AALBORG UNIVERSITY

Master Thesis

CFD-Based Control System for Gust Load Alleviation on a 2D Aerofoil

Manuel Titel
Electro-Mechanical System Design
Aalborg University
12. February 2024

Department of Materials and Production
Machine and Production
Fibigerstræde 14
9220 Aalborg Ø
https://www.mp.aau.dk/

Title:

Control System for Gust Load Alleviation
on a Two-Dimensional Aerofoil

Project:

EMSD4 - Master Thesis

Project period:

October 2023 - February 2024

Student:

Manuel Titel

Supervisor:

Chungen Yin

Number of pages: 37
Finished 12-02-2024

Preface

This project has been written in LaTeX. Sources are being referred to by the Harvard Method. Some sections are based upon methods of calculation from a single or multiple books. In these sections the book or books will be presented in the beginning of the section and a reference will not be added for every formula used. All figures and tables have been produced by the student unless otherwise stated.

Manuel Titel

Table of Contents

Abstract

This master thesis aims to develop a control system for gust load alleviation on a two-dimensional aerofoil using a trailing edge flap. A comprehensive CFD study, focused on accurate lift force prediction, was conducted using the Spalart-Allmaras turbulence model in Ansys Fluent. Available experimental data for surface pressure and boundary layer velocity profiles were used to validate the CFD results and showed good agreement. The effect of the flap deflection angle as well as the effect of different horizontal and vertical gusts on the aerofoil's lift force were simulated with the CFD model. Based on these CFD simulations, a reduced order model was developed, which can predict the required flap deflection angle to counteract the effects of incoming gusts. This reduced order model was then implemented in User Defined Functions, which were hooked to the CFD model and also defined the deforming mesh and aerofoil geometry as well as the changing velocity during the different gusts. The gust load alleviation system was tested for different representative gust cases using the CFD model. It was found that most of the gusts were alleviated well. Generally, the performance was better during gusts with smaller velocity gradients.

Chapter 1
Introduction

During flight, an aeroplane might be subjected to different vertical and horizontal gusts. The lift force produced by the wings is significantly affected by incoming gusts and increases the mechanical load on the wing. The wing structure and material would therefore have to be strong enough to withstand even the strongest gusts, which would result in additional weight and cost.

Gust Load Alleviation (GLA) in modern aeroplanes allows for weight and drag reduction, which in turn reduces material and fuel consumption (Li and Qin, 2022). In addition, it can provide improved passenger comfort (Spalart and Venkatakrishnan, 2016). The concept is also applicable to the blades of wind turbines (Andersen, 2010). Usually, aerodynamic control surfaces on the wings, such as trailing edge flaps are used to alleviate the load from incoming gusts. With the use of LIDAR sensors, incoming gusts can be measured with a forward detection range of 50 m, which corresponds to a certain lead time for the control surfaces to perform, depending on the speed of the aeroplane (Li and Qin, 2022).

In this thesis, a CFD model is developed for accurately predicting the lift force on a two-dimensional aerofoil. The CFD model is validated with surface pressure data and boundary layer velocity profiles from the experiments by Nakayama (1985). Representative cases of different vertical and horizontal gusts are defined and simulated using the CFD model. A reduced order model is developed, which can predict how the lift force will be affected by incoming gusts. The original aerofoil geometry is modified with a trailing edge flap, which will be used for alleviating the gust load. CFD simulations of the trailing edge flap at different deflection angles are conducted and incorporated in the reduced order model as well. This reduced order model is then implemented in a User Defined Function, which will be used to simulate the GLA system in the CFD model and test it for the different gust cases.

Gust load alleviation is a complex topic, some aspects of which are beyond the scope of this thesis. Here, the flow is modelled as two-dimensional, which does not account for the effects of three-dimensional finite wings and how the flow might be affected by other parts of the aeroplane. Furthermore, the wing is treated as rigid, so the dynamic structural response of the wing at different gust speeds and frequencies is not considered. This thesis is neither concerned with the aerodynamic efficiency of the trailing edge flap shape nor the design and limitations of the flap actuators.

Chapter 2
CFD Modelling

In this chapter, a CFD model for a two-dimensional aerofoil will be developed in Ansys Fluent. This model will later be used for the performance evaluation of the gust load alleviation system. As the gust load is mostly due to changes in lift force, the focus of the CFD model is to provide an accurate prediction of the lift coefficient C_L. In addition, the drag coefficient C_D is used to judge the quality of the model. The desired accuracy for these two coefficients is chosen to be at least 1 %.

2.1 Aerofoil Geometry

The aerofoil under consideration is the *Model A* aerofoil in Nakayama (1985) and will be referred to as the Nakayama aerofoil. It is a 10 %-thick conventional non-symmetric aerofoil with a chord length $c = 24\,\text{in} = 0.6096\,\text{m}$ and a blunt 0.6 mm thick trailing edge. The aerofoil geometry is shown in figure 2.1.

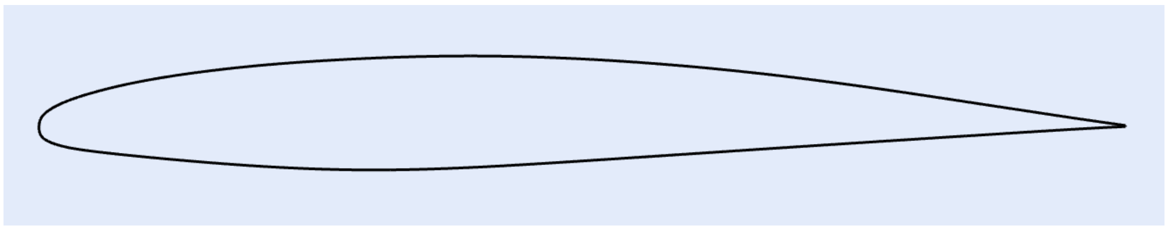


Figure 2.1. Aerofoil geometry

2.2 Initial Mesh

The focus of the CFD model is to accurately predict lift and drag, which arise from the pressure and viscous forces on the aerofoil surface. It is therefore essential, that the mesh properly resolves boundary layer around the aerofoil. In order to achieve this, the wall y^+-value should be equal to 1 or less (Ansys, 2022). The corresponding minimum wall spacing is calculated using the y^+-calculator from Cadence Design Systems (2024). Inserting the chord length and the values for air at standard conditions, under which the experiments by Nakayama (1985) were conducted, yields a minimum wall spacing of 11×10^{-6} m. Other parameters and the overall structure of the initial mesh are based on the meshes in Rumsey (2019) and the recommendations in Lu et al. (2021). Illustrative examples of the two meshes in the near-field are shown in figure 2.2 and 2.3.

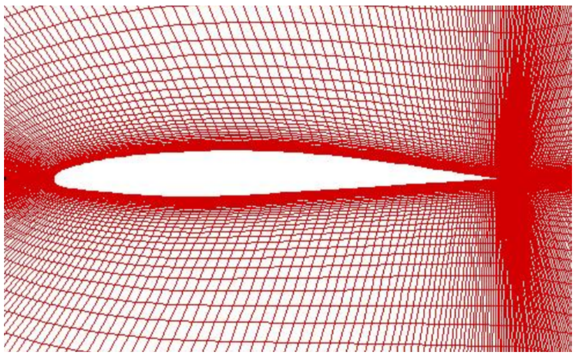


Figure 2.2. Grid in Rumsey (2019)

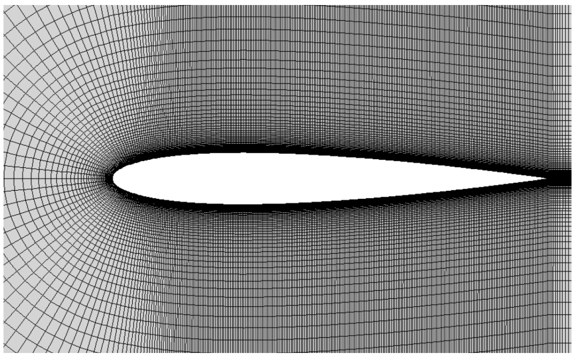


Figure 2.3. Grid in Lu et al. (2021)

Both studies found that a structured C-mesh resulted in accurate predictions of the lift coefficient. This type of mesh will thus also be used here. It should be noted that the two studies were conducted at different Reynolds numbers and that the aerofoil geometries were similar, but not identical to each other. Furthermore, some mesh parameters used in one study were not explicitly stated in the other. Therefore, some of these parameters are not directly comparable and have to be calculated or expressed differently in order to be applicable to the case presented here. In this comparison, the following parameters are used to define the C-mesh for the aerofoil under consideration:

- Number of nodes along the chord N_{chord}
- Minimum wall spacing on the aerofoil surface y_{min}
- Layer-wise growth rate in y-direction g
- Far-field extend as a multiple of the chord length l_{far}
- Number of nodes in the wake per unit length n_{wake}

It should be noted that a small n indicates a number of nodes per unit length, whereas a capital N indicates an absolute number of nodes. The number of nodes in radial direction N_{radial} can be determined from a given minimum wall spacing, growth rate and far-field extend, using equation 2.1:

$$l_{far} = \sum_{n=1}^{N_{radial}} y_{min} \cdot g^n \tag{2.1}$$

The advantage of using growth rate and minimum wall spacing to define the mesh is that these parameters are independent of the far-field extend, whereas the number of radial nodes is not.

Mesh Properties in Rumsey (2019)

In Rumsey (2019), the same aerofoil as in this thesis has been studied as a verification case for different CFD-codes and turbulence models. Mesh data for five C-type meshes with different refinement levels have been provided. When looking at the results obtained with increasing levels of mesh refinement in Rumsey (2023a), the improvement in accuracy becomes relatively small, when going from the third-finest to the second-finest mesh. This is especially apparent for the drag coefficient. The third-finest mesh is therefore chosen as a reference for this comparison. The given mesh properties are shown in table 2.1

Parameter	**Value**
N_{chord}	129
N_{radial}	150
l_{far}	$20c$
N_{wake}	153
Re	1.2×10^6
c	$0.6096\,\text{m}$

Table 2.1. Parameters in Rumsey (2019)

The minimum wall spacing and growth rate of this mesh have not been explicitly stated, but can be determined using the available information for the finest mesh, which has 385 radial nodes and a minimum wall spacing $y_{min} = 2.5 \cdot 10^{-6}\ m$. These values are inserted into equation 2.1:

$$\sum_{n=1}^{385} y_{min} \cdot g^n = \sum_{n=1}^{385} 2.5 \times 10^{-6}\,\mathrm{m} \cdot g^n = 20c = 20 \cdot 0.6096\,\mathrm{m} = 12.192\,\mathrm{m} \tag{2.2}$$

This yields a growth rate $g \approx 1.0315$ for the finest mesh. Each coarser grid was exactly every other point of the next finer grid, so the third-finest grid must have been every fourth point of the finest grid. The minimum wall spacing or the height of the first radial cell of the third-finest grid can thus be calculated as follows:

$$\sum_{n=0}^{3} 2.5 \times 10^{-6}\,\mathrm{m} \cdot 1.0315^n = 10.48 \times 10^{-6}\,\mathrm{m} \tag{2.3}$$

The height of the next cell in radial direction is then given by:

$$\sum_{n=4}^{7} 2.5 \times 10^{-6}\,\mathrm{m} \cdot 1.0315^n = 11.867 \times 10^{-6}\,\mathrm{m} \tag{2.4}$$

The growth rate of the third-finest mesh is then obtained by dividing the height of the second radial cell by the height of the first radial cell:

$$g = \frac{1.1867 \times 10^{-5}\,\mathrm{m}}{1.048 \times 10^{-5}\,\mathrm{m}} = 1.1321 \tag{2.5}$$

The number of wake nodes per unit length is calculated as:

$$n_{wake} = \frac{N_{wake}}{l_{far}} = \frac{153}{20c} = \frac{153}{20 \cdot 0.6096\,\mathrm{m}} = \frac{12.55}{m} \tag{2.6}$$

Recommended Mesh Properties in Lu et al. (2021)

In Lu et al. (2021), a comparative study of different meshes has been conducted for RANS-simulations of a NACA 0012 aerofoil. The aerofoil parameters and recommended mesh properties are shown in table 2.2.

Parameter	**Value**
N_{chord}	670
N_{radial}	150
g	1.10
l_{far}	$30c$ (wake $60c$)
N_{wake}	300
Re	6×10^6
c	1 m

Table 2.2. Parameters in Lu et al. (2021)

The minimum wall spacing has not been explicitly stated, but can be determined using equation 2.1:

$$l_{far} = 30c = 30 \cdot 1\,\mathrm{m} = \sum_{n=1}^{150} y_{min} \cdot 1.10^n \tag{2.7}$$

The above equation is satisfied if $y_{min} \approx 1.85 \times 10^{-6}\,\text{m}$.

The number of wake nodes per unit length is calculated as:

$$n_{wake} = \frac{N_{wake}}{l_{far,wake}} = \frac{300}{60c} = \frac{300}{60 \cdot 1\,\text{m}} = \frac{5}{m} \tag{2.8}$$

Selection of Initial Mesh

The mesh properties from Lu et al. (2021) and Rumsey (2019) as well as the selected properties for the initial mesh are summarised in table 2.3.

Parameter	**Lu et al. (2021)**	**Rumsey (2019)**	**Initial Mesh**
N_{chord}	670	129	200
y_{min}	$1.85 \times 10^{-6}\,\text{m}$	$10.48 \times 10^{-6}\,\text{m}$	$11.0 \times 10^{-6}\,\text{m}$
g	1.10	1.13	1.15
l_{far}	$30c$ (wake $60c$)	$20c$	$20c$
n_{wake}	5/m	12.55/m	5/m
Re	6×10^6	1.2×10^6	1.2×10^6
c	1 m	0.6096 m	0.6096 m

Table 2.3. Properties of different meshes

It is evident from the numbers in table 2.3 that the mesh from Rumsey (2019) is generally coarser than recommended by Lu et al. (2021). However, as pointed out by Lu et al. (2021), cases with higher Reynolds number require more cells around boundaries. Furthermore, in contrast to the mesh in Lu et al. (2021), the grid lines in Rumsey (2019) are skewed in the leading and trailing edge region, as can be seen by visual inspection of figure 2.2. In this way, the mesh remains structured and is locally refined, without increasing the number of cells. The smaller Reynolds number and the skewed grid lines in Rumsey (2019) could explain why this relatively coarse mesh still produces satisfactory results. It should be emphasised that the value of n_{wake} for Rumsey (2019) is only an average value and cannot represent the effect of the locally refined wake mesh close to the trailing edge.

The initial mesh is shown in figure 2.4. Skewing of the grid lines as in Rumsey (2019) is not used here. Instead, the horizontal grid spacing is kept close to constant along the aerofoil surface and in the wake, similar to Lu et al. (2021). Therefore, it is expected that N_{chord} must be larger than in Rumsey (2019) in order to properly resolve the leading end trailing edge region. For the initial mesh, $N_{chord} = 200$ is chosen. The minimum wall spacing as previously calculated from the desired y^+-value is very close to Rumsey (2019). The other initial parameters are chosen to be slightly coarser or as coarse as the coarsest parameters from the two studies.

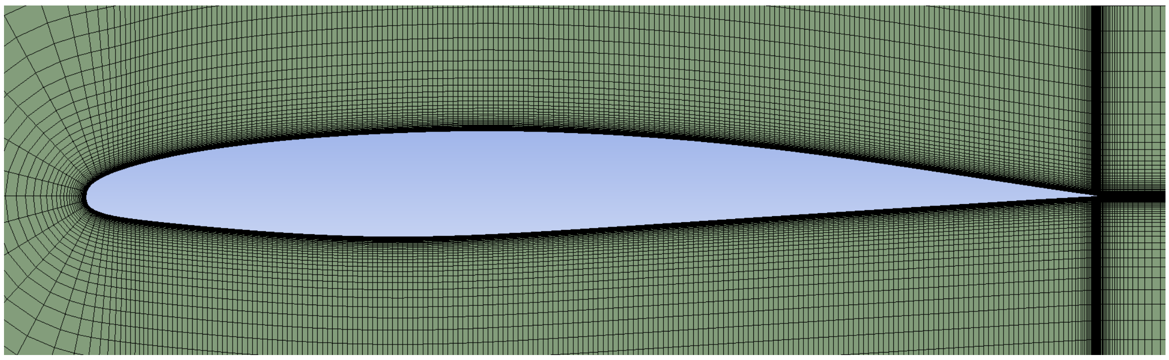


Figure 2.4. Initial mesh (near field)

In contrast to Lu et al. (2021) and Rumsey (2019), the aerofoil has a 0.6 mm thick blunt trailing edge, which is comparable to the viscous-sublayer thickness, as mentioned by Nakayama (1985). Initially, this trailing edge is not sharpened, but precisely modeled. The minimum wall spacing is kept constant along the whole aerofoil surface, including the blunt trailing edge. In order to create a smooth transition from the constant grid spacing to the minimum wall spacing at the trailing edge, the same growth rate g as in radial direction is used. As can be seen in figure 2.5, the horizontal grid spacing gradually decreases, when moving along the aerofoil surface toward the trailing edge and increases again from the trailing edge until the constant wake grid spacing is reached. This results in 106 additional vertical grid lines and a significant local refinement in the trailing edge region. In total, the initial mesh has 78980 cells and 79771 nodes.

