflection varies with the different wind velocities as described above. The forces are calculated using HVM.

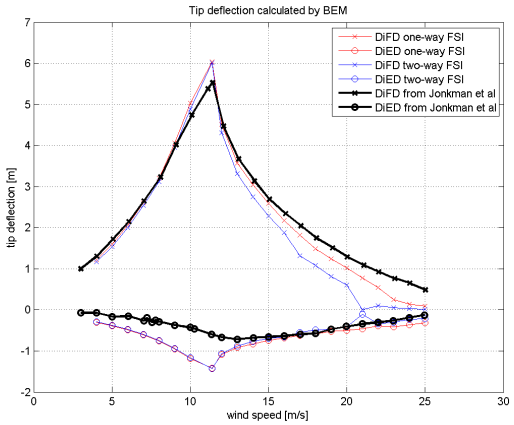


Figure 6.5.5: The tip deflection using our model and calculating forces with HVM compared to Jonkman et al report DiFD = Deflection in Flapwise Direction, and DiED = Deflection in Edgewise Direction.

As can be seen in the figure the results are almost identical to the BEM results. The reasons are the same as for the tip deflections when the force was calculated by the BEM code.

The torque and the thrust for the different wind velocities can be seen in the figure 6.5.6 below.

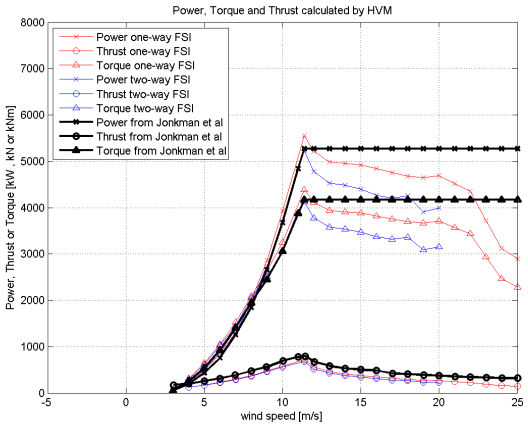


Figure 6.5.6: The torque and thrust calculated by HVM, compared to the Jonkman et al report.

6.5.3 Free Vortex Method (FVM))

The figure below shows how the tip deflection varies with different wind speeds when using FVM to calculate the forces.

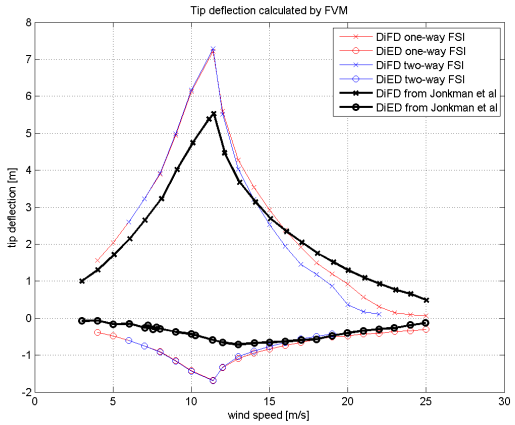


Figure 6.5.7: The tip deflection using our mode and calculating forces with FVM compared to Jonkman et al report DiFD = Deflection in Flapwise Direction, and DiED = Deflection in Edgewise Direction.

As can be seen in the figure, the tip deflections calculated by FVM are way off. At low wind speed the deflectionsa are way too high and at about 15 m/s the flap-wise deflection starts getting too low.

The reasons for these results are probably due to the extremely high over-prediction of the power, torque and thrust that the FVM code does at low wind speeds, and the eqxtreme under-prediciton of the power at high wind speeds. This can be seen in the figure 6.5.8 below.

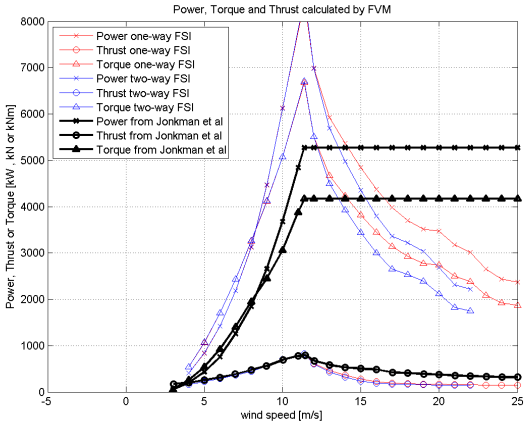


Figure 6.5.8: The torque and thrust calculated by FVM, compared to the Jonkman et al report.

6.6 Results of aeroelastic analysis

To test the aeroelastic code, we did a number of analyses. In the following figures, the frequency is defined as the absolute value of the complex eigenvalue $\omega_n = |\lambda|$, and the damping ratio as $\zeta = \frac{Re(\lambda)}{|\lambda|}$. In addition to the modes shown here there were modes appearing that had very high damping and a small imaginary part of their eigenvalues. The physical meaning of these modes, if any, are not clear.

The first analysis is one where the free stream velocity is varied, and the pitch and rotational velocity is set according to [6]. The resulting aeroelastic damping ratios and aeroelastic frequencies vs wind speed for the first and second flapwise modes are shown in figure 6.6.1. As expected, within moderate windspeeds the flapwise modes are highly damped. Included in the figure are the results of a three-blade analysis by Jonkman in [6]. In the case of a three-blade analysis there are the collective, forward whirl (progressive) and backward whirl (regressive) modes. These modes include centrifugal stiffening, gyroscopic effects and interaction with the tower and nacelle, so a 100 % agreement is not achievable. We only include them to see if there is a general trend common in the results. In comparison to Jonkmans results, the single blade analysis seems to follow the damping curve up to 12 m/s. There is a peak in both the frequency and damping curve starting after 12 m/s, that is not seen in Jonkmans results.

