NTNU

Norwegian University of
Science and Technology

Use of OpenFOAM in the Rolls-Royce Propulsion Open Water Simulation system

Solveig Masdal Hovden

Marine Technology
Submission date: June 2017
Supervisor: Sverre Steen, IMT
Co-supervisor: Jonas Eriksson, Rolls Royce

Norwegian University of Science and Technology
Department of Marine Technology

Preface

This thesis is a part of the master degree in Marine Technology at NTNU. It is carried out in the spring semester 2017. It is a collaboration with Rolls-Royce Marine Propulsion in Ulsteinvik. This master thesis presents a study of computational fluid dynamics on propellers. An automatic computational fluid dynamics system is set up and validated using two thrusters. The results are compared with numerical and experimental results.

I would like to thank my supervisor Professor Sverre Steen for the guidance throughout this project. I also thank Rolls-Royce Marine Propulsion in Ulsteinvik for the use of computational resources, office space and assistance. A special thanks to the Research & Technology department for their support and for providing me with a great work environment.

I wish to express my sincere gratitude to my supervisor, Jonas Eriksson, for the time spent guiding me throughout this project. This thesis would not be possible without your continuous support, help and suggestions. I am very grateful for everything I have learned from working with you.

Ulsteinvik, June 8, 2017

Solveig Masdal Hovden

Summary

Rolls-Royce Research & Technology propulsion department has developed a fully automated open water simulation tool called Propulsion Open Water Simulations. This system can be used for any propulsion and thruster system. This enables propeller designers without any prerequisite skills in Reynolds-averaged Navier-Stokes based computational fluid dynamics methods to run advanced simulations. The solver in Propulsion Open Water Simulations is the commercial license based software ANSYS Fluent. This is an expensive solution and it has therefore been proposed to change the solver to the open source software OpenFOAM. The main objectives of this master thesis are to investigate the possibility of using OpenFOAM as a solver in Propulsion Open Water Simulations, generate a mesh that is suitable for OpenFOAM and optimize the solver settings to ensure robustness and manageable computational effort. The validation of the system is done using Azipull120 and Azipull150. The results are validated using experimental data and numerical results from Propulsion Open Water Simulations.

A conformal mesh, suitable for OpenFOAM is created. The quality of the grid generated by BOXERMesh is not completely satisfactory with the setup used in this thesis. The mesh is computationally expensive. A reason for this is the leading edge of the propeller, it easily gets rough due to cells collapsing. Small cells are required in this region. Numerical schemes and solver parameters are chosen based on investigations and common practice. The results from the steady state and transient simulations are within i tolerable accuracy. All simulations converged smoothly, except for the four lowest advance ratios for the Azipull150.

OpenFOAM can be used as solver in Propulsion Open Water Simulations, but the high mesh requirements from OpenFOAM and the low-quality grids generated by BOXERMesh combined is a challenge. It is possible to overcome this, but a higher number of cells than for the current Propulsion Open Water Simulations system seems unavoidable. Changing the mesher could possibly be a solution. It should preferably be an open source mesher, with scripting possibility. The robustness of the system is important to avoid excessive manual work and delays in results. It also important to get reliable results for all simulations.

NTNU Trondheim
Norwegian University of Science and Technology
Department of Marine Technology

MASTER THESIS IN MARINE TECHNOLOGY

SPRING 2017

FOR

Solveig Masdal Hovden

Use of OpenFOAM in the Rolls-Royce Propulsion Open Water Simulation system

R&T Propulsion department has developed a fully automated open water simulation tool for virtually any kind of marine propulsion system under the name POWS, short for Propulsion Open Water Simulation. This enables propel designers to run advanced CFD simulations without any prerequisite skills in RANS based CFD methods. The solver in Propulsion Open Water Simulations is the commercial license based software ANSYS Fluent. This is an expensive solution and it has therefore been proposed to change the solver to the open source software OpenFOAM.

The main objectives of this master thesis are to investigate the possibility of using OpenFOAM as a solver in Propulsion Open Water Simulations, generate a mesh that is suitable for OpenFOAM and optimize the solver settings to ensure robustness and manageable computational effort.

It is important that the computations are validated against high-fidelity, independent calculations and/or model test results. Furthermore, the generality of the mesh and computational set-up should be discussed – meaning that the validity for other cases than those validated shall be discussed.

In the thesis the candidate shall present his personal contribution to the resolution of problem within the scope of the thesis work.

Theories and conclusions shall be based on mathematical derivations and/or logic reasoning identifying the various steps in the deduction.

The thesis work shall be based on the current state of knowledge in the field of study. The current state of knowledge shall be established through a thorough literature study, the results of this study shall be written into the thesis. The candidate should utilize the existing possibilities for obtaining relevant literature.

The thesis shall be organized in a rational manner to give a clear exposition of results, assessments, and conclusions. The text should be brief and to the point, with a clear language. Telegraphic language should be avoided.

The thesis shall contain the following elements: A text defining the scope, preface, list of contents, summary, main body of thesis, conclusions with recommendations for further work, list of symbols and acronyms, reference and (optional) appendices. All figures, tables and equations shall be numerated.

The supervisor may require that the candidate, in an early stage of the work, present a written plan for the completion of the work. The plan shall include a budget for the use of laboratory or other resources that will be charged to the department. Overruns shall be reported to the supervisor.

NTNU Trondheim
Norwegian University of Science and Technology
Department of Marine Technology

The original contribution of the candidate and material taken from other sources shall be clearly defined. Work from other sources shall be properly referenced using an acknowledged referencing system.

The thesis shall be submitted electronically (pdf) in DAIM:

- Signed by the candidate
- The text defining the scope (this text) (signed by the supervisor) included
- Computer code, input files, videos and other electronic appendages can be uploaded in a zipfile in DAIM. Any electronic appendages shall be listed in the main thesis. The candidate will receive a printed copy of the thesis.

Supervisor	: Professor Sverre Steen
Co-supervisor	: Jonas Eriksson, Rolls-Royce Marine
Start	: 12.01.2017
Deadline	: 11.06.2017

Trondheim, 12.01.2017

Sverre Steen
Supervisor

Contents

List of Figures

List of Tables

Chapter 1

Introduction

This master thesis presents a study of automated computational fluid dynamics on propellers using open-source software. Two propellers are investigated and the results are compared with other computational fluid dynamics results and experimental data.

1.1 Background

A ship propeller is a device that transfer rotational power from an engine to thrust. It consists of a revolving shaft with blades. The blades are foil shaped and generates thrust by creating pressure difference between the two sides. The ship is pushed through the water by this thrust. (Muntean, 2012) Common propeller types are,

conventional propellers which normally have 3-6 propeller blades and can have fixed or controllable pitch,

contra-rotating propellers where two propellers are placed behind each other and rotates in opposite directions,

ducted propeller which have a duct that increase the thrust and efficiency for low speed,

podded propulsion where the engine is placed in a pod and

semi-submerged propellers where only parts of the propeller is submerged, this performs well for high speeds. In addition there are

thrusters, the two main types of thrusters are tunnel thrusters and azimuthing thrusters. Tunnel thrusters are used for maneuvering at low speed, rotating thrusters can be used for low speed maneuvering, propulsion and dynamic positioning. (Steen, 2011)

Energy efficient ships are important to reduce cost and environmental impact. An important contribution to the overall efficiency is the propulsion and thruster efficiency. It can be discussed whether the propulsion efficiency found using model tests or simulations has any practical meaning since the external condition the ship faces is crucial to the real world efficiency of the ship. Even if the results can not be directly used for real world conditions, it is a useful parameter when comparing propellers. For propeller manufacturers to be able to sell propellers, they have to provide data to show that their propeller design is satisfactory and can compete with other companies designs. It is important that this data is consistent with the full scale measurements. This data is commonly displayed in an open-water diagram, here the propeller efficiency η, the thrust coefficient K_T and the torque coefficient K_Q are displayed.(Muntean, 2012)

To ensure that new propellers and thrusters meet the demands of the buyer in terms of thrust, torque and propeller efficiency, the propeller design needs to be validated. There are long traditions to do model test to ensure that these demands are met. Open water tests provides most of the knowledge required on the performance of the propeller. Open water test are performed with undisturbed inflow, without a hull present. For practical reasons the model tests are conducted on propellers that are smaller than the ones used on ships, scaling laws are introduced to account for this. These are approximations, but years of experience and data have made them sufficient. If conducted by experienced personnel open water tests can give a good estimate of the important characteristics.(Muntean, 2012)

In the later years numerical simulations have been used increasingly to test propeller performance. As robust computational fluid dynamics (CFD) software and high performance computing resources become more available this trend is expected to continue. In computational fluid dynamics the challenges of scaling can be avoided. CFD is a computational technique using numerical methods and algorithms to solve equations of fluid flow and heat transfer. The equations are the conservation of mass, momentum and en-

ergy. These can be combined to the Navier-Stokes equations. One method to solve them are direct numerical simulation (DNS) which solve the Navier-Stokes equations numerically, without the use of turbulence models. The entire range of spatial and temporal scales of turbulence must be resolved. This is computationally too demanding for industry applications, for now. The computational demand can be decreased by solving the Navier-Stokes equation in the same manner as DNS only in large scale and using turbulence models for smaller scale, this is called large eddy simulation (LES). Here filters are used to filter out which part to solve as DNS and which to solve using turbulence models. This is for now used mostly in academic work, but as the computer processors improve the method is more applicable for industry as well.(Hovden, 2016)

This thesis will focus on the most used method in industry, the Reynolds-Averaged Navier-Stokes (RANS) method. Here the turbulence models are used in all scales. To get the correct result from the simulation using the RANS equations the choice of turbulence model, numerical schemes and the grid used is important. It is important to have a mesh fine enough to capture changes in flow and geometry, but also coarse enough for the computational time to be manageable. The shape of the cell, non-orthogonality, skewness and aspect ratio, will also affect the results. The grid, and especially the grid near the wall, is important considerations when turbulence model is chosen. The turbulence models, model the effect of the turbulence, not the turbulence itself. This is a simplification and how good it is depends on the choice of model for the specific case and the parameters in the model. Model test are expensive and time demanding, but gives good approximations in general and are a trusted method in the marine industry. RANS based CFD methods are faster, cheaper and easier to replicate. Also in CFD an experienced user is required. Validation of the results are a challenge both for experimental and numerical methods, therefore a combination may be advisable to ensure reliable results.(Hovden, 2016)

Rolls-Royce Marine Propulsion in Ulsteinvik is a propeller manufacturer. As mentioned, new propeller and thruster designs needs to be validated to ensure good characteristics and to document it for the buyers. Making validation as efficient and easy as possible is important. Rolls-Royce Marine Research & Technology propulsion department has developed a fully automated open water simulation tool called Propulsion Open Water Simulations

(POWS). This system can be used for any propulsion and thruster system. This enables propeller designers without any prerequisite skills in RANS based CFD methods to run advanced simulations. The designer provides geometry file, propeller type and other data for the simulations and get a report back with the results. The computation is performed on the local High Performance Computer Cluster (HPCC). Open Water simulations for different propellers are stored in a reference library, this is done to be able to verify the results as modules in POWS change. This also makes it possible for designers to compare new designs with similar designs. The aim is to make a completely automated verification procedure. Results from POWS is automatically stored in a database.

The simulations in POWS are done using the commercial license based software ANSYS Fluent as solver. This is an expensive solution and it has therefore been proposed to change the solver to a license free one. Another advantage using license free software is that the only limiting factor for calculations is the hardware available. This master thesis will look into the possibility of using the open source software OpenFOAM in the fully automated system POWS. Using a new solver gives new mesh requirements, the tolerance of the non-orthogonality and the mesh conformity at the interface between the rotating and the static domain are examples of areas where this can differs. The cell size at the propeller blade edges and cell size at the interface is also important factors. The mesh needs to be robust to avoid divergence and efficient to make the calculations as little computational demanding as possible. Also the numerical schemes used are investigated, here the goal is to find a stable, accurate and inexpensive set of numerical schemes. As the solver is made for an automated system the stability requirement is of great importance. The turbulence model is decided based on common practice and the parameters included are found using simplified calculations. To validate the new solver and mesh the results are compared with results obtained using ANSYS Fluent and experimental results.

After this thesis is completed there is some work left if OpenFOAM is chosen to replace ANSYS Fluent as solver in POWS. A new, more general mesh is required and the new solver must be implemented in POWS. This would change the meshing, solving and post processing. Also the transient simulations require more thorough treatment to ensure stable simulations and correct results.

1.2 Objective

The main objectives of this master thesis are to

1. investigate the possibility of using OpenFOAM as a solver in POWS,
2. generate a mesh that is suitable for OpenFOAM and
3. optimize the solver settings to ensure robustness and manageable computational effort.

1.3 Limitations

The study is limited by the time and computational resources available. The study is done on podded thrusters with fixed pitch and no duct. Azimuthing is not included. In POWS it is possible to do simulation for several different thrusters. Including more thruster types, requires a new mesh and changes in the solver to account for new patches. One successful transient simulation is performed, more simulations are required to generate an efficient system for transient simulations.

Turbulence models are not investigated as this has been done previously and the turbulence model used is common practice. More schemes and solver parameters are available, due to limited time and computer resources not all are tested. The ones tested are chosen based on the findings presented in the theory chapter. From the output of the simulations this thesis will only consider thrust, torque and open water propeller efficiency. Pressure plots are only used to see if the results look reasonable.

1.4 Approach

Before any work directly related to POWS are done, some two dimensional test cases are set up. This is done to be able to use the systems more efficient and to investigate the differences between ANSYS Fluent and OpenFOAM in regards of mesh quality tolerance.

The new propeller mesh is generated using BOXERMesh. POWS already use BOXERMesh as mesh generator. There are differences in mesh tolerance

between the solvers, the previously used script for mesh generation does not converge using OpenFOAM. One of the problems is the non-conformity, OpenFOAM performs better with a conformal mesh interface. In a conformal mesh interface, every node on one side has a matching cell on the other side of the interface. Higher mesh quality than ANSYS Fluent, also seems to be required. Mesh quality is related to non-orthogonality, skewness, cell aspect ratio and cell openness. It is most difficult to resolve the geometry and flow gradients at the leading edge. These differences makes it necessary to make a new mesh generation script. The mesh is generated using a script, to makes it possible to implement it in an automatic system. The goal for the mesh is to make it robust without use of excessive computational resources.

After the mesh is investigated different numerical schemes are tested. One scheme is changed for each run to be able to see where the changes in results originates from. Also the level of under-relaxation is investigated. The investigations are performed on steady state simulations. Later transient simulations also were done.

1.5 Structure of the Report

The rest of the report is organized as follows. Chapter two explains briefly the theory behind the simulations and some general note about computational fluid dynamics on propellers. Chapter three gives a short overview of the software used in this thesis. The fourth chapter describes the approach to investigate the mesh, numerical scheme and solver. The results from the simulations are presented in chapter five. In chapter six the results are discussed and a conclusion is made in chapter seven. The eighth chapter explains what work remains to be done. In the appendix, the acronyms, standard scheme and additional simulation results are presented. Paragraphs marked with (Hovden, 2016), are copied from the project thesis. Some changes may be done, some additions or parts removed.

Chapter 2

Theory

This chapter is based on the theory chapter written in the project thesis, the section about governing equations is similar except from additional representation of the Navier-Stokes equations. In the rest of the chapter some general parts from the project thesis are included, but all topics are investigated more thoroughly and more specific for the master thesis.

The fluid flow can be described by differential equations that ensures the conservation of mass, momentum and energy. To solve the governing partial differential equations numerically, they are discretized and solved as a system of algebraic equations. To obtain the solution, the boundary conditions (BC) and initial conditions (IC) must be known. The output from these equations is the velocity and pressure field, in addition to the density and temperature distributions where this is of interest. The process is visualized in Figure 2.1.

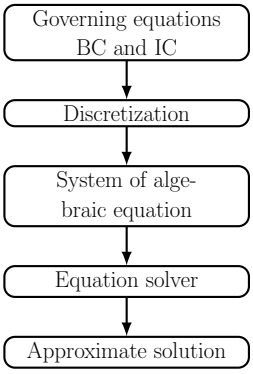


Figure 2.1: Solving the governing equations numerically. Found in lecture notes by Vasileska (xxxx).

2.1 Governing equations

The governing equations of fluid flow is based on the assumption that the fluid is dense enough to be considered a continuum. This means that even in a infinitesimally small part of the fluid there is enough particles to specify mean velocity and mean kinetic energy. This makes it possible to specify velocity, pressure, density and temperature.

2.1.1 Conservation laws

The principal equations of fluid dynamic are based on three conservation laws, conservation of mass, momentum and energy. The total variation inside an arbitrary volume can be expressed as the net flux across the boundary, internal forces, internal sources and external forces. When defining the conservation laws a finite control volume Ω is used. The volume is bounded by the closed surface $\partial\Omega$.

The conservation of mass or the continuity equation is showed in Equation 2.1. In a single-phase fluid, mass can not be created or disappear. The first term in Equation 2.1 is the time change of mass inside the volume, and the second term is the contribution from the convective flux. In Equation 2.1, ρ is the density, dS is a surface element, $\vec{v}$ is the flow velocity and $\vec{n}$ is the unit normal vector.

$$\frac{\partial}{\partial t}\int_{\Omega}\rho d\Omega + \oint_{\partial\Omega}\rho(\vec{v}\cdot\vec{n})dS = 0. \tag{2.1}$$

The second conservation law is the conservation of momentum or the momentum equation, showed in Equation 2.2. Where $\rho\vec{f_e}$ is the body force per unit volume and $\bar{\bar{\tau}}$ is a viscous stress tensor. The first term in Equation 2.2 is the time change of momentum in the control volume, the second term is the contribution from the convective flux tensor and the third is the external body forces. The fourth and fifth term, are the surface sources, first from pressure and then from viscous stresses.

$$\frac{\partial}{\partial t}\int_\Omega \rho\vec{v}d\Omega + \oint_{\partial\Omega} \rho\vec{v}(\vec{v}\cdot\vec{n})dS = \int_\Omega \rho\vec{f_e}d\Omega - \oint_{\partial\Omega} p\vec{n}dS + \oint_{\partial\Omega} (\bar{\bar{\tau}}\cdot\vec{n})dS. \quad (2.2)$$

The last of the conservation laws is the conservation of energy, or the energy equation which is showed in Equation 2.3. Where ρE is the total energy per unit volume, ρH is the total enthalpy per unit volume, k is the thermal conductivity coefficient, T is the absolute static temperature and $\dot{q_h}$ is the time rate of heat transport per unit mass. In Equation 2.3 the first term is the time change in energy per unit volume, the second term is the combination of the convective flux and pressure contribution. The third term is from diffusive flux, the fourth term is the volume sources and the fifth term is from shear and normal stresses.

$$\frac{\partial}{\partial t}\int_\Omega \rho E d\Omega + \oint_{\partial\Omega} \rho H(\vec{v}\cdot\vec{n})dS = \oint_{\partial\Omega} k(\nabla T\cdot\vec{n})dS + \int_\Omega (\rho\vec{f_e}\cdot\vec{v} + \dot{q_h})d\Omega + \oint_{\partial\Omega} (\bar{\bar{\tau}}\cdot\vec{v})\cdot\vec{n}dS. \quad (2.3)$$

2.1.2 The Navier-Stokes equations

The conservation laws can be combined to one set of equations called the Navier-Stokes equations. The complete system of the Navier-Stokes equation can be combined into one expression showed in Equation 2.4. The first term is the temporal derivatives of the conservative variables showed in Equation 2.5. The second term is the convection term showed in Equation 2.6, where $V = n_x u + n_y v + n_z w$. The third term is the viscous term showed in equation 2.7. τ is the viscous stresses and Θ describes the work of viscous stresses and heat conduction. The last term is the source terms (Equation 2.8).

$$\frac{\partial}{\partial t}\int_\Omega \vec{W}\mathrm{d}\Omega + \oint_{\partial\Omega} \vec{F_c}\mathrm{d}S - \oint_{\partial\Omega} \vec{F_v}\mathrm{d}S = \frac{\partial}{\partial t}\int_\Omega \vec{Q}\mathrm{d}\Omega \quad (2.4)$$

$$\vec{W} = \begin{bmatrix} \rho \\ \rho u \\ \rho v \\ \rho w \\ \rho E \end{bmatrix} \tag{2.5}$$

$$\vec{F_c} = \begin{bmatrix} \rho V \\ \rho u V + n_x p \\ \rho v V + n_y p \\ \rho w V + n_z p \\ \rho H V \end{bmatrix} \tag{2.6}$$

$$\vec{F_v} = \begin{bmatrix} 0 \\ n_x \tau_x x + n_y \tau_x y + n_z \tau_x z \\ n_x \tau_y x + n_y \tau_y y + n_z \tau_y z \\ n_x \tau_z x + n_y \tau_z y + n_z \tau_z z \\ n_x \Theta_x + n_y \Theta_y + n_z \Theta_z \end{bmatrix} \tag{2.7}$$

$$\vec{Q} = \begin{bmatrix} 0 \\ \rho f_{e,x} \\ \rho f_{e,y} \\ \rho f_{e,z} \\ \rho \vec{f_e} \cdot \vec{v} + \dot{q_h} \end{bmatrix} \tag{2.8}$$

For Newtonian fluid without source terms, the set of equations on differential form can be written:

$$\begin{aligned} \frac{\partial \rho}{\partial t} + \frac{\partial}{\partial x_i}(\rho v_i) &= 0 \\ \frac{\partial}{\partial t}(\rho v_i) + \frac{\partial}{\partial x_i}(\rho v_j v_i) &= -\frac{\partial p}{\partial x_i} + \frac{\partial \tau_{ij}}{\partial x_j} \\ \frac{\partial}{\partial t}(\rho E) + \frac{\partial}{\partial x_j}(\rho v_j H) &= \frac{\partial}{\partial x_j}\Big(k \frac{\partial T}{\partial x_j}\Big) + \frac{\partial}{\partial x_j}(v_i \tau_{ij}). \end{aligned} \tag{2.9}$$

The set of equations can be simplified by assuming incompressible flow:

$$\begin{aligned} \frac{\partial v_i}{\partial x_i} &= 0 \\ \frac{\partial v_i}{\partial t} + v_j \frac{\partial v_i}{\partial x_j} + \frac{1}{\rho}\frac{\partial p}{\partial x_i} - \nu \nabla^2 v_i &= 0 \\ \frac{\partial T}{\partial t} + v_i \frac{\partial T}{\partial x_j} - k\nabla^2 T &= 0. \end{aligned} \tag{2.10}$$

All equations in Section 2.1 are found in Blazek (2001). The section is mainly from Hovden (2016), Equation 2.4 with explanations is added.

2.2 Principles of Solution

When solving the governing equations the physical space and the time first needs to be discretized. Most methods use separate discretization in time and space. This gives the largest flexibility since different levels of approximation can be used on the temporal derivative, the convection term, the diffusion term and the source term. When using separate discretization in time and space, the spatial discretization or grid is used to make time-dependent equations which are advanced in time from known initial conditions. If there is no change over time, the solution is found when steady state is achieved for an iterative computation. In addition to the initial conditions, boundary conditions must be specified. (Hovden, 2016)

2.2.1 Spatial Discretization

There are three main methods of spatial discretization, finite difference, finite volume and finite element. All three methods rely on grids. Grids and grid generation will be more thoroughly discussed in Section 2.4. The solver used in this thesis uses the finite volume approach, which uses the integral formulation of the Navier-Stokes equations (Equation 2.4). This method discretize the governing equations by first dividing the physical space into polyhedral control volumes. The surface terms on the right-hand side of Equations 2.2 and 2.3 are approximated by the sum of the fluxes crossing the individual faces of the control volume. There are different ways to define the control volumes, cell-centered scheme and cell-vertex scheme are two common approaches. In cell-centered scheme the flow quantities are stored in the center

of the grid cells, and the control volume is therefore the same as the grid cells. In the cell-vertex scheme the flow quantities are stored at the grid points, control volume can be the union of all grid cells around or a volume centered around the point. In this thesis the cell-centered scheme is used. Advantages using the finite volume method is that the spatial discretisation is carried out directly in the physical space and it can be used for both structured and unstructured grids. All dependent variables share the same control volume, it is colocated or non-staggered. (Blazek, 2001)

Below, differencing schemes for the convection and diffusion terms are investigated more thoroughly. This investigation is based on Jasak (1996). The last term in the complete Navier-Stokes equations (Equation 2.4), the source term, is not investigated any further here. Some notes are included in Section 2.2.3.

