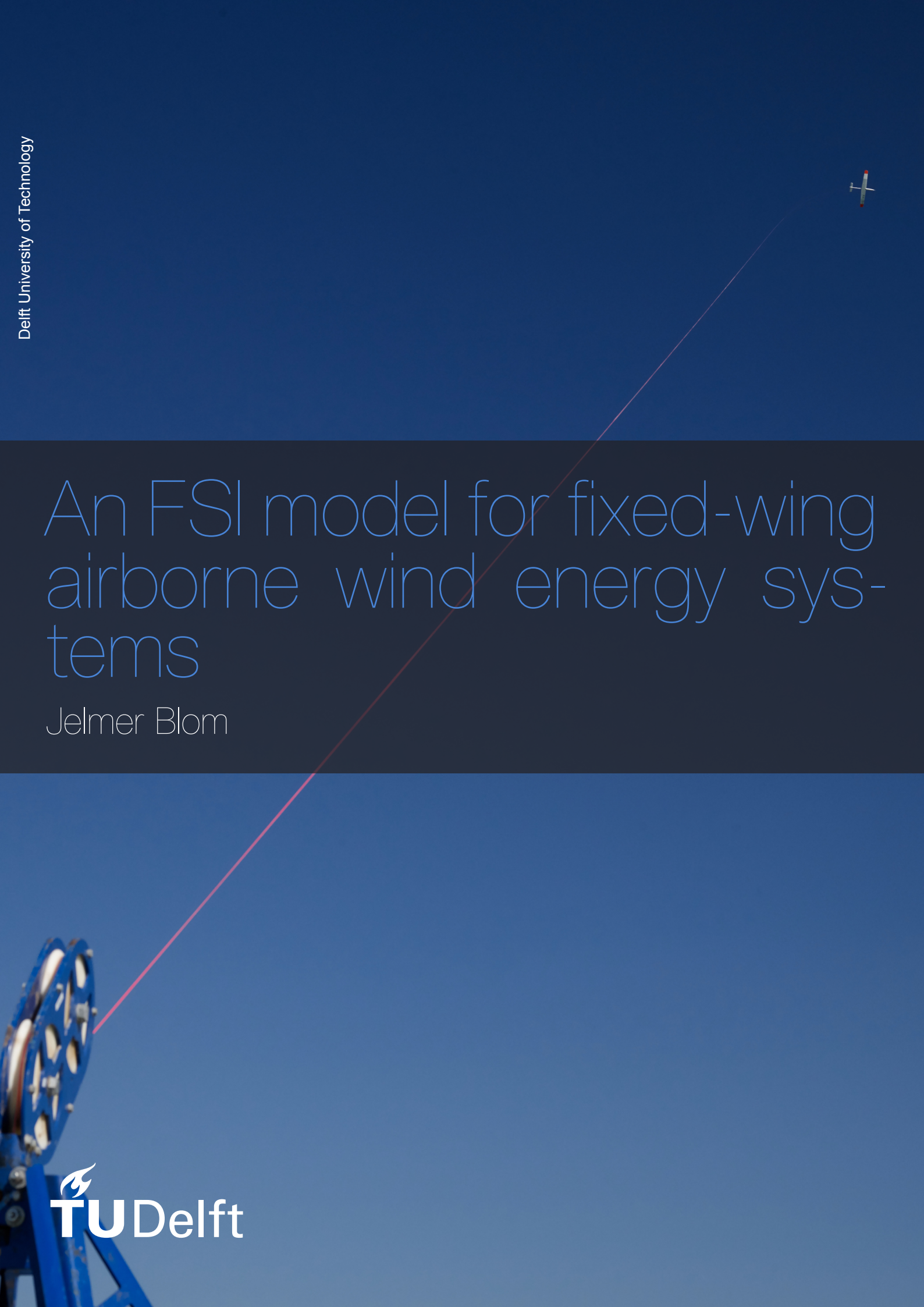
Delft University of Technology



An FSI model for fixed-wing airborne wind energy systems

Jelmer Blom

TU Delft

An FSI model for fixed-wing airborne wind energy systems

by

Jelmer Blom

To obtain the degree of Master of Science in Aerospace Engineering at the Delft University of Technology

Student number:	4369548	
Supervisor:	Dr.ir. R. Schmehl	TU Delft
Thesis Committee:	Dr.ir. R. Schmehl	TU Delft chair
	Prof.dr.ir. C.J. Simão Ferreira	TU Delft
	Dr.ir. A.H. Van Zuijlen	TU Delft
	ir. D. Eijkelhof	TU Delft
Institution:	Delft University of Technology	
Place:	Faculty of Aerospace Engineering, Delft	

Cover Image: Photo by Ampyx Power

TU Delft

Preface

Doing a thesis project during these challenging times was something I was not particularly looking forward to. Working at the university and talking to fellow researchers changed to working from home and online meetings. Although COVID has thrown up these extra hurdles along the way, in the end I'm left with a satisfied feeling and I feel confident that this research is able to contribute to the ongoing research in the field of Airborne Wind Energy.

I would not have been able to do this thesis project without the help of certain people. First of all, I would like to express my gratitude towards my supervisor Roland Schmehl, for introducing me to the topic of Airborne Wind Energy and guiding me throughout my project. Carlos Simao Ferreira for helping me at the start of my thesis project and pointing me in the right direction. Dylan Eijkelhof for the many talks, data and simply being someone to talk to about everything regarding my thesis work. Jelle Poland for the company and the many discussions about our subjects on the way to the beach.

Lastly, my friends and family for the support and much needed distractions every now and then, without which I would not have been able to do this.

Jelmer Blom
Rotterdam, March 2022

Summary

Conventional wind turbines have certain downsides that potentially can be solved by Airborne Wind Energy Systems (AWESs). As the need for renewable energy sources grow, research interest in AWESs have been growing over the past two decades. Eijkelhof et al. [19] have set up a simulation framework to evaluate the power production of a fixed-wing, ground-generation AWES, using a fixed-wing tethered aircraft and crosswind power generation. The MegAWES model, which is based on an upscaled version of the AP-3 model of Ampyx Power, is used in the framework and research. The current reference model is unable to account for viscous effects or multi-element airfoils, limiting the capabilities of the framework. The objective of this thesis project is to improve the prediction of the generated aerodynamic force of the aircraft by combining a structural and aerodynamic model into a fluid-structure interaction (FSI) model and eliminating the previously mentioned limitations.

The simulation framework is used as a preliminary design tool and therefore computational efficiency needs to be taken into account. The choice has been made to use the current results of the finite element (FE) method of Eijkelhof et al. [19] and couple it to an extended non-linear lifting line written in Matlab as current models are not able to fulfill all requirements of the aerodynamic model. To include viscosity and multi-element airfoils, the 2D program MSES is used. The structural model requires a force distribution on the upper and lower surface. Therefore, a method has been developed to extract a force distribution on both surfaces using the lifiting line result. The aerodynamic and structural models are coupled through a thin-plate spline method and an inverse distance weighting.

The new model is compared to the old reference model of Eijkelhof [18] and high fidelity results of Pynaert [49] for a circular flight path. The new model, when compared to the high fidelity results, show better accordance than the reference model. For the available data points, deviations from the high fidelity results dropped 16.5 to 25.2 % compared to the reference model. For all cases the deviation of the new model is within 10% of the high fidelity results.

For the new model, the FE grid is used to determine the geometric input data for the lifting line, eliminating geometric mismatches between the two models. The results of one flight state or iteration are used as input for the next state or iterations, lowering the time needed for the solution to converge. An FSI iteration takes roughly 2-2.5 seconds and for every flight state, generally 1 or 2 iterations are needed. Before the iterations are run, the 2D data needs to be generated and a pre-FSI run needs to be done.

Overall, the new model shows to be more accurate, relatively low in computational cost and free of the limitations that the reference model experienced. Therefore, the new model is a valuable addition to the research done on fixed-wing, large-scale AWESs and can be used in further research.

Contents

Nomenclature

Abbreviations

Abbreviation	Definition
2D	2 Dimensional
3D	3 Dimensional
AIC	Aerodynamic Influence Coefficient
AWE	Airborne Wind Energy
AWESs	Airbrone Wind Energy Systems
CFD	Computational Fluid Dynamics
DNS	Direct Numerical Simulation
FDM	Finite Difference Method
FE	Finite Element
FEM	Finite Element Method
FGS-GG	Fixed Ground Station Ground-Generation
FSI	Fluid-Structure Interaction
FVM	Finite Volume Method
LES	Large Eddy Simulation
LLT	Lifting Line Theory
RANS	Reynolds-Averaged Navier-Stokes
URANS	Unsteady Reynolds-Averaged Navier-Stokes
VLM	Vortex Lattice Method

Symbols

Symbol	Definition	Unit
b	Wing span	m
c	Chord	m
$\boldsymbol{c}_i$	Chordwise unit vector	
C_l	Lift coefficient	
C_p	Pressure coefficient	
D	Drag	N
e	Internal energy	J
$\boldsymbol{f}$	Body force	N
$\boldsymbol{F}$	Force vector	N
k	Thermal conductivity	W/mK
L	Lift	N
$\boldsymbol{M}$	Mass matrix	
$\boldsymbol{n}_i$	Normal unit vector	
p	Pressure	N/m^2
$\dot{\boldsymbol{q}}$	Heat transfer	J/sm^2
Q_∞	Freestream velocity	m/s
T	Temperature	K
t	Time	s
u	X-component of velocity	m/s
v	Y-component of velocity	m/s
V	Volume	m^3
$\boldsymbol{V}$	Velocity vector	m/s

Symbol	Definition	Unit
$\boldsymbol{V}_j$	Velocity at control point *j*	m/s
w	Z-component of velocity	m/s
w_b	Induced downwash bound vortices	m/s
w_i	Induced downwash wake vortices	m/s
x	Displacement vector	m
α	Angle of attack	^{o}
α_0	Zero lift angle of attack	^{o}
δ	Boundary layer thickness	m
δ_T	Thermal boundary layer thickness	m
η	Normalized spanwise position	
Γ	Circulation	m^2/s
γ	Specific heat ratio	
λ	Bulk viscosity coefficient	
Λ	Quarter chord sweep	^{o}
μ	Dynamic viscosity	Ns/m^2
$\boldsymbol{\phi}$	Velocity potential	m^2/s
ρ	Density	kg/m^3
τ	Stress	N/m^2

List of Figures

List of Tables

1 Introduction

The ability to accommodate the growing population on our planet, is related to the amount and available energy [64]. Up until recently, we obtained almost all of this energy from fossil fuels. However, a limited resource and their use is responsible for the greenhouse effect. To be able to sustain the growing population on Earth, there is a need for more sustainable energy sources. In 2015, 196 countries committed to the Paris Agreement, which obliges them to keep global warming well below 2 degrees Celsius and therefore cut back on greenhouse emission and support sustainable development [58]. New methods of large scale sustainable energy production are needed to be able to reach this.

In recent years a new branch has emerged in the sustainable energy sector: Airborne wind energy (AWE). Airborne wind energy systems (AWESs) aim at capturing the significantly increased wind velocities at higher altitudes. The winds at these altitudes are generally stronger and more consistent. In AWESs, a flying vehicle substitutes the rotor blades of a wind turbine and a tether the supporting structure, thus reducing the amount of material needed per unit of usable power compared to conventional wind turbines. Multiple concepts of AWES are being worked on nowadays, such as on-board power generation, ground-based power generation and flexible and fixed-wings [2].

AWESs can be divided into different configurations. The first subdivision can be made whether the systems make use of ground-based or on-board generation. Ground-generation AWESs use the aerodynamic force of the kite or aircraft. This force generates a torque by uncoiling the tether and this torque is then converted by the generator into electrical power. Ground-based systems can have either a fixed ground station, where the generator is stationary, or a moving ground station, where the generator moves [2]. On-board AWESs generate electrical energy in the air and transmit this from the airborne system to the ground. This is done through a tether that is able to carry conducting electrical cables. The on-board systems can produce the electrical energy either using crosswinds or non-crosswind, depending on the system. Figure 1.1 shows the different companies working on the various types of AWESs.

The most common flight mode used in AWESs is the crosswind flight mode. Loyd [39] showed that the power output when flying crosswind is significantly higher than non-crosswind flights. Another interesting choice for different systems is whether to use flexible or fixed-wings. Flexible wings are lighter, which is beneficial for the power production and in the case of a crash they cause less damage. The durability of flexible kites is limited to several hundreds of hours for current designs and materials [16]. Fixed-wings have a higher lift to drag ratio and can reach much higher velocities, resulting in higher power outputs [2]. It the case of a crash it will however cause more damage and total loss of the system. Maximizing the power output of these systems is one of the challenges being faced nowadays. In order to achieve higher power outputs, the aerodynamic performance needs to be optimized and the structure needs to be lightweight. Compared to conventional aircrafts, AWESs have a much higher wing loading in order to overcome the extra tensile force exerted on the system by the tether [63].

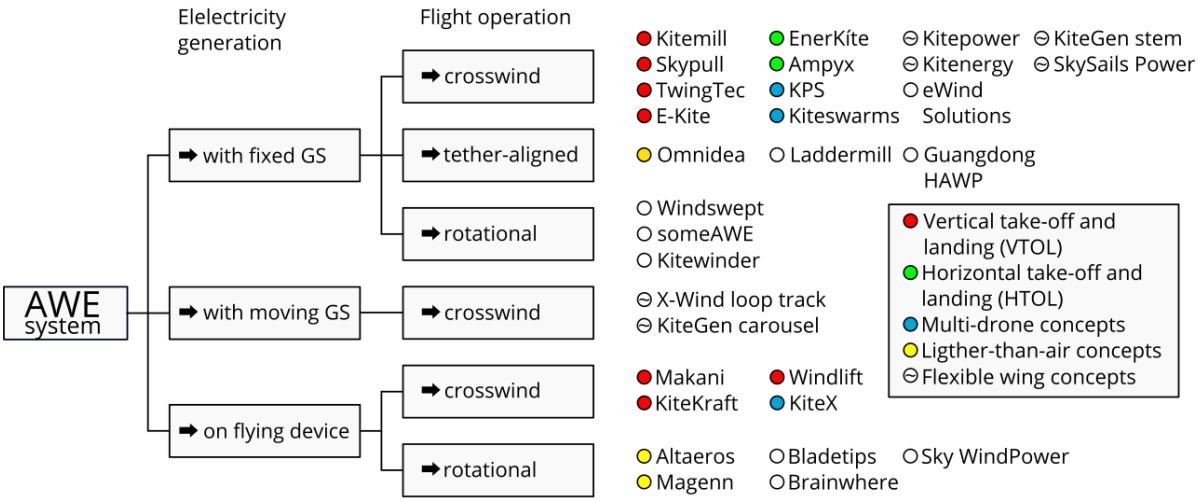
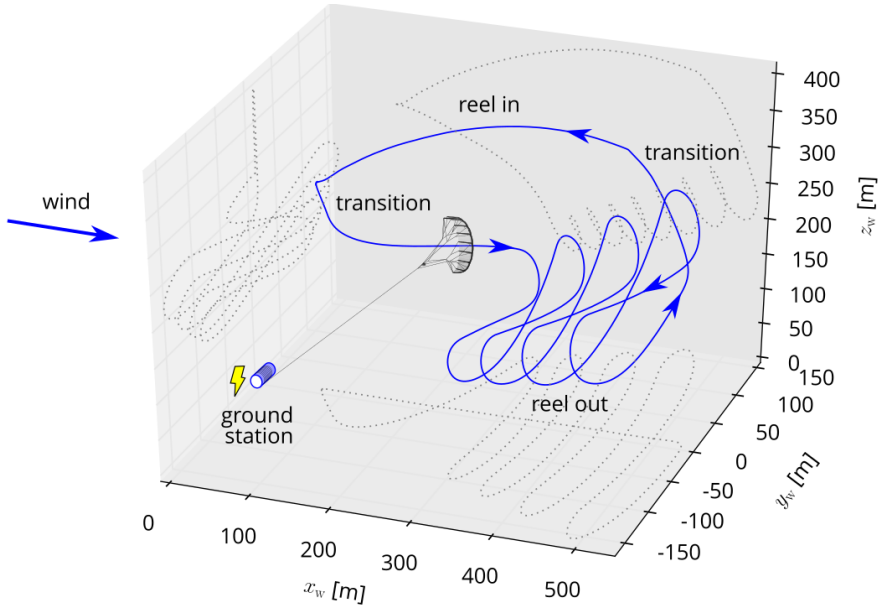


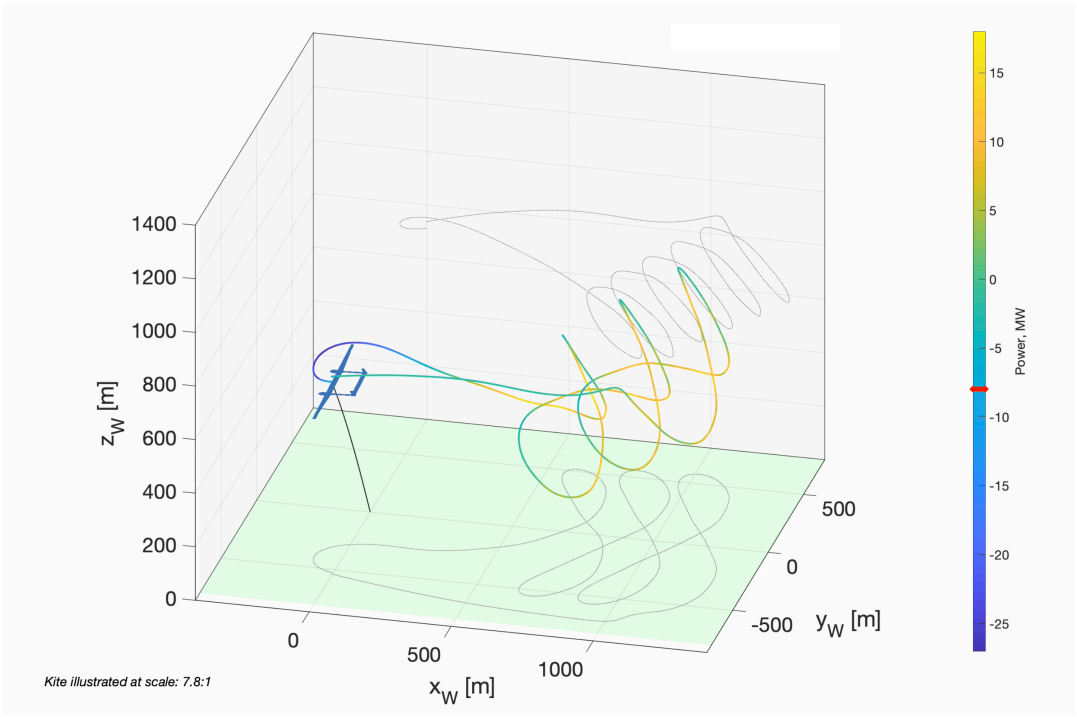
Figure 1.1: Classification of AWES concepts and companies on the various types of AWESs [55].

At an academic level as well as throughout the industry, research is being done on developing feasible AWESs. The Wind Energy section of the Aerospace faculty at the Delft University of Technology is one of the academic places where research is being conducted in the field of AWE. One of the systems that is being researched is the fixed-wing tethered aircraft. A simulation framework for a multi-MW AWES has been set up and described by Eijkelhof [18] and Eijkelhof et al. [19]. When evaluating small-scale fixed-wing systems, aeroelasticity is often not considered. However, when large scale systems are considered, such as the system evaluated by the framework of Eijkelhof, structural deformations due to wing loading become profound and need to be taken into account [63]. In the past, aeroelastic phenomena have been the cause for systems to crash, showing the need to include aeroelasticity in the design process [7, 38].

In this research, the focus will be on the fixed ground station, ground-generation airborne wind energy systems (FGS-GG-AWESs), using fixed-wing tethered aircrafts and crosswind power generation. The aircraft being analysed is the MegAWES model, which is based on the upscaled AP-4 model of Ampyx power and can be found in Appendix B. The model flies in pumping cycle operations, in which the tether is reeled out while flying figure of eights and afterwards reeled in. Figure 1.2a visualizes the different phases of a pumping cycle and Figure 1.2b shows the pumping cycle flown by the model in the current simulation framework.



(a) A pumping cycle and its different phases, visualized for a flexible kite [55].



(b) Pumping cycle for the MegAWES model.

Figure 1.2: Pumping cycle visualization [46].

The goal of the current research project is to improve the aerodynamic model used to predict the aerodynamic behaviour in the simulation framework of Eijkelhof et al. [19]. The current model pre-computes lift and drag coefficients of the fixed-wing aircraft using a 3D panel method (Apame), which has been developed by Filkovic [22]. These coefficients are stored in a look-up table and for specific flight states, the corresponding aerodynamic coefficients are found by interpolation. The current aerodynamic model uses an inviscid method and thus for higher angles of attack, the prediction of the aerodynamic coefficients will deviate from actual values. The range of angles of attack the aircraft will operate in, is therefore limited to the linear part of the lift curve to ensure reasonable computed aerodynamic coefficients. An aerodynamic model that accounts for the viscous effects can therefore be beneficial for the simulation framework. This research will mainly focus on developing a fluid-structure

interaction (FSI) model that predicts the aerodynamic behaviour accurately for large-scale airborne wind energy aircraft (high aspect ratios, multi-element airfoils, high lift). This model can then be used for mission optimisation and/or control optimisation. A full FSI model is desirable so that it can be used for different applications as well, such as airfoil optimization or more flexible aircraft. The model should contribute to the existing body of science and be useful for the AWE community for further research. This is formulated into the following research question:

> "How does an FSI model for the preliminary design of fixed-wing multi-megawatt airborne wind energy systems look like and can this FSI model be developed and implemented in the available simulation framework?"

The main research objective of this thesis is:

> "Provide an accurate and computationally efficient FSI model in the order of seconds for the simulation framework of multi-megawatt airborne wind energy systems that can be used for further development within this framework."

To be able to answer this research question, this thesis is split up into multiple parts. Chapter 2 discusses the literature review needed to answer the research question. Chapter 3 discusses the numerical implementation of the aerodynamic model and the resulting FSI model. Chapter 4 shows the validation cases used to validate the written aerodynamic model. Chapter 5 discusses the various choices and limitations of the new model. Chapter 6 shows the results of the final aerodynamic and FSI model. Chapter 7 draws a conclusion of the work done and discusses various recommendations regarding future work.

2

Literature review

This chapter discusses the theoretical background and existing solvers. The focus of this research project is on the aerodynamic side of the FSI model and therefore this will be more profoundly discussed. The structural modelling and coupling method will be discussed for completeness, but not elaborated upon.

2.1. Aerodynamic modelling

There are many different ways to model the flow around a body. These methods differ in fidelity and computational cost. The methods with the highest accuracy solve the complete Navier-Stokes equations. With an increasing number of assumptions these equations become easier to solve. When viscosity is omitted, one arrives at the Euler equations. When rotation in the flow is omitted, one arrives at the full potential equations. Lastly, when compressibility is not taken into account, the Laplace equations remain. With each of these extra assumptions, the accuracy and the computational cost decrease. This is visualized in Figure 2.1.

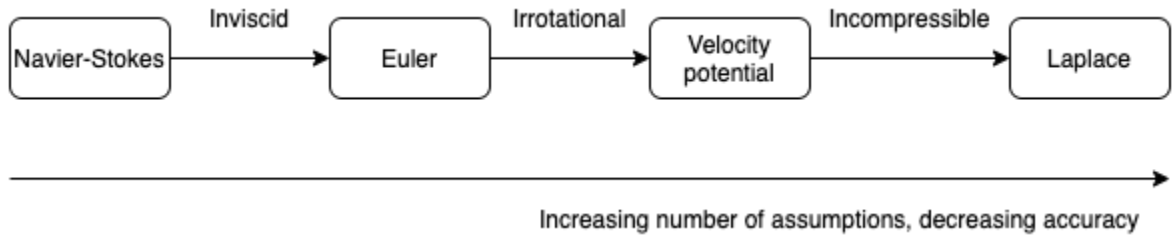


Figure 2.1: Overview of aerodynamic flow models based on Jameson [28]

2.1.1. Navier-Stokes equations

The Navier-Stokes equations are the most accurate mathematical description of a viscous, compressible, rotational and unsteady flow. The full set of equations consists of 5 equations: a continuity equation, three conservation of momentum equations and an energy equation. As these equations are extensive and will not be used further in the research, they are not given here. If interested, the equations can be found in the book of Anderson [4].

The Navier-Stokes equations do not have any general solutions. Therefore, obtaining practical results from these equations can be done through simplifications or other methods:

- First of all, there are a few viscous flow problems for which certain terms in the Navier-Stokes equations are equal to zero: Couette and Poisseuille flow. These simplifications reduce the equations to a set that can be solved.

- The equations can be simplified by neglecting certain terms. This however results in a solution that is not precise anymore, but an approximation of the actual solution, e.g. the boundary layer equations.
- The solution can be obtained by numerical techniques, such as Direct Numerical Simulation (DNS), Large Eddy Simulation (LES), the Unsteady Reynolds-Avaraged Navier-Stokes equations (URANS) and the steady Reynolds-Avaraged Navier-Stokes equations (RANS). These numerical techniques are needed to model turbulent flows [6].

Direct Numerical Simulation (DNS)
DNS is a direct simulation of the full time-dependent Navier-Stokes equations and the most accurate method there is at the moment. It is not exact due to discretization and possible numerical errors. This is at the moment only possible for low Reynolds number flows, due to the computational expense of this technique (the CPU time scales with Re^3). DNS is used for understanding turbulent structures, laminar-turbulent transition and the development of new turbulence models.

Large Eddy Simulation (LES)
Whereas DNS solves all turbulence in the flow, LES makes the assumption that small eddies have a more universal character and that large eddies transport the turbulent energy. Due to this assumption, LES only resolves the large eddies in the flow and approximates the effects of the smaller eddies. LES needs less grid points than DNS and therefore the computational time is lower compared to this method.

Reynolds-Averaged Navier-Stokes equations (RANS)
RANS decomposes the flow variables into an average and a fluctuation. These decomposed flow variables are plugged into the Navier-Stokes equations and therefore the equations are extended with two extra terms: the Reynolds-stress tensor and the turbulent heat flux vector. RANS thus requires the modelling of these two extra terms, however the overall grid can be much coarser compared to DNS or LES and hence reducing the computational time.

Unsteady Reynolds-Avaraged Navier-Stokes equations (URANS)
URANS is able, like LES, to capture transient flow phenomena. However, URANS is able to do this at a lower computational cost than LES [54]. Where RANS averages over the entire time domain, URANS averages over a single time step.

2.1.2. Boundary layer equations

The boundary layer equations are based on the Navier-Stokes equations, assuming that the viscous effects are confined within a thin boundary layer on the surface. Outside of this viscous boundary layer, the flow is assumed to be inviscid. The properties of the boundary layer are shown in Figure 2.2.

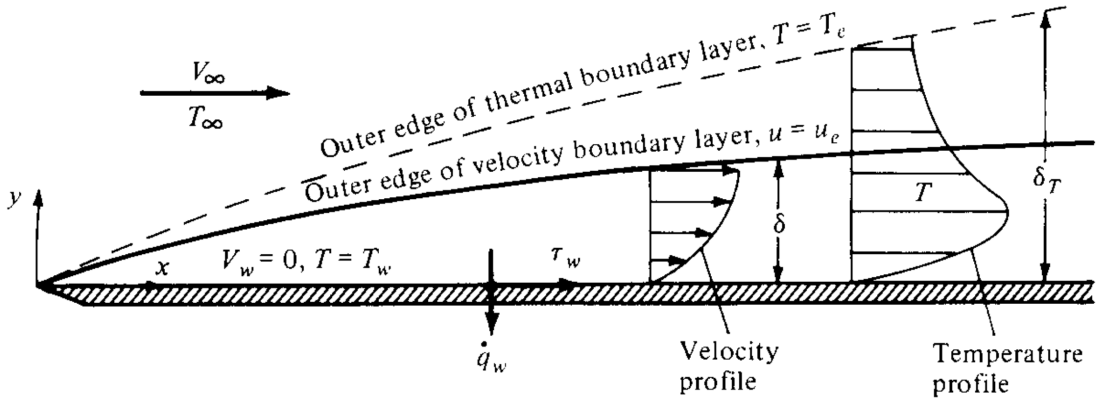


Figure 2.2: Boundary layer properties from Anderson [4].

