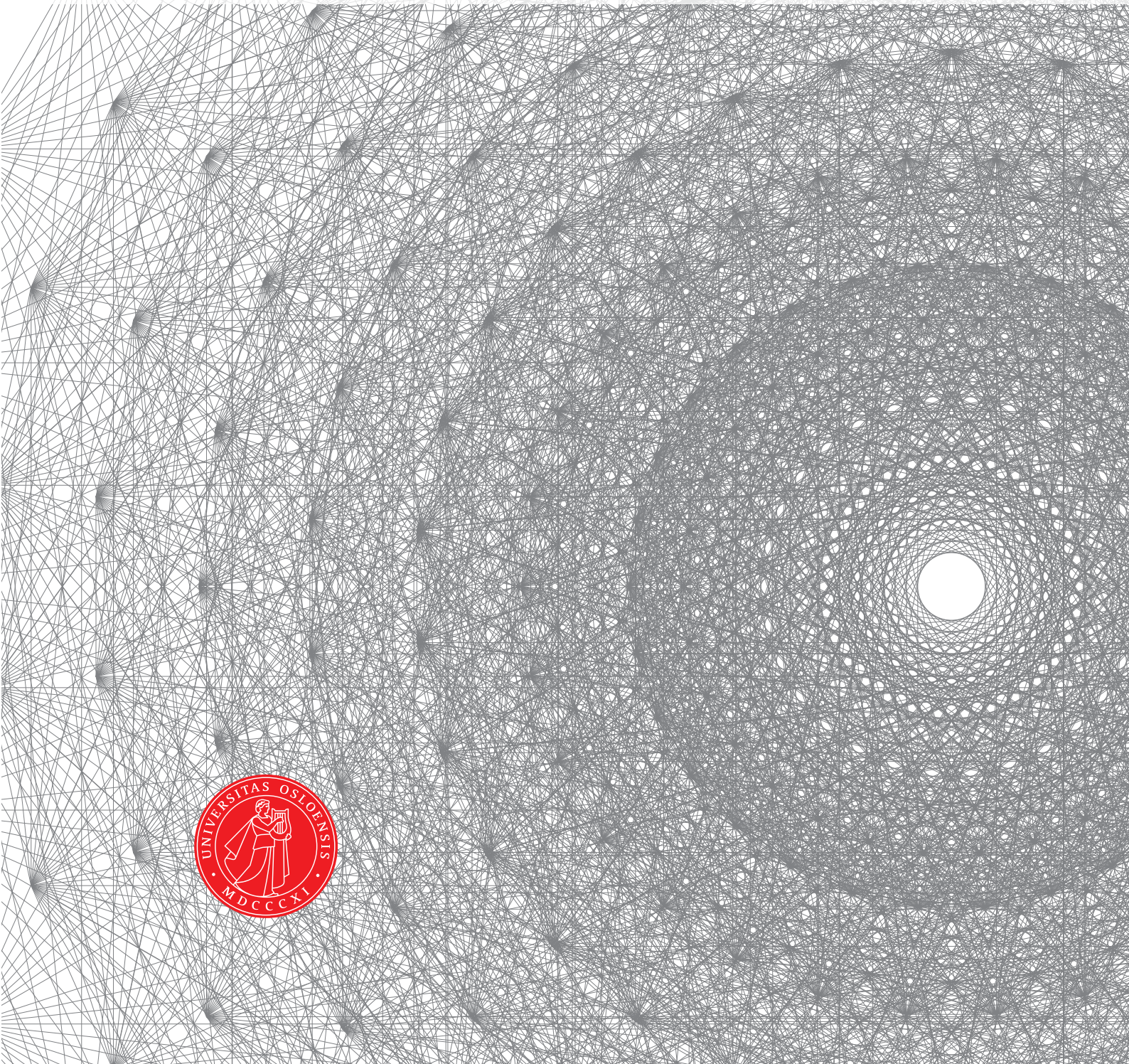
UiO : Department of Mathematics
University of Oslo

NEED A TITLE!!!

Peter Even Killingstad
Master's Thesis, Spring 2018



This master's thesis is submitted under the master's programme *Computational Science and Engineering*, with programme option *Mechanics*, at the Department of Mathematics, University of Oslo. The scope of the thesis is 30 credits.

The front page depicts a section of the root system of the exceptional Lie group E_8, projected into the plane. Lie groups were invented by the Norwegian mathematician Sophus Lie (1842–1899) to express symmetries in differential equations and today they play a central role in various parts of mathematics.

Abstract

write a short abstract here

Contents

Chapter 1

Introduction

Figure 1.1: Internal wave below the barge
Vizualisation of the internal wave below the barge made by simulating with a pycnocline at 0.2 m and $Fr_h = 0.95$

Chapter 2

Theory: Governing equations, Turbulence, and Numerical methods.

2.1 Governing equations

The study of viscous flow reaches back to acient times. Humans figured out, with clever intuision, trial and error, the importance of viscous friction. Long before any real theoretical understanding of fluid flow, streamlined weapons and boats have been made to overcome the effects of viscosity. In more recent years the understanding has taken huge leaps. Today we have tools to master viscous flow with the help from theory. The fundamental equation describing viscous flow is called Navier-Stokes equation[3]. With the following assumptions: incompressible, newtonian fluids. Density and viscosity may be different, but for one fluid it is constant. This gives the following equations:

$$\nabla \cdot \mathbf{u} = 0 \tag{2.1}$$

$$\frac{\partial \rho \mathbf{u}}{\partial t} + \nabla \cdot (\rho \mathbf{u}\mathbf{u}) = \nabla p + \mu \nabla^2 \mathbf{u} + \rho \mathbf{g} + \mathbf{f}_{st} \tag{2.2}$$

where (2.1) is the conservation of mass and (2.2) is the conservation of momentum. $\mathbf{u}$ is the velocity, p is the pressure, density is given by $\rho(\mathbf{x}, t)$, the dynamic viscosity is given by μ, $\mathbf{g}$ and $\mathbf{f}_{st}$ is the gravity and surface tension respectively. There are four independent variables, the spatial x, y and z coordinates, and the time t.

The equations are a set of coupled differential equations. In practice, these equations are to difficult to solve analytically, but solutions for some simple geometries and boundary conditions can be found. More complex cases, as almost all flows of engineering significance, needs to bee approximated with numerical techniques. Computational fliud dynamics has become the most viable tool to represent the complex Navier-Stokes equations. Some CFD models will be briefly introduced later in this chapter.

Using Navier stokes (2.1 and 2.2), the goal would be to investigate the forces in most industrial and academic applications. Finding solutions to the equations, either analytical or approximations, makes it possible to calculate the force vector by using:

$$\mathbf{F} = \int_{surface} \Big(\nu \big(\nabla \mathbf{u} + (\nabla \mathbf{u})^T \big) - p\mathbf{I} \Big) \cdot \mathbf{n} ds. \tag{2.3}$$

$\mathbf{n}$ is the unit vecctor normal to the surface, $\mathbf{I}$ is the unit matrix and ν is the kinematic viscosity.

Having obtained the force vector, it is usual to quantify the the drag force or resistance by introducing the drag coefficient, a dimensionless quantity given by

$$C_d = \frac{F_d}{\frac{1}{2}\rho S u^2}. \tag{2.4}$$

Here F_d is the drag force, ρ is density, u is the freestream velocity and S is the surface area.

Viscous flows can be divided into several regimes, and the primary controlling parameter is the dimensionless Reynolds number[3].

$$Re = \frac{u_0 l_0}{\nu} \tag{2.5}$$

where u is a velocity scale, l_0 is a characteristic geometric size and ν is the kinematic viscosity. Re represents the ratio of inertial forces to viscous forces within a fluid flow. Fluid properties can cause dramatic changes to the flow patterns. It is usual to divide flows into three distinct regimes.

- Low Re flow:
  Viscous forces dominates and the flow is smooth or laminar regime.
- Intermidieate Re flow:
  Flow is in a transitional region where it is partly fluctuating and partly laminar regime.
- High Re flow:
  Inertial forces dominates and the flow is fluctuating or in a turbulent regime.

Objects advancing through fluids such as water generate free surface waves. At the same time, if an object is advancing through stratified fluids, it generates internal waves. While free surface waves depend on the Froude number:

$$Fr = \frac{U_0}{\sqrt{gL_0}}, \tag{2.6}$$

internal waves depends on the densimetric Froude number:

$$Fr_h = \frac{U_0}{c^*}. \tag{2.7}$$

U_0 is the ship speed, g is gravity and L_0 is the ship length at water level. c^* represents the celerity of the longest internal waves. For an infinitely deep denser fluid, c^* is given as:

$$c^* = \sqrt{gh\frac{\Delta\rho}{\rho_0}} \tag{2.8}$$

where h is the distance from the free surface to the pycnocline, $\Delta\rho$ is the difference beetwen the density of the stratified fluid.

2.2 Turbulence

2.2.1 Physical concepts of turbulence

Turbulence is the phenomenon where a fluid flow appears to be chaotic and random. Unlike laminar flow, where the fluid flows in an orderly fashion, turbulent flow is rapidly fluctuating in all spatial dimensions. Structures of the flow is varying from large scales comparable to the dimensions of the physical boundaries to small scales.

The main characteristics of turbulence is the transfer of energy from larger spatial scales into smaller, happening in a three dimensional space and time(need reference).

To discuss the main characteristics of turbulence, a usefull concept is that of an "eddy". An eddy can be thought of as typical turbulence pattern of small- and large- scales all co-existing in the same fluid. The eddies consists of vortexes intertwined in a chaotic manner, beeing streched by the mean flow and pulled in random directions by one another(need reference). This mechanism ultimately leads to braking of the eddies into smaller ones, leading to an "energy cascade"(need reference).

The kinetic energy of the mean flow is extracted by the largest scale eddies. Energy from the largest eddies is further extracted to smaller scales, and the kinetic energy is finally dissipated into thermal energy by the small scale eddies(need reference).

The turbulent kinetic energy is defined by the units $\sim [m^2/s^2] \sim u_0^2$. As turbulence is dissipative, the dissipation rate has the units $[m^2/s^3]$. The dissipation rate of kinetic energy scales as $\epsilon \propto u_0^3/l_0$ [6], where u_0 and l_0 is the characteristic velocity and length of the flow.

The dissipation rate of kinetic energy is one of the most important results of turbulence theory, and is reffered to as the Kolmogorov relation [6]. A turbulent eddy with kinetic energy u_0^2 either looses its energy or breaks up into smaller eddies in one time scale or period $T \sim l_0/u_0$.