Convection schemes

The second term in the complete Navier-Stokes equations (Equation 2.4), the convection term, is calculated based on convection schemes. These calculates the values of flow parameters ϕ on the face from the values in the cell centers. Central differencing, upwind differencing or a combination is used. Central differencing (CD) is showed in Equation 2.11, where f is the face between the cells, P is the current cell and N is the neighbor. $f_x = \frac{\overline{fN}}{\overline{PN}}$ is an interpolation factor. P, N and f is showed in the schematic drawing in Figure 2.2.

$$\phi_f = f_x \phi_P + (1 - f_x)\phi_N. \tag{2.11}$$

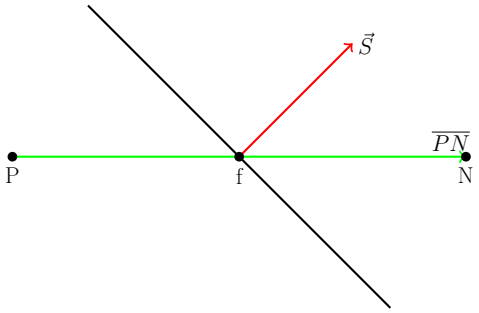


Figure 2.2: Schematic drawing of a non-orthogonal cell showing the cell centre P, the neighboring cell center N, the face intersection f and the surface normal $\vec{S}$. The figure is inspired by Jasak (1996).

CD is second order accurate, but unphysical oscillations can occur for convection-dominating problems. In Equation 2.12, upwind differencing (UD) is showed. Here F is the mass flux trough the face.

$$\phi_f = \begin{cases} \phi_P, & \text{for } F \geq 0 \\ \phi_N, & \text{for } F < 0. \end{cases} \tag{2.12}$$

UD is bounded, but inaccurate due to implicitly introducing numerical diffusion. In addition to central and upwind differencing there are blended differencing (BD), where it is attempted to achieve both boundedness and accuracy. Blended differencing can be expressed:

$$\phi_f = (1-\gamma)(\phi_f)_{UD} + \gamma(\phi_f)_{CD}, \tag{2.13}$$

where the blending factor γ varies between 0 and 1.

Diffusion schemes

The third term in the complete Navier-Stokes equations (Equation 2.4), is discretized using diffusion schemes. These schemes are similar to those of convection. For orthogonal meshes the product $\vec{S} \cdot (\nabla\phi)_f$ is showed in Equation 2.14.

$$\vec{S} \cdot (\nabla\phi)_f = |\vec{S}| \frac{\phi_N - \phi_P}{\overline{PN}} \tag{2.14}$$

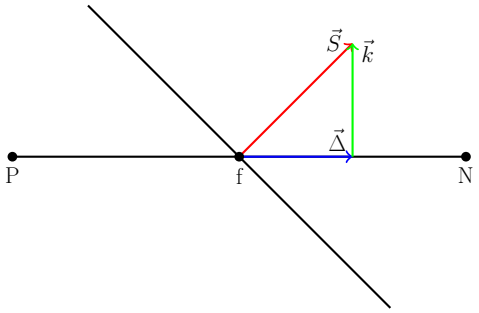


Figure 2.3: Schematic drawing of non-orthogonal cell showing the vectors $\vec{\Delta}$, $\vec{k}$ and $\vec{S}$. The figure is inspired by Jasak (1996).

Most meshes are non-orthogonal and a non-orthogonality correction is applied. Using the non-orthogonality correction, the product $\vec{S} \cdot (\nabla\phi)_f$ can be written as in Equation 2.15, where $(\nabla\phi)_f$ is showed in Equation 2.16. The $\vec{\Delta}$ and $\vec{k}$ vectors are showed in Figure 2.3.

$$\vec{S} \cdot (\nabla\phi)_f = \vec{\Delta} \cdot (\nabla\phi)_f + \vec{k} \cdot (\nabla\phi)_f \tag{2.15}$$

$$(\nabla\phi)_f = f_x(\nabla\phi)_P + (1 - f_x)(\nabla\phi)_N. \tag{2.16}$$

For the first term in Equation 2.15, Equation 2.14 can be used since $\vec{\Delta}$ is parallel with $\overline{PN}$. An advantage using $\vec{\Delta}$ which is parallel with $\overline{PN}$ is that the non-orthogonality correction only applies on non-orthogonal cells. This limits the less accurate approach to only where it is needed. The decomposition of $\vec{S}$ can be done using different methods, three of them are minimum correction, orthogonal correction and over-relaxed approach. For minimum correction, $\vec{S}$ is decomposed as shown in Figure 2.3, where $\vec{k}$ and $\vec{\Delta}$ are orthogonal. For orthogonal correction the length of $\vec{\Delta}$ is the same as $\vec{S}$ and the direction is parallel with $\overline{PN}$. For the over-relaxed approach $\vec{\Delta}$ is defined in Equation 2.17.

$$\vec{\Delta} = \frac{\overline{PN}}{\overline{PN} \cdot \vec{S}} |\vec{S}|^2. \tag{2.17}$$

High mesh non-orthogonality creates unboundedness. The non-orthogonality correction must be limited or disregarded if boundedness is more important than accuracy.

Limiters

Limiters are required by second- or higher-order upwind spatial discretization to avoid oscillations and false results in regions with high gradients. The limiters purpose is to reduce the slopes used when interpolating the variables from the cell centers to the cell faces. The limiters makes the solution more dissipative. This is reduced by only applying the limiters in the regions which it is required. (Blazek, 2001)

2.2.2 Temporal Discretization

Using separate discretization in time and space applied on Equation 2.4 leads to a system of coupled differential equations in time for each control volume.

$$\frac{\partial(\Omega \bar{M} \vec{W})}{\partial t} = -\vec{R}, \tag{2.18}$$

where Ω is the volume of the control volume, $\bar{M}$ is the mass matrix, $\vec{W}$ is the conservative variables (Equation 2.5) and $\vec{R}$ is the residual. The mass matrix $\bar{M}$ can be replaced by the identity matrix and uncouple the system for steady state simulations. For steady state the accuracy is determined by the approximation order of the residual. There are two main classes of time-stepping schemes, explicit and implicit. In the explicit schemes the time derivative is approximated by a forward difference and the residual is evaluated only at the current time step as shown in Equation 2.19.

$$\Delta \vec{W}^n = -\frac{\Delta t}{\Omega} \vec{R}^n, \tag{2.19}$$

where the mass matrix is replaced by the identity matrix. The time step can also be advanced over multiple time steps, called multistage time-stepping (Runge-Kutta). Explicit schemes are numerically cheap and use little memory, but stability severely restricts the time step size. Explicit schemes converges slowly to steady state for viscous flows with highly stretched grid and for stiff systems or systems with stiff source terms. For steady state solutions there are several methods to accelerate the solution. Different time steps can be used for different cells and equations, some implicitness can be introduced or multi-grid can be used. (Blazek, 2001)

In implicit methods the residual is computed for the future time step, in addition to the current. The implicit schemes for steady state can be written:

$$\left(\bar{M}\frac{\Omega}{\Delta t}+\beta\frac{\partial\vec{R}}{\partial\vec{W}}\right)\Delta\vec{W}^{n}=-\vec{R}^{n}, \tag{2.20}$$

where β is normally set to 1. In Equation 2.20 the residual $\vec{R}^{n+1}$ is linearized. The advantage of implicit schemes are that the time steps can be much greater than for explicit schemes. It is also robust and converges fast in the case of stiff systems or stiff source terms. The drawback of implicit methods is that they are hard to vectorize and they demand more computational effort.(Blazek, 2001) (Hovden, 2016)

Stability

As mentioned earlier the stability is the limiting factor on the time steps Δt. To ensure stability the Courant-Friedrichs-Lewy (CFL) condition,

$$CFL=\frac{u_x\Delta t}{\Delta x}+\frac{u_y\Delta t}{\Delta y}+\frac{u_z\Delta t}{\Delta z}\leq C_{max}, \tag{2.21}$$

is used. Where u is the flow velocity, Δt is the time step and Δx is the cell size. C_{max} is the limit that the CFL number is bound by. For a solution to be converging it has to be stable and consistent. A solution is consistent when the local truncation error tends to zero as the mesh size tend to zero.(Hovden, 2016)

2.2.3 Solving

Solving the discretized system of algebraic equation can be done using two main methods, direct and iterative. Direct methods solves the equation by a finite number of arithmetic operations. Iterative methods solves it by an initial guess, and then improve the solution until is tolerable using a number of iterations. Direct methods are used for small systems, they are too computationally expensive for larger systems. Iterative methods are less expensive, but there are some requirement to the matrix. The matrix needs to be diagonal dominant to guarantee convergence. Diagonally dominant means that the value of the diagonal is at least as large as the sum of the rest of

the values in the same row and larger in at least one of the rows. Solver convergence is improved as the diagonally dominance increases.(Jasak, 1996)

A correction implementation can be used to improve the matrix for the convection term. Here, all differencing schemes are treated as an improved upwind scheme, the upwind discretized part is treated in the matrix and the other parts are added to the source term. To guarantee that the diffusion term is diagonally equal, the mesh needs to be orthogonal. This is rarely the case. This is dealt with by including the orthogonal part in the matrix and the non-orthogonality correction to the source term. This only improves the matrix, boundedness is not guaranteed. Non-orthogonality contribution should be disregarded if boundedness is important. The diagonal dominance is increased by the temporal derivative, as there is only a diagonal coefficient and source term. The combined matrix are influenced by all factors mentioned above, the linear part of the source term and the temporal derivative enhance the diagonal dominance. (Jasak, 1996)

The enhancement of the temporal derivative is not present for steady state cases, instead under-relaxation is used. Diagonal dominance is here enhanced by introducing an additional term on both sides of the equation. This term is

$$\frac{1-\alpha}{\alpha} a_P \psi_P^n \tag{2.22}$$

and it is included on both sides of this equation:

$$a_P \psi_P^n + \sum_N a_N \psi_N^n = R_P. \tag{2.23}$$

This becomes

$$\frac{a_P}{\alpha} \psi_P^n + \sum_N a_N \psi_N^n = R_P + \frac{1-\alpha}{\alpha} a_P \psi_P^{n-1}. \tag{2.24}$$

Here α is the under-relaxation factor which vary between 0 and 1. When steady-state is achieved, $\psi_P^n = \psi_P^{n-1}$, additional terms cancel out. (Jasak, 1996)

2.2.4 Pressure-Velocity Coupling

In the discretized Navier-Stokes equations there is a linear dependence of velocity on pressure and vice-versa. This coupling between the equations is

called pressure-velocity coupling and needs to be treated especially. There are two types of algorithms, simultaneous and segregated. The simultaneous algorithm creates a matrix which is several times larger than the number of computational points and includes the coupling between the equations. There are different segregated algorithms, the one used in the steady state calculations of this thesis is called SIMPLE.

SIMPLE Algorithm

In the SIMPLE algorithm, first an initial guess is made, then the process below is iterated until convergence is achieved.

- An approximate velocity field is obtained from the momentum equation.
- The pressure gradients are calculated based on the previous time step.
- Velocity under-relaxation is applied.
- Pressure equation is solved.
- Conservative fluxes are calculated.
- Pressure under-relaxation is applied, $p^{new} = p^{old} + \alpha_P(p^P - p^{old})$. Where
  - p^{new} is the pressure field for the new momentum equation,
  - p^{old} is the pressure field used in the momentum prediction,
  - α_P is the pressure under-relaxation factor and
  - p^P is the pressure field from the solution of the pressure equation.
- Velocities are calculated if needed, using an explicit pressure correction.
- The other equations are solved using the fluxes, velocity and pressure fields.
- Convergence is checked, if not achieved, the process is restarted.

(Jasak, 1996) The pressure equation can be solved more times to update the explicit non-orthogonal corrector for each iterations, this is done for meshes with high non-orthogonality.

2.2.5 PISO Algorithm

The PISO algorithm is used for transient simulations. The steps in the PISO algorithm are:

- The momentum equation is solved using the pressure field from the previous time step. This gives an approximation of the new velocity field.
- The pressure equation is solved to estimate the new pressure field.
- Mass fluxes and velocities are corrected.

A combination of the PISO and SIMPLE algorithms called PIMPLE is used to solve the transient simulations in this thesis. Here the advantages by using under-relaxation factors are applied in addition to the PISO algorithm.

2.2.6 Initial Conditions

The solution steps in Section 2.2.4 requires initial conditions. Using correct initial condition is important in regards to stability, convergence and getting the right solution. Initial conditions determine the state of the fluid at the first step of the iteration. A good initial guess gives a faster converging solution and less risk of divergence. In additions to the initial guess there are methods to improve the initial condition before the calculations. One method is using potential theory to make the initial condition closer to the final solution. (Hovden, 2016)

2.2.7 Boundary Conditions

Using the solution method above requires specified boundary conditions. Boundaries can be divided into artificial and natural boundaries. Artificial boundaries are used on the edges when the modeled domain is a part of the real physical domain and natural boundaries are used when there are walls in the physical flow (Blazek, 2001). Boundaries can also be divided into physical, and numerical boundaries. Some commonly used physical boundary conditions when solving the Navier-Stokes equations are:

- no-slip wall,
- symmetry plane,

- velocity inlet and
- pressure outlet.

For viscous flow around a solid body the relative velocity between the surface and fluid is assumed zero at the surface, this is called no-slip condition. There is no flux trough the wall, this implies that the wall is impermeable and the pressure gradient is zero.(Jasak, 1996)

Using symmetry as a boundary means that the flow is symmetrical with respect to the boundary. There is no flux across the boundary and thus no normal velocity components. Also the gradient normal to the boundary of scalar quantities and tangential velocity must be zero and the gradient along the boundary of the normal velocity must be zero. Symmetry can be used to model zero-shear slip walls.(Blazek, 2001)

At velocity inlets, velocity in all directions are specified and the pressure gradient is set to zero. At pressure outlets, the gauge pressure is set and the velocity gradient is zero. To implement the physical boundary condition, numerical boundary conditions are used. There are two basic types of numerical boundary conditions(BC), Dirichlet and Von Neumann. For Dirichlet BCs the values are specified, while in Von Neumann BCs the gradient is specified.(Jasak, 1996)

2.3 Turbulence

Turbulent flow is irregular, diffusive, occurs at high Reynolds numbers, is three dimensional and dissipative. To make simulations including turbulence applicable for industrial use, an approximation is often required, the Reynolds-Averaged Navier-Stokes (RANS) equation. Here the flow variables are decomposed into a mean and a fluctuating part and inserted into the Navier-stokes equation. Then the equations are averaged, the result of this is an equation set similar to the Navier-Stokes equations with mean values and two additional terms. The additional terms are Reynold stresses and turbulent heat flux. To close the equations, the additional terms are modeled using turbulence models. The models, predict the effect of the turbulence, not the turbulence itself. This is an approximation and how well it is depends on the choice of model and parameters inserted into it. Near wall treatment,

schemes and grid will also affect the result. This section is from Hovden (2016), except for the $k-\omega$ SST subsection.

2.3.1 Reynolds-Averaged Navier-Stokes equations

First the Reynold-Averaged Navier-Stokes equations will be explained more thoroughly. The flow variables are decomposed

$$v_i = \overline{v}_i + v_i', \tag{2.25}$$

where v_i' is the fluctuating part and $\overline{v}_i$ is the mean part. $\overline{v}_i$ is found using time, spatial or ensemble averaging. This decomposition is inserted into the Navier-Stokes equations (Equation 2.10) and then averaged. The fluctuating part averaged is zero, but the product of the fluctuating parts averaged is not. Using time or ensemble averaging the mass and momentum equation of the incompressible Navier-Stokes (Equation 2.10) becomes:

$$\begin{aligned} \frac{\partial \overline{v}_i}{\partial x_i} &= 0 \\ \rho\frac{\partial \overline{v}_i}{\partial t} + \rho\overline{v}_j\frac{\partial \overline{v}_i}{\partial x_j} &= -\frac{\partial p}{\partial x_i} + \frac{\partial}{\partial x_j}\Big(\overline{\tau_{ij}} - \rho\overline{v_i'v_j'}\Big). \end{aligned} \tag{2.26}$$

Equations 2.26 are known as the Reynolds-Averaged Navier-Stokes equations. The equations are similar to two of the equations in Equation 2.10, except that the mean values of the velocity components are used and there is one extra term $\tau_{ij}^R = -\rho\overline{v_i'v_j'}$, called the Reynolds-stress tensor.(Blazek, 2001)

2.3.2 The Boussinesq approach

There are several methods to compute the additional term in Equation 2.26, the Boussinesq approach is a popular one due to its good results and that there is only one parameter to determine. The Boussinesq hypothesis assume that the turbulent shear stress relates linearly to the mean rate of strain, with the eddy viscosity μ_T as proportionality factor (μ_T is not a physical characteristic of the flow). The Boussinesq approach for Equation 2.26 can be written:

$$\tau_{ij}^R = -\rho\overline{v_i'v_j'} = 2\mu_T\overline{S_{ij}} - \frac{2}{3}\rho k\delta_{ij}. \tag{2.27}$$

$$k = \frac{1}{2}\overline{v_i'v_i'}. \tag{2.28}$$

Where $\overline{S_{ij}}$ is the Reynold-Averaged strain-rate tensor and k is the turbulent kinetic energy defined in Equation 2.28. The turbulent kinetic energy k is either found as a by-product of the turbulence model or omitted, this means that only the eddy viscosity μ_T must be determined. When applying the Boussinesq approach on Equation 2.26, μ is simply replaced by $\mu_L + \mu_T$ (a laminar and a turbulent part) in the viscous stress tensor. This means that the Navier-Stokes equation (Equation 2.10) can be extended to turbulent flow only by replacing the flow parameters with the mean values and adding the turbulent eddy viscosity to the laminar viscosity. (Blazek, 2001)

Some common turbulence models to determine the eddy viscosity μ_T, are the $k-\varepsilon$ model, $k-\omega$ model and the $k-\omega$ shear stress transport (SST) model. Based on the findings in my project thesis and common practice in industry the $k-\omega$ SST model is used in this masters thesis and therefore will be more throughly discussed here. The $k-\omega$ SST model is a combination of the $k-\varepsilon$ model and the $k-\omega$ model.

2.3.3 The $k-\varepsilon$ Model

A common turbulence model is the $k-\varepsilon$ model. This model express the eddy viscosity μ_T as a function of k and ε, where k is kinetic energy of the turbulence and ε is the dissipation rate of k. The eddy viscosity can be expressed:

$$\mu_T = C_\mu \rho \frac{k^2}{\varepsilon}. \tag{2.29}$$

Where C_μ is an empirical constant and ρ is the density. The model is based on the Boussinesq approach. k and ε are found from these two partial differential equations:

$$\frac{\partial}{\partial t}(\rho k) + \frac{\partial}{\partial x_i}(\rho k u_i) = \frac{\partial}{\partial x_j}\left[\left(\mu + \frac{\mu_T}{\sigma_k}\right)\frac{\partial k}{\partial x_j}\right] + G_k + G_b + \rho\varepsilon - Y_M + S_k \tag{2.30}$$

and

$$\frac{\partial}{\partial t}(\rho \varepsilon) + \frac{\partial}{\partial x_i}(\rho \varepsilon u_i) = \frac{\partial}{\partial x_j}\left[\left(\mu + \frac{\mu_T}{\sigma_\varepsilon}\right)\frac{\partial \varepsilon}{\partial x_j}\right] + C_{1\varepsilon}\frac{\varepsilon}{k}(G_k + C_{3\varepsilon}G_b) - C_{2\varepsilon}\rho\frac{\varepsilon^2}{k} + S_\varepsilon. \tag{2.31}$$

Where G_k and G_b represents the generation of turbulence kinetic energy due to the mean velocity gradient and buoyancy respectively, Y_M represents the contribution of the fluctuating part in compressible turbulence to the overall dissipation rate, and S_k and S_ε are source terms defined by the user. $C_{1\varepsilon}$, $C_{2\varepsilon}$ and $C_{3\varepsilon}$ are constants and σ_ε and σ_k are the turbulent Prandtl numbers. Equation 2.30 is derived from the exact equation, while Equation 2.31 is obtained from physical reasoning, not similar to the exact solution. The $k-\varepsilon$ model is economic, robust and reasonably accurate and can be used for a wide range of flows. (ANSYS, 2013)

2.3.4 The $k-\omega$ Model

The $k-\omega$ model is similar to the $k-\varepsilon$ model, $\omega \propto \frac{\varepsilon}{k}$. The standard $k-\omega$ model is based on transport equations for turbulence kinetic energy k and specific dissipation rate ω. The $k-\omega$ model is more sensitive to grid quality. The transport equations can be written:

$$\frac{\partial}{\partial t}(\rho k) + \frac{\partial}{\partial x_i}(\rho k u_i) = \frac{\partial}{\partial x_j}\left(\Gamma_k \frac{\partial k}{\partial x_j}\right) + G_k - Y_k + S_k \tag{2.32}$$

and

$$\frac{\partial}{\partial t}(\rho \omega) + \frac{\partial}{\partial x_i}(\rho \omega u_i) = \frac{\partial}{\partial x_j}\left(\Gamma_\omega \frac{\partial \omega}{\partial x_j}\right) + G_\omega - Y_\omega + S_\omega. \tag{2.33}$$

Where Γ_k and Γ_ω are effective diffusivity, S_k and S_ω are source terms defined by the user. G_ω is the generation and Y_ω is the dissipation of ω. G_k is the generation and Y_k is the dissipation of k. The turbulent viscosity is computed from:

$$\mu_T = \alpha^* \frac{\rho k}{\omega}, \tag{2.34}$$

where α^* damps the turbulence viscosity. (ANSYS, 2013)

2.3.5 The $k-\omega$ SST Model

The $k-\omega$ SST model merges a high Reynolds number $k-\varepsilon$ model with the $k-\omega$ model. This is done to exploit the good features of both. Since the $k-\omega$ model does not need any damping function it is used in the sub-layer of

the boundary layer. This makes the model more numerical stable than with the $k-\varepsilon$ model, the accuracy is similar. The $k-\omega$ model is also used in the logarithmic part of the boundary layer, it is better at compressible flows and adverse pressure flows. In the wake region of the boundary layer the $k-\varepsilon$ model is used, here the $k-\omega$ model is too sensitive to the freestream value of ω. Also in the free shear layer, the $k-\varepsilon$ model is used. (Blazek, 2001)

The turbulent eddy-viscosity formulation is also improved to account for the transport effects of the principal turbulent shear stress. A disadvantage of the $k-\omega$ SST model is that the distance to the nearest wall must be known explicitly. (Blazek, 2001)

For the $k-\omega$ SST model, the values for the turbulence kinetic energy k and specific dissipation rate ω, are given for the boundary and initial conditions. They are found using Equation 2.35 and Equation 2.36, respectively. Here U is mean flow velocity, $I = \frac{u'}{U}$ is turbulence intensity, ν is the kinematic viscosity and ν_t is the eddy viscosity.

$$k = \frac{3}{2}(UI)^2, \tag{2.35}$$

$$\omega = \frac{\rho k}{\nu}\nu_t^{-1}. \tag{2.36}$$

2.4 Grid generation

When solving the governing equations the physical space needs to be discretized, this is done by generating a grid. Making a high quality grid is one of the most important tasks when doing CFD computations. Convergence rate, CPU time and final results are heavily affected.