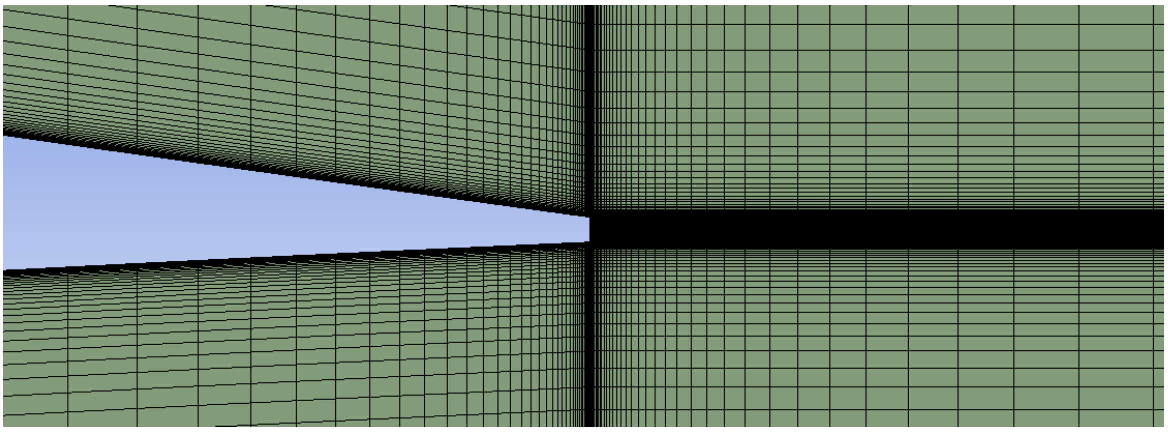


Figure 2.5. Initial mesh around the trailing edge region

2.3 Turbulence Model

In order to avoid high computational complexity, a RANS based turbulence model is used. Several studies exist, where this kind of turbulence model has successfully been used to calculate the flow around a two-dimensional aerofoil. For example, Narsipur et al. (2012) used the $k\omega$-SST model, while Nordangera et al. (2014) used the Spalart-Allmaras model. Both studies predicited C_L and C_D in good agreement with experimental data. Versteeg and Malalasekera (2007) state that both of these turbulence models are suited for external aerodynamics, with the Spalart-Allmaras model being specifically tuned for this purpose and the $k\omega$-SST model being applicable to more general problems. Polewski and Cizmas (2014) used both models on the Nakayama aerofoil and found that the $k\omega$-SST model yielded slightly better results for the velocity profiles. However, they also pointed out that the Spalart-Allmaras model still predicted the velocity profiles quite well, while being less computationally expensive. Here, the Spalart-Allmaras model is chosen, as it is less computationally expensive.

2.4 Boundary Conditions

The mesh of the whole domain with applied boundary conditions is shown in figure 2.6. Inlet and outlet boundaries of the C-mesh are defined as in Alulema et al. (2020) and Erkan and Özkan (2020). The blue edges of the domain boundary serve as an inlet, which is set to air at standard conditions with a velocity of 30 m/s in x-direction, as described in Nakayama (1985). The flow is made fully turbulent with a turbulent viscosity ratio of 3 at the inlet as in Rumsey (2023a). The red vertical edges of the domain in the wake are set to a pressure outlet boundary condition. It should be noted that the arrows do not indicate flow direction.

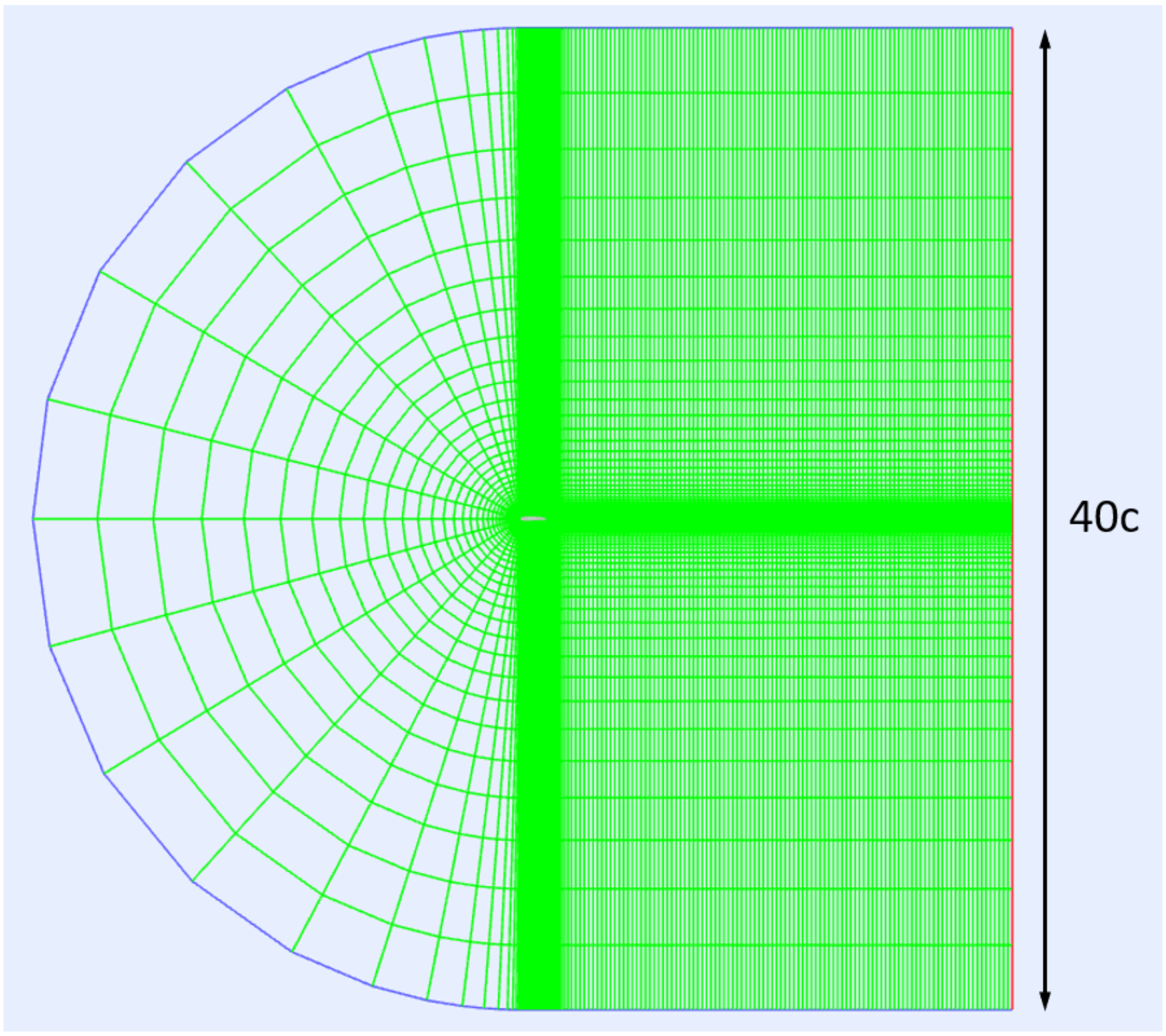


Figure 2.6. Mesh of the whole domain with boundary conditions

2.5 Solver Settings

Similar to Rumsey (2023a), vorticity based turbulence production and an upwind approach are used. It is mentioned in Rumsey (2021) that this essentially incompressible case has been simulated using compressible codes, which might cause minor differences when comparing the results to incompressible calculations. Here, the case is modelled as steady and incompressible, i.e. the pressure-based solver is used. The settings in Ansys Fluent are summarised in table 2.4.

Property	**Setting**
Time	Steady
Solver Type	Pressure-based
Turbulence model	Spalart-Allmaras
Turbulence production	Vorticity based
Gradient	Least squares cell based
Pressure	Second order
Momentum	Second order upwind
Turbulent kinetic energy	Second order upwind
Specific dissipation rate	Second order upwind

Table 2.4. Settings in Ansys Fluent

2.6 Initial Results

After 1000 iterations, the residuals of x-momentum, y-momentum, continuity and turbulent viscosity where all settled below 10^{-4} as can be seen in figure 2.7, so the solution was well converged.

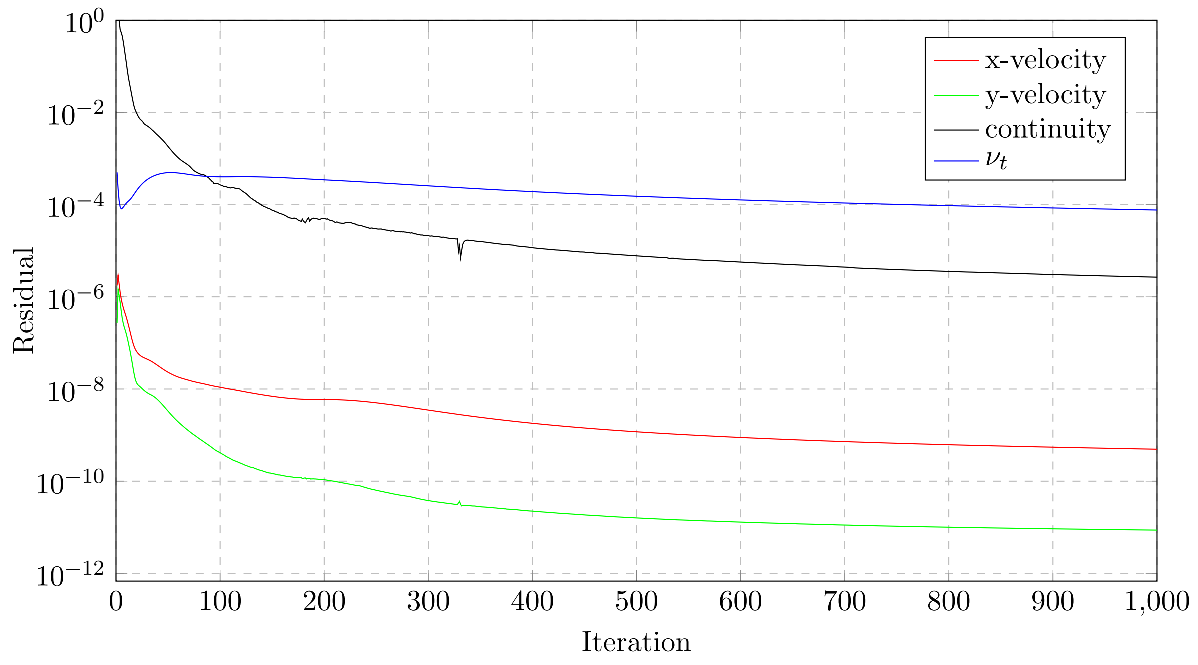


Figure 2.7. Residuals with initial mesh

Figure 2.8 and 2.9 show the convergence of C_L and C_D. After 200 iterations, the changes in C_L and C_D compared to iteration 190 were well below 1 %. The initial results were $C_L = 0.1613$ and $C_D = 0.0103$, which is very close to the results in Rumsey (2023a), with a difference of about 1 % for both C_L and C_D.

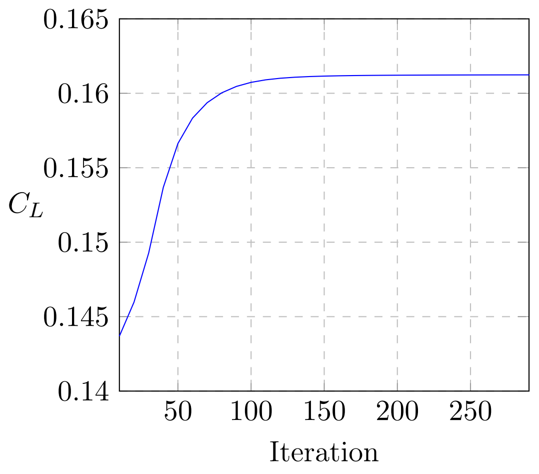


Figure 2.8. Convergence of C_L

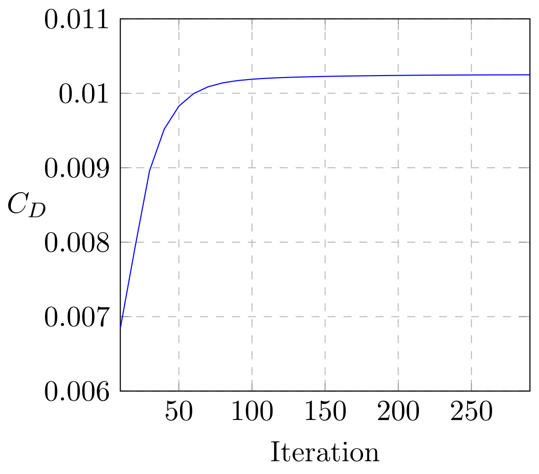


Figure 2.9. Convergence of C_D

The velocity and pressure field around the aerofoil are shown in figure 2.10 and 2.11, respectively. As expected for a non-symmetric aerofoil, the higher velocity and the lower pressure on the upper surface compared to the lower surface can clearly be seen, resulting in the positive lift coefficient at 0° angle of attack.

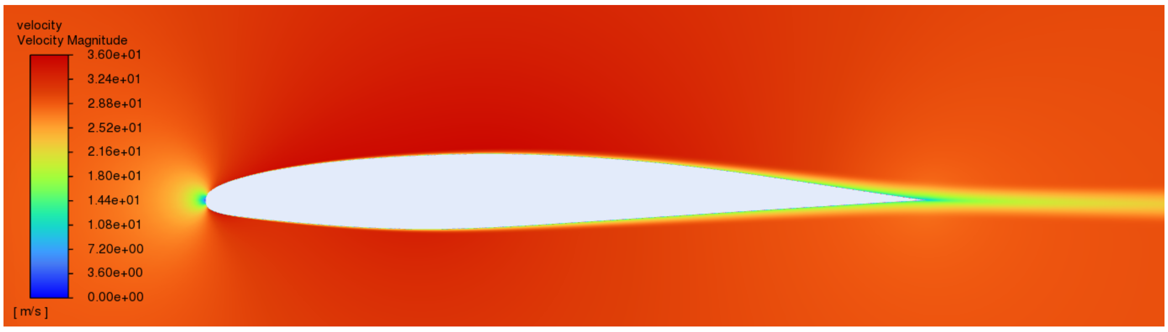


Figure 2.10. Velocity field around the aerofoil

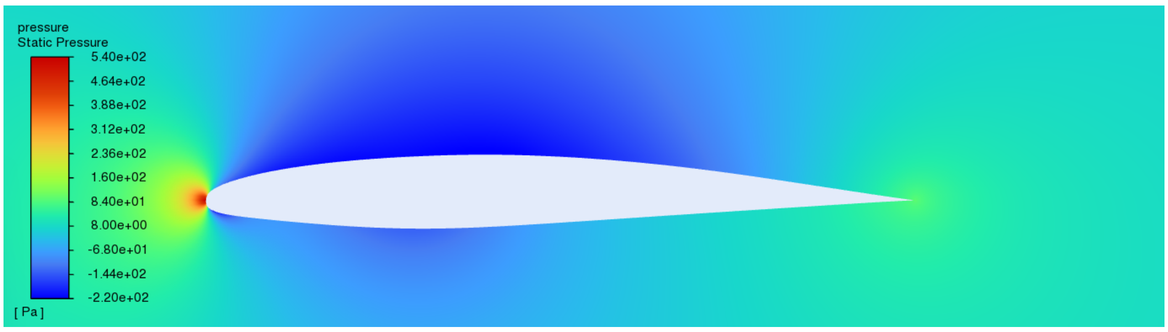


Figure 2.11. Pressure field around the aerofoil

For the initial mesh, figure 2.12 shows that the y^+-value on the whole aerofoil surface is almost always below 1 and never above 1.1. This is very close to the desired y^+-value and supports the good quality of the initial results.