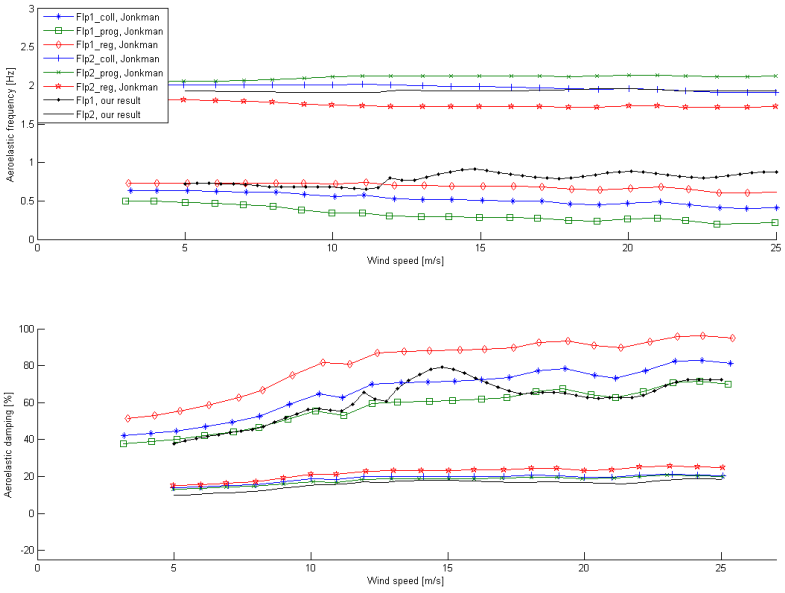


Figure 6.6.1: Aeroelastic frequency and damping in our analysis compared to those reported by Jonkman.

For the same case, we plot the damping in the first edgewise mode together with those in the lag modes by Jonkman.

In the second analysis, the free stream velocity is set to zero, and the rotational velocity is varied. This gives small angles of attack and is closer to classical flutter problem than the first analysis. The twist and pitch angle where set to zero, so the flow angle is zero during the whole analysis.

The results are shown in figure 6.6.3, and compared to those by Hansen in [7]. The damping in this case is the logarithmic increment $2\pi\frac{|\lambda|}{Im(\lambda)}$. Hansen's analysis is

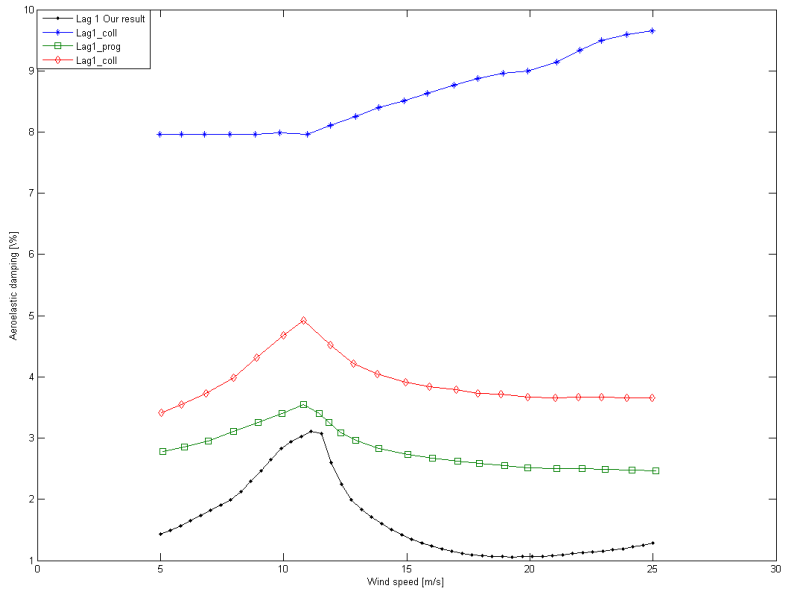


Figure 6.6.2: Aeroelastic frequency and damping in our analysis compared to the results by Jonkman

done with the tool HAWCstab which uses beam elements and, similar to our model, a modified Beddoes–Leishman dynamic stall model . In our case, we have carried out the analysis up to 35 RPM while in the Hansen analysis the goes to 30 RPM. The overall qualitative behaviour in the frequency curves is similar for both analyses. We see a flutter mode with a frequency 4.5 Hz at about 25 RPM, involving the third flapwise mode and the torsional mode.

The damping curves agrees well, at least at the lower rotor speeds. It is not completely clear if the cause of the discrepancies one can see is an error in the aeroelastic analysis in itself, or in differences in the blade structural model. The blade structural model differs in several crucial ways. For instance, the center of gravity is closer to the leading edge in our model than it is for Hansen. Another perhaps even more important point is that the first torsional eigenfrequency in the Hansen analysis is 8 Hz, while it in our case it is just over 6 Hz. On one hand, a higher torsional frequency in the Hansen case should mean that his blade is more stable, on the other the 1st torsional is closer to a flapwise frequency in his case, while in our the frequencies are more separated which increases stability. We see that the fourth flapwise mode is not very affected, possibly because it has a higher eigenfrequency than the torsional mode. Another problem could be that the relative flow angle is zero in our case, while the blade probaly was probably at least slightly twisted in Hansens analysis. As shown earilier, the relative flow angle is important.

To get an additional verification of the aeroelastic code, the beam model was also analyzed. The resulting aeroelastic frequency and damping can be seen in the figures below. In the pre-flutter range, we see a very good agreement in the frequency curves of the first torsional and first flapwise modes. The flutter frequency is about equal in both cases, and happens at almost exactly the rame rotor speed. However the free frequency of the third flapwise mode is higher and it no longer couples to the torsional mode, possibly because there are coupling terms missing in this very simple model.