There are two main types of grids, structured and unstructured. In structured grid, each grid point is ordered and uniquely identified by the indices i, j and k and the grid cells are hexaheadrals. For unstructured grids the elements can be tetrahedron, pyramid, prism, hexahedron or a mix. The cells have no particular orders and neighboring cells can not be directly identified by the indices. The main advantage using structured grid is that the cells comes in order, it is quick and easy to access the neighbor to a grid point by adding or subtracting an integer. The disadvantage is that it is time con-

suming and difficult to generate a good quality grid. Unstructured grids on the other hand is easier to generate, but requires more memory and needs sophisticated data structures to work with indirect addressing. (Blazek, 2001) (Hovden, 2016)

2.4.1 Mixed Prismatic/Cartesian Grids

To be able to utilize the advantages of different grid types a mixed grid can be used. One such grid is a mix between prismatic and Cartesian grid. Cartesian grids consists of cubes that align with the directions of the Cartesian coordinate axes. They are easy to generate even for complex geometries. They demand less computational time to generate and the fluid flow computation is faster due to the ordering of the cells. The main disadvantage using Cartesian grids is accuracy around boundaries. Body-fitted prismatic grids resolves the flow at the boundaries accurately, but demands complex grid generation tools for complex geometries. (Blazek, 2001)

2.4.2 Near-Wall Mesh

To obtain accurate results it is important that the turbulent flow is properly resolved. There is a strong interaction between the mean flow and the turbulence, areas with rapid change of mean flow should be sufficiently refined. The near wall mesh can be checked by plotting the non-dimensional wall distance y^+. The non-dimensional wall distance is showed in Equation 2.37, where u_* is the friction velocity defined in Equation 2.38. Here τ_w is the wall shear stress and ρ is the fluid density. y is the distance to the nearest wall and ν is the kinematic viscosity.

$$y^+ = \frac{u_* y}{\nu}. \tag{2.37}$$

$$u_* = \sqrt{\frac{\tau_w}{\rho}}. \tag{2.38}$$

The logarithmic law of the wall states that the average velocity of the turbulent flow at a certain point is proportional to the natural logarithm of the distance to the wall, $u^+ \propto \ln y^+$. This law is applicable for $30 < y^+ < 300$, called the log-law layer, the lower values are most desirable. Here standard or non-equilibrium wall functions are used. For $y^+ < 5$ there is a layer called

the viscous sub-layer, in this layer the variation of the dimensionless velocity is proportional to the dimensionless wall distance. Between the log-law layer and the viscous sublayer there is a layer called the buffer layer ($5 < y^+ < 30$) where none of the laws apply. In this layer there should be no wall-adjacent cells.(Hovden, 2016)

2.4.3 Grid Quality

There are many important features to investigate when looking at grid quality. The cell size should vary smoothly, the grid should be fine enough to resolve all desired features, the cells should not be too skewed and the ratio between edge length should not be too big. Only in the viscous layer the cells can be stretched, since there is a large velocity gradient in the normal direction to the wall, but tangential the gradient is small. It is important that there are no holes or overlapping regions.

OpenFOAM has a built in check for the mesh called *checkMesh*. This utility checks topology to see if the domain is defined correctly and performs the geometrical checks:

Boundary/cell openness is fatal if it fails. Value should be close to 0.

Aspect ratio is defined as: $\frac{1}{6}\frac{|ax|+|ay|+|az|}{v^{\frac{2}{3}}}$. Where ax, ay and az are areas of the bounding box of the cell and v is the volume. Optimal value is 1.

Cell volume/face area is fatal if negative. Difference between minimum and maximum should be as small as possible or have smooth transitions.

Non-orthogonality measures the angle between line connecting two cell centres and and the normal between their common face. 0 is optimal.

Face pyramids is a method for splitting cells into pyramids, the top of the pyramid is in the cell centre. Common with other problems if not OK.

Skewness measures the distance between the intersection of the line connecting two cell centres with the center of their common face. The value is divided on the distance between the cell centres, 0 is optimal.

At the end there is a conclusion that give the number of failed checks. Two of the most important checks are mesh non-orthogonality and skewness.

For non-orthogonality the maximum and average value are printed and all cells with value over 70 degrees are written in a set, that can be plotted. For skewness the maximum value is printed.

2.5 Multiple Moving Reference Frames

For computational fluid dynamics problems involving parts moving relative to each other, the domain have to be divided into different regions, these regions can translate or rotate relative to each other. In this thesis only angular velocities will be discussed.

2.5.1 The Multiple Reference Frame Model

The multiple reference frame model is a steady state approximation. A problem that is unsteady in the inertia frame of reference can be viewed as steady in the rotating frame of reference. The equations of motion can be transformed to the steady rotating frame in a way that make steady state solutions possible. This approach is only valid for problems with weak interactions between the stator and rotor.

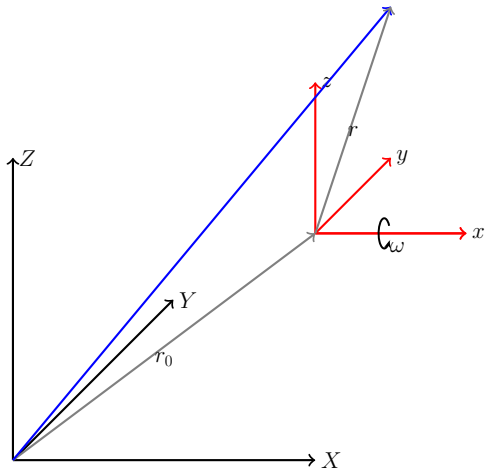


Figure 2.4: Stationary (black) and rotating (red) reference frame.

Figure 2.4 shows a stationary and rotating frame of reference. The position of the rotating reference frame relative to the stationary is denoted $\vec{r_0}$.

The rotating reference frame rotates with angular velocity ω. In a point $\vec{r}$ relative to the rotating frame of reference, the velocity relative to the rotating reference frame is

$$\vec{v_r} = \vec{v} - \vec{u_r}. \tag{2.39}$$

Where $\vec{v}$ is the velocity viewed from the stationary frame of reference and $\vec{u_r}$ is the velocity due to the moving frame, found in Equation 2.40,

$$\vec{u_r} = \vec{\omega} \times \vec{r}. \tag{2.40}$$

When solving the equations of motion in the rotating reference frame the accelerations are enlarged by adding terms in the momentum equations. The momentum equation can be expressed using the absolute velocities as dependent variables. The governing equations of fluid flow for a steady rotating reference frame can be written:

$$\begin{aligned} \frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \vec{v_r}) &= 0, \\ \frac{\partial}{\partial t}(\rho \vec{v}) + \nabla \cdot (\rho \vec{v_r} \vec{v}) + \rho(\vec{\omega} \times \vec{v}) &= -\nabla p + \nabla \bar{\bar{\tau}}, \\ \frac{\partial}{\partial t}(\rho E) + \nabla \cdot (\rho \vec{v_r} H + p \vec{u_r}) &= \nabla \cdot (k \nabla T + \bar{\bar{\tau}} \cdot \vec{v}). \end{aligned} \tag{2.41}$$

Equation 2.41 is based on Equation 2.9 for stationary frame of reference. At the interface between the rotating and stationary region a local reference frame transformation is performed. (ANSYS, 2013)

2.5.2 Sliding Mesh Model

The most accurate and computationally demanding simulation technique for multiple reference frames is the sliding mesh technique. Sliding mesh is an unsteady approach that is used when the interaction between stator and rotator is strong. The meshes moves relative to each other with an interface zone between, while they move relative to each other node alignment is not required. The fluxes needs to be computed across the two non-conformal interface zones for both zones.

2.5.3 Arbitrary Mesh Interface

Arbitrary mesh interface (AMI) is used to enable simulations across disconnected adjacent mesh domains. For instance in a system with rotating parts there is one mesh for the rotating domain and one for the static. In the interface the faces accepts weighted contribution from partially overlapping regions.

2.6 Computational Fluid Dynamics on Propellers

When doing computational fluid dynamic analyses on propellers some challenges needs to be addressed. There are large pressure and velocity gradients, challenging geometries and the propeller rotates relative to the incoming water and thruster body. To be able to capture the large gradients a dense mesh is important. The geometry is also challenging with the relative thin blades and sharp corners. The boundary layer thickness will wary along the span of the blade, due to the difference in velocity and chord length. The mesh needs to be able to capture this rotation. There are several techniques to do that, sliding mesh and moving reference frame are two common techniques.

2.6.1 Open Water Characteristics

Open water characteristics can be found either through open water model tests or open water simulations. In open water tests the propeller is tested with undisturbed inflow. Outcome from the open water tests are thrust T and torque Q presented in an open-water diagram, showed in Figure 2.5.

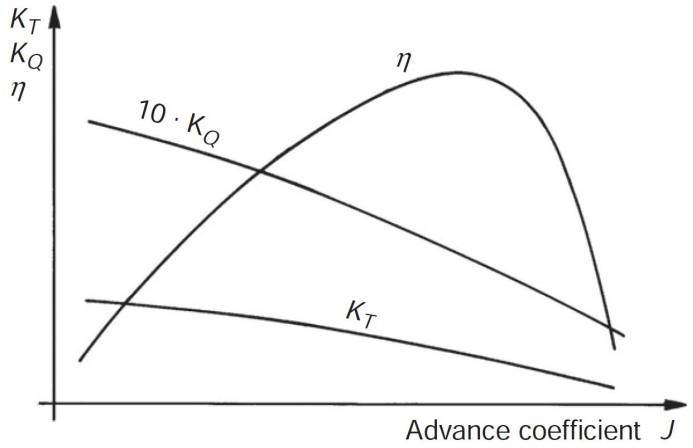


Figure 2.5: Open-water diagram found in (Bertram, 2012).

Where K_T is the dimensionless thrust and is defined in Equation 2.42. Here ρ is the density of the water, D is propeller diameter and n revolutions per second. K_Q is the dimensionless torque and is defined in Equation 2.43.

$$K_T = \frac{T}{\rho D^4 n^2} \tag{2.42}$$

$$K_Q = \frac{Q}{\rho D^5 n^2} \tag{2.43}$$

The open-water diagram also presents the open-water propeller efficiency, it is defined as the useful thrust power divided by the shaft power, showed in Equation 2.44.

$$\eta_0 = \frac{P_T}{P_D} = \frac{J}{2\pi}\frac{K_T}{K_Q} \tag{2.44}$$

The dimensionless thrust and torque and the open-water propeller efficiency are plotted against the advance coefficient J defined in Equation 2.45.

$$J = \frac{V_a}{nD} \tag{2.45}$$

(Bertram, 2012)

Chapter 3

Software

Below is a brief overview of the software used in this thesis. OpenFOAM is used as fluid flow solver, BOXERMesh is used for grid generation and ANSYS Fluent for validation.

3.1 OpenFOAM

OpenFOAM or Open Field Operation and Manipulation is used as solver. It is an open source toolbox developed at Imperial College in London. It includes different solvers, libraries, and utilities. The version used in this thesis is v1606+. There is no graphical user interface (GUI), only a folder structure. This structure is dependent on the problem solved, a simulation of fluid flow using the $k - \omega$ SST turbulence model is showed in Figure 3.1. The folders contain scripts written in C++. Since the source code is open and easy accessible it is possible to generate new functions or change exiting ones to get the information desired efficiently from each simulation.

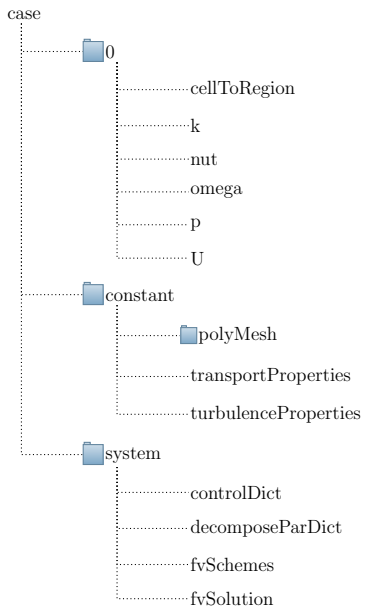


Figure 3.1: Folder system in OpenFOAM

The 0 folder contains the initial and boundary conditions, the system folder contains the information about the solving process and the constant folder contains the mesh, transport properties and turbulence properties. In addition to the folders shown there will be additional time directories after the simulation is done.

3.1.1 ParaView

Parts of the post processing is done using ParaView. ParaView is an open source post processing utility included in the OpenFOAM library. It imports the OpenFOAM cases directly and makes it possible to visualize the results. A Python script is used to generate comparable figures efficiently. The rest of the post processing is done using Python scripts and Matplotlib.

3.2 BOXERMesh

BOXERMesh is a grid generator developed by Cambridge Flow Solutions. It has both GUI and scripting possibilities. The domain is meshed using a Cartesian grid with an Octree structure. Different refinement levels are specified for volumes, faces and edges. Around the body a prismatic body-conformal mesh replaces the cut-cells (Figure 3.2). In meshes with multiple regions the interfaces are conformal. It is fully parallelized and robust, complex geometries can be meshed easily. The scripting possibility, using Lua scripts, makes it possible to implement the meshing in an automated system.

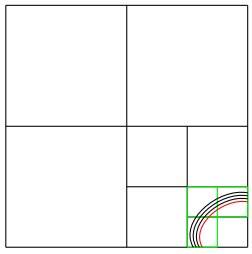


Figure 3.2: Schematic drawing of cell divisions. For each layer of division the cells are divided into eight cells. The red line is the body and the green cells are the cut-cells.

3.3 ANSYS Fluent

To verify the results ANSYS Fluent is used. This is a trusted program used by Rolls-Royce for years. In this thesis version 17.0 is used. ANSYS is a license based multi-physics software which provides a broad range of analysis tools. The whole process from model generation, mesh generation, problem definition, solving and post processing can be performed using ANSYS. ANSYS Fluent models fluid flow.

Chapter 4

Method

To investigate the possibility of changing the solver in POWS, the solver settings, schemes and mesh are considered. Both steady state and transient simulations are performed. Before any work directly related to POWS is done, a two-dimensional test case is set up.

4.1 Two-Dimensional Foil

To gain confidence in OpenFOAM and to investigate the mesh sensitivity, a two-dimensional case is made. Two-dimensional cases are a good starting point, when setting up simulations. It is useful to ensure that the system works as intended, before starting on complex three dimensional cases. Different parameters can be investigated quickly. This case is based on my project thesis. The same two-dimensional foil, NACA4412, is used and the initial mesh is the same as for the project thesis. The chord length of the foil is 1 m. The mesh is generated in ANSYS Workbench. Equal meshes and different solvers, ANSYS Fluent and OpenFOAM, are used to investigate the mesh sensitivity of the solvers. For the solver parameters default settings are used, the turbulence model is $k - \omega$ SST and the boundary conditions are similar for the two software. The lift and drag coefficients are compared to data from Abbott and Von Doenhoff (1959).

Using the mesh from the project thesis, evenly spaced angles of attack from -4 to 16 degrees with 4 degrees as interval is simulated using both ANSYS Fluent and OpenFOAM. Then the effect of first layer height, number

of inflation layers and nose refinement is investigated. To investigate the effect of the first layer height, it is varied from 2 mm to 22 mm with steps of 2 mm. The next investigation is numbers of inflation layers. To keep the total layer height similar, the growth rate is changed as the numbers of inflation layers are changed. The number of inflation layers and growth rate investigated are showed in Table 4.1.

Table 4.1: Inflation layers

Number of inflation layers	14	12	10	8	7
Growth rate	1.05	1.10	1.15	1.20	1.25

The last mesh parameter checked is number of cells at nose. It is varied from 5 to 50 cells with 5 cells as interval. The rest of the mesh is changed to make a smooth transition between the nose and the rest of the foil. The mesh density is highest at the nose and the trailing edge. All mesh tests are performed for angle of attack of 4 degrees. The meshes investigated are presented in Appendix B.8

4.2 OpenFOAM Setup

This section explains how the three-dimensional propeller simulations are set up using OpenFOAM. There are some differences between the steady state and transient simulations, first the steady state setup is explained and then the differences are stated. For the steady state simulations, the SIMPLE algorithm and MRF are used. For the transient simulation, the PIMPLE algorithm is used together with sliding mesh.

4.2.1 Steady State

To set up the steady state simulations, the folder system showed in Section 3.1 is used. The *0* folder contains the initial and boundary conditions. For the $k-\omega$ SST turbulence model the files are: turbulent kinetic energy k, eddy viscosity μ_T, specific dissipation rate ω, pressure p and velocity U. The internal field and the values on each patch are given.

The boundary conditions used are *cyclicAMI* or cyclic arbitrary mesh interface at the interface between the rotating and static domain. The inlet

is a velocity inlet and the outlet is a pressure outlet. *symmetryPlane* is applied on the rest of the outer walls of the domain. The no-slip wall boundary condition is applied on the thruster.

In the *constant* folder, the mesh is included in addition to the transport and turbulence properties. The *transportProperties* file states the transport model and material properties for the fluid. In the *turbulenceProperties*, the turbulence model is specified. The $k - \omega$ SST turbulence model is chosen based on the common practice in Rolls-Royce Marine and the findings from my project thesis. In addition to the folders shown in Figure 3.1, there is an additional folder in *constant* for systems with multiple domains moving relative to each other. For MRF simulation this is called *MRFProperties*, where the rotational axis, angular velocity and origin are specified.

The *system* folder contains information about the solving process. For steady state simulations, the most important information in *controlDict* is the solver and numbers of iterations. For parallel computing the decomposition of the case is also included in the system folder, where the numbers of processors and method are stated. *fvSchemes* contains the numerical schemes used in the computation. Schemes specified are time, divergence, gradient, Laplacian, interpolation and surface normal gradient. In the *fvSolution* file the parameters considered are *nNonOrthogonalCorrectors* and under-relaxation factors. The scheme and solution options will be more thoroughly discussed in Section 4.5.1. The simulations are initialized using potential flow calculations, this is specified in the *fvSolution* file.

After the simulation, there will be additional time directories, for steady state simulation normally only the final iteration is saved. The simulations are stopped when converged using a convergence check script provided by my supervisor Jonas Eriksson.

4.2.2 Transient Simulations

The rotational motion is modelled using sliding mesh instead of MRF, for transient simulation. Due to this the *MRFProperties* dictionary is replaced by *dynamicMeshDict*, the specification of the rotational motion is the same as for steady state. This is the only change in the *constant* folder.

Most of the changes are in the *system* folder. In *fvSchemes* the time scheme is different, but the rest of the schemes are the same. In the *controlDict* the important parameters are the solver, start time, time step and end time. The biggest difference is in the *fvSolution*. The PIMPLE algorithm is used, this algorithm requires more input than the SIMPLE algorithm. The input will be discussed in Section 4.5.2.

4.2.3 Post Processing

Functions are included in the simulation folders to write the data for residuals, forces, and moments to files. These files are read by Python scripts and plotted automatic after each simulation. The Python scripts used are based on scripts by Jonas Eriksson and modified to suit the needs in this thesis.

To visualize the mesh, pressure, and velocity, ParaView is used. A script is written to create a set of figures automatic. This is done to save time and get comparable figures.

4.2.4 Automation

The meshing, OpenFOAM set up and post processing is performed using a bash script in the Linux terminal. The bash script changes dummy variables in a meshing script, then runs the script. The script generates the mesh using BOXERMesh and this takes some time. This must be done before the OpenFOAM simulations are set up and is one of the reason why the automated script is useful. It is also practical to be able to run several different simulations sequential without any manual work.

The meshes for the rotating and static regions are transferred to OpenFOAM separately. After this, patches are renamed and the meshes are merged. Then different simulations for different advance ratios are set up. This is done by copying an empty OpenFOAM case with dummy variables and changing the values for inlet velocity, angular velocity and turbulence properties.

To run the simulation, the case is decomposed to a given number of processors, then the simulation is initialized using potential theory and run. A convergence script monitors the simulation and stops if convergence is

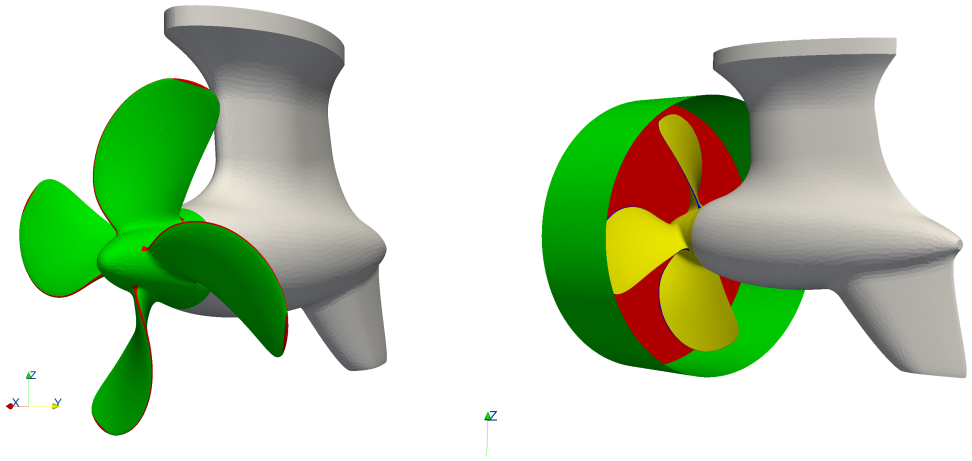
achieved. After this, the Python plotting functions and ParaView visualization are performed automatically.

4.3 Meshing

The mesh pused for POWS with ANSYS as solver did not converge using OpenFOAM. A reason for this is the non-conformity at the interface. At the interface, between the rotating domain around the propeller and the static outer region, an interpolation is performed. A conformal mesh makes this interpolation easier. Another possible reason for the stability problem is the leading edge of the propeller. Here it is most difficult to resolve the geometry and flow gradients. Low mesh quality can occur in this region. Due to the challenge with convergence, a new mesh generation script is made. The same software BOXERMesh is used, but the mesh for both regions are made simultaneously. This is required to make the mesh conformal.

The goal for the mesh generation is to make a script that meshes different geometries automatic. This is done using BoxerMesh and the scripting language is Lua. The script is generated and tested on the Azipull100 and used on Azipull120 and Azipull150 for validation. In addition to being stable and general the mesh should be computationally inexpensive to use, as it is meant for a program that continuously computes propeller characteristics.

The mesh is a hybrid between Cartesian and prismatic mesh. The outer parts of the domain is Cartesian mesh and the layer close to the propeller is prismatic layer. The geometrical models have six patches, three on the thruster and three on the interface. The patches on the thruster are the thruster body, the propeller and a refine patch along the edge of the blade. The different patches is showed for Azipull100 in Figure 4.1a and the interface is showed in Figure 4.1b.



(a) Patches of Azipull100. (b) Interface on Azipull100.

Figure 4.1: Patches and interfaces on Azipull100.

Figure 4.2 shows the domain from the side. The size of the domain is 10 propeller diameters in front, below and on both sides and 25 diameters behind the propeller. The distance to the top of the domain d is decided by the propeller house geometry. When generating the mesh, the domain size and size of the biggest cells are specified. Then the number of divisions are specified for surfaces, volumes, and edges. The refinement on the edges are specified by angle or radius of curvature.

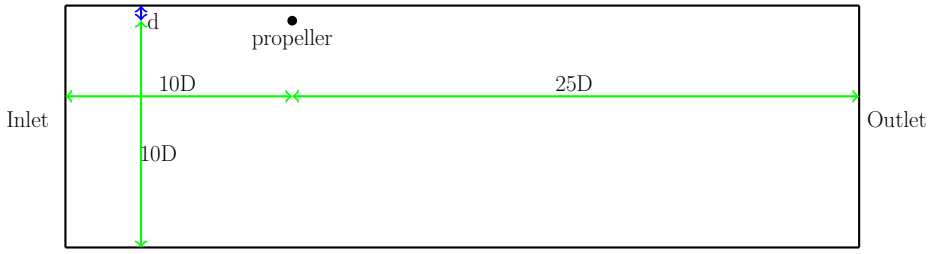


Figure 4.2: Schematic drawing of the propeller domain. The distance d displayed with the blue arrow is the distance between the top of the propeller house and the propeller center. This is specified for each thruster.