The full derivation of the equations can be found in Chapter 17 of Anderson [4], so here only the

resulting equations will be shown. The boundary layer equations are deduced from the full Navier-Stokes equations using two assumptions, namely $\delta \ll c$ and Re $\geq \frac{1}{\delta^2}$. The continuity, x-momentum, y-momentum and energy equations respectively become:

$$\frac{\partial(\rho u)}{\partial x} + \frac{\partial(\rho v)}{\partial y} = 0 \tag{2.1a}$$

$$\rho u \frac{\partial u}{\partial x} + \rho v \frac{\partial u}{\partial y} = -\frac{dp_e}{dx} + \frac{\partial}{\partial y}\left(\mu \frac{\partial u}{\partial y}\right) \tag{2.1b}$$

$$\frac{\partial p}{\partial y} = 0 \tag{2.1c}$$

$$\rho u \frac{\partial h}{\partial x} + \rho v \frac{\partial h}{\partial y} = \frac{\partial}{\partial y}\left(k \frac{\partial T}{\partial y}\right) + u \frac{dp_e}{dx} \mu \left(\frac{\partial u}{\partial y}\right)^2 \tag{2.1d}$$

2.1.3. Euler equations

As displayed in Figure 2.1, the Euler equations assume inviscid flow, which impacts the flow equations. The continuity equation does not depend on viscosity and therefore when viscosity is neglected, this equation remains unchanged:

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \boldsymbol{V}) = 0 \tag{2.2}$$

The momentum equations however do change. The momentum and energy equations change to Equation 2.3 [5].

$$\frac{\partial(\rho u)}{\partial t} + \nabla \cdot (\rho u \boldsymbol{V}) = -\frac{\partial p}{\partial x} + \rho f_x \tag{2.3a}$$

$$\frac{\partial(\rho v)}{\partial t} + \nabla \cdot (\rho v \boldsymbol{V}) = -\frac{\partial p}{\partial y} + \rho f_y \tag{2.3b}$$

$$\frac{\partial(\rho w)}{\partial t} + \nabla \cdot (\rho w \boldsymbol{V}) = -\frac{\partial p}{\partial z} + \rho f_z \tag{2.3c}$$

$$\frac{\partial}{\partial t}\left[\rho\left(e + \frac{\boldsymbol{V}^2}{2}\right)\right] + \nabla \cdot \left[\rho\left(e + \frac{\boldsymbol{V}^2}{2}\right)\boldsymbol{V}\right] = \quad \rho \dot{\boldsymbol{q}} - \nabla \cdot p\boldsymbol{V} + \rho \boldsymbol{f} \cdot \boldsymbol{V} \tag{2.3d}$$

Also notice that the thermal conductivity terms disappear due to the assumption of inviscid flow.

2.1.4. Velocity potential equations

The velocity potential equation combines the continuity, momentum and energy equations into a single equation with a single variable: the velocity potential. The velocity potential equation does make the extra assumpotions that the flow is irrotational and isentropic [5]. The equation is not given as it will not be used furthermore for this research project.

2.1.5. Laplace equation

The last assumption to be added to the previous assumptions is that of an incompressible flow. For an incompressible flow, the mass conservation becomes [4]:

$$\nabla \cdot \boldsymbol{V} = 0 \tag{2.4}$$

Given the fact that for an irrotational flow, the velocity potential ϕ can be defined such that $\boldsymbol{V} = \nabla \phi$, Equation 2.4 can be rewritten into the Laplace equation:

$$\nabla^2 \boldsymbol{\phi} = 0 \tag{2.5}$$

Another way of deriving this equation is to take the velocity potential equation and realizing that for incompressible flow $a \to \infty$. The big advantage of the Laplace equation is the fact that it is linear, therefore elementary flow solutions can be superimposed.

2.1.6. Solution methods

The different equations discussed previously do not directly describe parameters such as pressure, velocity or aerodynamic coefficients. To obtain something useful from these equations, solution methods are used. To solve (one of) the equations discussed above, multiple options can be considered. The two main categories of solution methods are the non-linear solution methods and the linear solution methods. In the coming subsections, the most common solution methods will be discussed.

Non-linear solution methods

Non-linear solution methods are used, as the name suggests, to solve non-linear equations such as the Navier-Stokes, Euler and full potential equations. These methods discretize the fluid domain into elements and apply mass, momentum and energy conservation laws on each element, e.g. FDM, FEM and FVM [6, 27].

Finite difference method (FDM)

FDM applies a Taylor expansion to the derivatives of the governing equations at each grid point. This method requires the mesh to be structured and this also immediately gives one of the downsides of this method. The truncation error that is introduced in the Taylor expansion depends on the difference between the grid points Δx. Therefore, the smaller the distance between the grid points, the more accurate this method will be.

Finite volume method (FVM)

The FVM is the most used method in CFD nowadays. This method divides the domain into smaller cells, called control volumes. FVM computes cell averaged values, whereas with FDM values are calculated at discrete points. FVM applies the integral conservation laws over each control volume. FVM are not restricted to structured grids as with the case of FDM, but can be generally applied to structured and unstructured grids.

Finite element method (FEM)

FEM has similarities to FVM in its approach, but there are differences. First of all, the domain is divided into arbitrary shapes, thus an unstructured grid. These control cells are called the elements, although they are in essence similar to the control volumes of FVM. Inside an element, the so-called shape functions are defined. These shape functions represent the variation of the solution inside the element. Lastly, it solves the integral conservation laws of each element.

Other less used discretisation methods such as the spectral element method and the gridless method are not discussed here, but the reader is referred [6] for more information.

Linear solution methods

The linear solution methods solve the Laplace equation, $\nabla^2 \phi = 0$. Therefore these solutions assume inviscid, irrotational, incrompressible, steady flow without body forces. Although the list of assumptions is becoming quite extensive, when these solution methods are applied properly, the results will correlate well with experimental data. As these methods assume inviscid flow, linear solution methods are often coupled to some secondary solution method, e.g. XFOIL. In that way, viscous effects can be taken into account. The Laplace equations are linear and therefore elementary solutions can be superimposed. These elementary flow solutions can be split up into uniform flow, source flow, doublet flow and vortex flow. Linear solution methods aim to find a singularity distribution to satisfy the set boundary conditions.

Lifting line method

In Prandtl's classicial lifting line method, the flow field of a wing of finite span *b* is replaced by a so-called bound vortex. All vorticity present in the flow field is collapsed into this vortex and all of the lift is produced at this line. The location of this vortex follows from thin airfoil theory. The center of pressure is the point at which the pitching moment coefficient of the airfoil is zero. Lift and drag can be represented as forces acting only on this point. However, the location of the center of pressure depends on the lift coefficient and thus on the angle of attack. Besides the center of pressure, the aerodynamic center is the point at which the moment coefficient does not change with angle of attack. The aerodynamic center for inviscid, incompressible flows at subsonic Mach numbers is located at a quarter chord behind

the leading edge. As the center of pressure depends on the angle of attack, this point can lie outside of the airfoil in certain cases, whereas the aerodynamic center location remains unchanged with an angle of attack variation. Therefore, the aerodynamic center is chosen as the location to place the bound vortex and evaluate the force [32].

Helmholtz's theorem states that a vortex filament can not end in space and therefore this bound vortex extends by two trailing vortices at each end downstream. These trailing vortices are joined by another vortex downstream, which is assumed to not influence the solution. Multiple horseshoe vortices are superimposed along the span of the wing to arrive at a Γ-distribution along the span. This is shown in Figure 2.3.

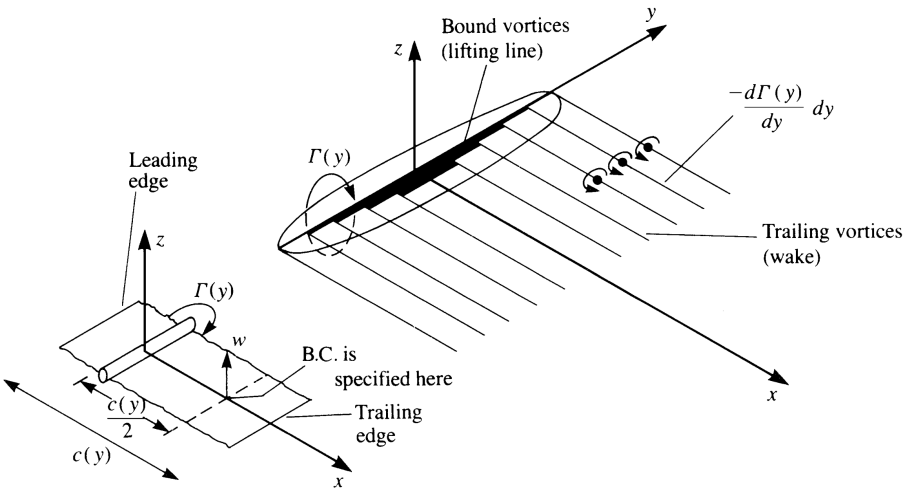


Figure 2.3: Lifting line model of horseshoe vortices from Katz et al. [31].

When evaluating the lifting line, one can either use the non-entry boundary condition or the angle of attack method.

Non-entry boundary condition method
When using the non-entry boundary condition, the control points are located at the three-quarter chord, as shown in Figure 2.3. As the blade is assumed to be a solid surface, the flow normal to the surface needs to be zero. As the surface of the blade is compressed into a single line at the quarter chord, it follows from Pistolesi's theorem that the control points are $\frac{1}{2}c$ behind the bound vortex, thus at three-quarter chord. The vortices induce a downward velocity on the wing called downwash. At the three quarter chord point a boundary condition is set such that the sum of the velocities normal to the wing are zero:

$$\boldsymbol{w}_b + \boldsymbol{w}_i + \boldsymbol{Q}_\infty \alpha = 0 \tag{2.6}$$

In Equation 2.6, w_b is the velocity induced by the wing vortices, w_i the induced velocity by the wake vortices and $Q_\infty \alpha$ the component of the free-stream flow. However, in the derivation that the control points are at the three-quarter chord, the assumption is made that the lift coefficient is equal to $2\pi \sin(\alpha)$, which is only valid in the linear part of the lift curve. To be able to directly use viscous 2D lift polars, another method could be used: the angle of attack method.

Angle of attack method
The angle of attack method places the control points at the quarter chord and thus on the bound vortex. The control points are located at the quarter chord so that there is no induced velocity by the bound vortex. This method calculates the perceived angle of attack at each of these control points. The starting point of this method is the geometry of the wing, and therefore the control and vortex points, and an initial circulation distribution that can be chosen as one wishes. Using the Biot-Savart law, aerodynamic influence coefficient matrices can be determined and used to calculate the induced velocities at each

control point. From these induced velocities, the perceived angle of attack can be found and used to determine the lift coefficient from viscous lift polar data. The lift coefficients are corrected for the sweep and therefore the occuring 3D flow phenomena. By employing the Kutta-Joukowsky theorem, a new circulation distribution can be determined. Convergence criteria are checked if the circulation distribution needs to be iterated upon. If convergence is reached, the angles of attack are outputted and used to find the spanwise lift distribution. Figure 2.4 shows a flowchart of this aerodynamic iteration loop.

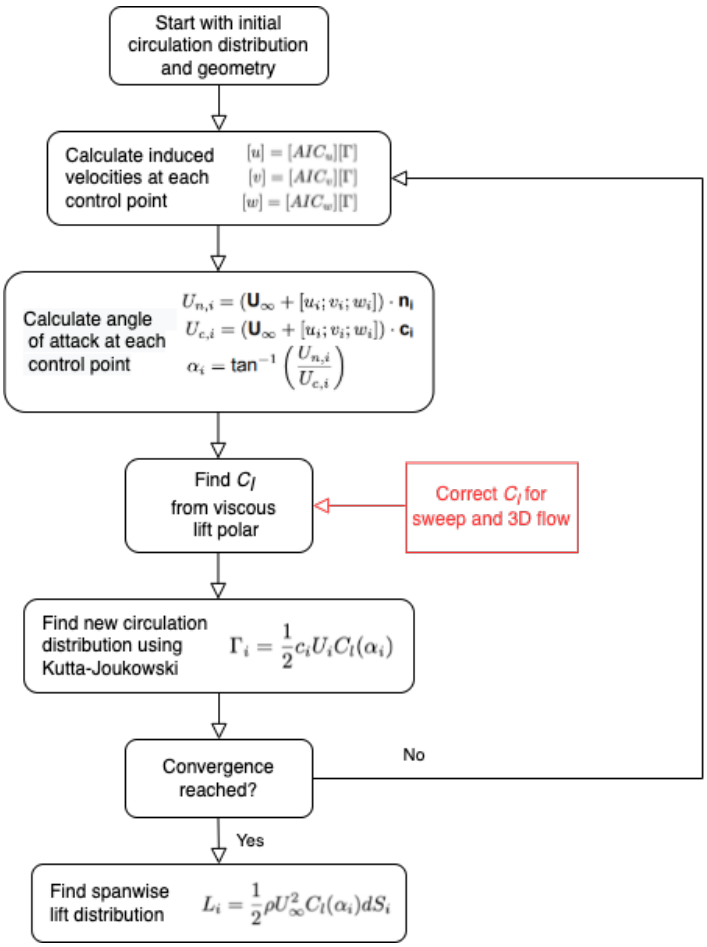


Figure 2.4: Flowchart of the aerodynamic iteration loop.

Phillips et al. [47] gives an adaptation of the lifting line model described by Katz et al. [31]. The induced velocity at any arbitrary point in space is given by Equation 2.7 according to the definitions of Figure 2.5.

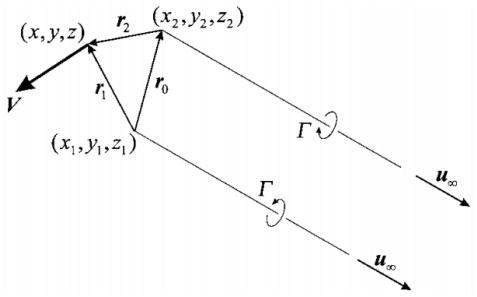


Figure 2.5: Vectors describing the horseshoe vortex from Phillips et al. [47]

$$\boldsymbol{V} = \frac{\Gamma}{4\pi}\left[\frac{\boldsymbol{u}_\infty \times \boldsymbol{r}_2}{\boldsymbol{r}_2(\boldsymbol{r}_2 - \boldsymbol{u}_\infty \cdot \boldsymbol{r}_2)} + \frac{(\boldsymbol{r}_1 + \boldsymbol{r}_2)(\boldsymbol{r}_1 \times \boldsymbol{r}_2)}{\boldsymbol{r}_1\boldsymbol{r}_2(\boldsymbol{r}_1\boldsymbol{r}_2 + \boldsymbol{r}1 \cdot \boldsymbol{r}_2)} - \frac{\boldsymbol{u}_\infty \times \boldsymbol{r}_1}{\boldsymbol{r}_1(\boldsymbol{r}_1 - \boldsymbol{u}_\infty \cdot \boldsymbol{r}_1)}\right] \tag{2.7}$$

The velocity at an arbitrary control point positioned on the bound vortex (quarter chord line) is given by Equation 2.8.

$$\boldsymbol{V}_j = \boldsymbol{V}_\infty + \sum_{i=1}^{N} \frac{\Gamma \boldsymbol{v}_{ij}}{\bar{c}_i} \tag{2.8}$$

Here $\boldsymbol{v}_{ij}$ is the induced velocity at each control point i by vortex j. When $i = j$ the downwash of the bound vortex segment is zero, because the control point lies on top of this vortex. Therefore $\boldsymbol{v}_{ij}$ can be written as follows:

$$\boldsymbol{v}_{ij} = \begin{cases} \frac{\bar{c}_i}{4\pi}\left[\frac{\boldsymbol{u}_\infty \times \boldsymbol{r}_{i_2 j}}{\boldsymbol{r}_{i_2 j}(\boldsymbol{r}_{i_2 j} - \boldsymbol{u}_\infty \cdot \boldsymbol{r}_{i_2 j})} + \frac{(\boldsymbol{r}_{i_1 j} + \boldsymbol{r}_{i_2 j})(\boldsymbol{r}_{i_1 j} \times \boldsymbol{r}_{i_2 j})}{\boldsymbol{r}_{i_1 j}\boldsymbol{r}_{i_2 j}(\boldsymbol{r}_{i_1 j}\boldsymbol{r}_{i_2 j} + \boldsymbol{r}_{i_1 j} \cdot \boldsymbol{r}_{i_2 j})} - \frac{\boldsymbol{u}_\infty \times \boldsymbol{r}_{i_1 j}}{\boldsymbol{r}_{i_1 j}(\boldsymbol{r}_{i_1 j} - \boldsymbol{u}_\infty \cdot \boldsymbol{r}_{i_1 j})}\right] & , i \neq j \\ \frac{\bar{c}_i}{4\pi}\left[\frac{\boldsymbol{u}_\infty \times \boldsymbol{r}_{i_2 j}}{\boldsymbol{r}_{i_2 j}(\boldsymbol{r}_{i_2 j} - \boldsymbol{u}_\infty \cdot \boldsymbol{r}_{i_2 j})} - \frac{\boldsymbol{u}_\infty \times \boldsymbol{r}_{i_1 j}}{\boldsymbol{r}_{i_1 j}(\boldsymbol{r}_{i_1 j} - \boldsymbol{u}_\infty \cdot \boldsymbol{r}_{i_1 j})}\right] & , i = j \end{cases} \tag{2.9}$$

Note that these expressions by Phillips et al. is a rewritten version of the expressions given by Katz et al. Phillips et al. rewrite the expression for the induced velocity that is given by Katz et al. using trigonometric relations. The expression given by Katz et al. returns zero induced velocity if the spatial vectors r_1 and r_2 are collinear, even if the point in space is not on the vortex segment. In the case of a single wing the control points lie on the quarter chord and if the bound vortices on this same wing are evaluated, it returns zero induced velocity for all bound vortex segments due to these spatial vectors being collinear. The method by Phillips and Snyder rewrite the expression for the induced velocity such that this singularity doesn't occur anymore for the case of 0 degrees between r_1 and r_2, but it only returns zero if the angle between r_1 and r_2 is $\pm\pi$ (control point lies on top of vortex segment). Although this method enables one to account for most of the bound vortex, the numerical implementation of this method is more tedious than the method of Katz et al. and therefore the method by Katz et al., described in Section 3.2.5, is used in this project.

If the perceived angle of attack at a spanwise location is known, the lift coefficient can be found using 2D data. If the lift coefficient of a wing section is known, determining the lift on the section is trivial. This method allows one to directly use viscous data, whereas the non-entry boundary condition method is limited to inviscid flow if no corrections are applied. Furthermore, the angle of attack method allows one to use 2D airfoil data of multi-element airfoils, which is required by AWESs.

Vortex lattice method

Where the lifting line method discretizes the wing spanwise, but only has one element in the chordwise direct, the vortex lattice method discretizes the wing spanwise, as well as chordwise. The wing is split up into trapezoid shaped panels. At any control point on the wing, the sum of the normal component of the freestream velocity and induced velocity needs to be equal to zero. Essentially, the non-entry boundary condition method is extended chordwise to arrive at the VLM. Both VLM and lifting lines neglect the thickness of the wings.

Panel methods

Panel methods cover a three-dimensional object with panels, just as the VLM does for two-dimensional objects. Each panel has an unknown distribution of singularities, which can be vortices, but also point-sources or doublets. Using flow tangency and calculating the induced velocities, a set of algebraic equations can be set up. The basis of 3D panel methods is that the Laplace equation (Equation 2.5) is solved to obtain the velocity field. Using the Euler-Bernoulli equation, the pressures can then be obtained. Integrating these pressures over the surfaces provides the total forces [22].

Contrary to lifting lines and VLM, panel methods do account for the thickness of bodies, see Figure 2.6. It is able to cover the entire object and not just the lifting surfaces, therefore taking effects into account that lifting lines and VLM do not.

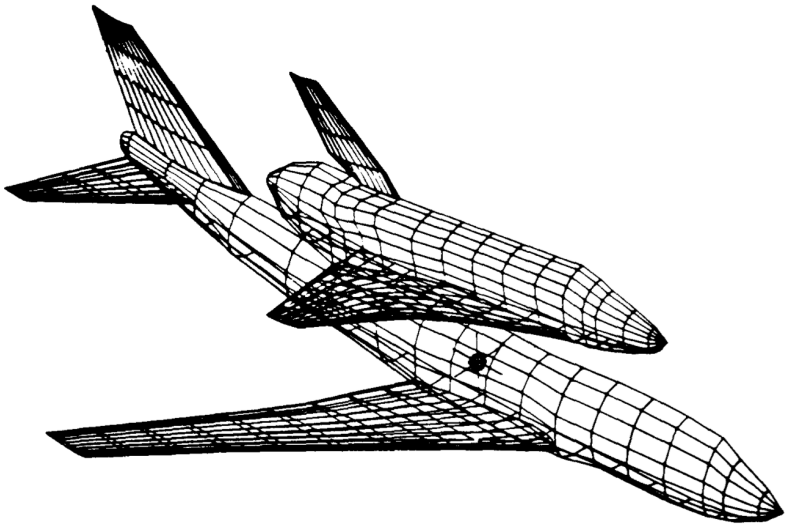


Figure 2.6: Panel distribution a Boeing 747 that carries the space shuttle orbiter [4].

2.2. Structural modelling

The structural model used for this thesis work is a 3D finite element method. The method used by Eijkelhof [18] is not changed for this work. The stiffness matrix outputted by MSC Nastran is directly used for the calculation of the displacements of the structural mesh. Eijkelhof creates the mesh and the different structural components in Matlab, after which the mass, stiffness and inertia matrices using the linear static analysis solver of MSC Nastran. The finite element model is built in nine steps by Eijkelhof and shortly elaborated on here for completeness:

1. The parameters of the aircraft, Appendix B are used to generate the planform. This entails the position of the ribs, sweep, taper and twist distribution.
2. The different skin elements are assigned an identification number in order to assign properties to the different elements. Next to that the composite lay-ups are assigned to the skin elements using these numbers.
3. The airfoils are interpolated from the tip to the root. In this case the airfoil at the tip is similar to the airfoil at the root and therefore the airfoil is only scaled using the chord length.
4. Nodes at each rib are determined using a node spacing (previously determined in a convergence study) and interpolated to the correct airfoil shape from step 3.
5. Spar sections and skin panels are added. This is done by performing the following steps. Interpolation between the ribs is performed to obtain a quadrilateral mesh. Afterwards, Delaunay triangulation is performed on each intersection of ribs and stringers, providing indices in between ribs that will form a triangular element in MSC Nastran. This results in the surface mesh of Figure 2.7.

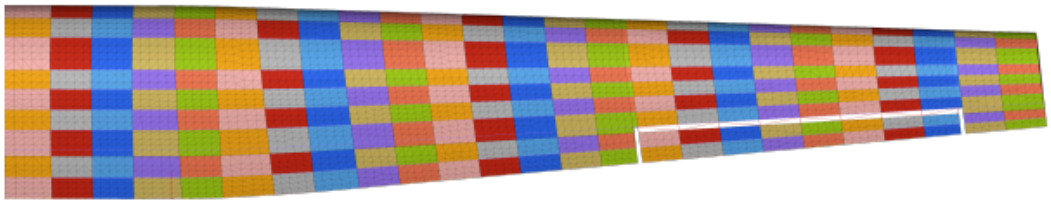


Figure 2.7: Top view of the surface mesh on the wing [18]

6. The stringers of the wing are included in the model using beam elements. Each beam element is a straight rectangular element with a certain width and composite lay-up.

7. The aileron mechanism is included in the model. Stiff rod elements are used to constrain the ailerons rotation around the axis. Where other researchers used a combination of multiple elements, this approach was taken so that the equations of motion, set up by Fasel et al. [20], do not need any modification.
8. Fuselages and tail are added to the model. This is done to account for the added mass and inertia using beam elements. The fuselages are represented using beam elements with a circular cross-sections, whereas the tail is represented using beam elements with a hollow bar cross-section.
9. The last addition is the rib element at each rib station. Using Delaunay triangulation, each rib is generated using triangular elements.

From this finite element model, a stiffness matrix can be obtained through Nastran. To find the displacements of the nodes on the structural mesh, this stiffness matrix can be utilised.

2.3. Coupling methods

Aerodynamic and structural meshes have different requirements and generally speaking, these meshes do not match. These non-matching meshes need to be coupled by some coupling method in order to utilise the forces on the structural mesh and the displacements on the aerodynamic mesh. In FSI computations, the loads computed by the flow solver need to be transferred to the structural nodes and the displacements the other way around.

Some sort of interpolation or projection needs to be applied to transfer the data between the two meshes. The general opinion is that the energy should be conserved over the interface of the fluid and structure. Conservation of virtual work forms the basis of this assumption. In this conservative approach, one transformation matrix is used to transfer data between the two meshes. As mentioned by Ahrem et al. [1], this can lead to unphysical oscillations in the forces received by the structural mesh.

Another option is to not use the same transformation matrix both ways. Two different transformation matrices are used, one for transferring forces to the structure and the other for the displacements to the fluid. This is called the consistent approach and avoids the unphysical oscillations. Contrary to the conservative approach, conservation of energy is not guarenteed anymore. However, when using a partitioned approach, this does not have to influence the solution [8].

Two coupling methods are used for the interpolation between the two meshes. First of all the thin plate spline method is used to interpolate the structural displacements onto the aerodynamic mesh. In terms of accuracy and efficiency, the thin plate spline method performs better than any other method [8]. The other way around however, another method is used: the inverse distance weighting [56]. When the thin plate spline method would be used to transfer the aerodynamic forces onto the structural mesh, this would lead to large oscillations [18].

The thin plate spline method and the inverse distance weighting used in this research are unchanged from the method set out by Eijkelhof [18]. Instead of changing the coupling method to fit the new aerodynamic model in the FSI, the aerodynamic model output is adapted such that the same coupling strategy can be used. As explained by Eijkelhof, the thin plate spline method gives an approximation for the displacements:

$$u_m(\vec{x}) = \sum_{i=1}^{n_{s,s}} \gamma_i \left\|\vec{x} - \vec{x}_{s,i}\right\|^2 \log_{10}\left(\left\|\vec{x} - \vec{x}_{si}\right\|^2\right) + p(x), \quad m = \{a, s\} \tag{2.10}$$

Where γ_i is an interpolation coefficient and $p(x)$ a linear polynomial. This equation can be rewritten into Equation 2.11 and Equation 2.12. The displacements on the aerodynamic mesh can be found using Equation 2.12.

$$\boldsymbol{T}_T = \begin{bmatrix}\boldsymbol{M}_{a,s} & \boldsymbol{P}_a\end{bmatrix} \begin{bmatrix}\boldsymbol{M}_{s,s} & \boldsymbol{P}_s \\ \boldsymbol{P}_s^T & 0\end{bmatrix}^{-1} \tag{2.11}$$

Where $\boldsymbol{M}_{a,s}$ is $\left\|\vec{x} - \vec{x}_{s,i}\right\|^2 \log_{10}\left(\left\|\vec{x} - \vec{x}_{si}\right\|^2\right)$ and $\boldsymbol{M}_{s,s}$ is the same but on the structural mesh.

$$\vec{u}_a = \boldsymbol{T}_T \vec{u}_s \tag{2.12}$$

The inverse distance weighting, shown in Equation 2.13, is used to interpolate the forces from the aerodynamic mesh to the structural mesh. The force at a node on the structural mesh will be proportional to the inverse of the distance from that point to each point on the aerodynamic mesh.

$$F_{s,i} = \frac{\|\vec{x}_a - \vec{x}_{s,i}\|}{\sum_{k=1}^{n_{s,s}} \|\vec{x}_a - \vec{x}_{s,k}\|} \cdot \boldsymbol{F}_a \tag{2.13}$$

2.4. Modelling of AWESs

Aeroelastic modelling of fixed-wing AWESs has developed over time and can be done using different methods. Multiple tools have been developed, utilising various aerodynamic models, structural models and coupling methods. Some of the more widely known solvers are ASWING [14], KiteFAST [35], QBlade [41] and SHARPy [10]. Each solver uses different methods and programming languages, each suited for their own use cases. The first decisions for a computational FSI solver involves a partitioned or a monolithic solver and what models to use regarding the computational cost.

