Imperial College London
AERO97051 - Applied Computational Aerodynamics

ACA Group Report

February 3, 2020

Academic responsible:	Dr Joaquim Peiro
Department:	Department of Aeronautics
Students:	Ann-Kristin Sturm - 01828559
	Nathan Davey - 01187454

1 Part A

1.1 PyFR

PyFR is an open source project written to solve advection-diffusion time problems including compressible flow, unsteady flow, inviscid flow and separation problems in 2D and 3D. It uses Euler and Navier Stokes equations as the governing flow equations, and the simulations relies solely on large eddy simulation (LES) and direct numerical simulation (DNS) for turbulence moddeling. There is no use of RANS which is a characteristic feature of PyFR. In terms of spatial discretization, first the domain is divided into cell elements. A polynomial approximation of the solution is then generated across each element. The use of polynomial approximations allows the use of higher order polynomials if desired, which are increasing the solution accuracy but at the cost of computation time. The solution is continuous within each element, but discontinuous at the element borders. A high order flux reconstruction (hence the FR in PyFR) scheme is employed across elements and is continuous across borders. The use of flux reconstruction allows for partial temporal locality, meaning we only need the flux on neighbouring elements to compute solutions. In terms of temporal discretization, implicit and explicit schemes are employed via dual time stepping (used with a second-order backwards difference formula). Since the code is highly scalable and parallalism is effectively utilised, the code is useful for large simulations such as turbulent flow over low pressure turbines, or analysing the flow of air through aircraft turbines to reduce noise.

1.2 Pantharei

The Pantharei solver uses 2nd order finite volume discretization in space (using a fourier structure grid) and 3rd order discretization in time to solve incompressible flows quickly and reliably. A fractional step method is also employed to deal with pressure velocity coupling in incompressible flows. The program uses steady RANS equations for turbulence modeling, as well as an explicit algebraic Reynolds stress model and standard Smagorinsky and dynamic sub-grid scale models for LES simulations. It's application extends to modeling flow transitions around a wing and the prediction of nanoparticle formulations, and can also be used for nonlinear optimisation problems such as propeller optimisation.

1.3 Incompact3D

Like PyFR, Incompact3D is dedicated to DNS/LES simulation. It is used mainly to simulate incompressible navier-stokes flows, which are discretized with finite-difference sixth-order schemes on a cartesian mesh. It uses an explicit or semi-implicit temporal scheme for time advancement (Adams-Bashforth or Runge-Kutta), and a fraction step method to solve a Poisson equation (using fast Fourier transforms). For turbulence, a dynamic Smagorinky model, as well as LES and wall-adapting local eddy viscosity models are used. Applications include the study of turbulent flows and how they work, and the transfer of energy between scales.

1.4 Nektar++

Nektar++ is used to simulate a variety of flows including incompressible, compressible, acoustic flows and even cardiac electrophysiology. The solver incorporates a spectral/hp elements solver, which has the geometric flexibility of an h-type method with the high resolution properties of spectral methods. Many temporal discretization algorithms are supported depending on the flow type being analysed (most of which are explicit), including Forward Euler and Runge-Kutta (2nd, 3rd and 4th order). Nektar++ utilizes RANS, LES/DNS and detached eddie (DNS) simulations for turbulence. Applications include motorsports, the study of electrical activity in the heart and HPC simulations.

2 Part B on 2D Aerofoil

2.1 Individual Task A - Nathan Davey

The reference and boundary conditions can be found in Table 1. An explanation of how they were calculated can be found in Part (i).

name	variable	value
Mach number	Ma	0.3
Reynolds number	Re	1 000 000
Chord length	c	1 m
Specific heat ratio	γ	1.4
Ideal gas Constant	R	287 J/kg K
Altitude	h	0 m
Temperature	T	288.15 K
Pressure	p	101 325 Pa
Density	ρ	1.225 kg/m^3
Flow velocity	v	102.09 m/s
Dyn. viscosity	μ	$1.251 \cdot 10^{-4}$ Pa·s
Friction coefficient	c_f	$3.745 \cdot 10^{-3}$
Wall stress	τ_w	23.9101 N/m^2
Wall velocity	u_τ	4.418 m/s
Target y^+	y^+	0.25
Prism layer near wall thickness	Δs	1.16×10^{-5} m
Boundary layer thickness	BL	0.0233 m

Table 1: Reference and Boundary conditions

2.1.1 (i)

The boundary layer thickness (BL) and the thickness of the first element at the wall (Δs) can be calculated with equations from the lecture notes and correlations from the tutorials. The BL can be calculated with a correlation taken from the tutorial, where c is the chord length (c):

$$BL = 0.37 \cdot \frac{c}{Re^{\frac{1}{5}}} \tag{1}$$

The Δs can be calculated as a function of the dynamic viscosity (μ) and the wall velocity (u_τ):

$$\Delta s = y^+ \cdot \frac{\mu}{\rho} \cdot u_\tau \qquad with \qquad \mu = \frac{\rho \cdot v \cdot c}{Re} \qquad and \qquad u_\tau = \sqrt{\frac{\tau_w}{\rho}} \tag{2}$$

Where y^+ is the chosen target y^+ value which is set to 1 for a first simulation and then decreased until the y^+ values are smaller than 1 for the whole airfoil. The wall stress (τ_w) can be calculated with the friction coefficient (c_f) and the flow velocity (v):

$$\tau_w = \frac{1}{2} \cdot c_f \cdot \rho \cdot v^2 \qquad with \qquad c_f = (2 \cdot log(Re) - 0.65)^{-2.3} \tag{3}$$

The friction coefficient can be calculated with several different correlations; in this coursework the equation given in the tutorials was chosen. The velocity could be calculated with the speed of sound (a) and the Mach number (M):