The mesh on the propeller blade edges are important to ensure stability

and accurate results. Here the number of divisions are the greatest. Other critical parts are the edges on the thruster body. The prismatic boundary layer is meshed using a layer with fixed initial size, growth rate and number of layers.

4.4 Numerical Schemes

The mesh applied in the scheme investigation is not the same mesh as the validation is based on. The scheme and solver test were performed on a mesh with 30% less cells. This is done to save computation time and to test the stability on a more demanding mesh. The density of the mesh is lower at the hub, leading edge and at the edges of the thruster body. Mesh statistics for the mesh is found in Table 4.2.

Table 4.2: Mesh statistics for the test mesh.

Number of cells	7.72e6
Maximum mesh non-orthogonality	83.0
Average mesh non-orthogonality	16.1
Maximum aspect ratio	164
Maximum skewness	3.12

First a standard set of schemes based on common practice and recommendations were made, this scheme is showed in Appendix B.2. A scheme that is known to work well is chosen to better see the influence of the parameters changed. Different changes in the *fvScheme* dictionary is performed to see the effect. The parameters compared are propeller thrust, propeller moment, convergence rate and stability. The changes tested are:

- `div(phi,U) bounded Gauss linearUpwind grad(U)`
- `div(phi,U) bounded Gauss limitedLinearV 1`
- `div(phi,U) bounded Gauss LUST grad(U)`

- `snGradSchemes` **`default`** `limited corrected 0.5`
- `snGradSchemes` **`default`** `limited corrected 0.33`

- `grad(U/p)` `cellMDLimited Gauss linear 1`

```
- grad(U/p)          cellLimited Gauss linear 1
- grad(U/p)          faceMDLimited Gauss linear 1
- grad(U/p)          faceLimited Gauss linear 1
```

The first three schemes are divergence schemes for the convective U term. The next two are surface normal gradient schemes and the last four are gradient schemes for U and p. The *fvSchemes* investigated are equal to the standard scheme except for one of the changes above in each run. This does not take interaction between different schemes into account, but it is clear where the consequence of each change originates.

Three advance ratios are tested, one before the maximum propeller efficiency, one close to the maximum and one after. The changes in the schemes can be divided into three different scheme types, surface normal gradient schemes, divergence schemes and gradient schemes. Other schemes as interpolation and time schemes are not considered in this thesis. For time schemes the *steadyState* option is used for the steady state simulation and for the transient simulation the Euler is used. For the interpolation, different orders of interpolation is available. The linear is widely used and will be used in this thesis.

4.4.1 Surface Normal Gradients

Surface normal gradient and Laplacian schemes are treated together, the choices are the same and they are mesh dependent. For Laplacian schemes the *Gauss* is the only choice for the discretisation, there are several interpolation schemes, but *linear* is usually used. This means that the only thing to decide for the Laplacian scheme is the surface normal gradient schemes which is treated below. The Laplacian schemes are changed as the surface normal gradient schemes are changed.

Three different schemes are tested. All three schemes are corrected but the limiting factor differ. The cases tested are unlimited, limited with limiting coefficient $\psi = 0.5$ and $\psi = 0.33$. The purpose of the limiters is to limit the size of the correction compared with the orthogonal part. As explained in Section 2.2.1 the orthogonality correction plays an important role in regards of stability and accuracy. For $\psi = 0.5$ the non-orthogonal part is smaller than the orthogonal part and for $\psi = 0.33$ the non-orthogonal part is smaller

than half of the orthogonal part. This is expected to make $\psi = 0.33$ more stable and $\psi = 0.5$ more accurate. More schemes are available, but they are not applicable for meshes with as high non-orthogonality as this mesh has. Greenshields (2015)

4.4.2 Divergence Schemes

The divergence schemes tested for the convective U term are *linearUpwindV*, *linearUpwind*, *limitedLinearV* and $LUST$. *linearUpwind* is a second order scheme which is upwind-biased and *limitedLinear* tends towards upwind for high gradients. A coefficient is specified between 0 and 1 (0 is *linear* and 1 gives the strongest limiting). $LUST$ is a blend of 75% *linear* and 25% *linearUpwind*. Linear is a second order unbounded scheme. For vector fields a V-scheme is an option. In the schemes without the V option enabled, the vectors are limited separately in each direction. With the V option the strongest limiting (based on the direction with the highest gradient) is used in all directions. This is a more stable and a less accurate approach. All divergence schemes tested are bounded, meaning $\nabla \cdot U \neq 0$ is allowed before steady state is achieved. This contributes to keep the solution variables bounded. (Greenshields, 2015)

4.4.3 Gradient Schemes

For the gradient schemes, all four limiting schemes are tested together with the Gauss linear scheme, *cellMDLimited*, *cellLimited*, *faceMDLimited* and *faceLimited*. In the cell limited schemes the gradients are limited along the line between the two adjacent cell centers. For face limited schemes, the gradients are limited at the face. Cell limited schemes are normally less dissipative than the face limited schemes. MD or Multi-dimensional means that the gradient is clipped in the direction normal to the face. This is less dissipative and less stable than clipping the gradients equally in all directions. Sorted from least to most dissipative the limiting schemes are:

- cellMDLimited,
- cellLimited,
- faceMDLimited and

- faceLimited.

To ensure boundedness and stability, limiters can be used on meshes with high non-orthogonality. The mesh which the schemes are tested on have high non-orthogonality.Sideroff (2010)

4.5 Solving

The same schemes can be used for steady state and transient simulations, except for the time scheme. For the solving process, there are more differences as different algorithms are used. For the steady state simulation the SIMPLE algorithm is used and for the transient simulation the PIMPLE algorithm is used, these algorithms are discussed in Section 2.2.3.

4.5.1 Steady State Simulation

For steady state the under-relaxation α is the only parameter investigated. The under relaxation is set different for the pressure and the velocity, k an ω. For pressure $\alpha_P = 0.5$ and $\alpha_P = 0.3$ are tested. For velocity, k and ω, $\alpha = 0.8$ and $\alpha = 0.5$ is tested. The two highest and the two lowest under-relaxation factors are tested together. Two schemes are tested to see the effect of the under-relaxation. The two most promising schemes are used, *linearUpwindV* and *limitedLinearV*.

The *nNonOrthogonalCorrectors* is used to specify the numbers of time the pressure correction equation is solved. It is set to 3 times as the non-orthogonality is high.

4.5.2 Transient simulations

The same schemes as for the steady state simulation are used, except for the time derivative, where *Euler* scheme is used. The transient calculation is performed on the Azipull120 and the advance ratio is at the top of the efficiency curve. The advance ratio calculated for is the top of the open water propeller efficiency curve. In *controlDict* the time step is chosen to make the rotating domain turn one degree for each time step. PIMPLE dynamic mesh is used instead of SIMPLE. As explained in Section 2.2.3, PIMPLE is a combination of SIMPLE and PISO. PIMPLE solves time steps as PISO, but

inside each time step several iterations with under-relaxation as in SIMPLE is performed. The parameters in the *fvSolution* dictionary must be changed to use the PIMPLE algorithm. If default values are used, PISO is run. PISO requires smaller time steps than are applicable here. The Courant number would reach too high values for stability to be possible.

The *nOuterCorrectors* gives the number of times the pressure velocity coupling is calculated, here 50 iterations are used. Inside each of the pressure velocity couplings the *nCorrectors* are used to specify the numbers of iteration for the pressure field, 3 iterations are performed. In additions to these options the *nonOrthogonality* option can be used for each time step, as in the steady state calculations. The numbers of time for the pressure correction equation is 2 fr the transient simulation. The computational effort demanded to solve the system is affected by the parameters mentioned above. Some reduction can be obtained by including a *residualControl*, this stops the iterations when a residual tolerance level is met.

The SIMPLE algorithm is used inside each time step to apply under-relaxation on the equations. This makes the calculations more stable. The lowest under-relaxation level, applied the steady state simulation is applied on the transient simulation.

4.6 Time

To get an idea about the solver speed, a simulation with the final Azipull120 mesh is performed using a coupled solver in ANSYS Fluent. In ANSYS Fluent, hybrid initialization and second order upwind scheme is used. The simulation is run on 96 cores on the HPCC, same as for the OpenFOAM simulations. The aim for this investigation is to compare current POWS, with OpenFOAM in regards of solver time.

4.7 Validation Data

The validation data for the Azipulls are from three different model test facilities in addition to data from numerical simulations. The numerical results

are from POWS, with the current solver and mesh.

4.7.1 Azipull120

The experimental data for Azipull120 is from MARINTEK (Alterskjær, 2010) and HSVA (Klug, 2007), both reports are classified. The model test data corrected for scale are plotted in Figure 4.3. Full scale predictions from POWS are also used as validation. To see how the results relate to POWS calculations.

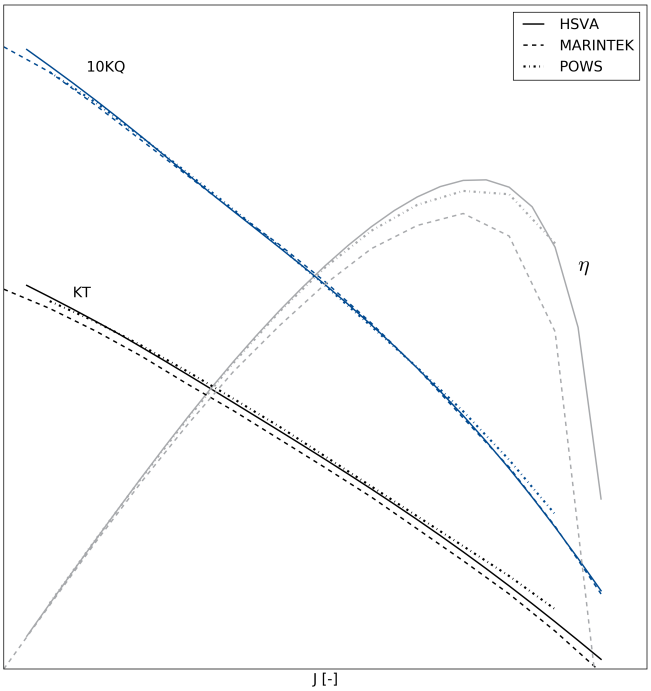


Figure 4.3: Full scale predictions

4.7.2 Azipull150

The data for the Azipull150 thruster is from MARIN and found in the classified report by Dang and Radstaat (2013). Full scale predictions from MARIN

and numerical results from POWS are showed in Figure 4.4.

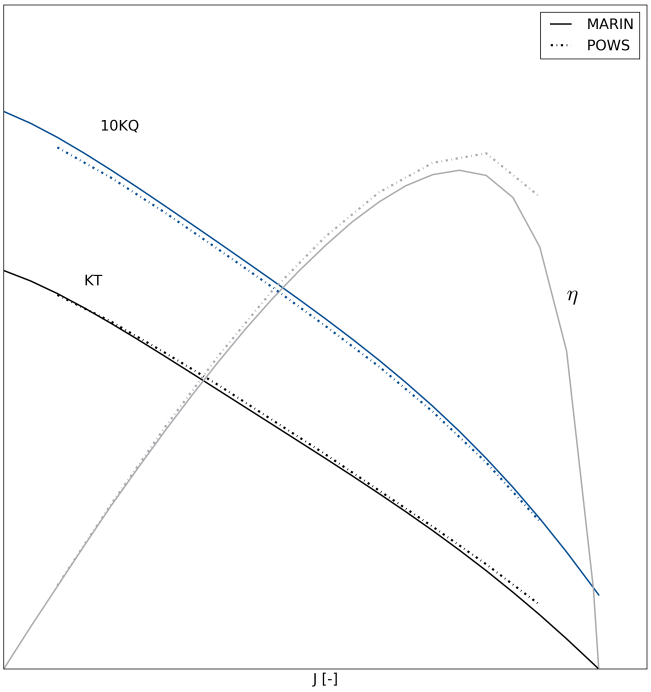
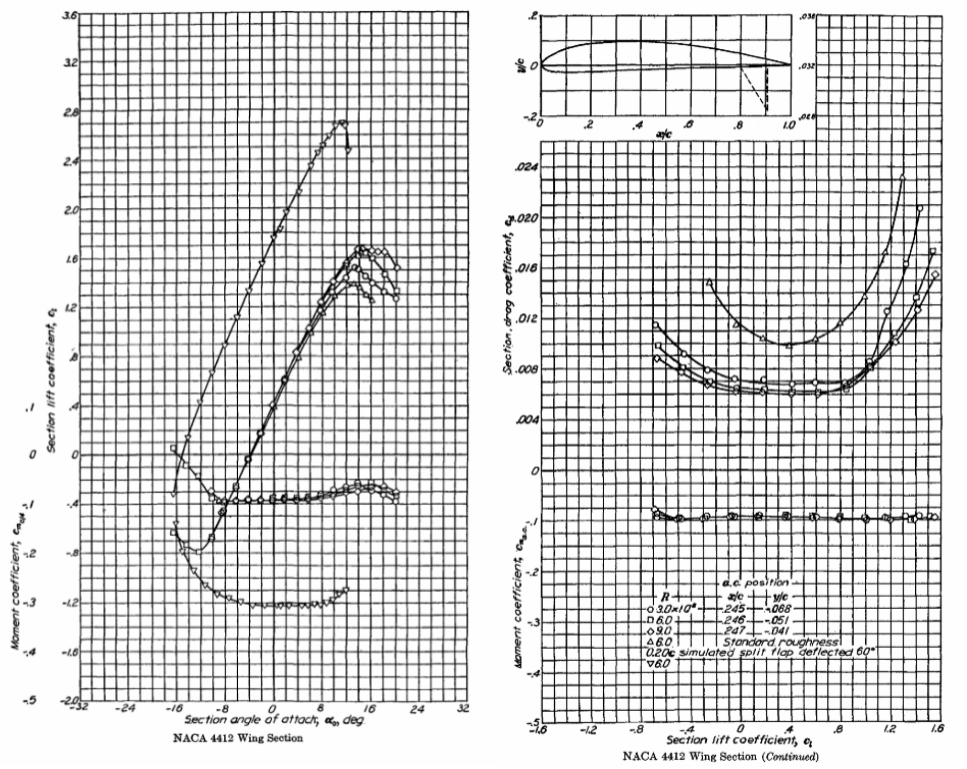


Figure 4.4: Model and full scale predictions.

4.7.3 Two-Dimensional Case

The validation data from the two-dimensional NACA4412 foil is found in Abbott and Von Doenhoff (1959). The data is showed in Figure 4.5.



(a) Lift coefficient (b) Drag coefficient

Figure 4.5: Airfoil data for NACA 4412 from Abbott and Von Doenhoff (1959).

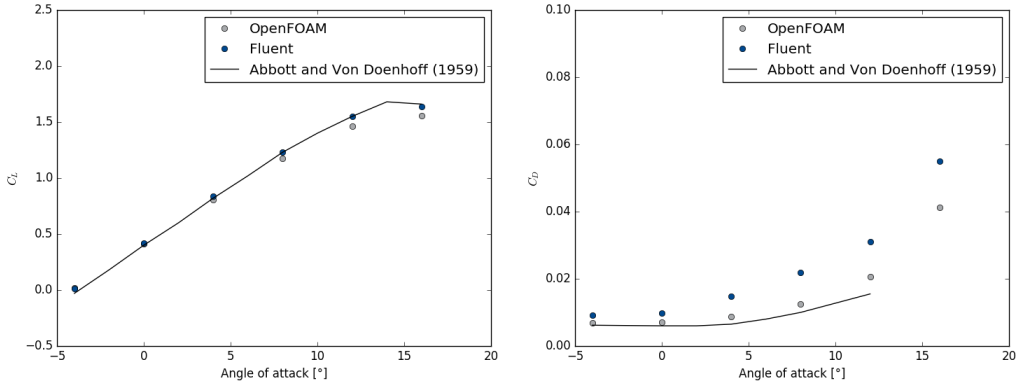
Chapter 5

Results

This chapter first presents the results for the two-dimensional case. Then the results from the mesh, scheme and solver investigations are presented. The solver time is briefly considered, before the validation cases are presented. The results presented are limited by the non-disclosure agreement with Rolls-Royce.

5.1 Two-Dimensional Foil

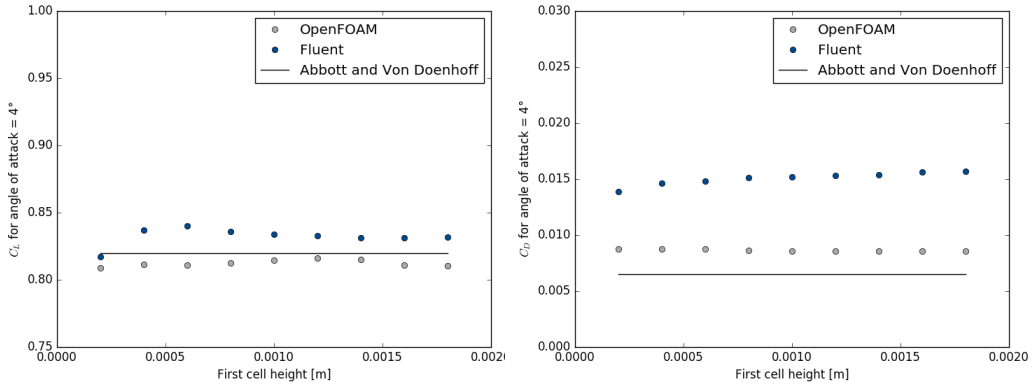
Figure 5.1 shows the lift and drag coefficients for angle of attack from -4 to 16 degrees. OpenFOAM, ANSYS Fluent, and experimental results from Abbott and Von Doenhoff (1959) are presented. The difference between OpenFOAM and ANSYS Fluent increases as the angle of attack increases. For the lift coefficient, the Fluent results are in most agreement with the experimental results. For the drag coefficient, OpenFOAM is in most agreement with the experimental results.



(a) Lift coefficient. (b) Drag coefficient.

Figure 5.1: Lift and drag coefficients for angle of attack from -4 to 16 degrees.

In Figure 5.2 difference in C_L and C_D due to changes in first layer height is showed. The lift and drag coefficients are not severely affected by the first layer height. All parameter investigations are done for angle of attack of 4 degrees.

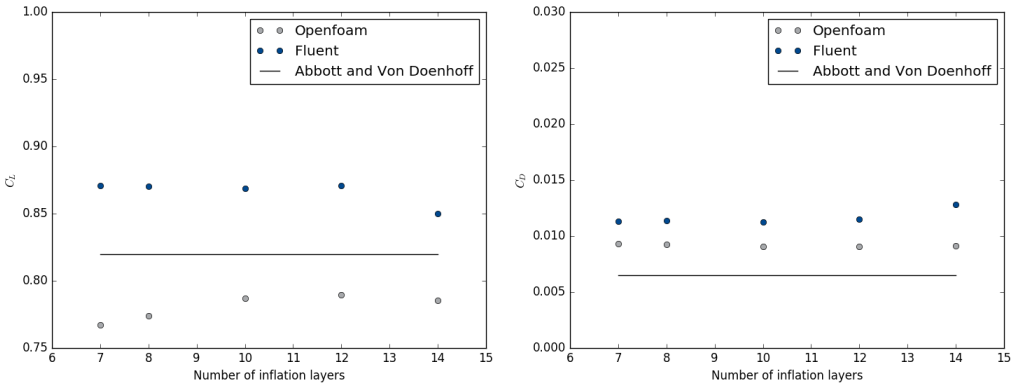


(a) (b)

Figure 5.2: Effect of first layer height.

For the investigations of number of inflation layers and numbers of cells on the leading edge, problems with divergence in OpenFOAM are experienced. To make the simulations more stable, potential flow calculations are used to

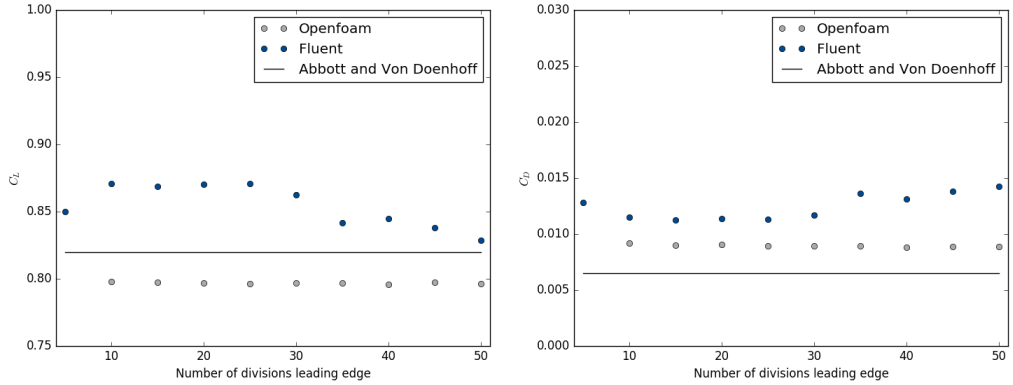
initialize the simulation. For 14 inflation layers, the first 200 iterations are also solved using *upwind* scheme. This is a stable and inaccurate scheme.



(a) (b)

Figure 5.3: Investigation of inflation layers at angle of attack 4 degrees.

After the study of the inflation layer, the number of cells on the nose is investigated, the results are presented in Figure 5.4. The mesh with fewest number of cells at the nose diverged for OpenFOAM. Here using potentialFOAM initialization and upwind for the first 200 iterations did not increase the stability enough. As the nose is refined, the trailing edge is also refined.



(a) (b)

Figure 5.4: Nose refinement at angle of attack 4 degrees.

5.2 Mesh

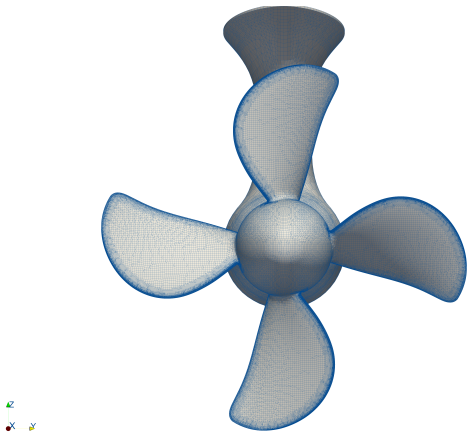
The same meshing script is used to mesh both Azipull120 and Azipull150, the refinement levels is shown in Table 5.1. The division in the z-direction is specified, it is 16 for the Azipull120 and 17 for the Azipull150, this is done to have similar outer cell size. The edge refinement on the thruster body is specified to act on areas with curvature smaller than 5 degrees and radius of curvature smaller than 120 mm.

Table 5.1: Refinement levels for different areas

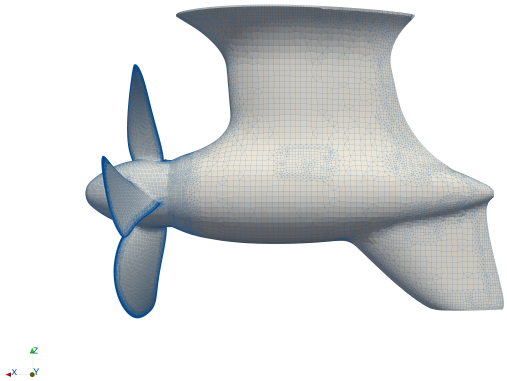
Propeller blade and hub	8
Leading edge	11
Trailing edge	10
Thruster body	5
Thruster body edges	6
Volume refinement around body	5
Interface	5
Interface edge	7

5.2.1 Azipull120

The grid generated for the Azipull120 propeller is showed in Figure 5.5. The grid is most dense at the propeller edges and have the same size on the rest of the propeller blades and hub as seen in Figure 5.5a. The refinement levels can be found in Table 5.1. The edge refinements for the thruster body is showed in Figure 5.5b.