As Reynolds number is very large for turbulent flow, i.e $\frac{u_0 l_0}{\nu} \gg 1$, the large scale eddies is independent of the viscosity. To see how viscosity is effecting the turbulent flow, "Kolmogorov micro-scales" can be constructed. Using kinematic viscosity ν and the dissipation rate ϵ, small scale velocity, length and time can be written out as:

$$\eta = \left(\frac{\nu^3}{\epsilon}\right)^{1/4}, \tag{2.9}$$

$$v = \left(\nu\epsilon\right)^{1/4}, \tag{2.10}$$

$$\tau = \left(\frac{\nu}{\epsilon}\right)^{1/2}, \tag{2.11}$$

where η is the micro length-scale, v is the velocity and τ is the time scale. Reynolds number in the "micro-scale" is given by $\frac{v\eta}{\nu} = 1$, hence the viscosity is of big importance at theese scales. We have that the "micro-scale" eddies is dominated by friction, and small-scale turbulence is almost independent of large scale turbulence for large enough reynolds number.

2.2.2 Boundary layer

Using Reynolds number (2.5) it is easily shown that, for a thin shear layer flow over a flat plate, inertial forces dominate. A Reynolds number based on $U = 1\text{m}s^{-1}$, $l_0 = 0.1\text{m}$ and $\nu = 10e-6m^2s^{-1}$ would be $Re_{l_0} = 10e5$, and inertial forces are much larger than viscous forces.

If Reynolds number is based on a length scale y that is decreasing towards 0, Re_y would eventually be $\mathcal{O}(1)$. The viscous forces is then either equal or bigger than the inertial forces.

This close to the wall, the mean velocity depends on the distance y, fluid density ρ, viscosity ν and the wall shear stress τ_w, and is usually called the turbulent boundary layer [8].

By using dimensinal analysis, flow behaviur can be expressed by means of dimensionless groups u^+ and y^+.

The dimensionless groups are given by [8]

$$u^+ = \frac{U}{u_*} \tag{2.12}$$

$$y^+ = \frac{yu_*}{\nu} \tag{2.13}$$

$$u_* = \sqrt{\frac{\tau_w}{\rho}} \tag{2.14}$$

u_* is the friction velocity.

Turbulent boundary layer consists of two regions:

- Inner region, consists of tre layers:
  - where viscous stresses dominates,
  - where turbulent stresses dominate,
  - where viscous and turbulent stresses are of similar magnitude.
- outer region
  - inertia dominated flow far from the wall.

The layer where viscous stresses dominates is called the viscous sub-layer. It is a linear relationship between the velocity and distance to the wall and is given as [8]

$$u^+ = y^+ \tag{2.15}$$

The layer is extremely thin, and lies at $y^+ < 5$ [8].

At the layer where turbulent stresses dominates, called the log-layer, the velocity and distance to the wall has a logarithmic relationship. It is given as [8]

$$u^+ = \frac{1}{\kappa}\ln(y^+) + B \tag{2.16}$$

The constant κ and B is found by doing measurements. The log-layer lies outside the viscous sub-layer, at $30 < y^+ < 300$ [**?**].

Between the viscous sub-layer and the loq-layer lies the buffer layer, where the viscous and turbulent stresses are of similar magnitude.

Equations for the viscous sub-layer and loq-layer (2.15 and 2.16) are usually called law of the wall and are of great importance in approximating and simulating turbulent boundary layers.

2.2.3 Computer modelling of turbulence

The modelling of turbulent fluid flows and the Navier-Stokes equations has seen huge advancements as computer speed increases. The number of applications of fluid flow predictions has grown and computerized analysis has become a crucial part in the field of fliud mechanics.

There are three main methods for numerically solving Navier-Stokes equations:

- Reynolds Averaged Navier Stokes(RANS)
- Large Eddy Simulations(LES)
- Direct Numerical Simulations(DNS)

LES and DNS is introduced rather briefly, while RANS will be introduced in a litle bit more detail.

RANS

For many enginering purposes, the focus is on the mean effect of turbulence, and it is unecessary to resolve the details on all scales.

A key part of RANS is investigating the effects of fluctuations on the mean flow using Reynolds de-composition. The velocity and pressure is decomposed as:

$$\mathbf{u} = \mathbf{U} + \mathbf{u}' \tag{2.17}$$

$$p = P + p', \tag{2.18}$$

where $\mathbf{u}$ is the instantaneous flow field, $\mathbf{U}$ is the mean flow field and $\mathbf{u}'$ is the fluctuating part. The same goes for the pressure, where P is mean and p' is the fluctuating part.

Substituting the de-composed velocity and pressure (2.17, 2.18) into Navier-Stokes equations (2.1, 2.2) and taking the time-mean, the following continuity and momentum equations using suffix notation is derived:

$$\frac{\partial U_i}{\partial x_j} = 0 \tag{2.19}$$

$$\frac{\partial U_i}{\partial t} + \frac{\partial}{\partial x_j}(U_i U_j) = -\frac{1}{\rho}\frac{\partial P}{\partial x_i} + \nu\frac{\partial^2 U_i}{\partial x_j \partial x_j} - \frac{\partial}{\partial x_j}(\overline{u_i' u_j'}). \tag{2.20}$$

Here U_i is the mean velocity, P is the mean pressure and $\overline{u_i'}$ is the mean fluctuating velocity.

In the momentum equation (2.20), the new term $\frac{\partial}{\partial x_j}(\overline{u_i'u_j'})$ is derivative of the Reynolds stress-tensor [11]. It appears from the convective part $\mathbf{u}\cdot\nabla\mathbf{u}$ of Navier-Stokes equations (2.2), and is not really stresses. The physical meaning of the term is the averaged effect of turbulent advection on the mean flow field [11].

With the new term, the RANS equations is unclosed, with 6 more unknowns appearing from the Reynolds stress-tensor. The four equations has in total 10 unknowns (pressure, three velocity components and six "stresses"). In order to close the problem, enough equations must be found to solve for all the unknowns. In many models, such as one-equation and two-equation models (see [9] chapter 4), the Boussinesq eddy-viscosity approximation [8] is assumed to be valid. The Reynolds stresses are modelled as follows:

$$\tau_{ij} = \overline{u_i'u_j'} = \frac{2}{3}k\delta_{ij} - \nu_t\left(\frac{\partial U_i}{\partial x_j} + \frac{\partial U_j}{\partial x_i}\right) \tag{2.21}$$

$$k = \frac{1}{2}\overline{u_i'u_i'} = \frac{1}{2}\left(\overline{u_1'^2} + \overline{u_2'^2} + \overline{u_3'^2}\right) \tag{2.22}$$

where k is the turbulent kinetic energy per unit mass, ν_t is the turbulent or eddy viscosity and δ_{ij} is the Kronecker delta.

Substituting the Boussinesq eddy-viscosity approximation (2.21) into the RANS momentum equation (2.20) leads to:

$$\frac{\partial U_i}{\partial t} + \frac{\partial}{\partial x_j}(U_iU_j) = -\frac{1}{\rho}\frac{\partial P'}{\partial x_i} + \frac{\partial}{\partial x_j}\Big[\left(\nu + \nu_t\right)\frac{\partial U_i}{\partial x_j}\Big]. \tag{2.23}$$

where $P' = \left(P + \frac{2k}{3}\delta_{ij}\right)$ called the modified pressure[10], often used in CFD.

To complete the closure of the RANS equations, the eddy viscosity term ν_t needs to be modelled. Dimensional analysis dictates that ν_t needs to be proportinal to the product of a characteristic velocity and a characteristic length scale [2, 9]

This paper is using two turbulence models for comparison, namely the two-equation models k-ϵ and the shear stress transport model k-ω SST.

k-ϵ model

The standard k-ϵ model equations is found in a lot of litterature such as [8] and [11].

The k-ϵ model is perhaps the most widely used, giving good results in classical shear flows. It does however have shortcomings in accurately predicting adverse pressure gradients and boundary layers. despite its shortcomings, the model is recommended in cases with multiphase problems by som literature, i.e [2].

The model equations are specified as follows:

Kinematic eddy viscosity equation:

$$\mu_t = \rho C_\mu \frac{k^2}{\epsilon} \tag{2.24}$$

Turbulent kinetic energy equation:

$$\frac{\partial \rho k}{\partial t} + U_j \frac{\partial \rho k}{\partial x_j} = \frac{\partial}{\partial x_j}\Big[\frac{\big(\mu + \mu_t\big)}{\sigma_k}\frac{\partial k}{\partial x_j}\Big] - \rho\epsilon + \rho\tau_{ij}\frac{\partial U_i}{\partial x_j} \tag{2.25}$$

Turbulence dissipation rate equation:

$$\frac{\partial \rho \epsilon}{\partial t} + U_j \frac{\partial \rho \epsilon}{\partial x_j} = \frac{\partial}{\partial x_j}\Big[\frac{\big(\mu + \mu_t\big)}{\sigma_\epsilon}\frac{\partial \epsilon}{\partial x_j}\Big] + \rho C_{\epsilon 1}\frac{\epsilon}{k}\tau_{ij}\frac{\partial U_i}{\partial x_j} - \rho C_{\epsilon 2}\frac{\epsilon^2}{k} \tag{2.26}$$

The model equations use the Boussinesq assumption given in equation (2.21) and contains five adjustable constants: $C_\mu = 0.09$, $\sigma_k = 1.0$, $\sigma_\epsilon = 1.3$, $C_{\epsilon 1} = 1.44$ and $C_{\epsilon 2} = 1.92$

k-ω SST

K-ω model is made by Menter [18] and is found in littarture and such as [8] and [9].

The k-ω SST turbulence model is a more advanced model which combines k-ϵ and k-ω models. The k-ϵ model has its shortcomings, as stated above. The $k-\omega$ model was made to better predict adverse pressure gradients and boundary layers, but preformed poorer at free streams. Menter [4] proposed a new model which combined the k-ϵ and the k-ω models. In litterature like [8] it is stated that k-ω SST model is superior in approximating adverse pressure gradients and boundary layers.