2.4.1. Partitioned versus monolithic

Computational aeroelasticity can be split up in two major categories: monolithic and partitioned solvers, see Figure 2.8. Monolithic solvers fully couple the governing equations for the fluid and the structure. The coupled equations are solved in one go at every time step. A disadvantage of such a monolithic solver is that it is specific to a single problem.
On the other hand, partitioned methods have two different solvers: one for the aerodynamic part and one for the structural part. These two solvers are coupled such that the loads are projected onto the structural mesh and the displacements on the aerodynamic mesh. This coupling is an extra source of errors, however if applied correctly, a partitioned solver can be as accurate as a monolithic solver. The coupled system of a partitioned solver is iterated until convergence [26, 30].

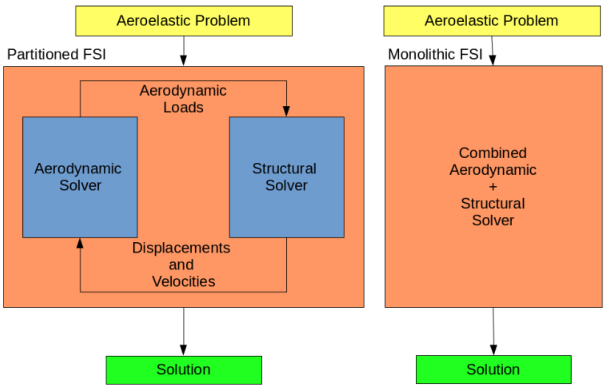


Figure 2.8: Difference between monolithic and partitioned solvers [36]

One major advantage of the partitioned approach is that for the fluid and structure domain two separate solvers can be used that are suited for the particular problem. Since the two solvers for each domain are solved separately, the solvers can be replaced by different methods if required. Felippa et al. [21] give an extensive overview of general concepts of the partitioned approach for coupled systems.

2.4.2. Computational cost

For the final FSI model to be useful for the simulation framework, its computational cost needs to be relatively low. It will be used in early design stages/optimization and therefore a trade-off should be made between fidelity and computational cost of the model. An overview of the computational cost of different aerodynamic and structural modelling methods is given in Figure 2.9.

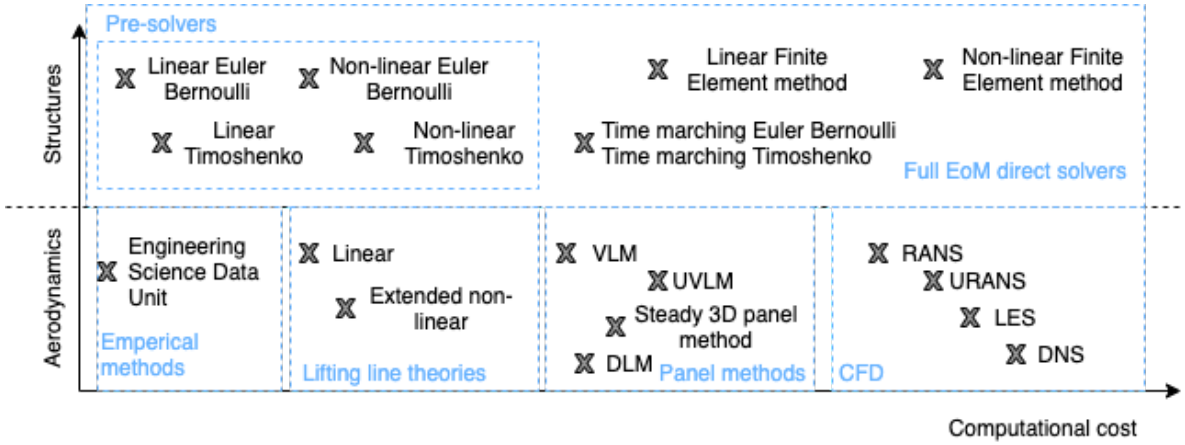


Figure 2.9: Computational cost of different modelling methods, adapted from [17].

2.5. Existing solvers

AWESs are being modelled by researchers and companies in many ways and forms. Currently there are solvers that are able to analyse the aerodynamics as well as the structures and there are solvers that only evaluate one of the two.

2.5.1. FSI solvers

A short overview of full FSI solvers for the aerodynamic criteria is given by Table 2.1.

FSI solvers	Suited for large scale AWESs	Computationally in the order of seconds	Viscosity & multi-element	Wake
OpenAeroStruct [29]	√	√	X	X
ASWING [14]	√	√	√	X
PROTEUS [61]	√	√	X	X
MACH [33][34]	√	X		
Dettman [12]	√	√	X	X
Candade et al. [9]	√	√	√	X
KiteFAST [35]	√	√	√	√
QBlade [41]	√	√	√	√
SHARPy [10]	√	√	√	√

Table 2.1: Full FSI solvers for the different aerodynamic criteria

OpenAeroStruct

Jasa et al. [29] provide an explanation about the reasons for the development and details regarding the tool OpenAeroStruct. The motivation for OpenAeroStruct was to provide a low-fidelity open source aeroelastic analysis tool. Other low fidelity aerodynamic and structural tools had their limitations according to Jasa et al., e.g. limited to 2D, no derivatives, poor interface with optimizers.
OpenAeroStruct is an open source (written in Python) aeroelastic tool that combines a vortex lattice method with a spatial beam model. The coupling of the aerodynamic and structural solvers is done in a simplified way. OpenAeroStruct makes sure that the panels of the VLM are aligned with the structural nodes. The forces on each panel is evenly distributed over the adjacent nodes. The displacement the other way around is done in a similar way: the average of the two adjacent nodes is taken for each panel. However, as mentioned before, generally speaking the aerodynamic and structural meshes have different coarseness. Due to the coupling used, either the aerodynamic mesh is too coarse or the structural mesh is finer than needed.

ASWING
ASWING is a tool developed by Mark Drela and combines a lifting line with a non-linear Euler-Bernoulli beam model. The tool is written in Fortran and can be downloaded with an academic license or a commercial license can be bought. The primary intent of ASWING was to evaluate the V-n diagram (velocity versus load factor) to identify potential failure scenarios.
Viscosity can be added in ASWING through sectional lift data if available. Wijnja [62] used ASWING to evaluate the Makani M600 concept.

PROTEUS
PROTEUS is an aeroelastic tool developed at the Delft University of Technology and described by Werter and De Breuker [61]. Initially it was used as a framework for the initial conceptual design of composite wings. Throughout the years, PROTEUS developed and more functionalities have been added to this framework, e.g. the expansion of the aerodynamic model with fuselages and tail described by Natella et al. [44]. This also makes it quite difficult to find a good overview of the capabilities and user experience of PROTEUS.
The initial PROTEUS tool combines a VLM with a non linear beam model. The VLM assumes inviscid flow and the beam model uses a cross sectional modeller with a non linear Timoshenko beam model.

MACH
A higher fidelity framework is that of the MDO Lab of the University of Michigan: MACH [34]. It is a tool written in Python with parts of incorporated FORTRAN code. MACH uses a finite-volume CFD solver for the aerodynamics and the Toolbox for the Analysis of Composite Structures (TACS) for the structures. TACS uses thin-shell elements in a finite element method to accurately capture the structural behaviour of composite wings [33].

Dettman
Dettman combines a vortex lattice method in combination with an Euler-Bernoulli beam model. The full model is meant for conceptual design and is a low fidelity and computationally fast model. No viscosity is included in the VLM, so the model does deviate for higher angles of attack. The VLM is loosely based on Tornado, with the addition of the ability to model control surfaces. The main programming language of Dettman is Phyton and parts of the VLM are written in C++. As the model of Dettman was a MSc Thesis at the KTH Royal Institute of Technology, the model is not readily available to be used.

Candade et al.
Candade et al. [9] combine a VLM with 2D CFD data. Candade et al. utilise a 2D cross sectional modeller to capture the behaviour of composite wings and couple this to a 1D Timoshenko beam model. Combining these two methods and utilising the 2D cross sectional behaviour, ensures an accurate, yet computationally fast model. The subject of this model is a tethered swept wing without fuselages or control surfaces. Bridles are attached directly onto the wing. The entire computational model is used in the research of Candade et al. [9] to explore the influence of the bridle on the wing shape.

KiteFAST
KiteFAST consists of multiple packages and this packages can be run separately or in combination. An FSI model can be obtained by combining the packages KiteAeroDyn for the kite aerodynamics and MBDyn for the structural dynamics. The KiteAeroDyn module calculates the aerodynamic forces by using a vortex step method. The 2D forces and moments are computed as concentrated loads acting on the aerodynamic center at each 2D cross section. The 3D forces are obtained by integrating the 2D forces over the span of the wing. KiteAeroDyn can be coupled to the multi-body dynamics tool MBDyn to obtain an aero-elastic tool. MBDyn is an open source multi-body dynamics analysis tool developed by the Politecnico di Milano university.

QBlade
QBlade is a tool developed for the design of wind turbines. However, within QBlade, modules are present which also serve other applications. Within QBlade, there is a lifting line free wake vortex implemented. XFOIL is implemented and therefore 2D airfoil data can be used in the lifting line method. If desired, 2D airfoil data can also be imported. This aerodynamic module is linked with Chrono, which

is a multi-physics analysis tool that can be used for structural analysis [57]. Together this forms an FSI module that can evaluate different shapes.

SHARPy
Simulation of High Aspect Ratio planes in Python (SHARPy) is a tool developed by Imperial College London. SHARPy is a toolbox for dynamic aeroelastic simulations of aircrafts and wind turbines. SHARPy is an open-source, modular toolbox with extensive capabilities, e.g. linear and non linear analysis, multibody dynamics and model order reduction techniques. The core code of SHARPy is written in Python, however computationally expensive routines are in libraries coded in C++ and Fortran. The unsteady VLM solver is able to model the wake with different fidelities.

2.5.2. Aerodynamic solvers

Besides the full FSI solvers, 3D aerodynamic solvers can be tested against the requirements set above. The results hereof are displayed in Table 2.2.

3D aerodynamic solvers	Suited for large scale AWESs	Computationally in the order of seconds	Viscosity & multi-element	Wake
AVL [15]	√	√	√	√
Tornado [43]	√	√	X	√
VSAERO [42]	√	√	√	√
SU^2 [45]	√	X		
APAME [22]	√	X		
XFLR5 [65]	√	√	X	√
ARC3D [48]	√	X		

Table 2.2: The 3D aerodynamic solvers for the different aerodynamic criteria

AVL
AVL is a vortex lattice method developed by Drela and Youngren. Due to its inviscid nature, the VLM assumes small angles of attack and sideslip, which is the most significant limitation of AVL. To model the fuselage, a slender-body method is used and the authors warn the user to proceed with caution due to the limited experience [15].
AVL does support multiple surfaces at a wing section, such as ailerons, flaps or a combination of both.

Tornado
Similar to AVL, Tornado uses a VLM to evaluate the Laplace equation. However, Tornado is compiled in MATLAB. The absence of viscous effects limits Tornado to the linear part of the lift curve. Tornado does support any number of control surfaces or ailerons, however the fuselages are not taken into account.

VSAERO
VSAERO (or Viscous Separation Aerodynamics Program) combines a linear potential flow and a boundary layer model. It is capable of evaluating any aircraft shape, as long as the paneling has been done properly. The original program of VSAERO is well described by Maskew [42]. However since then, the program has had revisions and it is not well known if these documents are still applicable to newer versions of the program. VSAERO is part of the software package offered by AMI Aero [3] and therefore not open source. The software package AMI Aero also includes other aerodynamic solvers, such as VLAERO+ (a vortex lattice method), USAERO (an unsteady aerodynamic solver) and MGAERO (a Cartisian Euler method).

SU^2
The Stanfort University Unstructured, better known as SU^2, is capable of solving arbitrary partial differential equations and is therefore not only applicable to solving aerodynamic problems. The SU^2 framework is set up to ensure an open source package, portability to any software with a C++ compiler

and reusability. The main capabilities of the SU^2 framework are high fidelity validation studies, optimizations and grid adaptation [45].

APAME
APAME is a 3D panel method set up and described by Filkovic [22]. As APAME is an inviscid method, it is limited to the inviscid region. The program produces very good results for the lift, but drag calculations are off when compared to higher fidelity CFD methods.

XFLR5
XFLR5 is the first hybrid solver that is discussed. Hybrid solvers combine 2D analysis to a 3D method. XFLR5 builds upon the XFOIL code and adds 3D analyses such as LLT, VLM and a 3D panel method. Therefore the limitation of not being able to analyse multi-element airfoils is also applicable to XFLR5.

ARC2D/ARC3D
ARC2D and ARC3D are two programs that follow the same algorithm [48]. Inviscid as well as viscous flow can be analyzed in 2D and 3D. The solvers use either the Navier-Stokes or Euler equations and solve these using a finite difference method. Therefore the computational cost will be relatively high.

Furthermore, there are higher fidelity solvers available. Vimalakanthan et al. describe the aerodynamic analysis of Ampyx' AWES [60]. The authors use a Reynolds Averaged Navier-Stokes (RANS) Comupational Fluid Dynamics (CFD) model for the 2D aerodynamic forces. These forces are fed to a lifting line model to determine the overall forces on the aircraft. The aerodynamic coefficients are implemented as lookup tables by sweeping a large combination of control surface deflections and angle of attack.
Malz et al. [40] also make use of CFD data to determine the aerodynamic coefficients. Next to this, extensive test flights have been done to validate this data. During these test flights, the aerodynamic coefficients were determined for each control surface if possible. The coefficients are then represented by polynomial functions of the angle of attack [37].
Rapp et al. [52] use CFD in combination with XFLR5 to obtain the main dependencies of the aerodynamic coefficients on states and the different control surface deflections.
All of these methods use CFD data or full CFD simulations and are therefore not suitable for this research.

Next to XFLR5 and ARC2D there are more programs that are capable of analysing 2D airfoils. A short overview of the programs looked into for this project are given here.

XFOIL/RFOIL
The first version of XFOIL was written by Mark Drela in 1986. The goal of this program was to use high-order panel methods in combination with viscous/inviscid interaction. Since the first version of XFOIL, the code has had several revisions. However, the main goal of XFOIL is still the same: the analysis and design of subsonic airfoils. The analysis of multi-element airfoils is not possible with the current versions of XFOIL.
RFOIL is an adaptation of the XFOIL code by the Institue of Wind Energy in Delft. RFOIL includes a method to predict the effect of rotation on airfoil characteristics. Details regarding the RFOIL code are described by Van Rooij [53]. For higher angles of attack, the RFOIL code produces more accurate results than XFOIL, when compared to experimental data [50].

VGK
VGK uses a FDM to solve the full potential equations and viscosity is only evaluated within the boundary layer. VGK is limited to single-element airfoils and the boundary layer needs to be attached for VGK to produce results that represent real flow.
The results of VGK correlates well to experimental data when the boundary layer is attached and present shock waves are weak [23]. If local separation of the boundary layer occurs or it is close to separation, VGK is able to still give a good indication.

JavaFoil

JavaFoil is an open source, relatively simple program that calculates the velocity distribution over the surface of an airfoil using a panel method. The pressure distribution is found with the use of the Bernoulli equation. Furthermore, JavaFoil is able to evaluate multi-element airfoils.
JavaFoil does have its limitations. It is not able to model laminar separation bubbles or flow separation. When stall occurs, separation is modelled to some extent using empirical corrections for 'conventional' airfoils. For airfoils beyond stall, the solution becomes inaccurate.

Viiflow
Viiflow couples a 2D panel method with the integral boundary layer equations to arrive at a viscous-inviscid method. With a given displacement thickness, the boundary layer equations are solved and the viscous-inviscid coupling is down using a Newton method. It is written in Python and is able to evaluate multi-element airfoils. Viiflow shows a significant decrease in computational time compared to XFOIL [51].

MSES
MSES is a tool developed for the analysis of multi-element airfoils. It uses an Euler flow model that is solved using a finite volume method. Similar to Viiflow, MSES includes a viscous boundary layer. This will increase the computational time compared to the previously discussed tools. For a two element airfoil using a typical grid (220×40), MSES will require about 10 seconds per iteration. For inviscid cases, about 2 iterations are required and for viscous cases it can take around 15 to 20 iterations [13].

2.5.3. Reviewed solvers
In the previous section, multiple existing 2D and 3D solvers have been discussed. The main focus has been on lower fidelity solvers as the FSI model should be able to evaluate different flight states in the order of seconds. The discussed solvers aid towards answering the subquestions of Section 2.6, which will be done in Section 3.1.

2.6. Research questions
Besides the main research question discussed in Chapter 1, other research questions need to be addressed. To be able to answer the main research question, it is broken down into a set of more detailed subquestions:

1. "How would an aerodynamic model look like that is able to evaluate viscosity, multi-element airfoils in the order of seconds and is suited for the simulation framework of large-scale, fixed-wing AWESs?"
2. "How would a structural model look like that is capable of evaluating large-scale, fixed-wing AWESs in the order of seconds?"
3. "How would the new aerodynamic and structural model be coupled?"

Furthermore, a question can be asked whether or not high fidelity modelling at this stage is needed and when this should be done:

> "Is high-fidelity aerodynamic modelling needed for large-scale fixed-wing AWESs and at what stage would this be done?"

3

Implementation of the FSI model

In this chapter, the implementation of the FSI model is discussed. First of all, the choice for the different models is discussed. Followed by the implementation details of the aerodynamic model, structural model and the FSI model.

3.1. Method selection

From Table 2.1 and Table 2.2, 5 options remain for the aerodynamic solver: KiteFAST, QBlade, SHARPy, AVL (although AVL has limited wake modelling capabilities) and VSAERO. Next to these, it is also possible to program an aerodynamic solver.
VSAERO is part of the software package AMI Aero, which you need to contact in order to get an invoice. Therefore for this thesis project and the use of the simulation framework, VSAERO is not a viable option.

The remaining options are thus: KiteFAST, SHARPy, AVL, QBlade and to program a solver. All of these options need a way to include viscosity and multi-element airfoils. This can be done with the 2D airfoil software of MSES or with Viiflow. Both programs are able to capture the viscous effects of multi-element airfoils. MSES is the more established program of the two and from within the Aerospace section there is access to the MSES code. For those reasons, MSES will be used for the 2D analysis.

KiteAeroDyn uses the aerodynamic force calculating module in KiteFAST. The development of KiteFAST was abruptly stopped and thoughts regarding the program have been published in the report of Damiani [11]. The aerodynamics module in KiteFAST is tuned to the Makani M600 concept. Therefore, KiteAeroDyn assumes one fuselage, two wings, a vertical stabilizer, horizontal stabilizers and pylons. As indicated in the input files of KiteAeroDyn, it is not possible to input less than 1 pylon per wing. To be able to run other kite geometries, such as the MegAWES model that is used in the simulation, the code needs adjustments.

QBlade is a program used for the analysis of wind turbines. The option to analyse AWESs is build into the program, however very little is known about this. The downside of QBlade, SHARPy and AVL is that all of these programs are not written in Matlab and therefore can not be directly implemented into the simulation framework. Next to this, there are some computational downsides to these programs, e.g. QBlade is only available for Windows operating systems.

Besides the discussed downsides of the solvers, a force distribution on the upper and lower surface of the wing is needed. The current finite element model used in the simulation framework is preferable as its capabilities are used for the optimization of the structure of the wing. Force distributions on both surfaces are needed as input for this model and there are no models currently out there that are low-fidelity, are able to split up the force between both surfaces and adhere to the rest of the aerodynamic criteria. For those reasons, the decision is made to program a lifting line in Matlab and combine it with MSES to include the viscosity and multi-element airfoils. The pressure distributions produced by MSES are used to obtain a chordwise force distribution on the upper and lower surface of the wing.

The combination of these methods is able to adhere to all set aerodynamic criteria and can be easily modified in the future by other researchers.

3.2. Aerodynamic model

Section 3.2.1 discusses the generation of the 2D data with MSES, Section 3.2.2 the generation of the geometry of the aircraft, Section 3.2.3 the sweep correction factors for geometries with higher sweep, Section 3.2.4 the generation of the wake points, Section 3.2.5 the generation of the Aerodynamic Influence Coefficient matrix (AIC), Section 3.2.6 the calculation of the circulation distribution, Section 3.2.7 the force calculation and lastly Section 3.2.8 the force distribution chordwise.

3.2.1. Generation of 2D data

Before the lifting line code is able to run, 2D viscous airfoil data needs to be generated. This done in three stages:

- Parameter sweep over positive angles of attack for the lift polars
- Parameter sweep over negative angles of attack for the lift polars
- Generation of the pressure coefficient distribution files

The first two stages of this process are separated to aid in the convergence of MSES. If the parameter sweep would immediately be performed from the most negative angle of attack upwards, the first point will most likely not converge. To be able to evaluate these angles of attack, one parameter sweep is done from $\alpha = 0$ upwards and one sweep from $\alpha = 0$ downwards.

Furthermore, 7 inputs are required for the script to run. First of all, the airfoil coordinate file needs to be present in the main folder. For the specifics on the format of this file, the reader is referred to [13]. The other inputs for the script are the Mach numbers, Reynolds numbers, array of positive angles of attack, array of negative angles of attack, the number of elements of the airfoil and the airfoil extension. The latter is needed for MSES to run and this extension will be used by MSES for all executable files.

Angle of attack sweep

The angle of attack sweeps are done for the Reynolds numbers that are inserted. First a spec file is created for the positive angles. Again, for explanations about the format of the files and information of the executable files, the reader is referred to [13]. MSET is run for the airfoil and the MDAT file is created with the airfoil extension. The MSES file is edited for the specified Reynolds number and corresponding Mach number, after which the code runs the MPOLAR function. MPOLAR creates the file with the lift coefficient data and this file is automatically moved to the designated folder. These actions are repeated for all inserted Reynolds numbers.
When this process is finished, it is repeated for the array of negative angles of attack.

Pressure coefficient distributions

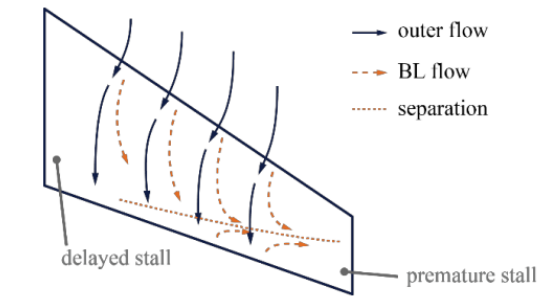
To generate the pressure coefficient distributions, the script loops over the Reynolds numbers and the angles of attack. The function runs MSET for the specified angle of attack, edits the MSES file for the correct Reynolds and Mach numbers and executes MSES with a maximum of 100 iterations. After running the MSES function, the pressure coefficient distribution over the surface is extracted with the MEDP function. The distributions are moved to the designated folder after extraction.

3.2.2. Geometry generation

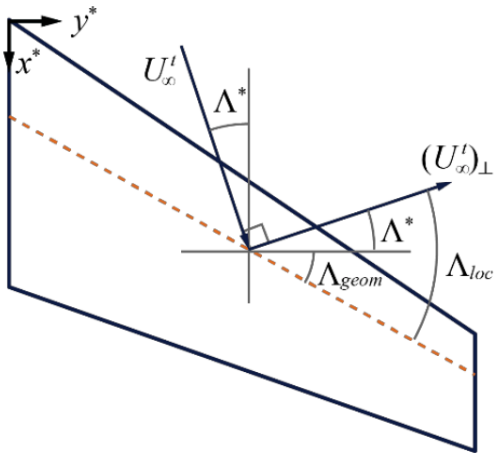
The geometry of the aircraft is generated in two stages. First of all, the finite element grid is utilised and all of the info that can not be extracted from this grid is generated afterwards.
The geometry of the front part of the aircraft is extracted from the finite element grid. In that way, the inputs for the lifting line and the structural model are similar. The control points, vortex points, local chord, sweep, twist, dihedral and span are extracted from the grid.
Secondly the rest of the geometry is generated using function files and calculated from the parameters. If the geometry is adapted, this should be done in geometry function files.

3.2.3. Sweep correction

Previously, the 2D polars have been generated for the different wings and airfoil sections. However, 3D flow phenomena will effect these polars in certain circumstances. The difference between 2D polars and, as Goitia et al. [25] state, 2.5D polars originate from the sweep of the wing. If the wing is swept, the sweep generates a spanwise boundary flow component, as can be seen in Figure 3.1a.



(a) Backwards swept wing and the different flows over the wing from Goitia et al. [25]



(b) Generic sweep formulation [25]

Due to this spanwise boundary layer flow, the lift coefficient will differ at different sections of the wing. At the root, the maximum lift coefficient is higher than at the tip (for a backward swept wing). This is because the boundary layer is thinner at the root, due to the delayed stall and the opposite happens at the tip. If the wing is swept forward, the stall occurs earlier at the root and is delayed at the tip.

When dihedral or sideslip of the wing increases, the determination of the sweep is not as trivial anymore. The sweep becomes a function of the geometrical sweep as well as sweep regarding flow angles, as displayed in Figure 3.1b. The latter is determined using the projection of the free-stream velocity onto the different panel surfaces. This projection of the free-stream velocity is determined with Equation 3.1, where $\mathbf{n_i}$ is the normal unit vector of each panel.

$$\mathbf{U}^{\mathbf{t}}_{\infty\mathbf{i}} = \mathbf{U}_\infty - (\mathbf{U}_\infty \cdot \mathbf{n_i})\mathbf{n_i} \tag{3.1}$$

In the local axis system, the sweep for each panel can be determined with Equation 3.2

$$\Lambda_{loc,i} = \Lambda_{geom,i} + \Lambda_i^* = \Lambda_{geom,i} + \cos^{-1}\left(\frac{\|\mathbf{U}^{\mathbf{t}}_{\infty\mathbf{i}}\|}{\mathbf{U}^{\mathbf{t}}_{\infty\mathbf{i}} \cdot \mathbf{i}^*}\right) \tag{3.2}$$

Goitia et al. [25] propose a method for correcting 2D lift polars for these 3D effects. For an infinite swept wing, this correction is Equation 3.3 and Equation 3.4. In Equation 3.4, α_0 is de zero lift angle of attack. Note that if $\Lambda_{loc,i}$ is equal to zero, the 2.5D polars are equal to the 2D polars.

$$C_{L(2.5D)} = C_{L(2D)}\cos^2\Lambda_{loc,i} \tag{3.3}$$

$$\alpha_{(2.5D)} = (\alpha_{(2D)} - \alpha_0)\cos\Lambda_{loc,i} + \alpha_0 \tag{3.4}$$

This correction is evaluated with MSES for a NACA4416 airfoil at sweep angles of 0,30 and 60, at M = 0.2 and Re = 6E6. This is compared with CFD data from Goitia et al. [25] in Figure 3.2. The correction of the lift polars for the different sweep angles agrees well with the CFD results. For very low angles of attack, the polar deviates from the CFD results. However, for 0 degrees sweep, the lift polar already deviates from the CFD results and therefore this deviation is not due to the sweep correction.

The approach for finite swept wings is similar to that of the infinite wing. The 2.5D polars are generated using an effective sweep angle. This effective sweep angle at each section is:

$$\Lambda_{eff,i} = \Lambda_{loc,i}\kappa_{\Lambda,i} \tag{3.5}$$

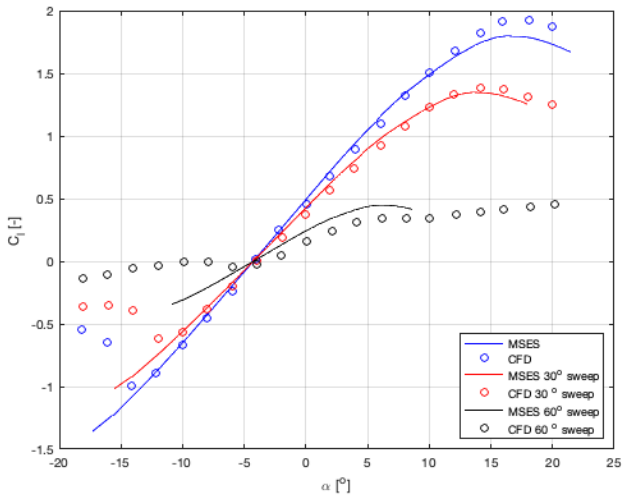


Figure 3.2: Correction of 2D data for an infinite NACA4416 wing at different sweep angles

This effective sweep correction, $\kappa_{\Lambda,i}$ is proposed to be a function of the aspect ratio and the spanwise position. This function is assumed to consist of two different effects: one from the tip and one from the root. The proposed function is described by the following equations:

$$\kappa_\Lambda(y) = 1 + \kappa_{root}(\eta'_{root}) + \kappa_{tip}(\eta'_{tip}) \tag{3.6}$$

$$\eta'_{root}(y) = \frac{y - y_{root}}{\bar{c}} = AR(\eta(y) - \eta_{root}) = AR\eta \tag{3.7}$$

$$\eta'_{tip}(y) = \frac{y - y_{tip}}{\bar{c}} = AR(\eta(y) - \eta_{root}) = AR(\eta - 1) \tag{3.8}$$

Effectively, $\eta'_{root}(y)$ and $\eta'_{tip}(y)$ measure how many chord lengths a spanwise position is from either the root or the tip. The two functions $\kappa_{root}(\eta'_{root})$ and $\kappa_{tip}(\eta'_{tip})$ are designed with Bézier curves by Goitia et al. [25] to fit experimental data. These curves and the resulting κ_Λ are plotted in Figure 3.3. For increasing aspect ratios, the correction factor in the middle of the wing is 1. In the limit of infinity, the effective sweep therefore becomes the local sweep as previously discussed. For aspect ratios below approximately 5, the effect of the root will overpower the effect from the tip. This correction therefore is only applicable for aspect ratios higher than 5, to prevent this nonphysical effect to occur.