(a) Front view of mesh on Azipull120.



(b) Side view of mesh on Azipull120.

Figure 5.5: Body fitted mesh on the Azipull120

Figure 5.6 shows the whole domain from the side. It can be observed that there are some cells of each refinement level between the coarsest level and the refined areas. This is done to have smooth transitions between the regions. A close up of the domain is showed in Figure 5.7. Here the refinement of the edge on the interface is visible in addition to the refined volume around the body. Table 5.2 shows some information about the Azipull120 mesh.

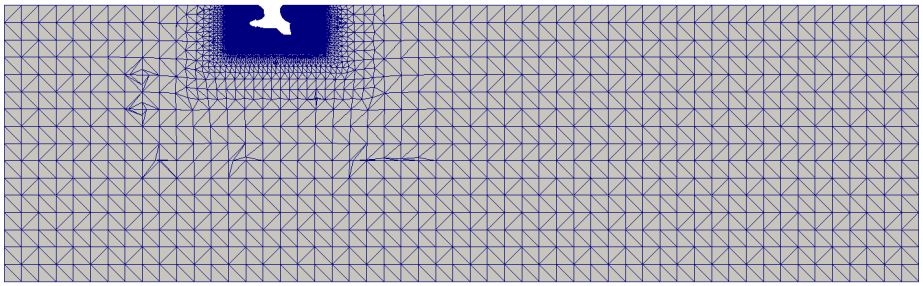


Figure 5.6: Far field of the domain for Azipull120

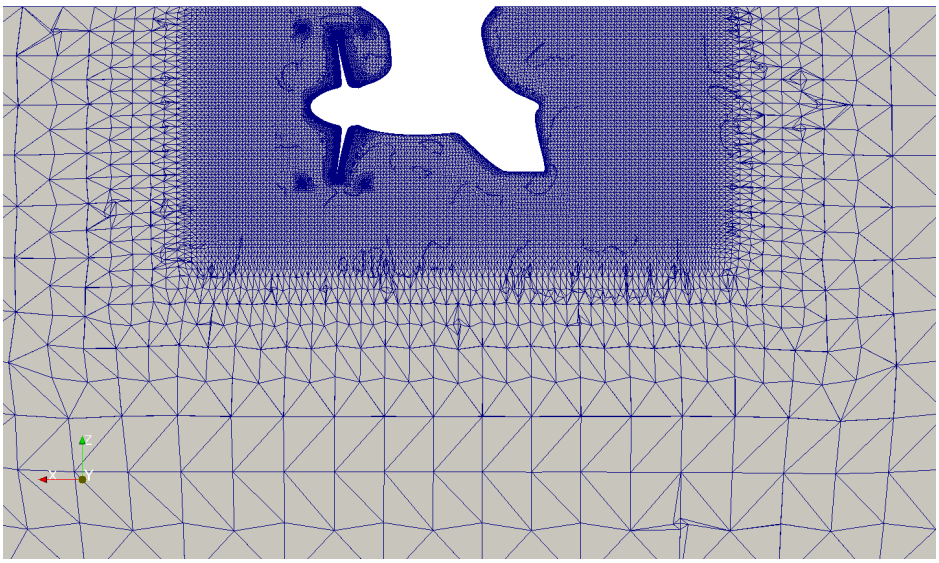


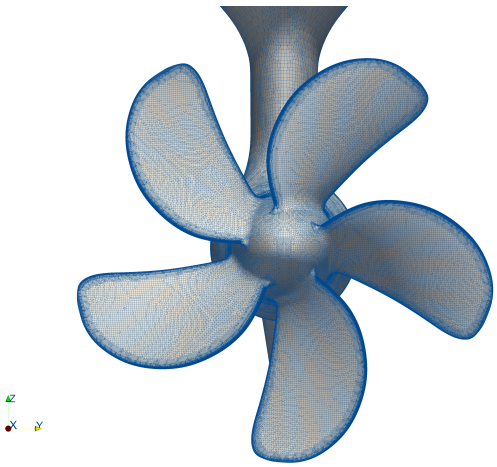
Figure 5.7: Close up of the domain for Azipull120

Table 5.2: Mesh statistics for the Azipull120 mesh.

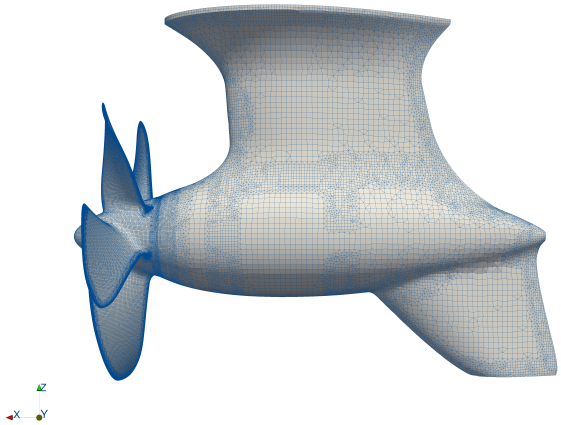
Number of cells	10.8M
Maximum mesh non-orthogonality	85.9
Average mesh non-orthogonality	16.1
Max aspect ratio	150

5.2.2 Azipull150

Figure 5.8 shows the mesh for the Azipull150. Figure 5.8b shows the edge refinement on Azipull150, compared with Figure 5.5b it can be observed that the edge refinement criterion is met for a larger area on the Azipull150.



(a) Front view of mesh on Azipull150.



(b) Side view of mesh on Azipull150.

Figure 5.8: Body fitted mesh on the Azipull150

The whole domain and a close up of the domain is showed in Figure 5.9 and Figure 5.10 respectively. Information about the numbers of cells and quality of the mesh can be found in Table 5.3.

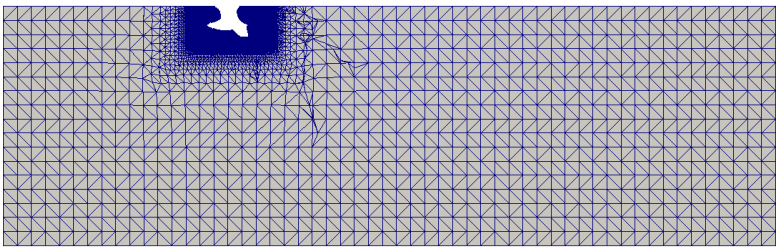


Figure 5.9: Far field of the domain for Azipull150

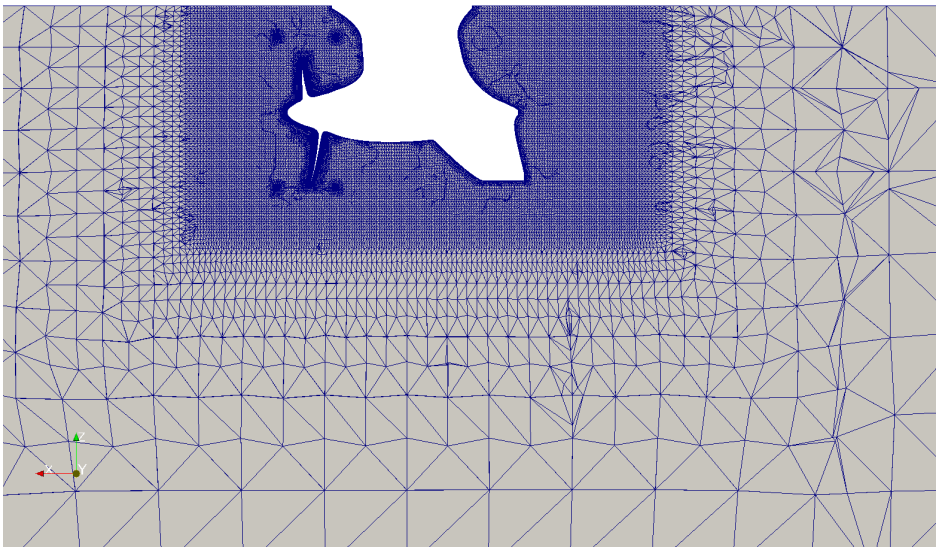


Figure 5.10: Close up of the domain for Azipull150

Table 5.3: Mesh statistics for the Azipull120 mesh.

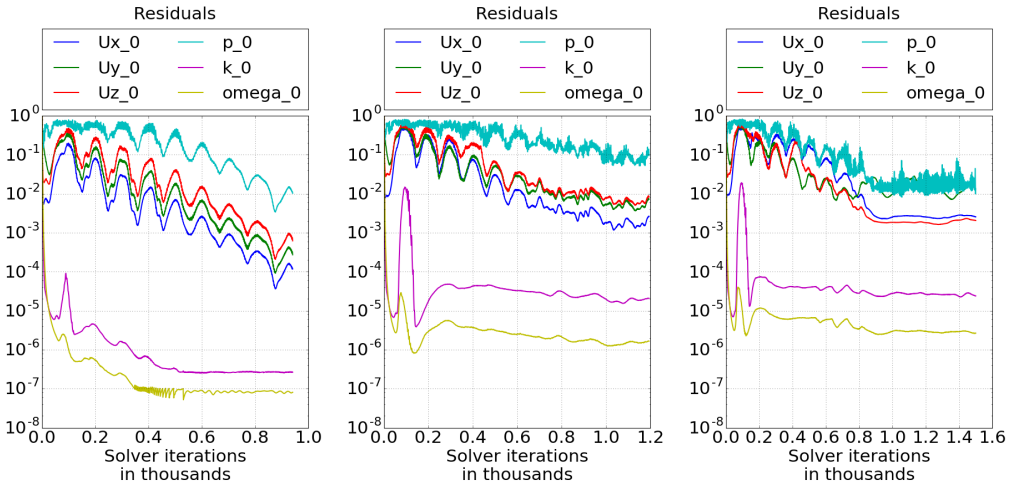
Number of cells	16.7M
Maximum mesh non-orthogonality	84.9
Average mesh non-orthogonality	16.0
Max aspect ratio	133

5.3 Schemes

In this section, the results from the different schemes are presented. The scheme categories investigated are the surface normal gradient, Laplacian schemes, divergence schemes and gradient schemes. All results are presented for the advance ratio at the top of the propulsion efficiency curve.

5.3.1 Surface Normal Gradient and Laplacian Schemes

Three different surface normal gradient schemes are investigated, all three are suitable for meshes where the non-orthogonality is high. The mesh used for scheme tests have maximum non-orthogonality of 83 and an average of 16. All three schemes are corrected but the limiting differ. The cases tested are unlimited, limited with limiting coefficient $\psi = 0.5$ and limiting coefficient $\psi = 0.33$. The residual plots are plotted in Figure 5.11. The final residuals are lowest for the *corrected* scheme and highest for $\psi = 0.5$.

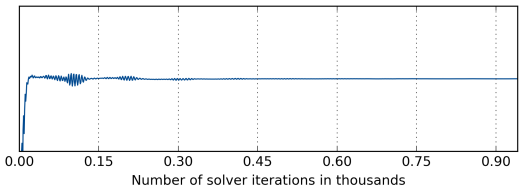


(a) $\psi = 1$/corrected (b) $\psi = 0.5$ (c) $\psi = 0.33$

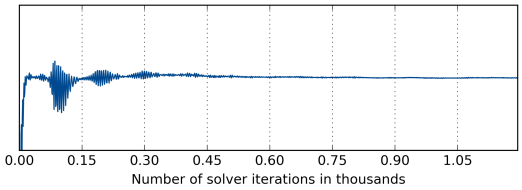
Figure 5.11: Residuals for the surface normal gradient schemes.

In Figure 5.12 the propeller thrust are plotted against the numbers of iterations. The limited scheme with limiting coefficient $\psi = 0.33$ has the most oscillations and the *corrected* has least. The iterations stop at a convergence criterion. If the criterion is not met, the simulation will stop at

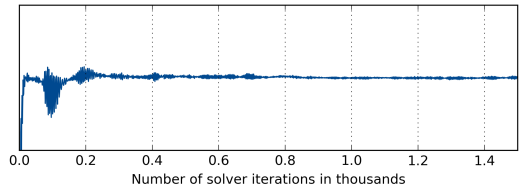
1500 iterations. Here the $\psi = 0.33$ calculation reached maximum number of iterations, due to propeller house force not converging. This is a small force compared to the propeller force, about 6%. The relative difference between the corrected scheme and the limited schemes with coefficient $\psi = 0.5$ and $\psi = 0.33$ for KT and KQ are presented in Table 5.4.



(a) $\psi = 1$/corrected



(b) $\psi = 0.5$



(c) $\psi = 0.33$

Figure 5.12: Propeller thrust from different surface normal gradient schemes. The y-scale is 100% of the mean of the 100 last iterations.

ψ	KT	KQ
0.5	0.5 %	0.5 %
0.33	1.8 %	2.0 %

Table 5.4: Percentage difference from the standard scheme.

Figure 5.13 shows the open water diagram for the three different schemes compared with the model test data from HSVA and MARINTEK. The numerical results are in good agreement with the experimental and closest to the results from HSVA.

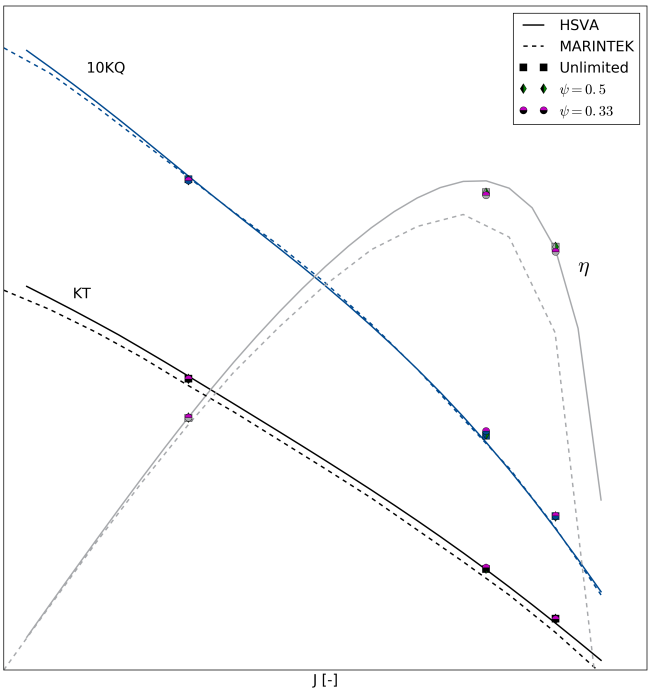


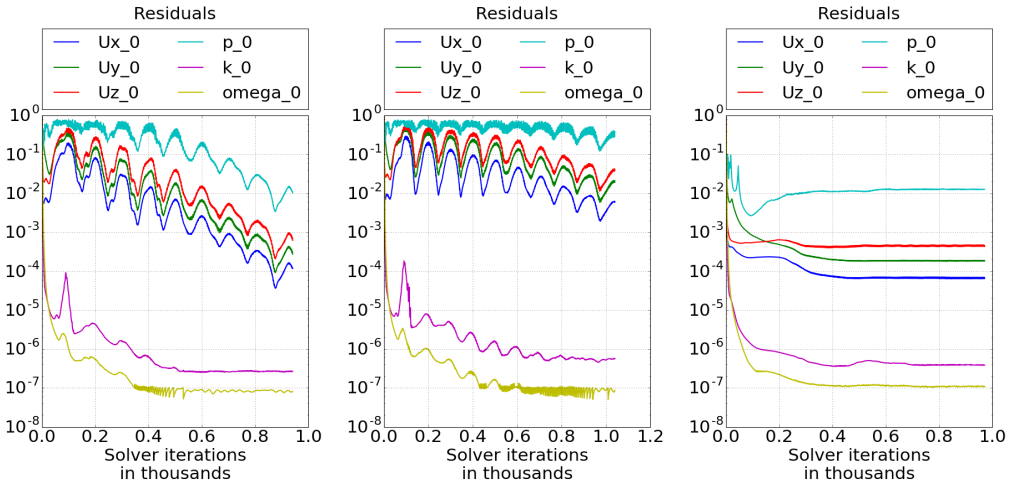
Figure 5.13: Open water diagram of the different limiters.

5.3.2 Divergence Schemes

The divergence schemes tested are:

- *linearUpwindV*,
- *linearUpwind*,
- *limitedLinearV* and
- *LUST*.

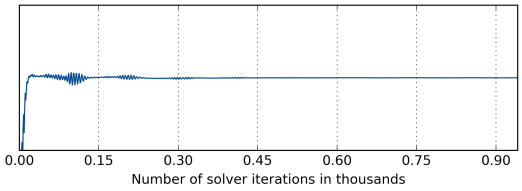
The *LUST* scheme is not presented as it diverged. The residual plots for each scheme is presented in Figure 5.14. The *linearUpwindV* and *linearUpwind* schemes have similar oscillations, but the residuals of the *linearUpwindV* scheme stop at a lower level and it reaches the convergence criterion with less iterations. The *limitedLinearV* shows no oscillatory motion, but the final residual is high.



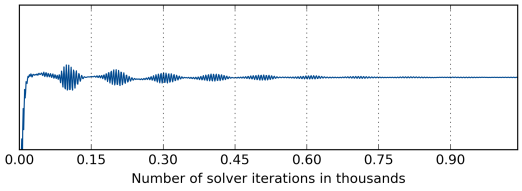
(a) linearUpwindV (b) linearUpwind (c) limitedLinearV

Figure 5.14: Residual plots for the divergence schemes.

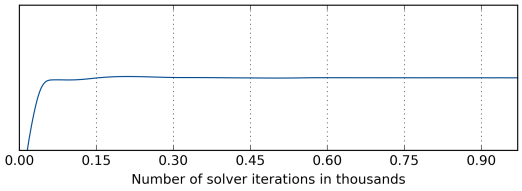
Figure 5.15 shows the propeller thrust for each iteration. *linearUpwind* has the strongest oscillations, while *limitedLinearV* does not have any visible fluctuations. The difference between the mean of the KT and KQ values of the 100 last iterations are showed in Table 5.5.



(a) linearUpwindV



(b) linearUpwind



(c) limitedLinearV

Figure 5.15: Propeller thrust force for different divergence schemes. The Y-scale is 100% of the sample average for all plots.

Scheme	KT	KQ
linearUpwind	-0.2 %	-1.1 %
limitedLinearV	0.2 %	0.1 %

Table 5.5: Percentage difference from the standard scheme.

Figure 5.16 shows the open water diagram for the propeller. The result presented are experimental results from HSVA and MARINTEK for the whole curve and numerical results from three points. The numerical results are presented for all four divergence schemes investigated. $LUST$ does only

have a converged result for the lowest advance ratio (propeller house force has not converged).

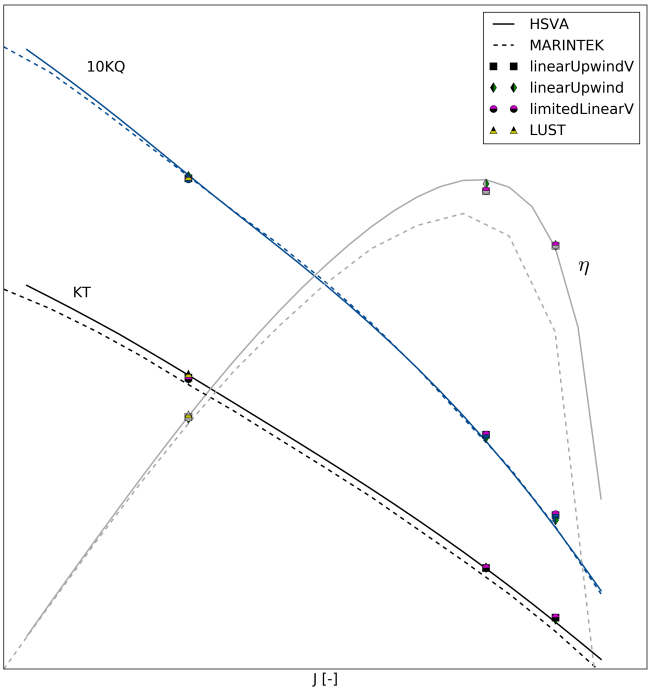
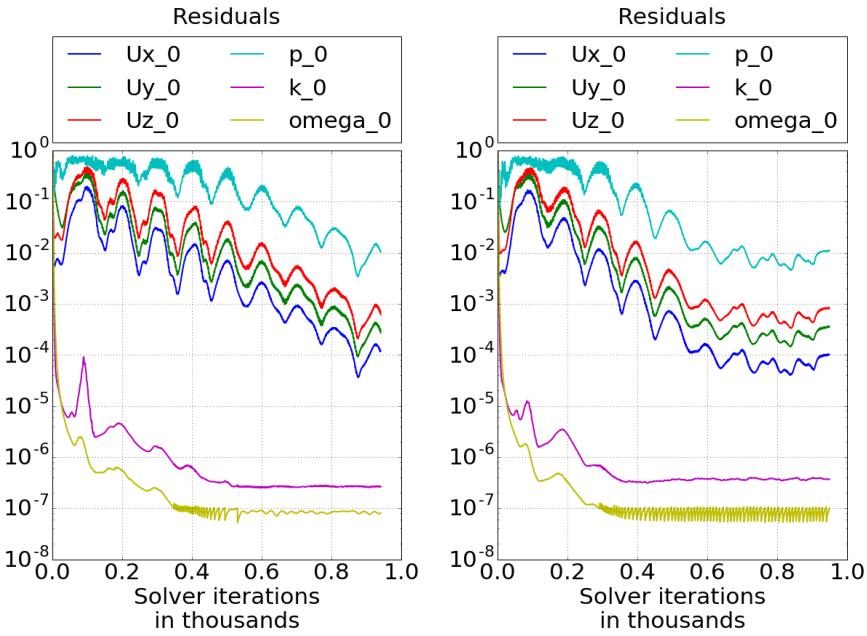


Figure 5.16: Open water diagram for divergence schemes.

5.3.3 Gradient Schemes

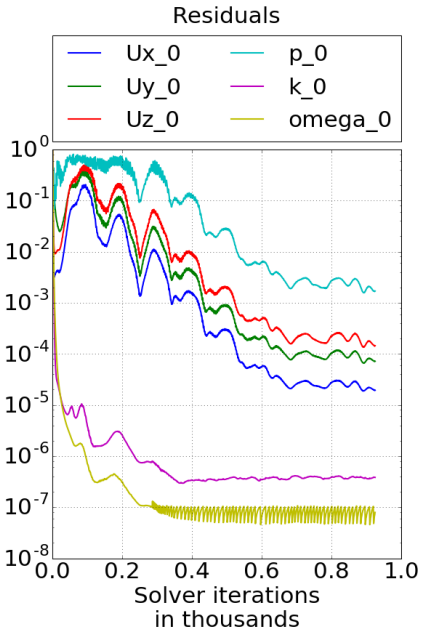
For the gradient schemes, all four limiting schemes are tested together with the Gauss linear scheme, *cellMDLimited*, *cellLimited*, *faceMDLimited* and *faceLimited*. The *faceLimited* scheme did not converge and will not be presented. The results are compared with the results obtained with no limiter. The residuals are showed in Figure 5.17 and the time series of the propeller thrust are showed in Figure 5.18. *cellLimited* have the most desirable residuals, a steady low value is reached with few iterations. *faceMDLimited* reaches the stable values faster, but the stable values are higher. *cellMDLimited*

reaches steady residuals at the same time as *cellLimited* with higher residuals. The unlimited scheme does not reach a steady level during the iterations performed. The force plots (Figure 5.18) does only show minor differences. The *faceMDLimited* shows slightly smaller oscillations.