Turbulence resources such as [5] gives thurough desrcription of the model equations. The two-equation model is specified as follows:

Kinematic eddy viscosity equation:

$$\mu_t = \frac{\rho a_1 k}{max\big(a_1\omega, \Omega F_2\big)} \tag{2.27}$$

Turbulent kinetic energy equation:

$$\frac{\partial \rho k}{\partial t} + U_j \frac{\partial \rho k}{\partial x_j} = \frac{\partial}{\partial x_j}\Big[\big(\mu + \sigma_k \mu_t\big)\frac{\partial k}{\partial x_j}\Big] + \rho P - \beta^* \rho\omega k \tag{2.28}$$

Turbulent specific dissipation rate equation:

$$\frac{\partial \rho \omega}{\partial t} + U_j \frac{\partial \rho \omega}{\partial x_j} = \frac{\partial}{\partial x_j}\Big[\big(\mu + \sigma_\omega \mu_t\big)\frac{\partial k}{\partial x_j}\Big] + \frac{\gamma}{\nu_t}\rho P - \beta\rho\omega^2 + 2\big(1 - F_1\big)\frac{\rho\sigma_{\omega 2}}{\omega}\frac{\partial k}{\partial x_j}\frac{\partial \omega}{\partial x_j} \tag{2.29}$$

Here $P = \tau_{ij}\frac{\partial U_i}{\partial x_j}$, and τ_{ij} is the Boussinesq assumption (2.21).

LES

While RANS has the main focus on the mean flow, LES is resolving large scale turbulence. While the effects of large eddies on the flow is resolved, the effect of the small scale eddies is included by a sub-grid scale models [8].

In LES modeling, a spatial filter is used to separate small sclaes from large scales. The method is started off with a filtering function and a "cutoff" witdh, where all scales greater

than the "cutoff" width is resolved.

A filtering operation of the filter function function is done in the following manner [8]:

$$\overline{\phi}(\mathbf{x},t) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} G(\mathbf{x},\mathbf{x}'\Delta)\phi(\mathbf{x}',t)dx_1'dx_2'dx_3', \tag{2.30}$$

where $G(\mathbf{x},\mathbf{x}'\Delta)$ is the filtering function, $\overline{\phi}(\mathbf{x},t)$ is the filtered function, $\phi(\mathbf{x}',t)$ is the original unfiltered function and Δ is the "cutoff" witdh. The overbar indicates spatial filtering and not time averaging as with RANS.

Filtering the Navier-stokes equations (2.1 and 2.2) gives the LES continuity and momentum equations as follows:

$$\frac{\partial \overline{u}_i}{\partial x_j} = 0 \tag{2.31}$$

$$\frac{\partial \overline{u}_i}{\partial t} + \frac{\partial \overline{u}_i\overline{u}_j}{\partial x_j} = -\frac{1}{\rho}\frac{\partial \overline{p}}{\partial x_i} + \nu\frac{\partial^2 \overline{u}_i}{\partial x_j \partial x_j} - \frac{\partial \tau_{ij}}{\partial x_j}. \tag{2.32}$$

Here the overline denotes filtered flow variable, and τ_{ij} is the sub-grid scale stresses. The sub-grid scale stresses are part of the unresolved sub-grid scales.

For further introduction of the LES model, Books such as [8] and [7] and papers such as [2] is recomended.

DNS

DNS involves numerical solution of the full Navier-Stokes equations. The method resolves all scales, including the kolmogorov scales (2.9, 2.11 and 2.10). It takes the closed form of the four equations and four unknowns and solves it on a sufficiently fine mesh and small enough time step. For flows with small enough Reynolds number (2.5), DNS can serve as a benchmark for the other turbulence models [2].

Using "Kolmogorov's micro- scales" (2.9 and 2.11), ratio's of the largest and smallest scales can be obtained. The ratio of the largest and the smallest scales are proportional to $Re^{3/4}$ and the ratio of the largest and smallest time scale is proportional to $Re^{1/2}$. $Re = 10^4$ requires a spatial resolution of $\mathcal{O}(10^3)$ in each direction, and the simulation must run for atleast 100 time steps. Computing meshes with 10^9 grid points with 100 time steps is very demanding, even with a modest Reynolds number. Computing industrial flows with higher Reynolds number is impossible with current technology [2, 8].

2.3 Volume of fluid

When there is multiple fluids in a computation, there is need for an interface tracking or interface capturing. To handle multiple-fluid interactions, the volume of fluid method used.

For each fluid component, a volume fraction is introduced. If V is a volume of a cell

in a computaional domain, and $\alpha(\mathbf{x}, t)$ is the volume fraction, the volume fraction for two fluids is defined as [13]

$$\alpha(\mathbf{x}, t) = \begin{cases} 1, & \mathbf{x} \in \Omega_l \\ 0, & else \end{cases} \tag{2.33}$$

where Ω_l is the part of the domain covered by one fliud l.

On each grid cell, the integral of the color function is approximated. The discrete volume fraction is written as [13]

$$\alpha_i = \frac{1}{V} \int_V \alpha(\mathbf{x}, t) dV \tag{2.34}$$

where the subscript i denotes the i'th fluid in a system.

For miscible fluids the volume fractions are governed by the advection-diffusion equation given as [12]

$$\frac{\partial \alpha_i}{\partial t} + \nabla \cdot (\alpha \mathbf{u}) = D \nabla^2 \alpha, \tag{2.35}$$

Where D denotes diffusivity between miscible fluids. The following constraint must be satisfied due to mass conservation:

$$\Sigma_{i=1}^{n} \alpha_i = 1 \tag{2.36}$$

Desity and viscosity are defined as:

$$\rho = \Sigma_i \rho_i \alpha_i \tag{2.37}$$

$$\mu = \Sigma_i \mu_i \alpha_i \tag{2.38}$$

2.4 Numerical discretization

Numerical discretization is the prosses of transferring differential equations, such as the Navier- Stokes equations (2.1 and 2.2), into descrete counterparts. A discretization method is needed in order to evaluate the equations on computers.

There are three discretisation methods generally used when approximating equations. Finite difference method, finite element method and finite volume method.

As in most commercial well-established CFD codes, this thesis is using the finite volume method. Fvm is one of the most versitaile discretization techniques used in cfd [8].

An outline for the disctretization procedure can be presented with the following steps:

- Integration of the governing equations of fluid flow all over the finite control volumes of the domain.
- Conversion of the resulting integral equations into a system of algebraic equations.

A steady one convection-diffusion equation of a property ϕ without a source term is given as:

$$\nabla \cdot (\rho \mathbf{u} \phi) = \nabla \cdot (\Gamma \nabla \phi), \tag{2.39}$$

where ρ is the density, $\mathbf{u}$ is a known velocity and Γ is diffusivity.

Integrating convection-diffusion equation (2.39) over a control volume (CV) gives:

$$\begin{aligned} \int_{CV} \nabla \cdot (\rho \mathbf{u} \phi) dV &= \int_{CV} \nabla \cdot (\Gamma \nabla \phi) dV \\ \Rightarrow \int_A \mathbf{n} \cdot (\rho \mathbf{u} \phi) dA &= \int_A \mathbf{n} \cdot (\Gamma \nabla \phi) dA. \end{aligned} \tag{2.40}$$

Gauss integration, i.e. Gauss theorem [8] is applied to change the integration over a control volume to an integration over an area.

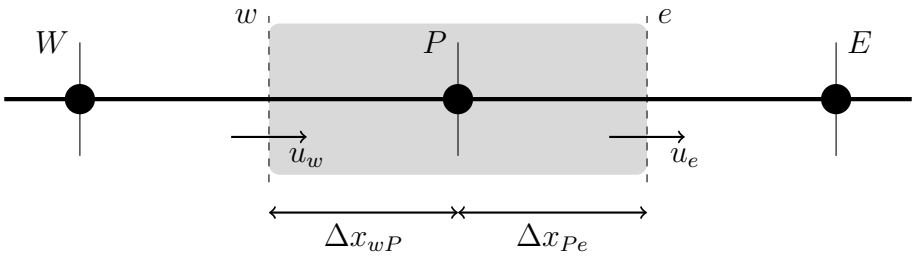


Figure 2.1: Control Volume around a cell center P

In one dimension and using the control volume shown in figure 2.1 the integration of the convection-diffusion equation (2.40) gives:

$$\left(\rho u A \phi\right)_e - \left(\rho u A \phi\right)_w = \left(\Gamma A \frac{d\phi}{dx}\right)_e - \left(\Gamma A \frac{d\phi}{dx}\right)_w. \tag{2.41}$$

In order to convert the resulting equation (2.41) into a system of algebraic equation, differencing schemes is used.

Using central differencing on $d\phi/dx$ at the faces e and w and rewriting the property ϕ at the same faces as:

$$\phi_w = \frac{\phi_W + \phi_P}{2} \tag{2.42}$$

$$\phi_e = \frac{\phi_P + \phi_E}{2} \tag{2.43}$$

Substitution of (2.42) and (2.43) into the resulting equation (2.41) yields the central difference expression:

$$\frac{\rho u_e A_e}{2}(\phi_P + \phi_E) - \frac{\rho u_w A_w}{2}(\phi_W + \phi_P) = \Gamma A_e \frac{(\phi_E - \phi_P)}{\Delta x_{PE}} - \Gamma A_w \frac{(\phi_P - \phi_W)}{\Delta x_{WP}} \tag{2.44}$$

Rewriting the central difference expression (2.44), and solving for ϕ_P gives:

$$\begin{aligned} \Big[(\frac{\rho u_e}{2} - \frac{\Gamma}{\Delta x_{PE}})A_e - (\frac{\rho u_w}{2} - \frac{\Gamma}{\Delta x_{WP}})A_w\Big]\phi_P = \\ \Big[(-\frac{\rho u_e}{2} + \frac{\Gamma}{\Delta x_{PE}})A_e\Big]\phi_E + \Big(\frac{\rho u_w}{2} + \frac{\Gamma}{\Delta x_{WP}})A_w\Big]\phi_W \end{aligned} \tag{2.45}$$

If it is well defined by boundary conditions, the above equation (2.45) can be solved as a system of linear equation i.e.:

$$\mathbf{Ax} = \mathbf{b} \tag{2.46}$$

Where $\mathbf{A}$ is a $m \times n$ matrix, $\mathbf{x}$ is a column vector with n entries and b is a column vector with m entries.

Using the FVM, the discretizations is carried directly in the physical domain. There is no need for any transformation between the physical and computational coordinate, making the FVM flexible and popular method for computational fluid dynamics [15].

Chapter 3

OpenFOAM

All computanial fluid dynamis is structured around numerical algorithms that tackles fliud flow problems. Classical solvers are implemented in well-established CFD codes such as CFX/ANSYS, FLUENT and OpenFOAM. The software used in this thesis is OpenFOAM.

As in all well-established CFD codes, the work flow consists of three main elements:

- pre-processor
- solver
- post-processor

Pre-processing consists of input of a flow problem, in order to make it well-defined before the solving process begins.

The solver is solving the flow problem, by implemented numerical methods and algorithms suitable for the specific problem.