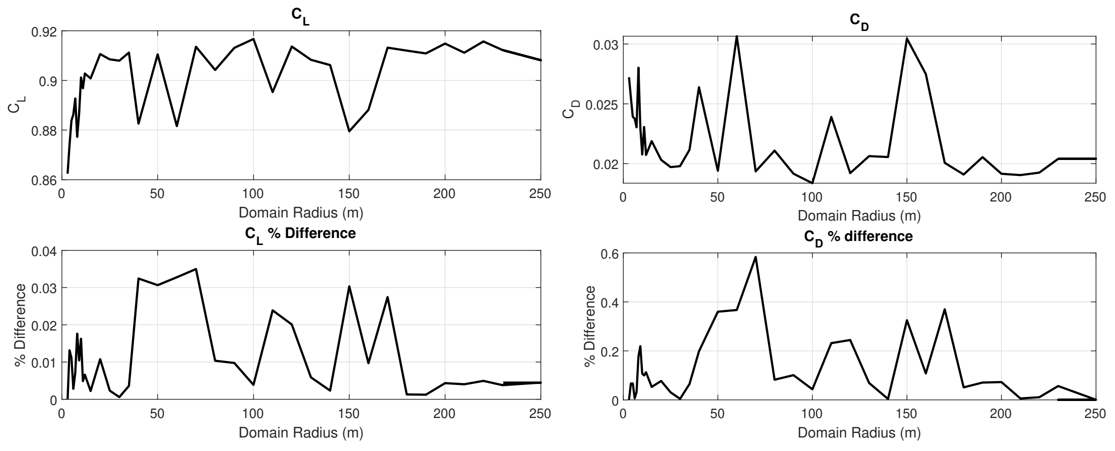
$$v = M \cdot a \qquad with \qquad a = \sqrt{\gamma \cdot R \cdot T} \tag{4}$$

The number of prism layers was selected based on the desire to have the stretch factor close to 1.5. Given this requirement, we can calculate the number of prism layers using:

$$\sum_{i=1}^{N} \Delta s \times 1.5^{i-1} = BL \tag{5}$$

Where N is the number of prism layers. This can be solved iteratively to achieve a desired layer thickness, resulting in a number of prism layers about 17. The mesh was then visually inspected to ensure the outer most boundary layer elements had a similar size to the immediately adjacent elements to avoid discontinuities. A final value of 20 m was settled upon.

2.1.2 (ii)

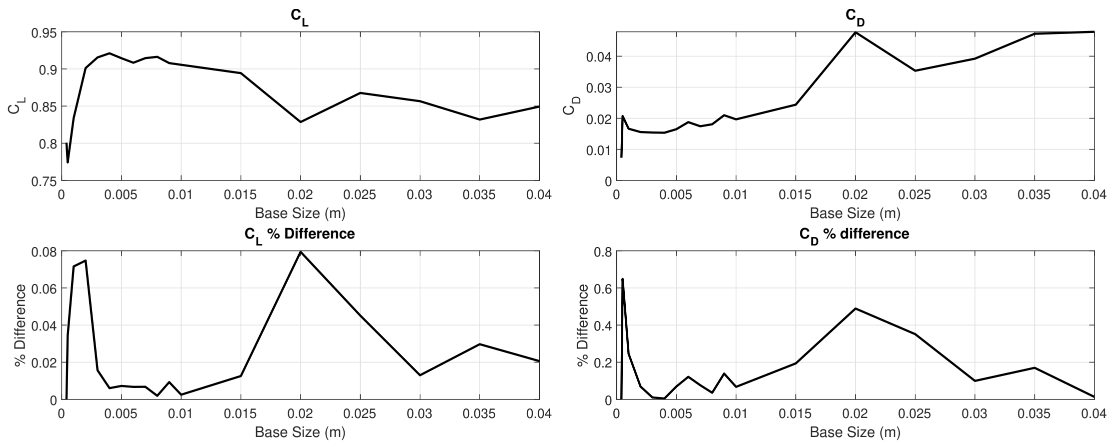


(a) Lift coefficient for varying domain radii sizes. **(b)** Drag coefficient for varying domain radii sizes.

Figure 1: Convergence of lift and drag coefficients for varying radii sizes. % difference between adjacent values is given for comparison between fluctuations

From figure 1 we observe convergence of C_L at around 30 meters, with some small fluctuations at particular values afterwards. We desire to have a domain as small size as possible to reduce computational cost, so a point soon after convergence (35 m) was chosen to take forward. Based on the percentage difference in values we observe that C_D has more difficulty converging (there is an order of magnitude difference in percentage difference for C_D values compared to C_L values). This may be due to do drag estimation being heavily reliant on the forces in the stream wise direction i.e, wake effects. These can be more turbulent and require a larger domain size to fully capture the flow behaviour, resulting in convergence difficulty.

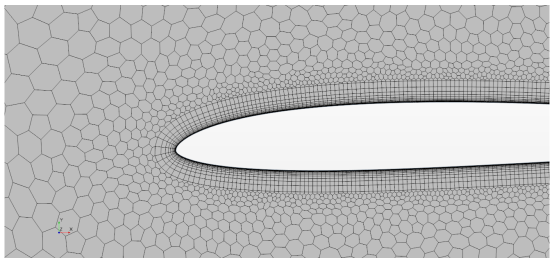
2.1.3 (iii)



(a) Lift coefficient for varying base sizes. **(b)** Drag coefficient for varying base sizes.

Figure 2: Convergence of lift and drag coefficients for varying base size.

The base size appears to show a region of convergence between values of 0.003 m to 0.1 m. Beyond 0.1 m, the solution for C_D diverges drastically and likewise for C_L. The reason for this is apparent when comparing a mesh with a base size of 0.005 and 0.04 as shown in figure 3. We observe that a larger base size results in a more coarse approximation of the airfoil curvature, resulting in diverging lift characteristics. Additionally, the discontinuity between the boundary layer element size and surrounding field element size increases with increasing base size. Ideally we would select a base size as large as convergence would allow, again to decrease the computational cost associated with many small elements. As such, a base size of 0.0075 would be selected.


(a) Airfoil mesh with base size set to 0.005 m

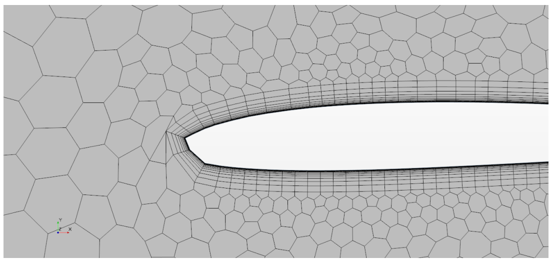

(b) Airfoil mesh with base size set to 0.04 m

Figure 3: Comparison of mesh quality for varying small and large base size.