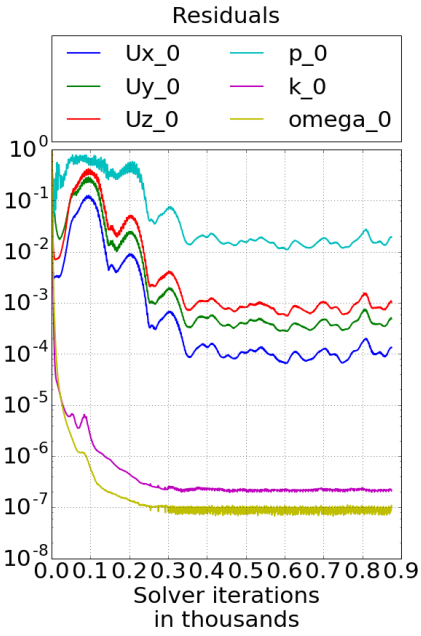


(a) No limiting

(b) Cell MD limited

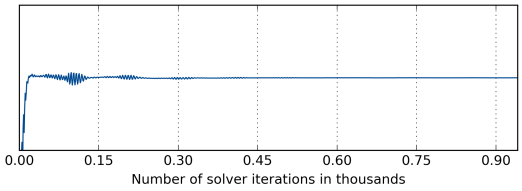


(c) Cell limited

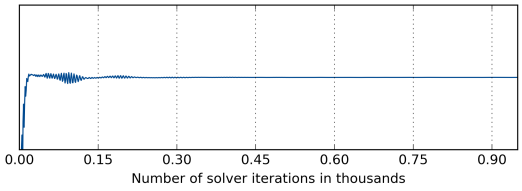


(d) Face MD limited

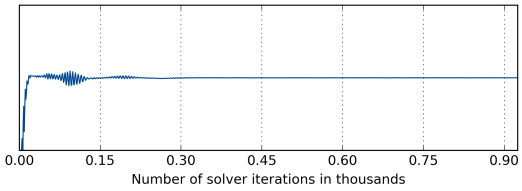
Figure 5.17: Residual plots for the gradient schemes.



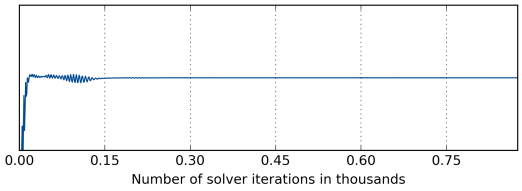
(a) Not limited



(b) Cell MD limited



(c) Cell limited



(d) Face MD limited

Figure 5.18: Force for each iteration for the gradient schemes. The y-scale is 100% of the mean of the 100 last iterations.

Table 5.6 shows the relative difference between the cell and face limited schemes compared to the unlimited scheme. The *faceMDLimited* scheme differs most from the unlimited scheme.

Table 5.6: Relative difference between the standard scheme and the face and cell limited schemes.

Scheme	KT	KQ
cellMDLimited	1.1 %	1.1 %
cellLimited	2.0 %	1.1 %
faceMDLimited	3.4 %	3.4 %

Figure 5.19 presents the open water diagram for the gradient schemes investigated, the results are compared with experimental results.

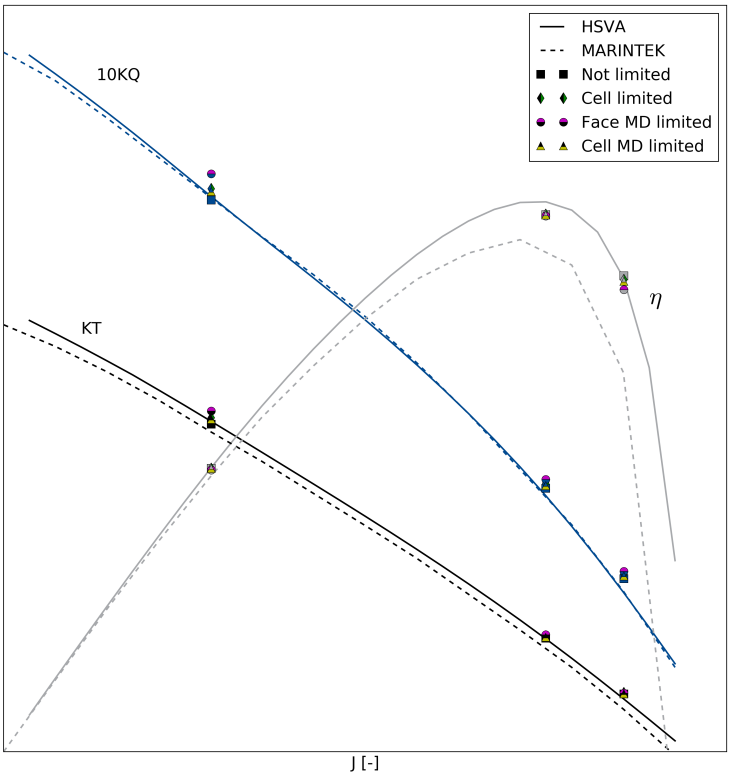
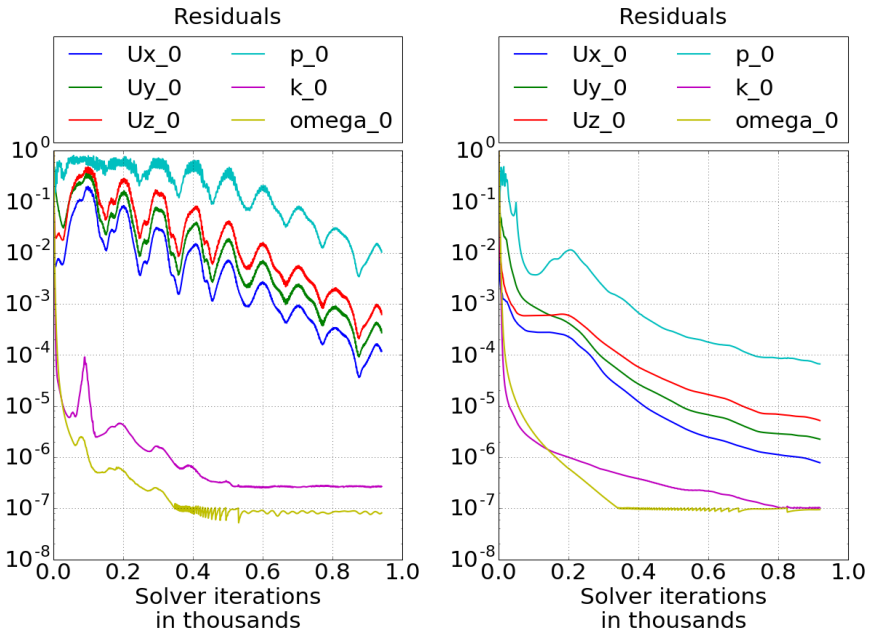


Figure 5.19: Open water diagram for the gradient schemes.

5.4 Solver

The effect of the under-relaxation factors is checked for two different schemes. From the scheme investigations, the two schemes with the most favorable features are chosen, the standard scheme and the cell limited scheme.

Figure 5.20 shows the difference due to relaxation factor on the standard scheme. The fluctuations disappear and the residuals decrease to a lower level.

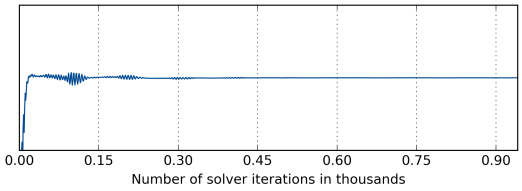


(a) Standard scheme with under-relaxation 0.5 for p and 0.8 for U, k and ω.

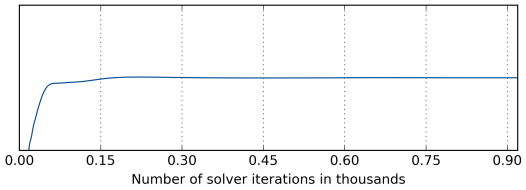
(b) Standard scheme with under-relaxation 0.3 for p and 0.5 for U, k and ω.

Figure 5.20: Residual plot showing the effect of under-relaxation on the standard scheme.

In Figure 5.21a the propeller thrust for each iteration is plotted. Figure 5.21a shows more fluctuations in the beginning than Figure 5.21b.



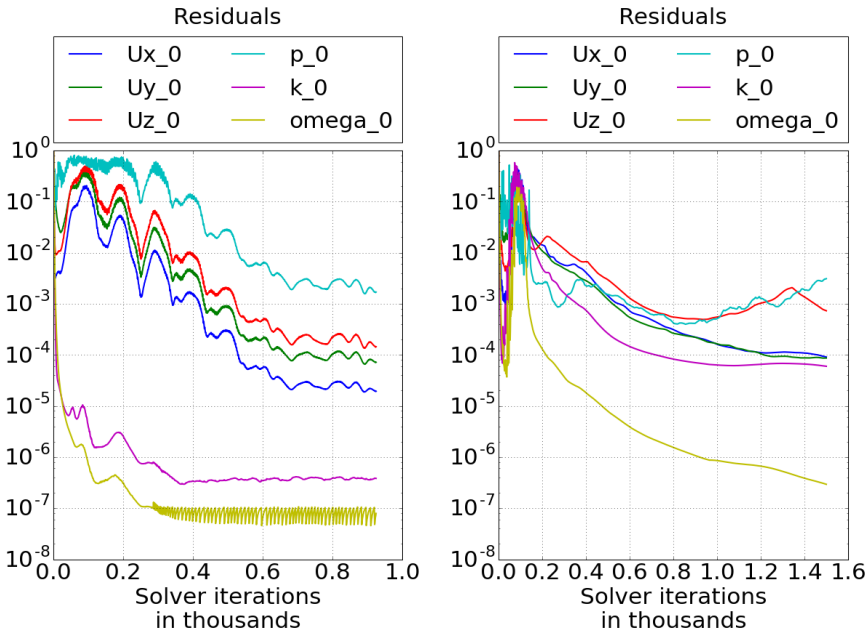
(a) Standard scheme with under-relaxation 0.5 for p and 0.8 for U, k and ω.



(b) Standard scheme with under-relaxation 0.3 for p and 0.5 for U, k and ω.

Figure 5.21: Force for each iteration for the standard scheme. The y-scale is 100% of the mean of the 100 last iterations.

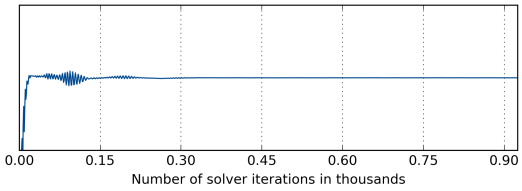
Figure 5.22 shows the difference for the under-relaxation factor for the cell limited schemes. In the first 200 iterations there are some unfavorable behavior for the lowest under relaxation (Figure 5.22b). Figure 5.23 shows the force for each iteration for the two under-relaxation levels tested for the cell limited scheme. For the lowest under-relaxations, large fluctuations are present.



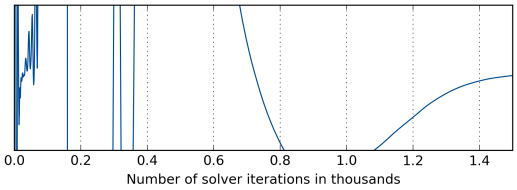
(a) Cell limited scheme with under-relaxation 0.5 for p and 0.8 for U, k and ω.

(b) Cell limited scheme with under-relaxation 0.3 for p and 0.5 for U, k and ω.

Figure 5.22: Standard scheme



(a) Cell limited scheme with under-relaxation 0.5 for p and 0.8 for U, k and ω.



(b) Cell limited scheme with under-relaxation 0.3 for p and 0.5 for U, k and ω.

Figure 5.23: Force for each iteration for the standard scheme. The y-scale is 100% of the mean of the 100 last iterations.

The difference between the under-relaxation levels for the two schemes is presented i Table 5.7. The open water diagram presents both schemes with both under-relaxation levels. The cell limited with low under-relaxation factors differ most from the other results.

Table 5.7: Relative difference between the schemes with under-relaxation factor 0.3 for p and 0.5 for U, k and ω, and 0.5 for p and 0.8 for U, k and ω.

Scheme	KT	KQ
standard	0.1 %	0.2 %
cellLimited	1.3 %	2.3 %

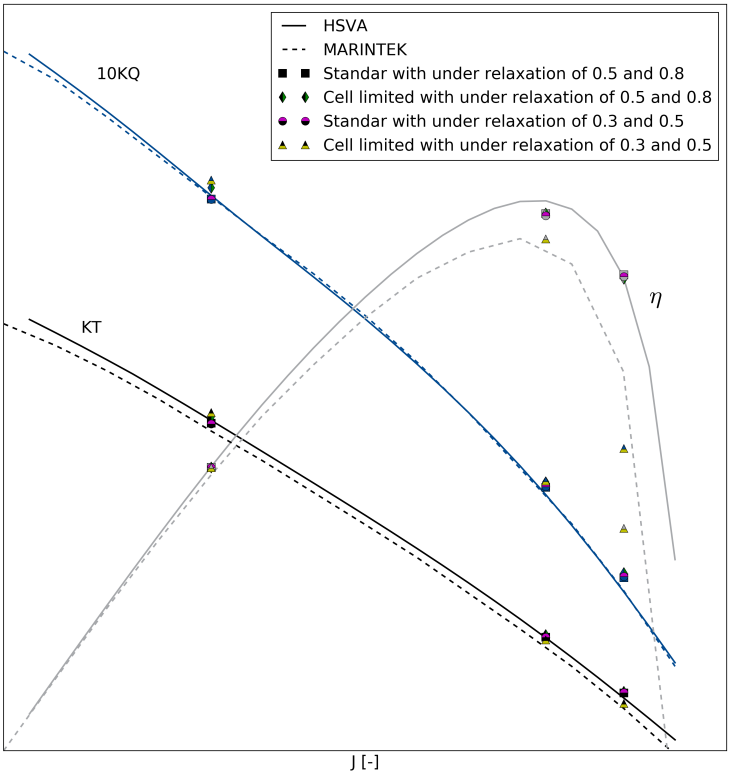


Figure 5.24: Open water diagram for different relaxation factors.

5.5 Time

Figure 5.25 shows the propeller thrust plotted against clock time. This investigation is performed to get an impression of the differences in solver time. The distance between the gray lines is 1% of the final results of the simulation. Figure 5.26 shows the same for propeller moment. The time before a steady solution is reached differs significantly as showed in Figures 5.25 and 5.26.

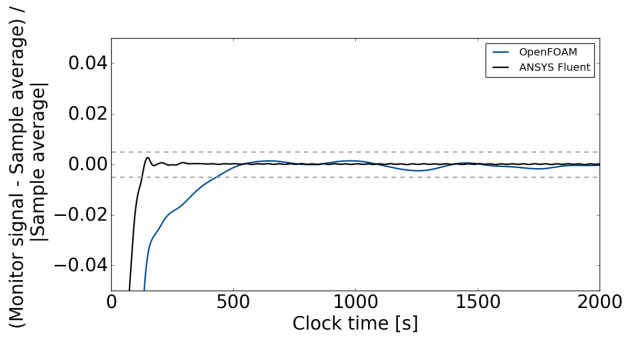


Figure 5.25: Propeller thrust plotted against clock time.

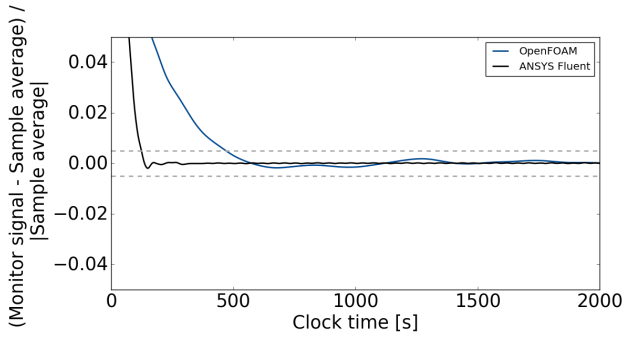


Figure 5.26: Propeller moment plotted against clock time.

5.6 Validation

Aside from the open water diagram, the results are presented for a point at the top of the propulsion efficiency curve. Results from low and high advance ratios for both propellers are presented in Appendix B.3, B.4, B.5, B.6 and B.7.

5.6.1 Azipull120

The open water diagram for the Azipull120 is presented in Figure 5.27. The results calculated are compared with data from POWS and experimental data from HSVA and MARINTEK. The results are in good agreement with POWS. The results are closer to the HSVA data than MARINTEK. The

steady state and transient simulations for the top of the efficiency curve are in good agreement. The simulation converged for all advance numbers tested for Azipull120.

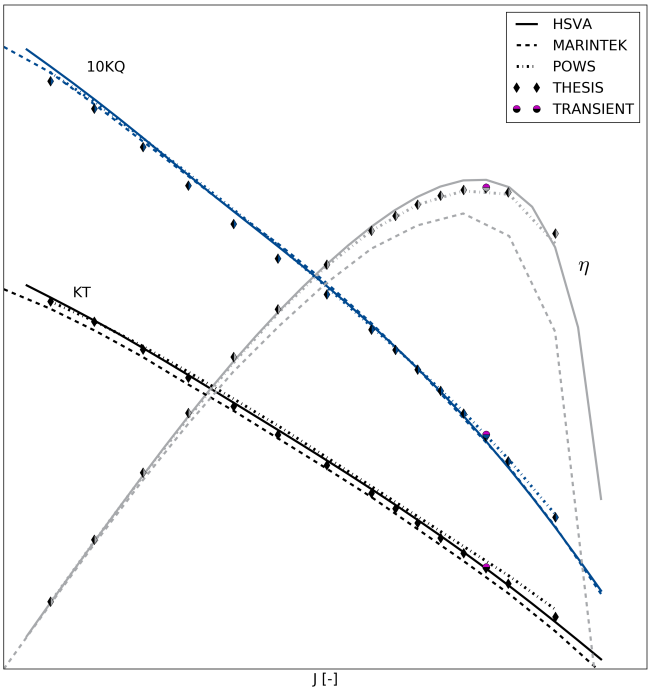
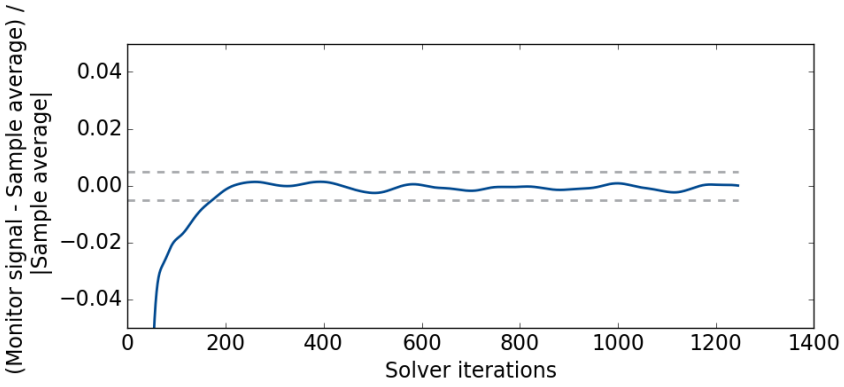
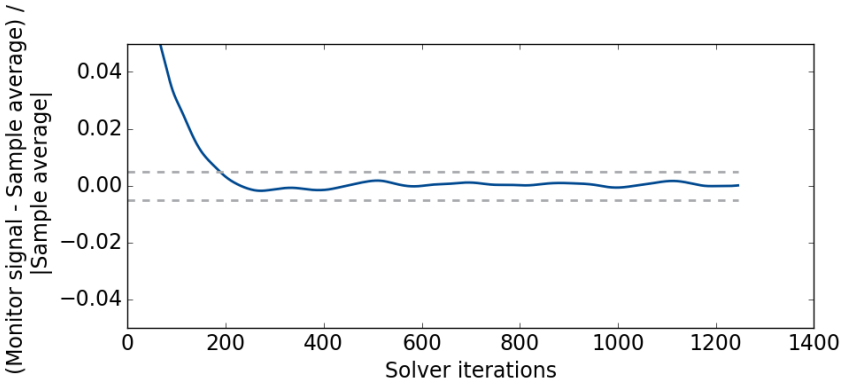


Figure 5.27: Open water diagram for the Azipull120, included are the results from HSVA, MARINTEK, TOWS and this thesis. One transient simulation is also included.

The propeller thrust and moment is presented in Figure 5.28. The distance between the gray lines is 1% of the final results of the simulation. The thrust and moment reaches a stable value with some fluctuations in a few hundred iterations. In Figure 5.29 the residuals are shown. All residuals are below $5 \cdot 10^{-5}$, this is a low value. Some small fluctuations can be observed.



(a) Propeller thrust for the Azipull120.



(b) Propeller moment for the Azipull120.

Figure 5.28: Thrust and moment for each iteration for Azipull120

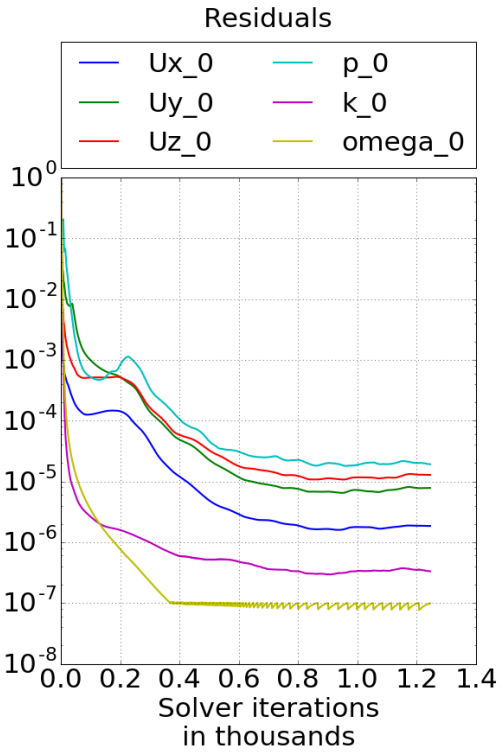
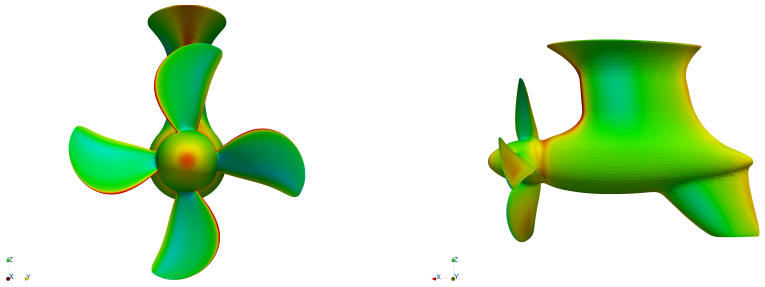


Figure 5.29: Residual for Azipull120.

Contour plots of the pressure is showed in Figure 5.30. Figure 5.30a shows high pressure at the hub, leading edge and the neck of the thruster body as expected.



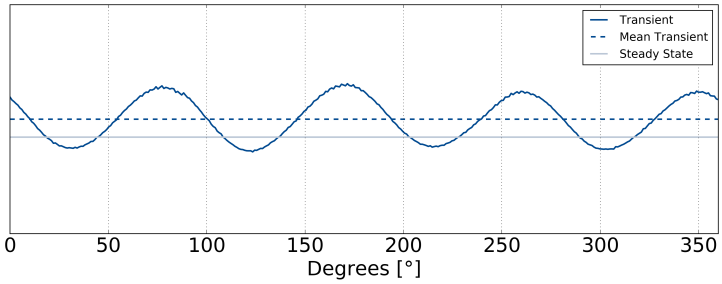
(a) Front of Azipull120.

(b) Side of Azipull120.