Post-processing consists of verification and validation of the solutions given by the solver. It essential to investigate data output and visualize. The complexity of fluid flows demands thurough investigation by e.g. comparing with existing experiments, either numerical or experimental.

3.1 Introduction to OpenFoam

This paper is using the free, open sorce software OpenFOAM (Field Operation and Manipulation). It is a C++ library of source code for solvers and utilities.

Figure 3.1 illustrates an initial state of an OpenFOAM case. Three directories are located in the case folder: 0, constant and system.

The 0 directory contains files for the different parameters essential for the problem. Each file defines initial values and boundary conditions for the numerical experiment.

The *constant* directory contains, as the name suggests, all the constants of the case. Constant variables such as gravitation *g*, *transportProperties* as density and viscosity and *turbulenceProperties* are defined in separate files. The subdirectory *polyMesh* contains the mesh geometry.

The *system* directory contains information about meshing of the numerical experiment, how to discretize and solve the equations. The file controlDict controls ehich solver is used, timecontrols and data output controls. As the names suggests, *fvSchemes* and *fvSolution* contains information about discretization schemes and solution algorithms respectively.

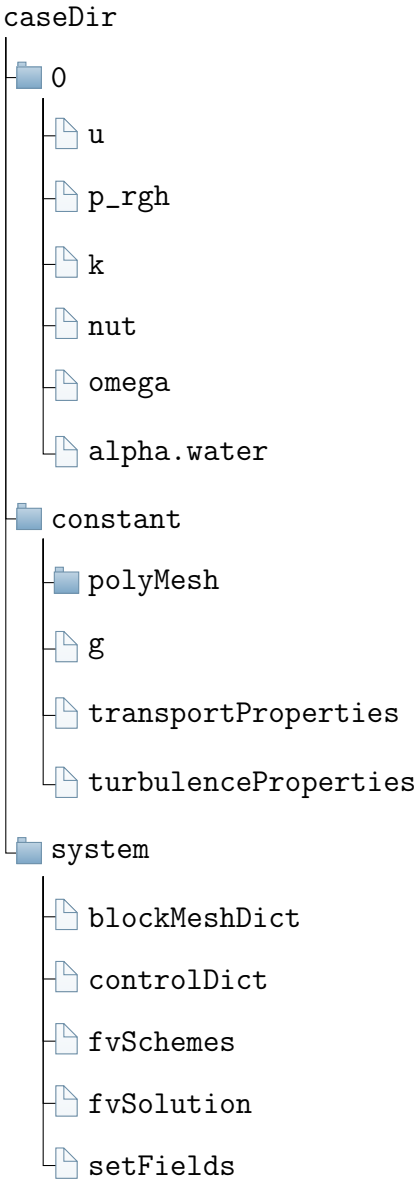


Figure 3.1: Example of case directory structure in openFOAM.

3.2 TwoLiquidMixingFoam

OpenFOAM has solvers for a wide range of applications. As mentioned in the section Introduction to OpenFOAM (3.1), the file *controlDict* found in figure 3.1 defines which

solver to use.

Many multiphase solvers can be found, both miscible and immiscible. This thesis is using the incompressible multiphase solver twoLiquidMixingFoam, where the liquids are miscible.

Three equations are beeing solved. The first equation is the alpha diffusion equation, given by [17]

$$\frac{\partial \alpha_1}{\partial t} + \nabla \cdot (\mathbf{U}\alpha_1) = \nabla \cdot \Big(\big(D + \frac{\nu_t}{S_C} \big) \nabla \alpha_1 \Big) \tag{3.1}$$

where

- α_1 is the volume fraction.
- $\alpha_2 = 1 - \alpha_1$
- $\rho = \alpha_1 \rho_1 + \alpha_2 \rho_2 = \alpha_1 \rho_1 + (1 - \alpha_1)\rho_2$.
- D is the molecular diffusivity.
- ν_t is the turbulent eddy viscosity.
- s_c is the Schmidt number, given as $\mu/(\rho D)$.

The continuity and momentum equation is given, respectively, as:

$$\nabla \cdot \mathbf{U} = 0 \tag{3.2}$$

$$\frac{\partial \rho \mathbf{U}}{\partial t} + \nabla \cdot (\rho \mathbf{U}\mathbf{U}) = -\nabla(p_{rgh}) - gh\nabla\rho + \nabla \cdot (\rho \boldsymbol{\tau}) \tag{3.3}$$

where

- $\boldsymbol{\tau} = -\frac{2}{3}\overline{\mu_{eff}}\nabla \cdot \mathbf{U}\mathbf{I} + \overline{\mu_{eff}}\nabla\mathbf{U} + \overline{\mu_{eff}}\big(\nabla\mathbf{U}\big)^T$.
- $\overline{\mu_{eff}} = \alpha_1(\mu_{eff})_1 + \alpha_2(\mu_{eff})_2$.
- $(\mu_{eff})_i = (\mu - \mu_t)_i$. Subscript i denotes either fluid 1 or 2.

The term $\nabla(p_{rgh})$ and $gh\nabla\rho$ is obtained by using $P = p_{rgh} + \rho gh$. The Solver is using the Boussinesq eddy-viscosity approximation (refeqn:bossinesq), and p is the modified pressure and containes the kinetic energy per unit mass $\frac{2}{3}k\delta_{ij}$.

3.3 Turbulence models

This paper is using the RANS method to approximate turbulence. In order to close the RANS equations, turbulence models are beeing used. Two different two-equation models are implemented for comparison.

The turbulence models used in this paper are the k-ω SST and the k-ϵ turbulence model. In section Introduction to openFOAM 3.1, the *constant* directory shown in figure 3.1 contains the file *turbulenceProperties*. Here it is defined what simulation type that is beeing used, i.e. RANS. It also specifies which turbulence model to use.

3.4 Numerical schemes

Having defined the equations to be solved in in section *TwoLiquidMixingFoam* (3.2), the way in which to discretize the equations are defined in the file *fvSchemes* found in figure 3.1. The file consists of sub-dictionaries with name corresponding to terms within the equations given in section *TwoLiquidMixingFoam* (3.2). For each term a numerical scheme must be specified.

Time derivative term $\partial/\partial t$ are discretized by using an *Euler* scheme, a first order implicit difference scheme and is given in the sub-dictionary *ddtSchemes*.

The other terms are discretized using finite volume method. As explained in section *Numerical discretization* 2.4, the fvm discretization procedure is done by using gaussian integration and converting the resulting terms into algebraic equations using differencing schemes.

Gradient terms ∇ is set in the sub-dictionary *gradSchemes*. The terms are discretized by using *Gauss linear*, where the *Gauss* entry denotes Gaussian integrationand the *linear* entry denotes a central differencing scheme.

Divergence terms $\nabla\cdot$ is set in the sub-dictionary *divSchemes*. There are several divergence terms needing discretization, and specified as:

- *div(rhoPhi,U) Gauss linearUpwind grad(U)*
- *div(phi,alpha) Gauss vanLeer*
- *div(phirb,alpha) Gauss linear*
- *div(phi,k) Gauss upwind*
- *div(phi,omega) Gauss upwind*
- *div(((rho*nuEff)*dev2(T(grad(U))))) Gauss linear*

The *phi* entry in the *div(phi,...)* is the volumetric flux of velocity. Gauss entry is the Gaussian integration and last entry is the differencing scheme.

Laplacian terms ∇^2 is set in the sub-dictionary *laplacianSchemes*, and is discretized as *Gauss linear* with the Gaussian integration and a linear differencing scheme.

3.5 Solution algorithms

There are several solution algorithms implemented in openFOAM. Solution procedures are needed in order to obtain solutions to pressure and momentum. In multiphase modelling, additional solution procedures are needed in order to obtain solution to the phase fraction and the volume of fluid method.

There is three different algorithms implemented in OpenFOAM for solving the Navier-Stokes equations (2.2) and (2.1):

- Semi-implicit method for pressure-linked equations SIMPLE algorithm.
- pressure-implicit split-operator PISO algorithm.
- Pressure-implicit method for pressure-linked equations PIMPLE algorithm.

Numerical techniques are required for coupling the pressure and momentum quantities. All algorithms are iterative procedures for coupling momentum and pressure. SIMPLE is a steady state algorithm, PISO is a transient algorithm and PIMPLE is a semi transient algorithm.

The twoLiquidMixingFoam solver is using the PIMPLE algorithm for coupling the pressure and momentum. Since it is a multiphase solver, a multi-dimensionsal limiter for explicit solution i.e MULES algorithm is used for solving the phase fraction alpha.

Chapter 4

Simulation Design

Wether an numerical experiment is succesfull depends largely on the pre-processing. A computational domain has to be made well suited to handle the flow problem. Many factors has to be considered in order to have a good numerical experiment, with mesh quality and and boundary conditions as the key factors.

4.1 Simulation geometry

The geometry of the numerical experiment is dependent on computational demands. Ideally the geometry of the domain would include the entire barge and a farfield consisting of air, fresh- and salt-water. A domain including all aspects of the fluid problem would bee much more suitable as the case would bee more physical.

With a three dimensional problem such as the "dead-water" phenomenon, the computational costs constrains the experiment to some parts of the flow problem. This and the scope of this thesis, which is limited to five months, makes it necessary to make compromise between physics and assumptions. The experiment becomes less physical, but the idea is to isolate certain aspects of the problem to investigate, namely the effects of the internal wave.

The geometry of the computational domain consists of a barge surface and a farfield. Farfield includes inlet, outlet, atmosphere, bottom, front and back in order to have a closed domain. Only the draft of the barge is included, making the top of the domain located at the free-surface.

Figure 4.1 is showing simulation gemoetry with patches atmosphere, front, inlet and outlet included. The Barge is colored red. The dimensions of the barge is taken from [1] and 0.6 m long (x-dir), 0.45 m wide (y-dir) and 0.35 m wide (z-dir). The geometry used in the numerical experiment is using a symmetry plane in y-direction, halving the computational domain. It further only includes what is below the free surface i.e. the draft of the barge. The dimensions of the barge is therefore 0.6m long, 0.225m wide and draft is 0.1m.

Farfield boundaries are located sufficiently far away to minimize effects of the boundaries on the solution. international Towing Tank Conference has a practical guidelines for ship CFD applications which states that inlet, outlet and "exterior boundary" should be located $1 - 2 \times$ length at draft [14]. This experiment is not modelling the free surface as

it is a boundary. It does however modell an internal wave and in combination with highly unstreamlined draft geometry, the resulting farfield dimensons are:

- inlet located 10.0 m upstream in front of barge.
- outlet located -15.0 m downstream of barge.
- bottom located at -4.0 m below barge.
- back is located 3.0 m at the width side of barge.

