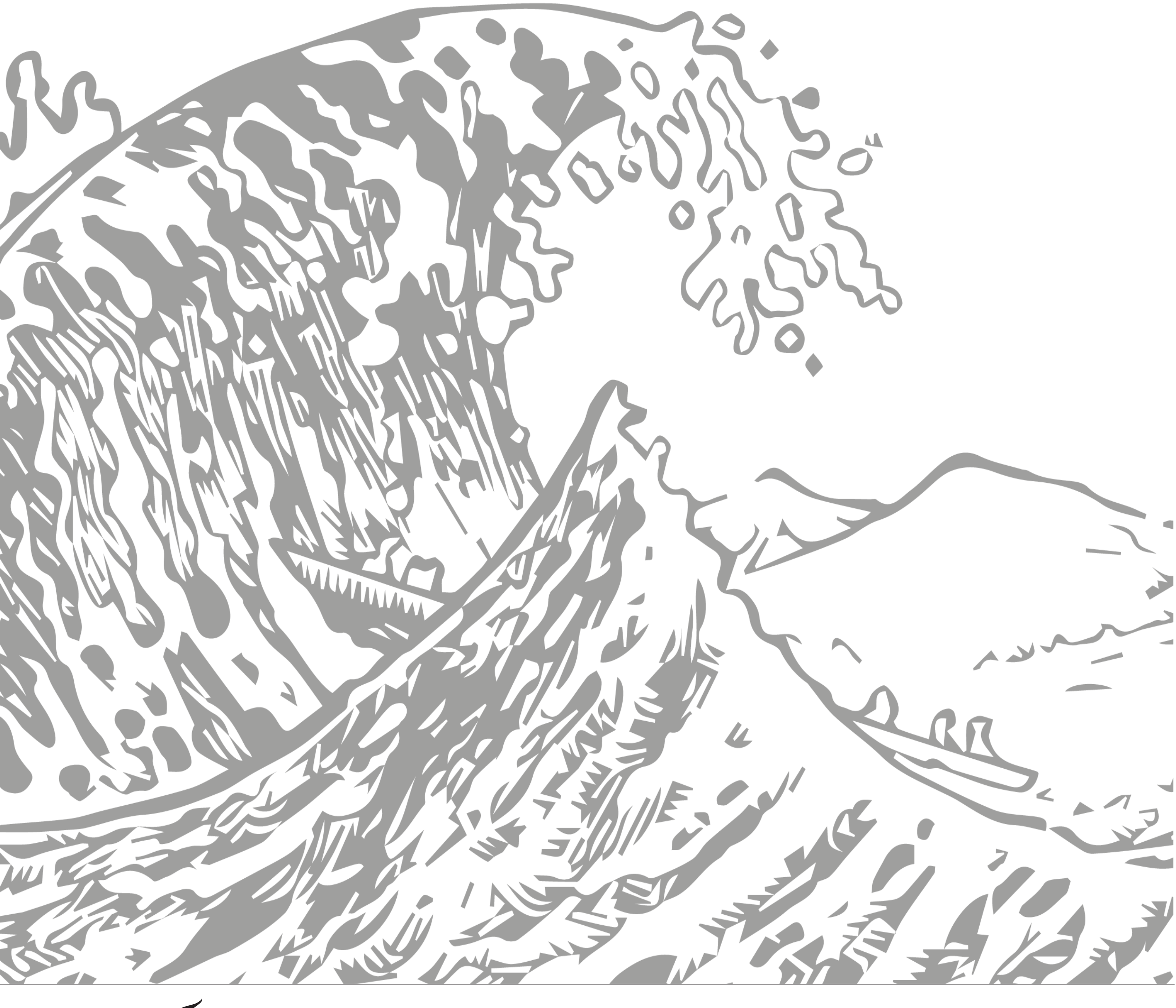
Numerical investigation of the effect of nappe non-aeration on caisson sliding force during Tsunami breakwater over-topping using OpenFOAM.

CIE 5050-09 Additional Graduation Work

Hydraulic Engineering
Environmental Fluid Mechanics

AKSHAY PATIL

October 4, 2018



Delft University of Technology

Challenge the future

Numerical investigation of the effect of nappe non-aeration on caisson sliding force during Tsunami breakwater over-topping using OpenFOAM.

CIE 5050-09 Additional Graduation Work

Akshay Patil 4655745

Project Duration : August-October 2018

Thesis committee: Prof.dr.ir. W.S.J. Uijttewaal
Associate Prof.dr.ir. J.D. Bricker

Abstract

During the 2011 Great East Japan Tsunami, the world's deepest breakwaters in Kamaishi and Ofunato, which had been designed to reduce tsunami damage, partially failed after over-topping. The failure was due to a combination of sliding, over-topping induced foundation scour and foundation bearing capacity failure. Hind casting this failure required detailed two-phase (air-water) Computational Fluid Dynamics (CFD) (OpenFOAM in this case) simulation, though uncertainty in the turbulence model used had a large effect on the simulated forces on the breakwater. This study focuses on understanding the physics necessary to correctly model the problem of breakwater over-topping by tsunami (Bricker *et al.* 2013). To correctly model the physical behavior of the system, physical model studies were carried out (Mudiyanselage 2017). However, validation of the numerical model will allow easy up-scaling of the flow physics. In order to have complete flexibility in the CFD code while at the same time have sufficient reliability, OpenFOAM will be used.

As the physical model studies have already been carried out, this study will only focus on replicating the experimental results in the numerical model. A validation procedure will study the effect of grid size (using grid convergence study) and type of simulation (2-Dimensional/3-Dimensional) used to model the experiment numerically. The final goal of the study is to understand the relationship between the overflowing discharge and the additional force generated due to non-aeration of the over-flow jet. The entrainment of air from the cavity under the over-flow jet is also of prime interest in this study.

Overall,the numerical results indicate that non-aeration could provide about 17.85% additional force leading to failure by sliding/overturning. It was observed that the forcing on the caisson is not constant but cyclic. This is a key result from this study affecting the design of the caisson breakwater. Additionally, the aeration mechanism and overflow jet breakup during the flow was also investigated. A mechanism for aeration of the cavity has been proposed as observed in the numerical model results. However, due to limited scope of the study and time constraints, further investigations with respect to the overflow discharges was not possible.

I dedicate this to my parents and my brother.

Contents

List of Figures

List of Tables

Acknowledgment

I would like to express my deepest appreciation to my supervisor, Dr. Ir. Jeremy Bricker, who has been the guiding light for me to be able to complete the additional graduation work. His constant support, encouragement, help and excellent technical advise have been invaluable. Without the positive outlook and kind words from Jeremy it would have been a steep climb to reach the end of this thesis work.

I would also like to thank Prof. Dr. Wim. Uijttewaal for his critical insights and suggestions in the analysis process. I consider myself fortunate to have you on the supervising committee. Thank you for providing the essential counsel in the final stages of the research despite the time constraints and tight schedule.

Sincere thanks to the computational facilities provided by HPC08 computing cluster. I would like to thank TU Delft for providing the required infrastructure for carrying out the research. This thesis would not be possible without the computing facilities. Additionally, I would like to thank all the people using the cluster for being patient and understanding as I was using large number of computing cores during the summer so I could finish my work on time. I extend my thanks to Sanduni for providing me all the data required for validation.

This research work would not be possible without the constant support and encouragement from family and friends. Staying away from everyone I love proved to be quite a challenge and I appreciate the faith and patience from my family and friends. I would also like to thank all my friends here in the beautiful city of Delft. Special thanks to Mruga for helping me with the layout and other typesetting related details in the report. I would also like to thank Chris for the help with the cover image and proof-reading the report.

Akshay Patil
Delft, October 2018

Chapter 1

Introduction

"Entropy is the price of structure"

- Ilya Prigogine

1.1 General area of the project

After three decades of persistent effort and nearly $1.6 billion spent, the work on the world's deepest breakwater (one of the two) in Kamaishi, was completed in 2009. During the 2011 Great East Japan Tsunami, the breakwaters in Kamaishi and Ofunato, partially failed after over-topping. Failure was due to a combination of sliding, over-topping induced foundation scour and scour bearing capacity failure. Following this disastrous event, the city of Kamaishi was left with a crumbled breakwater and defenseless against the sea. Despite the failure of the breakwater during the design event, the Japanese government drafted a plan to rebuild the now damaged Kamaishi and Ofunato breakwaters. One of the major arguments in favor of the reconstruction of the breakwater (especially Kamaishi) goes as quoted below (Onishi 2011).

> *"Defenders say that if Kamaishi's breakwater is not fixed, people and businesses will move away even faster for fear of another tsunami."*
>
> **- Norimitsu Onishi (Newyork Times)**

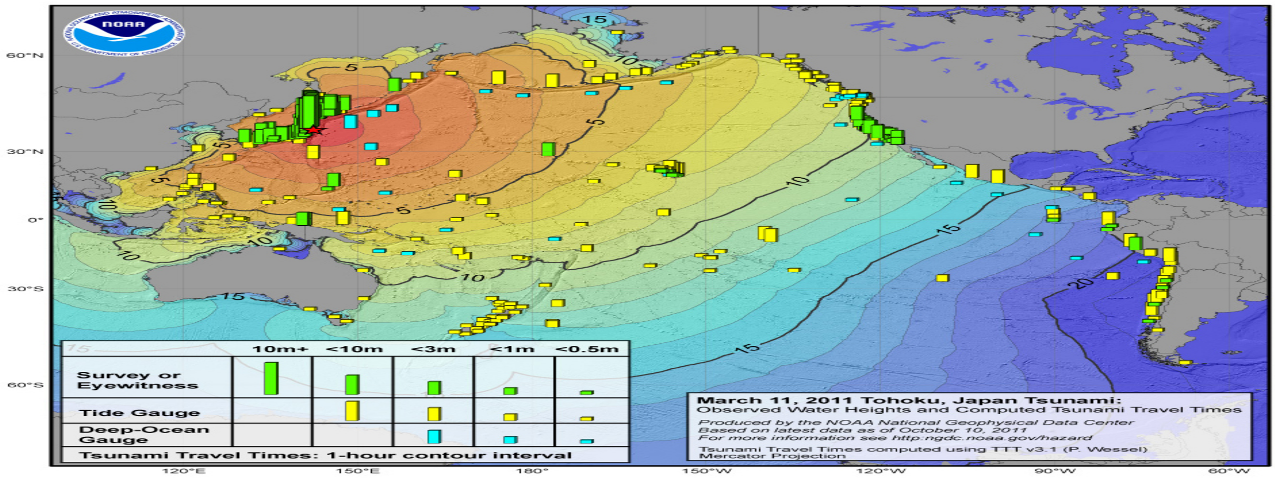

60°N
30°N
0°
30°S
60°S
120°E
150°E
180°
150°W
120°W
90°W
60°W
5
10
15
20
10m+
<10m
<3m
<1m
<0.5m
Survey or Eyewitness
Tide Gauge
Deep-Ocean Gauge
Tsunami Travel Times: 1-hour contour interval
March 11, 2011 Tohoku, Japan Tsunami:
Observed Water Heights and Computed Tsunami Travel Times
Produced by the NOAA National Geophysical Data Center
Based on latest data as of October 10, 2011
For more information see http:ngdc.noaa.gov/hazard
Tsunami Travel Times computed using TTT v3.1 (P. Wessel)
Mercator Projection

Figure 1.1: Wave height during the 2011 tsunami in Japan.(Hunter *et al.* 2012)

As a result, understanding and quantifying the effectivity of the breakwaters and the assessment of the stability during design events is befitting. The aim of this study is to investigate the influence of the overflow jet non-aeration on the overall stability of the breakwater. The strategy is to validate the experimental investigation carried out previously by S. Mudiyanselage (Mudiyanselage 2017).