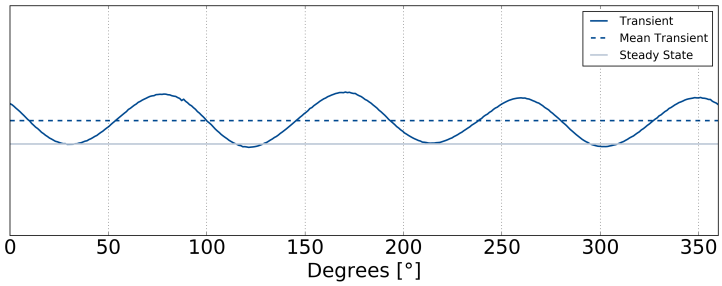
Figure 5.30: Pressure contours for the Azipull120

5.6.2 Transient Simulation

Figure 5.31, shows the differences between the steady state and transient simulations for propeller thrust and moment. The mean value of the transient simulation is higher than the steady state for both thrust and moment. The figure shows one turn of the propeller.



(a) Propeller thrust.



(b) Propeller moment.

Figure 5.31: Thrust and moment for transient and steady state simulations.

Figure 5.32 presents an animation of the propeller. This animation works in Adobe Reader. A similar animation, with more time steps is attached. The contours and isosurfaces are as expected.

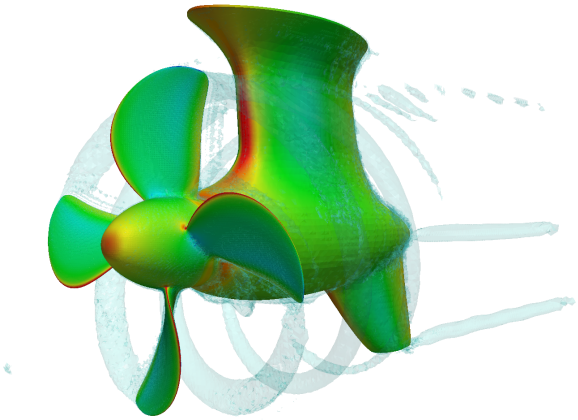


Figure 5.32: Animation of pressure contours with isosurface of the Q-criterion.

5.6.3 Azipull150

Figure 5.33 presents the open water diagram for the Azipull150. The results for the four lowest advance ratios did not converge. The converged results are in good agreement with POWS. Compared with the experimental data, the efficiency is over predicted.

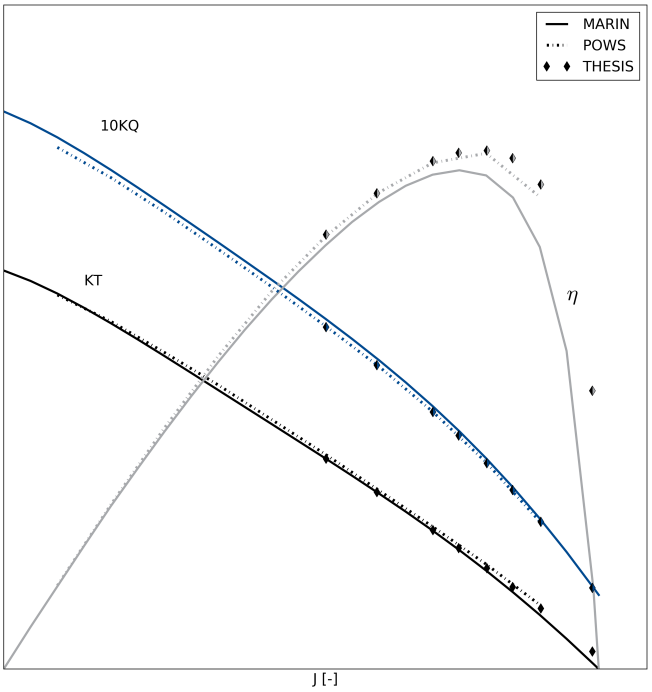
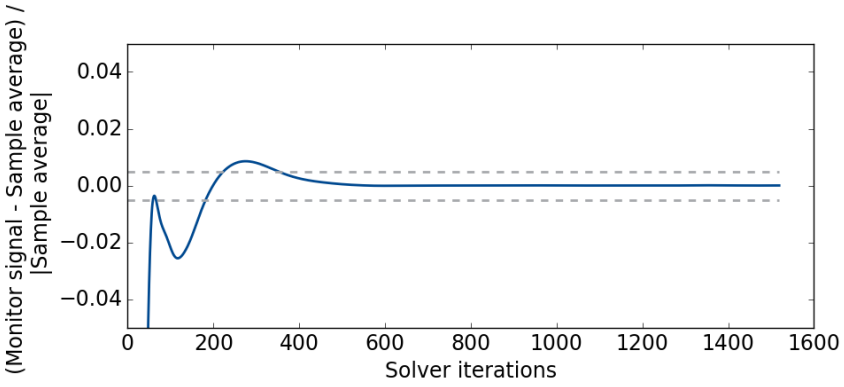
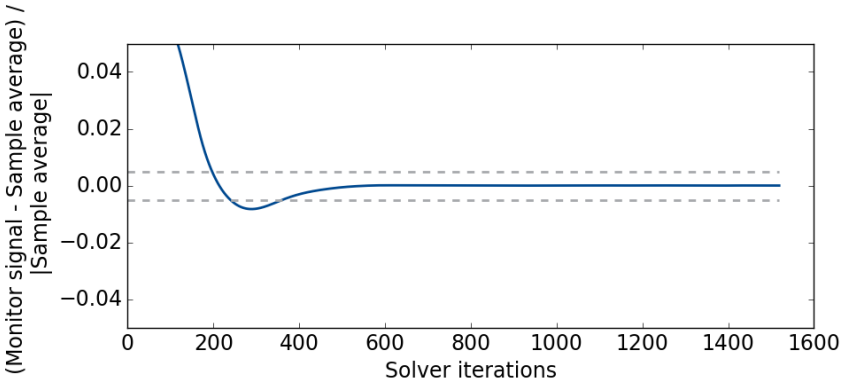


Figure 5.33: Open water diagram for the Azipull150, included are the results from MARIN, POWS and this thesis. The plots for $10KQ$ are showed in blue, the plots for KT are showed in black and the η plots are in gray.

The propeller thrust and moment is presented in Figure 5.34. The distance between the gray lines is 1% of the final results of the simulation. The thrust and moment reaches a stable value without visible fluctuations in a few hundred iterations. All residuals presented in Figure 5.35 are below $1 \cdot 10^{-5}$, which is a low value.



(a) Propeller thrust for the Azipull150



(b) Propeller moment around the Azipull150.

Figure 5.34: Thrust and moment for each iteration for Azipull150

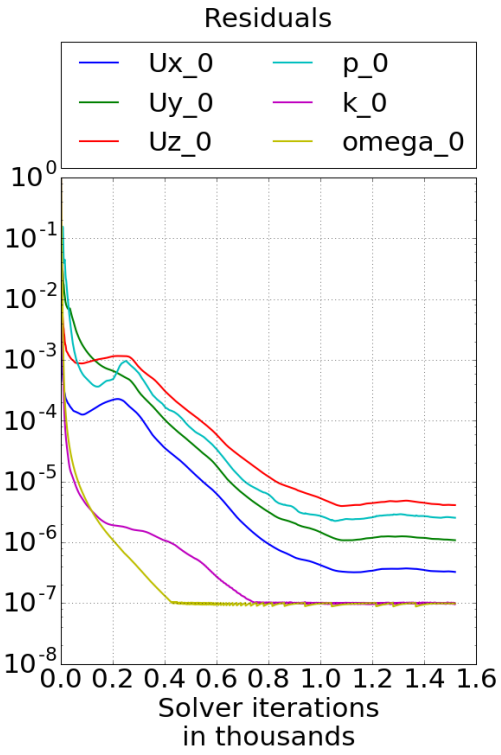
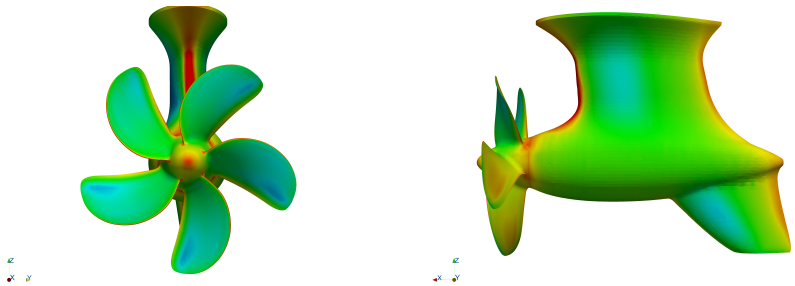


Figure 5.35: Residual for Azipull150.

Figure 5.36 shows the contour plots for Azipull150. Figure 5.30a shows high pressure at the hub, leading edge and the neck of the thruster body as expected.



(a) Front view of pressure contours on Azipull150.

(b) Side view of pressure contours on Azipull150.

Figure 5.36: Pressure contours for the Azipull150

Chapter 6

Discussion

6.1 Two-Dimensional Foil

The difference between the two solvers increase as the angle of attack increases. A reason for this may be that the separated area increases and this is the most challenging to model. Changing the first layer height on the foil, did not affect the results severely, except for C_L at 2mm. There are small differences between the solvers.

For the highest number of inflation layers, 14, problems with convergence occurred. The *checkMesh* utility, reported an *OK* mesh. *checkMesh* does not check if the area/volume of neighboring cells are severely different. Visual inspection of the mesh shows that this is the case around the inflation layer. This could be a reason for problems with convergence. The problem with convergence is solved by using the *upwind* scheme for the divergence. This is an inaccurate scheme. Here the difference between the C_L values are greater than for the first layer height, but the differences in C_D are smaller.

In the nose refinement investigation, for the lowest number of divisions the OpenFOAM case did not converge. Potential flow initialization and *upwind* start are not enough for a stable simulation. This is an expected result, as the leading edge gets very rough. In general, OpenFOAM shows more tendency towards divergence than ANSYS Fluent for the same meshes.

6.2 Mesh

The number of cells is large compared with POWS. A reason for this is the leading edge of the propeller, here the numbers of divisions are the greatest. At the leading edge it is most difficult to resolve the geometry and flow gradients. Meshes with around 30 % less cells have proven to converge on each propeller, but not the same mesh on both validation propellers. The coarser meshes does not capture the geometry accurately, the leading edge gets saw-shaped and cells collapse. This can lead to the whole boundary layer collapsing. This problem is reduced by reducing the cell size in this region. Manually created meshes with similar total number of cells, usually capture the edges more smoothly. The current meshes in POWS have rough leading edges, ANSYS Fluent solves this without problems. From the two-dimensional test case, it can be observed that a coarse mesh at the leading edge easier leads to divergence in OpenFOAM than ANSYS Fluent.

The geometries directly delivered from the propeller designers have one face called *refine*, this is located on the propeller edge. For the Azipull120 the connection between the blade and hub is also included in the refine patch. The mesh used for the scheme and solver investigation have this refinement. This is a large area which are resolved with a small cell size, this will affect the number of cells greatly. On the final mesh, this part of the geometry is included in the same patch as the propeller blades and hub, instead of the refine patch. This is done to save cells and have similar meshes for the two thrusters. To better capture the leading edge, the refine face is split on the tip. This makes the edge smoother. In addition, the leading and trailing edge is split to make it possible to have a finer mesh level at the leading edge than at the trailing edge. This saves a considerable number of cells.

The difference between the mesh size for the two propellers is significant, this is due to the larger diameter and one extra blade for the Azipull150. To capture all important curvature on the thruster body on the Azipull120, edge refinement is used. The same edge refinement did also capture all important areas on the Azipull150. On Azipull150 some areas that did not need refinement are also refined. The reason for this unnecessary refinement is differences in curvature for the two thruster bodies. This increased the number of cells unnecessary. Making a more advanced meshing script could avoid this. These unnecessary cells do not heavily affect the total number of

cells.

The mesh density at the propeller blade does seem excessive, but problems with divergence occurred when decreased. The reduction of cells is in the magnitude of 100 000 cells, around 1 % of the total mesh size, for Azipull120. The domain size is not investigated, the cells in the outer parts of the domain is large and the total number of cells will not be influenced severely. The domain size chosen is based on common practice. The volume refinement around the thruster is made as small as possible without creating divergence around the top of the propulsion efficiency curve for both the validation thrusters. The domain size is dependent on the diameter of the propeller. The propeller diameter compared with the size of the thruster body vary, this makes the domain also vary compared to the thruster body.

The maximum non-orthogonality is high for both meshes, 85.9 for the Azipull120 and 84.9 for the Azipull150. Cells with non-orthogonality over 70 degrees are considered high. To account for the high non-orthogonality the surface normal gradient and Laplacian schemes are corrected and the pressure correction equation is solved three times. The mesh non-orthogonality can be an important parameter when considering stability. However, the maximum value is only one cell, this makes it dangerous to use alone as a decision criterion. The average non-orthogonality and maximum aspect ratio are acceptable for both meshes. The non-orthogonal cells are inspected visually in ParaView to see if there are groups of them or if they are in critical regions. The meshes are also inspected to see if the geometry is badly captured or critical regions have too big cells. The capturing of the geometry is also affected by the CAD file imported, since the quality of this may differ.

A conformal mesh is used to make the interpolation between the two regions easier. In addition, small cells at the edge of the interface is required for a stable solution. The prismatic boundary layer around the thruster body heavily affects the number of cells. The same initial height, number of inflation layers and growth rate as in POWS is used. This is not investigated any further. It must be noted that the grid is made by an inexperienced user and can probably be improved.

6.3 Schemes

6.3.1 Surface Normal Gradient Schemes

To maintain accuracy for non-orthogonal meshes an explicit non-orthogonal correction is used. The size of the non-orthogonal correction is dependent on the non-orthogonality of the mesh. The cases tested in this thesis have full non-orthogonality correction and corrections with limiters. Limiters can be useful if the non-orthogonality is high to limit the size of the correction. Limiters will bound the solution. The results show no improvement in stability and are in less agreement with the experimental data for KQ. For KT and η the experimental data differ and are not compared with. It does not seem to be a problem with unboundedness and the non-orthogonality limiters will not be used in the final scheme.

6.3.2 Divergence Schemes

As expected when removing the more strict limiting applied by the V-scheme the calculation became less stable, this can be seen by comparing Figures 5.14a and 5.14b. The accuracy of the calculation is more difficult to determine. The two schemes are plotted in Figure 5.16, the 5.14b approach is closer to the experimental KQ values which are in most agreement with each other. The *limitedLinearV* shows almost no fluctuations, but the final residual for pressure is high. The KT and KQ results are similar to the other two schemes tested. The $LUST$ scheme proved to be too unstable for this mesh, this is an expected result as the scheme is less stable than the *linearUpwind* and the V-scheme option is not enabled. The divergence scheme with the most favorable traits is the *linearUpwindV* scheme, this will be used in the final scheme.

6.3.3 Gradient Schemes

Introducing limiters on the gradient scheme is expected to increase the stability. Only *cellLimited* seemed to have this effect from looking at the residual plots. The final residuals for both MD schemes are high. From the force plots the *faceMDLimited* shows the least oscillations. The difference from the unlimited schemes are largest for the *faceMDLimited* and smallest for the *cellMDLimited* as expected. The only limiter diverging is the *faceLim-*

ited. This was not expected as it is expected to be the most stable approach. Both the unlimited and the *cellLimited* scheme is investigated further, as both shows promising results.

6.4 Solver

Under-relaxation is used to create a more stable solver, usually more iterations are required. To reduce the fluctuating behavior and increase the stability, lower under-relaxation is tested. For the standard scheme the fluctuations are reduced as expected and the residuals decrease smoothly to a low value (Figure 5.20). In the force plot in Figure 5.21a, the simulation for the low under-relaxation increase slower and smoother to a stable value. The simulation with high under-relaxation reaches a value close to the final value faster, but large fluctuations are present for the first few hundred iterations.

The effect of under-relaxation is also tested for a case with cell limited gradient scheme. Here the effect did not make the simulation more stable. There are some unfavorable behavior showed in the beginning of the residual plot in Figure 5.22. Figure 5.23 shows that the thrust does not converge smoothly. The effect from the change in under-relaxation is largest for the final thrust and moment on the cell limited gradient scheme.

In the open water diagram showed in Figure 5.24 the cell limited scheme with low under-relaxation factors differs mostly from the other numerical results and the experimental results. There is no visible difference between the standard scheme with different relaxation levels. These are the results in most agreement with the experimental results. The standard scheme with low under-relaxation has the most favorable residual and force plots.

6.5 Validation

For the Azipull120 there are some visible fluctuations in Figure 5.28. The amplitudes of the fluctuations are smaller than 0.5%. For the low advance ratio, the fluctuations almost disappear (Figure B.4) and for the high advance ratio (Figure B.7) the fluctuation amplitudes are around 0.5%. If looking at the lowest advance ratio simulated for the Azipull120 the fluctuations are

large in the beginning and then damped to small fluctuations. For the Azipull150 there are no visible fluctuations on any of the results. The reason for this difference may be the number of propeller blades, differences in mesh density or the underlying geometry.

From the residual plots for the Azipull120 it can be observed that the residual are highest for the lowest advance ratio (Figure B.2), but also high for the highest (Figure B.8). Most fluctuations occur also here. For the Azipull150, the high advance ratio has the highest residual. The four simulations with lowest advance ratios diverges. This is not an expected result as the other advance ratios showed a stable solving process, but the lowest advance ratios are usually the most difficult. The most important part of the open water diagram is successfully simulated, but the system should be stable enough to work for all advance ratios. I have not been able to generate a grid that does this and still works well for the Azipull120, without the use of a significantly more expensive mesh. The time and computer resources did not allow new rounds with meshing and running of simulations. Some adjustments to the mesh, some more cells or better placed refinement areas could possibly fix the problem. The result, however, reflects on an instability problem present for the whole master thesis. Small changes in mesh easily leads to diverging results. This is not convenient for an automatic system used for different thruster designs. All pressure contour plots look as expected and the stagnation pressure is highest for the high advance ratio as expected.

The results for steady state simulations are within a tolerable accuracy. The results are close to those obtained by POWS for both propellers. For the Azipull120, it is difficult to compare with the model tests as they differ. The thrust coefficient KT is between the two model tests and slightly under the POWS results. The torque coefficient KQ seems to be slightly under predicted compared with the other results. The propeller efficiency η is in good agreement with POWS, slightly under the HSVA results and above the MARINTEK results. The propeller efficiency η is a derived result, that is a combination of the KT and KQ values (found in Equation 2.44). For the Azipull150 the propulsion efficiency is predicted higher compared with the experimental results from MARIN. For the $10KQ$ and KT values, the differences are small.

The result for the transient simulation is close to the steady state simu-

lation (Figure 5.27). The mean of the transient simulation is slightly higher than for steady state. The pressure contours and Q-criterion in Figure 5.32, looks as expected.

6.6 Time

The solver time test is done to get an impression of the difference in solver time for ANSYS Fluent and OpenFOAM. The solver time is an important decision criteria, when deciding if the solver is suitable for POWS. Only one case is tested and many factors differ. Different solving techniques are used, OpenFOAM use the SIMPLE algorithm and ANSYS Fluent use a coupled solver. Different schemes are also used. Also, the initialization is different, ANSYS Fluent uses hybrid initialization and OpenFOAM uses potential flow calculation. The differences are due to different options available for each software. The aim for this investigation is to compare POWS, as it is, with OpenFOAM in regards of solver time.

Even with all the differing parameters some impression of differences between the solvers are possible to get. The final forces from the two solvers, are in good agreement with each other. ANSYS Fluent reaches a steady state significantly faster than OpenFOAM. OpenFOAM is within 0.5 % of the final values for propeller thrust and moment after 474 seconds, while ANSYS Fluent reaches this value after 127 seconds. ANSYS Fluent is over 3.7 times faster than OpenFOAM to reach this criterion. This is not a suitable convergence criterion, it is only included to get an impression of the solver times. For a solution to be converged, a number of iterations should be within a given tolerance. One iteration for a coupled and an uncoupled solver is not the same and the numbers of iterations inside the tolerance should not necessarily be the same. Significantly more simulations must be performed to be able to say something precise about the solver time. The fluctuations using ANSYS Fluent also have smaller amplitudes and higher frequency.

6.7 General

It is time consuming to set up OpenFOAM as a solver as all information must be given by the user. Since there is no user interface this process demands

understanding of the solver and processes within. It is most challenging the first time, but continuous work must be done to ensure optimum solver and scheme settings.

To generate each simulation, an automated system is required. The standard folders used in OpenFOAM simulations, convergence scripts and plotting scripts must be included in the case folder. The values for velocity, turbulence, revolutions per second and propeller diameter amongst other must be changed for each simulation. In addition, the mesh information needs to be transferred to the folder where the simulation is performed. For this thesis, a simple bash script is made to mesh, set up and run the cases. OpenFOAM is suitable for automated systems, as everything can be managed from the terminal.

Chapter 7

Conclusion

A conformal mesh that is suitable for OpenFOAM is created. The mesh is computationally expensive. A reason for this is the leading edge of the propeller, here it is most difficult to resolve the geometry and flow gradients. The number of divisions is the greatest at the leading edge. Using one level coarser mesh on the leading edge could reduce the mesh size with around 30 %. The coarser meshes do not capture the geometry accurately, the leading edge gets saw shaped as the meshing code have problems with detecting the leading edge of the propeller. This problem is reduced by reducing the cell size in this region and splitting the edge. The current meshes in POWS have rough leading edges, ANSYS Fluent solves this without problems.

The results from the scheme investigations are expected results and in agreement with common practice. For the surface normal gradient and the Laplacian scheme, the corrected option without any limiting proved to be most stable on the simulations run. For the divergence, *linearUpwindV* is used for the velocity and *upwind* for the turbulence parameters. It is more important for the turbulence parameters to be bounded than accurate. The stability is the most heavily weighted criteria as the systems show problems with convergence. The accuracy is also considered, but not as heavily. Low under-relaxation generally increases stability and numbers of iteration before the convergence criteria are met. In this case the numbers of iterations did not increase severely and the solution became more stable. The under-relaxation used for the validation is 0.3 for the pressure field and 0.5 for the velocity, k and ω equations.

The results for steady state simulations are within a tolerable accuracy. The results are close to those obtained by POWS for both propellers. It is difficult to compare the Azipull120 results with the model tests as they differ. The thrust coefficient KT is between the two model tests and slightly under the POWS results. The torque coefficient KQ seems to be slightly under predicted compared with the other results. The propeller efficiency η is in good agreement with POWS, slightly under the HSVA data and above the MARINTEK data. For the Azipull150 the propeller efficiency is predicted higher than the experimental data from MARIN. For the $10KQ$ and KT values the differences are small. The transient simulation is in good agreement with the steady state simulation.

For the Azipull150 the simulation in the most important region converged without problems, but the four simulations with lowest advance ratio did not converge. These usually are the most challenging. The divergence is still unexpected as all other simulations converged smoothly. The time and computer resources did not allow new rounds with meshing and running of simulations. Some adjustments to the mesh, some more cells or better placed refinement areas could possibly fix the problem. The result however reflects on an instability problem present for the whole master thesis. Small changes in mesh easily leads to diverging results. This is not convenient for an automatic system used for several different thruster designs.

It is time consuming to set up OpenFOAM as a solver as all information must be given by the user. Since there is no graphical user interface, this requires understanding of the solution process. OpenFOAM is suitable for automated systems, everything can be performed from the terminal.

The quality of the grid generated by BOXERMesh is not completely satisfactory with the setup used in this thesis. It must be noted that the grid is made by an inexperienced user and can probably be improved. The main problem is the leading edge on the propeller. This edge easily get rough due to cells collapsing. This can lead to collapsing of the prismatic boundary layer. Changing the mesher could possibly be a solution to the instability problem, this is not investigated. To replace BOXERMesh, the mesher preferably should be open source and possible to automate.