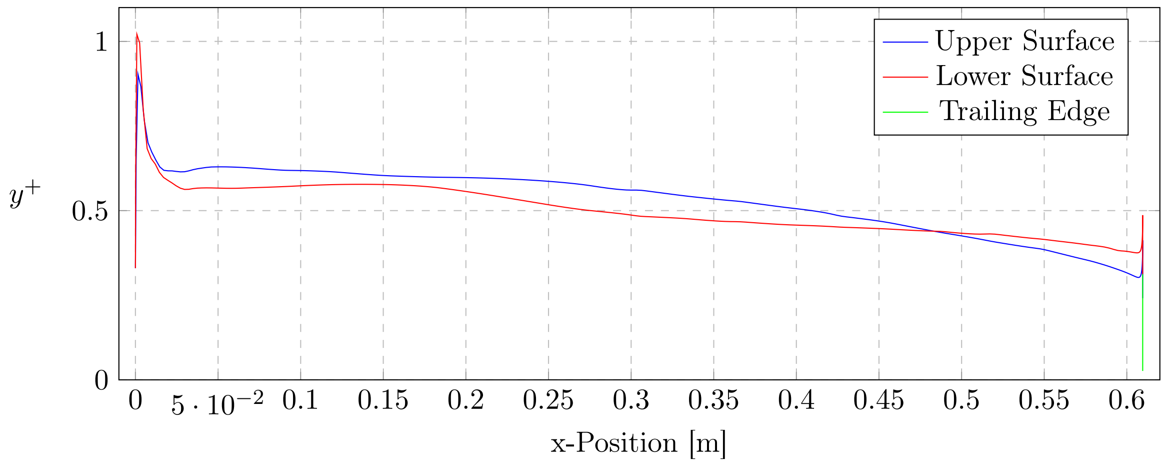


Figure 2.12. y^+ on the aerofoil surface with the initial mesh

2.7 Mesh Refinement

In order to determine whether the obtained results are grid independent, the mesh was gradually refined using a refinement factor r. For each refinement step, the initial number of divisions on any line along which the domain has been divided was multiplied by r. This means that the growth rate decreases with increasing refinement factor, as given by

$$g = \sqrt[r]{g_{initial}} \tag{2.9}$$

Figure 2.13 and 2.14 show the values for C_L and C_D after 1000 iterations for different levels of mesh refinement. In most cases, the changes in C_L and C_D after 200 iterations were insignificant. The initial mesh with $r = 1$ is marked in red.

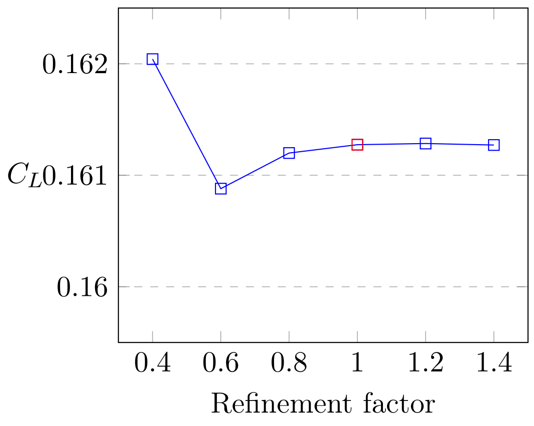


Figure 2.13. Refinement influence on C_L

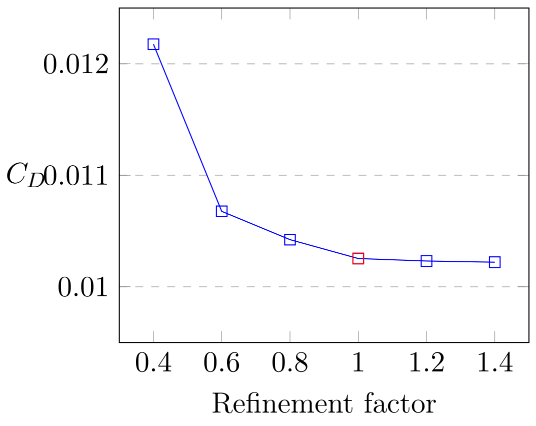


Figure 2.14. Refinement influence on C_D

It was found that the initial mesh yielded very good results, which could not be significantly improved by further refinement. Coarsening the mesh showed noticeable changes, especially when looking at the drag coefficient. Therefore, the initial mesh is already sufficiently fine to be accurate, but at the same time coarse enough to keep the computational cost low.

2.8 Extended Far-field

As mentioned in Rumsey (2023a), the far-field should be extended further to $500c$ for verification purposes. Thus, the initial mesh has been extended to $500c$ using the same initial growth rate $g = 1.15$ in all directions. This led to an increase in the number of cells by 40 %, but did not affect the results significantly. The observed changes were $-0.74\,\%$ for C_L and $0.63\,\%$ for C_D, which is in good agreement with the changes observed in Rumsey (2023b). As these changes are below 1 %, the initial far-field extend of $20c$ is deemed sufficient and used for all further simulations.

2.9 Sharpened Trailing Edge

The high mesh resolution along the blunt trailing edge adds significant complexity to the mesh, which would not be necessary with a sharp trailing edge. It is therefore investigated, whether a modified aerofoil geometry with a sharpened trailing edge would still yield accurate predictions of C_L and C_D. For the results to be as comparable as possible, the mesh is essentially the same as the initial mesh, but the wake cells behind the 0.6 mm thick trailing edge have been removed and the remaining cells have been slightly stretched in y-direction to evenly cover the empty space. In this way, the number of cells could be reduced by 13 %. A close up of the mesh around the blunt and sharpened trailing edge is shown in figure 2.15 and 2.16, respectively. For reference, both figures show the last 0.4 mm of the aerofoil.

First, the trailing edge was sharpened by joining the last to coordinate points ($x = c$) at the midpoint of the trailing edge ($y = 0$). This increased the included angle at the trailing edge, while the chord length remained unchanged. The lift coefficient changed by $-3.1\,\%$ and the drag coefficient by $-0.2\,\%$. This change in lift coefficient is considered to large for a satisfactory result.

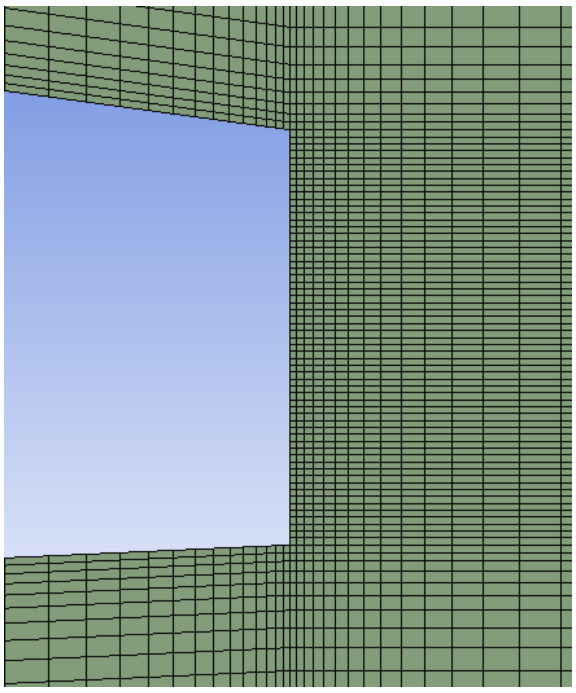


Figure 2.15. Close up of blunt trailing edge with initial mesh

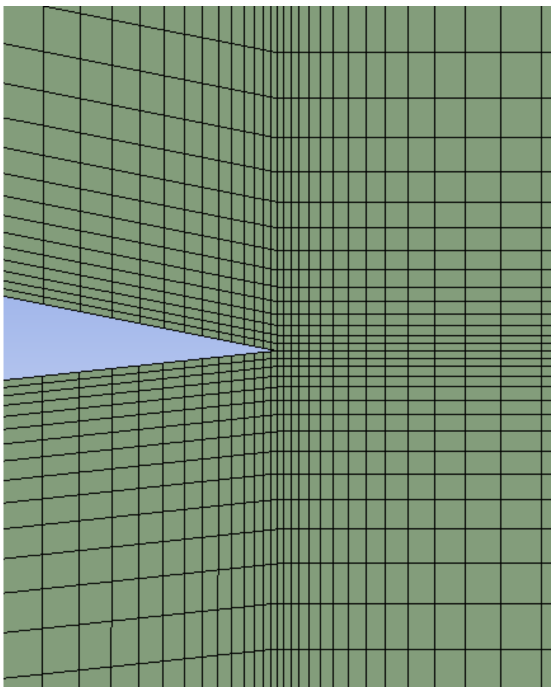


Figure 2.16. Close up of sharp trailing edge with modified initial mesh

Alternatively, the trailing edge was sharpened by extending the tangents on upper and lower surface and moving the trailing edge to the intersection point. In this way, the included angle at the trailing edge remains the same, but there is a slight increase in chord length. The observed changes were 0.1 % for C_L and −0.1 % for C_D, which is sufficiently accurate. This type of sharpened trailing edge will therefore be used in all further calculations. It should be noted that the calculation of lift and drag coefficient depends on the chord length as a reference value and therefore, the slight increase in chord length would have to be accounted for. However, it was found that this results in slightly worse results, with C_L changing by 0.4 % and C_D by −0.5 %. Thus, the original chord length is used as a reference value for the calculation of C_L and C_D with this sharpened trailing edge model.

The two differently sharpened trailing edges are shown in figure 2.17. For reference, the blunt trailing edge is shown as well and the x-axis is indicated by the dotted line.

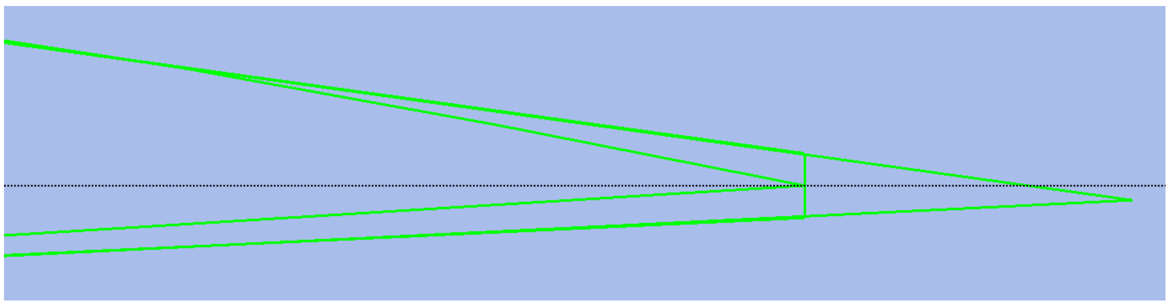


Figure 2.17. Differently sharpened trailing edges

As mentioned previously, the mesh was locally refined in horizontal direction around the trailing edge region. The initial motivation to do that was to maintain a constant minimum wall spacing and growth rate in all directions. However, this was only necessary with the blunt trailing edge and might potentially not be needed with the sharp trailing edge. Thus, the mesh was coarsened, by keeping the horizontal grid spacing constant along the whole aerofoil surface and increasing it from the trailing edge to smoothly transition to the constant wake grid spacing. In this way, the cell count could be reduced even further by 28 %, compared to the initial mesh for the blunt trailing edge. While C_D was not affected by the coarser mesh, C_L changed by 1.5 %, which is considered to large a difference. Thus, the local mesh refinement around the trailing edge proved to be beneficial for the sharp trailing edge as well and is therefore kept for all further calculations.

Chapter 3
Model Validation

The CFD model, which has been shown to yield grid independent results for C_L and C_D, will be validated using the experimental data from Nakayama (1985). The lift and drag coefficient have not been experimentally determined. However, pressure data from the aerofoil surface as well as boundary layer velocity profiles are available. A good agreement with these data indicates a good prediction of pressure and viscous forces and is therefore assumed to result in a good prediction of C_L and C_D.

3.1 Surface Pressure

The pressure on the upper and lower aerofoil surface is plotted in figure 3.1 and 3.2, respectively. The CFD results are shown for the initial mesh with $r = 1.0$ and the finest mesh from the refinement study with $r = 1.4$. On both upper and lower surface, the CFD results from the two different meshes are in perfect agreement with each other.

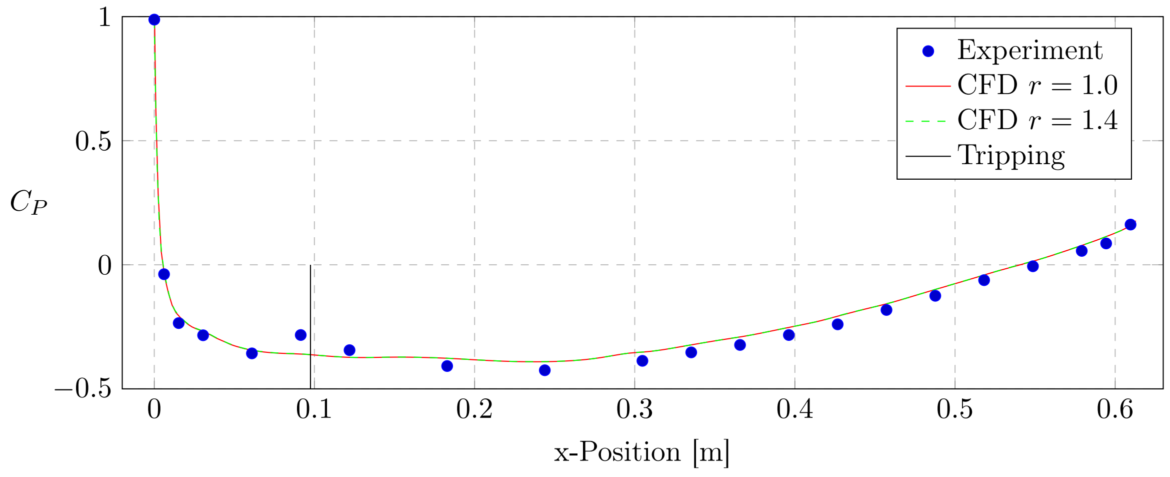


Figure 3.1. Pressure coefficient on upper aerofoil surface

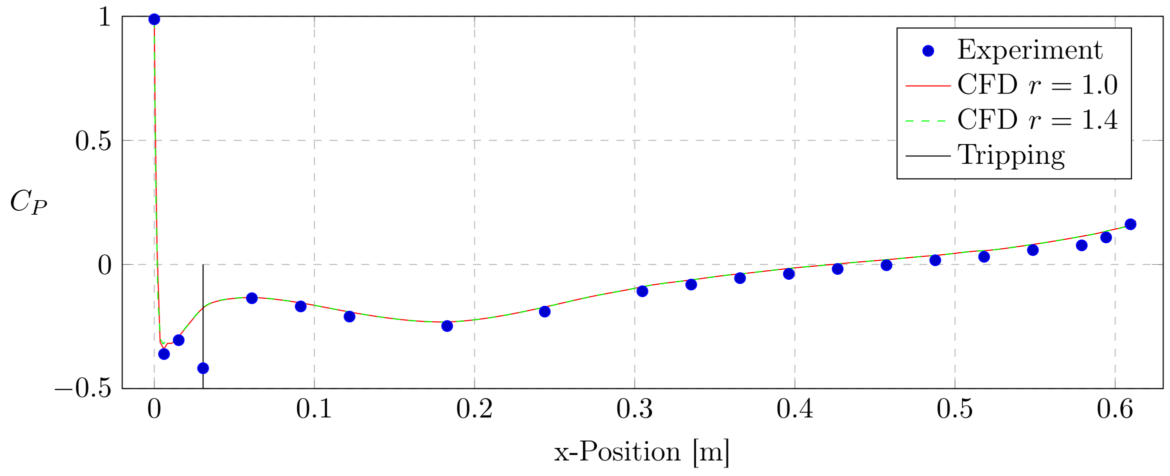


Figure 3.2. Pressure coefficient on lower aerofoil surface

It should be noted that in the experiments, the boundary layer was tripped with a thin wire at $0.16c$ on the upper surface and at $0.05c$ on the lower surface, which produces noticeable kinks in the experimental data. According to Tabatabaei et al. (2022), tripping is commonly used in wind-tunnel testing to fix the position of the laminar–turbulent transition and has a significant effect on relevant characteristics of the boundary layer and aerodynamics. In both figures, the CFD results are generally very close to the experimental data. Before the tripping, there is almost perfect agreement with the experiment. After the tripping, the CFD model slightly underpredicts the pressure coefficient. Oh and Choi (1997) pointed out that the downstream development of the boundary layer may depend heavily on the tripping location and showed that a precise modelling of the tripping location can significantly improve the agreement between CFD and experiment. It is therefore likely that the small difference between CFD results and experiment arises, because the boundary layer tripping has not been included in the model. However, the CFD model is still deemed sufficiently accurate for the purposes of this study.

3.2 Velocity Field

In Nakayama (1985), velocity measurements were made at selected locations on the upper and lower aerofoil surface as well as in the wake. From a given x-Position, which is expressed in terms of the chord length c, the probes were gradually moved away from the aerofoil surface in normal direction and in the wake they were moved in y-direction. The green lines in figure 3.3 indicate the positions, where the velocity measurements have been taken. The measurements at $x = 1.800c$, $x = 2.190c$ and $x = 3.000c$ are not included in the figure.

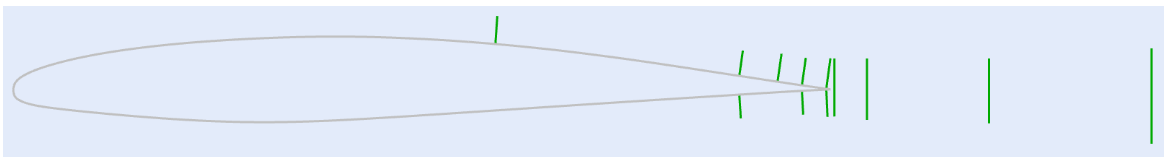


Figure 3.3. Velocity measurement positions

In figure 3.4, the velocity profile at $0.593c$ is shown, which has only been measured on the upper surface. The CFD model is in almost perfect agreement with the experiment.