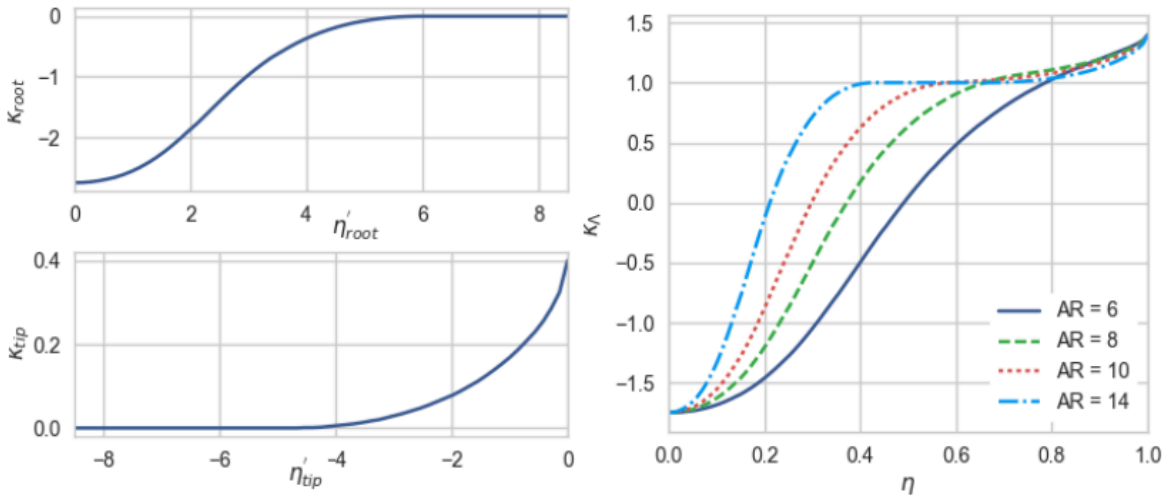


Figure 3.3: $\kappa_{root}, \kappa_{tip}$ and κ_Λ for different aspect ratios from [25]

Using this correction factor κ_Λ, the effective sweep can be determined using Equation 3.5. Equation 3.3 and Equation 3.4 are replaced by Equation 3.9 - Equation 3.11.

$$C_{L(2.5D)} = C_{L(2D)} F_\Lambda^2 \tag{3.9}$$

$$\alpha_{(2.5D)} = (\alpha_{(2D)} - \alpha_0) F_\Lambda + \alpha_0 \tag{3.10}$$

$$F_\Lambda = \begin{cases} 2 - \cos(\Lambda_{eff}), & \Lambda_{eff} < 0 \\ \cos(\Lambda_{eff}), & \Lambda_{eff} \geq 0 \end{cases} \tag{3.11}$$

This sweep correction is implemented in the lifting line. For each panel section, F_Λ is calculated. However, the quarter chord sweep of the current aircraft is very close to zero. When the sideslip angles are low, the sweep correction becomes negligible. In the current model, the maximum value of F_Λ is 1.00002 and the minimum value is 0.999990. In the current model, this sweep correction is neglected. For every section, the lift polar needs to be corrected with a different value for F_Λ and this has a negative impact on the computational cost. Therefore, due to the added computational effort and the minimal impact on the result, this sweep correction is neglected in the lifting line. However, when the geometry of the aircraft is adjusted, this correction can be applied on the different sections.

3.2.4. Wake

The current wake model implemented in the lifting line is a frozen wake vortex model. Induction by the aircraft on the wake or self-induced velocities are neglected and geometric constraints and the free-stream velocity determine the shape of the wake. The wake is, similar to the control and vortex points, generated in four separate parts: one for the main wing, one for the elevator and two for the different vertical rudders.

The wake generation function needs the vortex points, local chord, flow angles, free-stream velocity, time step size and amount of time steps that the wake will be convected as input. The first wake point will be positioned a quarter chord behind the trailing edge of the local section. The second wake point will be convected with $\mathbf{U}_\infty dt$.

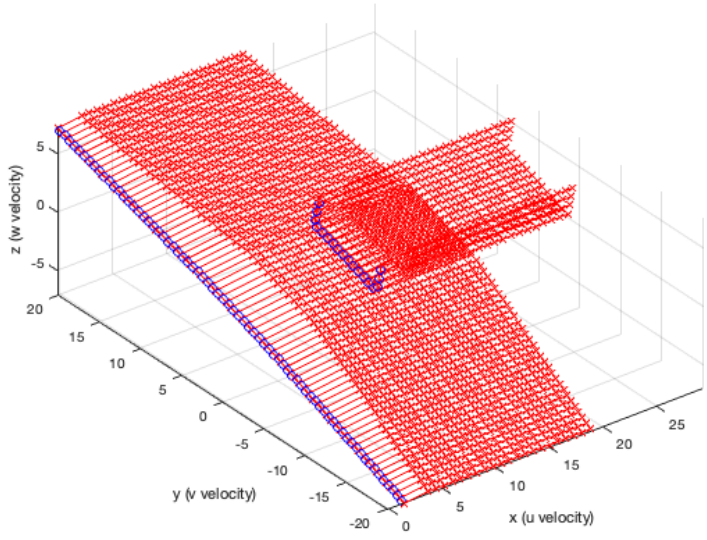


Figure 3.4: Control and wake points as generated by Matlab

This results in four matrices of size $N \times M$, where N is the number of time steps +2 and M is the number of vortex points. The +2 of N are the vortex points located on the wing and the second row are

the points shed a quarter chord behind the trailing edge. After these two rows, wake points are added with the previously mentioned increments. The resulting geometry is displayed in Figure 3.4.

3.2.5. Aerodynamic Influence Coefficient (AIC) matrix

Now that all the wake and control points are known, the AIC matrix can be computed. The choice is made to follow the method explained by Katz et al. [31] instead of the method set out in Section 2.1.6 by Phillips et al. [47]. The reason for this choice is the easy numerical implementation of the method by Katz compared to the method by Phillips et al.

The starting point for the calculation of the AIC matrix is the Biot-Savart law. The general form of the Biot-Savart law for the induced velocity at an arbitrary point due to existing vorticity is:

$$\mathbf{u} = \frac{1}{4\pi} \int_V \frac{\omega \times \mathbf{r}}{\|r^3\|} dV \tag{3.12}$$

This expression can be simplified for a straight vortex filament with constant circulation Γ. The induced velocities in $x-, y-$ and $z-$direction are calculated using the set of equations of Equation 3.13. Point 1 and 2 are the beginning and end of the vortex filament, point p is the the point where the induced velocity is required and the direction of Γ determines the sign.

Next to these equations, the code checks if R_1, R_2 or R_{1_2sqr} is smaller than a certain number (in this case this number is chosen to be 0.01). If one of these values is smaller than 0.01, the induced velocity is set equal to zero. Otherwise, when the control points are located very closely to the vortex segments, the induced velocity approaches infinity, which is nonphysical.

$$\begin{aligned}
R_1 &= \sqrt{(x_p - x_1)^2 + (y_p - y_1)^2 + (z_p - z_1)^2} \\
R_2 &= \sqrt{(x_p - x_2)^2 + (y_p - y_2)^2 + (z_p - z_2)^2} \\
R_{1_2x} &= (y_p - y_1)(z_p - z_2) - (z_p - z_1)(y_p - y_2) \\
R_{1_2y} &= -(x_p - x_1)(z_p - z_2) + (z_p - z_1)(x_p - x_2) \\
R_{1_2z} &= (x_p - x_1)(y_p - y_2) - (y_p - y_1)(x_p - x_2) \\
R_{1_2sqr} &= R_{1_2x}^2 + R_{1_2y}^2 + R_{1_2z}^2 \\
R_{0_1} &= (x_2 - x_1)(x_p - x_1) + (y_2 - y_1)(y_p - y_1) + (z_2 - z_1)(z_p - z_1) \\
R_{0_2} &= (x_2 - x_1)(x_p - x_2) + (y_2 - y_1)(y_p - y_2) + (z_2 - z_1)(z_p - z_2) \\
K &= \frac{\Gamma}{4\pi R_{1_2sqr}} \left(\frac{R_{0_1}}{R_1} - \frac{R_{0_2}}{R_2} \right) \\
U &= K R_{1_2x} \\
V &= K R_{1_2y} \\
W &= K R_{1_2z}
\end{aligned} \tag{3.13}$$

The different coefficients of the AIC matrix are determined by adding the influence of all the vortex segments of a single horseshoe vortex. For a certain control point, the equations from Equation 3.13 are run for every vortex segment. A circulation of unit strength is assumed for all vortex segment and the sign is determined according to the right hand rule. For each control point, this comes down to $3+2$ (time steps) number of vortex segments. The first 3 are the influence of the bound vortex and the first part of both the legs of the horseshoe vortex to the points shed a quarter chord behind the trailing edge. The rest of the vortex segments of the legs of the horseshoe vortex are looped over and stored in a 3D array. Summing over the third dimension and adding all the segments results in the influence of the entire horseshoe vortex on this control point. This process is repeated for all control points and all horseshoe vortices, resulting in a $N \times N$ matrix, where N is the amount of control points (and thus also the amount of horseshoe vortices).

This $N \times N$ matrix is composed out of 16 smaller matrices, since it is composed from four different

lifting lines. Each lifting line has an influence on itself and the other three wings. Thus, this results in $4 \times 4 = 16$ matrices. The code combines these 16 matrices into one big matrix: the AIC matrix. In total there are 3 AIC matrices, one for each directional velocity and thus $3 \times 16 = 48$ smaller matrices. The only inputs required to compute these matrices, are the 4 sets of control points, 4 sets of wake points and the cut-off distance.

3.2.6. Circulation distribution calculation

When the AIC matrices have been compiled, the iteration loop can be initiated to find the circulation distribution. This iteration loop consists of the following steps:

- The following set of equations is solved for the induced velocities:

$$\begin{aligned} [u] &= [AIC_u][\Gamma] \\ [v] &= [AIC_v][\Gamma] \\ [w] &= [AIC_w][\Gamma] \end{aligned} \tag{3.14}$$

  For the first iteration, an initial circulation distribution is inserted of a $N \times 1$ array of ones. The resulting u,v and w arrays are all of size $N \times 1$.
- For each panel, the velocity normal to the panel and the tangential velocities are determined using, Equation 3.15, where $\mathbf{n_i}$ is the normal unit vector of panel i and $\mathbf{c_i}$ is the unit vector in chordwise direction.

$$\begin{aligned} U_{n,i} &= (\mathbf{U}_\infty + [u_i; v_i; w_i]) \cdot \mathbf{n_i} \\ U_{c,i} &= (\mathbf{U}_\infty + [u_i; v_i; w_i]) \cdot \mathbf{c_i} \end{aligned} \tag{3.15}$$

- The angle of attack at each panel is determined:

$$\alpha_i = \tan^{-1}\left(\frac{U_{n,i}}{U_{c,i}}\right) + \alpha_{twist} \tag{3.16}$$

- Using the Kutta-Joukowski theorem and the viscous lift polars, a new circulation distribution can be found using:

$$\Gamma_i = \frac{1}{2} c_i U_i C_l(\alpha_i) \tag{3.17}$$

- The newly found circulation distribution of Equation 3.17 and the initial circulation distribution are combined to find the circulation distribution for the next iteration:

$$[\Gamma] = 0.95[\Gamma_{old}] + 0.05[\Gamma_{new}] \tag{3.18}$$

- The convergence is checked to determine if the loop is terminated. This number is defined as max(abs$(\Gamma_{old} - \Gamma_{new})$). As long as this number is larger than 1E-5, the loop repeats itself. When the convergence is smaller than 1E-5, the iteration is stopped.

3.2.7. Aerodynamic force calculation

The aerodynamic force on the aircraft can be calculated straight forwardly when the circulation distribution has been found. This circulation distribution provides an angle of attack for each panel from the final iteration. Using this angle of attack, the lift and drag of the different wings can be found using Equation 3.19.

$$\begin{aligned} L_i &= \frac{1}{2}\rho U_\infty^2 C_l(\alpha_i) dS_i \\ D_i &= \frac{1}{2}\rho U_\infty^2 C_d(\alpha_i) dS_i \end{aligned} \tag{3.19}$$

Also, the moments can be determined easily by multiplying the forces of each panel with the respective arm.

3.2.8. Chordwise distribution

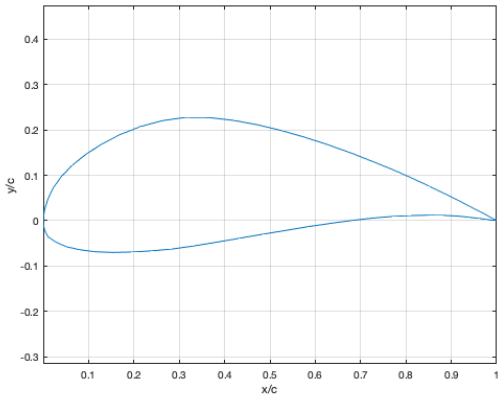
Initially, the idea was to not have information regarding the chordwise force distribution. However, after the model was working, the idea changed and the chordwise force distribution is required. To not change the working model, chordwise information is obtained using the pressure distribution from the 2D viscous polars. As the lifting line model only gives a force per spanwise section and the structural model requires a spanwise as well as a chordwise distribution, the following approach is taken to convert this point force at a spanwise location to a chordwise distributed force. As an example, the reconstructed MRevE airfoil from Eijkelhof [18], the pressure coefficient distribution and the lift coefficient distribution is shown in Figure 3.5.

From the pressure coefficient, the lift coefficient over the airfoils upper and lower surface is determined using:

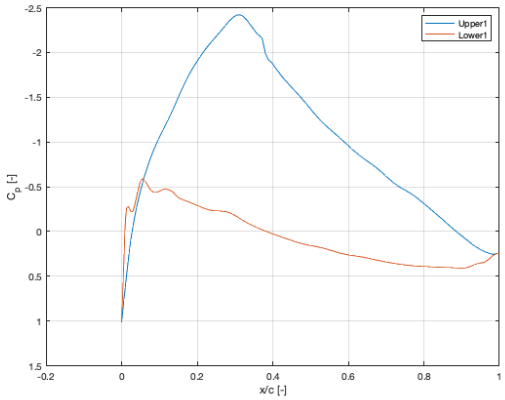
$$C_l = \int C_{p,l} - C_{p,u} d\frac{x}{c} \tag{3.20}$$

Note that in Equation 3.20 the influence of the skin friction is neglected. However, this factor will be discussed in Section 4.5.

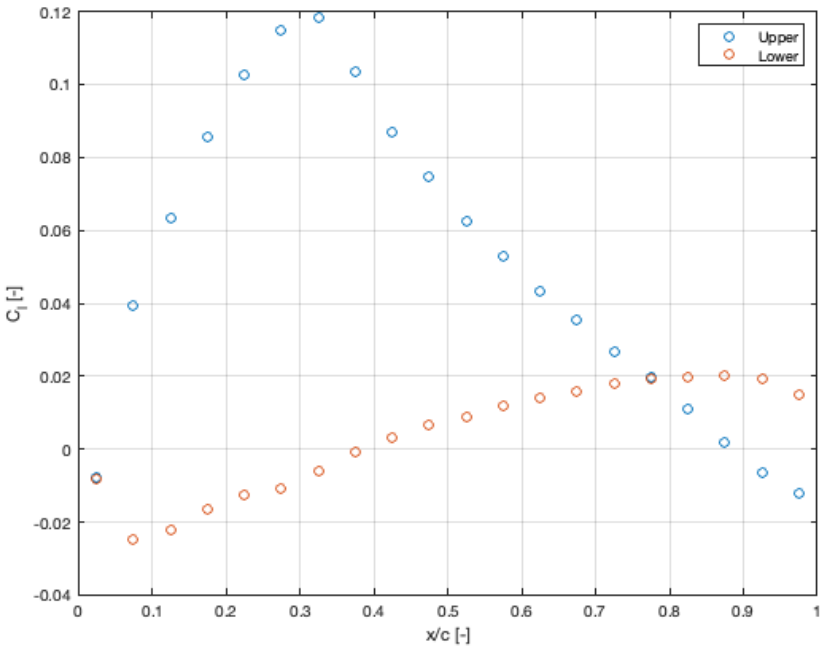
For both the upper and the lower surface, the lift coefficients for a given number of sections is determined. Figure 3.5c uses 20 sections for each surface. Each airfoil surface is normally distributed for this number of sections in $\frac{x}{c}$. Each section approximates the pressure distribution integral numerically and places this lift coefficient in the middle of the panel, as can be seen in Figure 3.5c. Afterwards, the code determines the percentage per panel compared to the total lift coefficient of the entire airfoil. This percentage is multiplied with the lift force outputted by the lifting line, to arrive at a chordwise distribution of this force.



(a) The reconstructed MRevE airfoil from [18]



(b) Pressure coefficient distribution over the MRevE airfoil at $\alpha = 0, M = 0.15$ and $Re = 1.2E7$



(c) Lift coefficient distribution over the MRevE airfoil at $\alpha = 0, M = 0.15$ and $Re = 1.2E7$

Figure 3.5: The reconstructed MRevE airfoil, pressure coefficient distribution and lift coefficient distribution.

3.3. FSI model

The different aspects of the aerodynamic model and their implementation have been discussed in Section 3.2. In this section, the implementation of the structural model and its coupling will be discussed.

3.3.1. Initialization

Before the FSI iteration loop is started, the deflection, structural twist, dihedral due to the deformation and the overall force vector are set to zero. This is to make sure that for the first iteration loop, the aerodynamic model evaluates the undeformed wing.

Furthermore, the convergence criteria are set. There are three criteria implemented and if one of the three is reached, the code exits the iteration loop. First of all, a maximum number of iterations is set to make sure that if the iteration loop does not converge, it will not run infinitely. The maximum number of iterations is set to 10, as the experience has been that the loop has never exceeded this number when it converges. The other two convergence criteria are for the lift and deflection. The loop stops if the

difference in lift between two iterations is less than 100 Newton or if the maximum deflection between two iterations anywhere on the wing drops below 1 mm.

3.3.2. Iteration loop

The iteration loop runs the aerodynamic model and structural model in a loop for every state, which are determined by an array of angles of attack, roll angles and velocities. The loop runs in the following order:

1. The deflection, structural twist, structural dihedral and force are taken from the previous state or iteration as input for the iteration. In case there is no previous state or iteration, these parameters are taken to be zero.
2. The aerodynamic model is run using the deflection, structural twist and dihedral as inputs, outputting the force distributions on both surfaces and a new force vector.
3. The structural model is run with the force distributions of the surfaces, giving the deflection, twist and dihedral due to the deformation. The forces outputted by the aerodynamic model are transferred to the structural mesh within the structural model, using the coupling method described in Section 2.3. At the end of the structural model, the deformations are transferred back to the aerodynamic mesh and thus the output of the structural model is defined on the aerodynamic mesh.
4. The convergence criteria are checked. If none of the criteria is reached, step 2 and 3 are repeated.
5. Before proceeding with the next state, the force vector, structural twist, structural dihedral and deflection are stored in matrices. This gives matrices for the four variables where the result for every state can be found.

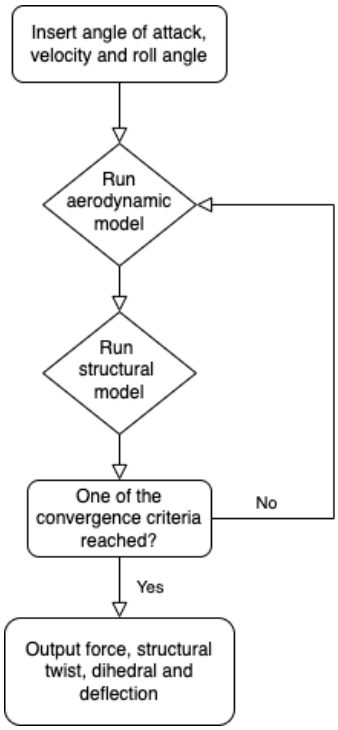


Figure 3.6: Flowchart of the FSI loop.

4

Aerodynamic model validation

Some general cases have been put into the written lifting line code in order to validate the model. Next to that, the chordwise force distribution method is discussed.

4.1. One point

A horseshoe vortex is created in matlab to validate the induced velocities on the control point located on the bound vortex. Katz et al. [31] show that the induced velocity of a semi infinite straight vortex filament is equal to:

$$v = \frac{\Gamma}{4\pi d} \tag{4.1}$$

The control point is located at the origin, vortex points at (0,1),(0,-1) and to simulate 'infinity', the other two points are located at (100,1) and (100,-1). With d being 1 in this case. According to Equation 4.1, the code should give 0.79577 (for a Γ value of 10). The Matlab code indeed gives 0.7957.

4.2. Multiple vortices

If multiple vortices are added, the induced velocities can also be easily determined using the same method. Vortices are added at a d of 3,2,-2 and -1 as can be seen in the following figure:

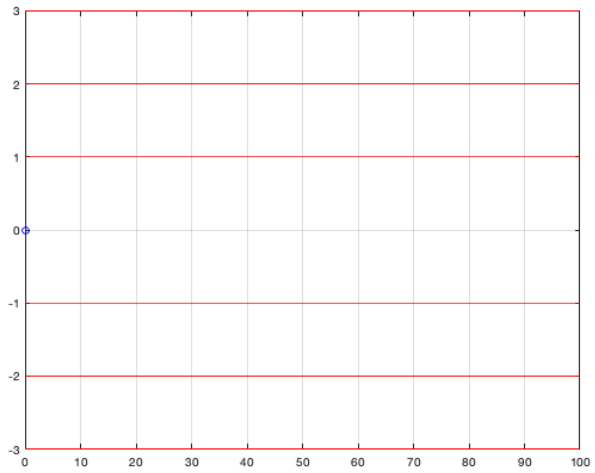


Figure 4.1: Multiple vortices validation, represented in 2D

The values according to Equation 4.1 should be: 0.2652 0.3979 0.7958 -0.7958 -0.3979 -0.2652. The code gives values of: 0.2651 0.3978 0.7957 -0.7957 -0.3978 -0.2651. The very small discrepancies

are due to the fact that 100 is not infinity, but if these points are taken closer to infinity, the solution approaches the correct values.

4.3. Three horseshoe vortices

The third validation is three horseshoe vortices of different strengths. The circulation of the three vortices is 5,10 and again 5. These horseshoe vortices are shown in the following figure:

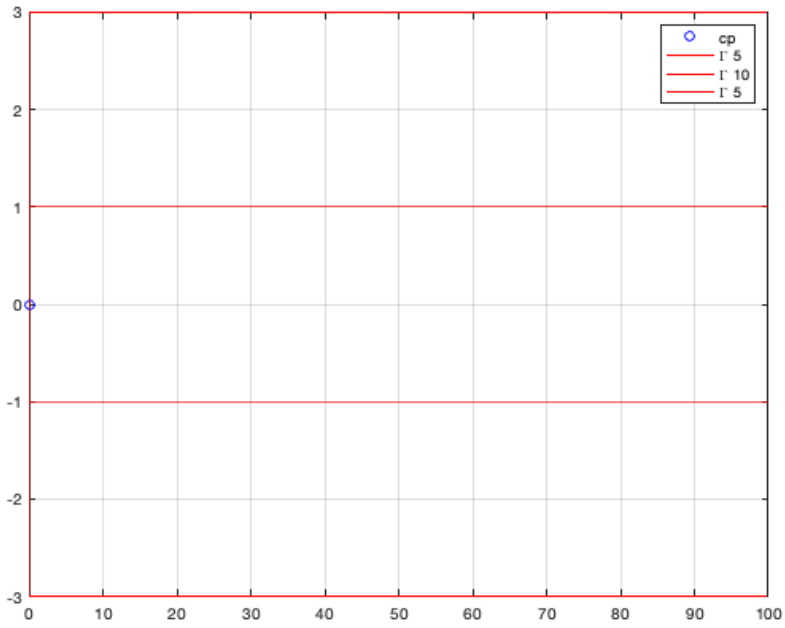


Figure 4.2: 3 Horseshoe vortices

In total there are 6 trailing vortices (2 per horseshoe) and 3 bound vortices. The bottom and top one have a circulation of 5 and the middle one is chosen to be 10. The influence of the bound vortices is expected to be zero, this is due to the fact that the angle between r1 and r2 is zero and looking at the biot-savart law, if the cross product between these vectors is zero, the induced velocity will be zero. Calculating the induced velocities of the trailing vortices with Equation 4.1, the code should give: -0.1326 0.3979 -0.7958 -0.7958 0.3979 -0.1326.
The code gives the values: -0.1326 0.3979 -0.7957 -0.7957 0.3979 -0.1326, which corresponds to these values. Next to this, the induced velocities by the bound vortices are zero. This corresponds to what was expected.

4.4. Multiplane and box-wing

As the MegAWES model has a tail with wings in the same plane, the code needs to be tested for a similar case. This can be done with multiplane and box wings. Gagnon et al. [24] evaluate a box-wing at M=0.3. The CFD solver used, solves the steady Euler equations. Thedens compares the results of the CFD solver of Gagnon et al. with results from APAME. The box-wing configuration is shown in Figure 4.3.

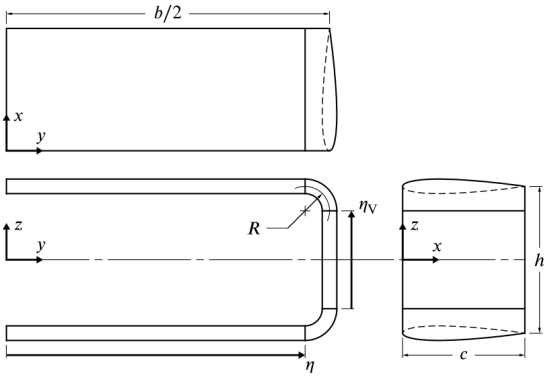


Figure 4.3: Box-wing configuration [24]

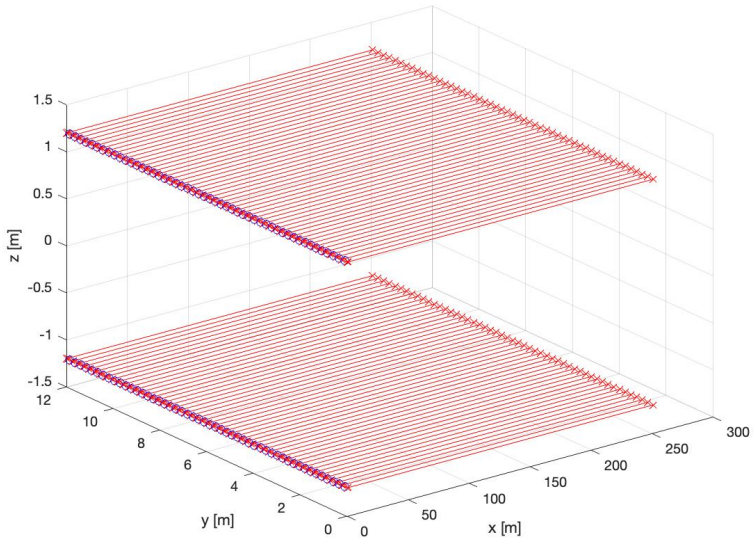
The wings span b, fillet radius R and height over span $\frac{h}{b}$ are all defined as a function of the chord c, with $b = 12c$, $R = 0.15c$ and $\frac{h}{b} = 0.2$. The wing its airfoil is a NACA0012 airfoil over the entire wing. As the wing is evaluated at a Mach number of 0.3, the Reynolds number is equal to 7.24E6.