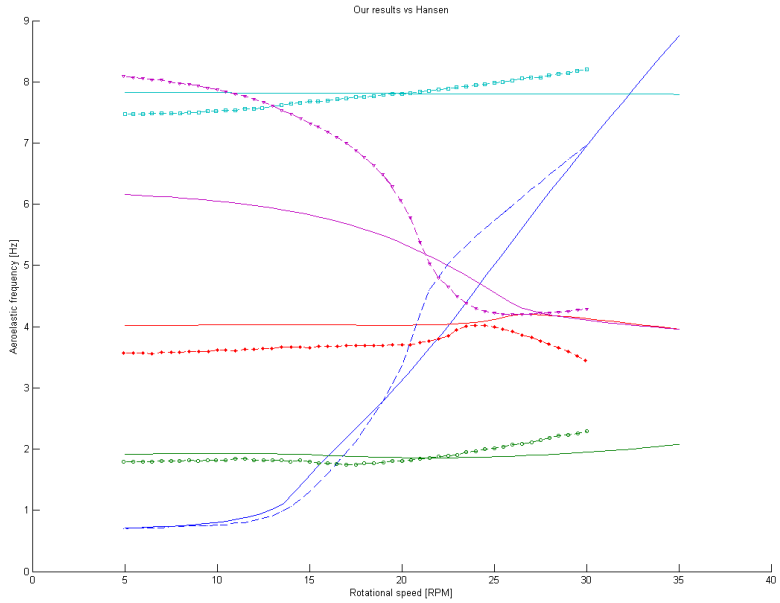


Figure 6.6.3: Aeroelastic frequency in our shell model analysis compared to those reported by Hansen.

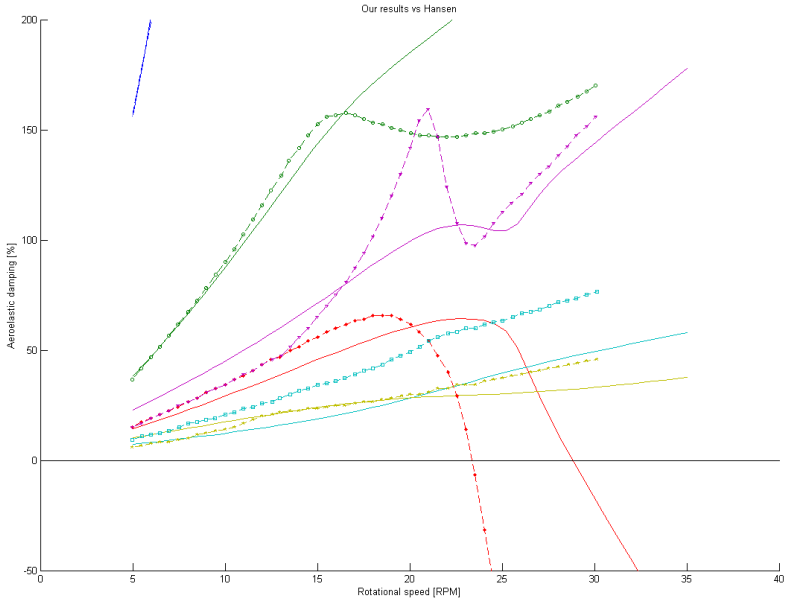


Figure 6.6.4: Aeroelastic damping in our shell model analysis compared to those reported by Hansen.

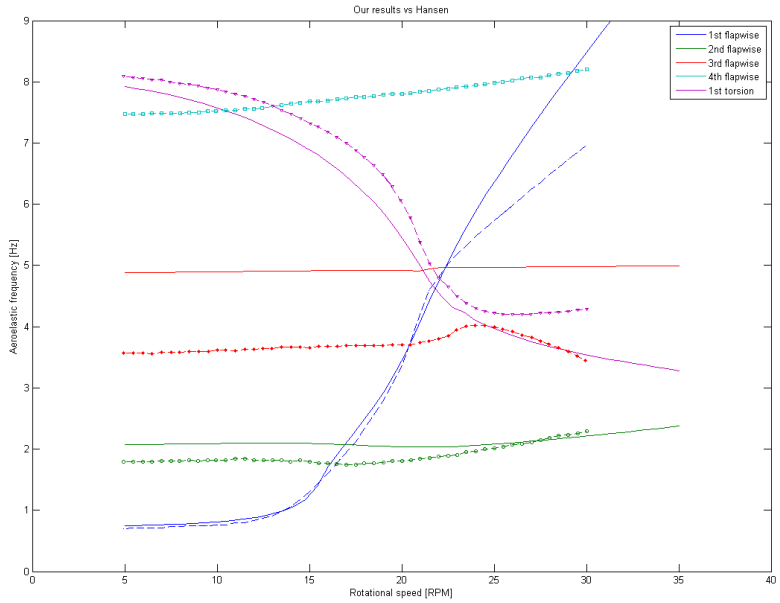


Figure 6.6.5: Aeroelastic frequency in our analysis with a beam model compared to those reported by Hansen.

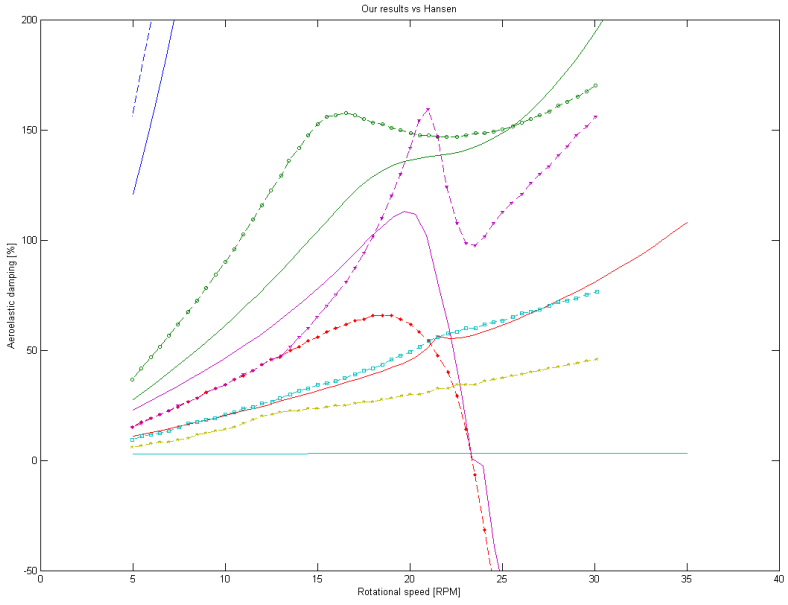


Figure 6.6.6: Aeroelastic frequency in our analysis with a beam model compared to those reported by Hansen.