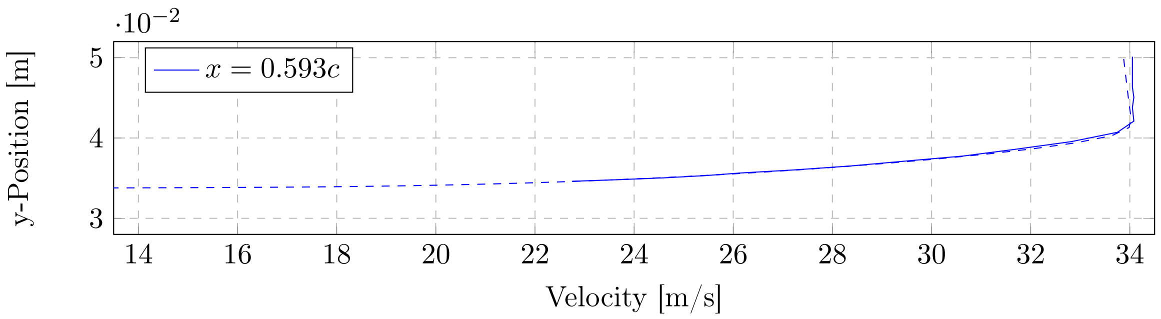


Figure 3.4. Velocity profile at $0.593c$ on upper surface from experiment (solid) and CFD (dashed)

The other velocity profiles on the upper and lower surface, which are much closer to the trailing edge, are shown in figure 3.5. Experimental data at $0.940c$ are only available on the upper surface. Within the boundary layer, the CFD model slightly overpredicts the velocities at positive y-positions, while those at negative y-positions are in perfect agreement with the experimental data.

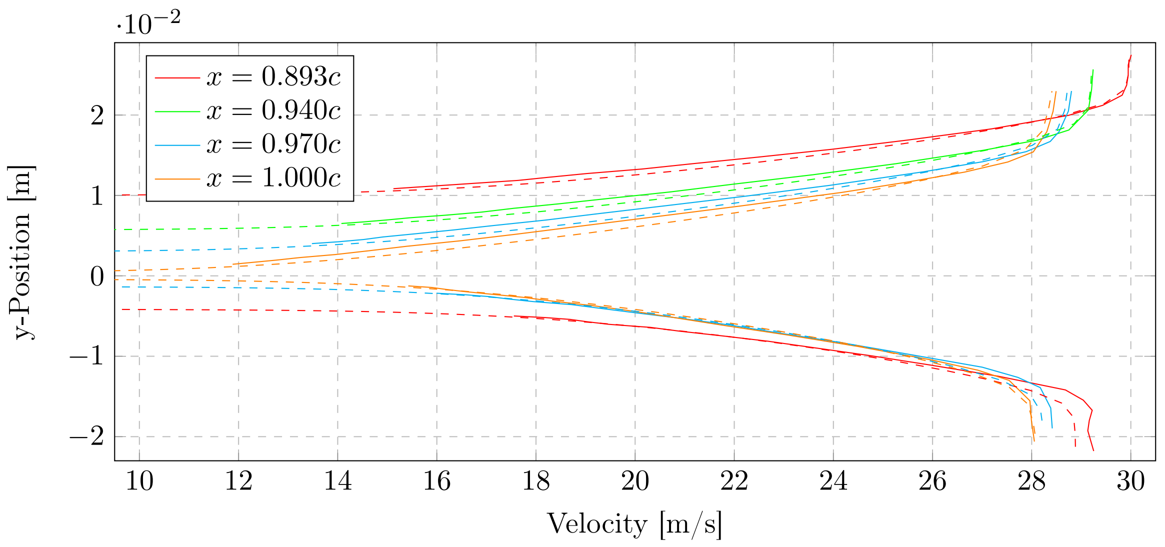


Figure 3.5. Velocity profiles on upper and lower surface from experiment (solid) and CFD (dashed)

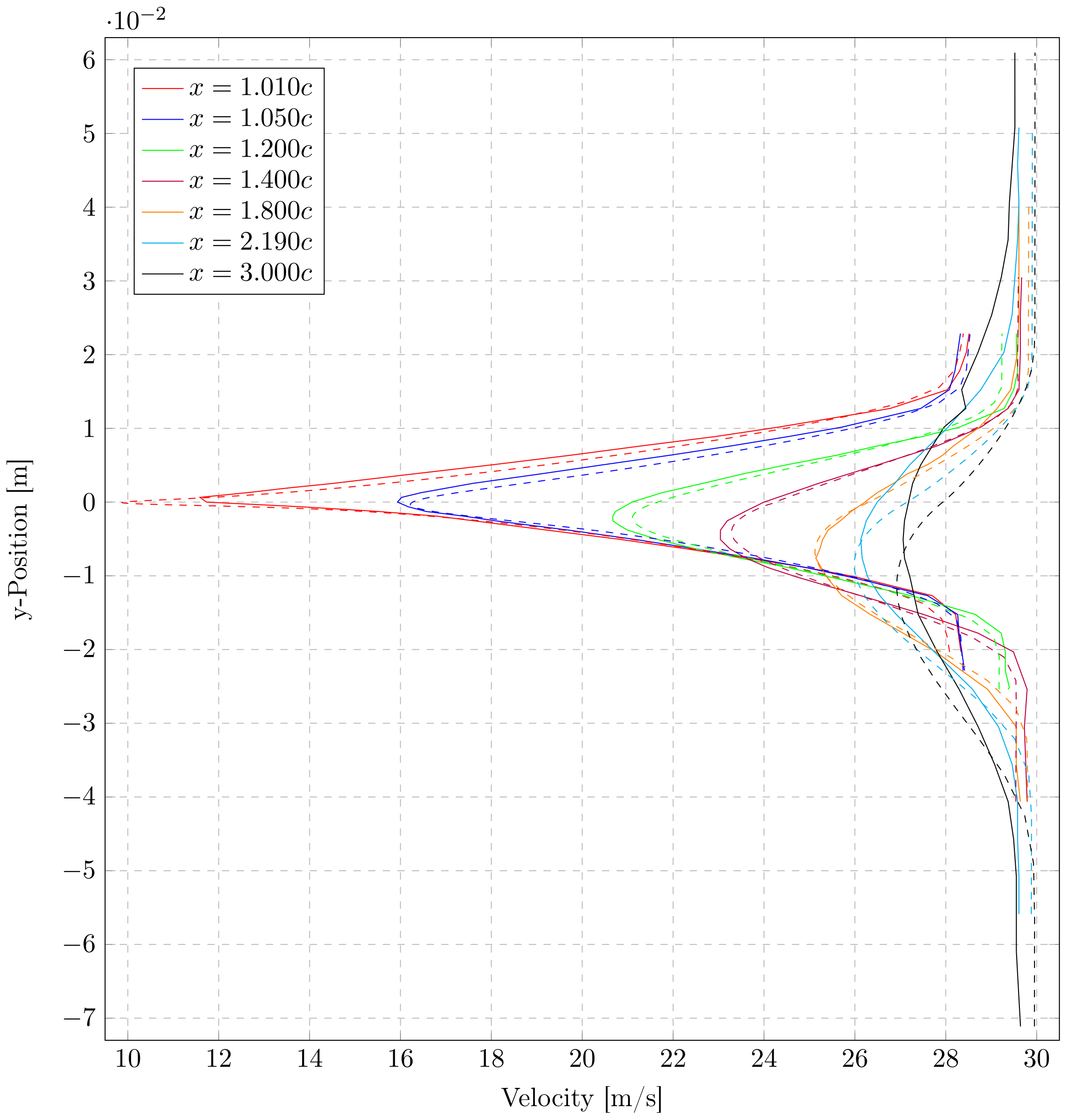


Figure 3.6. Velocity profiles in the wake from experiment (solid) and CFD (dashed)

When approaching the free-stream velocity, the agreement with the upper surface velocity profiles is quite good, while there are some differences of up to 0.5 m/s in the lower surface velocity profiles. In figure 3.6, the velocity profiles in the wake are shown. For the profiles, that are close to the trailing edge, i.e. at $1.010c$ and $1.050c$, the overprediction of the velocity for positive y-positions is similar to the profiles in figure 3.5, with the negative y-positions being closer to the experiment. Further away from the trailing edge, at $2.190c$ and $3.000c$, the differences between CFD model and experimental data become quite large. Similar differences were found in Rumsey (2023a) and reproduced by Polewski and Cizmas (2014), with both the Spalart-Allmaras and the $k\omega$-SST model. As the larger differences only occur far away from the aerofoil, they are not a concern for the purposes of this study and the CFD model is deemed sufficiently accurate for predictions of C_L and C_D.

It should should be noted that the axes in all three figures 3.4, 3.5 and 3.6 have the same scaling, even though they might cover different ranges. This makes them visually well comparable, as any differences between experiment and CFD model appear with the same size.

Chapter 4
Gust Load Modelling

The lift force produced by an aerofoil can be largely affected by incoming gusts. Before these additional loads can be alleviated, a model is needed, which can accurately predict the gust induced changes in lift force.

4.1 Gust Shapes

The aerofoil might be part of an aeroplane and thus be subjected to symmetrical and lateral gusts in level flight. These gusts will be modelled based on the specifications from EASA (2007). The shape of the gust is taken as follows:

$$U = \begin{cases} \frac{U_{ds}}{2}\left[1 - cos\left(\frac{\pi s}{H}\right)\right] & \text{for } 0 \leq s \leq 2H \\ 0 & \text{for } s > 2H \end{cases} \tag{4.1}$$

where

- s = distance penetrated into the gust [m]
- U_{ds} = design gust velocity [m/s]
- H = gust gradient [m] (distance for the gust to reach its peak velocity)

As the velocity of 30 m/s is well below the typical speed of an aeroplane, the specifications from EASA (2007) do not apply here. Instead, U_{ds} is chosen to be 10 % of the equilibrium velocity, i.e. ±3 m/s. For the gust gradient, two cases are chosen, such that a short gust with a duration 1 s and a long gust with a duration of 5 s are obtained. The corresponding values for H are calculated as follows:

$$H(1\,\mathrm{s}) = 30\,\frac{\mathrm{m}}{\mathrm{s}} \cdot 1\,\mathrm{s} = 15\,\mathrm{m} \qquad\qquad H(5\,\mathrm{s}) = 30\,\frac{\mathrm{m}}{\mathrm{s}} \cdot 5\,\mathrm{s} = 75\,\mathrm{m} \tag{4.2}$$

Inserting these values in equation 4.1 results in the positive and negative gust shapes shown in figure 4.1. The smaller H is, the faster the gust load alleviation system has to respond.

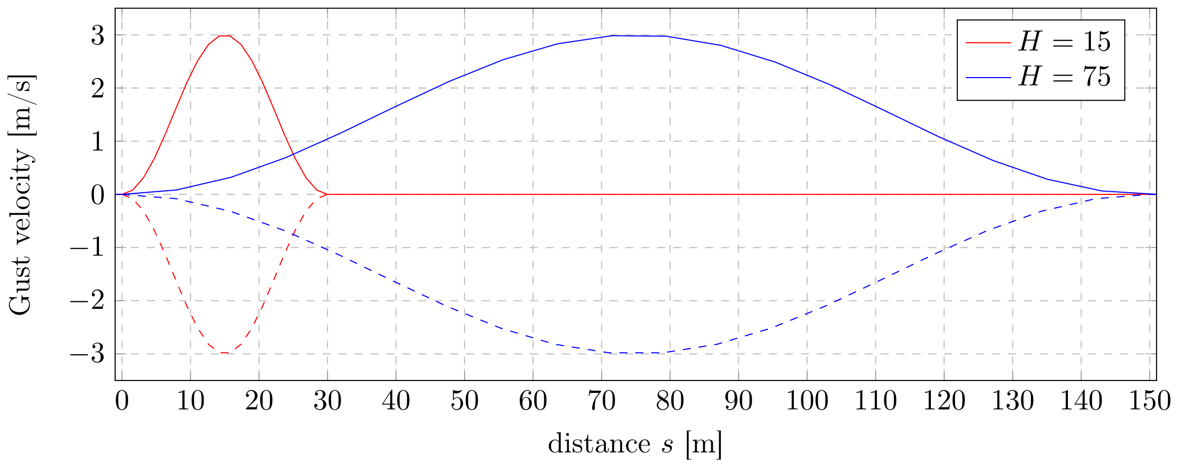


Figure 4.1. Positive (solid) and negative (dashed) gust shapes with $H = 1$ and $H = 5$

4.2 Horizontal Gusts

Generally, the lift force can be calculated with the following formula:

$$F_L = C_L \, \frac{\rho v^2}{2} A \tag{4.3}$$

The velocity v is the velocity opposed to the direction of flight and can therefore be expressed as

$$v = U_{eq} + v_{x,gust}(t) \tag{4.4}$$

where $U_{eq} = 30\,\mathrm{m/s}$ and $v_{x,gust}(t)$ is the horizontal component of the gust, which is a function of time and calculated using equation 4.1 for the gust shape, where $s = U_{eq} \cdot t$. The effect of horizontal gusts on the lift force is thus given by the following expression:

$$F_L = C_L \, \frac{\rho (U_{eq} + v_{x,gust}(t))^2}{2} A \tag{4.5}$$

4.3 Vertical Gusts

In the case of a vertical gust, the effective angle of attack α changes, which affects the lift coefficient. Therefore, steady CFD simulations were run for different angles of attack. The different angles were simulated by rotating the inlet velocity vector as well as the lift and drag direction vectors accordingly. The angle was varied in the range from $-2°$ to $15°$ in steps of $1°$. The angle at which the aerofoil produces zero lift was found to be at $-1.5°$. Above $8°$, the number of iterations required for convergence increased approximately by a factor of 3 and small oscillations of C_L were observed. Above $15°$, the solution did not converge. The results are shown in figure 4.2.

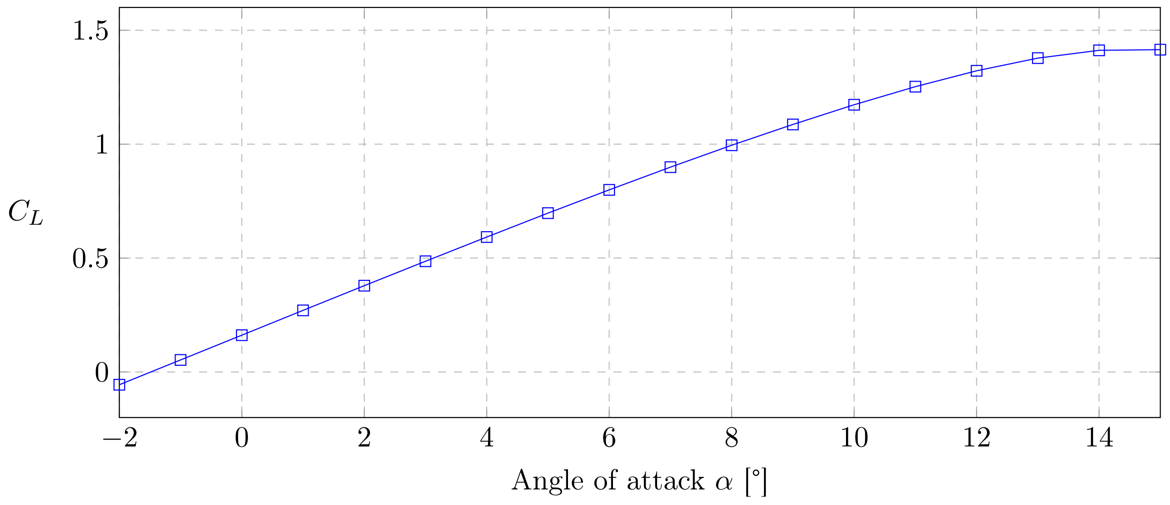


Figure 4.2. Lift coefficient at different angles of attack

The unreliable behaviour of the CFD model at larger angles of attack is attributed to the increasing unsteady effects, which can not be captured by the steady simulation. Furthermore, the model has only been validated for $\alpha = 0°$. However, for relatively small angles of attack, the CFD model is still assumed to be accurate enough.

4.4 Optimum Lift to Drag Ratio

Ideally, the wing would operate at its optimum lift to drag ratio, to achieve the best aerodynamic efficiency. The lift to drag ratio was calculated using the results for C_L from section 4.3 and dividing them by the corresponding drag coefficient. Figure 4.3 shows the lift to drag ratio at different angles of attack.