Both Gagnon et al. and Thedens evaluated the angle of attack for a target lift coefficient C_L of 0.5. The CFD solver of Gagnon et al. found the angle of attack to be 5.41, whereas Thedens achieved a lift coefficient of 0.5 at an angle of attack of 5.86.

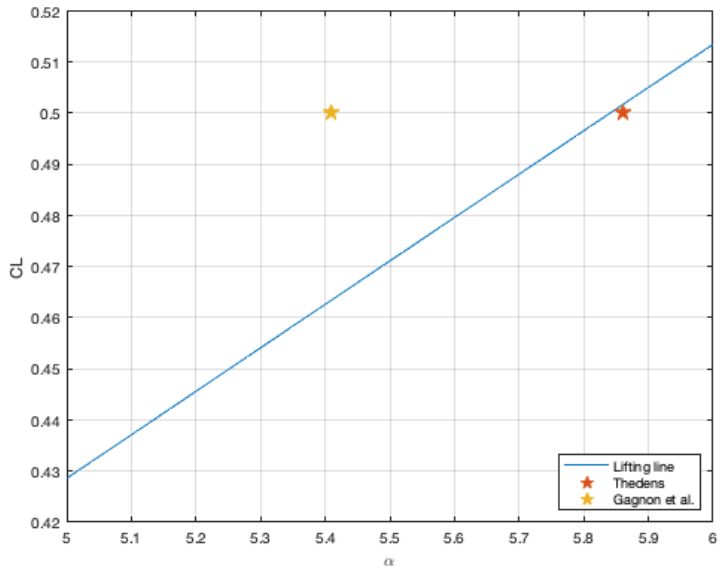
First of all, only the multiplane wing is run using the written Matlab code to see if the code is able to evaluate multiple wings in the same plane. Next, the full box-wing is evaluated.

4.4.1. Multiplane wing

The multiplane wing is the box wing of Figure 4.3 without the vertical endplate to join the two wings.



(a) Multiplane wing including wake



(b) 3D lift coefficient for the multiplane wing for angles of attack between 5 and 6 degrees

Figure 4.4: Multiplane wing

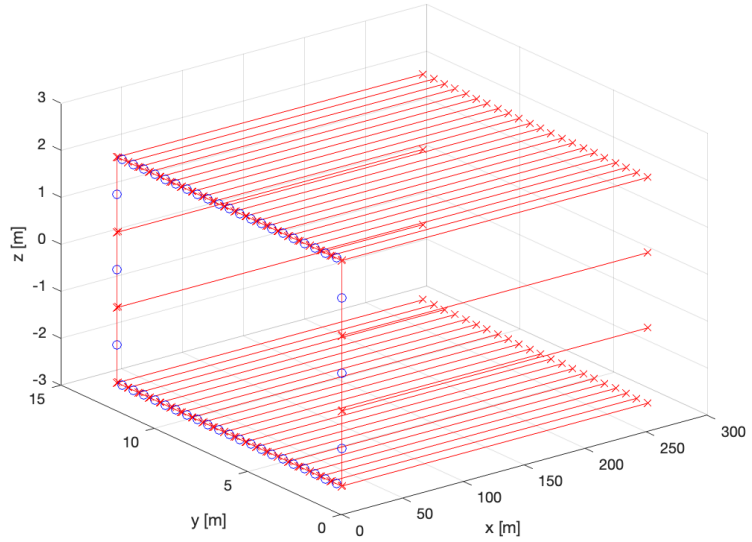
A sensitivity study is performed for the number of sections per wing. The lift coefficient for more than 18 sections per wing changes with less than 0.1% and therefore 18 sections per wing is taken for the multiplane wing. As the endplate joining the two wings will not produce lift if the wing is only under an angle of attack, the result for this multiplane wing should be close to the results from Thedens and Gagnon et al. for the box-wing. The 3D lift coefficient for this multiplane wing is displayed in Figure 4.4b.

For the target lift coefficient of 0.5, the multiplane wing needs to be at an angle of attack of 5.84 degrees, which is close to the 5.86 from Thedens. Gagnon et al. found this angle of attack to be 5.41 degrees.

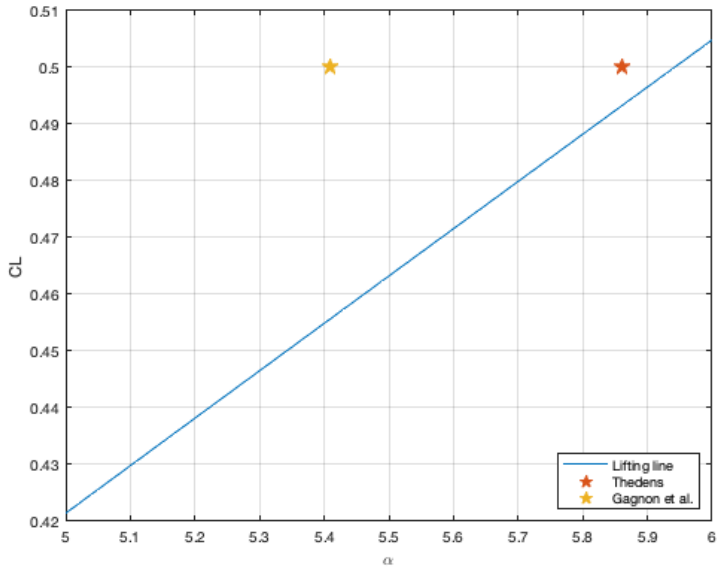
4.4.2. Box-wing

Now that the multiplane has been tested, the full box-wing could be implemented in Matlab. The multiplane wing of Figure 4.4a is connected at both sides by a vertical wing, Figure 4.5a. Whereas the

model used by Thedens and Gagnon et al. have a radius between the vertical and horizontal wing to reduce compressibility effects, the wings here have been connected by a 90 degree angle.



(a) Box-wing including wake



(b) 3D lift coefficient for the box-wing for angles of attack between 5 and 6 degrees

Figure 4.5: Box-wing

Again, a sensitivity study has been done to see how many sections are needed so that the lift coefficient changes less than 0.1% when increasing this number. For the box-wing this resulted in 21 sections per wing.

The angle of attack for the box-wing to achieve the desired lift coefficient of 0.5 is now 5.94. Again, Gagnon et al. found an angle of attack of 5.41 and Thedens of 5.86.

4.5. Chordwise force distribution

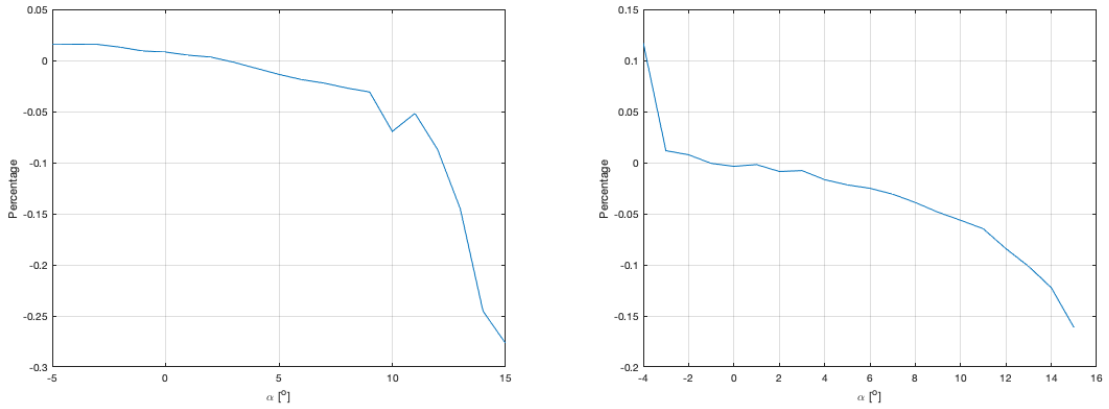
As discussed in Section 3.2.8, the code extracts the lift coefficient data from the pressure coefficient data using numerical integration approximations. For a given amount of sections per element, the lift coefficient is determined per panel. To validate if the approach taken is valid, two different cases are investigated: a single and a multi-element airfoil. The Makani M600 multi-element airfoil and the MRevE airfoil from Eijkelhof [18]. The MRevE airfoil is an adjusted version of the first element of the M600 multi-element airfoil. The camber of the MRevE airfoil is increased to be able to reach higher lift coefficients. The multi-element airfoil of the M600 and its distributions can be found in Figure 4.7. The MRevE airfoil has been previously discussed and shown in Figure 3.5.

To check the validity of the chosen method, both airfoils are evaluated over a range of angles of attack. The difference between the pressure distribution method and the values given by MSES is expressed in percentage using Equation 4.2 and for the two airfoils this results in Figure 4.6.

$$\text{Percentage} = \frac{\Delta C_l}{C_{l,\text{MSES}}} 100 \tag{4.2}$$

The expectation for the lift coefficient for increasing angles of attack, is that the deviation from the MSES lift coefficient will grow more profoundly. For increasing angles of attack, the flow around the airfoil will become more turbulent. The transition point on the upper surface will move upstream and the transition point on the lower surface downstream. The upper point however will move further upstream than the lower point downstream, resulting in a more turbulent flow over the entire surface. Turbulent flow results in an increase in skin friction and this factor is not taken into account when computing the lift coefficient with the pressure distributions. This phenomenon also occurs for negative angles of attack, but the other way around: the transition point on the upper surface moves downstream and the transition point on the lower surface upstream. Therefore the deviation is expected to grow when moving away from an angle of attack of zero. This is indeed observed in Figure 4.6.

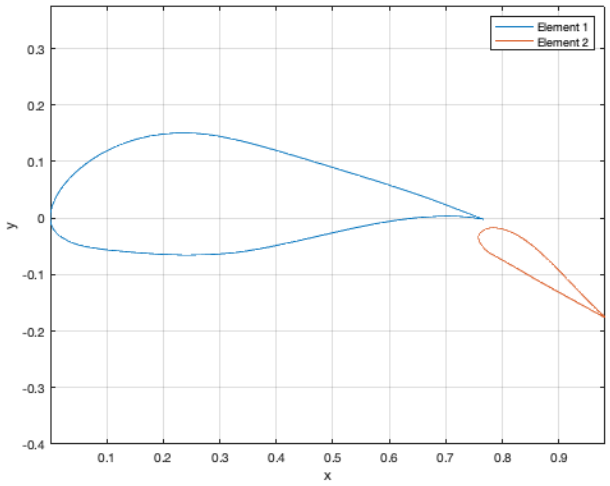
Below the stall angle of attack, the deviation is well below 1% compared to the lift coefficient determined with MSES. On top of that, the lift coefficients are not used as absolute numbers, but to determine a distribution over the surface and then multiplied with the lift force computed by the lifting line. Therefore this method is accurate enough for this model to determine chordwise force distribution.



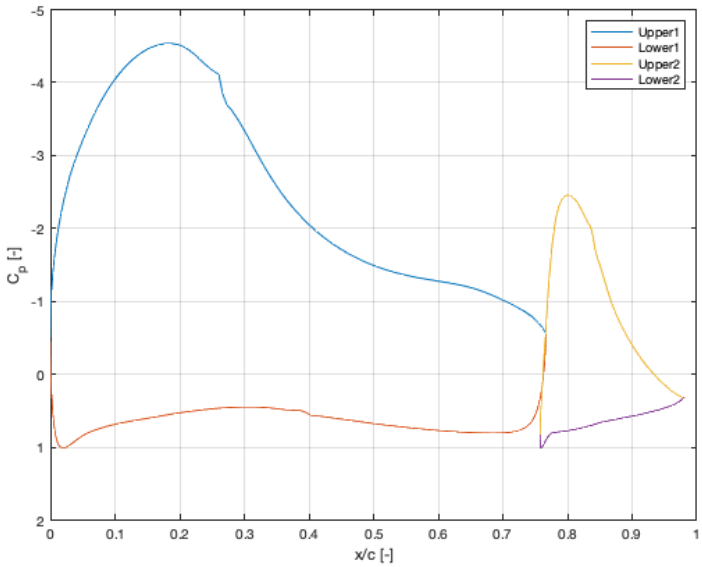
(a) The deviation in percentage of the M600 multi-element airfoil compared to MSES

(b) The deviation in percentage of the MRevE airfoil compared to MSES

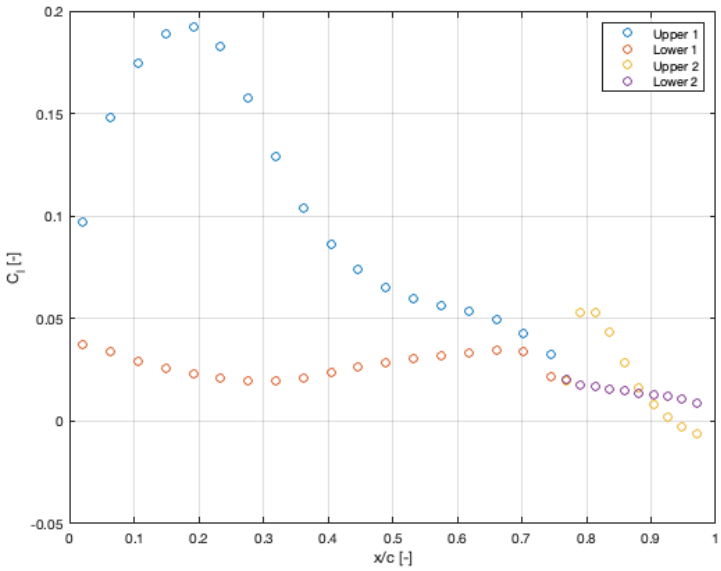
Figure 4.6: The deviation in percentage from the chosen method compared to the values of MSES



(a) The M600 multi-element airfoil



(b) Pressure coefficient distribution over the M600 airfoil at $\alpha = 0, M = 0.2$ and $Re = 7E6$



(c) Lift coefficient distribution over the M600 airfoil at $\alpha = 0, M = 0.2$ and $Re = 7E6$

Figure 4.7: Makani M600 multi-element airfoil, pressure coefficient distribution and lift coefficient distribution.

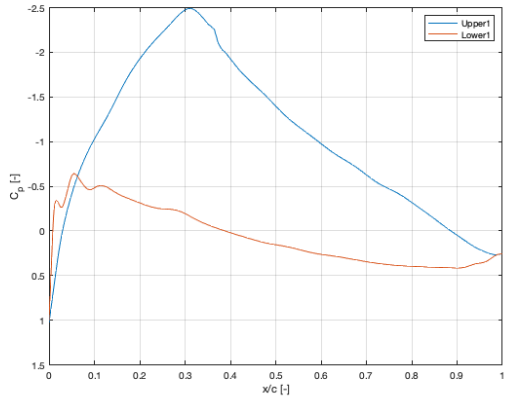
5

Model considerations

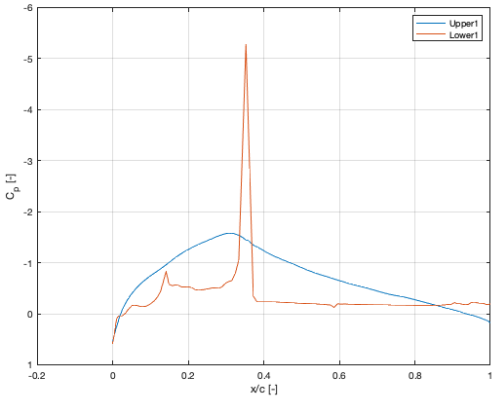
In this chapter, various aspects, choices and limitations of the FSI model are discussed.

5.1. Reynolds number variations

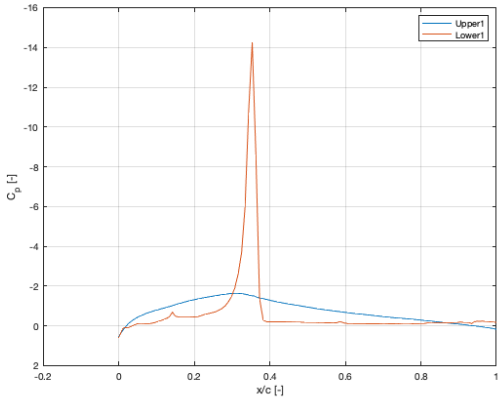
In this section, the chordwise force distribution method of Section 4.5 is discussed. The presented method works well for a large angle of attack range. However, these results are evaluated at a single Reynolds number. When the Reynolds number is increased, MSES returns peculiar pressure distributions for certain Reynolds numbers.



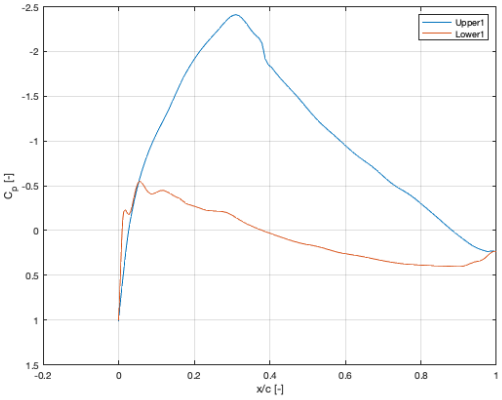
(a) Pressure distribution, Re = 2E7, $\alpha = -0.25$



(b) Pressure distribution, Re = 2E7, $\alpha = 0$



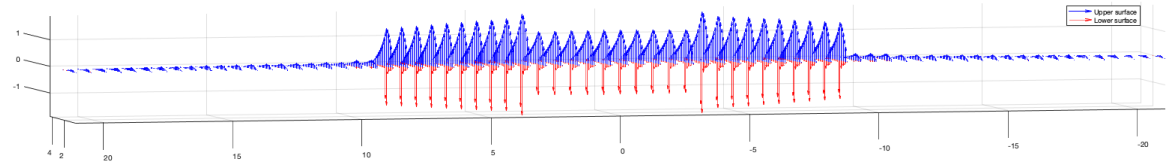
(c) Pressure distribution, Re = 2E7, $\alpha = 0.25$



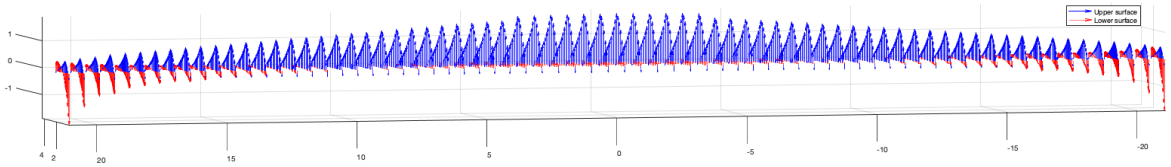
(d) Pressure distribution, Re = 1.6E7, $\alpha = 0.25$

Figure 5.1: Pressure distributions for the MRevE airfoil at 2 Reynolds numbers

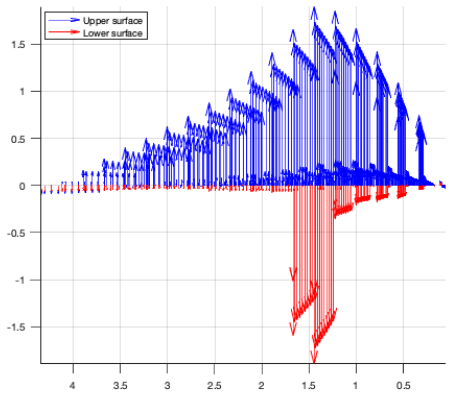
Subfigures (a)-(c) of Figure 5.1 show the pressure distributions of the MRevE airfoil at a Reynolds number of 2E7 and angles of attack of -0.25,0 and 0.25. Whereas for an angle of attack of -0.25, the pressure distributions looks normal, the same airfoil at the same Reynolds number starts to develop a pressure peak on the lower surface. This pressure peak is present for the rest of the angles of attack at this Reynolds number. Figure 5.1d, shows the same airfoil at a lower Reynolds number, where the pressure peak is not present. Next to this, the mpolar function of MSES is not able to converge for this Reynolds number and airfoil. When using these pressure distributions, the wing loading becomes very concentrated around the pressure peaks and much higher than on any other part of the wing where no pressure peaks arise. This is due to the fact that these distributions are used to determine the chordwise force distribution (Section 4.5). The wing loading for a scenario where the angles of attack are such that the pressure distributions with pressure peaks are used and a scenario where this is not the case are shown in Figure 5.2. The result of the pressure peaks is the resulting chordwise force distribution that serves as an input for the structural model. In the case of the pressure peaks, the structural model returns certain points on the lower surface to actually break away from the rest of the wing and the input for the next iteration of the aerodynamic model will not represent the actual wing. Therefore it is necessary to remove the pressure peaks, otherwise the entire FSI model will break down.



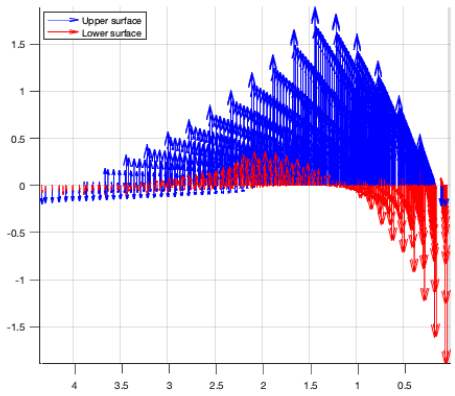
(a) Wing loading at higher loading



(b) Wing loading at lower loading



(c) Side view of the wing loading at higher loading



(d) Side view of the wing loading at lower loading

Figure 5.2: The wing loading at two loading scenarios: one in the range of the pressure peaks and one outside of this range

The question arises if the pressure peaks do actually arise, if it is due to the poorly designed airfoil. As explained earlier, the airfoil is reverse designed from images. It could also be due to the limitations of MSES. To answer this question, the MRevE airfoil is evaluated with XFLR5 [65]. XFLR5 uses the analysis capabilities of XFOIL and therefore should be capable of analysing the MRevE airfoil. The comparison of both programs is shown in Figure 5.3 at a Reynolds number of 2E7 and an angle of attack of -0.25. Over the majority of the airfoil, MSES predicts a higher ΔC_p compared to XFLR5. This results in a higher predicted lift coefficient by MSES than predicted by XFLR5, but XFLR5 does show a similar pressure distribution.

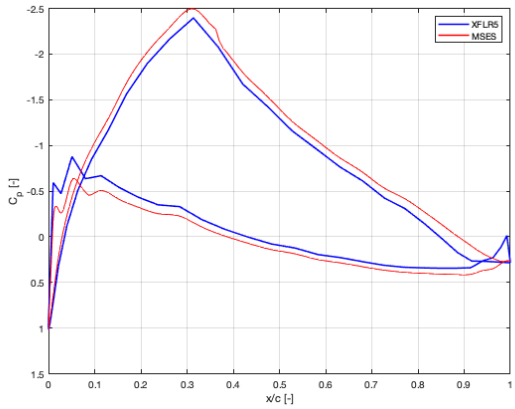


Figure 5.3: XFLR5 and MSES compared for Reynolds = 2E7 and $\alpha = -0.25$

If XFLR5 is used to evaluate the conditions of Figure 5.1, the resulting pressure distributions to not show the same peaks as predicted by MSES:

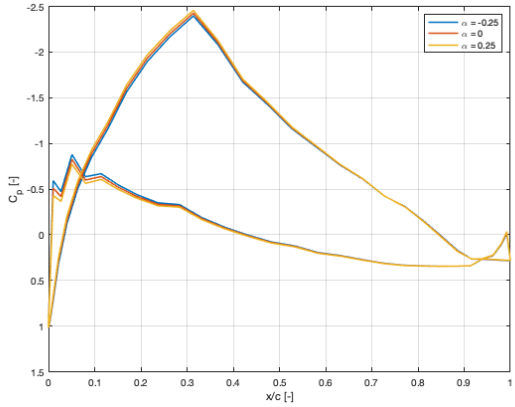


Figure 5.4: MRevE airfoil evaluated with XFLR5 for Reynolds = 2E7 and $\alpha = -0.25, 0, 0.25$

Figure 5.4 strengthens the idea that the pressure peaks of Figure 5.1 are due to some MSES induced problem. To generate the pressure distributions, one run is done from the first negative angle of attack down and one run from 0 upwards. This indicates that MSES struggles to find a proper solution for a zero angle of attack. When the positive angle of attack run is adjusted from 0 and upwards to 0.01 and upwards, MSES returns pressure distributions without any pressure peaks. For all of the Reynolds numbers, this solves the problem of the pressure peaks. The underlying reason why MSES has problems converging for an angle of attack of 0 at certain Reynolds numbers is not known. As improving the source code of MSES is outside the scope of this thesis, no extra effort is put into solving this particular problem.

Where this solves the problem for the pressure distributions, mpolar is still unable to converge for all of the Reynolds numbers. Converging issues still occur and the mpolar function can not produce lift coefficient polars for all Reynolds numbers. The choice is made to limit the Reynolds number for the lifting line to a lift polar that is fully converged (in this case 1.2E6). As the lift polars are used to determine the spanwise lift distribution and the pressure distributions are used to determine the chordwise distribution, this choice entails that overall the lift is slightly underpredicted due to the use of lift polars at lower Reynolds numbers. For determining the chordwise distribution, the correct Reynolds numbers are used.

5.2. Sensitivity analysis

In this section, a sensitivity analysis is performed on the number of panels spanwise as well as chordwise.

5.2.1. Spanwise panels

The biggest contributor to the computational effort is the amount of spanwise panels. First of all, adding extra spanwise panels increases the time needed for the spanwise force distribution. Next to this, the chordwise force distribution needs to be determined for every panel. The computational time can be reduced if the number of panels is chosen such that the calculated force does not change significantly when more panels are added. Two different scenarios have been evaluated to determine this number: one relevant to the cycle used by Eijkelhof [18], Figure 5.6.

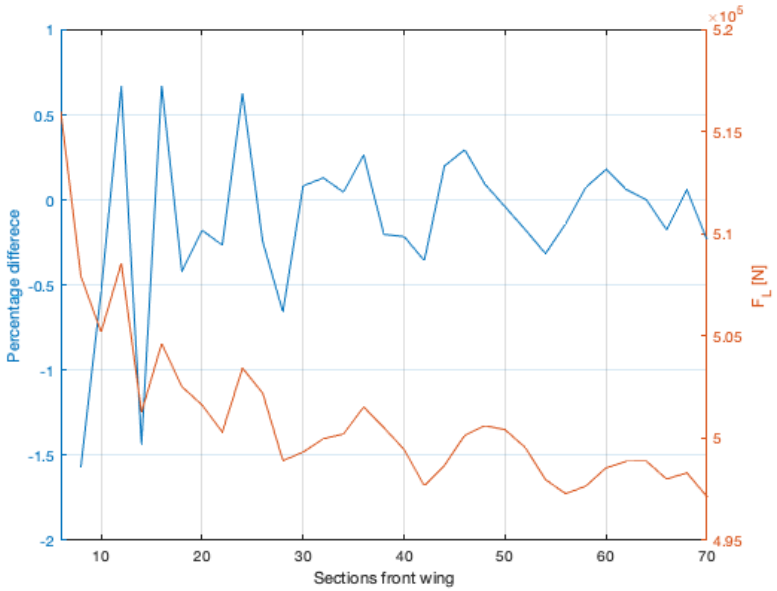


Figure 5.5: Sensitivity analysis of number of panels front wing in lower loading

Both figures show the absolute lift force generated by the aircraft as well as the percentage difference from the previous number of panels. Figure 5.5 shows the loading case relevant to the cycle of Eijkelhof. An angle of attack of -3 degrees, roll angle of -20 degrees and a flight velocity of 80 meters per second are used. The number of sections of the front wing is evaluated from 6 to 70 for these flight parameters. When the number of sections is increased, the lift force converges, however not smoothly. The rough character of the difference is probably due to the interaction between the wake of the front and the rear. Figure 5.6 shows a more general loading case of an angle of attack of 5 degrees, roll angle of 0 and a flight velocity of 100 meters per second. This case shows a smoother convergence. Looking at both Figure 5.5 and Figure 5.6, when 30 spanwise sections are chosen, the difference when adding more sections is less than 0.1 %. This difference is deemed as small enough that adding more sections is not worth the extra computational effort. For that reason the choice is made to use 30 spanwise sections.