7 Summary and Conclusions

- A method to automatically create a finite element model of a rotor blade from a parametrized description has been developed.
- A way to do aeroelastic calculatons on a finite element shell model has been successfully developed.
- For one-way Fluid Structure Interaction BEM is the best code to use. Especially when calculating on rather unphysical cases like one would when doing flutter analysis.
- At low windspeeds the difference between one-way and two-way FSI is really small. However for higher wind speeds (> 15 m/s) the effects of deformation of the blade really starts to show.
- For two-way FSI, BEM has the benefit of working for most cases, whereas with the current code, HVM and FVM might have convergency issues. This happens mostly when the pitch is too high so that the tip experiences negative lift.

  When doing flutter analysis on a two-way FSI blade BEM is definately prefered.

  For a general case where one wants to know the deformation HVM might be better.

8 Suggestions for future research

An aeroelastic analysis that involves just a single blade is unconservative in comparison to an analysis including the full turbine system, i.e. all three blades, tower, drivetrain etc that includes the effects of rotation on the blades. It would therefore be of value to do a model that includes these things. One possibility is to use the Rotordyn package in NX NASTRAN, or a MATLAB program based on rotor dynamics. A good report on this is [20]. The tower and shaft could possibly be modeled with beam elements. A program that writes the stiffness, mass and damping matrices for three blades, including centrifugal and gyroscopic effects is already done in this thesis, but further work is necessary.

The state-space formulation of the aeroelasitc analysis lends itself well to model reduction. A reduced model could be used to test a number of different excitations, like wind shear, tower shadow effects, different forms of turbulence etc. One could include a number of wind excitations based on those in IEC, to test if a new blade design is up to code. The Risø model used in the stability analysis has a term for varying free stream velocity that we have omitted.

The thesis has shown that BEM gives a good accuracy for steady simulations, rendering the use of the vortex methods rather unnecessary from a Fluid-Structure Interaction point-of-view. However for unsteady simulations the effects of BEM vs the vortex methods have not been tested. Therefore further development of a vortex code for unsteady simulations would be interesting. The benefits of using a vortex method for unsteady cases would most likely be that the wake characteristics would be more accurately calculated (how it behaves). When using vortex methods for unsteady simulations the effect of other wind turbines etc would also be possible to calculate.

Bibliography

[1] Bo Carlsson, Per Wärn, Åke Nylinder, Kåre Ljung, Göran Marström, et al. Suitability of carbon fibre reinforced plastic, cfrp, versus glass fibre reinforced plastic, gfrp, in wind turbine blades – a total cost analysis. Technical report, University of Kalmar, 2009.

[2] P.K Mallick. *Fiber-Reinforced Composites*. CRC Press, 2008.

[3] M. H. Hansen. Aeroelastic instability problems for wind turbines. Technical report, Risø National Laboratory, 2007.

[4] C. Lindenburg. Aeroelastic modelling of the lmh64-5 blade. Technical report, Dowec, 2002.

[5] Brian Resor. Integrated design and system analysis at sandia. 2010. 2010 Sandia Wind Turbine Blade Workshop.

[6] J. Jonkman, S. Butterfield, W. Musial, and G. Scott. Definition of a 5 mw reference wind turbine for offshore system developement. Technical report, National Renewable Energy Laboratory, 2009.

[7] G. Bir and J. Jonkman. Aeroelastic instabilities of large offshore turbines. *Journal of Physics: Conference Series 75 (2007) 012069*, 2007.

[8] D. Todd Griffith and Thomas D. Ashwill. The sandia 100-meter all-glass baseline wind turbine blade. Technical report, Sandia National Laboratories, 2011.

[9] Hamidreza Abedi. Aerodynamic loads on rotor blades. Master's thesis, Chalmers University of Technology, 2011.

[10] Martin O. L. Hansen. *Aerodynamics of Wind Turbines*. Earthscan, 2008.

[11] Jr John D. Anderson. *Fundamentals of Aerodynamics*. McGraw-Hill, 2001.

[12] Horia Dumitrescu and Vladimir Cardos. Wind trubine aerodynamic performance by lifting line method. Technical report, Institute of Applied Mathematics, Romanian Academy, 1998.

[13] Ashish Bagain and J. Gordon Leishman. Rotor free-wake modeling using a pseudoimplicit relaxation algorithm. Technical report, 1995.

[14] Ashish Bagain and J. Gordon Leishman. Rotor free-wake modeling using a pseudo-implicit technique - including comparisons with experimental data. Technical report, 1994.

[15] Mark Drela. Xfoil user guide, 2001. accessed 25-may-2012.

[16] Martin O.L Hansen, J.N Sorensen, S Voutsinus, N Sorensen, and H.Aa Madsen. State of the art in wind turbine aerodynamics and aeroelasticity. Technical report, Risø National Laboratory, 2006.

[17] Theodore Theodorsen. General theory of aerodynamic instability and the mechanism of flutter. Technical report, NACA, 1935.

[18] P.K. Chaviaropoulos. Flap/lead-lag aeroelastic stability of wind turbine blades. Technical report, Applied Research and Technology Development Department, Center of Renewable Energy Sources, 2001.

[19] Morten Hartvig Hansen, Mac Gaunaa, and Helge Aagaard Madsen. A beddoes-leishmann type dynamic stall model in state-space and indical formulations. Technical report, Risø National Laboratory, 2004.

[20] M. H. Hansen. Improved modal dynamics of wind turbines to avoid stall-induced vibrations. *Wind Energ., 6: 179-195. doi: 10.1002/we.79*, 2003.

A Appendix

A.1 List of scripts & Files

Following is a list and description of the files that make up the structural part of this thesis.