1.2 Major challenges of interest and relevance of this study

The major challenges associated with this study are the qualitative assessment of the numerical model that will be employed. Additionally, a method to formalize the numerical setup using OpenFOAM will also be presented in this study. Although some studies using the physical model have been carried out (Mudiyanselage 2017), the implementation of the numerical model and its validation are yet to be successfully assessed qualitatively. The physical model studies show promising results however, an equivalent/comparable study has not been done in terms of the numerical model. As a result, validation of a numerical model is one of the major challenges.

Understanding the failure modes of a structure can help to determine the required safety measures that have to be put in place. As this study is centered around the physics of the failure mechanism (generation of negative pressure leading to failure), a better understanding of the effect of the non-aeration of the overflow jet and caisson stability can provide insights into breakwater design.

1.3 Main goal of the study

In order to carry out the validation of the numerical model, a standard tool of Computational Fluid Dynamics (CFD) named OpenFOAM will be used. OpenFOAM is an open source CFD software package which can solve the Navier-Stokes equations numerically employing the Finite Volume Method (FVM). This package has been widely used in the domain of CFD and has numerous validated applications including the current physical system under consideration (See literature Chapter 2). Appropriate implementation of the numerical model and its validation with the previous experiments (Mudiyanselage 2017) is the main aim of this study. The reliability of the numerical model can be assessed by comparison of the numerical results with the physical model results. To recapitulate, a synergy of physics of the problem, numerical modeling and data analysis will result in the assessment of the final deliverables in this study.

1.4 Layout of the report

To summarize, the layout of this report will be as described as below:

- Literature review
- Research questions, objectives and sub-goals
- Methodology
  - Review of the experimental study (Mudiyanselage 2017))
  - Brief discussion of OpenFOAM structure and numerics
- Results and Discussion
- Conclusions and recommendations
- Appendices (Include the detailed numerical setup and other details)

NOTE: The case setup and model have been discussed in the Appendices

Chapter 2

State-of-the-art/Literature Review

"Plato is my friend, Aristotle is my friend, but my greatest friend is the truth"

- Issac Newton

The CFD community is highly active in developing verified (See the article by (Roache 1997) for the definition) versions of OpenFOAM code. As a result, the verification is out of the scope of this study. The computer code, which will be employed for the assessment of the physics of the problem, will be used without having to deal with the verification of the algorithm/code. This is a justified approach as the testing and verification of the computer code/algorithm is already a software standard during its release (OpenFOAM & Community 2018). Although one does not have to verify the code, it is important to realize that appropriate solver packages from the OpenFOAM library are used. This choice can be based on the type of physics observed in the domain under investigation. For example, to study compressible flows, employing an in-compressible flow solver would give completely incorrect results.

Extensive physical model studies have been carried out (Arikawa *et al.* 2012). The study performed by T. Arikawa et. al employed a numerical model named CADMAS-SURF/3D. The results obtained in the numerical model studied by T. Arikawa et. al do not correspond one-to-one with the expected results from the physical models they performed. However, the results can provide essential insights into the modeling process. Previous studies carried out using OpenFOAM also provided an explanation for the failure mode of the breakwater (Bricker *et al.* 2013). The above mentioned studies provide a basis for carrying out the numerical investigation using OpenFOAM to verify its applicability in case of non-aeration of overflow jet.

Similar physical model studies have also been carried out by S. Mudiyanselage (Mudiyanselage 2017), which provide comparable insights into the physics of the process under investigation. The physical model from this (Mudiyanselage 2017) study will be used for the validation purposes. The motivation behind doing this is that the study was carried out at TU Delft. As a result, the experimental data is readily available in complete detail. Additionally, the numerical model set-up has already been preconditioned by the author (Mudiyanselage 2017), which makes it easy to set-up the model without any delay. However, as there have been problems with numerical simulation and validation in the previous studies (Mudiyanselage 2017) a different approach will be used in the current study.

To abridge the above the discussion, it is clear that even with a sophisticated numerical model the modeling process needs further investigation. It is safe to start with a simple model at first and progressively increase complexity in the numerical model. As a result, with motivation from the above discussed literature, it is befitting to study the current physical system using the tools mentioned previously in Chapter 1.

Chapter 3

Research question, objectives and sub-goals

"Probable impossibilities are to be preferred to improbable possibilities"

- Aristotle

The central research question in this case is the assessment for the applicability of OpenFOAM as a CFD toolbox for numerical simulation of the turbulent flow in the presence of a breakwater. In order to carry out this study a set of research questions have been formulated as described below.

- **Numerical Aspect**
    - Can OpenFOAM be used to simulate turbulent flows that occur during over-topping of a breakwater?
    - What kind of simplifications are required in order to have adequate resolution and at the same time limit the computational resources utilized?
    - Does the standard $k - \epsilon$ turbulence model perform adequately?
    - How does the force on the caisson compare in the aerated and non-aerated jet case?
- **Physics Aspect**
    - Provide an explanation of the aeration mechanism of the cavity under the nappe.
    - Provide a general relationship between the upstream caisson head (H_d) and the additional force (F_{ad}) generated due to non-aerated cavity under the nappe (See figure 3.1).

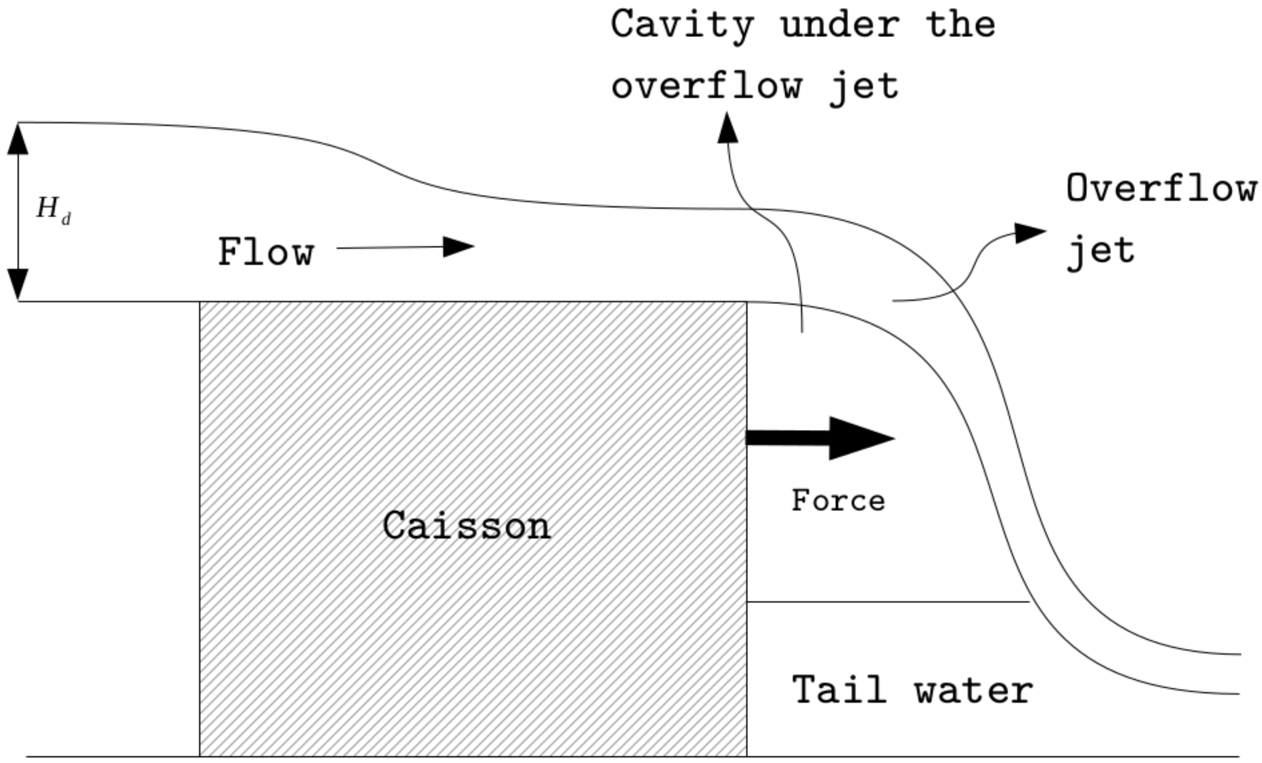


Figure 3.1: Schematic of the flow over the caisson

Assessment of the numerical model is imperative before the physics of the problem is studied. Moreover, it is a coupled process as the physics of the problem is required to adequately set-up the numerical model. The best way to deal with this dilemma is to progressively increase complexity in the model. This helps to detect any errors in the methodology employed for the set-up of the numerical model. An additional benefit of having the physical model results is that the results can be used to get a first order estimate of the expected results and the numerical parameters needed by the model. Once all the simplifications are known, they pave way for more complicated models. Using this methodology the performance of the numerical model can be assessed and tweaked as per requirement.

Chapter 4

Methodology

"Perhaps some day in the dim future it will be possible to advance the computations faster than the weather advances and at a cost less than the saving to mankind due to the information gained. But that is a dream"

- Lewis Fry Richardson

4.1 Review of the experimental study

4.1.1 Introduction

The experimental studies carried out by S. D. Mudiyanselage (Mudiyanselage 2017) will be used to validate the numerical model. Understanding the governing physics during the caisson overflow was the main objective of the study. The Reynolds (Turbulence), Froude (Flow regime) and Weber (Surface tension) scaling cannot be achieved at the same time in the scaled physical model. As a result, a valid scaling parameter has to be chosen. In this case the three parameters which were considered are as presented in Table 4.1.

Table 4.1: Dimensionless numbers governing the flow physics

Scaling Parameter	Value (Kamaishi)	Value (Scaled Model)
Reynolds Number ($Re = \frac{UL}{\nu}$)	$1.6 \cdot 10^6$	15000
Froude Number ($Fr = \frac{U}{\sqrt{g \cdot h}}$)	0.2-1.5	0.2-1.5
Weber Number ($We = \frac{\rho U^2 L}{\sigma}$)	35000	20

In table 4.1, U = Velocity [m/s], L = Length Scale [m], ν = Kinematic viscosity of water [m^2/s], g = Acceleration due to gravity [m/s^2], h = Flow depth [m], ρ = density [kg/m^3], σ = Surface Tension [N/m].

As seen in the above table, only Froude scaling could be achieved in the experimental studies. As a result, the flow quantities such as length scale and discharge will scale based on the Froude number. There is a large disparity in Reynolds numbers for the two cases. The large discharge during the tsunami over-topping of Kamaishi leads to a very high Reynolds number (high inertia force) compared to the physical model (Mudiyanselage 2017).As the transition from laminar to turbulent flow is not sharply defined, a value for $Re > 2000$ is considered to be turbulent

(Hydraulics 2018). As a result, the flow at the prototype and the scaled model are fully turbulent. The Weber number also has a large disparity in the two cases in table 4.1. However, at the prototype scale the inertial forces >> surface tension force. As a result, the effect of surface tension on the flow is negligible. In all cases the caisson behaves as a broad crested weir (Mudiyanselage 2017). This behavior, where a critical depth ($Fr \approx 1$) is observed at the upstream end and super-critical ($Fr > 1$) flow conditions at the downstream end of the caisson structure, was validated using OpenFOAM. Differential pressure sensors were used in order to measure the pressure. The pressure readings were taken every 0.001 [sec] (See figure 4.2).