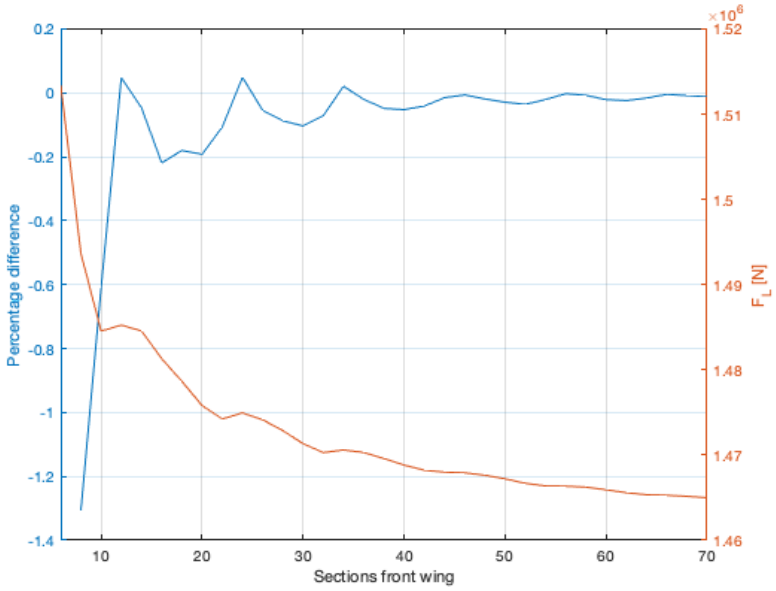


Figure 5.6: Sensitivity analysis of number of panels front wing in high loading

5.2.2. Chordwise panels

Next to the amount of spanwise panels, the amount of chordwise panels needs to be determined. To be able to do so, the same two cases as discussed in Section 5.2.1 are evaluated. Figure 5.7 shows the deflections for varying number of chordwise panels and Figure 5.8 the structural twist.

Differences in both deflection and structural twist are due to the varying stiffness of the wing chordwise. Depending on where the force is chosen to be applied chordwise, both deflections and structural twist can vary for the number of sections. Although both the deflections and the structural twist do not show a big dependency on the number of chordwise sections, it does influence the result. However, next to the influence itself, computational effort should also be taken into account. With an increasing number of chordwise panels, the computational effort increases.

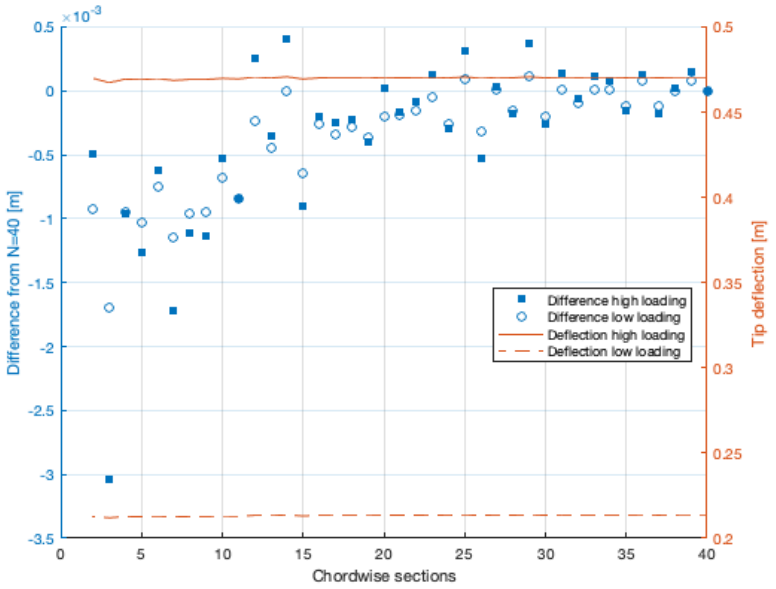


Figure 5.7: Tip deflection for varying number of chordwise panels

Next to the deflections, Figure 5.7 also shows the difference to the result of 40 chordwise panels. 40 chordwise panels is assumed to be enough to capture the chordwise effect properly. For the other number of chordwise panels, the result is compared to the result of 40 panels and the difference is shown. A similar approach is taken for the structural twist, as can be seen in Figure 5.8.

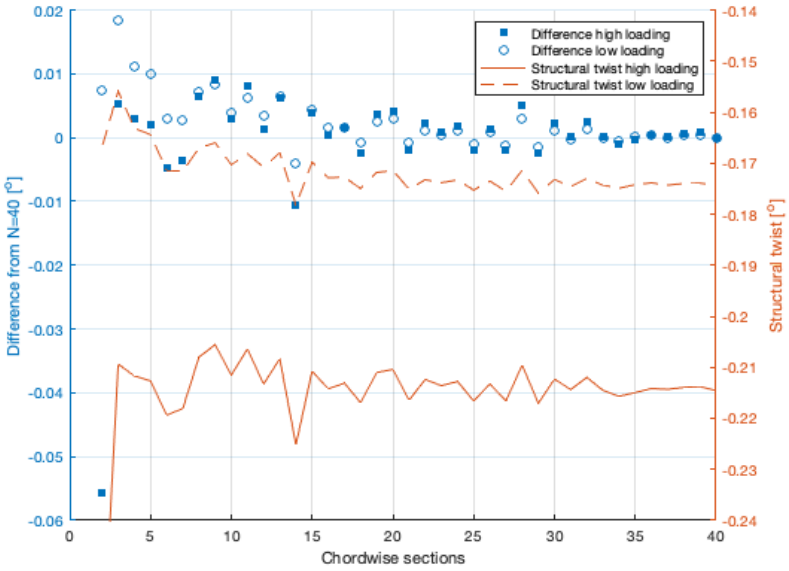


Figure 5.8: Structural twist for varying number of chordwise panels

Looking at both Figure 5.7 and Figure 5.8, 5 chordwise panels are already sufficient to capture the correct behaviour of the wing. Due to the stiff nature of the wing, it is expected that not many sections are needed to correctly predict the deflections and structural twist. Using 5 chordwise panels, the difference in tip deflection compared to 40 panels is in the order of 1 mm and the structural twist differs under 0.01 degrees. As computational effort is important for a preliminary design tool, the results of 5 chordwise panels is deemed to be accurate enough.

5.3. Wake sensitivity

In this section the length and the discretization of the wake is discussed. Both of these variables need to be set before running the FSI model and this does not change throughout the loop.

5.3.1. Wake length

The length of the modelled wake influences the resulting lift force predicted by the model. The wake model currently implemented in the FSI model is a frozen vortex wake model, meaning that the wake only convects with the freestream velocity and does not self induce. Adding the wake roll-up in the model will increase the computational effort and this increase is at this moment not deemed worth the increase in accuracy. The length of the wake in the model is determined by three factors: the freestream velocity, the time step size and the number of time steps. To determine the length of the wake, the high loading case of Section 5.2.1 is taken ($\alpha = 5^o, U_\infty = 100\frac{m}{s}$ and a roll angle of 0). Figure 5.9 shows the lift force for different wake lengths and the difference the increase in wake length induces per meter. When the wake is modelled for 60 meter, increasing the wake length with 1 meter, the total lift force changes with less than 0.01 %. As the change in lift force is less than 0.01 %, a wake length of 60 meter is assumed to capture the influence of the wake sufficiently.

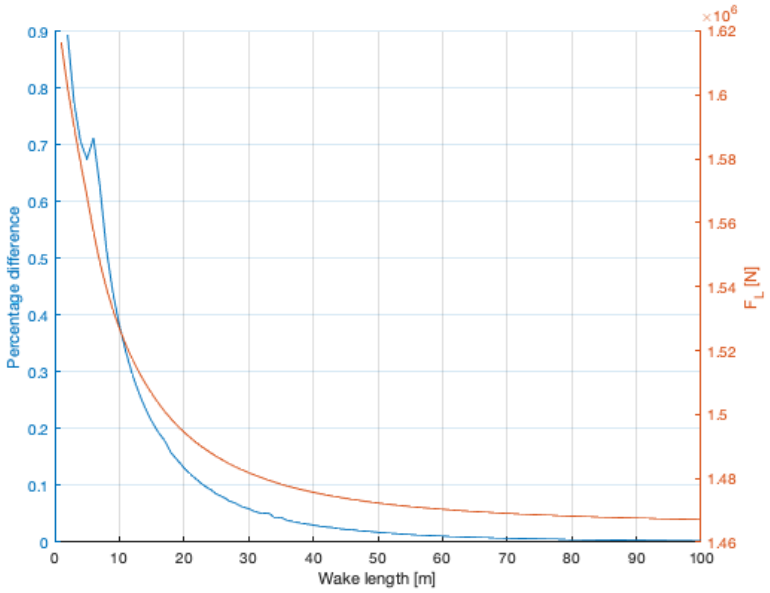
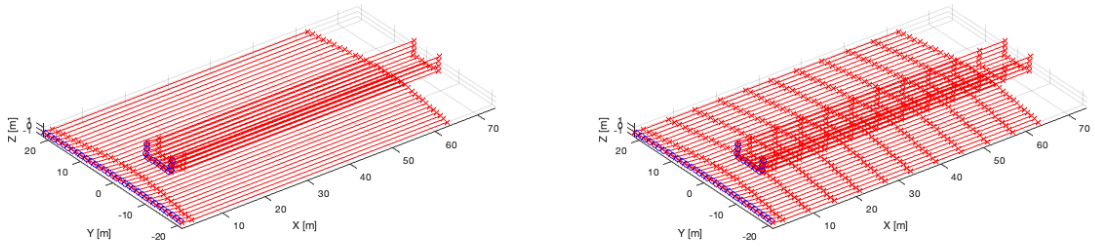


Figure 5.9: Lift for different wake lengths

5.3.2. Wake discretization

As the length of the wake is chosen to be 60 meter, the input for the discretization of the wake needs to be determined. Figure 5.10 shows two examples of wake discretization. Figure 5.10a is the case where the entire wake consists of a single vortex element. The first wake points is shed a quarter chord behind the trailing edge and the second set of wake points are located 60 meters downstream. Figure 5.10b shows the case where the wake is discretized in 10 vortex filaments. The effect of the discretization can be that certain vortex filaments will be excluded from the aerodynamic calculation, as discussed in Section 3.2.5.



(a) Wake with 1 vortex element

(b) Wake with 10 vortex elements

Figure 5.10: Discretization of the wake using 1 or 10 elements

When the number of time steps and the time step size is varied, the result is Figure 5.11. The product of the the time step size, number of time steps and freestream velocity is kept constant at 60 meter. When the wake is discretized in more vortex filaments, the predicted lift force increases, due to the exclusion of certain vortex filaments by the rear part of the aircraft. The difference in lift force between a wake comprised of a single filament and a wake of 500 filaments is slightly over 0.5 %. As the computational effort increases with a more discretized wake, a wake with a single filament is chosen.

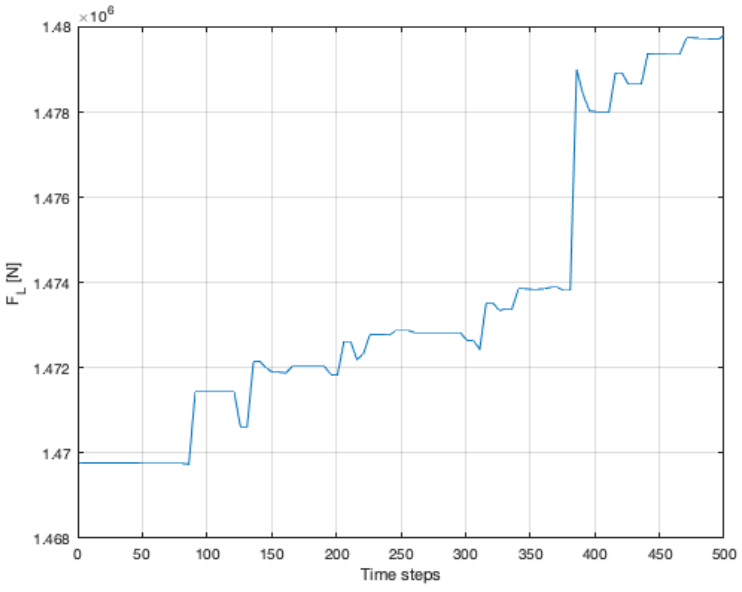


Figure 5.11: Lift force for varying wake discretizations

5.4. Computational time

The time needed for the FSI model to run is one of the important parameters for a preliminary design tool. When running a cycle, following states will generally be very similar. For that reason, the calculated aerodynamic force and deformation are used as the starting values for the following state. Therefore the solution will converge faster and reduce the time needed for the simulation. Table 5.1 shows the time needed for the different parts of the FSI cycle. The times for the lifting line, chordwise force distribution and structures is per iteration of the FSI cycle. Generally, the first state needs about 5 iterations to converge. Due to the previously explained method of using this output as the input for the following state, further states only need 1 or 2 iterations to converge. The times of Table 5.1 are based on the time needed by a MacBook Pro 2018 with a 2,2 GHz 6-Core Intel Core i7.

Pre-FSI run	Lifting line	Chordwise force distribution	Structures
±27 s	±1 s	±0.1 s	±1 s

Table 5.1: Computational times of the different parts of the FSI cycle

The time needed for the pre-FSI run is the most time consuming and largely due to setting up the transformation matrices that are used to couple the aerodynamic and structural meshes. Furthermore it extracts the lifting line geometric input data from the finite element grid. Not much can be done about this time needed as the set-up of the matrices simply takes computational effort.

Next to the times indicated in Table 5.1, the MSES files need to be generated. The time needed for this depends severely depends on the amount of Reynolds numbers evaluated and the range of angles of attack. Nonetheless, it will take anywhere up to 1 or 2 hours. This number is based on the fact that for this research 141 angles of attack at 5 different Reynolds numbers for 2 different airfoils were evaluated. The time needed is almost fully due to the time needed by MSES to evaluate the airfoil and therefore not much could be done to reduce this time without diving into the source code of MSES.

5.5. Wing adaptation

The new model will be tested against the reference model of Eijkelhof [19] and the high fidelity results of Pynaert [49]. As the simulation framework is being developed since these results have been obtained, the FE grid of the current wing in the framework is different from the FE grid used when Pynaert obtained his results. The angle of attack is different for the two wings. As the input for the lifting line is

extracted from the FE grid, the used grid and the resulting angle of attack need to be similar to be able to compare the results. The difference in angle of attack is shown in Figure 5.12. The washout of the wing can clearly be seen and there is an offset between the two grids of about 5 degrees. To compare results between the different models, a correction is applied to the new grid of 5 degrees. This ensures comparable aerodynamic results from the lifting line. The pressure distributions used for the chordwise force distribution are taken at the corrected angles of attack as well.
Although this ensures proper aerodynamic results, when the deformations are calculated for the different models, this is done with the FE grid of the respective model. Therefore, the deformations calculated can be different from one another even though the same FE method is used.

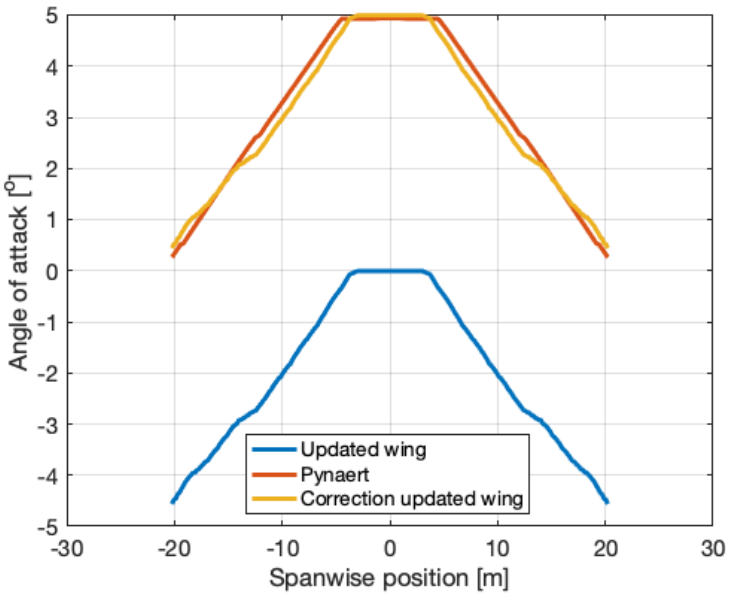


Figure 5.12: Spanwise twist angle of the two different grids

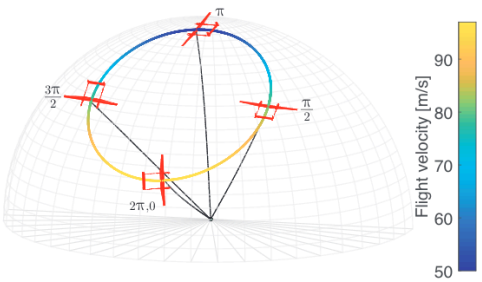
6

Results

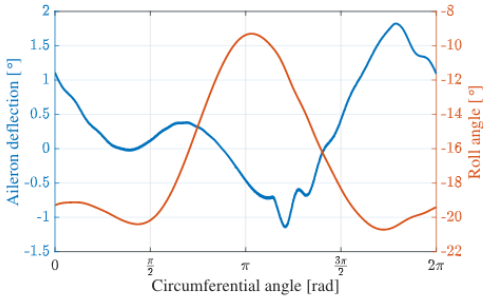
This chapter is divided into multiple sections. First of all the results for the aerodynamic model are discussed in Section 6.1.

6.1. Aerodynamic model

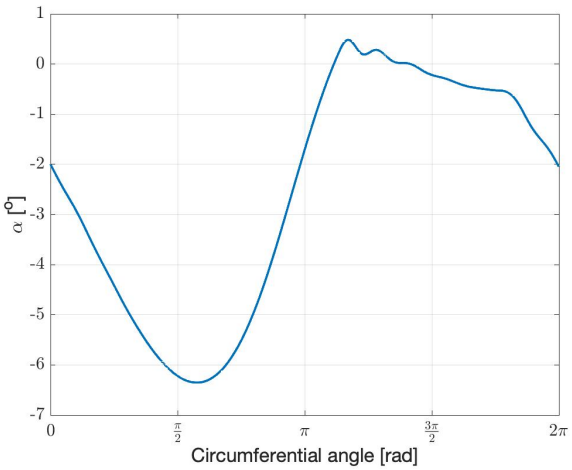
The aerodynamic model is tested against the current model of Eijkelhof [18]. The flight path of the aircraft is circular and the position on the path can be described with the circumferential angle. The flight path, flight velocities, roll angles and angles of attack are displayed in Figure 6.1 and these are the inputs for the model.



(a) The flight path and velocities



(b) Roll angles over flight path



(c) Angles of attack over flight path

Figure 6.1: Input data for the reference model of Eijkelhof [18]

Over the flight path, 4716 data points are available for these parameters. These are inserted in the lifting line model and the resulting lift force for the reference model of Eijkelhof, the CFD and FSI results of Pynaert and the result of the developed lifting line. Pynaert has taken circumferential angles of 0, $\frac{\pi}{2}$, π and $\frac{3\pi}{2}$ and tested the reference model of Eijkelhof using CFD. Pynaert coupled his CFD result to MSC Nastran for a full FSI model. One thing that should be kept in mind for the result of Pynaert is that Pynaert did not use the exact input data for the angle of attack from Eijkelhof, but instead calculated the angle of attack using the lift force given by the reference model and the inviscid $C_L - \alpha$ curve. The angles of attack that Pynaert used are therefore not exactly similar to the ones used by Eijkelhof and used for the lifting line.

The lift on the aircraft for the different models is presented in Figure 6.2. The written lifting line code predicts lower lift for all circumferential angles compared to the reference model. However, the lifting line still slightly overpredicts lift compared to the CFD and FSI results. For increasing lift, the lifting line result deviates more from the results from Pynaert. At the lower velocity points in the flight path, the lifting line results are much closer to the CFD and FSI results.

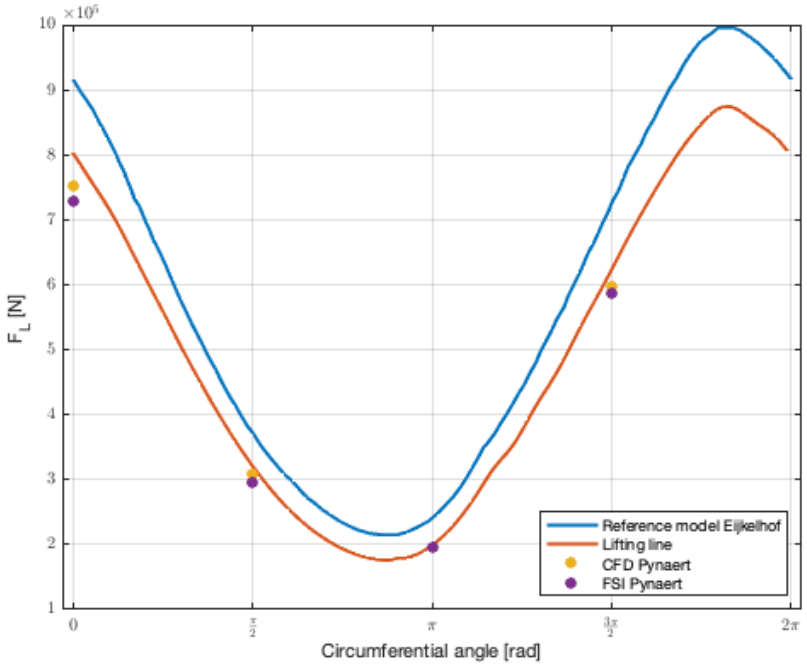


Figure 6.2: The lift on the aircraft for the 4 different models

Multiple reasons for the slight overprediction of the lift compared to the results from Pynaert can be identified:

- The fuselages are not in the lifting line model. Over the largest part of the flight path, the angle of attack is negative. Vimalakanthan et al. [60] state that the contribution of the fuselages is 5% of the total lift for the AP3 model. This estimation is at higher angles of attack, nevertheless is does show that the fuselages influence the resulting lift. Vimalakanthan et al. investigated the AP-3 model of Ampyx Power and as the aircraft used for this research is based on an upscaled version of this model, this estimation is reasonable.
- The interaction effect at the root of the wing and fuselages is not accounted for.
- Since the fuselages are not modelled, the lifting line slightly overpredicts the area of the wings. Where the fuselages are supposed to be located, the lifting line assumes there is more lifting surface and therefore overpredicts the wing area.
- The angles of rotation for the aircraft are not accounted for. These roll rates add a velocity on the different control points and therefore influence the angle of attack and resulting lift.

- The ailerons and their deflections are not included. These ailerons will change the 2D characteristics over certain parts of the front wing. As can be seen in Figure 6.1b, the aileron deflection over this cycle is small. Nevertheless, this will influence the resulting lift.

6.2. FSI model

Using the resulting forces from the aerodynamic model, the structural model can be used to determine the deformation of the wing. When an aerodynamic force distribution over the wing is inserted, the structural model determines the deformation of each of the finite element grid points. Figure 6.3 shows the deflection of the wing tip at a circumferential angle of zero, which is 0.35 m.

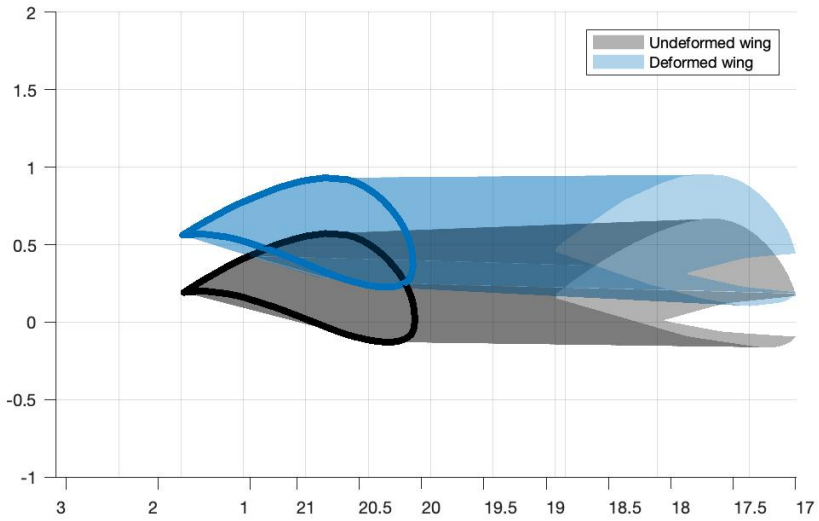
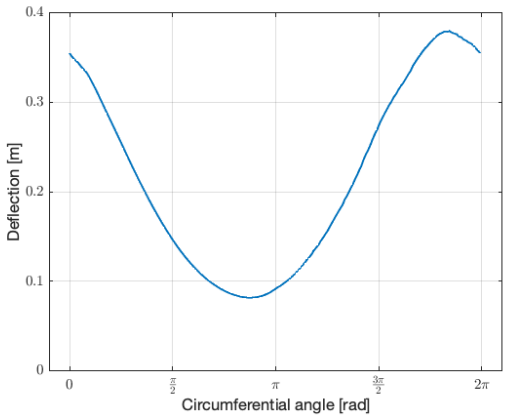
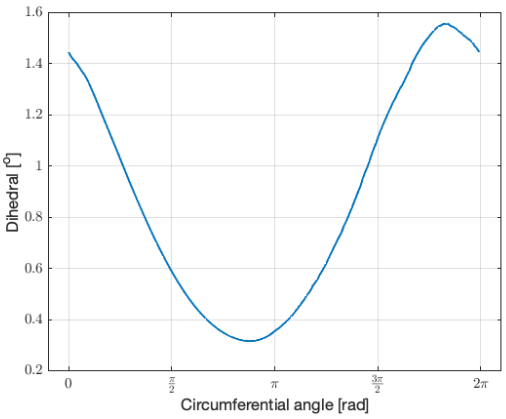


Figure 6.3: Deformation of the wing tip at high loading (circumferential angle of 0)

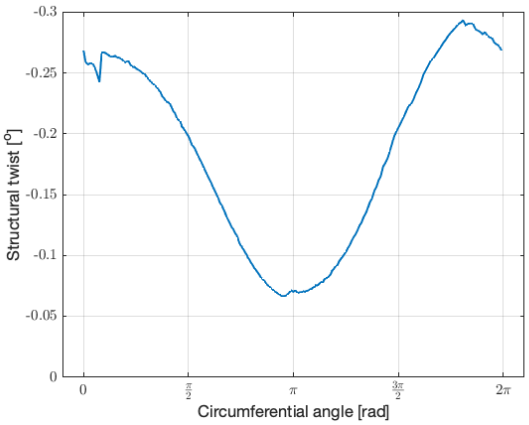
Figure 6.4 show the results for the deflection, dihedral and structural twist of the wing tip throughout the full cycle. The deflections of the tip are for all angles 0.02-0.15 m lower than the results of Eijkelhof [18]. As the aerodynamic forces predicted by Eijkelhof are higher throughout the entire cycle, it is expected that the deformation will also be larger. Wijnja et al. [63] analysed the Makani M600 aircraft and measured tip deflections. Although this aircraft is different from the used model, the deflections of the tip are of similar order.



(a) Tip deflection over the cycle



(b) Tip dihedral over the cycle



(c) Structural twist at the tip over the cycle

Figure 6.4: Structural deformations of the tip over the cycle

Next to the variation of the tip, the spanwise deflections and dihedral are computed. Figure 6.5 shows these for a circumferential angle of 0 degrees, the first point of Figure 6.2. As the wing is fixed in the middle, the deflection is zero at a spanwise position of zero. The dihedral does not go to zero, as there is no control point in the middle of the wing. If there would be a control point there, the dihedral would also be zero at this point.