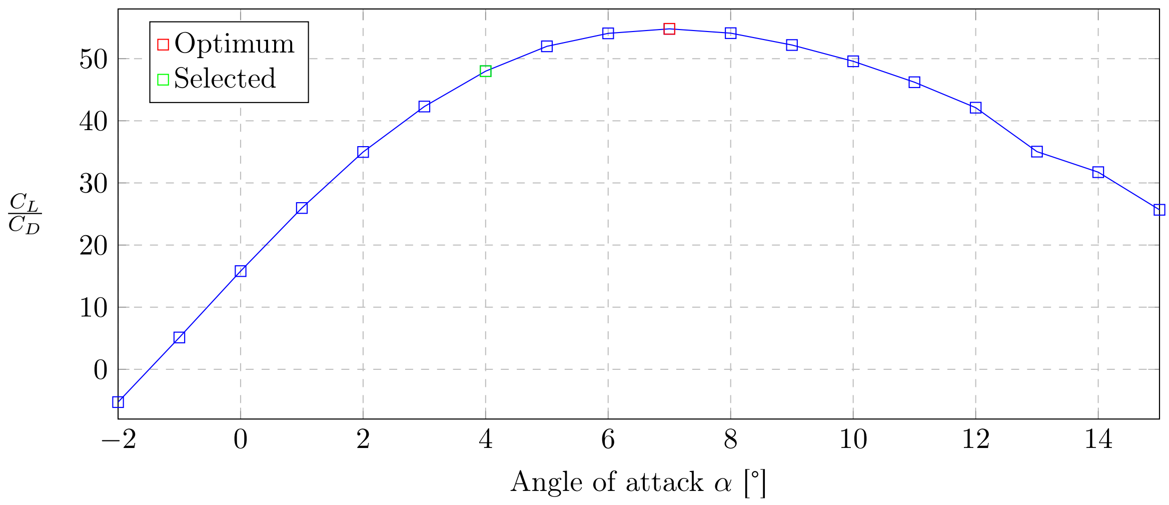


Figure 4.3. Lift to drag ratio at different angles of attack

It was found that the optimum lift to drag ratio is 54.8 at an angle of attack of 7°. However, positive vertical gusts would increase this angle of attack further to the point were the CFD model has shown oscillatory and unreliable behaviour. Therefore, the aerofoil is chosen to operate at an angle of 4°, with a lift to drag ratio of 48.0. This is still relatively close to the optimum, but leaves room for modelling positive and negative vertical gusts. At this angle of attack, the forces of gravity and lift are assumed to be at equilibrium, if no gusts are present. The lift coefficient at the chosen equilibrium point is $C_{L,eq} = 0.59227$.

4.5 Reduced Order Model for Vertical Gusts

To be able to predict the effects of vertical gusts in the gust load alleviation system, a reduced order model is developed. Up to $\alpha = 8°$, the relationship between lift coefficient and angle of attack in figure 4.2 is well approximated by a linear function. The general expression for for C_L as a function of α is then:

$$C_L(\alpha) = s \cdot \alpha + b \tag{4.6}$$

The function is chosen to be exact at the equilibrium point $\alpha = 4°$, thus:

$$C_L(\alpha = 4°) = s \cdot 4° + b = C_{L,eq} \tag{4.7}$$

The slope s is calculated using the values at 2° and 6° as follows:

$$s = \frac{C_L(6°) - C_L(2°)}{6° - 2°} = \frac{0.79939 - 0.37878}{4°} = \frac{0.10515}{1°} \tag{4.8}$$

The constant b is then found as by inserting equation 4.8 into equation 4.7 and rearranging:

$$b = 0.59227 - \frac{0.10515}{1^\circ} \cdot 4^\circ = 0.17167 \tag{4.9}$$

The final reduced order model is then given by:

$$C_L(\alpha) = \frac{0.10515}{1^\circ} \cdot \alpha + 0.17167 \tag{4.10}$$

As can be seen in figure 4.4, the linear approximation becomes increasingly inaccurate at angles above 8°. However, the CFD model is not well suited for simulating larger angles of attack either. Therefore, any simulation results for gust cases with angles above 8° should be treated with care and can generally not be relied upon.

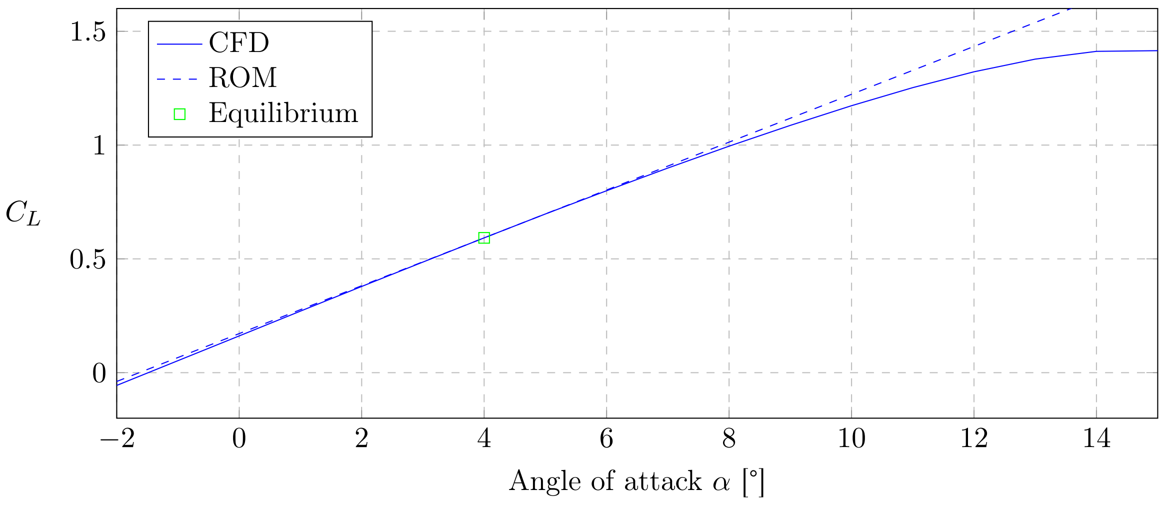


Figure 4.4. Reduced order model for lift coefficient at different angles of attack

The effective angle of attack during a vertical gust is a function of time and calculated as follows:

$$\alpha(t) = 4^\circ + \arctan\left(\frac{v_{y,gust}(t)}{U_{eq}}\right)\frac{180^\circ}{\pi} \tag{4.11}$$

where $v_{y,gust}(t)$ is the vertical gust velocity and calculated using equation 4.1. Using equation 4.11, the effective angle of attack is 9.71°, when the positive vertical gust reaches its peak, which is where the CFD model is not very reliable. This gust case will still be simulated, but the results should be viewed with caution. When the negative vertical gust reaches its peak, the effective angle of attack is −1.71°, which is very close to the validation case.

Chapter 5
Trailing Edge Flap Modelling

In order to alleviate the gust loads, a trailing edge flap is used. Thus, a model is developed, which can accurately predict how the deflection angle of the flap affects the lift force. In this way, the required flap deflection angle to counteract a given gust load can be calculated.

5.1 Flap Geometry

Usually, the flap is a separate part of an aerofoil. However, here it is simply modelled by deforming the aerofoil geometry. This reduces the computational complexity and makes it possible to keep using the structured mesh the CFD-model was validated with. The length of the trailing edge flap is chosen to be $0.15c$. The flap rotates around its starting point at $x = 0.85c$ on the chord line ($y = 0$), resulting in sharp edges in the aerofoil geometry at the beginning of the flap. Also, the flap becomes slightly thinner with increasing deflection angle. An illustrative example of the deformed aerofoil geometry with flap deflection angles of $+45°$ and $-45°$ is shown in figure 5.1

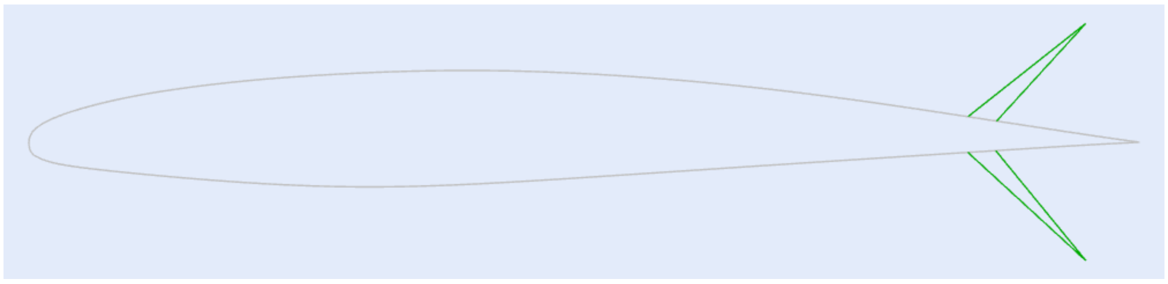


Figure 5.1. Flap Geometry

5.2 Deforming Mesh

As the flap moves up or down, the mesh has to be deformed accordingly. This is achieved with a User Defined Function (UDF) in Ansys Fluent. The UDF loops over all faces of the mesh and adjusts the position of every node as needed. The boundary nodes as well as the nodes left to where the flap begins, i.e. $x < 0.85c$, remain untouched. All other nodes are shifted by an amount Δx and Δy, based on how the deflection angle θ has changed compared to the previous time step. For a given previous flap angle θ^{n-1} and a new flap angle θ^n, the position of the trailing edge changes by

$$\Delta x = (\cos\theta^n - \cos\theta^{n-1}) \cdot 0.15c \tag{5.1}$$

$$\Delta y = (\sin\theta^n - \sin\theta^{n-1}) \cdot 0.15c \tag{5.2}$$

All nodes in the wake, i.e. $x > c$, are moved by the same amount as the trailing edge, which means that the relative position between wake nodes does not change. The remaining nodes in the flap region, i.e. $0.85c < x < c$, are moved by a fraction of the trailing edge movement, depending on their distance to the rotation point, as given by:

$$\Delta x_{flap} = \frac{x_{node} - 0.85c}{0.15c} \cdot \Delta x \tag{5.3}$$

$$\Delta y_{flap} = \frac{x_{node} - 0.85c}{0.15c} \cdot \Delta y \tag{5.4}$$

No remeshing, layering or smoothing is necessary, as every grid node is controlled directly through the UDF. The source code for the UDF and additional details on the implementation can be found in Appendix A. The deformed mesh around the flap at a deflection angle of 20° and −20° is shown in figure 5.2 and 5.3, respectively.

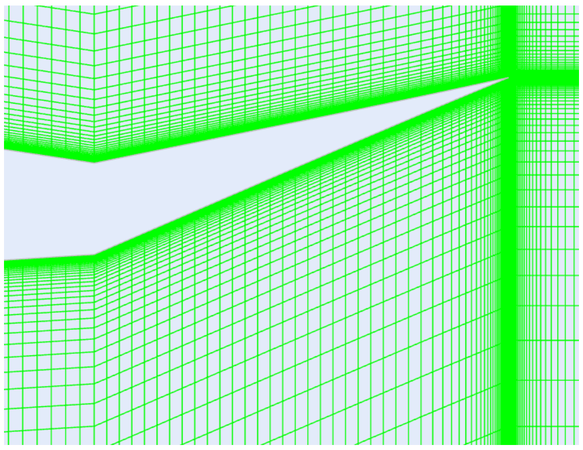


Figure 5.2. Deformed mesh around the flap at $\theta = 20°$

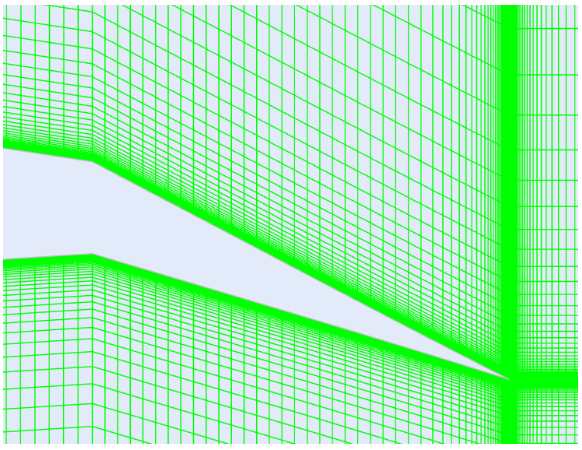


Figure 5.3. Deformed mesh around the flap at $\theta = -20°$

5.3 Flap CFD Simulation

To investigate the effect of the trailing edge flap on the lift coefficient, steady CFD simulations were run for a range of different angles of attack and flap deflection angles. The velocity was always 30 m/s and the deflection angle was varied from +10° to −10° in steps of 2°. It should be noted that the steady simulation cannot account for the dynamic effects on the lift force, which are caused by the motion of the trailing edge flap. However, this effect depends on the angular velocity of the flap, which in turn depends on the gust gradient. Thus, the steady simulation of a given flap deflection angle is considered a good reference. The results are shown in figure 5.4.

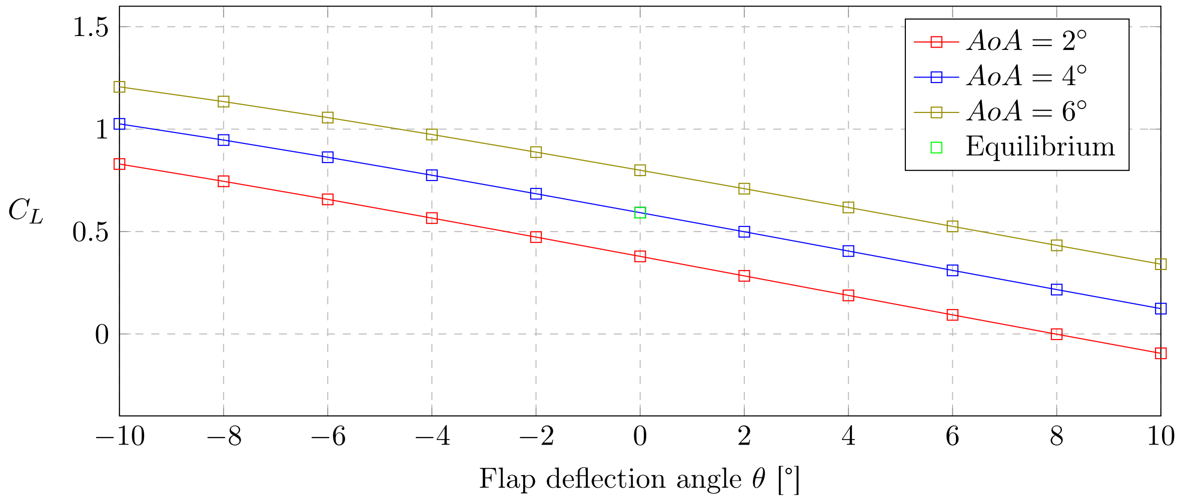


Figure 5.4. Lift coefficient at different flap deflection angles

5.4 Flap Reduced Order Model

In the gust load alleviation control loop, a reduced order model is used to calculate the required flap deflection angle. From visual inspection of figure 5.4, it can be seen that the curves are almost straight lines and can therefore be approximated well by linear functions. The reduced order model is exact at the chosen equilibrium point, i.e. an angle of attack of 4° and a flap deflection angle of 0°. The slope is calculated from the results at $\theta = 10°$ and $\theta = -10°$, which yields:

$$C_L(\theta) = \frac{C_L(10°) - C_L(-10°)}{20°}\theta + C_{L,eq} = \frac{0.1237 - 1.0256}{20°}\theta + C_{L,eq} = \frac{-0.0451}{1°}\theta + C_{L,eq} \quad (5.5)$$

The other curves can also be represented by the model, if $C_{L,eq}$ is replaced by the reduced order model for $C_L(\alpha)$ from equation 4.10. The complete reduced order model is then given by the following equation, which expresses C_L as a function of α and θ:

$$C_L(\alpha, \theta) = \frac{-0.0451}{1°}\theta + \frac{0.10515}{1°}\alpha + 0.17167 \quad (5.6)$$

In figure 5.5, the predictions for C_L from the reduced order model are shown as dashed lines along with the CFD results as solid lines. Generally, the reduced order model is very close to the CFD model, with deviations becoming slightly larger with increasing flap deflection. At higher angles of attack, the reduced order model tends to overpredict increases in C_L, while it tends to underpredict decreases in C_L at lower angles of attack.