2.2 Individual Task B - Ann-Kristin Sturm

The flow field for the different angles of attack was initialized by adapting the flow direction and not changing the geometry for each angle of attack. The latter would have led to a recalculation of the mesh for each angle of attack, which is not efficient. The flow directions for the different angles of attack are calculated by $[cos(\alpha)\ sin(\alpha)]$, the field velocity using $([cos(\alpha) \cdot v\ sin(\alpha) \cdot v])$. Additionally, the flow direction in the force coefficient report (cd: $[cos(\alpha)\ sin(\alpha)]$, cl: $[-sin(\alpha)\ cos(\alpha)]$) had to be changed, the values can be found in Table 2.

Angle of attack [°]	Flow direction [-]	Field velocity [m/s]	Flow direction for force coeffs [-]
0	[1, 0]	[102.9, 0]	c_d: [1, 0] c_l: [-0, 1]
2	[0.9994, 0.0349]	[102.03, 3.56]	c_d: [0.9994, 0.0349] c_l: [-0.0349 ,0.9994]
4	[0.9976, 0.0698]	[101.84, 7.12]	c_d: [0.9976, 0.0698] c_l: [-0.0698, 0.9976]
6	[0.9945, 0.1045]	[101.53, 10.67]	c_d: [0.9945, 0.1045] c_l: [-0.1045, 0.9945]
8	[0.9903, 0.1392]	[101.10, 14.21]	c_d: [0.9903, 0.1392] c_l: [-0.1392, 0.9903]

Table 2: Flow field initializing

The lift and drag coefficients for each angle of attack can be found in Table 3.

Angle of attack [°]	c_l [-]	c_d [-]
0	0.2405	0.0098
2	0.4753	0.0105
4	0.7054	0.012
6	0.9264	0.0152
8	1.12	0.0232

Table 3: Lift and Drag coefficients for each angle of attack

The Plots 4 and 5 present a monitor plot of the default residuals (energy, momentum, etc.) and a plot of the convergence history of the two force coefficients (lift and drag).
After 1000 iterations the Residuals decreased below a value of $1 \cdot 10^{-2}$. It would have been possible to stop the simulations earlier (after the residuals had leveled off), but the simulation was allowed to

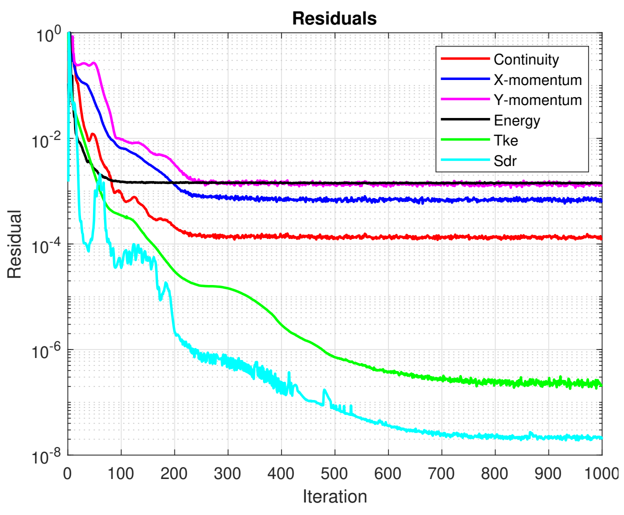


Figure 4: Default residuals for an angle of 4 °

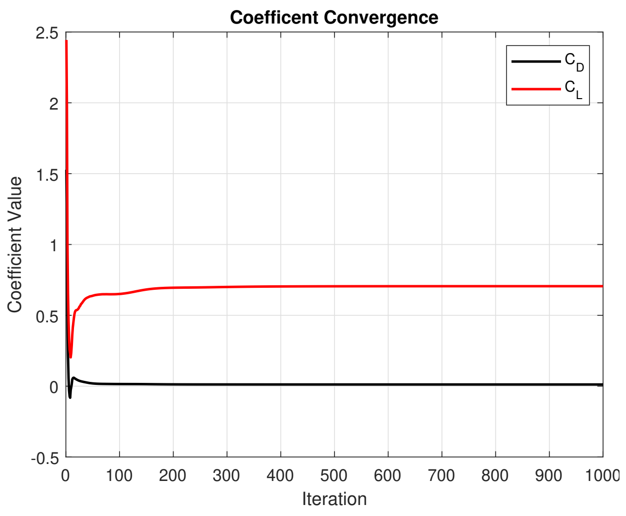


Figure 5: Convergence history for the lift and drag coefficient for an angle of attack of 4°

complete the 1000 iterations to ensure convergence. Beyond this the Residuals do not show significant change with further iterations, indicating the solution has converged. The same applies to the values for the lift and drag coefficients which is shown in Figure 5, thus the simulation is converged and the values for the lift and drag coefficient could be extracted.
Simulating an angle of 10° (or in general higher than 8 °) results in no convergence. It can be observed that the residuals and the drag and lift coefficients have a repeating wave form. At higher angles of attack, the flow around the aerofoil becomes unsteady and produces separations at the trailing edge, resulting in a lack of convergence. In the simulation a steady flow solver was used which is not capable of calculating converged values for an unsteady flow.
The pressure coefficient profile for an angle of attack of 4° in comparison to an in Xfoil calculated pressure coefficient profile is depicted in Figure 6.

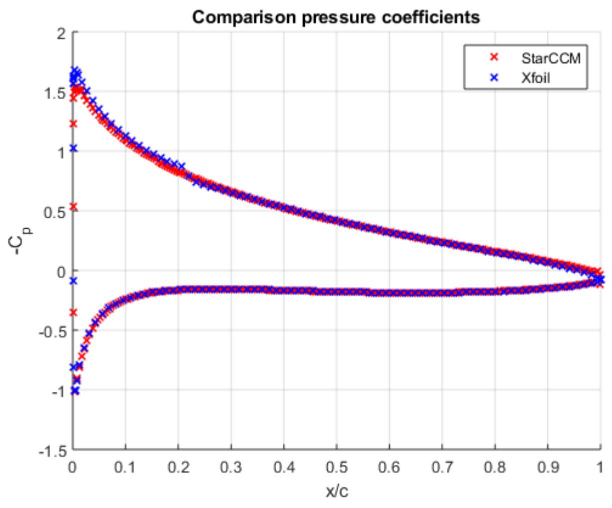


Figure 6: Comparison of the pressure coefficients form Xfoil and StarCCM

Both programs calculate similar values for pressure. The Xfoil data shows a small separation bubble at x/c = 0.2 which is not apparent in the data from StarCCM. The reason lies in the flow model which was chosen in StarCCM, which uses a fully turbulent simulation and does not account for transition. By contract, Xfoil employs a model for predicting the onset of turbulent flow and by extension transition. Xfoil slightly overestimates the pressure coefficient values for $x/c < 0.2$ because it is using a wake trajectory from an inviscid solution. The viscous effect would normally decrease the lift and change the trajectory. A correction for this is not performed in Xfoil because it would result in very long calculation times [1].