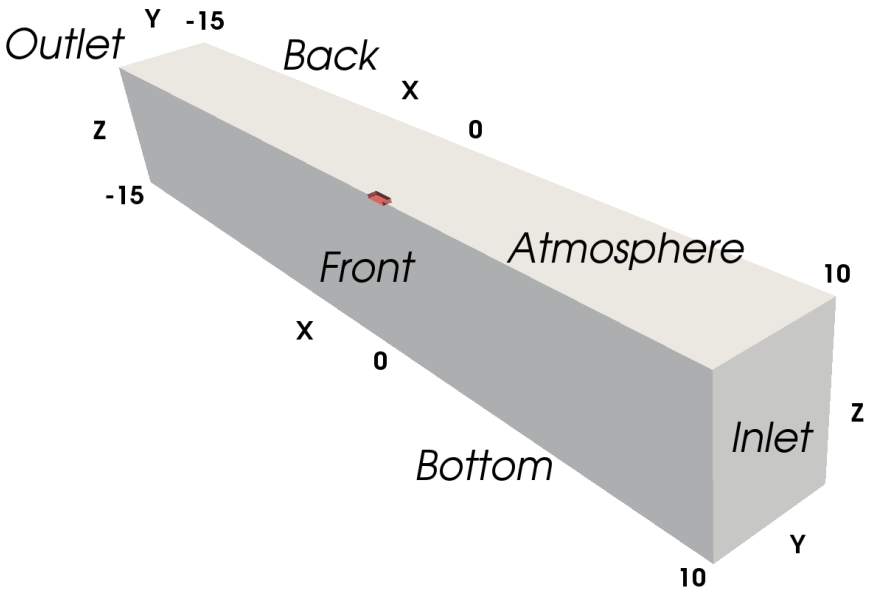


Figure 4.1: Geometry.
Geometry used in the computational experiment. Draft of barge is colored red at $X = 0$ while farfield with patches inlet, outlet, front and atmosphere included.

4.2 Simulation set-up

Simulations is done with a constant draft $D = 0.1$ m. The different experiments is conducted by varying the densimetric Froude number in the range $0.3 \leq Fr_h \leq 1.35$. The densimetric Froude number is varied by changing speeds at constant pycnocline depths giving $h/D = 1$, 1.5 and 2.

Figure 4.2 is showing a schematic overview of the numerical experiment.

The experiments are done with a moving reference frame, rather than having the barge moving, in order to be able to run simulations sufficiently long.

The densities of the fluids are set to 997 kg m^{-3} for fresh water and 1024 kg m^{-3} for salt water. Kinematic viscosity is set to $\nu = 1.79 \cdot 10^{-6} m^2 s^{-1}$ corresponding to waters at 0°C. The speed varies as $0.08 \leq U \leq 0.22$ corresponding to Reynolds number (2.5) $26815 \leq Re \leq 73743$.

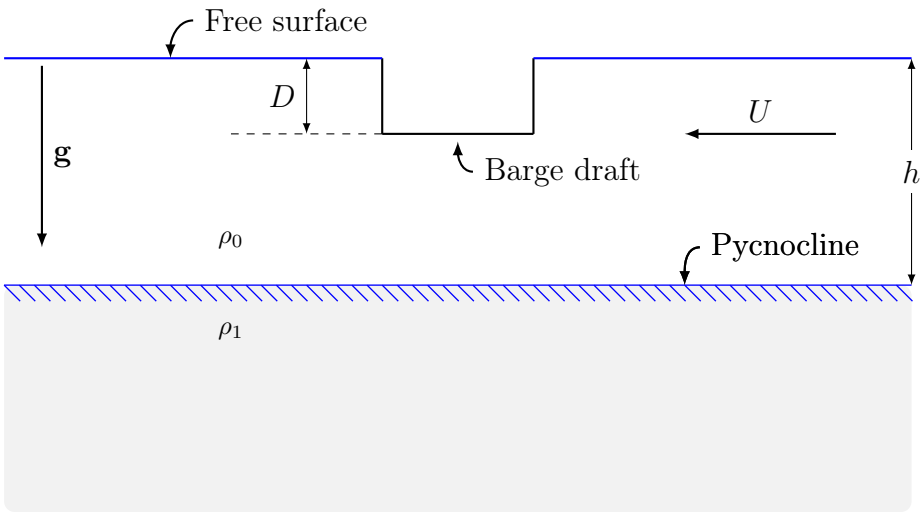


Figure 4.2: Simulation set-up.
Simulations is done by varying speeds U at constant $h/D = 1$, 1.5 and 2. Densities ρ_0 and ρ_1 is that of fresh- and salt-water respectively for stratified fluid simulations. For non-stratified fluid simulations, $\rho_0 = \rho_1$

4.3 Boundary and Initial conditions

Before simulating and solving the numnerical experiment, the momentum and continuity equations (2.2 2.1) must have appropriate initial and boundary conditions.

Boundary conditions are required component of the mathematical model that directs the motion of the flow. It specifies the fluxes such as mass and momentum into and out of the computational domain.

OpenFOAM is representing boundaries as patches consisting of faces, and all inital and and boundary data are assigned to the patches. Table 4.1 is showing an overview of all initial and boundary conditions for simulations using the turbulence modell k-omega SST .

Boundary \ Variable		U	p_rgh	alpha.saltWater	k	omega	nut
Inlet	Type	fixedValue	fixedFluxPressure	fixedValue	fixedValue	fixedValue	fixedValue
	Value	internalField	internalField	internalField	internalField	internalField	internalField
Outlet	Type	O.P.M.V.	zeroGradient	V.H.F.R.	inletOutlet	inletOutlet	zeroGradient
	Value	internalField	-	internalField	internalField	internalField	-
Barge	Type	M.W.V.	F.F.P.	zeroGradient	kqrW.F.	omegaW.F.	nutUSpaldinW.F.
	Value	(0 0 0)	internalField	-	internalField	internalField	internalField
atmosphere	Type	slip	fixedValue	zeroGradient	zeroGradient	zeroGradient	zeroGradient
	Value	-	internalField	-	-	-	-
atmosphereFrontOfBarge	Type	inletOutlet	fixedValue	zeroGradient	zeroGradient	zeroGradient	zeroGradient
	Value	internalField	internalField	-	-	-	-
Front	Type	symmetryPlane	symmetryPlane	symmetryPlane	symmetryPlane	symmetryPlane	symmetryPlane
	Value	-	-	-	-	-	-
Back	Type	symmetryPlane	symmetryPlane	symmetryPlane	symmetryPlane	symmetryPlane	symmetryPlane
	Value	-	-	-	-	-	-
Bottom	Type	symmetryPlane	symmetryPlane	symmetryPlane	symmetryPlane	symmetryPlane	symmetryPlane
	Value	-	-	-	-	-	-

Table 4.1: Boundary and initial conditions twoLiquidMixing with k-omega SST turbulence model.
O.P.M.V. = OutletPhaseMeanVelocity, V.H.F.R. = variableHightFlowRate, M.W.V. = movingWallVelocity, F.F.P. = fixedFluxPressure, kqrW.F. = kqrWallFunction, omegaW.F. = omegaWallFunction, nutUSpaldingW.F. = nutUSpaldingWallFunction

4.3.1 U

For velocity U, typical dirichlet boundary conditions is a applid at the inlet and on the barge. At the inlet there is *fixedValue* while a no-slip condition is set at the barge. The no-slip condition is a of a special kind, namely *movingWallVelocity*, which is applied for "moving" walls when it is a moving reference frame.

At outlet there is an *outletPhaseMeanVelocity*. This boundary condition adjusts the velocity for the given phase to achieve the specified mean, thus causing the phase-fraction to adjust according to the mass flow rate.

Boundary conditions at the free surface

Since the simulation only includes what happens below the water surface, and the main focus of the experiments is the investigation of the internal wave it, has been rather tricky to decide boundaries for the free surface. The free surface waves that is generated should bee of very small wave length and height due to low Froude numbers (2.6), as $Fr \leq 0.09$. An approach has been to apply a Neumann boundary condition of zero gradient at the top. With a zero gradient at the top, the bow of the barge would have been well approximated as the fluid hits the bow and it would allow outflow. At the stern however, the lower pressure would allow inflow. This would have been a good approximation, if the inflow had been consisting of air. The problem is that the inflow at the stern consists of water, causing the interface at the stern to mix and loosing of the stratified water.

Another approach as been to use a slip condition, which is a mix of Dirichlet and Neumann condition. The flow would then be allowed to flow freely in x- and y direction, but be set at z-direction. This would not allow for in- and outflow.

With the low Froude number and the goal of the invistigation beeing the internal wave, this could have been a good approximation. The *slip* condition caused an numerical artifact at at a point of the bow of the barge where another boundary condition of symmetry was set. The *symmetryPlane* condition at the front of the domain allows outflow, causing a one cell intersecting both boundary conditions having very large velocity.

A comprimise has been done, with the top of the domain beeing split into two patches, namely *atmosphere* and *atmosphereFrontOfBarge*. The *atmosphere* patch having a *slip* boundary condition and the *atmosphereFrontOfBarge* having an *inletOutlet* condition. The *inletOutlet* condition is working as a Neumann condition, with the specification of inflow, in case there is any.

These boundary conditions has a very engineering approach to them, and is not very physical. But as stated earlier in this repert, the main focus is the effect of the internal wave. With the air excluded completely from the simulations, and the low Froude numbers, the final boundary conditions of the free surface is sufficient with the goal of the experiment.

4.3.2 alpha.saltWater

The phase fraction *alpha.saltWater* is set in the file *setFieldsDict* where the initial state of the fluid is set.

As for the boundaries, a fixed value of either 1 or 0 is set, where 1 is salt water and 0 is fresh water, is set at the inlet. At the outlet, a *variableHightFlowRate* condition is used. It is a phase fraction condition based upon the flow conditions. Values of *alpha.water* is constrained to lay between specified values of upper and lower bounds of 1 and 0, i.e.

- If *alpha.water* > 1:
  - apply a fixed value, with a uniform level 1.
- If $0 \leq$ *alpha.saltWater* ≤ 1:
  - apply a *zeroGradient* condition.
- If *alpha.water* < 0:
  - apply a fixed value, with a uniform level 0.

At the *barge*, a Neumann condition of *zeroGradient* is applied. The same goes for *atmosphere* and *atmosphereFrontOfBarge*.

4.3.3 p_rgh

For pressure, a *fixedFluxPressure* is used at the inlet and on the barge. The pressure is not known at inlet. The *fixedFluxPressure* is a Neumann condition that is accounting for the flux specified by the velocity set at the boundary.