For detailed methodology about the experimental setup, a reference to the report will be made for the sake of brevity (See Report (Mudiyanselage 2017)). A general schematic of the experiment can be seen in figure 4.1.

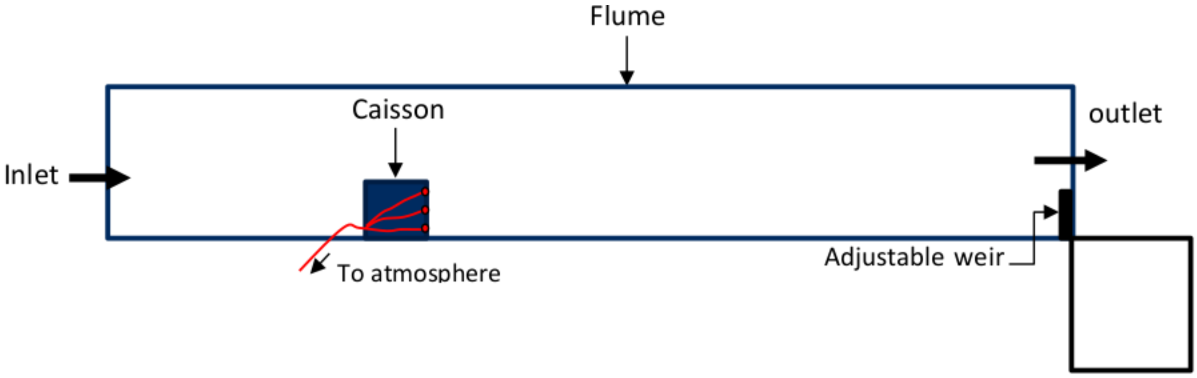


Figure 4.1: Schematic of the experimental setup (Mudiyanselage 2017). The red wires marked as "to atmosphere" are the location of the pressure sensors deployed on the front back face of the caisson.

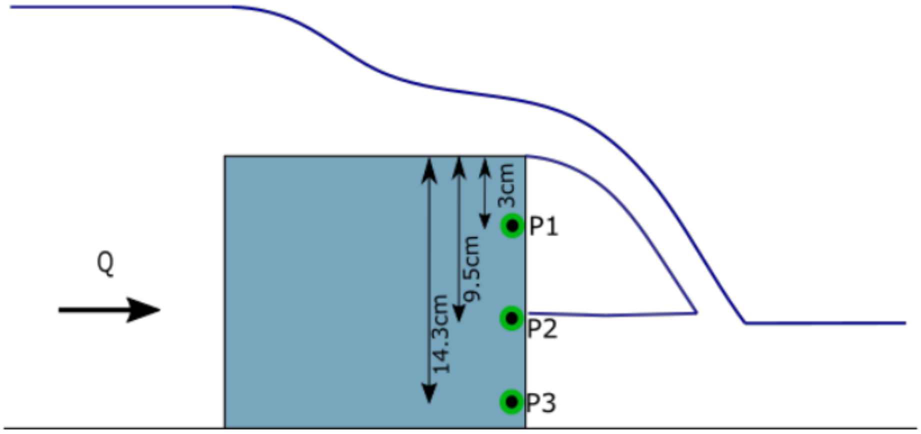
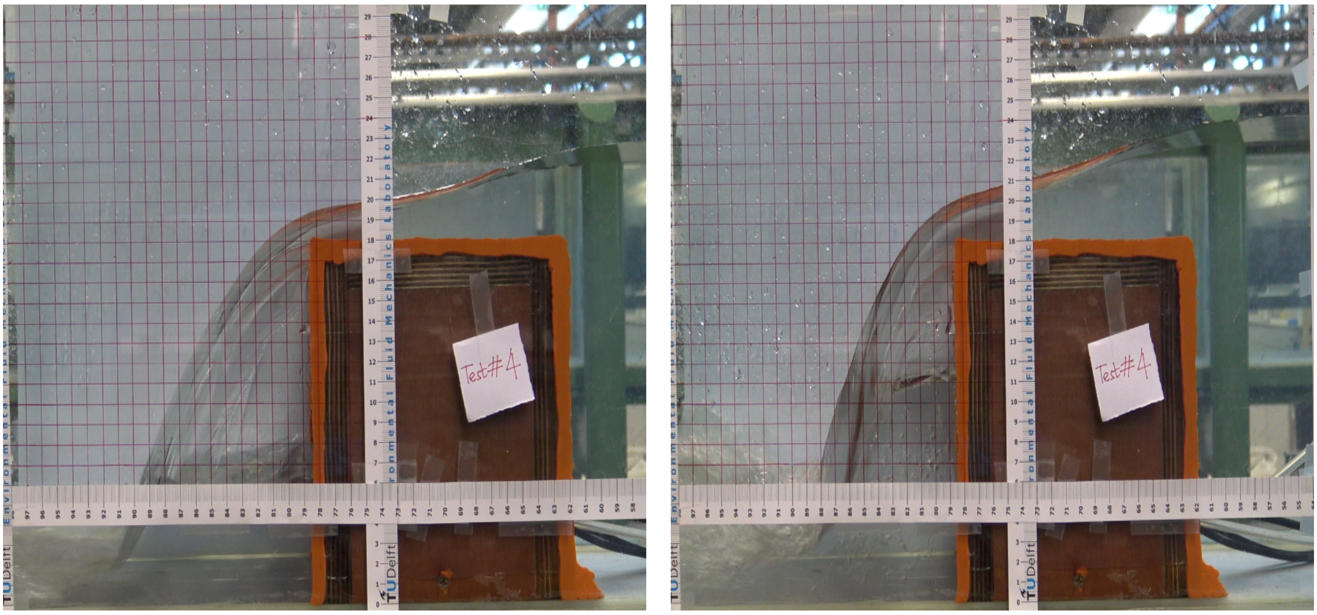


Figure 4.2: Location of the pressure sensors in the scaled model

4.1.2 Behavior of the overflow jet

As the flow over the caisson commences, a cavity below the overflow jet is formed. Initially as there is no aeration mechanism, the cavity decreases in size. The initially free flowing overflow jet transitions to a clinging jet as the flow develops in time (For details see (Mudiyanselage 2017)). The two distinct behaviors of the overflow jet due to lack of aeration can be seen in figure 4.3.


Test#4
Test#4
Environmental Fluid Mechanics Laboratory
TU Delft
Environmental Fluid Mechanics Laboratory
TU Delft

(a) Free nappe at $t = 28[s]$(Initial flow condition with sufficient aeration of the cavity under the nappe)

(b) Depressed nappe at $t = 320[s]$(Flow conditions without sufficient aeration of the cavity under the nappe)

Figure 4.3: Overflow jet behaviour. Eventually the nappe transitions into clinging nappe if sufficient aeration is not provided(Mudiyanselage 2017)

As additional force is generated due to the non-aeration of the cavity below the nappe, a detailed investigation was conducted to assess the behaviour of the overflow jet and the cavity below it. The pressure change (resulting in an additional force F_{ad} see figure 3.1) was estimated using the difference of the water level in front of and behind the overflow jet.

The air under the overflow jet is at sub-atmospheric pressure. This causes the overflow jet to move towards the caisson wall, thus pulling the jet closer to the wall. As a result, eventually the overflow jet hugs the caisson wall.

4.1.3 Validation of the ballistic model against the physical model results

The difference between the aerated and non-aerated case can be seen in figure 4.4 (Mudiyanselage 2017). It is clear from the figure that the non aerated case behaves as discussed in the previous section. Similar behaviour is also observed during the experimental studies. This data will be used to validate the numerical model to be studied in the present case.

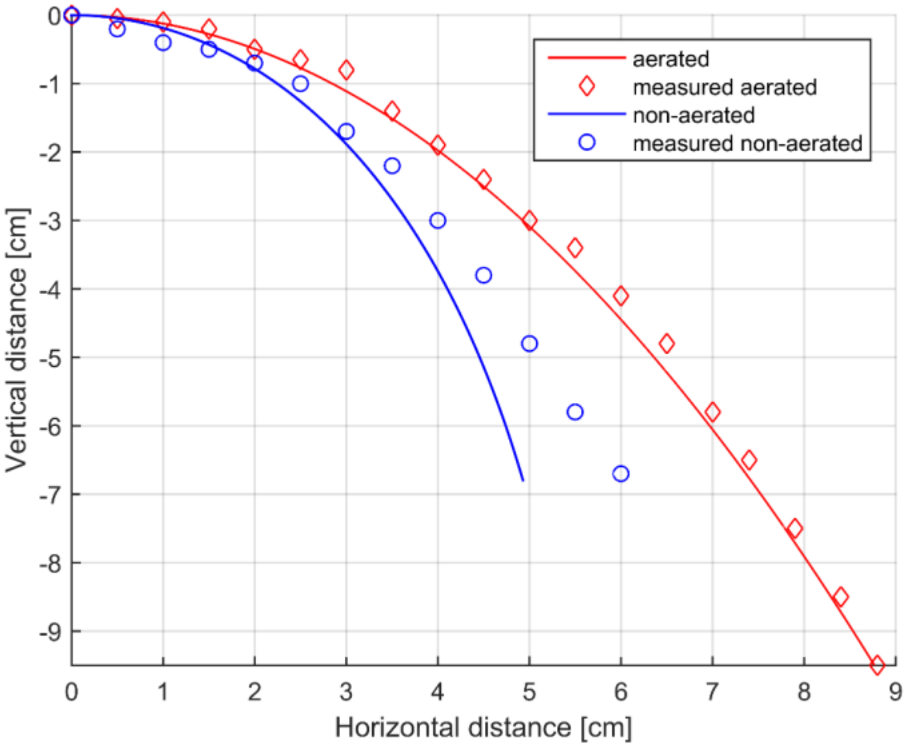


Figure 4.4: Nappe trajectory comparison (The red line represents the trajectory for aerated nappe and the blue line represents the trajectory for non-aerated nappe. The difference in the heights of the trajectory is due to the change in tail water level in the non-aerated case.)

The contribution of the dynamic pressure difference is low ($\sim 4\%$). This suggests that air entrainment due to the overflowing jet is the major mechanism which results in pressure drop and a subsequent water level rise.

4.1.4 Air entrainment mechanism

The effects of air entrainment during caisson over-topping flow have been addressed in the experimental studies. As seen in figure 4.5, the overflowing jet impinges the tail water (green circle) at a specific position. As a result of the momentum exchange/inertial force of the jet, the air from the cavity (under the lower nappe) is pushed/sucked into the tail water. As only a small part of the air entrained is transferred back into the re-circulation zone, the net effect is loss of air from the cavity (See figure 4.6). This leads to gradual reduction of air from the cavity, as a result the pressure in the cavity is sub-atmospheric. Unless there is an equilibrium between the supply and entrainment of air, the cavity eventually loses all of the air. Although the entrainment process is straight-forward to understand, the supply of air back to the cavity is a complex mechanism. The supply of air could occurs in the following ways:

- Momentary break of jet at the ends (end contraction) which draws air into the cavity
- Sufficient amount of air reaches the cavity due to highly turbulent mixing in the tail water

The process by which this equilibrium is achieved will be assessed in the numerical model. An explanation of the air-entrainment back to the cavity will be provided.