OpenFOAM can be used as solver in POWS, but the high mesh require-

ments from OpenFOAM and the low-quality grids generated by BOXERMesh combined is a challenge. It is possible to overcome this, but a higher number of cells than for the current POWS system seems unavoidable. It is possible to run simulations with approximately the same number of cells. For a general automated robust system for several propeller designs, this is difficult. The robustness is important to avoid excessive manual work and delays in results. It also important to get reliable results for all simulations.

Chapter 8

Further Work

The solver time is a challenge to investigate. To decide if the new solver is feasible, more information about the total run time of a simulation is required. Several cases on both systems should be performed on the same number of processors to get an better impression.

If OpenFOAM is chosen to replace ANSYS Fluent as solver in POWS, more work is required. A new meshing script must be generated. If possible the mesh on the edges of the propellers, should be improved. In addition, new parts such as ducts must be included.

Implementing the new mesh script, solver and post processing in POWS must be performed. This will lead to large changes in the system. More simulations of different cases needs to be performed to ensure stability and correct results for a wider range of cases.

Transient simulations needs a more thorough treatment. More simulations is required and more investigation of the solution parameters. Azimuthing can also be considered.

Appendix A

Acronyms

AMI Arbitrary mesh interface

BC Boundary condition

BD Blended differencing

CD Central differencing

CFD Computational fluid dynamics

IC Initial condition

MRF Multiple Refrence frame

NVA Normalized variation approach

OpenFOAM Open Field Operation and manipulation

PDE Partial differential equation

PISO Pressure-implicit split-operator

POWS Propeller open water simulations

RANS Reynold-averaged Navier-Stokes

SIMPLE Semi-implicit method for pressure-linked equation

TFI Transfinite Interpolation

TVD Total variation diminishing

UD Upwind differencing

Appendix B

Additional Information

B.1 Animation of Transient Simulation

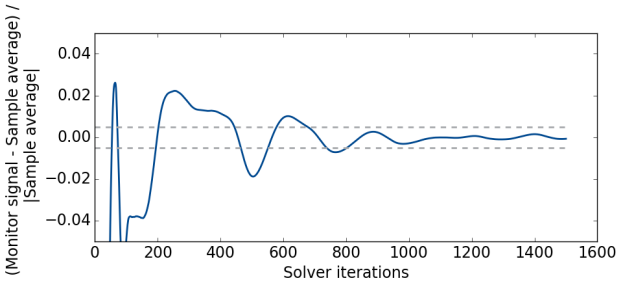
An animation of the pressure contour plot with a isosurface of the Q-criterion, is attached. The file is called *TransientSimulationAnimation*.

B.2 Standard Scheme

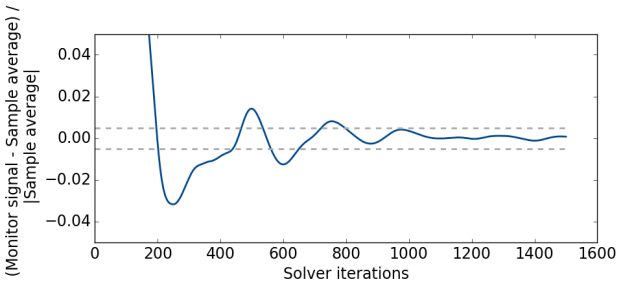
```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                      |                                            |
|
| \\      /  F ield              | OpenFOAM: The Open Source CFD Toolbox
|
|  \\    /   O peration          | Version:  2.2.1
|
|   \\  /    A nd                | Web:      www.OpenFOAM.org
|
|    \\/     M anipulation       |
|
\*---------------------------------------------------------------------------*/
FoamFile
{
version     2.0;
format      ascii;
class       dictionary;
location    "system";
object      fvSchemes;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

ddtSchemes
{
default          steadyState;
}

gradSchemes
{
default          Gauss linear;
}

divSchemes
{
default          none;
```

```
div(phi,U)      bounded Gauss linearUpwindV grad(U);
div(phi,k)      bounded Gauss upwind;
div(phi,omega)  bounded Gauss upwind;
div((nuEff*dev2(T(grad(U))))) Gauss linear;
}

laplacianSchemes
{
default         Gauss linear corrected;
}

interpolationSchemes
{
default         linear;
}

snGradSchemes
{
default         corrected;
}

wallDist
{
method meshWave;
}

// ************************************************************************* //
```

B.3 Azipull120 Lowest Advance Ratio



(a) Propeller thrust for the Azipull120



(b) Propeller moment for the Azipull120.

Figure B.1: Thrust and moment for each iteration for Azipull120

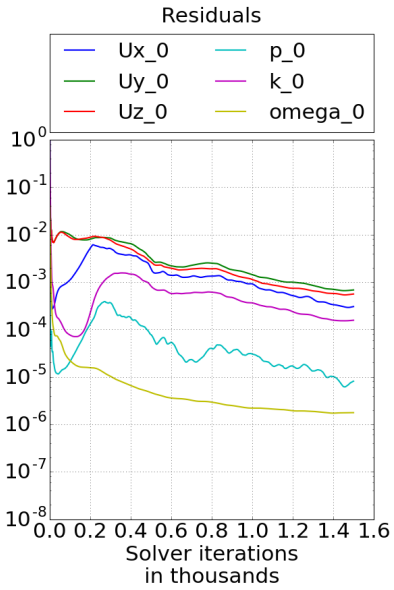
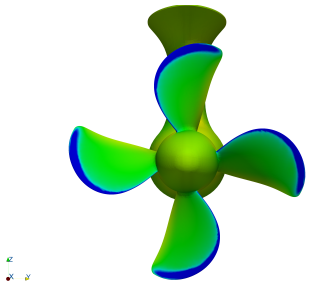
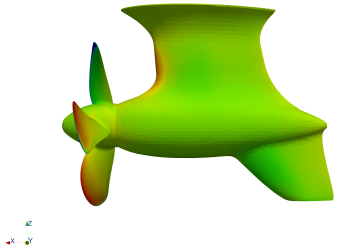


Figure B.2: Residual for Azipull120.



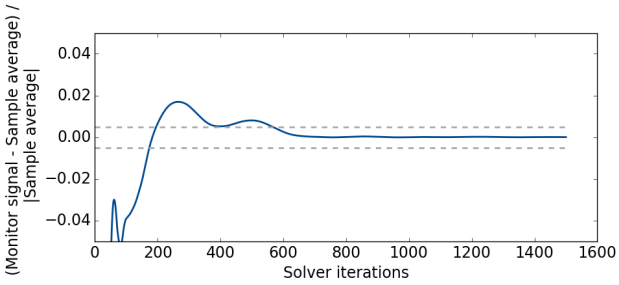
(a) Front of Azipull120.



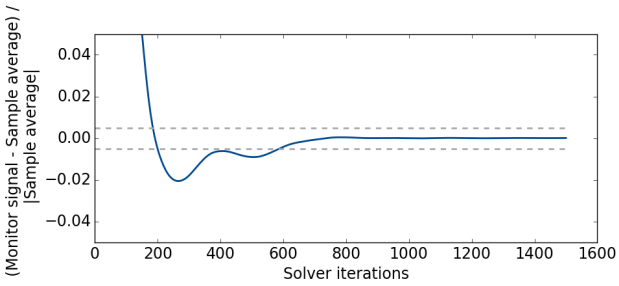
(b) Side of Azipull120.

Figure B.3: Pressure contours for the Azipull120

B.4 Azipull120 Low Advance Ratio



(a) Propeller thrust for the Azipull120



(b) Propeller moment for the Azipull120.

Figure B.4: Thrust and moment for each iteration for Azipull120

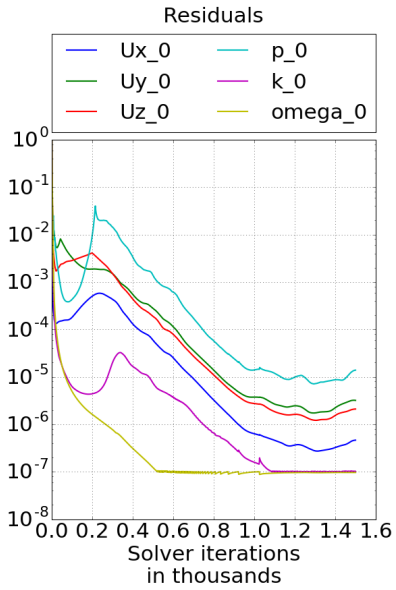
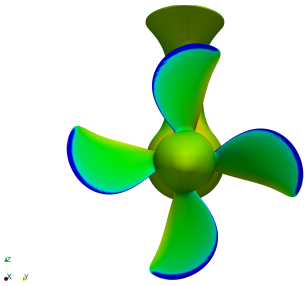
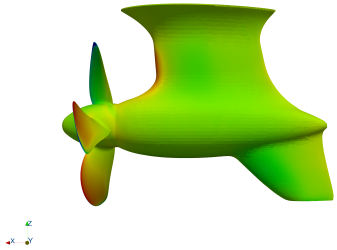


Figure B.5: Residual for Azipull120.



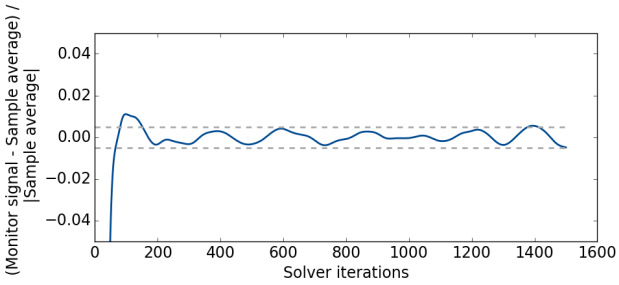
(a) Front of Azipull120.



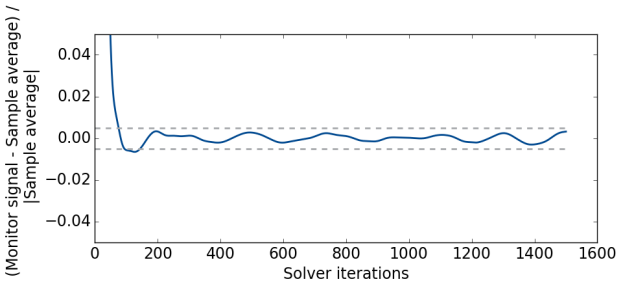
(b) Side of Azipull120.

Figure B.6: Pressure contours for the Azipull120

B.5 Azipull120 High advance ratio



(a) Propeller thrust for the Azipull120



(b) Propeller moment for the Azipull120.

Figure B.7: Thrust and moment for each iteration for Azipull120

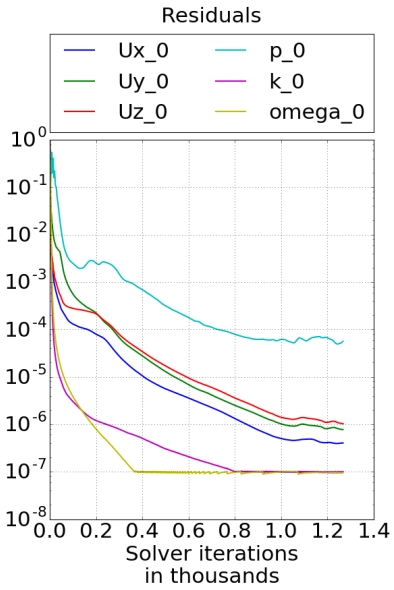
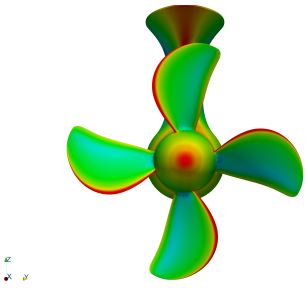
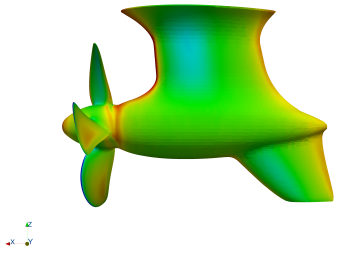


Figure B.8: Residual for Azipull120.



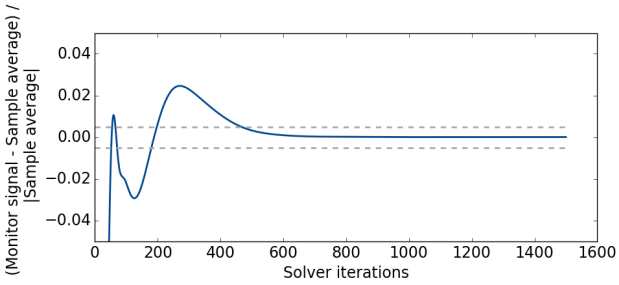
(a) Front of Azipull120.



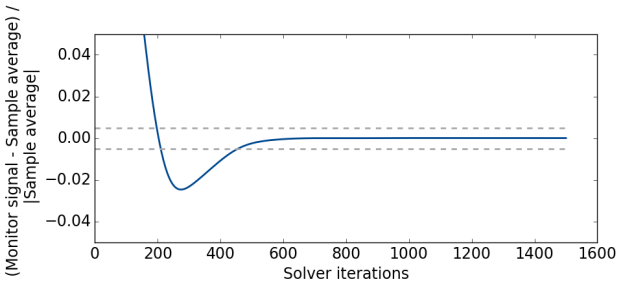
(b) Side of Azipull120.

Figure B.9: Pressure contours for the Azipull120

B.6 Azipull150 Low Advance Ratio



(a) Propeller thrust for the Azipull150



(b) Propeller moment around the Azipull150.

Figure B.10: Thrust and moment for each iteration for Azipull150

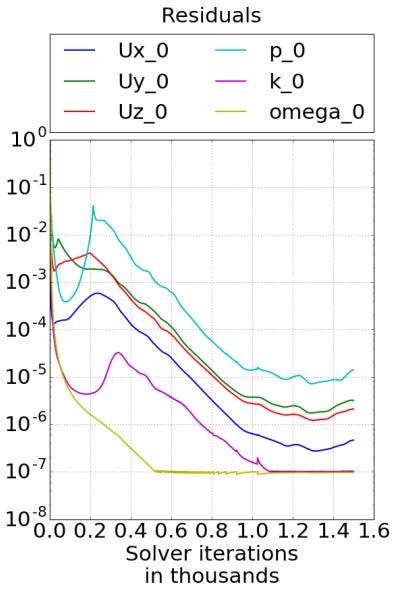
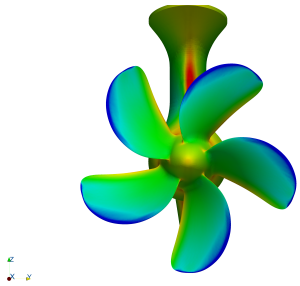
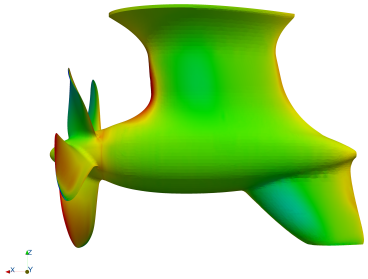


Figure B.11: Residual for Azipull150.



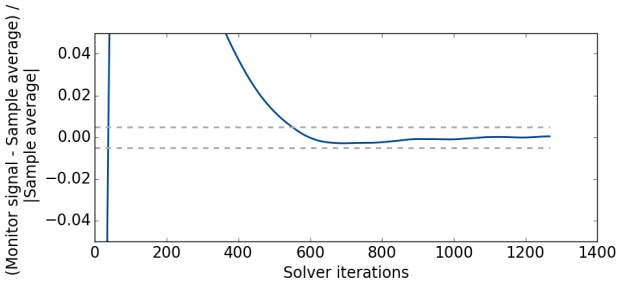
(a) Front of Azipull150.



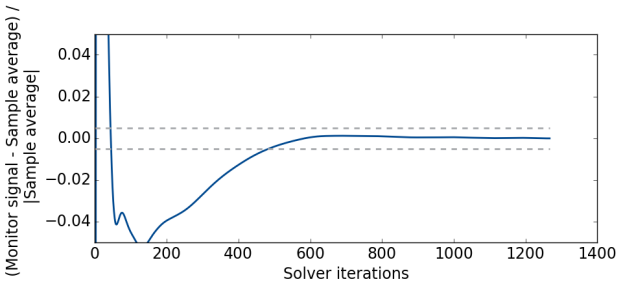
(b) Side of Azipull150.

Figure B.12: Pressure contours for the Azipull150

B.7 Azipull150 High Advance Ratio



(a) Propeller thrust for the Azipull150



(b) Propeller moment around the Azipull150.

Figure B.13: Thrust and moment for each iteration for Azipull150

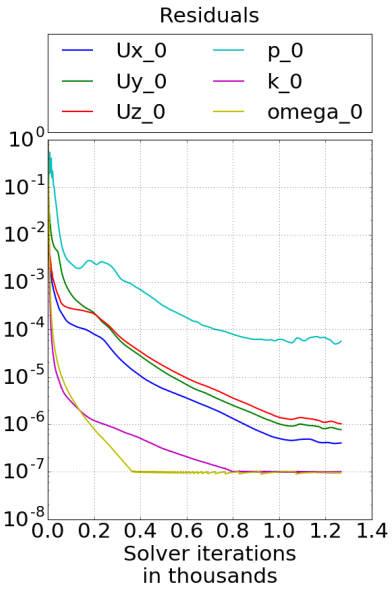
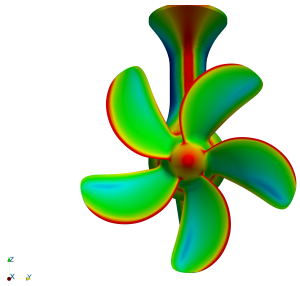
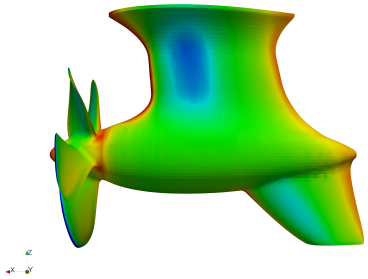


Figure B.14: Residual for Azipull150.



(a) Front of Azipull150.



(b) Side of Azipull150.

Figure B.15: Pressure contours for the Azipull150

B.8 Two-Dimensional Meshes

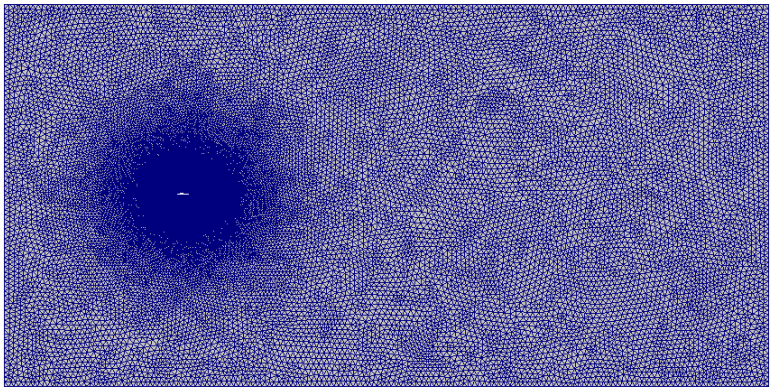
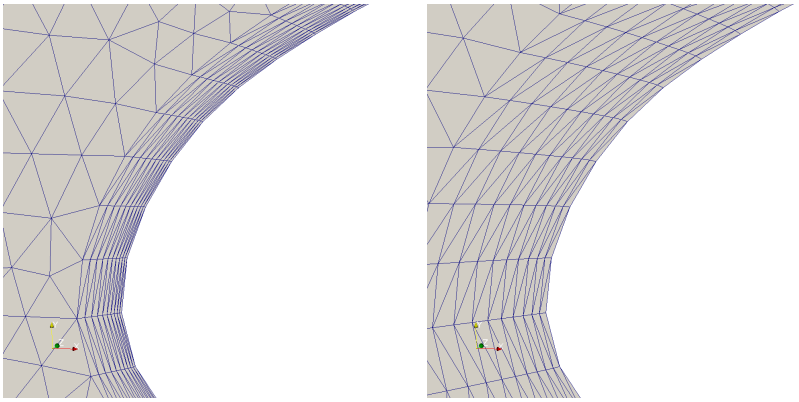
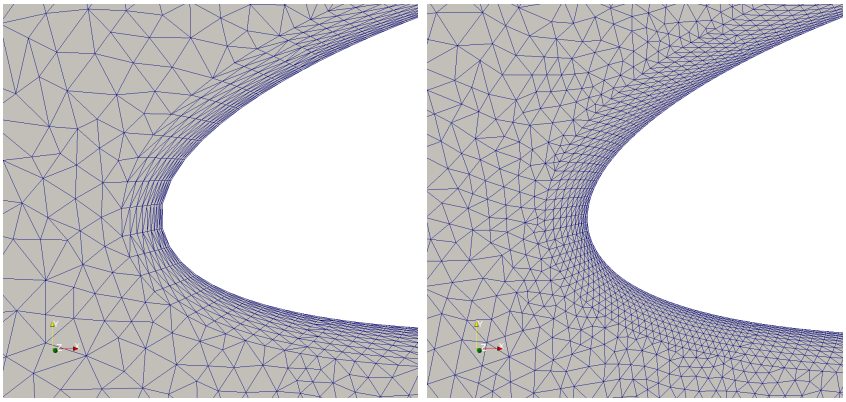


Figure B.16: Initial mesh.



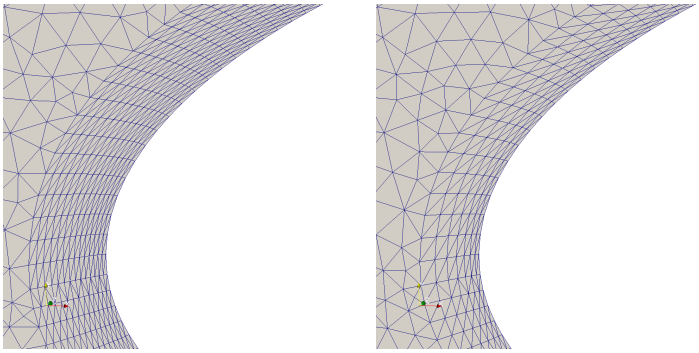
(a) 2mm. (b) 22mm.

Figure B.17: First layer height.



(a) 5 cells. (b) 50 cells.

Figure B.18: Nose refinement.



(a) 7. (b) 14.

Figure B.19: Number of inflation layers.

Bibliography

Abbott, I. and Von Doenhoff, A. (1959). *Theory of Wing Sections, Including a Summary of Airfoil Data*. Dover Books on Aeronautical Engineering Series. Dover Publications.

Alterskjær, S. A. (2010). Report no. 530635.00.0.

ANSYS (2013). *ANSYS Fluent Theory Guide*.

Bertram, V. (2012). *Practical Ship Hydrodynamics (2nd Edition)*. Elsevier.

Blazek, J. (2001). *Computational Fluid Dynamics: Principles and Applications*. Elsevier Science.

Dang, D. I. J. and Radstaat, G. (2013). Report no. 25911-2-dt-dwb.

Greenshields, C. (2015). Openfoam user guide: 4.4 numerical schemes. `https://cfd.direct/openfoam/user-guide/fvschemes/`. [Online; accessed 29-April-2017].

Hovden, S. M. (2016). Cfd study of flow around a 2-d foil at high attack angles.

Jasak, H. (1996). Error analysis and estimation for the finite volume method with applications to fluid flows.

Klug, H. (2007). Hsva-model no. 4480-2111.

Muntean, T. V. (2012). *Propeller efficiency at full scale : measurement system and mathematical model design*. Eindhoven: Technische Universiteit Eindhoven.

Sideroff, C. (2010). Openfoam guide/limiters. https://openfoamwiki.net/index.php/OpenFOAM_guide/Limiters. [Online; accessed 01-June-2017].

Steen, S. (2011). *TMR4247 Marin teknikk 3 - Hydrodynamikk*. Akademika forlag.

Vasileska, D. (xxxx). Lecture notes in EEE533 Semiconductor device and process simulation, Arizona State University.