2.3 Joint Task C

In Figure 7 a comparison of the lift and drag coefficients for different angles of attack, calculated by StarCCM and Xfoil is depicted.

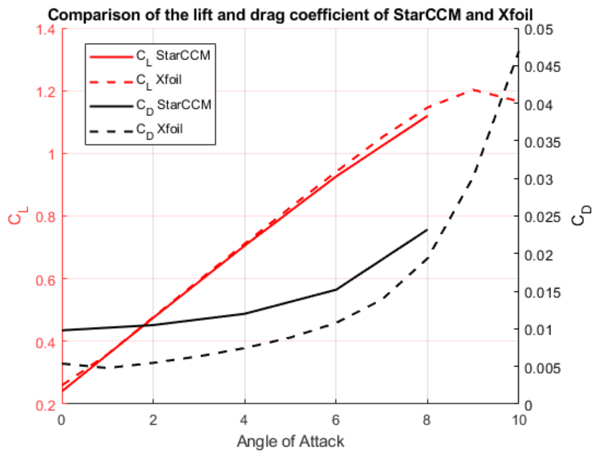


Figure 7: Comparison of the lift and drag coefficient

Running the simulations, it was observed that the y^+ value increase for higher angles of attack. In order to obtain a $y^+ < 1$ for all simulations, the simulation with the highest angle of attack was run first in order to adapt the Δs to provide a $y^+ < 1$ and so one single mesh could be used. The general principle was to refine the mesh (smaller base size, more prism layers, smaller Δs, etc.) such that $y^+ < 1$ during simulations and run simulations with higher angles of attack.
The used model for the simulations was the k - ω -SST model. It is a further development of the k - ω model and combines the good properties of the k - ϵ model with those of the k - ω model. The k - ϵ model can perform calculations inside the flow field without major deviations. However, problems arise in flows which detach from the wall due to changing pressure gradients (e.g. at high angles of attack). The k - ϵ model provides solutions near walls that are too optimistic. Using a k - ω model results in more precise results near the wall. However, the accuracy inside the flow field is less than that of the k - ϵ model. The k - ω -SST model combines the accuracy of the k - ω model near the wall and the accuracy of the k - ϵ model in the rest of the flow area. [2, 3] Thus, accurate solutions are generated in the entire flow field and the model fits our purposes well.
In order to determine which program is more reliable, the values from StarCCM and Xfoil were compared with experimental data. It could be shown that StarCCM calculates an accurate lift coefficient but the drag coefficient calculations are not as reliable. StarCCM overestimates the drag coefficient. For small angle of attacks ($< 6°$) Xfoil is accurate but at higher angles of attack Xfoil overestimates the experimental data. StarCCM appears more reliable because the lift coefficient matches with the experimental data and even if the drag coefficient is overestimated, it is consistently overestimated by a certain ratio.

3 Part C on 3D Wing

The geometry shows that the x-axis is directed towards the trailing edge, the z-axis is directed perpendicular to the x-axis and towards the top surface of the airfoil, and the y-axis comes out of the x-z-plane. The flow direction, velocity, and lift and drag coefficient vectors are defined in reference to this axis.
In a first step the altitude, pressure p, density ρ, temperature T, dynamic viscosity μ and flow velocity v have to be calculated. This can be done by calculating the Reynolds number for different heights with the aid of a standard atmosphere table, and then comparing it with the given Re of 1 000 000. We can use the following equations to calculate the unknowns ρ, T and μ based on known values Re, M, γ and R:

$$Re = \frac{\rho \cdot v \cdot c}{\mu} \qquad with \qquad v = M \cdot \sqrt{R \cdot T \cdot \gamma} \tag{6}$$

$$\Rightarrow Re = \frac{\rho \cdot \sqrt{T}}{\mu} \cdot c \cdot M \cdot \sqrt{\gamma \cdot R} \tag{7}$$

The unknown part from the equation ($\frac{\rho \cdot v \cdot c}{\mu}$) can be determined for each altitude with the values of the standard atmosphere table. The known part ($c \cdot M \cdot \sqrt{\gamma \cdot R}$) for an M of 0.4 is 1.222, and for an M of 0.8 is 2.444. Therefore, c was chosen to be 0.15244 m as given in the coursework sheet. By calculating the Re and comparing it with the given Re the following parameters result:

Parameter	$M = 0.4$	$M = 0.8$
Altitude [m]	3 750	10 190
Velocity [m/s]	130.2	238.53
Dyn. viscosity [Pa $\cdot$ s]	$1.67 \cdot 10^{-5}$	$1.447 \cdot 10^{-5}$
Temperature [K]	263.8	221.254
Density [kg/m^3]	0.8414	0.398
Pressure [Pa]	63 720	25 290

Table 4: Simulation settings

The base size, the number of prism layers, the thickness of the prism layer mesh, and the thickness of the first cell of the prism layer are shown in Table 5. The values for the thickness of the first cell of the prism layer are calculated in the same way as explained in Part 2.1.1. In order to provide a $y^+ < 1$ for the viscous part, a target y^+ of 0.1 was chosen. The two different meshs (300 000 cells ($\pm$5%) and 600 000 cells ($\pm$5%)) were found by running simulations with different combinations of base size and number of prism layers. The final values and the resulting cells are also shown in Table 5, the mesh is the same for both of the Mach numbers. The numbers of prism layers and base sizes were calculated and chosen (as explained in Part 2.1.1) so that the number of cells matches as good as possible to the given cell size values.