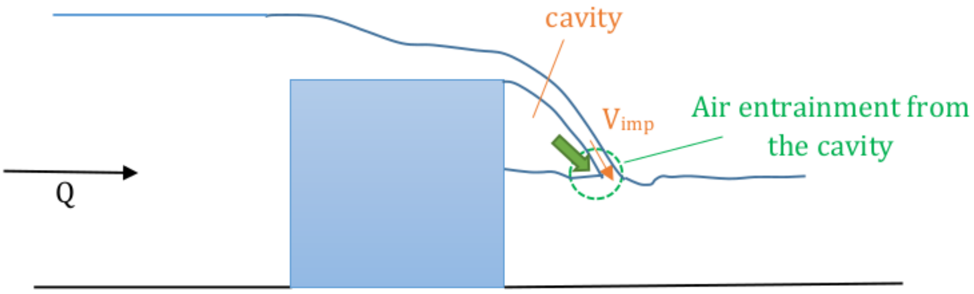


Figure 4.5: Air entrainment due to overflow jet impinging on the tail water (Air entrainment mainly occurs in the region which is encircled).

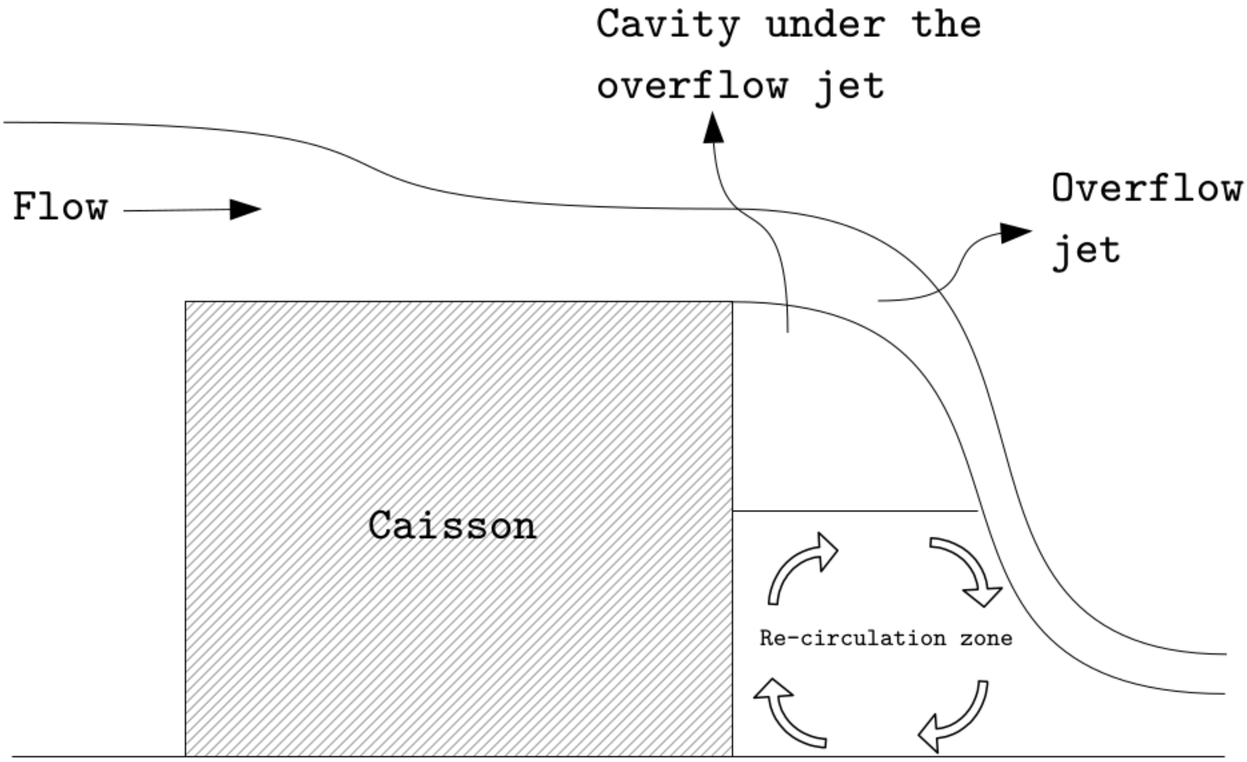


Figure 4.6: Recirculation zone under the cavity

4.2 Analysis of effective density

In the experimental investigation, the pressure sensors as seen in figure 4.2 were only functional when fully submerged. As a result, only sensor P3 was actively recording pressure data. In order to estimate the pressure inside the cavity a different approach was employed. Using the hydro-static theory, the formulation of the pressure difference ΔP was derived as given in equation 4.1.

$$\Delta P = \frac{\rho_{eff}}{\rho} \cdot P_{aerated} - P_{non_aerated} + d_{na} \cdot \rho_{eff} \cdot g \tag{4.1}$$

where,
ΔP is the pressure reduction under the nappe, ρ_{eff} is the effective density under the non-aerated nappe, $P_{aerated}$ is the reading of the pressure sensor in the aerated case, $P_{non-aerated}$ is the reading of the pressure sensor in the non-aerated case, g is gravitational acceleration and d_{na} is the difference between the tail water level across the overflow jet.

Using the ballistic model to predict the nappe trajectories results in greater deviation towards the caisson than the experimentally measured trajectory. However, using the formulation in equation 4.1 to account for change in density under the cavity results in a better agreement between the estimated and the measured trajectories (Patil *et al.* 2018). This can be seen in figure 4.7.

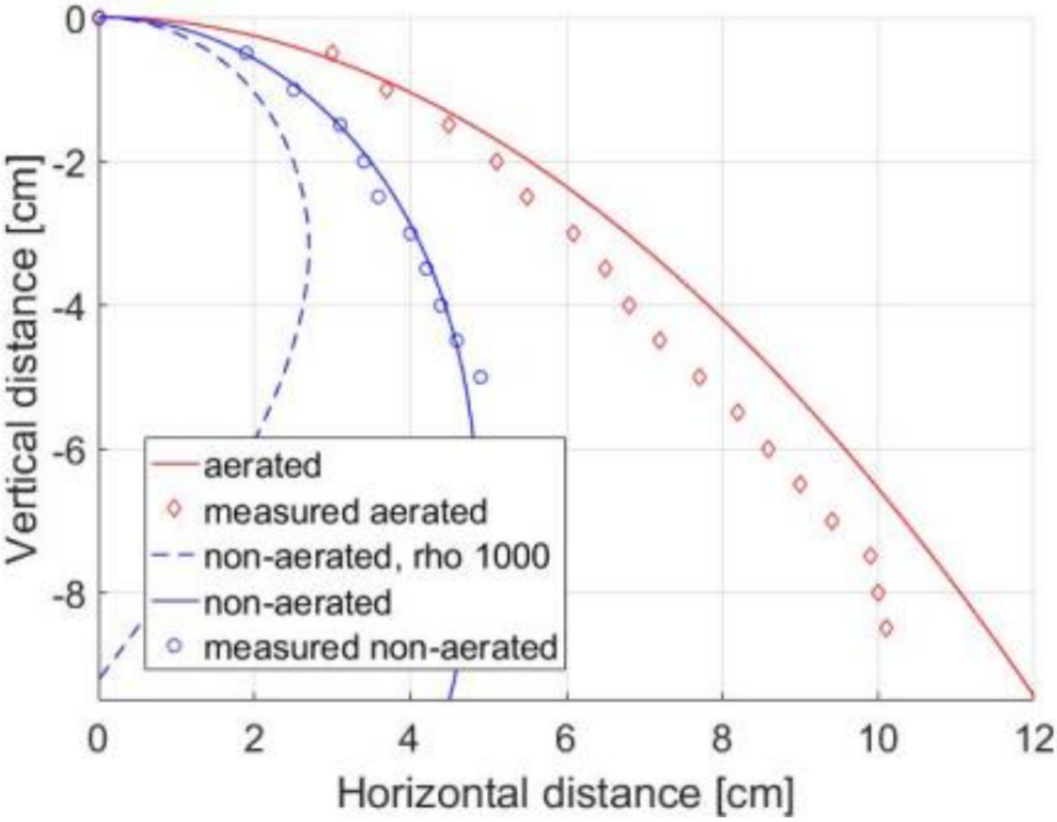


Figure 4.7: Comparison of nappe trajectories for two cases with different formulations

This explanation is feasible due to the high bubbling under the cavity in the non-aerated jet case. Using momentum conservation analysis of the incoming and the outgoing fluxes brings the theory into agreement with the experimental measurements (See figure 4.8), refer (Patil *et al.* 2018) for details.

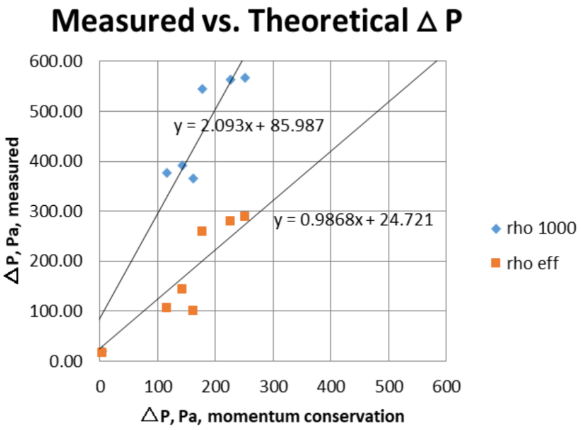


Figure 4.8: Scatter plot of the effect of effective density inclusion and exclusion (Patil *et al.* 2018)

The results propose an additional 9% to 14% increase in the sliding force due to the sub-atmospheric cavity pressure in the non-aerated case (Patil *et al.* 2018).

4.3 Governing equations

Multi-phase fluid flow is flow which consists of distinct phase or component. For brevity the fluids that will be considered in this section have a discrete/distinguishable phase(s) and are not well mixed (above the molecular level). There are a few ways mutiphase fluid flow can be studied :

- Experimental investigation (Scaled/Prototype scale study)
- Theoretical investigation (Mathematical model study)
- Numerical investigation (Discrete models study)

In general, a scale model is used to study the flow physics. This is followed by a reliable computational/numerical model to extrapolate the results to prototype scale. Consequently, analysis of the flow physics heavily relies on understanding the theoretical and numerical model reliability. Theoretically, it is possible to come up with a numerical formulation to resolve each multi-phase component separately. However, the computational/time requirement is just too large to be fulfilled. Simplified models like the *Ballistic model* (See section 4.1.3) exist, however they do not provide detailed information about other flow features explicitly. In most cases, a wide variety of fluid flow parameters are used for design purposes. This makes it essential to study numerical models to understand the scale effects and other related physics.

Consider a *Control Volume* as seen in figure 4.9. The flow across the control volume can occur in either/all directions at a given time instance. Using this definition of a control volume the differential form of the equations will be discussed in the section below.

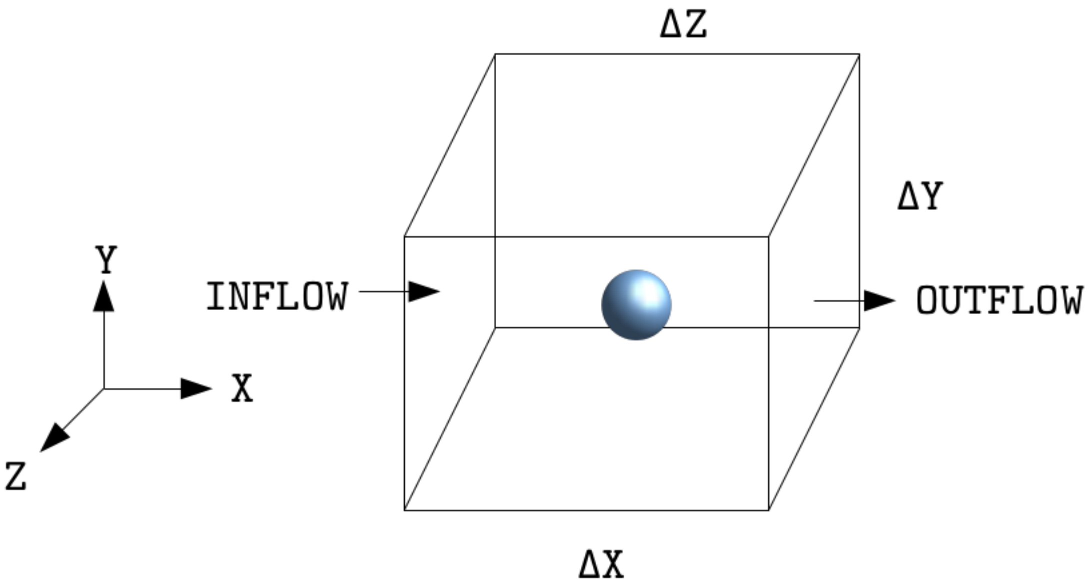


Figure 4.9: Schematic of a control volume

4.3.1 Flow equations

The conservation equations along with the Navier-Stokes equation describe the fluid flow.