JNCJ_Blade.m an example of how a blade cell can be defined, in this case the NREL 5 MW blade with our laminate properties.

JNCJ_LayupCalc.m help function that calculates the laminate A, B and D-matrices to use in crosec for the cross sectional displacements. Uses equation from [2]

JNCJ_FemapInput.m a function that writes files readable to the FEMAP script. It also calculates equivalent sectional beam properties with crosec.

JNCJ_StiffnessAndMassMatrix.m reads the stiffness and mass matrix, on the NX Nastran output4 form, into MATLAB.

JNCJ_ElementConnecticity.m reads topological information from FEMAP, i.e. element connectivity and nodal coordinates, output by JNCJ_FemapOutputCoord.BAS.

JNCJ_AreaNormals.m function to calculate areas of the elements

JNCJ_FindNodesOnZero.m constrains nodes on z = 0, and creates list of free and constrained nodes.

JNCJ_SectionalFEM.m calculates the matrix Msec_fem, that determines sectional displacements from nodal displacements, according to equation (3.3.9).

JNCJ_DampingMatrix.m creates a modal damping matrix with given damping ratio, from equation (2.3.16).

JNCJ_Gravity.m creates the force vector under gravity

JNCJ_RisoCoord.m creates a matrix that gets the effective angle of attack, quarter chord downwash angle etc. , from the sectional displacements and stall variables

JNCJ_SSeom.m formulates the eom on state-space form, including the stall variables in the state vector, but without any cross coupling due to linearized force.

JNCJ_AerodynCoeffLin.m linearizes the aerodynamic coefficients as functions of the displacements and stall variables.

JNCJ_dfdC.m creates nodal forces from aerodynamic coefficients

JNCJ_FullAeroElasticAmatrix.m makes the final linearization and combines with the equation from modal_reduction_1 .

JNCJ_AeroElastic.m List of commands to make an aeroelastic analysis

JNCJ_ModesByImag.m After an aeroelastic eigenvalue analysis, this program sorts modes by imaginary part of eigenvalue

JNCJ_ReadCrosec.m reads results from the program Crosec

JNCJ_PlotCompare1.m Comparison of cross sectional propeties with DOWEC data 1

JNCJ_PlotCompare2.m Comparison of cross sectional propeties with DOWEC data 2

JNCJ_FEMAP.BAS script in FEMAP API that reads the files written by JNCJ_FemapInput.m, creates and meshes a model of the blade, meshes it and creates a number of analyses

JNCJ_FemapOutputCoord.BAS script in FEMAP API that outputs element connectivity and nodal coordinates to be read by JNCJ_ElementConnectivity.m

JNCJ_CalculatePressures.m The main pressure distribution function, depending on what data are available, and the angle of attack, this program calls different functions that calculate the pressure distribution

JNCJ_CalculatePD_High.m Itterative method for calculating the pressure distribution for high angle of attacks

JNCJ_CalculatePD_Low.m Itterative method for calculating the pressure distribution for low angle of attacks

JNCJ_LoadPresDist.m Loads and interpolates a pressure distribution which has the same value of C_L / C_D as the fluid force calculation

JNCJ_StorePresDist.m A program that stores alot of pressure distributions for a given airfoil, should be used when a new airfoil is used and no data for JNCJ_LoadPresDist.m exists

JNCJ_onewayFSIBEM.m BEM program that calculates the forces on a windturbine, depending on settings the JNCJ_CalculatePressures.m program can be called, and the deformation of the blade can be calculated using the program JNCJ_CalculateDeflection.m

JNCJ_onewayFSIHVM.m HVM program that calculates the forces on a windturbine, depending on settings the JNCJ_CalculatePressures.m program can be called, and the deformation of the blade can be calculated using the program JNCJ_CalculateDeflection.m

JNCJ_onewayFSIFVM.m FVM program that calculates the forces on a windturbine, depending on settings the JNCJ_CalculatePressures.m program can be called, and the deformation of the blade can be calculated using the program JNCJ_CalculateDeflection.m

JNCJ_twowayFSIBEM.m BEM program that calculates the forces and the deformation on a windturbine, Where the force and deformation is calculated for a deformed blade, The program makes use of JNCJ_CalculatePressures.m and JNCJ_CalculateDeflection.m

JNCJ_twowayFSIHVM.m HVM program that calculates the forces and the deformation on a windturbine, Where the force and deformation is calculated for a deformed blade, The program makes use of JNCJ_CalculatePressures.m and JNCJ_CalculateDeflection.m

JNCJ_twowayFSIFVM.m FVM program that calculates the forces and the deformation on a windturbine, Where the force and deformation is calculated for a deformed blade, The program makes use of JNCJ_CalculatePressures.m and JNCJ_CalculateDeflection.m

JNCJ_readFluidProfiles.m Reads the geometry that has been created for the solid model, and outputs the data ready to be used by any of the fluid programs (onewayBEM, onewayHVM, onewayFVM, twowayBEM, twowayHVM and twowayFVM)

xfoil.exe Receives commands from either JNCJ_StorePresDist.m, JNCJ_CalculatePD7.m or JNCJ_CalculatePD3.m and calculates a pressure distribution, and the C_L, C_D and C_M that that pressure distribution gives

A.2 A typical command file for xfoil

```
PLOP
G

LOAD NACA64_618.dat
PANE
MDES
FILT 1.00
EXEC
EXEC
EXEC
EXEC
EXEC
PANE
XYCM
0.25
0
OPER
VPAR
N 9.00
VACC 0.0100
XTR
1.0000
1.0000

VISC 10148276.1718
MACH 1.071419
ITER 1000
pacc
Polar1.txt

CL
1.0248
CPWR PressureBlade1.txt
PACC

QUIT
```