Parameter	coarse mesh	fine mesh
Base size [m]	0.014	0.009
Number of prism layers [-]	21	26
Thickness of the prism layer [m]	$7.05 \cdot 10^{-7}$	$7.05 \cdot 10^{-7}$
Thickness of the first cell of the prism layer [m]	0.00356	0.00356
Total number of cells [-]	295 746	606 857

Table 5: Mesh settings

With an angle of attack of 2°, the flow direction was calculates as [0.9994, 0.0, 0.0349]. The flow velocity vector for $M = 0.4$ is [130.12, 0.0, 4.54] and for $M = 0.8$ is [238.38, 0.0, 8.32].
In Figure 8 - 11 the scenes for the mesh are depicted.

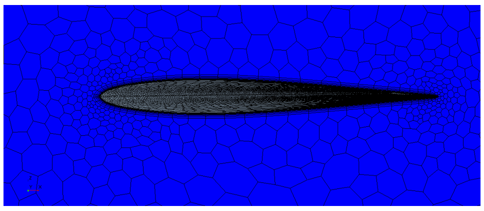


Figure 8: xz-plane coarse mesh

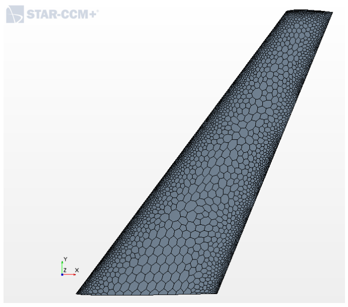


Figure 9: Wing surface coarse mesh

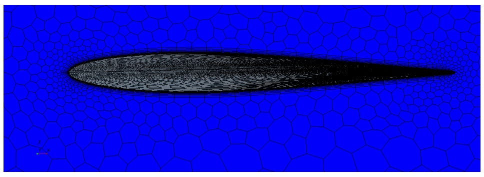


Figure 10: xz-plane fine mesh

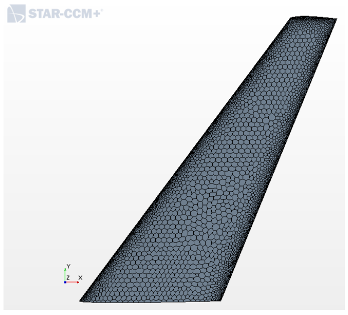


Figure 11: Wing surface fine mesh

3.1 Individual Task A (inviscid flow)

In Figure 12 - 16 the pressure, residual and coefficient convergence plots are depicted. An explanation for the calculation of the simulation settings can be found in section 3. As this is an inviscid flow, no $y+$ had to be calculated and dynamic viscosity was not required.

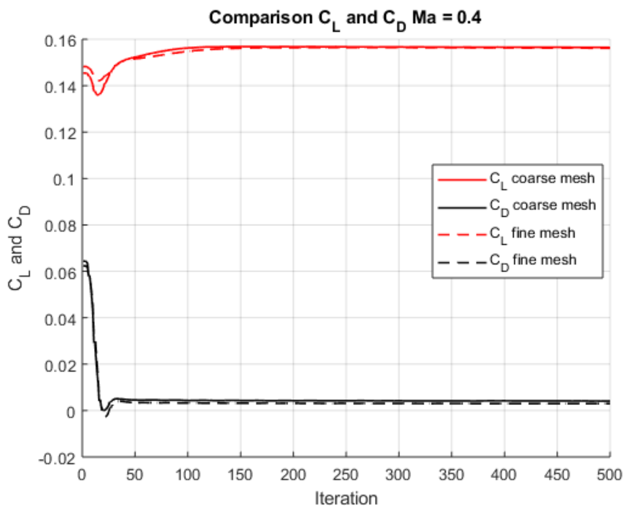


Figure 12: C_L and C_D comparison of course and fine mesh for Mach = 0.4

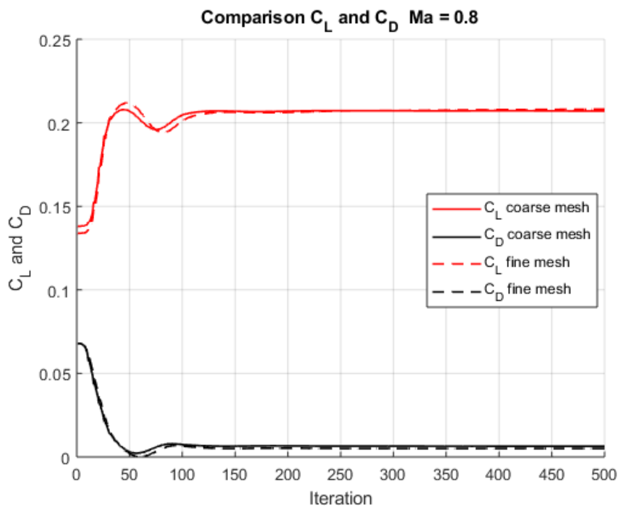


Figure 13: C_L and C_D comparison of course and fine mesh for Mach = 0.8

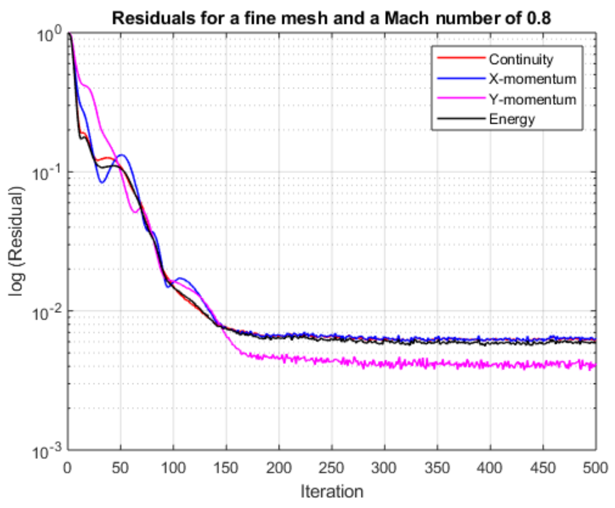


Figure 14: Residuals of a fine mesh with Mach = 0.8

Mach	Mesh	C_L	C_D
0.4	coarse	0.1562	0.003
	fine	0.1565	0.00413
0.8	coarse	0.2071	0.00643
	fine	0.2081	0.00502