Conservation of mass (Continuity equation)

The conservation of mass/continuity equation describes the change in the storage due to inflow and outflow. Equation 4.2 describes the conservation of mass in a compressible flow regime.

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{u}) = 0 \tag{4.2}$$

where,
ρ is the density of the fluid, $\nabla \cdot$ is the divergence operator and $\mathbf{u}$ is the velocity vector field.
In most cases (hydraulics of overflow jet), water is considered to be in-compressible. This makes way for a useful simplification where the first term on the left hand side of equation 4.2 can be neglected. Additionally, as $\rho = const.$ it can be pulled out of the divergence term. The resulting equation is converted from a mass conservation to a volume conservation equation.

$$\nabla \cdot (\mathbf{u}) = 0 \tag{4.3}$$

The equation above tells that the divergence of the velocity field is equal to 0 everywhere in the control volume. This means that the local volume dilation rate is zero everywhere in the control volume. A simplified way to understand this is as follows. If there is a contraction/expansion the change in the density of fluid is negligible.

However, the change in the geometry causes a change in the velocity field only.

Effects of compressibility can be estimated using the **Mach** number.

$$M = \frac{u}{c} \tag{4.4}$$

where,
u = Local flow velocity (about $\mathcal{O}(10)$ in case of over flow jet) $[m/s]$
c = Speed of sound in the medium/fluid (about $\mathcal{O}(1500)$) $[m/s]$
Using this information the **Mach** number $M = 10/1500 = 1/150 << 1$. Hence it is safe to assume in-compressible flow regime in the case of overflow jet.

Navier-Stokes equation

Unlike the continuity equation which describes the mass/volume conservation, the Navier-Stokes (N-S) equations do not describe a conservation phenomenon. They describe the motion of viscous fluid substances. The N-S equations can also be thought of as the Newton's second law ($F = ma$) equivalent for fluid flow. They describe a dissipative system. As the analytical solution for the N-S equations along with other conservation equations does not exist for non-regular 3-dimensional geometry, the only way to study the fluid physics governed by N-S equations is through numerical solutions. As we already discussed the in-compressible nature of flow, the N-S equation presented below is for in-compressible flow.

$$\frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u} \cdot \nabla)\mathbf{u} = -\frac{\nabla \mathbf{p}}{\rho} + \nu \nabla^2 \mathbf{u} + F_{sys} \tag{4.5}$$

where,
$\mathbf{u}$ = velocity field, $\mathbf{p}$ = pressure field, ρ = density (constant), $\nu = \mu/\rho$ = kinematic viscosity, $F_{sys} = Force/\rho$ = external forcing in the system and ∇ = differential operator.

It is important to note that the N-S equations can be split into the the individual Cartesian components if required. The N-S equation along with the continuity equation can be solved numerically to obtain the velocity and pressure field for a given well-posed mathematical problem. The theory above can be extended to multi-phase flows with some additional formulations. However, for sake of brevity the details will not be discussed in this study (See Tomas 2016 for details).

4.4 Numerical solution techniques

As discussed earlier, the analytical solution of the N-S equations along with the continuity equations for complex geometry and 3-dimensions is not readily available. As a result, a numerical formulation for the solution has to be thought of. Traditionally there are three ways to solve the N-S equations numerically (with computational effort required in the descending order):

- Direct Numerical Simulation
- Large Eddy Simulation
- Reynolds Averaged Navier-Stokes Turbulence Modeling

Almost every approach listed above has a trade-off somewhere in the computational modeling process. The details of the three methodologies will be discussed in the following section.

Direct numerical simulation

Analytical solution to the simplest of turbulent flow problems is not readily available. As a result, a complete solution field where the variables are known as a function of space and time can only be obtained numerically (Moin & Mahesh 1998). Most of the practical engineering problems have a wide range of turbulent scales involved. A brief overview of the range of turbulent scales can be seen in the typical energy cascade curve(Stro 2013).In order to resolve the smallest dissipation scales the numerical grid required is of $\approx \mathcal{O}(10^{-3})[m]$. Direct numerical simulation (DNS) is generally used as a research tool in most cases (Moin & Mahesh 1998). The smallest turbulent structures are known as the Kolmogorov scales defined as below.

Kolmogorov Scales (Kolmogorov 1962)

$$\eta = (\frac{\nu^3}{\epsilon})^{\frac{1}{4}} \qquad \tau = (\frac{\nu}{\epsilon})^{\frac{1}{2}}$$

here,
η is the Kolmogorov dissipation scale [m]
τ is the Kolmogorov time scale [s]
ν is the kinematic viscosity of the fluid $[m^2s^{-1}]$
ϵ is the dissipation rate of the turbulent kinetic energy $[m^2s^{-3}]$

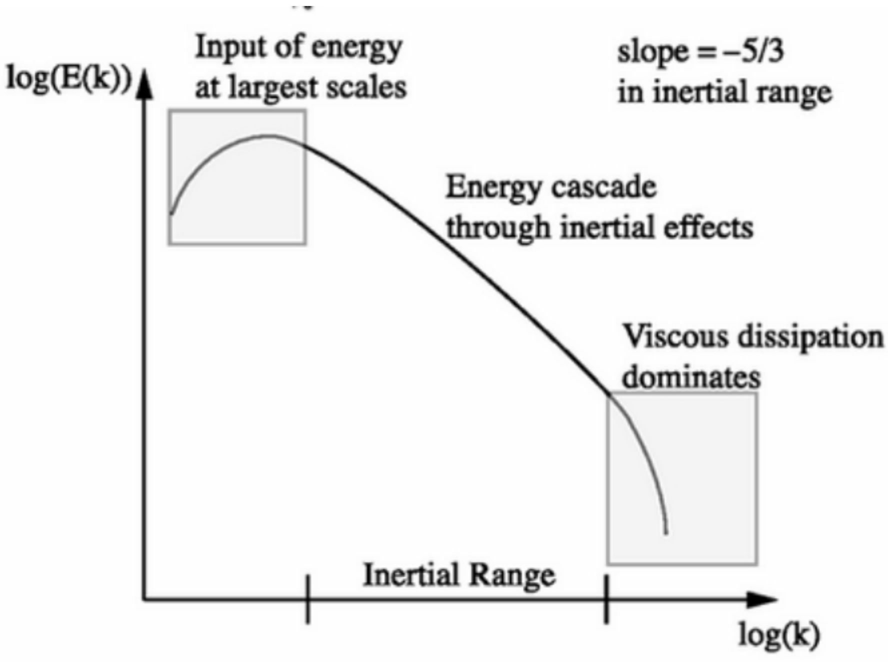


Figure 4.10: Energy cascade curve showing the wide range of scales involved in a turbulent flow (Stro 2013)

After some algebraic manipulation, the following relationship can be derived for the dissipation scales in a turbulent flow field.

$$\frac{L}{\eta} \sim Re^{3/4} \tag{4.6}$$

where,
L = Macro-scale [m]
η = Micro-scale (Kolmogorov scale) [m]
Re = Reynolds number [-]

As the current fluid flow problem under investigation is 3-dimensional, the grid size requirement would be equation 4.6 raised to the power 3.

$$N_p \sim (Re^{3/4})^3 \tag{4.7}$$

The Reynolds number in the case of overflow jet is about 15000 (See table 4.1). Using this, the number of points required for sufficient resolution of turbulent structures is $N_p = 15000^{9/4} \sim 2.5 \cdot 10^9$. To give a perspective of how big this requirement is, the computational limit of a standard 32-bit computing machine is about 2,147,483,647 $\sim 2^{31} - 1 \sim 2.147 \cdot 10^9$ (Mersenne prime). The current numerical model uses grid points $\mathcal{O}(10^6)$. Clearly, using DNS in the study would require gargantuan computational resources. As a result, this is out of scope with the current computational capabilities.

Large eddy simulation

The major hurdle in computational modeling of turbulent flow is the grid resolution for the dissipation scale turbulent structures. As a result, one way out of this problem is to try and model the dissipation scale using a different formulation and still retain the larger turbulent structures. This is possible by means of Large eddy simulation (LES) technique. This greatly reduces the grid size requirement. LES technique uses a filter function in a way where the turbulence structures (temporal as well) smaller than the grid size is modeled using a *Sub-grid Scale Model* (See Tomas 2016 for details). This methodology can be used if the large scale structures are of importance in the analysis without having to expend a fortune on the computations. It is the middle ground between DNS and the RANS technique discussed in the next section in terms of accuracy and computational requirements. Due to time limitations in the current study, employing LES type simulation is not feasible.

Reynolds averaged Navier-Stokes turbulence modeling

The most effective (computationally) way to model turbulent flows without sufficient loss in accuracy is to employ the Reynolds averaged Navier-Stokes (RANS) turbulence modeling approach. The motivation behind doing the RANS is to average out and retain the large scale turbulent dynamics of the flow. The averaging is carried out in such a way that the small scale fluctuations are modeled. There are two ways to achieve this averaging operation (Pope 2013).