A.3 Example of blade cell

```
% Name of Blade, this will be the name of
% the folder where data is saved
Blade.Name = 'NREL5';

% no sections used to create FE-model
nSect = 34 ;
Blade.nsections = nSect;

% no sections used in fluid model
Blade.nFluidSections = 19;

% Definition of materials used in blade
%     mat. no. ->
% row
% 1     E_L
% 2     E_T
% 3     G_LT
% 4     ny_LT
% 5     rho
Blade.MatlInBlade = [41.8*1e9 13.6*1e9 27.7*1e9 0.256*1e9 3.5*1e9;
                     14.0*1e9 13.3*1e9 11.8*1e9 0.256*1e9 2.5*1e9;
                      2.63*1e9 11.8*1e9 7.2*1e9 0.022*1e9 1.4*1e9;
                      0.28     0.5      0.39    0.3       0.3
                      1920     1780     1780    200       1100];

% Use of material
Blade.SparMatl = 1;
Blade.ShearMatl = 2;
Blade.SkinMatl = 3;
Blade.FoamMatl = 4;


Esp = Blade.MatlInBlade(1,Blade.SparMatl);
Gsp = Blade.MatlInBlade(3,Blade.SparMatl);
rsp = Blade.MatlInBlade(5,Blade.SparMatl);

Esh = Blade.MatlInBlade(1,Blade.ShearMatl);
Gsh = Blade.MatlInBlade(3,Blade.ShearMatl);
rsh = Blade.MatlInBlade(5,Blade.ShearMatl);

Eskin = Blade.MatlInBlade(1,Blade.SkinMatl);
Gskin = Blade.MatlInBlade(3,Blade.SkinMatl);
rskin = Blade.MatlInBlade(5,Blade.SkinMatl);

% Radial position of sections
Zs(1:nSect) = ...
    [0.000 0.005 0.007 0.009 0.011 ...
     0.013 0.024 0.026 0.047 0.068 ...
     0.089 0.114 0.146 0.163 0.179 ...
     0.195 0.222 0.249 0.276 0.358 ...
     0.439 0.520 0.602 0.667 0.683 ...
     0.732 0.764 0.846 0.894 0.943 ...
     0.957 0.972 0.986 1.000 ]*61.5;
```

```
% Chord length of sections
Chord(1:nSect)= ...
    [3.5018 3.5018 3.5018 3.5018 3.5018 ...
     3.5018 3.5621 3.5738 3.7257 3.8770 ...
     4.0289 4.2035 4.4372 4.5535 4.6445 ...
     4.6912 4.6648 4.6051 4.5184 4.2576 ...
     3.9538 3.6377 3.3315 3.0867 3.0258 ...
     2.8419 2.7195 2.4139 2.2257 1.7368 ...
     1.4606 1.1291 0.7429 0.3075];

% Twist angle of chord line relative to rotor plane
TwistAng(1:nSect) = ...
    [13.308 13.308 13.308 13.308 13.308 ...
     13.308 13.308 13.308 13.308 13.308 ...
     13.308 13.308 13.308 13.177 13.046 ...
     12.915 12.133 11.350 10.568 9.166 ...
     7.688 6.180 4.743 3.633 3.383 ...
     2.735 2.348 1.380 0.799 0.280 ...
     0.210 0.140 0.070 0.000]*pi/180;

% Fractional pitch
Xpitch(1:nSect)= ...
    [0.500 0.500 0.500 0.500 0.500 ...
     0.500 0.499 0.498 0.483 0.468 ...
     0.453 0.435 0.410 0.400 0.390 ...
     0.380 0.378 0.377 0.375 0.375 ...
     0.375 0.375 0.375 0.375 0.375 ...
     0.375 0.375 0.375 0.375 0.375 ...
     0.375 0.375 0.375 0.375  ];

% Create transitional profile types
PT3 = transitP(Zs(3), Zs(2),Zs(16),{'Cylinder'},{'DU40_A17'});
PT4 = transitP(Zs(4), Zs(2),Zs(16),{'Cylinder'},{'DU40_A17'});
PT5 = transitP(Zs(5), Zs(2),Zs(16),{'Cylinder'},{'DU40_A17'});
PT6 = transitP(Zs(6), Zs(2),Zs(16),{'Cylinder'},{'DU40_A17'});
PT7 = transitP(Zs(7), Zs(2),Zs(16),{'Cylinder'},{'DU40_A17'});
PT8 = transitP(Zs(8), Zs(2),Zs(16),{'Cylinder'},{'DU40_A17'});
PT9 = transitP(Zs(9), Zs(2),Zs(16),{'Cylinder'},{'DU40_A17'});
PT10 = transitP(Zs(10),Zs(2),Zs(16),{'Cylinder'},{'DU40_A17'});
PT11 = transitP(Zs(11),Zs(2),Zs(16),{'Cylinder'},{'DU40_A17'});
PT12 = transitP(Zs(12),Zs(2),Zs(16),{'Cylinder'},{'DU40_A17'});
PT13 = transitP(Zs(13),Zs(2),Zs(16),{'Cylinder'},{'DU40_A17'});
PT14 = transitP(Zs(14),Zs(2),Zs(16),{'Cylinder'},{'DU40_A17'});
PT15 = transitP(Zs(15),Zs(2),Zs(16),{'Cylinder'},{'DU40_A17'});

PT3 = PT3{1}; PT4 = PT4{1}; PT5 = PT5{1}; PT6 = PT6{1};
PT7 = PT7{1}; PT8 = PT8{1}; PT9 = PT9{1}; PT10 = PT10{1};
PT11 = PT11{1}; PT12 = PT12{1}; PT13 = PT13{1}; PT14 = PT14{1};
PT15 = PT15{1};

% Assign profile types
ProfileType(1:nSect) = ...
   {'Cylinder'    'Cylinder'     ...
    PT3 ...
    PT4 ...
```

```
    PT5 ...
    PT6 ...
    PT7 ...
    PT8 ...
    PT9 ...
    PT10 ...
    PT11 ...
    PT12 ...
    PT13 ...
    PT14 ...
    PT15 ...
    'DU40_A17' 'DU40_A17'  ...
    'DU35_A17' 'DU35_A17'  ...
    'DU30_A17' ...
    'DU25_A17' ...
    'DU21_A17'    'DU21_A17' ...
    'NACA64_618'  'NACA64_618'   'NACA64_618' 'NACA64_618'  'NACA64_618' ...
    'NACA64_618'  'NACA64_618'   'NACA64_618' 'NACA64_618'  'NACA64_618' ...