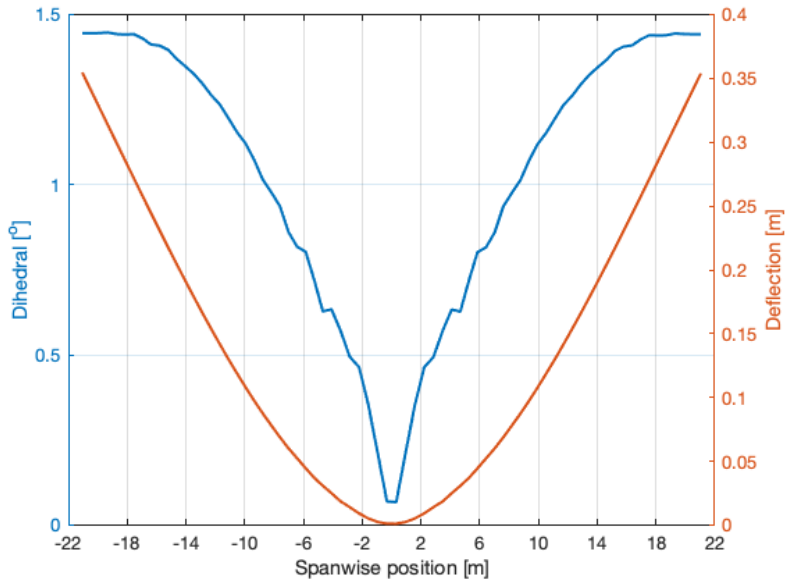


Figure 6.5: Spanwise deflection and dihedral for a circumferential angle of 0

Figure 6.4 and Figure 6.5 show that the deformations over the cycle are small. Considering the fact that the aircraft analyzed is over 40 meter in total span, tip deflections of centimeters is not much. The influence of the amount of structural twist and dihedral is expected to not be great as well. Due to the washout of the wing, the lift at the tips is low, if not negative, see Figure 5.2b. At the tip the dihedral is largest, but due to this effect, the overall lift will not be influenced much. The same reasoning holds for the structural twist. Due to the washout, the structural twist has less influence on the overall lift when approaching the tip. Looking at the spanwise variation of the structural twist and dihedral of Figure 6.6 at a loading case of the cycle, this can be seen clearly. The rough spanwise variation of the twist is due to the interpolation of the thin-plate spline method. The structural twist and dihedral angles close to the middle are close to zero and therefore the influence on the lift produced by this part of the wing is small. When approaching the tips, these angles become larger, but the washout of the wing limits the effect on the lift.

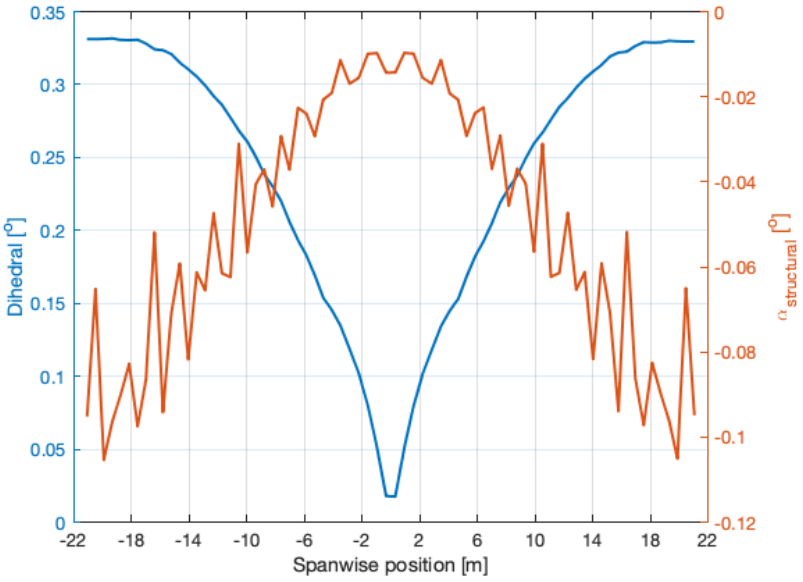


Figure 6.6: Spanwise structural twist and dihedral for the lower loading case of the evaluated power cycle.

The deflection, dihedral and structural twist are all used as input for the aerodynamic model. If the FSI model is run for the entire cycle, the result is Figure 6.7. The results from the FSI model are slightly lower than the result for the lifting line. Due to the high stiffness of the wing, the results for the FSI model are close to the lifting line results.

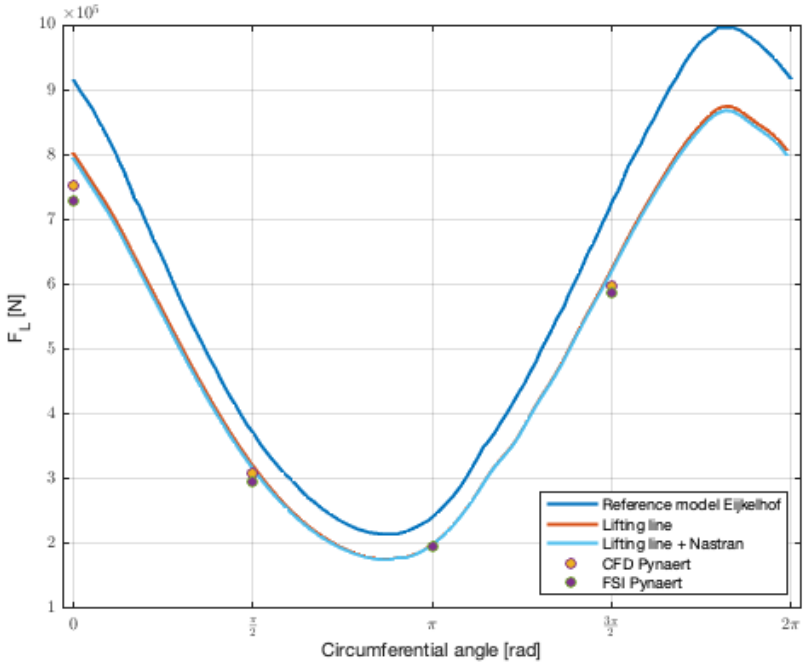


Figure 6.7: The written FSI model compared to the reference models

The overall result shows that the new model is closer to the results of Pynaert over the entire cycle. Table 6.1 shows the difference in percentage of the new and reference model compared to the FSI results of Pynaert. The new model shows to be a big improvement from the reference model over the entire cycle. The general trend is that for higher loading, the new model deviates more from the FSI results than for lower loading. This seems to be caused by the the difference in CFD and FSI results from Pynaert. The data from Pynaert shows a larger difference between FSI results and aerodynamic results. In the new model, the difference for this aircraft is very small, where this is more pronounced in Pynaerts results.

	0	$\frac{\pi}{2}$	π	$\frac{3\pi}{2}$
Reference model	25.5 %	25.0 %	26.3 %	24.3 %
New model	9.0 %	6.9 %	1.1 %	6.3 %

Table 6.1: Difference of the old reference model and the new model compared to the high fidelity model of Pynaert at the different circumferential angles.

The change in lift when including the structural model is more pronounced for the results of Pynaert. Figure 6.8 shows the tip deflection over the cycle for the different models. The tip deflections of the used model are similar for the four data points of Pynaert and therefore it can be concluded that the bigger deviation between aerodynamic and FSI models does not stem from the deflections itself.

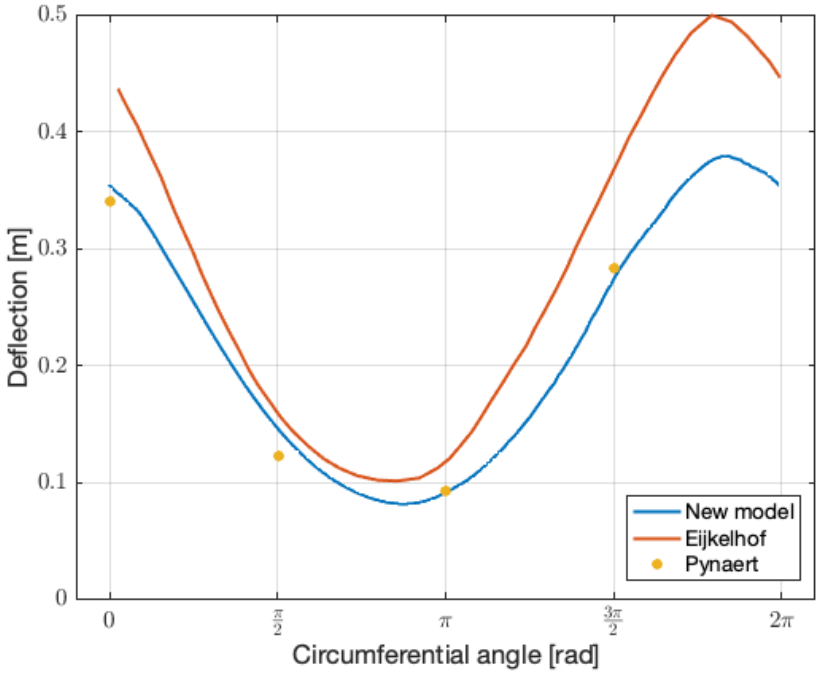


Figure 6.8: The deflections for the different models

Secondly, the structural twist can be looked at to see if this deviates between the new model and the results from Pynaert. Figure 6.9 shows that there is a difference between the structural twist results of Pynaert and the new model. As the structural twist directly influences the angle of attack and therefore the generated lift, this explains the bigger difference between the FSI and aerodynamic results of Pynaert. The structural twist follows the general trend of the loading and at high loading angles, the discrepancy between the two models increases, therefore resulting in higher deviations between the FSI and CFD model of Pynaert at higher loading.

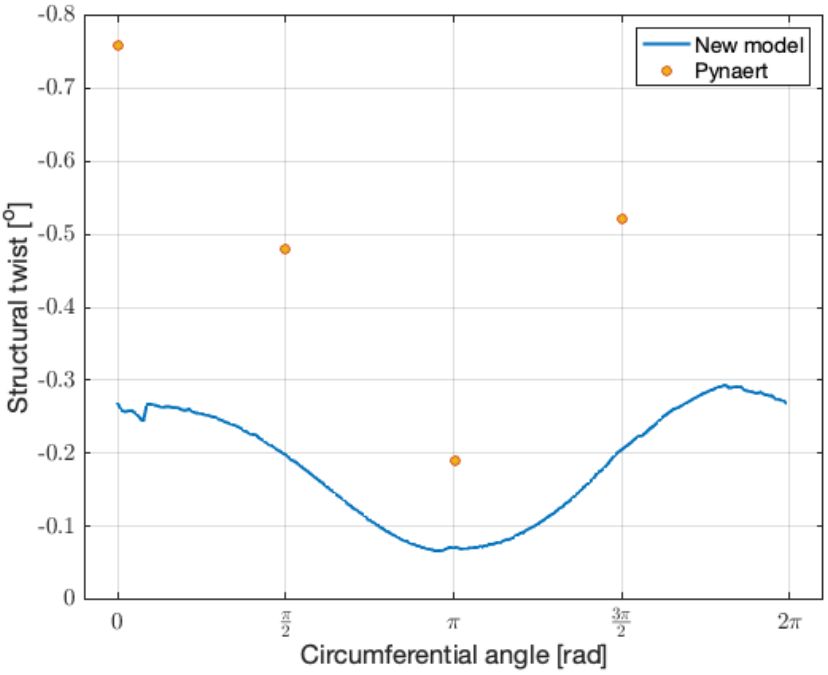


Figure 6.9: The twist at the tip due to deformation of Pynaert and the new model

Overall the new model shows to be an improvement compared to the reference model of Eijkelhof. Next to the improved results for the evaluated power cycle, the new model is not limited to the linear part of the lift curve as was the case in the reference model. MSES is able to run from Matlab, eliminating the need for knowledge regarding MSES. Generally, CFD results take in the order of hours, whereas the new model is able to evaluate flight states in the order of seconds. Although the results still deviate from high fidelity methods, the aerodynamic behaviour can be much better predicted compared to the reference model, without the limitations the reference model experienced. However, there remain some uncertainties regarding the new model. The difference in structural twist could be the result of the different FE grids used, but also a difference in chordwise force distribution between the method used and CFD.

7

Conclusion and recommendations

This chapter aims to draw a conclusion from the previously presented results, answer the research questions and give recommendations regarding future work. Next to that an outlook will be given on further use of the model.

7.1. Conclusion

Airborne wind energy aims to solve existing issues experienced by conventional wind turbines, such as saturation of wind farms, material costs and fluctuating wind speeds. As airborne wind energy keeps on developing, the need for a public reference system to compare research of universities as well as companies has grown. The reference model of Eijkelhof et al. [19] provided such a reference system. Within this reference model, there was room for improvement and this research has focused on the FSI model of the framework. The aerodynamic model used by Eijkelhof is not able to capture viscous effects, analyse multi-element airfoils or easily run different geometries. With the aim to incorporate these aspects into an FSI model, the research objective of this research was as follows:

> "Provide an accurate and computationally efficient FSI model for the simulation framework of multi-megawatt airborne wind energy systems that can be used for further development within this framework."

First three subquestions needed to be answered in order to arrive at an FSI model:

1. "How would an aerodynamic model look like that is able to evaluate viscosity, multi-element airfoils in the order of seconds and is suited for the simulation framework of large-scale, fixed-wing AWESs?"
2. "How would a structural model look like that is capable of evaluating large-scale, fixed-wing AWESs in the order of seconds?"
3. "How would the new aerodynamic and structural model be coupled?"

The best way to incorporate all of the aspects into an aerodynamic model that is easy to adjust was found to program a solver in Matlab. The 2D viscous airfoil program MSES was coupled to a lifting line within Matlab. Next to the lifting line, a method was implemented to obtain the chordwise force distribution on the upper and lower surface of the wing from the 2D pressure distributions. Although this method is more time intensive than taking an existing solver, it is the best option for further research, which will be elaborated upon in the recommendations.
MSC Nastran was used to obtain the deformations of the wing as this method is used by Eijkelhof for research regarding the structural optimization of the wing. Therefore, this method was already implemented for the current model and reverting to a simpler structural model would not be preferable for future use of the FSI model.
The aerodynamic and structural model have been coupled using the inverse distance weighting and a thin-plate spline method. These methods have been proven to work in the old reference model and as the aerodynamic model was adjusted in such a way that the output of the new aerodynamic model

resembles the same output as the old reference model, this coupling method could be used for the new model without major adjustments.
The resulting FSI solver is used to evaluate the power cycle previously run by Eijkelhof, resulting in Figure 7.1. The result of the cycle shows that the new model is better capable of calculating the aerodynamic force compared to the reference model, as can be seen in Table 7.1. Furthermore, the new model is not limited to low angles of attack and capable of evaluating multi-element airfoils. Although the results are closer to the high fidelity results, there remain some uncertainties about the accuracy of the new model. The difference in structural twist could be due to the different FE grids used or a different chordwise force distribution obtained through the used method compared to CFD.

	0	$\frac{\pi}{2}$	π	$\frac{3\pi}{2}$
Reference model	25.5 %	25.0 %	26.3 %	24.3 %
New model	9.0 %	6.9 %	1.1 %	6.3 %

Table 7.1: Difference of the old reference model and the new model compared to the high fidelity model of Pynaert at the different circumferential angles.

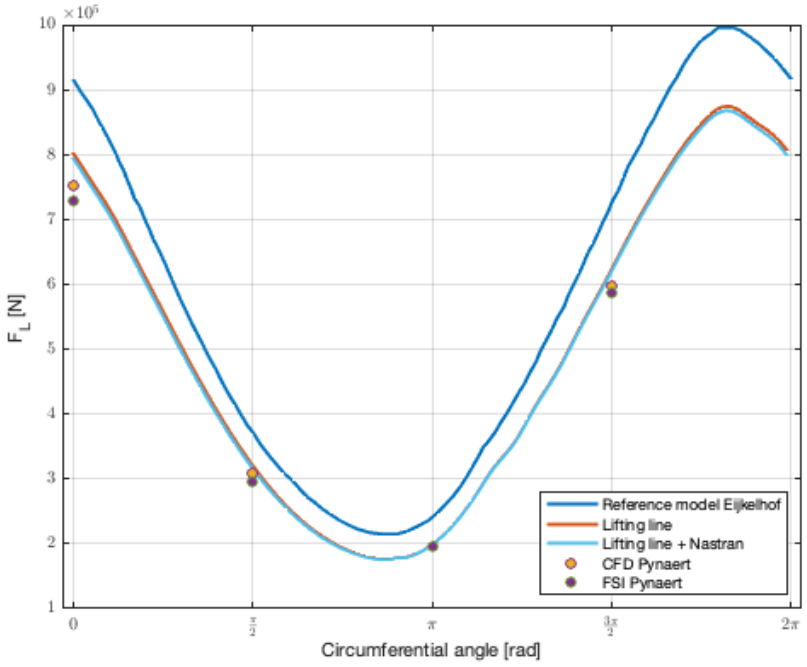


Figure 7.1: The written FSI model compared to the reference models

The last research question to be answered was if high fidelity modelling is needed at this stage:

> "Is high-fidelity aerodynamic modelling needed for large-scale fixed-wing AWESs and at what stage would this be done?"

As Table 7.1 shows, the reference model deviates about 25 % over the entire cycle from the high fidelity results. When using the model in a simulation framework to estimate the power production of AWESs, an overprediction in force of this magnitude will result in a large overprediction of the power production. The new model shows a significant decrease in this deviation and therefore, when implemented, would predict the power production more accurately. When it is used as a preliminary design tool, low fidelity modelling is able to give a good indication of the force. However, as high fidelity modelling is inherently more accurate, when moving from preliminary design to more detailed design, one should switch to higher fidelity models.

Next to the accuracy of the model, computational efficiency is of importance as it will be used mainly as a preliminary design tool. One FSI iteration takes about 2 to 2.5 seconds, comprising of slightly over one second for the lifting line and the chordwise force distribution and slightly over 1 second for the structure side. A pre-FSI run needs to be done to compile the transformation matrices to transfer forces and displacements between the two meshes and extracts the wing geometry from the finite element mesh that takes roughly 27 seconds. Compiling lift polars and pressure distributions for the airfoils using MSES needs to be done once before the FSI is initiated. Depending on the range of angles of attack and Reynolds numbers, this can take up to 1 or 2 hours. As the output of one state is used as the input for the following state, generally only one or two iterations are needed per state for the solution to converge.

Overall, the FSI model shows an improved accuracy for all input data used. As it incorporates MSES within Matlab and includes viscosity and multi-element airfoils, the new model is able to contribute to the ongoing research of AWESs. Improving the modelling accuracy of AWESs is one of the needed improvements in order to compete with conventional wind energy methods and that is where this research is able to contribute. It can be used as a new reference model in the simulation framework and can be easily build upon due to its set-up in Matlab. Everybody with a background in Matlab and the user guide should be able to adapt the model for a specific use case.

7.2. Recommendations

Although the new model shows promising results over the evaluated power cycle at a much lower computational cost than CFD, there are still a lot of improvements that can be implemented. Some improvements can be made to increase accuracy of the model, others to remove limitations or reduce the computational time.

First of all, time could be invested to work out the problems of MSES. As discussed in Section 5.1, MSES was unable to resolve the pressure distributions for the MRevE airfoil when starting at an angle of attack of 0. Besides, the mpolar function was unable to resolve the lift polars for certain Reynolds numbers, resulting in a lower Reynolds number polar used for all the airfoils and introducing an underprediction in lift. As XFLR5 was used to validate that the airfoil could be resolved at 0 angle of attack, there is some intrinsic error in MSES that causes this to fail. For this research only two types of airfoils were used and therefore eliminating the non converged files was fairly easy. However, when research is going to be done for multiple different airfoil runs, this task becomes very time-intensive. Investigating the source code of MSES could therefore benefit the accuracy of the model and eliminating any redundant processes taking place could also speed up the file generation process. As adjusting the MSES source code was outside the scope of this research, no time has been invested to adapt MSES.

The aerodynamic model itself shows promising results, although multiple improvements can still be made:

- The fuselages are not in the lifting line model. Over the largest part of the flight path, the angle of attack is negative. Vimalakanthan et al. [60] state that the contribution of the fuselages is 5% of the total lift for the AP3 model. This estimation is at higher angles of attack, nevertheless it does show that the fuselages influence the resulting lift. Vimalakanthan et al. investigated the AP-3 model of Ampyx Power and as the aircraft used for this research is based on an upscaled AP-3 model, this estimation is reasonable.
- The interaction effect at the root of the wing and fuselages is not accounted for.
- Since the fuselages are not modelled, the lifting line slightly overpredicts the area of the wings. Where the fuselages are supposed to be located, the lifting line assumes there is more lifting surface and therefore overpredicts the wing area.
- The angles of rotation for the aircraft are not accounted for. These roll rates add a velocity on the different control points and therefore influence the angle of attack and resulting lift.
- The ailerons and their deflections are not included. These ailerons will change the 2D characteristics over certain parts of the front wing. As can be seen in Figure 6.1b, the aileron deflection over this cycle is small. Nevertheless, this will influence the resulting forces and moments.

- No real sideslip correction is implemented in the model at this point. At this moment the inputs for the aerodynamic model in the framework does not include the sideslip angle, but if this changes, a correction is needed for the sideslip.

The chordwise force distribution method that is used in this research shows reasonable results, however, a proper validation study can be done with CFD data. As MSC Nastran is used for the structural side of the FSI model, a force distribution is needed on the upper as well as the lower surface of the wing. Other methods such as a vortex lattice method are unable to resolve this for the two different surfaces, as they solve singularity strengths. This method shows however the low computational cost and therefore it has benefits for the preliminary design phase. A 2D CFD validation study could show the actual accuracy of this method compared to higher fidelity modelling.

The frozen wake model implemented currently is computationally efficient, but for more detailed wake modelling and interaction it should be adapted. If the need is there to evaluate wind farms of AWESs, the wake model can be adapted to a free vortex wake and the interaction between wakes of different systems can be accounted for. This will increase the computational cost and if a single system is evaluated, this increased computational cost is not preferable. For N wake nodes, a frozen wake models computational effort is in the order of N, but when a free wake model is introduced, the effort is in the order of N^2.

The higher purpose of the FSI model is to more accurately predict the power production of the airborne wind energy system at low computational cost. At the moment it is not able to do so, as it is not implemented within the simulation framework of Eijkelhof. To be able to reach this goal, the FSI model should be implemented in the Simulink environment of the framework.

References

[1] Regine Ahrem, Armin Beckert, and Holger Wendland. “A new multivariate interpolation method for large-scale spatial coupling problems in aeroelasticity”. In: (2001). URL: https://www.researchgate.net/publication/228731651.

[2] Uwe Ahrens, Moritz Diehl, and Roland Schmehl. *Airborne Wind Energy*. Springer Berlin Heidelberg, 2013. ISBN: 978-3-642-39964-0. DOI: 10.1007/978-3-642-39965-7.

[3] *AMI Aero*. URL: https://amiaerollc.com/Software.html.

[4] John D. Anderson. *Fundamentals of aerodynamics*. Mcgraw-Hill, 1991, p. 772. ISBN: 0070016798.

[5] John D. Anderson. *Modern Compressible Flow With Historical Perspective*. Third. McGraw-Hill, 2002. ISBN: 978-0072424430.

[6] J Blazek. *COMPUTATIONAL FLUID DYNAMICS: PRINCIPLES AND APPLICATIONS*. Elsevier Science Ltd, 2001. ISBN: 0080430090.

[7] National Transportation Safety Board. *Aircraft Accident Brief: Pilot/Race 177, The Galloping Ghost*. Createspace Independent Publishing Platform, 2015. ISBN: 1514672987.

[8] A. de Boer, A. H. van Zuijlen, and H. Bijl. “Comparison of conservative and consistent approaches for the coupling of non-matching meshes”. In: *Computer Methods in Applied Mechanics and Engineering* 197.49-50 (Sept. 2008), pp. 4284–4297. ISSN: 00457825. DOI: 10.1016/j.cma.2008.05.001.

[9] Ashwin A. Candade, Maximilian Ranneberg, and Roland Schmehl. “Structural Analysis and Optimization of a Tethered Swept Wing for Airborne Wind Energy Generation”. In: *Wind Energy* 23.4 (2020), pp. 1006–1025. DOI: 10.1002/we.2469.

[10] Alfonso del Carre et al. “SHARPy: A dynamic aeroelastic simulation toolbox for very flexible aircraft and wind turbines”. In: *Journal of Open Source Software* 4.44 (Dec. 2019), p. 1885. ISSN: 2475-9066. DOI: 10.21105/joss.01885.

[11] Rick Damiani. *Makani Final Thoughts*. Tech. rep. URL: https://kitefast.readthedocs.io/en/latest/source/user/kitefast/index.html.

[12] Aaron Dettmann. *Loosely coupled, modular framework for linear static aeroelastic analyses*. 2019. URL: http://www.diva-portal.org/smash/record.jsf?pid=diva2%3A1360722&dswid=6909.

[13] Mark Drela. *A User’s Guide to MSES 3.05*. Tech. rep. 2007.

[14] Mark Drela. *ASWING 5.99 - Technical description*. Tech. rep. 2015.

[15] Mark Drela and Harold Youngren. *AVL User Guide*. 2017. URL: https://web.mit.edu/drela/Public/web/avl/.

[16] Storm Dunker. “Ram-air wing design considerations for airborne wind energy”. In: *Green Energy and Technology* (2013), pp. 517–546. ISSN: 18653529. DOI: 10.1007/978-3-642-39965-7{_}31.

[17] Gaétan Dussart et al. “Flight Dynamic Modelling and Simulation of Large Flexible Aircraft”. In: *Flight Physics - Models, Techniques and Technologies*. InTech, Feb. 2018. DOI: 10.5772/intechopen.71050.

[18] Dylan Eijkelhof. *Design and Optimisation Framework of a Multi-MW Airborne Wind Energy Reference System*.

[19] Dylan Eijkelhof et al. “Reference Design and Simulation Framework of a Multi-Megawatt Airborne Wind Energy System”. In: *Journal of Physics: Conference Series*. Vol. 1618. 3. IOP Publishing Ltd, Sept. 2020. DOI: 10.1088/1742-6596/1618/3/032020.

[20] Urban Fasel et al. “Reduced-order dynamic model of a morphing airborne wind energy aircraft”. In: *AIAA Journal* 57.8 (2019), pp. 3586–3598. ISSN: 00011452. DOI: 10.2514/1.J058019.

[21] Carlos A Felippa, K C Park, and Charbel Farhat. “Partitioned analysis of coupled mechanical systems”. In: *Computer Methods in Applied Mechanics and Engineering* 190.24-25 (2001), pp. 3247–3270. URL: https://www.sciencedirect.com/science/article/abs/pii/S0045782500003911.

[22] Daniel Filkovic. *Apame*. 2008. DOI: 10.1177/000271624423100120.

[23] M. M. Freestone. *VGK method for two-dimensional aerofoil sections. Part 1: Principles and results.* 2014. ISBN: 978 0 85679 992 1.

[24] Hugo Gagnon and David W. Zingg. “Aerodynamic optimization trade study of a box-wing aircraft configuration”. In: *56th AIAA/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference* January (2015), pp. 1–16. DOI: 10.2514/6.2015-0695.

[25] Hasier Goitia and Raúl Llamas. “Nonlinear vortex lattice method for stall prediction”. In: *MATEC Web of Conferences* 304 (2019), p. 02006. DOI: 10.1051/matecconf/201930402006.

[26] Sang Truong Ha et al. “A comparative study between partitioned and monolithic methods for the problems with 3D fluid-structure interaction of blood vessels”. In: *Journal of Mechanical Science and Technology* 31.1 (Jan. 2017), pp. 281–287. ISSN: 1738494X. DOI: 10.1007/s12206-016-1230-2.

[27] Charles Hirsch. *Numerical Computation of Internal and External Flows Fundamentals of Computational Fluid Dynamics*. Second. Elsevier Ltd., 2007. ISBN: 978-0-7506-6594-0.

[28] Antony Jameson. “Aerodynamics”. In: *Encyclopedia of Computational Mechanics*. Ed. by Erwin Stein, René De Borst, and Thomas Hughes. John Wiley, 2004. Chap. 11. ISBN: 0470846992.

[29] John P. Jasa, John T. Hwang, and Joaquim R.R.A. Martins. “Open-source coupled aerostructural optimization using Python”. In: *Structural and Multidisciplinary Optimization* 57.4 (Apr. 2018), pp. 1815–1827. ISSN: 16151488. DOI: 10.1007/s00158-018-1912-8.

[30] Ramji Kamakoti and Wei Shyy. “Fluid-structure interaction for aeroelastic applications”. In: *Progress in Aerospace Sciences* 40.8 (Nov. 2004), pp. 535–558. ISSN: 03760421. DOI: 10.1016/j.paerosci.2005.01.001.

[31] Joseph Katz and Allen Plotkin. *Low-Speed Aerodynamics*. Second Edi. Cambridge University Pr., 2001. ISBN: 978-0521665520.