For the barge, the velocity is set to 0 by the *noSlip* boundary condition, so the Neumann condition becomes a *zeroGradient* in reality.

At the athmospere and atmosphereFrontOfBarge patches there is Dirichlet boundary of *fixedValue* 0. At the outlet there is a Neumann boundary condition of *zeroGradient*.

4.3.4 Turbulence properties k, nut, omega and epsilon

All of the turbulence properties nut, k, omega and epsilon are set with *zeroGradient* at the top patches *atmosphere* and *atmosphereFrontOfBarge*.

At the inlet, Dirichlet boundaries are set as a fixed value, calculated from equations for setting farfield turbulence conditions.

Farfield turbulent kinetic energy is set by using:

$$k = \frac{3}{2}(UI)^2, \tag{4.1}$$

where U is the freestrem speed and I is the turbulence intensity, usually set below 1% for cases similar to this experiment.

Farfield omega conditions is set by using:

$$\omega = \frac{\sqrt{k}}{C_\mu^{1/4} l_t} \tag{4.2}$$

where $C_\mu = 0.09$ and l_t is the turbulent length scale, set to $l/100$ where l is the length of the barge.

Turbulent dissipation rate epsilon is set by using:

$$\epsilon = \frac{c_\mu^{3/4} k^{3/2}}{l_t} \tag{4.3}$$

Finally, the turbulent viscosity is set by using:

$$\nu_t = \frac{k}{\omega} \tag{4.4}$$

Wall functions

Wall funtions are used to avoid resolving all scales in the boundary layer. The wall functions use the dimensional analysis of the boundary layer and law of the wall to estimate values near the wall.

At the barge, wall functions are applied as boundary conditions for all the turbulence properties.

Turbulent viscosity ν_t is using the wall function *nutUSpaldingWallFunction*. The function is using a special curve fit of the law of the wall, in order to be applicable in the whole boundary layer [16].

For turbulent kinetic energy k, the wall function *kqrWallFunction* has been used. It uses the log-law layer (2.16) to estimate values for the boundary. Using this wallFunction would require a distance of the first cell adjecent to the barge to be $y^+ > 30$. This has not been fulfilled with the meshes used in this report, but brief test runs with a wall function applicable for smaller values of y^+, has been conducted after the mistake was discovered. *kLowReWallFunction* did not make any signifacant effects on the calculated drag in these test runs.

For omega and epsilon, the wall functions *omegaWallFunction* and *epsilonWallFunction* has been used. Both wall functions are applicable for a wide range og y^+ values.

4.4 Mesh and mesh convergence

The outcome of a numerical experiment is highly dependent on the mesh quality. A multiphase simulation investigating drag and internal waves needs a mesh suitable to capture the phenomenon. At the same time the mesh has to account for computational cost, as the computational resources available is rather limited for the scope of this thesis.

4.4.1 Meshing procedure

The background mesh, or rather the base mesh is made by defining a *blockMeshDict* file that generates a mesh that is refined in the area of the pycnocline shown in figure 4.3. The base mesh is made to be uniform in the spatial x- and y-direction, except for z-direction, where it is refined to better capture the pycnocline in the entire computational domain. Number of cells in the base mesh is 12668. In order to increase mesh quality, refinement is needed.

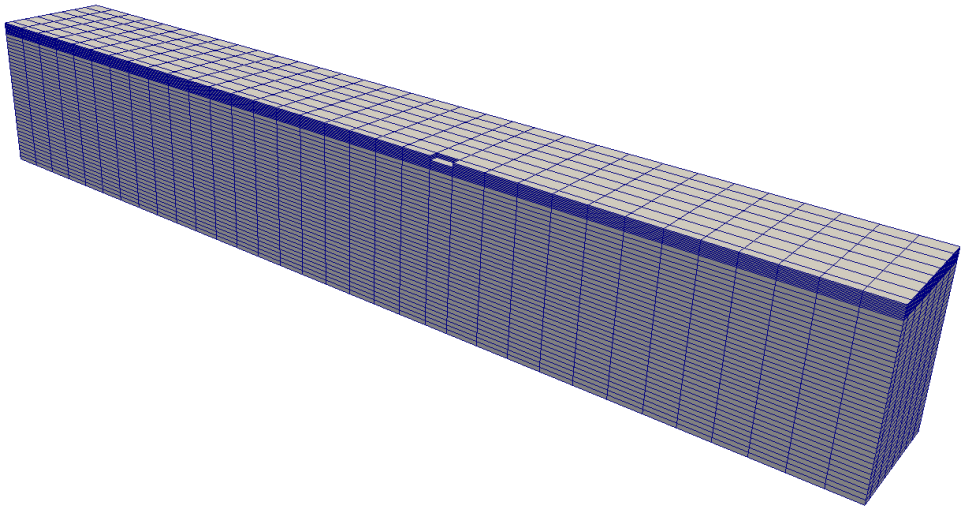


Figure 4.3: Base mesh generated by blockMesh
The base mesh made in blockMeshDict and generated by blockMesh consists of 12668 *cells. It has uniform spatially distribution in x- and y- direction, while z-direction is refined in the area where pycnocline is located.*

To refine the mesh further, *topoSetDict* and *refineMeshDict* files has been used. In a *topoSetDict*, an area within the mesh is defined which where it is desirable to refine the mesh. The *refineMeshDict* refines the mesh in the specified area in the desired direction. *RefineMeshDict* refines the mesh by splitting every cell within the specified area and in the specified direction.

By using *topoSetDict* and *refineMeshDict* it is easy to refine the mesh in the important areas within the domain i.e. boundary layer and below the barge at the pycnocline. It is further a useful tool in the means of maintaining cell aspect ratios, which is the ratio of the longest to the shortest side of the cell. Cell aspect ratio can have significant effect on waves in openFOAM [19]. The areas of the domain which is almost un disturbed can be un-refined, making the mesh less computational demanding.

Figure 4.4 is showing the mesh after refinenemt which consists of 744794 cells.

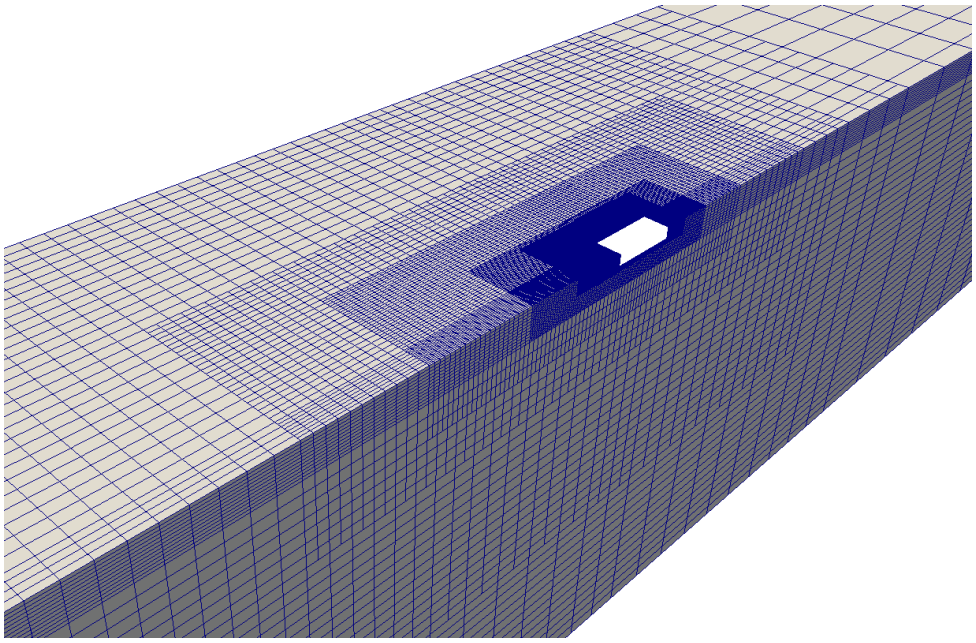


Figure 4.4: Mesh after refinement
The mesh after beeing refined consists of 744794 *cells. Max aspect ratio* = 32.7

4.4.2 Convergence tests

Three mesh grids has been systematically refined to evaluate grid convergence. It is done by using a grid refinement ratio of 2 on the base mesh. The resulting course, medium and fine grids have 4.2×10^5, 7.4×10^5 and 1.3×10^6 cells respectively. The reason for not having a refinement ratio of 2 on the resulting grids is that the same *topoSetDict* and *refineMeshDict* files are used. The resulting grids does not necessarily get the same refinement ratio, as the refinement procedure using *topoSetDict* and *refineMeshDict* depends on the base mesh.

Two speeds at two different pycnocline depths are used to check for convergence, resulting in four tests. For each pycnocline depth, two velocities gives densimetric Froude number corresponding to near-peak drag coefficient and densimetric Froude number close to super critical.

Figure 4.5 and 4.6 are showing test runs with pycnocline located at 0.1m ($h/D = 1$) below the barge and with $Fr_h = 0.86$ and $Fr_h = 1.35$ respectively. Turbulence model used in the tests are the k-ω SST. The tests show that there is not much difference between the medium and fine grids, while the corse grid is oscillating around a value of C_d i.e. it has not converged.

Table 4.2 shows the avarage y^+ values obtained from the convergence tests. The y^+ values lies all within the buffer layer, suggesting that the Wall functions must be applicable for a wide range of y^+ values.