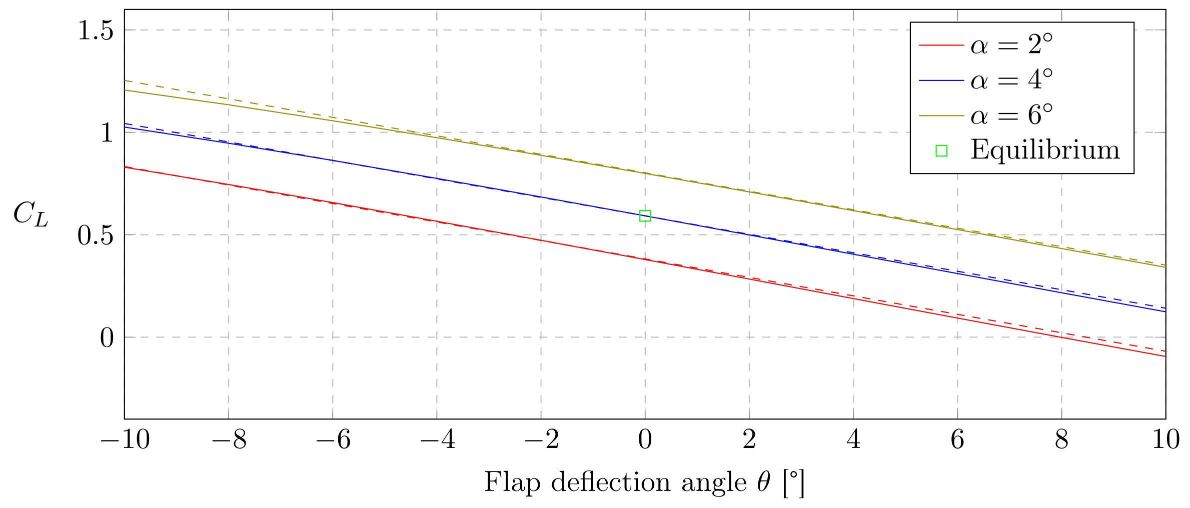


Figure 5.5. C_L at different flap deflection angles with reduced order model (dashed) and CFD model (solid)

Chapter 6
Gust Load Alleviation System

In this chapter, the gust load alleviation system will be developed based on the reduced order models derived in the previous sections. It will then be tested against the gust cases, that were described earlier.

6.1 Required Flap Deflection Angle

Ideally, the gust load alleviation system keeps the lift force equal to the equilibrium lift force during the gust. Thus, the lift force at equilibrium conditions must be equal to the lift force resulting from the combined effect of gust velocity, direction and flap deflection. This equality can be expressed using expression 4.3:

$$C_{L,eq}\,\frac{\rho U_{eq}^2}{2}A = C_L(\theta,\alpha)\,\frac{\rho(U_{eq}+v_{x,gust}(t))^2}{2}A \tag{6.1}$$

ρ and A are constants and cancel out, which yields:

$$C_{L,eq} = C_L(\theta,\alpha)\,\left(\frac{U_{eq}+v_{x,gust}(t)}{U_{eq}}\right)^2 \tag{6.2}$$

The reduced order model for $C_L(\theta,\alpha)$ from equation 5.6 is then inserted into the above equation.

$$C_{L,eq} = \left(\underbrace{\frac{-0.0451}{1^\circ}\theta}_{\text{flap deflection}} + \underbrace{\frac{0.10514}{1^\circ}\alpha + 0.17167}_{\text{vertical gust}}\right)\cdot\underbrace{\left(\frac{U_{eq}+v_{x,gust}(t)}{U_{eq}}\right)^2}_{\text{horizontal gust}} \tag{6.3}$$

The different terms that arise from the flap deflection and the vertical and horizontal gust velocity have been indicated above. Their combined contributions must ideally result in the same value as the lift coefficient at equilibrium. To achieve this, the equation is solved for the required flap deflection angle θ, which yields:

$$\theta = \frac{1^\circ}{0.0451}\left(\frac{0.10515}{1^\circ}\alpha + 0.17167 - C_{L,eq}\left(\frac{U_{eq}}{U_{eq}+v_{x,gust}(t)}\right)^2\right) \tag{6.4}$$

In the case of a horizontal gust, the equation can be simplified, as $C_L(\alpha = 4^\circ) = C_{L,eq}$:

$$\theta = \frac{1^\circ}{0.0451}\left(C_{L,eq}\left(1-\left(\frac{U_{eq}}{U_{eq}+v_{x,gust}(t)}\right)^2\right)\right) \tag{6.5}$$

During a vertical gust, $v_{x,gust}(t) = 0$ and the equation simplifies to:

$$\theta = \frac{1^\circ}{0.0451}\left(\frac{0.10515}{1^\circ}\alpha + 0.17167 - C_{L,eq}\right) \tag{6.6}$$

6.2 Gust Load Alleviation Tests

In order to test performance of the gust load alleviation system, the previously described gust cases were simulated using the CFD model. User Defined Functions for the x- and y-component of the inlet velocity were hooked to Ansys Fluent to simulate the gusts. The reduced order model for the required flap deflection angle was incorporated into the UDF for the dynamic mesh. The full source code and additional information on the UDF implementation can be found in Appendix A.

The time step for the transient CFD simulations must be small enough to properly resolve the gust shapes of the short gusts, which have large velocity gradients. Initial tests showed that a time step of $\Delta t = 0.05\,\mathrm{s}$ was sufficiently accurate. Smaller time steps did not alter the results, except for a slight smoothing of the lift force curves. First, the flow field was initialised with a well converged steady solution at equilibrium. Then, the transient simulation was started, maintaining the equilibrium state for the first second. The lift force at equilibrium was not affected by the switch from steady to transient calculation. After one second, the gust started and at the end of the gust, further 3 seconds where simulated, to ensure that the solution was again settled at equilibrium. Two simulations were run for each case, one without GLA, where the dynamic mesh was disabled and trailing edge flap did not move, and the other with activated GLA, i.e. moving flap and dynamic mesh. In the following figures, the lift force prediction from the reduced order model (ROM) is also shown for comparison with the CFD simulation.

When comparing the figures for vertical and horizontal gusts, it should be noted that peak change in lift force without GLA is about 40 N for a horizontal and about 200 N for a vertical gust, even though they visually appear with approximately the same size.

6.2.1 Short Horizontal Gust Test

The results for the short horizontal gust case are shown in figure 6.1 and 6.2. It is observed that the reduced order model slightly overpredicts the positive gust peak and underpredicts the negative gust peak. Furthermore, the lift curve is slightly shifted to the right. At the end of the gust, the lift force in the CFD model takes about 0.5 s to settle at equilibrium, which by definition is not predicted by the reduced order model. With activated GLA, the lift force stays very close to equilibrium, with a slight undershoot and overshoot during the positive and negative gust, respectively. In both cases, the peak of the overshoot and undershoot is reached shortly after the peak of the lift force and the undershoot is slightly less pronounced than the overshoot.

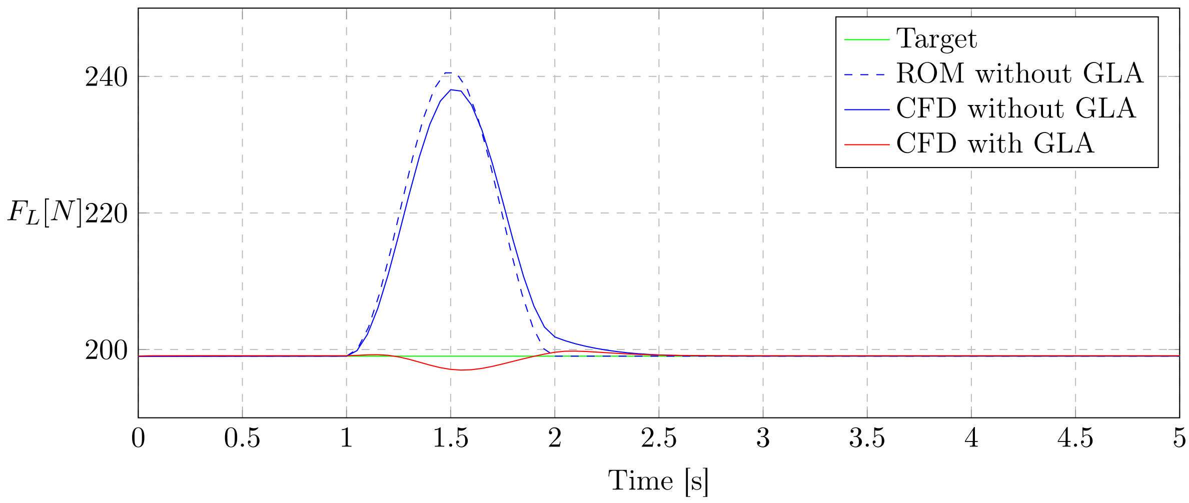


Figure 6.1. Short Positive Horizontal Gust ($H = 15\,\text{m}$)

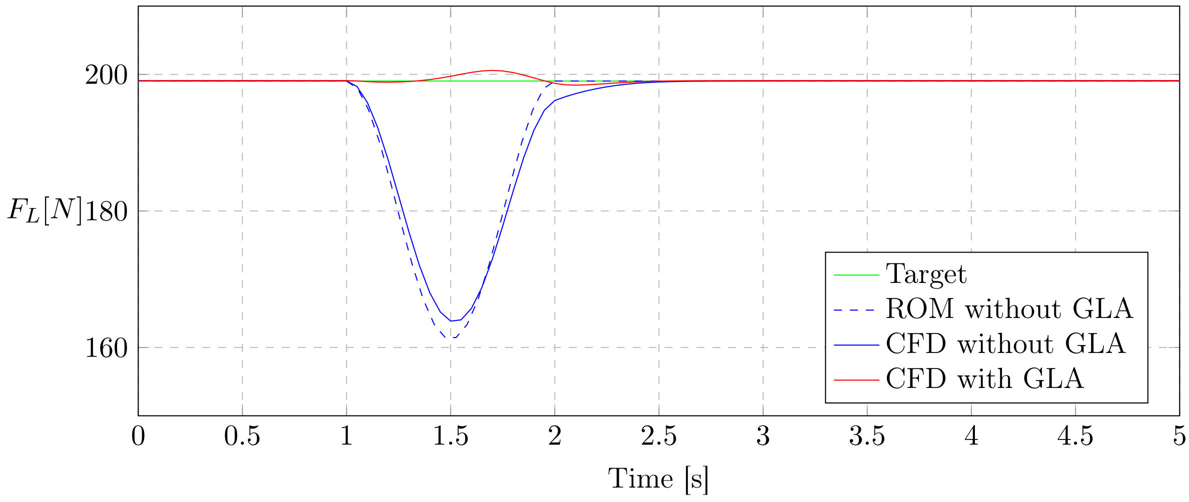


Figure 6.2. Short Negative Horizontal Gust ($H = 15\,\text{m}$)

6.2.2 Long Horizontal Gust Test

The long horizontal gust case is shown in figure 6.3 and 6.4. In contrast to the short horizontal gust, the agreement between CFD and reduced oder model is almost perfect and the lift force is settled at equilibrium at the end of the gust as predicted by the ROM. However, similar to the short horizontal gust, the right shift of the CFD lift curve is still present, albeit much smaller. The GLA system also produces a slight over- and undershoot, which is smaller than in the shorter gust case and reaches its peak at about the same time as the lift force.

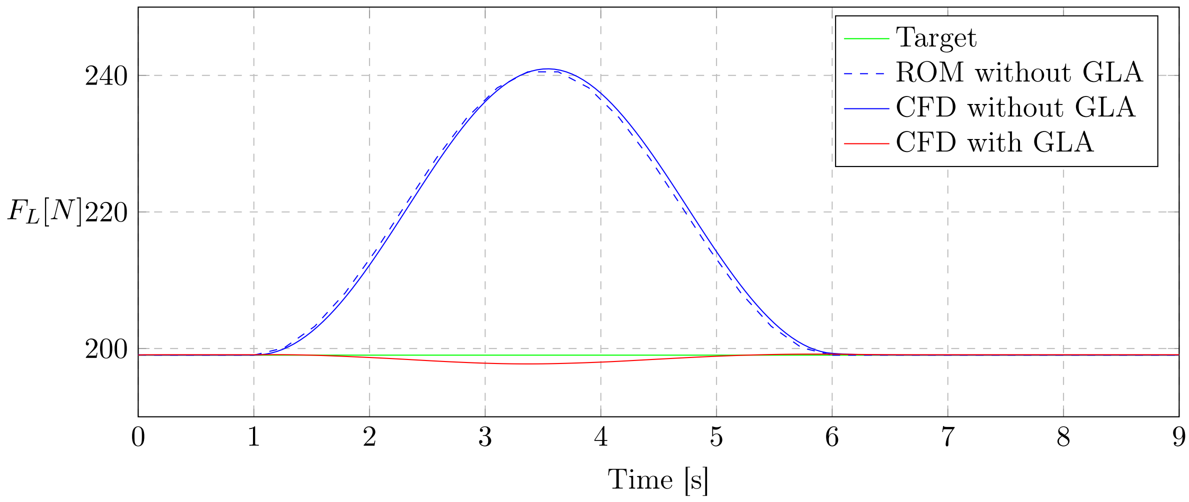


Figure 6.3. Long Positive Horizontal Gust ($H = 75\,\mathrm{m}$)

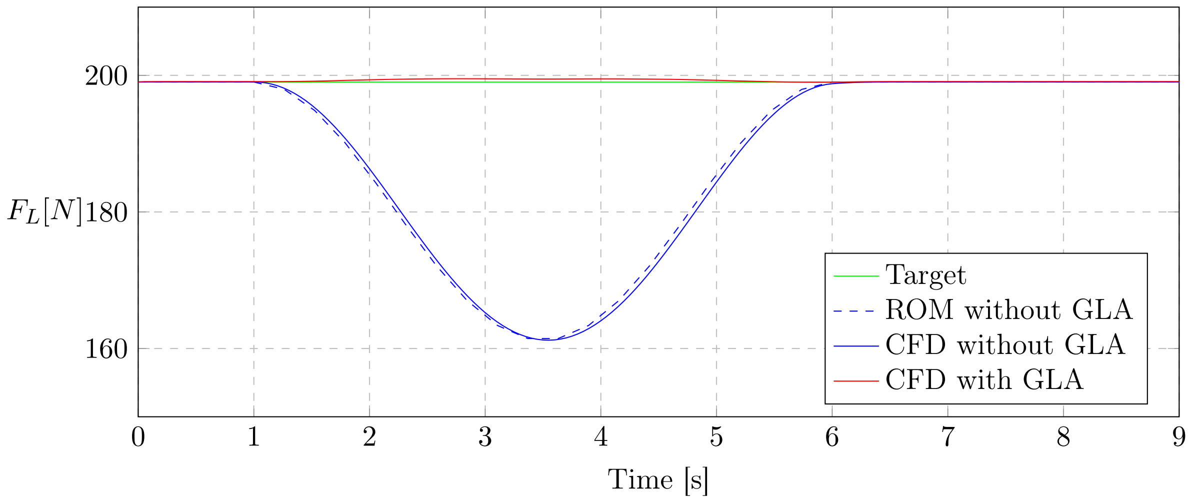


Figure 6.4. Long Negative Horizontal Gust ($H = 75\,\mathrm{m}$)

6.2.3 Short Vertical Gust Test

The results for the short vertical gust case are shown in figure 6.5 and 6.6. The same previously seen disagreement between ROM and CFD is quite large in these two cases. The lift force in the CFD model is settled at equilibrium about 0.7 s after the gust. However, the GLA system generally still alleviates most of the gust load. In the positive gust case, the GLA produces an undershoot with its peak shortly before the lift force peak, which is followed by an overshoot with its peak shortly before the end of the gust. For the negative gust, the GLA first produces an overshoot followed by an undershoot with the same peak times as in the positive case.