- Time average (Stationary flows)
- Spatial average (Uniform flows)

However, in many cases the flows under investigation are unsteady and non-uniform. In that case *Ensemble* averaging is employed. As this is the case with the current flow under investigation, ensemble type averaging will be discussed in the section below. The ensemble average is described as per equation 4.8

$$\overline{u_i}(x_i,t) = \frac{1}{N}\sum_{n=1}^{N} u_{ni}(x_i,t) \tag{4.8}$$

where,
u_i =velocity of the i^{th} component, $x_i = i^{th}$ co-ordinate, t = time, N = number of realizations.
The above operation of averaging the flow variables is termed as *Reynolds Averaging*. Using a similar transformation the continuity equation as described in equation 4.3 can be expressed as below.

$$\nabla \cdot \overline{\mathbf{u}} = 0 \tag{4.9}$$

In order to carry out a similar operation for the momentum equation, the instantaneous velocity can be thought of as a sum of the mean and the fluctuating component. As seen in figure 4.12, the instantaneous velocity component can be decomposed using equation 4.10. However, because of the non-linear term in the momentum equation this operation is not that straight forward. This can be seen in the proof below.

$$u_i = \overline{u} + u_i^{'} \tag{4.10}$$

$$\begin{aligned}\overline{u_i u_j} &= \overline{(\overline{u_i} + u_i^{'})(\overline{u_j} + u_j^{'})}\\ &= \overline{\overline{u_i}\,\overline{u_j} + \overline{u_i}u_j^{'} + u_i^{'}\overline{u_j} + u_i^{'}u_j^{'}}\\ &= \overline{\overline{u_i}\,\overline{u_j}} + \overline{\overline{u_i}u_j^{'}} + \overline{u_i^{'}\overline{u_j}} + \overline{u_i^{'}u_j^{'}}\end{aligned}$$

If the fluctuating part is separated out from the mean part in equation 4.10, then $\overline{u_i^{'}} = \overline{u_j^{'}} = 0$. This along with the fact that $\overline{\overline{X}} = \overline{X}$ the equation can be simplified as described below.

$$\overline{u_i u_j} = \bar{u}_i\bar{u}_j + \overline{u_i^{'}u_j^{'}} \tag{4.11}$$

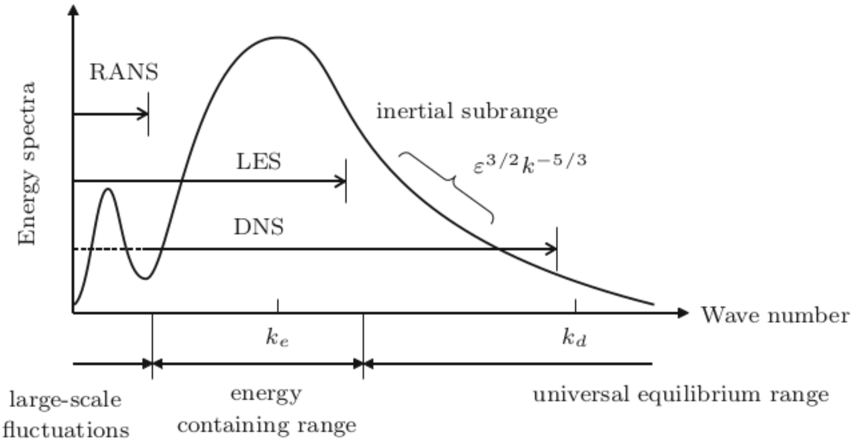


Figure 4.11: A general depiction of the range of resolution for different strategies presented for turbulence modeling (Kajishima & Taira 2017)

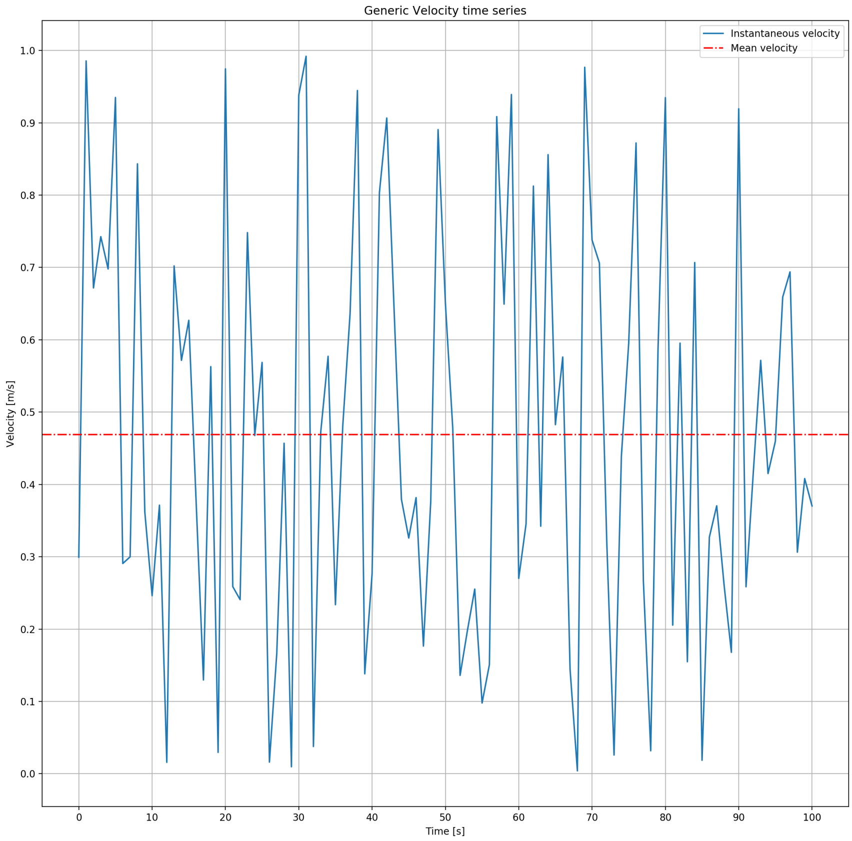


Figure 4.12: General velocity field along with the mean velocity component

Doing some algebra with the results from above and using the incompresible momentum equation yields the *Reynolds* averaged (RANS) equation for momentum.

$$\frac{1}{\rho}\left(\frac{\partial \bar{u}_i}{\partial t} + \bar{u}_j \frac{\partial \bar{u}_j}{\partial x_j}\right) = \frac{\partial}{\partial x_j}\left[-\bar{p}\delta_{ij} + \mu\left(\frac{\partial \bar{u}_i}{\partial x_j} + \frac{\partial \bar{u}_j}{\partial x_i}\right) - \rho\overline{u_i^{'}u_j^{'}}\right] \tag{4.12}$$

The three terms on the right hand side represent stresses. The first term $-\bar{p}\delta_{ij}$ represents the mean pressure field while the second term represents the viscous stresses due to molecular momentum transfer while the third term represents stress that is a result of the velocity fluctuations in the flow field (*Reynolds* stresses). This last term is the fundamental computational challenge as there are about six additional unknowns in the set of equations (due to symmetric nature of the stresses), this makes it important to model the effect of the *Reynolds* stresses in a computationally feasible manner. This is also known as finding a closure (closure problem) to the set of equations (as Number of equations < Number of unknowns).
For sake of brevity, the details of the turbulence modeling will not be discussed, however a reference to some literature which deal with the following topics in greater depth has been provided (Lesieur 2008), (Pope 2013), (Furbo 2010). This study makes use of the standard $k - \epsilon$ turbulence model.

The k-ϵ turbulence model

The following section is based mainly on the text *Turbulence: Introduction to theory and application of turbulent flows* (Nieuwstadt *et al.* 2016).
Consider the balance equation for ϵ (Dissipation rate of turbulent energy) as given below

$$\frac{D\epsilon}{Dt} \equiv \frac{\partial \epsilon}{\partial t} + \overline{u_j}\frac{\partial \epsilon}{\partial x_j} = -\frac{\partial(\overline{u_j'\epsilon'})}{\partial x_j} + P_\epsilon - D_\epsilon \tag{4.13}$$

where, P_ϵ can be thought of as the production of ϵ while D_ϵ is the destruction of ϵ. Using this formulation a turbulence model can be constructed. A length scale definition can be formulated $\mathcal{L} \sim \mathcal{U}^3/\epsilon$, which is similar to the Kolmogorov relation. The Kolmogorov relation deals with the micro-scale structure, it is easy to object the above proposed approach as we are dealing with integral turbulent length scales. However, the equation still holds as the quantity ϵ can be interpreted as the energy per unit time that is transported along the cascade (see figure 4.10) to be dissipated at the micro-scales. Turbulence in the equilibrium state would require that the dissipation rate (ϵ) be determined by the rate of kinetic energy supplied at the top of the energy cascade. This helps the use of ϵ to define the length scale $\mathcal{L}$ at the macro-scale.
Now knowing that $K \sim \mathcal{UL}$ and using the kinetic energy k the following expression can be given

$$K = c_\mu \frac{k^2}{\epsilon} \tag{4.14}$$

where c_μ is a constant. The above equation is known as the $k - \epsilon$ model. Additional transport equation has to be defined for k while some modification in equation 4.13 have to be made to have complete closure of the problem.

The application of the $k-\epsilon$ model can be found in many problems relevant to engineering application. However, the lack of any theoretical framework of the K- theory can questions its applicability. In general, the model constants are defined for completely turbulent flows. As a result, the $k - \epsilon$ model only applicable for fully turbulent flows.

4.5 OpenFOAM numerics

OpenFOAM is mainly designed to solve problems in continuum mechanics. It is a C++ library which contains various numerical solvers and utilities. The basic structure of OpenFOAM can be seen in figure 4.13.

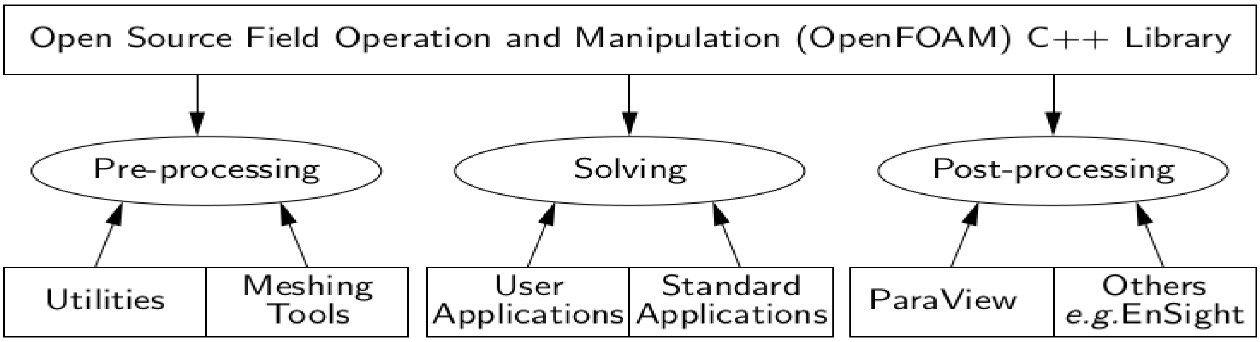
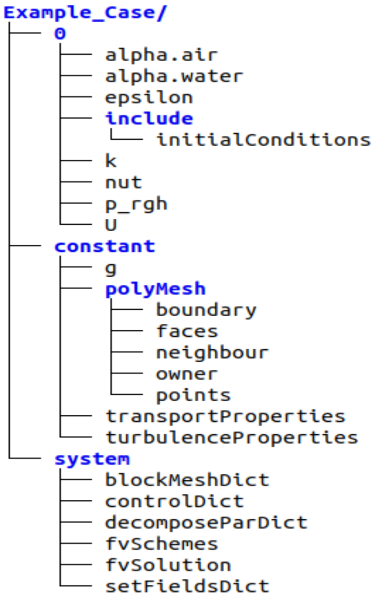


Figure 4.13: Overview of OpenFOAM structure (OpenFOAM & Community 2018)

OpenFOAM is an open-source software, which makes it affordable. In contrast to using proprietary software, the source code can be easily tweaked as per user needs. This makes OpenFOAM a powerful tool. However, the learning curve for OpenFOAM is steep. OpenFOAM uses a case-folder based system to carry out simulations. Figure 4.14 shows an example case structure for a multiphase simulation using the `interFoam` solver.



```
Example_Case/
├── 0
│   ├── alpha.air
│   ├── alpha.water
│   ├── epsilon
│   ├── include
│   │   └── initialConditions
│   ├── k
│   ├── nut
│   ├── p_rgh
│   └── U
├── constant
│   ├── g
│   ├── polyMesh
│   │   ├── boundary
│   │   ├── faces
│   │   ├── neighbour
│   │   ├── owner
│   │   └── points
│   ├── transportProperties
│   └── turbulenceProperties
└── system
    ├── blockMeshDict
    ├── controlDict
    ├── decomposeParDict
    ├── fvSchemes
    ├── fvSolution
    └── setFieldsDict
```

Figure 4.14: Tree structure of an OpenFOAM case

The case directory (Example_Case) consists of three main directories

1. **0** : This directory contains the files which include the initial conditions information for all the vector-Fields and scalar-Fields that are solved in the system.