Mesh \ Fr_h	0.86	1.35
Coarse	11.14	13.10
Medium	8.39	11.33
Fine	6.8	8.95

Table 4.2: Average y^+ values for pycnocline at 0.1m
Average y^+ value obtained from running convergence tests.
All y^+ values lies within the buffer layer, with the finest mesh tending towards viscous sub-layer

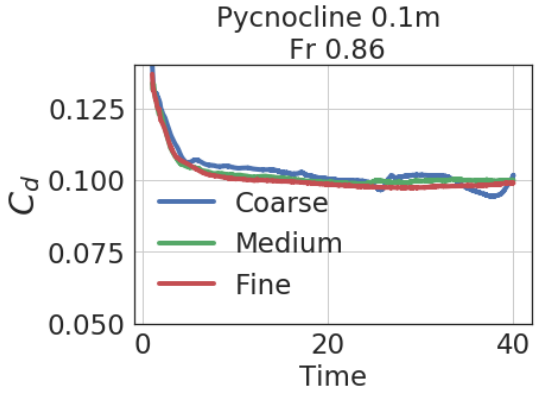


Figure 4.5: C_d as a function of time.
Grid convergence test run with three different grids; coarse, medium and fine. Pycnocline located at 0.2 m below the barge, and $Fr_h = 0.61$. The medium and fine grids are converging towards the same drag coefficient, while the coarse grid is occilating around the medium and fine grids

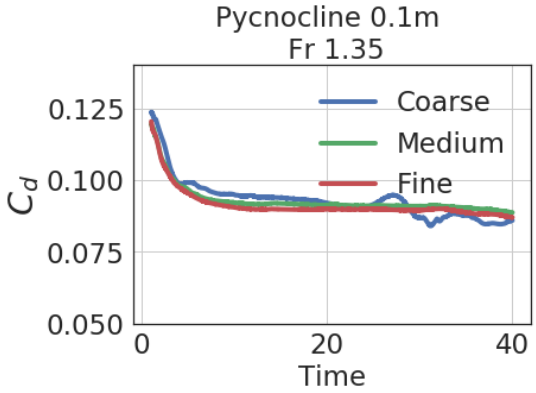


Figure 4.6: C_d as a function of time.
Grid convergence test run with three different grids; coarse, medium and fine. Pycnocline located at 0.2 m below the barge, and $Fr_h = 0.61$. The medium and fine grids are converging towards the same drag coefficient, while the coarse grid is occilating around the medium and fine grids

Figure 4.7 and 4.8 are showing test runs with pycnocline located at 0.2m ($h/D = 2$) below the barge and with $Fr_h = 0.61$ and $Fr_h = 0.95$ respectively. Turbulence model used in the tests are the k-ω SST. The tests show the same trends as the tests done with pycnocline located 0.1m below the barge.

Table 4.3 shows the avarage y^+ values obtained from the convergence tests with pycnocline located at 0.2m below the barge. The y^+ values lies all within the buffer layer, suggesting that the Wall functions must estimate the whole turbulent boundary region

Mesh \ Fr_h	0.61	0.95
Coarse	11.89	14.50
Medium	9.74	11.53
Fine	7.73	9.51

Table 4.3: Average y^+ values for pycnocline at 0.2m
Average Y^+ value obtained from running convergence tests.
All Y^+ values lies within the buffer layer, with the finest mesh tending towards viscous sub-layer

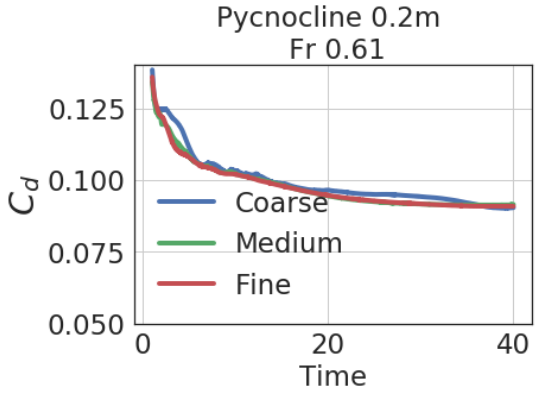


Figure 4.7: C_d as a function of time.
Grid convergence test run with three different grids; coarse, medium and fine. Pycnocline located at 0.2 m below the barge, and $Fr_h = 0.61$. The medium and fine grids are converging towards the same drag coefficient, while the coarse grid is occilating around the medium and fine grids

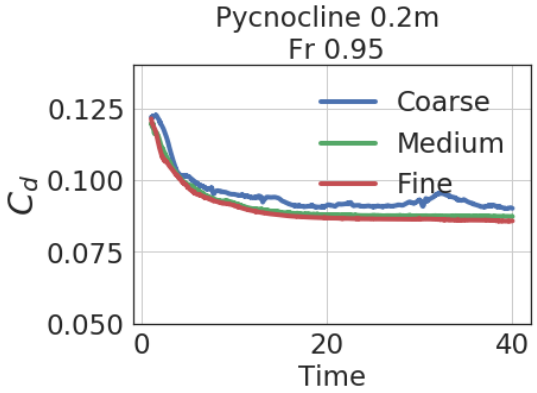


Figure 4.8: C_d as a function of time.
Grid convergence test run with three different grids; coarse, medium and fine. Pycnocline located at 0.2 m below the barge, and $Fr_h = 0.61$. The medium and fine grids are converging towards the same drag coefficient, while the coarse grid is occilating around the medium and fine grids

From the convergence tests it is clear that it is not much to gain convergence wise by using a finer mesh. The resulting C_d's are very similar comparing the medium and fine grids. By using the medium mesh, computational time is saved and many more simulations can be done within the time frame of this thesis.

Chapter 5

Results

5.1 Comparison of turbulence models

Comparison of the turbulence models was done by running numerical experiments with the exact same mesh. The tests was done with a pycnocline depth of 0.2m below the barge. Two cases are presented, the first with a densimetric Froude number $Fr_h = 0.69$ and the second with $Fr_h = 0.95$. The first test lies whithin the range where Fr_h should give peak C_d and the second test where Fr_h should give a lower C_d that tends towards a non-stratified fluid case.

The comparison is done by checking convergence of the C_d for the two cases. The motivation for comparing the C_d and not any other parameters is that of the main scope of this project, investigating the increase in C_d in the range of $0.6 \leq Fr_h \leq 0.8$ in stratified fluids. It is essential to have a turbulence model that best approximate the C_d

The first test is shown in figure 5.1. From the figure it is shown that the turbulence model k-ω SST gives a good convergence of the C_d. The model k-ϵ on the other hand sees an increace of C_d throughout the entire expriment and gets even more unstable as time increases.

THe second test is shown in figure 5.2, and is showing the same trend as seen in the first test. K-ω SST gives a good convergence, while k-ϵ gets an increase in C_d as time increases. In the second test, k-ϵ even starts to oscillate as the time increases.

From the tests it is concluded that k-ω SST model is the most reliable investigating the "dead water" phenomenon going further.

Even though the theory states that the k-ω turbulence model yields better results when simulating adverse pressure gradients and boundary layer [8], it serves a purpose to compare with other models. As stated in theory, the k-ϵ model can get reliable results when simulating multiphase fluids. The comparison of the models only confirms the theory that the k-ω SST model is superior in estimating boundary layers and adverse pressure gradients.

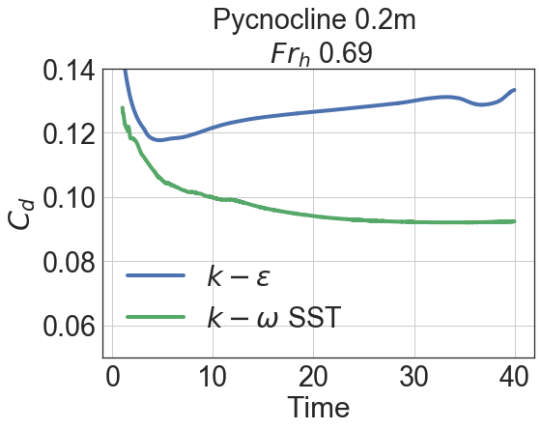


Figure 5.1: C_d as a function of time. *Simulations done with a pycnocline at* $0.2m$ *and* $Fr_h = 0.69$. $k-\epsilon$ *Turbulence model yields a higher* C_d *than the* $k-\omega$ *SST model. While the* $k-\omega$ *SST model converges,* $k-\epsilon$ *model has a steady growth of* C_d *and gets more unstable as time increases*

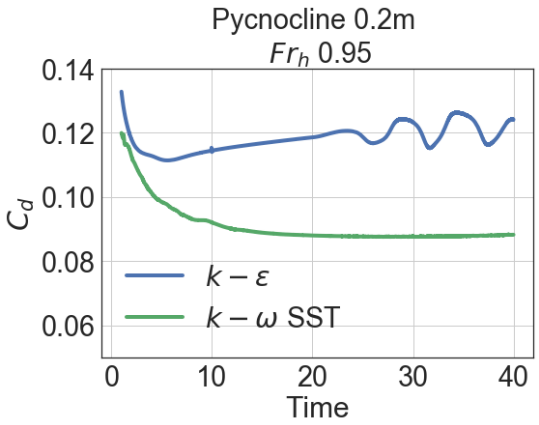


Figure 5.2: C_d as a function of time. *Simulations done with a pycnocline at* $0.2m$ *and* $Fr_h = 0.95$. $k-\epsilon$ *Turbulence model yields a higher* C_d *than the* $k-\omega$ *SST model. While the* $k-\omega$ *SST model converges,* $k-\epsilon$ *model has a steady growth of* C_d *and starts to oscillate as time increases*

5.2 internal wave

5.3 Drag

At critical densimetric froude numbers, the drag increases. The reason for the increase in drag is because we get a crest of the internal wave at the stern of the barge. The crest restricts the passage area and accelerates the flow. This in turn results in a lower pressure in the stern of the ship. We get both a higher friction drag, and the lower pressure for the stratified case increases the pressure drag. In total we get an increase in the stratified case.

Key points to include:

- crest at stern of barge
- increase of velocity because of restriction of passage area
- results in thinner boundary layer, thus hicgher friction.
- also results in decrease of the pressure, such that total drag coefficient increses

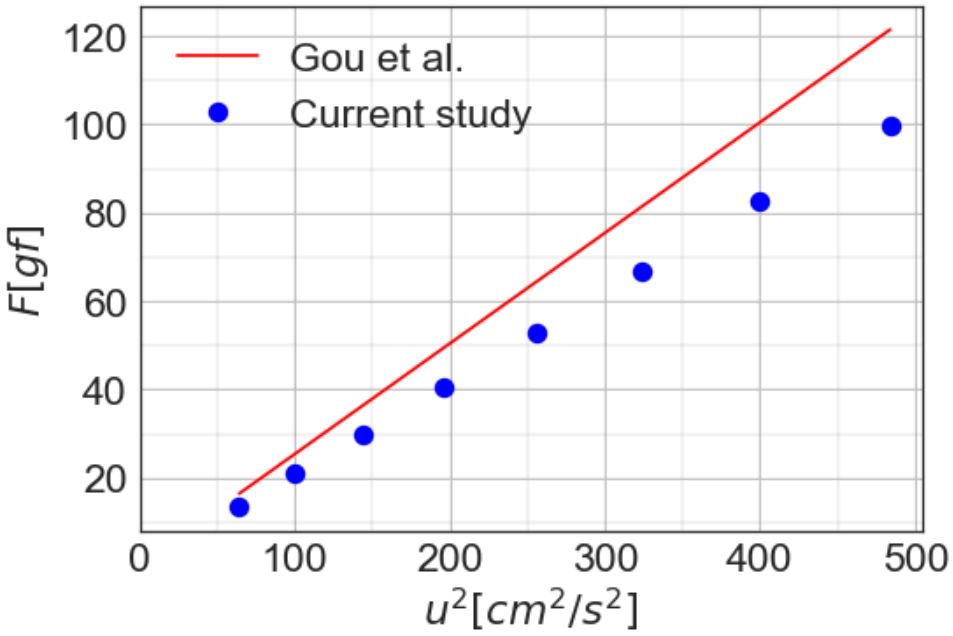


Figure 5.3: Drag force as function of velocity squared.
Resistance in homogeneous fluid with barge draft $= 0.1$. *Simulations is in good agreement with experimental data obtained on an identical barge done by Gou et. al. [1] with a deviance of* ≈ 20 *%. Resistance is directly proportianal to the square of the towing speed, which means that the drag coefficient is constant.*