Figure 15: C_L and C_D values for inviscid flow

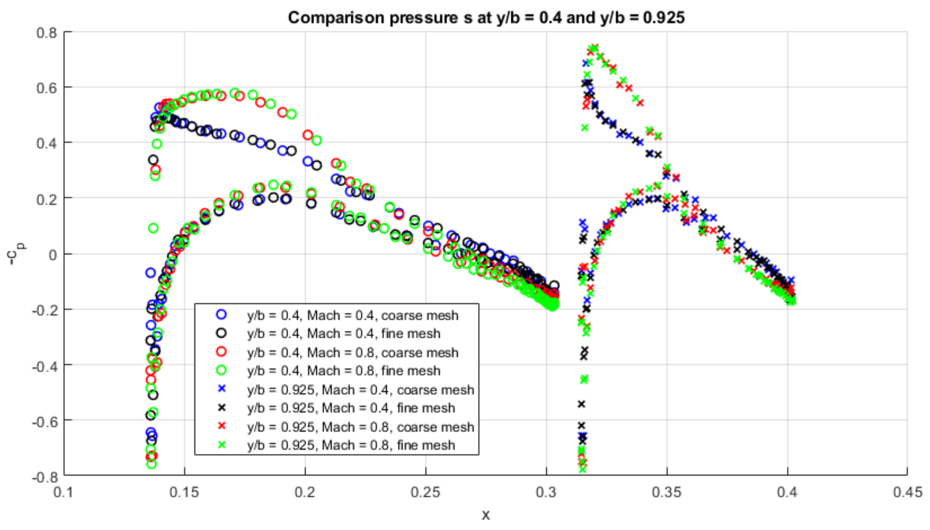


Figure 16: Comparison of the pressure coefficients for all mesh types and mach numbers

3.2 Individual Task B (viscous flow) - Ann-Kristin Sturm

The initial values for the simulation with viscous flow (the flow velocity, pressure, temperature, the viscosity, etc.) are the same as in Part 3. Due to the fact that the flow is viscous now, a dynamic viscosity has to be entered and the y^+ value can be determined. For the different mesh types, the chosen values are shown in Table 4 and Table 5 (for viscous flow, the mesh parameters do not change, in Part 3 a sufficient small Δs was chosen so that a $y^+ < 1$ above the whole aerofoil was produced). In Figure 17 - 21 the relevant plots are depicted.

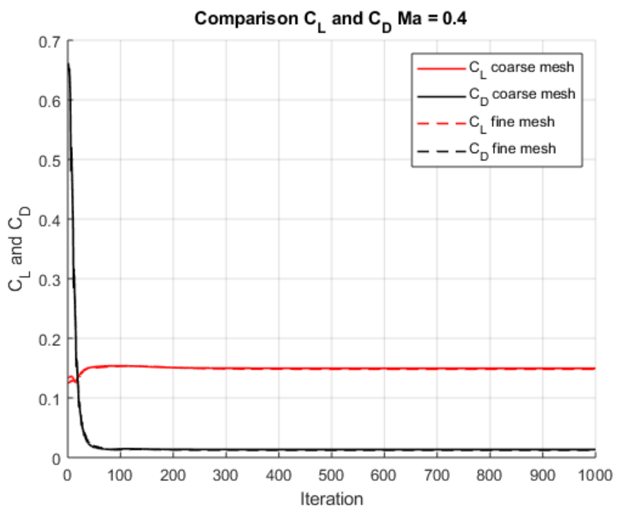


Figure 17: C_L and C_D comparison of course and fine mesh for Mach = 0.4

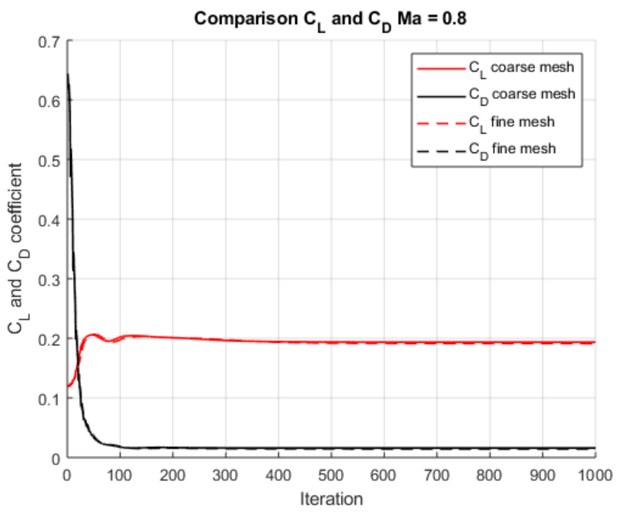


Figure 18: C_L and C_D comparison of course and fine mesh for Mach = 0.8

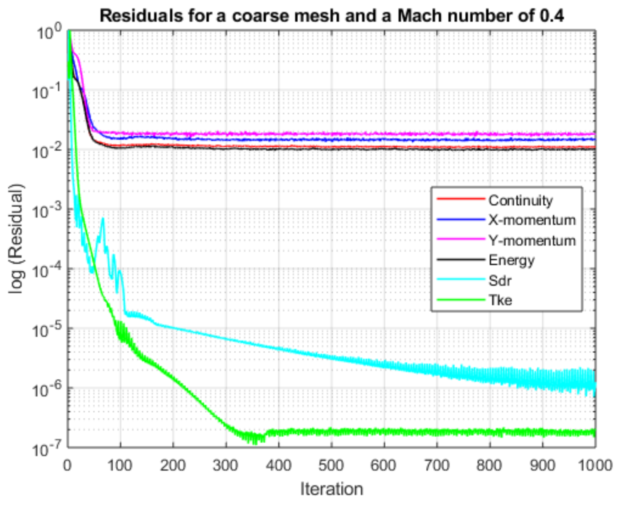


Figure 19: Residuals of a coarse mesh with Mach = 0.4

Mach	Mesh	C_L	C_D
0.4	coarse	0.1497	0.0135
	fine	0.1484	0.0124
0.8	coarse	0.1935	0.0157
	fine	0.1913	0.0145