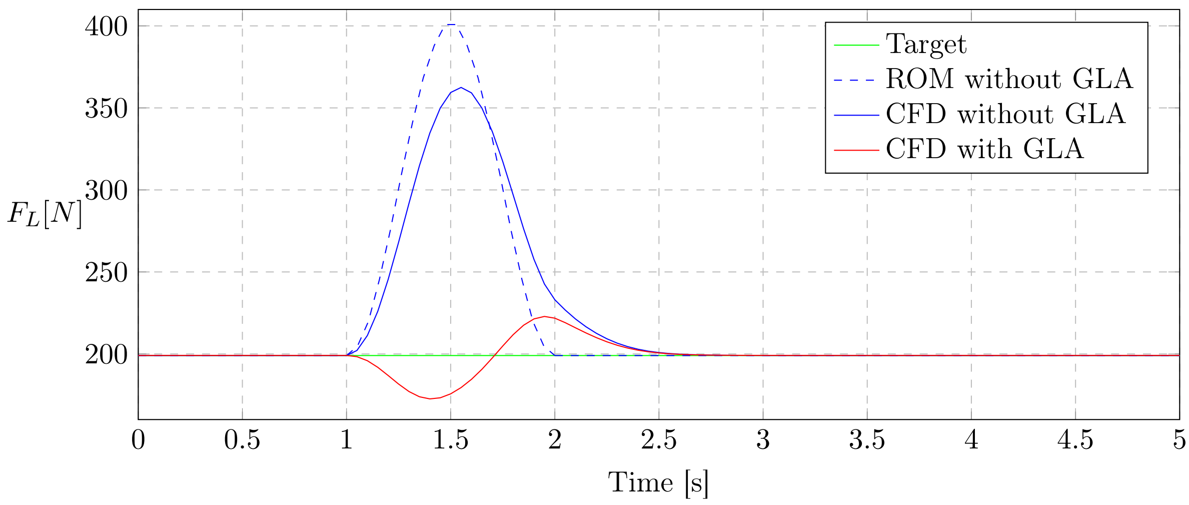


Figure 6.5. Short Positive Vertical Gust ($H = 15\,\text{m}$)

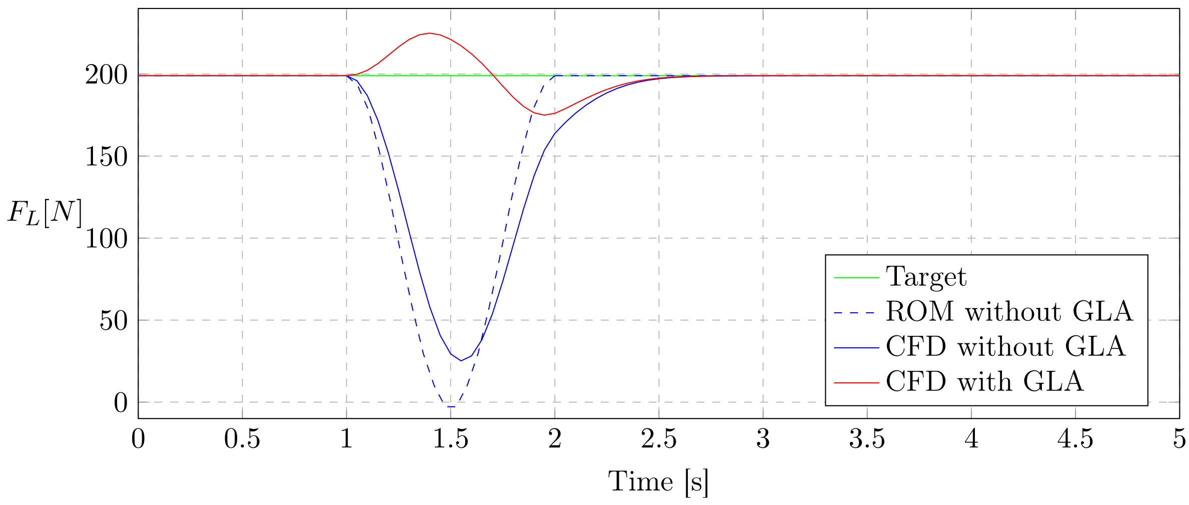


Figure 6.6. Short Negative Vertical Gust ($H = 15\,\text{m}$)

6.2.4 Long Vertical Gust Test

The long vertical gust case is shown in figure 6.7 and 6.8. Generally, the characteristics are similar to the short vertical gust case, albeit much smaller. However, one major difference between the long negative and positive vertical gust is observed. During the positive gust, the ROM overpredicts the lift force, whereas the CFD model overpredicts the lift force in the negative case.

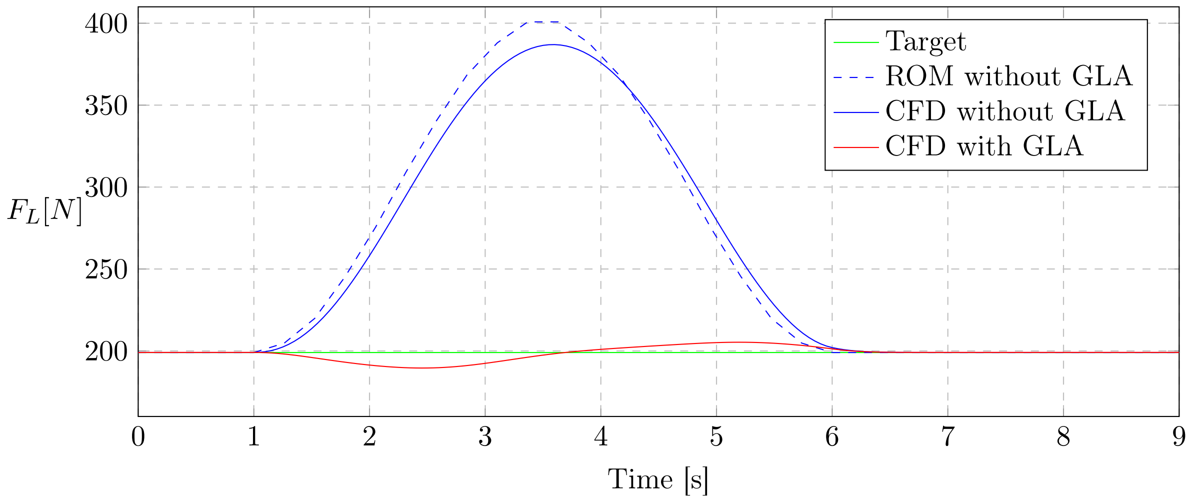


Figure 6.7. Long Positive Vertical Gust ($H = 75\,\mathrm{m}$)

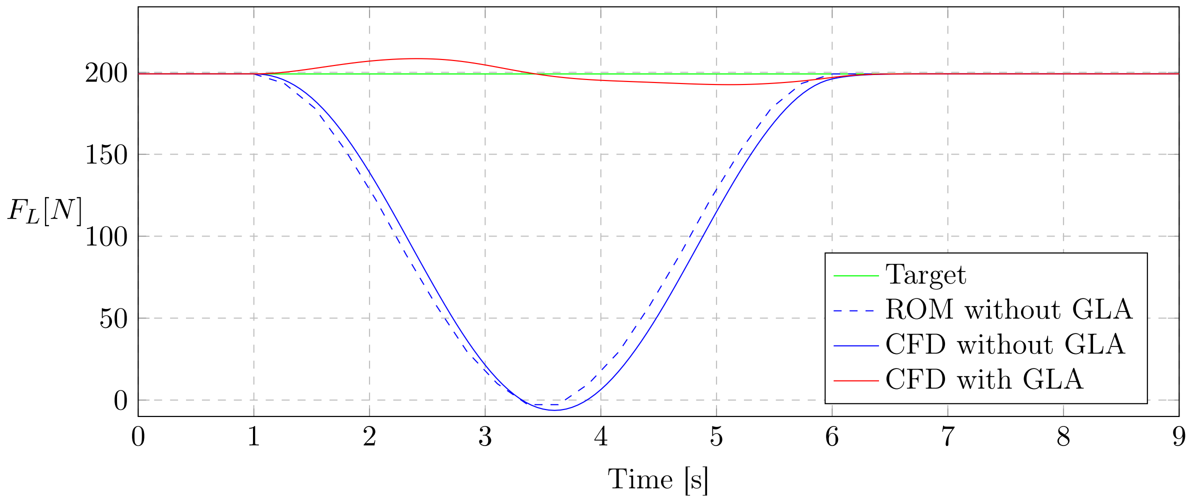


Figure 6.8. Long Negative Vertical Gust ($H = 75\,\mathrm{m}$)

Chapter 7 Discussion

CFD Model
The initial mesh had the desired y^+-value and produced very good results for C_L and C_D, which could not be improved by refinement or extending the far-field. Sharpening the trailing edge with the tangent intersection method reduced the computational complexity, while maintaining effectively the same accuracy as with the blunt trailing edge. The initial local mesh refinement in horizontal direction around the trailing edge proved to be essential for the accuracy of C_L and C_D, whereas refinement at the leading edge did not improve the results.

Model Validation
The CFD models agreement with experimental surface pressure data and boundary layer velocity profiles was generally very good. One reason for the small differences might be that the tripping of the boundary layer was not included in the CFD model and the flow was simulated as fully turbulent instead. However, the tripping cannot explain the large differences in the wake velocity profiles. As these disagreements were also observed by others, even with finer meshes, this might lead to the conclusion that these effects cannot be accurately modelled with steady RANS-based CFD. However, as the wake velocity profiles are not important for an accurate prediction of lift and drag, the CFD model can be considered sufficiently accurate for the purposes of this study.

ROM and GLA
In most cases, the over- and undershoot of the GLA is mostly due to the over- and under prediction of the gust induced lift force by the ROM. This over- and under prediction is mostly a transient phenomenon, as it is also observed in cases were the steady ROM is in perfect agreement with the steady CFD. In almost all cases the positive and negative versions of a given gust case show close to symmetrical behaviour. Only during the long positive vertical gust the overprediciton by the ROM is still quite significant, while the ROM and CFD are very close in the negative case. Therefore, this overprediction in the positive case is most likely due to the fact that the ROM is not in good agreement with the steady CFD, whereas it is in the negative case. However, as previously mentioned, during the positive vertical gust, the angle of attack reaches $9.71°$ at the peak, where the CFD model has shown unreliable behaviour and the results should be viewed with special caution. The general observation that the gust lift curve from the CFD model was shifted to the right, must also be a transient phenomenon. The dynamic effect of the trailing edge flap movement was not significant in most cases.

Chapter 8
Conclusion

The CFD model predicts lift and drag accurately, as long as the angle of attack is relatively close to zero. The validation showed good agreement with experimental surface pressure data and the boundary layer velocity profiles. The disagreement between CFD and experiment in the wake is not considered relevant for C_L and C_D. At larger angles of attack the CFD model becomes increasingly unreliable, as the model has only been validated for $\alpha = 0°$.

The ROM is only exact at the chosen equilibrium point. The differences between ROM and CFD model are mostly due to transient behaviour, which is not captured by the ROM. Steady differences between ROM and CFD are significant at $\alpha > 8°$.

It is concluded that the GLA system generally works well, but has some difficulty with large gust gradients. In all cases, small under- or overshoots were observed, which were more pronounced during short gusts, positive gusts, and vertical gusts. However, even in the worst cases, the GLA system alleviated approximately 80 % of the gust load.

Chapter 9
Future Work

The accuracy of the CFD model could be improved further, for a better prediction of the velocity field in the wake. The validation could also be extended to cover a range of different angles of attack. However, such experimental data is not yet available for the Nakayama aerofoil.

The ROM could be improved to also include non-linear and transient flow effects, which would result in a better prediction of gust effects and could thereby improve the GLA systems performance.

Finally, the problem could be extended to three dimensions to account for finite wings and other parts of the aeroplane. The structural dynamic response of the wings could be investigated, and the shape of the trailing edge flap could be aerodynamically optimised. The length of the flap could also be adjusted, to fit the requirements or limitations of possible actuators.

Bibliography

Alulema et al., 2020. Victor Alulema, Esteban Valencia Torres, Danilo Pillajo, Monica Jacome, Juan López and Byron Ayala. *Degree of Deformation and Power Consumption of Compliant and Rigid-linked Mechanisms for Variable-Camber Morphing Wing UAVs*. 2020.

Andersen, 2010. P. B. Andersen. *Advanced Load Alleviation for Wind Turbines using Adaptive Trailing Edge Flaps: Sensoring and Control*. Risø National Laboratory for Sustainable Energy. Risø-PhD No. 61(EN), 2010.

Ansys, 2022. Ansys. *Ansys Fluent Theory Guide*. 2022.

Cadence Design Systems, 2024. Cadence Design Systems. *y^+-Calculator*, 2024. URL `https://www.cadence.com/en_US/home/tools/system-analysis/computational-fluid-dynamics/y-plus.html`. Accessed on 11.02.2024.

EASA, 2007. EASA. *Certification Specifications for Large Aeroplanes CS-25*. 2007.

Erkan and Özkan, 2020. Onur Erkan and Musa Özkan. *Investigation of the Flow Over NACA 63-415 Airfoil*. Black Sea Journal of Engineering and Science, 3, 2020.

Li and Qin, 2022. Yonghong Li and Ning Qin. *A Review of Flow Control for Gust Load Alleviation*. Applied Sciences, 12, 2022.

Lu et al., 2021. S. Lu, J. Liu and R Hekkenberg. *Mesh properties for RANS simulations of airfoil-shaped profiles: A case study of rudder hydrodynamics*. Journal of Marine Science and Engineering, 9(10), 2021.

Nakayama, 1985. A. Nakayama. *Characteristics of the flow around conventional and supercritical aerofoils*. J. Fluid. Mech, 160, 1985.

Narsipur et al., 2012. S. Narsipur, B. Pomeroy and M. Selig. *CFD Analysis of Multielement Airfoils for Wind Turbines*. 30th AIAA Applied Aerodynamics Conference, 2012.

Nordangera et al., 2014. K. Nordangera, R. Holdahlb, A. Kvarvingb, Adil. Rasheedb and T. Kvamsdal. *Implementation and comparison of three isogeometric Navier–Stokes solvers applied to simulation of flow past a fixed 2D NACA0012 airfoil at high Reynolds number*. Computer Methods in Applied Mechanics and Engineering, 284, 2014.

Oh and Choi, 1997. C. S. Oh and D. H. Choi. *An Improved Navier-Stokes Procedure for Analysis of Two-Dimensional Aerofoils in Laminar and Turbulent Flows*. International Journal for Numerical Methods in Fluids, 25, 1997.

Polewski and Cizmas, 2014. M. Polewski and P. Cizmas. *Several Cases for the Validation of Turbulence Models Implementation*. Applied Sciences, 11, 2014.

Rumsey, 2019. Christopher Rumsey. *Grids*, 2019. URL `https://turbmodels.larc.nasa.gov/airfoilwake_grids.html`. Accessed on 10.12.2023.

Rumsey, **2021**. Christopher Rumsey. *Airfoil Verification*, 2021. URL `https://turbmodels.larc.nasa.gov/airfoilwakeverif.html`. Accessed on 05.02.2024.

Rumsey, **2023a**. Christopher Rumsey. *SA Results (20c)*, 2023a. URL `https://turbmodels.larc.nasa.gov/airfoilwakeverif_sa.html`. Accessed on 21.01.2023.

Rumsey, **2023b**. Christopher Rumsey. *SA Results (500c)*, 2023b. URL `https://turbmodels.larc.nasa.gov/airfoilwakeverif500c_sa.html`. Accessed on 10.12.2023.

Spalart and Venkatakrishnan, **2016**. P. Spalart and V. Venkatakrishnan. *On the role and challenges of CFD in the aerospace industry.* The Aeronautical Journal, 120, 2016.

Tabatabaei et al., **2022**. N. Tabatabaei, R. Vinuesa, R. Örlü and P. Schlatter. *Techniques for Turbulence Tripping of Boundary Layers in RANS Simulations.* Flow, Turbulence and Combustion, 108, 2022.

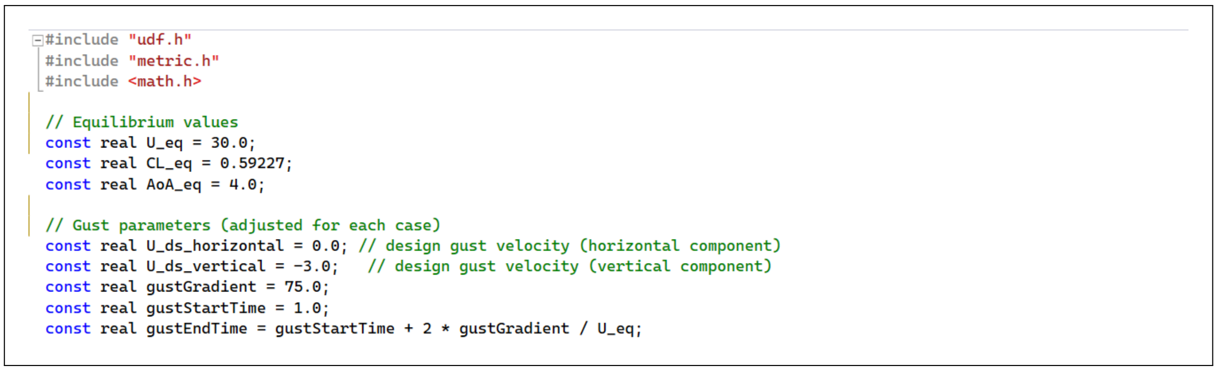
Versteeg and Malalasekera, **2007**. H. Versteeg and W. Malalasekera. *An Introduction to Computational Fluid Dynamics. The Finite Volume Method.* Pearson Education Limited, 2nd edition, 2007. ISBN 978-0-13-127498-3.

Appendix A
User Defined Functions

In the following sections, the User Defined Functions, that have been used to model the GLA system, are described. It should be noted that all UDFs were written in the same .c-file, so they could easily share global variables and functions. This file was recompiled before each gust case with some adjusted parameters. The code includes some comments to facilitate the understanding. It should be noted that the original code file included many lines with the *Message* command, which provided feedback to the user during the simulation. To improve the code readability, these lines have been left out here, as they are not essential for the code to work.