2. **constant** : This directory contains the files associated with the mesh, physical properties, turbulence model properties and potentially other files such as mesh refinement criterion etc.

3. **system** : This directory contains the files pertaining to the mesh generation, numerical schemes, input-output control, and other utilities which can initiate and build the case.

Detailed information about the case files has been included in Appendix A. The detailed numerics of OpenFOAM are too broad to be discussed in the report. As a result, a reference to the documentation has been provided (OpenFOAM & Community 2018). The most critical features will be discussed in the section below.

Solver (`interFoam`) details

Most of the text below is based on the work by (Tomas 2016). This thesis report provides details about flux schemes, solver details etc which will not be discussed in this section.
The `interFoam` solver is a 2-phase solver for in-compressible flows. It employs a single momentum equation which solves for an averaged value of density and velocity (between the two fluids). As a result, it uses a continuity equation which is averaged for the two fluids.

$$\nabla \cdot \overline{\mathbf{u}} = 0 \tag{4.15}$$

where,
$\overline{\mathbf{u}} = \alpha_{l1} u_{l1} + \alpha_{l2} u_{l2}$ [m/s]
α_{l1} = phase fraction value of fluid 1 [-]
α_{l2} = phase fraction value of fluid 2 [-]
u_{l1} = velocity of fluid 1 [m/s]
u_{l2} = velocity of fluid 2 [m/s]

Along with this description of the mixed model, the momentum equation employed also follows a similar approach.

$$\frac{D\overline{u}}{Dt} = -\frac{1}{\overline{\rho}}\nabla p + \frac{\overline{\mu}}{\overline{\rho}}\nabla^2 \cdot \overline{\mathbf{u}} + g + \frac{F_s}{\overline{\rho}} \tag{4.16}$$

Using a similar averaging methodology as discussed above the equations as discussed below can be obtained. For capturing the fluid interface, an advection equation is used (Weller 2008)

$$\frac{D\alpha_l}{Dt} + \nabla \cdot (u_c \alpha_l (1 - \alpha_l)) = 0 \tag{4.17}$$

Flux limiters

Flux limiters (also known as slope limiters) are generally used when using higher order schemes to prevent spurious oscillations, when there exists a sharp fluid interface or in case of shocks in the flow. As the current study involves a sharp interface between the two phases (air & water) an appropriate flux limiter has to be employed. In this case for the interface the **van Leer** flux limiter was chosen. This definition can be seen in appendix A in the `fvSchemes` listing. The van Leer limiter is defined as follows (van Leer 1979):

$$\phi_{vl}(r) = \frac{r + |r|}{1 + |r|}, with \lim_{r \to \infty} \phi_{vl}(r) = 2 \tag{4.18}$$

4.6 Final layout of the numerical study

To summarize, the final approach of this study can be seen in figure 4.15.

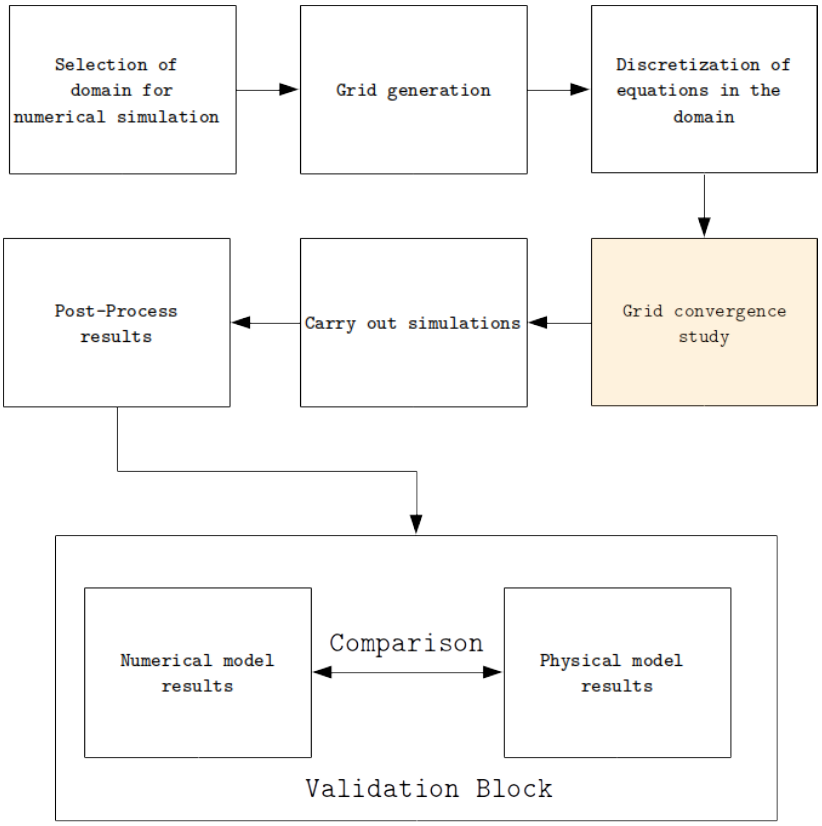


Figure 4.15: Schematic of the numerical model study

Due to the time constraint only 3 grid sizes will be checked for grid size effect on the numerical computations.

After the validation procedure is complete, the numerical results can be compared with the experimental results. A flow chart of the results that will be presented can be seen in figure 4.16. The grid/mesh naming convention has been presented in the next chapter.

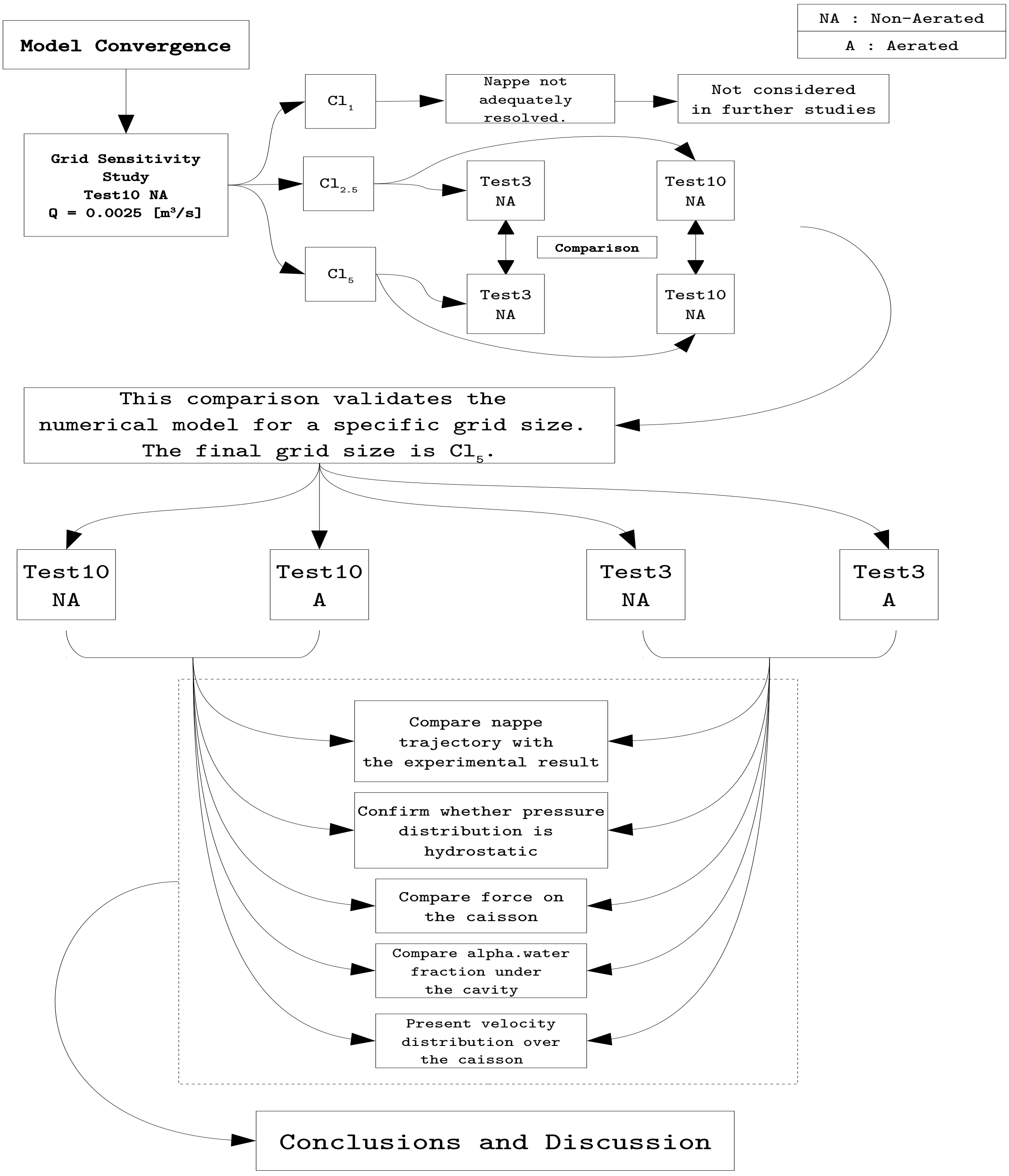

NA : Non-Aerated
A : Aerated
Model Convergence
Grid Sensitivity Study Test10 NA Q = 0.0025 [m³/s]
Cl1
Cl2.5
Cl5
Nappe not adequately resolved.
Not considered in further studies
Test3 NA
Test10 NA
Comparison
Test3 NA
Test10 NA
This comparison validates the numerical model for a specific grid size. The final grid size is Cl5.
Test10 NA
Test10 A
Test3 NA
Test3 A
Compare nappe trajectory with the experimental result
Confirm whether pressure distribution is hydrostatic
Compare force on the caisson
Compare alpha.water fraction under the cavity
Present velocity distribution over the caisson
Conclusions and Discussion

Figure 4.16: The figure presents the details of the validation block and the overall flow of the research carried out.

Chapter 5

Results and Discussions

"It doesn't matter how beautiful your theory is, it doesn't matter how smart you are. If it doesn't agree with experiment, it's wrong"

- Richard P. Feynman

5.1 Convergence of the numerical model

In order to assess the reliability of the numerical model, the iterative solver solutions have to be verified for residuals (See Greenshields 2015 for definition) in the computed variables (*p_rgh,k and epsilon*). The general tolerance for the residuals was set to $\sim 10^{-3}$. A sample plot for the residuals versus time (simulation) has been presented below.