    'NACA64_618'  };

Blade.Chord = Chord;
Blade.Xpitch = Xpitch;
Blade.TwistAng = TwistAng;
Blade.Z = Zs;

v = zeros(nSect,1);


Sh = 1:nSect;
Sh1X = v; Sh2X=v;

% Spar cap width over the majority of the blade
Sh1X(Sh) = zeros(size(Sh ))-.62*.615;
Sh2X(Sh) = zeros(size(Sh ))+.62*.615;
Shb = v; Shb(Sh)=1; % = 1 if shear webs present in section

% Make spar cap taper off at tip of the blade
Sh1X(end) = Sh1X(end) + .3;
Sh1X(end-1) = Sh1X(end-1) + .2;
Sh1X(end-2) = Sh1X(end-2) + .1;
Sh1X(end-3) = Sh1X(end-3) + .01;

Sh2X(end) = Sh2X(end) - .3;
Sh2X(end-1) = Sh2X(end-1) - .2;
Sh2X(end-2) = Sh2X(end-2) - .1;
Sh2X(end-3) = Sh2X(end-3) - .01;

% Trailing edge reinforcement x-position
TrRe = 1:nSect;
TrReX(TrRe) = Chord(TrRe).*(1-Xpitch(TrRe))-.3;

% Leading edge reinforcement x-position
LeReX(TrRe) = Chord(TrRe).*(-Xpitch(TrRe))+.3;

% Make trailing edge reinforcement taper off at tip of the blade
TrReX(end-4) = 0.85;
```

```
TrReX(end-3) = 0.6;
TrReX(end-2) = 0.45;
TrReX(end-1) = 0.3;
TrReX(end) = 0.15;
TrReX = TrReX';
LeReX = LeReX';

% Make leading edge reinforcement taper off at tip of the blade
LeReX(end-8) = -0.85;
LeReX(end-7) = -0.8;
LeReX(end-6) = -0.75;
LeReX(end-5) = -0.65;
LeReX(end-4) = -0.55;
LeReX(end-3) = -0.45;
LeReX(end-2) = -0.33;
LeReX(end-1) = -0.22;
LeReX(end) = -0.1;

%TrReX(TrReX<0)=-Chord(TrReX<0).*Xpitch(TrReX<0)+.001;
TrReb = v; TrReb(TrRe) = 1; % = 1 if reinforcement present in third section

nstr = 5;

Reb = [Shb Shb TrReb Shb ];
ReX = [Sh1X Sh2X TrReX LeReX];

% Thickness of foam in the aft panel
Aftpanel =[0.0000 0.0000 0.0000 0.0000 0.0000 ...
           0.0004 0.0013 0.0049 0.0113 0.0189 ...
           0.0227 0.0227 0.0227 0.0227 0.0227 ...
           0.0227 0.0227 0.0227 0.0227 0.0227 ...
           0.0227 0.0227 0.0227 0.0227 0.0208 ...
           0.0170 0.0113 0.0057 0.0038 0.0019 ...
           0.0019 0.0019 0.0019 0.0000];

%Thickness of foam in the fore panel
LEpanel = [0.0000 0.0000 0.0000 0.0000 0.0000 ...
           0.0004 0.0013 0.0049 0.0113 0.0189 ...
           0.0227 0.0227 0.0227 0.0227 0.0227 ...
           0.0227 0.0227 0.0227 0.0227 0.0227 ...
           0.0227 0.0227 0.0227 0.0227 0.0208 ...
           0.0170 0.0113 0.0057 0.0038 0.0019 ...
           0.0019 0.0019 0.0019 0.0000];

%Thickness of uni-directional material in the spar caps
thsp =    [0      0.0004 0.0008 0.0011 0.0015 ...
           0.0038 0.0049 0.0049 0.0076 0.0113 ...
           0.0193 0.0257 0.0356 0.0420 0.0450 ...
           0.0514 0.0514 0.0514 0.0484 0.0450 ...
           0.0420 0.0386 0.0321 0.0257 0.0242 ...
           0.0178 0.0129 0.0064 0.0034 0.0019 ...
           0.0019 0.0019 0.0019 0 ];

%Thickness of uni-directional material in the LE reinforcement
thTrRe1 = [0      0.0004 0.0030 0.0030 0.0030 ...
           0.0030 0.0030 0.0034 0.0049 0.0068 ...
```

```
           0.0095 0.0125 0.0151 0.0189 0.0227 ...
           0.0227 0.0227 0.0227 0.0113 0.0113 ...
           0.0057 0.0030 0.0015 0.0015 0.0015 ...
           0.0015 0.0015 0.0015 0.0015 0.0015 ...
           0.0015 0.0015 0.0015 0 ];

%Thickness of uni-directional material in the TE reinforcement
thTrRe2 = [0         0         0         0         0 ...
           0         0         0         0         0 ...
           0.0227    0.0227    0.0227    0.0227    0.0227 ...
           0.0227    0.0227    0.0227    0.0151    0.0151 ...
           0.0076    0.0038    0.0038    0.0038    0.0038 ...
           0.0038    0.0038    0.0038    0.0038    0.0038 ...
           0.0038    0.0038    0.0038    0];

% Root buildup
thRoot =  [0.0340 0.0340 0.0340 0.0340 0.0340 ...
           0.0340 0.0340 0.0340 0.0340 0.0102 ...
           0.0057 0.0019 0      0      0 ...
           0      0      0      0      0 ...
           0      0      0      0      0 ...
           0      0      0      0      0 ...
           0      0      0      0 ];

% Skin thickness + Root
thskin = 0.0019*ones(1,nSect);
thskin = thskin + .8*thRoot;

for i = 1:nSect

    % Number of strips at section
    Blade.Sect{i}.nstr = nstr;

    % Z coordinate of section
    Blade.Sect{i}.Z = Zs(i);

    % Chord of section
    Blade.Sect{i}.Chord = Chord(i);

    % Twist angle of section