[32] Mrinal Kaushik. *Theoretical and Experimental Aerodynamics*. Springer Nature Singapore Pte Ltd., 2019. ISBN: 9789811316777. DOI: https://doi.org/10.1007/978-981-13-1678-4.

[33] Graeme J. Kennedy and Joaquim R.R.A. Martins. “A parallel finite-element framework for large-scale gradient-based design optimization of high-performance structures”. In: *Finite Elements in Analysis and Design* 87 (Sept. 2014), pp. 56–73. ISSN: 0168874X. DOI: 10.1016/j.finel.2014.04.011.

[34] Gaetan K.W. Kenway, Graeme J. Kennedy, and Joaquim R.R.A. Martins. “A CAD-free approach to high-fidelity aerostructural optimization”. In: *13th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference 2010*. 2010. ISBN: 9781600869549. DOI: 10.2514/6.2010-9231.

[35] *KiteFAST User Guide*. URL: https://kitefast.readthedocs.io/en/kitefast/source/user/kitefast/index.html.

[36] Rachel Leuthold. *Multiple-Wake Vortex Lattice Method for Membrane-Wing Kites LM MW*. 2015.

[37] Giovanni Licitra et al. “Viability Assessment of a Rigid Wing Airborne Wind Energy Pumping System”. In: (2017). DOI: 10.1109/PC.2017.7976256.

[38] Don Logan. *Lockheed F-117 Night Hawks: A Stealth Fighter Roll Call*. Schiffer Publishing, Ltd., 2009. ISBN: 0764332422.

[39] Miles L. Loyd. “CROSSWIND KITE POWER.” In: *Journal of energy* 4.3 (1980), pp. 106–111. ISSN: 01460412. DOI: 10.2514/3.48021.

[40] E. C. Malz et al. “A reference model for airborne wind energy systems for optimization and control”. In: *Renewable Energy* 140 (Sept. 2019), pp. 1004–1011. ISSN: 18790682. DOI: 10.1016/j.renene.2019.03.111.

[41] David Marten. *QBlade: A Modern Tool for the Aeroelastic Simulation of Wind Turbines vorgelegt von*. 2019.

[42] B Maskew. *A Computer Program for Calculating the Non-linear Aerodynamic Characteristics of Arbitrary Configurations USER'S MANUAL*. 1982. URL: https://ntrs.nasa.gov/search.jsp?R=19850006493.

[43] Thomas Melin. *USER's GUIDE: Tornado*. Tech. rep. 2001. URL: https://www.researchgate.net/publication/320924935_Tornado_10_USER's_GUIDE_Reference_manual.

[44] Mario Natella and Roeland De Breuker. "The effects of a full-aircraft aerodynamic model on the design of a tailored composite wing". In: *CEAS Aeronautical Journal* 10.4 (Dec. 2019), pp. 995–1014. ISSN: 18695590. DOI: 10.1007/s13272-019-00366-5.

[45] Francisco Palacios et al. "Stanford University Unstructured (SU2): An open-source integrated computational environment for multi-physics simulation and design". In: *51st AIAA Aerospace Sciences Meeting including the New Horizons Forum and Aerospace Exposition 2013*. 2013. ISBN: 9781624101816. DOI: 10.2514/6.2013-287.

[46] *Personal communication with D. Eijkelhof*.

[47] W. F. Phillips and D. O. Snyder. "Modern adaptation of Prandtl's classic lifting-line theory". In: *Journal of Aircraft* 37.4 (2000), pp. 662–670. ISSN: 00218669. DOI: 10.2514/2.2649.

[48] Tom Pulliam. "ARC2D-EFFICIENT SOLUTION METHODS FOR THE NAVIER-STOKES EQUATIONS (CRAY VERSION) Solution Methods In Computational Fluid Dynamics". In: (1994). URL: https://www.researchgate.net/publication/24339093.

[49] Niels Pynaert. *Fluid-Structure Interaction Using Computational Fluid Dynamics Of A Multi-Megawatt Airborne Wind Energy Reference System*. May 2021.

[50] Giridhar Ramanujam, Hüseyin Özdemir, and H. W.M. Hoeijmakers. "Improving airfoil drag prediction". In: *Journal of Aircraft*. Vol. 53. 6. American Institute of Aeronautics and Astronautics Inc., 2016, pp. 1844–1852. DOI: 10.2514/1.C033788.

[51] Maximilian Ranneberg. "VIIFlow—A new inverse viscous-inviscid interaction method". In: *AIAA Journal* 57.6 (2019), pp. 2248–2253. ISSN: 00011452. DOI: 10.2514/1.J058268.

[52] Sebastian Rapp et al. "Cascaded Pumping Cycle Control for Rigid Wing Airborne Wind Energy Systems". In: *Journal of Guidance, Control, and Dynamics* 42.11 (Nov. 2019), pp. 2456–2473. ISSN: 1533-3884. DOI: 10.2514/1.g004246.

[53] R.P.J.O.M. van Rooij. "Modification of the boundary layer calculation in RFOIL for improved airfoil stall prediction". In: (1996).

[54] S. M. Salim, K. C. Ong, and S. C. Cheah. "Comparison of RANS, URANS and LES in the Prediction of Airflow and Pollutant Dispersion". In: *The World Congress on Engineering and Computer Science, WCECS*. Newswood Ltd., International Association of Engineers, 2011, p. 1185. ISBN: 9789881925176.

[55] Roland Schmehl. *Airborne Wind Energy*. June 2019. URL: http://awesco.eu/awe-explained/.

[56] Donald Shepard. "A two-dimensional interpolation for irregularly-spaced data function". In: *ACM '68: Proceedings of the 1968 23rd ACM national conference*. 1968, pp. 517–524. DOI: 10.1145/800186.810616.

[57] Alessandro Tasora et al. "Chrono: An Open Source Multi-physics Dynamics Engine". In: *International Conference on High Performance Computing in Science and Engineering*. 2016. DOI: 10.1007/978-3-319-40361-8{_}2. URL: http://www.springer.com/series/7407.

[58] Unfccc. *ADOPTION OF THE PARIS AGREEMENT - Paris Agreement text English*. Tech. rep.

[59] Koen Van Den Kieboom and Ali Elham. "Combined aerostructural wing and high-lift system optimization". In: *17th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*. American Institute of Aeronautics and Astronautics Inc, AIAA, 2016. ISBN: 9781624104398. DOI: 10.2514/6.2016-3510.

[60] K. Vimalakanthan et al. "Aerodynamic analysis of Ampyx's airborne wind energy system". In: *Journal of Physics: Conference Series*. Vol. 1037. 6. Institute of Physics Publishing, June 2018. DOI: 10.1088/1742-6596/1037/6/062008.

[61] N. P.M. Werter and R. De Breuker. “A novel dynamic aeroelastic framework for aeroelastic tailoring and structural optimisation”. In: *Composite Structures* 158 (Dec. 2016), pp. 369–386. ISSN: 02638223. DOI: 10.1016/j.compstruct.2016.09.044.

[62] J Wijnja. *Aero-elastic analysis of a large airborne wind turbine Prediction of divergence, control reversal and effectiveness, and flutter of a tethered wing*. 2013.

[63] Jelle Wijnja et al. “Aeroelastic analysis of a large airborne wind turbine”. In: *Journal of Guidance, Control, and Dynamics* 41.11 (2018), pp. 2374–2385. ISSN: 15333884. DOI: 10.2514/1.G001663.

[64] E. A. Wrigley. “Energy and the english industrial revolution”. In: *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences* 371.1986 (Mar. 2013). ISSN: 1364503X. DOI: 10.1098/rsta.2011.0568.

[65] XFLR5. *Analysis of foils and wings operating at low Reynolds numbers*. Tech. rep. 2009. URL: https://docplayer.net/47798105-Xflr5-analysis-of-foils-and-wings-operating-at-low-reynolds-numbers.html.

A

User guide FSI model

This user guide explains the different aspects of the FSI model developed in this research project. Step by step, the guide will walk through the code and explain the different functions. The goal of this guide is to enable anyone that has this guide and the Matlab code to run the code and adapt it.

A.1. Downloading required files

The files can be downloaded from: https://github.com/JelmerBlom/Aero-model.git. One should generally run the scripts from the folder 'Aero-model' if not indicated otherwise.

A.2. Running MSES from Matlab

To generate the 2D airfoil data, MSES is run with the file 'Generate_MSESfiles.m'. To be able to run this Matlab script, make sure that you are in the 'Run_MSES' folder, otherwise the script is not able to run. This script generates the required data in three stages:

- Parameter sweep over positive angles of attack for the lift polars
- Parameter sweep over negative angles of attack for the lift polars
- Generation of the pressure coefficient distribution files

The first two stages of this process are separated to aid in the convergence of MSES. If the parameter sweep would be immediately be performed from the most negative angle of attack upwards, the first point will most likely not converge. To be able to evaluate these angles of attack, one parameter sweep is done from $\alpha = 0.01$ upwards and one sweep from $\alpha = 0.01$ downwards.

To be able to run the 'Generate_MSESfiles.m' script, one should be in the folder with the Unix executable files. Three folders need to be added within this folder: 'CP_distributions', 'Polar_pos' and 'Polar_neg'. The generated files will automatically be stored in these folders. If other names are preferred, one could change the name of the folders, but need to be aware that it needs to be changed in the script as well in that case.

Furthermore, 7 inputs are required for the script to run. First of all, the airfoil coordinate file needs to be present in the main folder. For the specifics on the format of this file, the reader is referred to [13]. The other inputs for the script are the Mach numbers, Reynolds numbers, array of positive angles of attack, array of negative angles of attack, the number of elements of the airfoil and the airfoil extension. The latter is needed for MSES to run and this extension will be used by MSES for all executable files. The Mach numbers, Reynolds numbers and angles of attack need to be inserted as arrays, e.g. Equation A.1. The first number in the array of Mach numbers should correspond to the first Reynolds number etc.

$$\begin{aligned} \text{Mach} &= [0.1 \quad 0.2 \quad 0.3] \\ \text{Reynolds} &= [1e7 \quad 2e7 \quad 3e7] \end{aligned} \tag{A.1}$$

A.2.1. Angle of attack parameter sweep

Both the positive and negative angle of attack sweep is done in similar fashion. The files are generated in the following manner, where EXT indicates the extension of the airfoil to be evaluated. For general formatting requirements, the reader is referred to [13].

1. The spec.EXT file is created in the format required by MSES. The spec.EXT file is a list of the angles of attack to be evaluated.
2. The mses.EXT is created, which is a file indicating the flow conditions. The commands for creating this file are specified in the file 'actions.txt'.
3. The mses.EXT is edited for the specific Reynolds and Mach number.
4. The 'mpolar' executable is run for the specified angle of attack range.
5. The resulting polars are moved to the folders 'Polar_pos' or 'Polar_neg'.
6. Step 2-5 are repeated for all Reynolds and corresponding Mach numbers.

A.2.2. Pressure distributions

The pressure distributions are extracted by running 'mset' and using the specified-pressure editor of MSES 'medp' to extract the distributions. During the angle of attack parameter sweeps, the surface pressure data is also stored in the 'polarxP' files, however not all sweeps are able to converge as discussed in Section 5.1. Therefore the pressure distributions are generated using the first method from the following steps:

1. The mses.EXT is created.
2. The mdat.EXT is created, using 'actions.txt'.
3. The mses.EXT is edited for the specific Reynolds and Mach number.
4. The initialization program 'mset' is run.
5. 'mses' is run to obtain the 2D data for the specified angle of attack.
6. The pressure distribution is extracted using 'medp', the commands are specified in the file 'findcp.txt'.
7. The resulting file is moved to the folder 'CP_distributions'.
8. Step 3-7 are repeated for every angle of attack.
9. Step 2-8 are repeated for every combination of Reynolds and Mach numbers.

A.3. Pre-FSI run

Although the 2D airfoil data has been generated, more input is needed before the actual aerodynamic and structural model can be run within the FSI iteration loop.

First of all, the parameters from MSC Nastran are imported. The structural properties of the front wing are determined outside the Matlab script and need to be imported to calculate the deformations. For the current wing geometry, these parameters are loaded from the 'PreFSI.mat' file. Using these parameters, both transformation matrices can be determined. The 'Pre_FSI' function uses the FE grid, wing parameters, chord positions, spanwise positions and the number of panels to determine the thin-plate spline transformation matrix and the inverse distance weighting matrix.

The geometry of the aircraft is generated in two stages. First of all, the finite element grid is utilised and all of the info that can not be extracted from this grid is generated afterwards.
The geometry of the front part of the aircraft is extracted from the finite element grid. In that way, the inputs for the lifting line and the structural model are similar. The control points, vortex points, local chord, sweep, twist, dihedral and span are extracted from the grid.

The rest of the geometry is generated using function files and calculated from the parameters. The four function files that generate the rest of the geometry are:

- *Generate_mainwing()*, this generates general parameters of the front wing. The parameters that are extracted from the FE grid are inserted in the resulting struct.
- *Generate_elevator()*, which generates the elevator of the aft part.
- *Generate_vert()*, which generates the vertical wings on the tail.

A.4. Aerodynamic model

The aerodynamic model is one of the major building blocks of the resulting FSI model. In this section, the model will be elaborated on and discussed where certain parameters can be adjusted. This will be done in a systematic manner, such that changes to the model can relatively easy be made. Firstly, the in- and outputs of the function will be discussed. After which each part of the code will be walked through and discussed which parameters can be adjusted in each part.

A.4.1. In- and outputs

The aerodynamic model is written in a function file and therefore it appears as a single line in the FSI script. This is done to keep the overall FSI script as clean as possible.
To be able to run, the aerodynamic model needs 21 inputs to run, in the following order:

1. The flow velocity
2. The angle of attack
3. The sideslip angle
4. The roll angle
5. Number of chordwise panels
6. Dihedral at each panel due to the deformation
7. Twist at each panel due to the chordwise deformation
8. Deflection at the quarter chord line due to the deformation
9. Control points on the front wing
10. Vortex points on the front wing
11. Local chord at the control points
12. Local chord at the vortex points
13. Geometric front wing twist
14. Mainwing parameter struct
15. Elevator parameter struct
16. Vertical wing parameter struct
17. Front wing parameters from FE grid
18. Number of spanwise panels front wing
19. Number of panels elevator
20. Time step size
21. Number of time steps

If the aerodynamic model is run without a structural model or deformations, inputs 6-8 can simply be set to zero.

If the aerodynamic model is run with these inputs, it will return four outputs: The total force vector, the force matrix of the upper surface, the force matrix of the lower surface and the lift force on each panel.

A.4.2. Initialization

In this section of the code, all other parameters are determined or calculated that are needed to find the force distribution on the aircraft besides the inputs of Section A.4.1. These are set out in the following list:

1. The number of panels for the vertical wing is set, which is chosen to be 3.
2. Three different airfoils are specified: one for the front wing, one for the aileron sections and one for the tail. For the front wing this is the previously discussed MRevE airfoil and for the tail it is a NACA0012 airfoil. The different panels on the front and rear part of the aircraft all get assigned an airfoil shape. The code rounds the number of panels that fit the aileron width and starting point from the tip.
3. The air density is set at $\rho = 1.225 \frac{kg}{m^3}$.

4. The 'core_radius' is set at 0.01 m, see Section 3.2.5.
5. The rotation matrix for the aircraft is set up using the angle of attack, roll angle and sideslip angle.
6. The 2D airfoil data is imported from the generated MSES files. Due to the convergence issues of MSES, at the moment only polars at a Reynolds number of 1.2e7 are imported. If the MSES convergence issues would be solved, more polars should be imported to have a more accurate polar at each section.
7. The deflection (if there is any) is used to determine the new control and vortex points on the front wing.
8. The control and vortex points are generated for the aft part of the aircraft.
9. The normal and chordwise unit vectors are determined for all panels.
10. The areas of each panel is calculated.
11. Sweep correction factors are calculated for the front wing. At the moment these correction factors are commented out of the code, as the sweep on the current geometry is so little that the correction factors are very close to 1. If geometries will be evaluated with higher sweep angles, the correction factors can be uncommented and with these correction factors the angle of attack and lift coefficients can be corrected according to Section 3.2.3.
12. The full geometry is rotated using the rotation matrix to arrive at the specified state.
13. The dihedral at each section is determined using the geometrical dihedral and the dihedral due to the deformation of the front wing.
14. The wake is determined for the front as well as the aft part of the aircraft. With the current frozen wake model, this is done before the iteration loop. If at some point a free vortex wake would be desirable, the wake should be generated within the iteration loop, such that the induced velocities at the wake points can be determined. This would increase the accuracy of the solution, although the computational cost increases as well.
15. The AIC matrices are determined using the function *AIC_calc*. The only inputs for this function are the 4 sets of control points, 4 sets of wake points and the cut-off distance. The way these AIC matrices are built, is explained in more detail in Section 3.2.5.

A.4.3. Iteration loop

Now that all the parameters and matrices are known, the circulation distribution can be found using an iteration loop. Before the actual loop is initiated, the convergence criterion and maximum number of iterations can be set. The loop consists of the following steps:

1. The following set of equations is solved for the induced velocities:

$$\begin{aligned} [u] &= [AIC_u][\Gamma] \\ [v] &= [AIC_v][\Gamma] \\ [w] &= [AIC_w][\Gamma] \end{aligned} \tag{A.2}$$

   For the first iteration, an initial circulation distribution is inserted of a $N \times 1$ array of ones. The resulting u,v and w arrays are all of size $N \times 1$.
2. For each panel, the velocity normal to the panel and the tangential velocities are determined using:

$$\begin{aligned} U_{n,i} &= (\mathbf{U}_\infty + [u_i; v_i; w_i]) \cdot \mathbf{n_i} \\ U_{c,i} &= (\mathbf{U}_\infty + [u_i; v_i; w_i]) \cdot \mathbf{c_i} \end{aligned} \tag{A.3}$$

   In Equation 3.15, $\mathbf{n_i}$ is the normal unit vector of panel i and $\mathbf{c_i}$ is the unit vector in chordwise direction.
3. The angle of attack at each panel is determined:

$$\alpha_i = \tan^{-1}\left(\frac{U_{n,i}}{U_{c,i}}\right) + \alpha_{twist} + \alpha_{twist,structural} \tag{A.4}$$

4. Using the Kutta-Joukowski theorem and the viscous lift polars, a new circulation distribution can be found using:
$$\Gamma_i = \frac{1}{2} c_i U_i C_l(\alpha_i) \tag{A.5}$$
5. The newly found circulation distribution of Equation A.5 and the initial circulation distribution are combined to find the circulation distribution for the next iteration:
$$[\Gamma] = 0.95[\Gamma_{old}] + 0.05[\Gamma_{new}] \tag{A.6}$$
6. The convergence is checked to determine if the loop is terminated. This number is defined as max(abs$(\Gamma_{old} - \Gamma_{new})$). As long as this number is larger than 1E-5, steps 1-5 are repeated. When the convergence is smaller than 1E-5, the iteration is stopped. Reaching the maximum number of iteration is the second way the iteration loop can be stopped. This number is set to 10000 as experience has shown that if the solution converges, the number of iteration needed is well below 10000 and therefore 10000 can be seen as a number beyond which the solution will not converge.

The function *Aero_force* determines the total force vector and moments on the aircraft using the converged induced velocities at each panel.

A.4.4. Chordwise force distribution

The spanwise force distribution on the front wing has been determined in the iteration loop. However, the structural model is not able to use this spanwise distribution, but needs a chordwise distribution on the lower and upper surface of the wing. Therefore, the last section of the aerodynamic model determines the chordwise distribution of the force on both surfaces using the pressure distributions and the spanwise force distribution. The theory of this method is explained in Section 3.2.8.

The aerodynamic model determines the chordwise force distributions through the following steps:

1. The Reynolds number of each panel is determined, which differs due to the change in local chord length.
2. An array of Reynolds numbers used for the generation of the MSES files needs to be specified here. For this research this is an array from 8E6 to 2.8E7 with intervals of 4E6. These have been chosen because these are the minimum and maximum Reynolds numbers encountered in the evaluated flight path of Eijkelhof [18]. If other Reynolds numbers are used for the generation of the 2D airfoil data, the same Reynolds number array needs to be inserted here.
3. A similar array as for the Reynolds numbers needs to be specified for the angles of attack. The angles of attack used for the 2D data range from -15 to 20 with an interval of 0.25. Again, if these angles are changed for the generation of the 2D data, this array needs to be adjusted in similar fashion.
4. For all panels, the actual Reynolds numbers and Reynolds numbers used for the generated data are compared. For each panel a Reynolds number used for the data generation that is closest to the actual Reynolds number is determined. The same thing is done for the angle of attack, resulting in a Reynolds number and angle of attack for each spanwise panel that is also used in the data generation.
5. For each spanwise panel of the front wing, the chordwise force distribution is determined with the function *Chordwise_force*. This function needs the Reynolds number and angle of attack determined in step 4, the airfoil extension, the number of chordwise panels, the number of airfoil elements and the spanwise lift distribution as inputs. From these inputs, the function outputs the force magnitudes on both surfaces and the chordwise position of these forces. The method itself is explained in more detail in Section 3.2.8.

The result from step 5 is the input needed for the structural model of the FSI loop.

A.5. Structural model

The structural model is compared to the aerodynamic model relatively simple, as the more difficult part of the structural model is setting up the FE model and running it in MSC Nastran. In this FSI model, this is already done and only the results from the Nastran model are taken to calculate the deformations.

The structural model needs 5 inputs to run: the force matrices for the upper and lower surface of the wing, both the transformation matrices and the number of sections of the *halfspan*. Furthermore, the 'Pre-FSI' file needs to be loaded to obtain the structural parameters, e.g. the stiffness matrix. The structural model outputs the deflection at the quarter chord, the structural twist and the new deformed FE grid.

The structural model works according to the following steps:

1. The forces on the aerodynamic mesh are transferred to the structural mesh using the inverse distance weighting transformation matrices (one for the upper surface and one for the lower surface).
2. The structural deformations are determined using:

$$\boldsymbol{F} = \boldsymbol{K}\boldsymbol{x} \tag{A.7}$$

3. The deformations on the structural mesh are transferred back to the aerodynamic mesh with the use of the thin-plate spline transformation matrix.
4. The twist due to the structural deformation is determined by taking the deformed LE and TE. When these are deformed differently, there is a resulting angle between the two points.
5. The deformation of the lifting line is taken as the deformation at the quarter chord of the upper surface of the FE grid. This introduces the assumption that the airfoil shape remains undeformed under the deformation. As the wingbox is very stiff, this assumptions is deemed acceptable.

A.6. FSI loop

Now that all the building blocks of the FSI model have been explained, the iteration loop itself is the last part of the model. The iteration loop runs the aerodynamic model and structural model in a loop for every state. The loop runs in the following order:

1. The deflection, structural twist, structural dihedral and force are taken from the previous state or iteration as input for the iteration. In case there is no previous state or iteration, these parameters are taken to be zero. At the moment, the first state is thus an array of zeros for all four parameters. If one wants to change the initial state, it should be changed here.
2. The time step size is determined using the current velocity.
3. The deflection, structural twist, dihedral and total force are set to the value of the previous state.
4. The aerodynamic model is run using the deflection, structural twist and dihedral as inputs, outputting the force distributions on both surfaces and a new force vector.
5. The structural model is run with the force distributions of the surfaces, giving the deflection, twist and dihedral due to the deformation. The forces outputted by the aerodynamic model are transferred to the structural mesh within the structural model, using the coupling method described in Section 2.3. At the end of the structural model, the deformations are transferred back to the aerodynamic mesh and thus the output of the structural model is defined on the aerodynamic mesh.
6. The convergence criteria are checked. If none of the criteria is reached, steps 3-5 are repeated.
7. Before proceeding with the next state, the force vector, structural twist, structural dihedral and deflection are stored in matrices. This gives matrices for the four variables where the result for every state can be found.

A.7. General comments

If the FSI model returns an array of NaN, this is due to the range of polar data. If MSES is unable to converge for the full range and the lifting line model requires a lift coefficient outside of the converged range, it will produce a NaN for the circulation. When one NaN is in the array of the circulation, this will produce an entire array of NaN's in the following iteration as a result of the matrix multiplication.

B

MegAWES planform

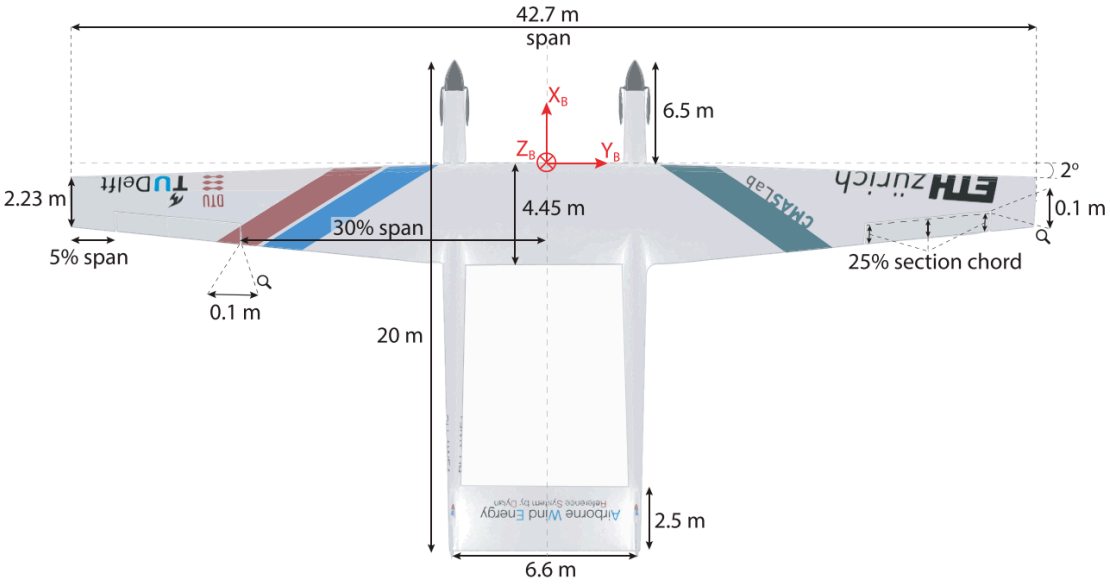


Figure B.1: Wing planform of the used model [18]

Parameter		Value	C/O
Center of Gravity	[m,m,m]	-1.67, 0, 0.229	O
Total aircraft mass	[kg]	6885.2	O
Wing:			
Span	[m]	42.7	C
$Chord_{root}$	[m]	4.45	C
$Chord_{tip}$	[m]	2.23	C
LE Sweep	[°]	2	C
Aspect Ratio	[-]	12.1	C
Surface Area	[m^2]	150.3	C
$Airfoil_{root}$	[-]	$RevE_{HC}$	C
$Airfoil_{tip}$	[-]	$RevE_{HC}$	C
Front spar	[% c_{local}]	33.3	O
Back spar	[% c_{local}]	43.4	O
$Aileron_{root\text{-}inner\ rib}$	[% $b_{1/2}$]	60	C
$Aileron_{root\text{-}outer\ rib}$	[% $b_{1/2}$]	90	C
$Aileron_{LE\text{-}spar}$	[% c_{local}]	75	C
Total number of ribs	[-]	50	O
Horizontal Tail/Elevator:			
Span	[m]	6.7	C
Chord	[m]	2.5	C
Airfoil	[-]	NACA 0012	C
Vertical Tail:			
Span	[m]	3	C
Chord	[m]	2.8	C
Airfoil	[-]	NACA 0012	C
Fuselages:			
Length	[m]	20	C
Radius	[m]	0.6	C
$X_{Nose\text{-}LE_{wing}}$	[m]	6.5	C
$Y_{Root\text{-}Fuselage}$	[m]	3.8	C

Figure B.2: Planform parameters [18]

C

MSES modification

The MSES version used in this research is a modified version of the original MSES program. Eijkelhof modified the original MSES code to improve stall prediction. The method used is from Van Rooij [53], which used this method to adjust XFOIL for better stall prediction, resulting in RFOIL. This method is also described by Van Den Kieboom et al. [59].

In the wake interval of the boundary layer source code files, the shear lag coefficient is adjusted. For both the laminar and turbulent interval the following shear lag coefficient, proposed by Van Rooij [53] is used:

$$K_c = 4.65 - .95\text{tanh}(.275H_k - 3.5) \tag{C.1}$$

This is the only modification made to the original MSES source code.