Figure 20: C_L and C_D values for viscous flow

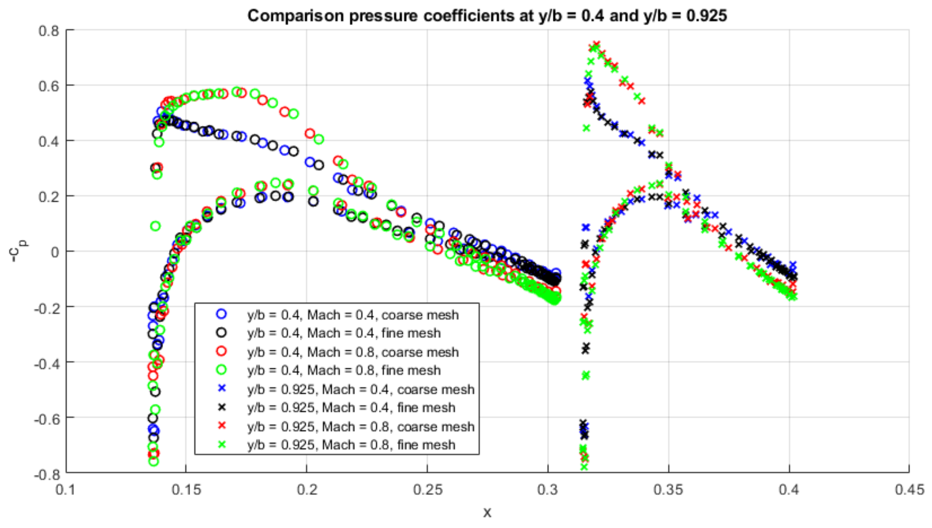


Figure 21: Comparison of the pressure coefficients for all mesh types and mach numbers

3.3 Joint Task C

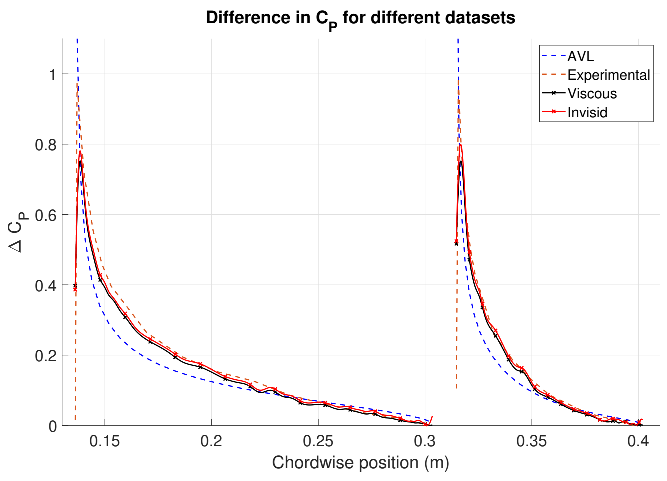


Figure 22: Change in pressure coefficient for AVL, Experimental and Star CCM+ data at Mach = 0.4 with the fine mesh

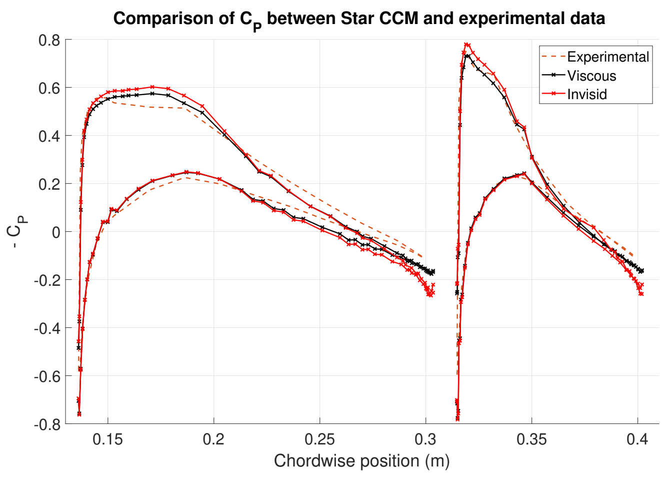


Figure 23: Pressure coefficient at Mach = 0.8 with the fine mesh

In appendix A, the Mach scalar scenes for viscous and inviscid flows with $M = 0.8$ at the aerofoil root are depicted. The supersonic region is shown as a red region in the scalar scene. The shock is observed at the boundary between the red supersonic region (upstream) and orange high speed flow region (downstream). For both flow cases the supersonic region is visible, but it is even more apparent in the viscous case. A coarser mesh shows more of the supersonic region and predicts the shock earlier than the fine mesh. So for accurate shock predictions, a finer mesh should be used.
Prism layers are required to calculate the forces and coefficients accurately near the wall, but they also have repercussive effects which affect the mesh in the rest of the domain. For inviscid flow only a few prism layers are required in order to calculate the forces near the wall, as an inviscid flow simulation does not use the boundary layers for the calculations. In viscous flow simulations there is a boundary layer present, so it requires prism layers in order to give accurate results. As a result, the viscous simulations require more prism layers than the inviscid simulations. The y^+ value was calculated as explained in Part 2.1.1. After a decrease of the y^+ target value, the simulation was run again and the value of y^+ value was checked to ensure $y^+ < 1$ was met.
We observe an entropy increase in the chordwise direction at the wall in appendix B. This is due to transonic flow as flow accelerates over the top of the wing at the tip, resulting in a normal shock around position 0.36 m. The fine mesh appears to predict a larger entropy increase than the coarse mesh. As shock waves are discontinuous phenomenon, we would expect a finer mesh to capture the wave characteristics more accurately.

References

[1] XFOIL 6.9 User Primer.

[2] Stefan Lecheler. Numerische Strömungsberechnung: Schneller Einstieg durch ausführliche praxisrelevante Beispiele. Vieweg und Teubner Verlag / GWV Fachverlage GmbH, Wiesbaden, 2009.

[3] Shawn Wassermann. Die auswahl des richtigen turbulenzmodells für ihre cfd-simulation, 2017.