    Blade.Sect{i}.TwistAng = TwistAng(i);

    % Pitch of section
    Blade.Sect{i}.Xpitch = Xpitch(i);

    % Profile type of section
    Blade.Sect{i}.ProfileType = ProfileType(i);

    % Material parameters for FEMAP
    Blade.Sect{i}.E1out = Blade.MatlInBlade(1,:);
    Blade.Sect{i}.E2out = Blade.MatlInBlade(2,:);
    Blade.Sect{i}.Gout =  Blade.MatlInBlade(3,:);
    Blade.Sect{i}.nyout = Blade.MatlInBlade(4,:);
    Blade.Sect{i}.rout =  Blade.MatlInBlade(5,:);
```

```
    % Number of plys in each laminated section
    Blade.Sect{i}.l = [5 4 4 4 4 4 4 4 4 5 3 3];

    % Definition of laminates
    % Format:
    % [ Material no. Thickness Angle ]

    % Trailing edge reinforcement 1
    Blade.Sect{i}.L{1} = [3 thskin(i) 0
                          1 .6*thTrRe1(i) 0;
                          4 .25*Aftpanel(i) 0;
                          3 .003*.615 0
                          5 0.005*0.615 0];

    % Skin 1
    Blade.Sect{i}.L{2} = [3 thskin(i) 0;
                          4 Aftpanel(i) 0;
                          3 .003*.63 0
                          5 0.005*0.615 0];

    % Spar 1
    Blade.Sect{i}.L{3} = [3 thskin(i) 0;
                          1 0.7*thsp(i) 0;
                          3 .003*.63 0
                          5 0.005*0.615 0];

%Skin 3
    Blade.Sect{i}.L{4} = [3 thskin(i) 0;
                          4 LEpanel(i) 0;
                          3 .003*.63 0
                          5 0.005*0.615 0];

    % LE Reinf
    Blade.Sect{i}.L{5} = [3 thskin(i) 0
                          1 1.4*thTrRe1(i) 0;
                          3 .003*.615 0
                          5 0.005*0.615 0];
    % LE Reinf
    Blade.Sect{i}.L{6} = [3 thskin(i) 0
                          1 1.4*thTrRe1(i) 0;
                          3 .003*.615 0
                          5 0.005*0.615 0];

    % Skin 4
    Blade.Sect{i}.L{7} = [3 thskin(i) 0;
                          4 LEpanel(i) 0;
                          3 .003*.63 0
                          5 0.005*0.615 0];

% Spar 2
    Blade.Sect{i}.L{8} = [3 thskin(i) 0;
                          1 1.3*thsp(i) 0;
                           3 .003*.63 0
                          5 0.005*0.615 0];

%Skin 5
```

```
    Blade.Sect{i}.L{9} = [3 thskin(i) 0;
                          4 Aftpanel(i) 0;
                          3 .003*.63 0
                          5 0.005*0.615 0];

% Trailing edge reinforcement 2
    Blade.Sect{i}.L{10} = [3 thskin(i) 0
                          1 .6*thTrRe1(i) 0;
                          4 .25*Aftpanel(i) 0;
                          3 .003*.63 0
                          5 0.005*0.615 0];

    % Shear web 1
    Blade.Sect{i}.L{11} = [2 0.003*.63 0 ;
                          4 0.08*.615 0 ;
                          2 0.003*.63 0 ];

    % Shear web 2
    Blade.Sect{i}.L{12} = [2 0.003*.63 0 ;
                          4 0.08*.615 0 ;
                          2 0.003*.63 0 ];

    for k = 1:2*nstr+2
        Blade.Sect{i}.th(k) = sum(Blade.Sect{i}.L{k}(:,2));

        [A B D] = layupcalc(Blade.Sect{i}.L{k}(:,1)',Blade.Sect{i}.L{k}(:,2)',Blade.Sect{i}
        Blade.Sect{i}.E(k) = (A(1,1)*A(2,2)-A(1,2)^2)/(A(2,2)*Blade.Sect{i}.th(k));
        Blade.Sect{i}.G(k) = A(3,3)/Blade.Sect{i}.th(k);
        Blade.Sect{i}.r(k) = sum(Blade.MatlInBlade(5,Blade.Sect{i}.L{k}(:,1)).*Blade.Sect{i
    end


    % Print x coordinates of reinforcements
    for  j= 1:nstr-1
        if Reb(i,j) == 1
            Blade.Sect{i}.xstr(1,nstr-j)  = ReX(i,j);
            Blade.Sect{i}.xstr(2,nstr-j)  = ReX(i,j);
        end
    end

    Blade.Sect{i}.connect = [3 4 ;
                             7 8];

end

% Some modifications
Blade.Sect{1}.l = [5 4 3 4 4 4 4 3 4 5 3 3];
Blade.Sect{1}.L{3} = [3 thskin(1)/3 0;
                      3 thskin(1)/3 0;
                      3 thskin(1)/3 0];

Blade.Sect{1}.L{8} = [3 thskin(1)/3 0;
                      3 thskin(1)/3 0;
                      3 thskin(1)/3 0];
```

```
Blade.Sect{1}.E(nstr-1)=Eskin;
Blade.Sect{1}.E(nstr+2)=Eskin;
Blade.Sect{1}.G(nstr-1)=Gskin;
Blade.Sect{1}.G(nstr+2)=Gskin;
Blade.Sect{1}.r(nstr-1)=rskin;
Blade.Sect{1}.r(nstr+2)=rskin;
Blade.Sect{1}.th(nstr-1)=thskin(1);
Blade.Sect{1}.th(nstr+2)=thskin(1);
```