A.1 Global Variables and Functions

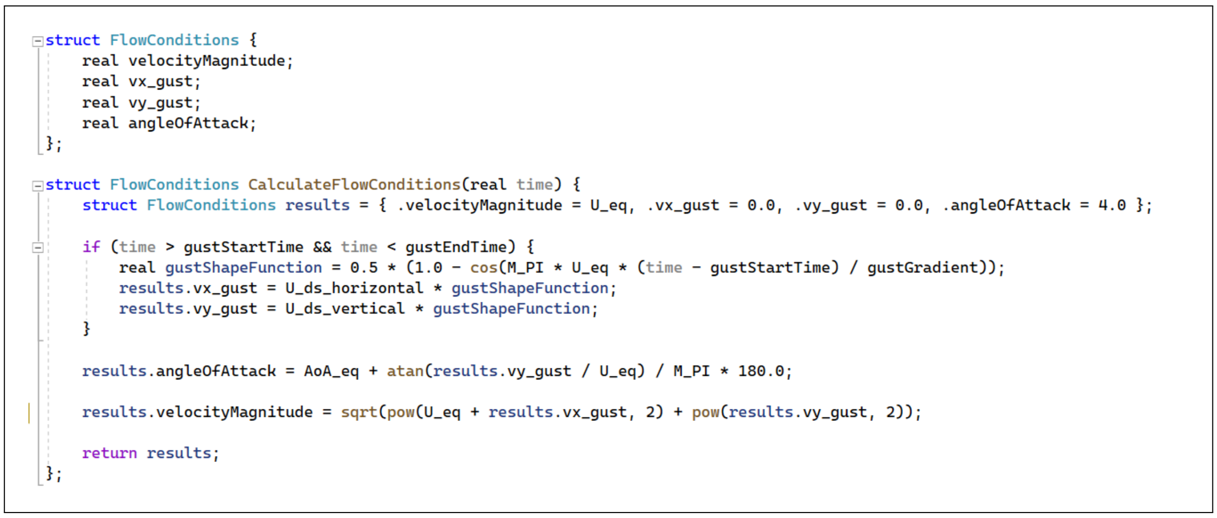
The included header files and the constant variables, which are known at compile time and do not change during the simulation, are shown in figure A.1.



```c
#include "udf.h"
#include "metric.h"
#include <math.h>

// Equilibrium values
const real U_eq = 30.0;
const real CL_eq = 0.59227;
const real AoA_eq = 4.0;

// Gust parameters (adjusted for each case)
const real U_ds_horizontal = 0.0; // design gust velocity (horizontal component)
const real U_ds_vertical = -3.0;   // design gust velocity (vertical component)
const real gustGradient = 75.0;
const real gustStartTime = 1.0;
const real gustEndTime = gustStartTime + 2 * gustGradient / U_eq;
```

Figure A.1. Header files and constant variables

The struct *FlowConditions* contains all relevant in formation about a gust at a given time. It is calculated using the *CalculateFlowConditions* function as shown in figure A.2

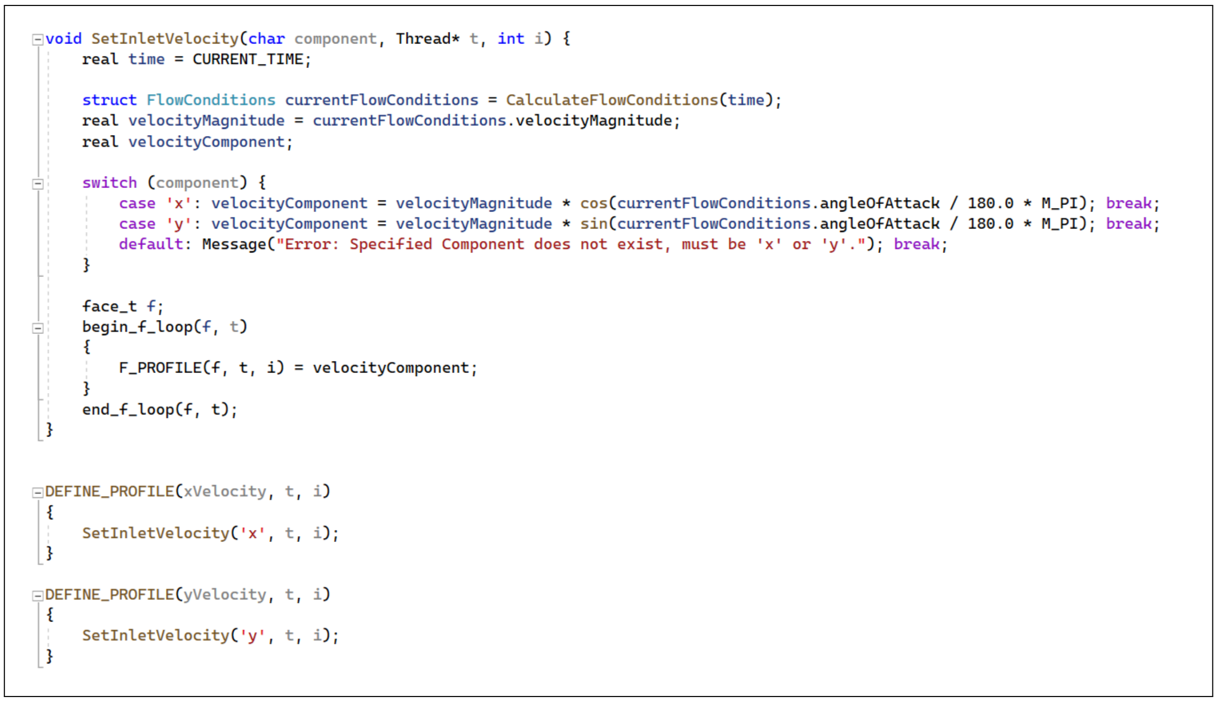


```c
struct FlowConditions {
    real velocityMagnitude;
    real vx_gust;
    real vy_gust;
    real angleOfAttack;
};

struct FlowConditions CalculateFlowConditions(real time) {
    struct FlowConditions results = { .velocityMagnitude = U_eq, .vx_gust = 0.0, .vy_gust = 0.0, .angleOfAttack = 4.0 };

    if (time > gustStartTime && time < gustEndTime) {
        real gustShapeFunction = 0.5 * (1.0 - cos(M_PI * U_eq * (time - gustStartTime) / gustGradient));
        results.vx_gust = U_ds_horizontal * gustShapeFunction;
        results.vy_gust = U_ds_vertical * gustShapeFunction;
    }

    results.angleOfAttack = AoA_eq + atan(results.vy_gust / U_eq) / M_PI * 180.0;

    results.velocityMagnitude = sqrt(pow(U_eq + results.vx_gust, 2) + pow(results.vy_gust, 2));

    return results;
};
```

Figure A.2. Flow Conditions Calculation

A.2 UDFs for Varying Inlet Velocity

Two UDFs using the *DEFINE_PROFILE* macro are hooked to the x- and y-component of the inlet velocity. As their implementation is almost identical, they both call the *SetInletVelocity* function, which calculates the x- and y-component for the current flow conditions and loops over each inlet face to set the calculated value, as shown in figure A.3.

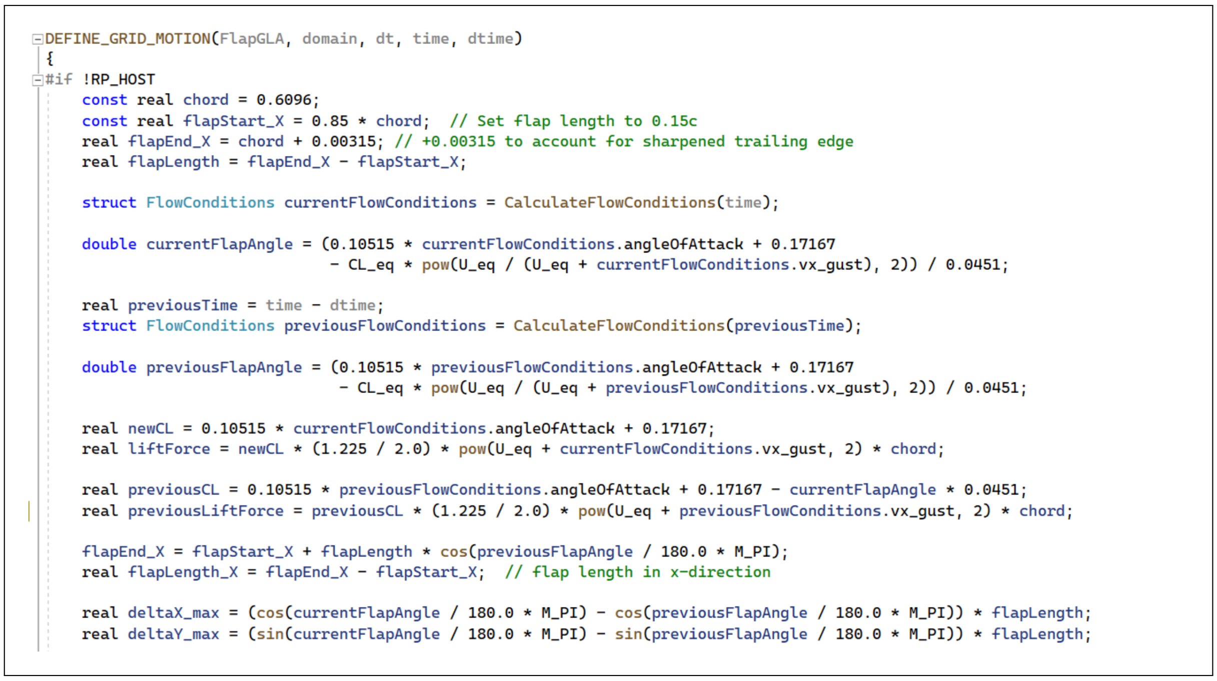


```
void SetInletVelocity(char component, Thread* t, int i) {
    real time = CURRENT_TIME;

    struct FlowConditions currentFlowConditions = CalculateFlowConditions(time);
    real velocityMagnitude = currentFlowConditions.velocityMagnitude;
    real velocityComponent;

    switch (component) {
        case 'x': velocityComponent = velocityMagnitude * cos(currentFlowConditions.angleOfAttack / 180.0 * M_PI); break;
        case 'y': velocityComponent = velocityMagnitude * sin(currentFlowConditions.angleOfAttack / 180.0 * M_PI); break;
        default: Message("Error: Specified Component does not exist, must be 'x' or 'y'."); break;
    }

    face_t f;
    begin_f_loop(f, t)
    {
        F_PROFILE(f, t, i) = velocityComponent;
    }
    end_f_loop(f, t);
}


DEFINE_PROFILE(xVelocity, t, i)
{
    SetInletVelocity('x', t, i);
}

DEFINE_PROFILE(yVelocity, t, i)
{
    SetInletVelocity('y', t, i);
}
```

Figure A.3. Inlet velocity calculation

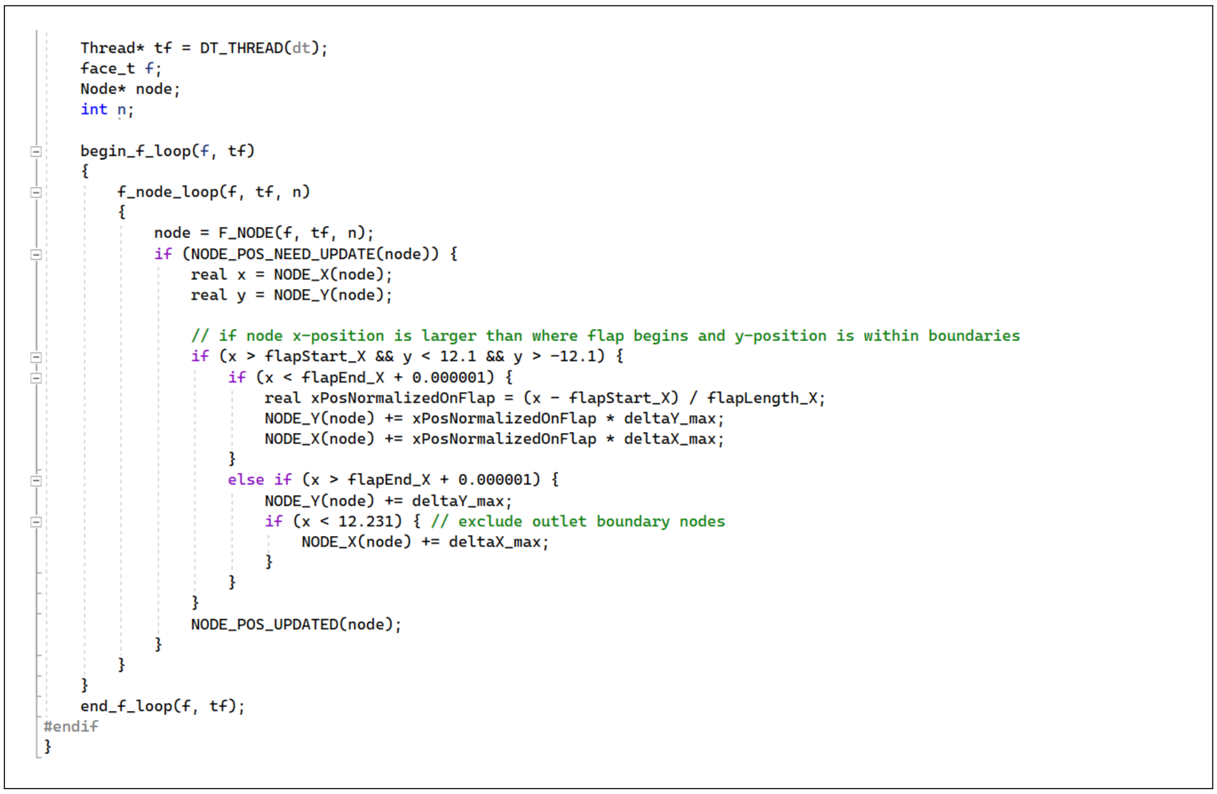
A.3 UDF for Deforming Mesh and Flap Deflection

The *DEFINE_GRID_MOTION* macro is used to define the mesh deformation and hooked to the Dynamic Mesh environment in Ansys Fluent. Figure A.4 shows the calculation of the required flap angle, based on the current flow conditions and the reduced order model. In figure A.5, the loop, which positions each mesh node according to the calculated flap angle is shown.



```c
DEFINE_GRID_MOTION(FlapGLA, domain, dt, time, dtime)
{
#if !RP_HOST
    const real chord = 0.6096;
    const real flapStart_X = 0.85 * chord;  // Set flap length to 0.15c
    real flapEnd_X = chord + 0.00315; // +0.00315 to account for sharpened trailing edge
    real flapLength = flapEnd_X - flapStart_X;

    struct FlowConditions currentFlowConditions = CalculateFlowConditions(time);

    double currentFlapAngle = (0.10515 * currentFlowConditions.angleOfAttack + 0.17167
                             - CL_eq * pow(U_eq / (U_eq + currentFlowConditions.vx_gust), 2)) / 0.0451;

    real previousTime = time - dtime;
    struct FlowConditions previousFlowConditions = CalculateFlowConditions(previousTime);

    double previousFlapAngle = (0.10515 * previousFlowConditions.angleOfAttack + 0.17167
                              - CL_eq * pow(U_eq / (U_eq + previousFlowConditions.vx_gust), 2)) / 0.0451;

    real newCL = 0.10515 * currentFlowConditions.angleOfAttack + 0.17167;
    real liftForce = newCL * (1.225 / 2.0) * pow(U_eq + currentFlowConditions.vx_gust, 2) * chord;

    real previousCL = 0.10515 * previousFlowConditions.angleOfAttack + 0.17167 - currentFlapAngle * 0.0451;
    real previousLiftForce = previousCL * (1.225 / 2.0) * pow(U_eq + previousFlowConditions.vx_gust, 2) * chord;

    flapEnd_X = flapStart_X + flapLength * cos(previousFlapAngle / 180.0 * M_PI);
    real flapLength_X = flapEnd_X - flapStart_X;  // flap length in x-direction

    real deltaX_max = (cos(currentFlapAngle / 180.0 * M_PI) - cos(previousFlapAngle / 180.0 * M_PI)) * flapLength;
    real deltaY_max = (sin(currentFlapAngle / 180.0 * M_PI) - sin(previousFlapAngle / 180.0 * M_PI)) * flapLength;
```

Figure A.4. Flap deflection angle calculation



```c
    Thread* tf = DT_THREAD(dt);
    face_t f;
    Node* node;
    int n;

    begin_f_loop(f, tf)
    {
        f_node_loop(f, tf, n)
        {
            node = F_NODE(f, tf, n);
            if (NODE_POS_NEED_UPDATE(node)) {
                real x = NODE_X(node);
                real y = NODE_Y(node);

                // if node x-position is larger than where flap begins and y-position is within boundaries
                if (x > flapStart_X && y < 12.1 && y > -12.1) {
                    if (x < flapEnd_X + 0.000001) {
                        real xPosNormalizedOnFlap = (x - flapStart_X) / flapLength_X;
                        NODE_Y(node) += xPosNormalizedOnFlap * deltaY_max;
                        NODE_X(node) += xPosNormalizedOnFlap * deltaX_max;
                    }
                    else if (x > flapEnd_X + 0.000001) {
                        NODE_Y(node) += deltaY_max;
                        if (x < 12.231) { // exclude outlet boundary nodes
                            NODE_X(node) += deltaX_max;
                        }
                    }
                }
                NODE_POS_UPDATED(node);
            }
        }
    }
    end_f_loop(f, tf);
#endif
}
```

Figure A.5. Deforming mesh node loop