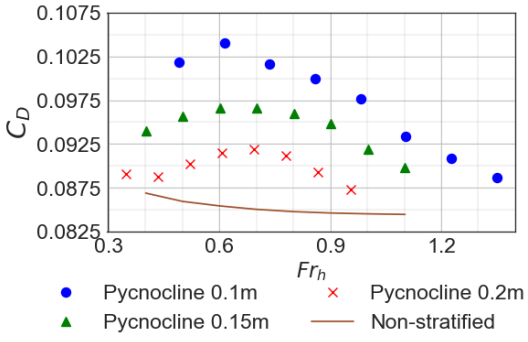


Figure 5.4: Drag coefficient as function of densimetric Froude number Fr_h
Drag coefficients (C_d) *obtained by running simulations with* $h/D = 1$, 1.5 *and* 2. C_d *peaks within a range of* $Fr_h = 0.6$ *and* 0.8. C_d *tends towards simulations of non-stratified fluid for all pycnocline locations when* $Fr_h \leq 0.6$ *and* $Fr_h \geq 0.8$.

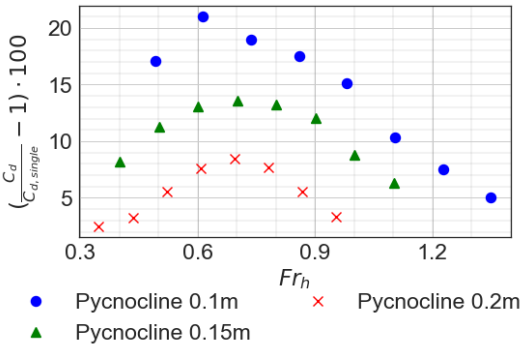


Figure 5.5: Drag coefficient as function of densimetric Froude number.
Drag coefficients (C_d) *obtained by running simulations with different interface height.* C_d *peaks within a range of* $Fr_h = 0.6$ *and* 0.8. *For all interface hights there is the same trend that* C_d *tends towards simulations of homogeneous fliud (single phase) as densimetric Froude number* $Fr_h \leq 0.6$ *and* $Fr_h \geq 0.8$.

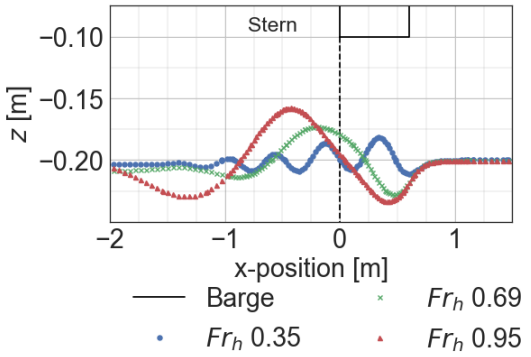


Figure 5.6: Pycnocline as a function of x-position.
Pycnocline below the barge at the symmetry plane as a function of x-position. Initially $h/D = 2$. As Fr_h increases, wave amplitude increase and the crest located further back. Location of crest has significant effect on flow field as shown in figure 5.7. The internal wave restricts the passage area, causing a significant increase in speed. This effects on C_d are shown in figure 5.4.

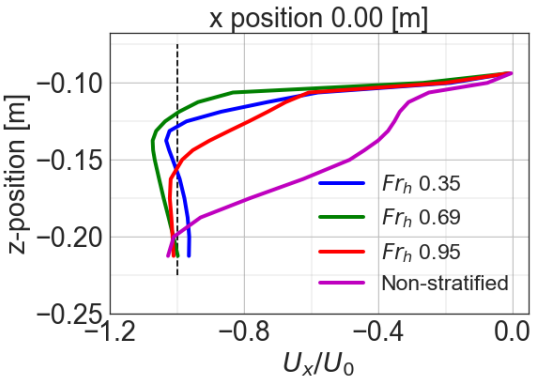


Figure 5.7: velocity profiles U_x/U_0 as function of z-position.
Dimensionlesss velocity profiles U_x/U_0 below barge at $x = 0.0m$ with pycnocline at $0.2m$. The velocity profile with $Fr_h = 0.69$ has a signifiant increase in speed and greater gradient.

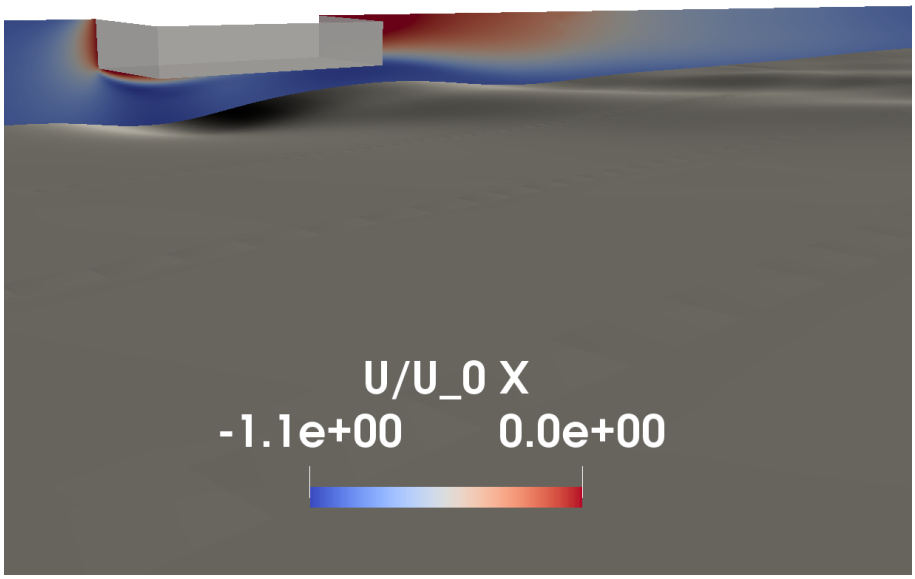


Figure 5.8:

Chapter 6

Conclusions and further recommendations

Bibliography

[1] Y. Guo, W. Xu, X. Xinwei, and B. Teng,
Experiment study on the towing resistance of a barge in a two-layer fluid.
In The 32nd International Workshop on Water Waves and Floating Bodies, 2017.

[2] C.D. Argyropoulos, N.C. Markatos, (2015). Recent advances on the numerical modelling of turbulent flows, Applied Mathematical Modelling
https://www.sciencedirect.com/science/article/pii/S0307904X14003448#bi0005

[3] F.M. White, Viscous Fluid FLow
(second ed.), McGraw Hill, New York (1991)

[4] F. R. Menter
Review of the shear-stress transport turbulence model experience from an industrial perspective,
International Journal of Computational Fluid Dynamics (2009)
https://www.tandfonline.com/doi/full/10.1080/10618560902773387?scroll=top&needAccess=true#_i3

[5] Turbulence Modeling Resource
https://turbmodels.larc.nasa.gov/sst.html

[6] Frans T.M. Nieuwstadt, Bendiks J. Boersma, Jerry Westerweel, Turbulence, Introduction to Theory and Applications of Turbulent Flows
Springer, Delft (2015)

[7] L. C. Berselli T. Iliescu W. J. Layton
Mathematics of Large Eddy Simulation of Turbulent Flows
Springer, Pisa, Blacksburgh and Pittsburgh (2005)

[8] H. K. Versteeg, W. Malalasekera, An introduction to Computational Fluid Dynamics, The Finite Volume method
(second edition) , Pearson, Loughborough (2007)

[9] D. Wilcox
Turbulence Modelling for CFD
(third ed.), DCW Industries, Inc (2006)

[10] S. P. Pope.
Turbulent Flows.
Cambridge University Press, 2000.

[11] P. Durbin, B. A. Petterson Reif
Statistical Theory and Modeling for Turbulent Flows
(second ed.), Wiley & Sons, West Sussex (2011)

[12] By Kai Bao, Xiaolong Wu, Hui Zhang and Enhua Wu
Volume fraction based miscible and immiscible fluid animation
Comp. Anim. Virtual Worlds (2010) http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.459.7375&rep=rep1&type=pdf

[13] Gregor Černe, Stojan Petelin † and Iztok Tiselj
Coupling of the Interface Tracking and the Two-Fluid Models for the Simulation of Incompressible Two-Phase Flow
Journal of Computational Physics 171, 776–804 (2001) https://ac.els-cdn.com/S002199910196810X/1-s2.0-S002199910196810X-main.pdf?_tid=30dd869d-b73f-4eed-ab23-6157367259eb&acdnat=1523609023_364c3bac9fcaf02d04d3586a5b9a78c4

[14] International Towing Tank Conference,
Recommended Procedures and Guidelines, Practical Guidelines for Ship CFD Applications https://ittc.info/media/1357/75-03-02-03.pdf

[15] Fadl Moukalled, Marwan Darwish, and Luca Mangani.
The Finite Volume Method in Computational Fluid Dynamics An Advanced Introduction with Open-FOAM(R) and Matlab
volume 113 of Fluid Mechanics and its applications. Springer-Verlag, 2015.

[16] D. B. Spalding,
A Single Formula for the "Law of the Wall",
Journal of Applied Mechanics, (1961).

[17] Rok Krpan and Boštjan Končar
Simulation of Turbulent Wake at Mixing of Two Confined Horizontal Flows Science and Technology of Nuclear Installations,Volume 2018 "Jožef Stefan" Institute, Ljubljana, Slovenia https://www.hindawi.com/journals/stni/2018/5240361/

[18] Menter, F. R.
Two-Equation Eddy-Viscosity Turbulence Models for Engineering Applications
AIAA Journal, Vol. 32, No. 8, August 1994, pp. 1598-1605.

[19] Johan Roenby, Bjarke Eltard Larsen, Henrik Bredmose and Hrvoje Jasak
A new volume of fluid method in OpenFOAM
VII International Conference on Computational Methods in Marine Engineering,MARINE 2017