A Mach scalar scene

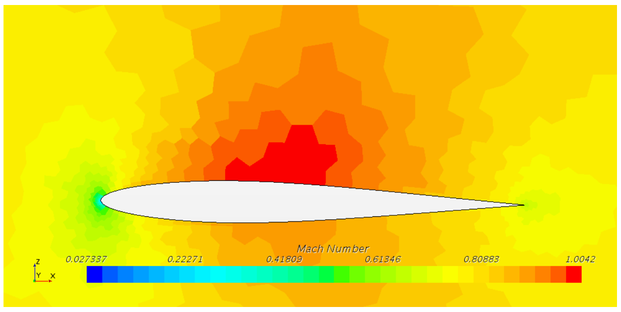


Figure 24: Mach number distribution for a coarse mesh and inviscid flow

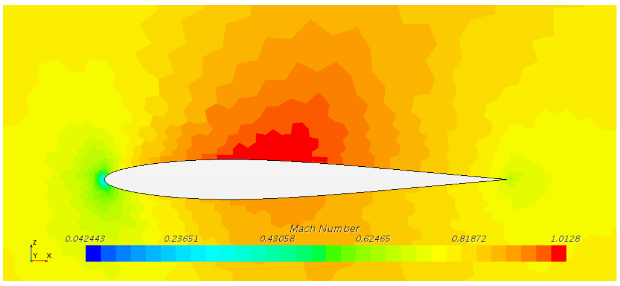


Figure 25: Mach number distribution for a fine mesh and inviscid flow

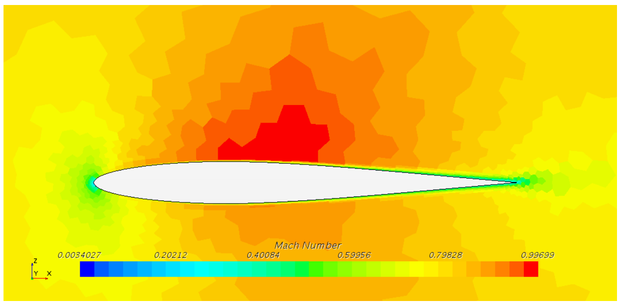


Figure 26: Mach number distribution for a coarse mesh and viscous flow

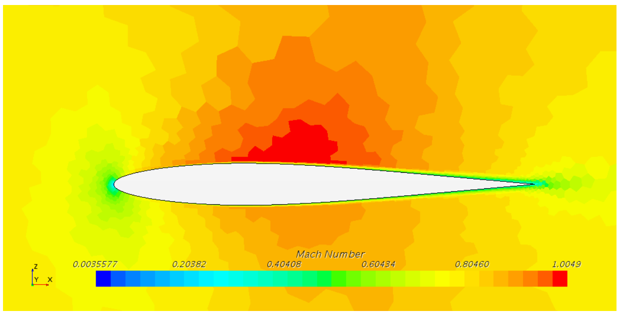


Figure 27: Mach number distribution for a fine mesh and viscous flow

B Entropy Plot

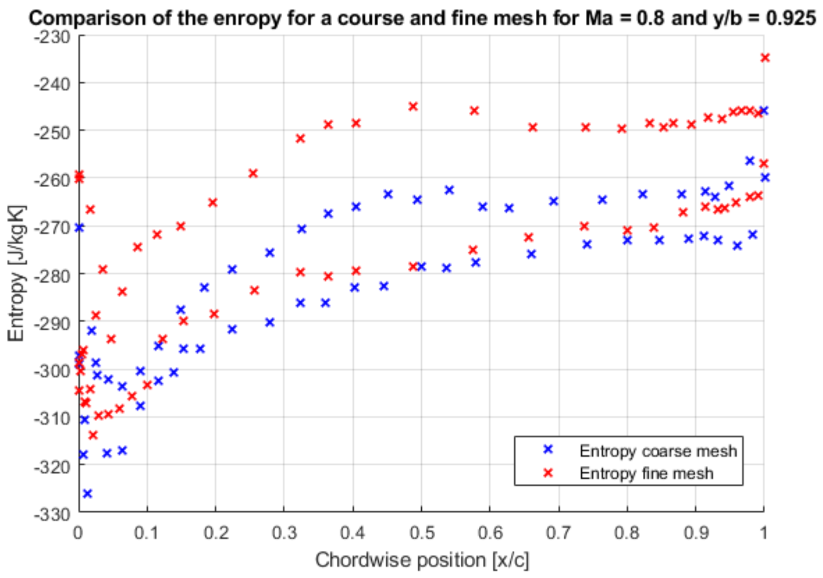


Figure 28: Wall entropy at a location close to the wing tip for the invicid solution

C Data Fitting

When the StarCCM C_P data is plot, we observe a characteristic 'duck' curve, as seen in figure 29. This indicates that StarCCM has a slight bias to 'pond' like flow conditions, attracting interference in the data from the local wildlife. We can reduce the effect of this bias by applying a fox correction factor of 1.

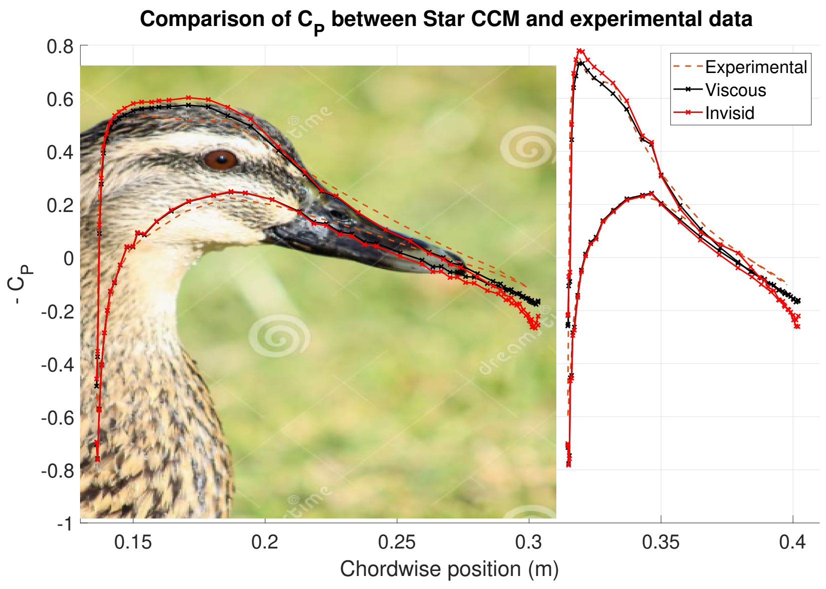


Figure 29: The coefficient of pressure plots with a duck fitting applied