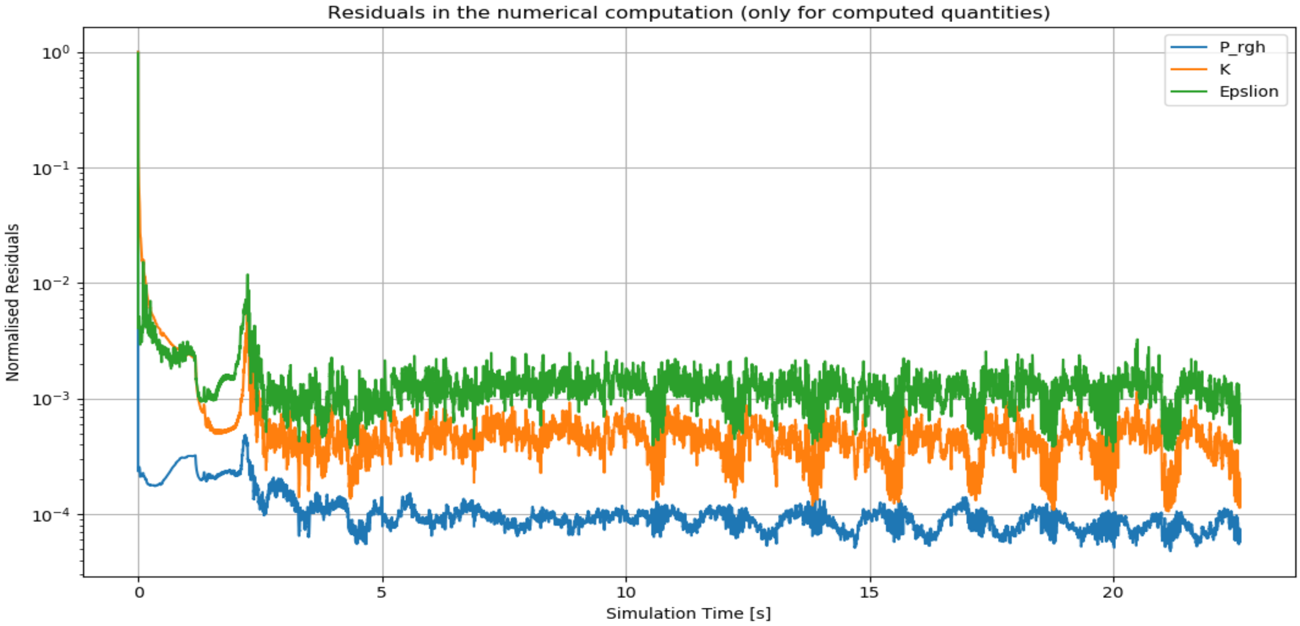


Figure 5.1: Residuals time series

The residuals history provides a safe way to assess the numerical model convergence. As the general trend of the numerical model falls below the convergence criteria , the numerical solutions can be trusted. This behaviour of computed residual quantities was observed in all the simulations that were done.

5.2 Grid sensitivity study

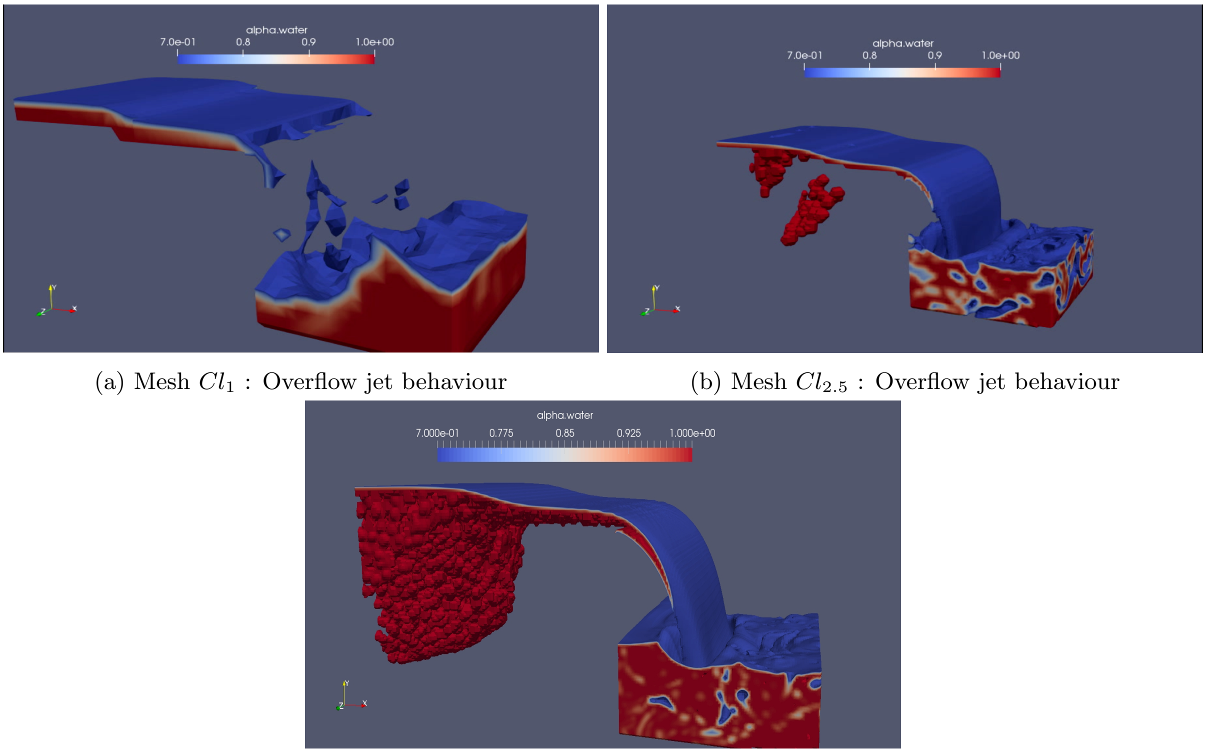
As discussed previously, three grid sizes have been checked for interface capture assessment using the `interFoam` solver. The three grid sizes which were used are listed in the table below.

Table 5.1: Grid sensitivity ratios studied. The grid size ratio is the same in all directions (x, y and z co-ordinate).

Mesh Name	**Size Ratio**
Cl_1	1 [cm] : 1 Grid Cell
$Cl_{2.5}$	1 [cm] : 2.5 Grid Cells
Cl_5	1 [cm] : 5 Grid Cells

5.2.1 Visual inspection of the overflow jet

The grid sensitivity of the interFoam solver can be visually estimated. The results for the mesh size as mentioned in Table 5.1 have been discussed below.



(a) Mesh Cl_1 : Overflow jet behaviour

(b) Mesh $Cl_{2.5}$: Overflow jet behaviour

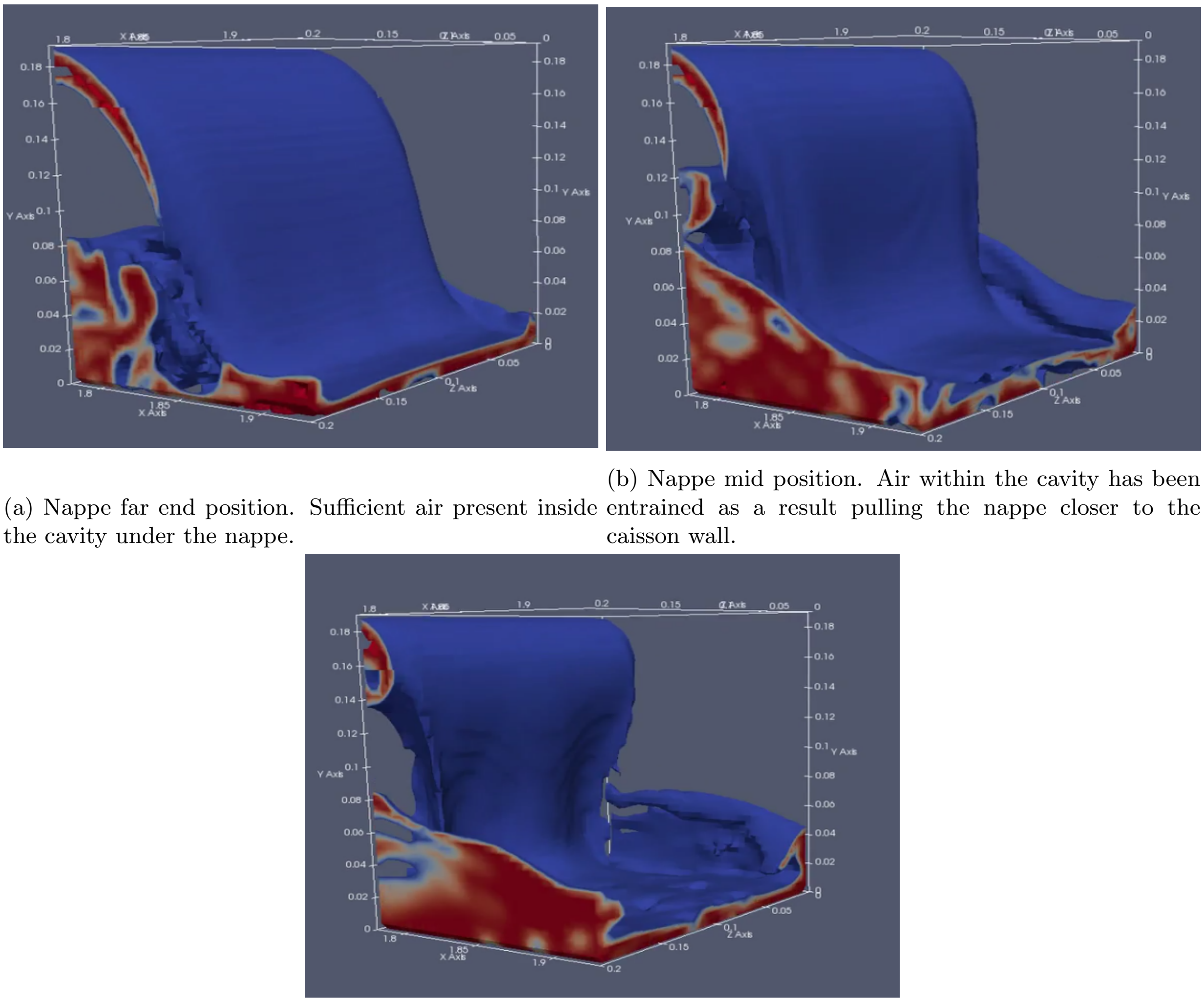
(c) Mesh Cl_5 : Overflow jet behaviour

Figure 5.2: Overflow jet behaviour with different mesh size ($Q = 0.0025$ $[m^3/s]$). The alpha.water fraction with 0.7 or more has been extracted as an isoVolume in the figure.

Clearly mesh Cl_1 does not capture the over flow jet nappe. As a result, Cl_1 cannot be used for further computations/analysis. The other two mesh size capture the over flow jet nappe sufficiently. As a result a qualitative comparison with the experimental data can be done for mesh $Cl_{2.5}$ and Cl_5. This qualitative comparison for mesh sizes $Cl_{2.5}$ and Cl_5 were done for two test cases from the experimental investigation.

5.2.2 Validation of overflow jet (nappe) trajectory

The overflow jet trajectory in the numerical model has been compared with the physical model results.



(a) Nappe far end position. Sufficient air present inside the cavity under the nappe.

(b) Nappe mid position. Air within the cavity has been entrained as a result pulling the nappe closer to the caisson wall.

(c) Nappe hugging the caisson wall. All the air within the cavity has been entrained. This results in the nappe hugging the caisson wall.

Figure 5.3: Non-aerated nappe behaviour for mesh Cl_5 (Test3 $Q = 0.00405$ $[m^3/s]$). This behaviour is cyclic and the nappe flutters about the mid position.

The overflow jet in the non-aerated case flutters about its mean position. This behaviour is due to the negative pressure build up inside the cavity under the nappe. The tail water level under the nappe rises and falls depending on the air-entrainment rate. As seen in figure 5.3, initially the cavity has sufficient air. The nappe at this moment is the furthest away from the caisson wall. However, as the air from the cavity is entrained the negative pressure builds up. This pulls the tail water higher and the nappe closer to the caisson wall. Eventually, the nappe hugs the wall for a brief instance of time. As the incoming momentum is high enough the nappe breaks at the sides of the flume wall and draws in sufficient amount of air and leaves the caisson wall again. This cycle of attachment and detachment of the jet to caisson wall continues as time progresses.

Considering this behaviour of the overflow jet, the experimental results cannot be compared one to one with the numerical model results. The experimental results have been derived (approximately) from the mean position of the nappe. For completeness the three positions of the nappe in the non-aerated case can be seen in the figures presented below.

Test 10

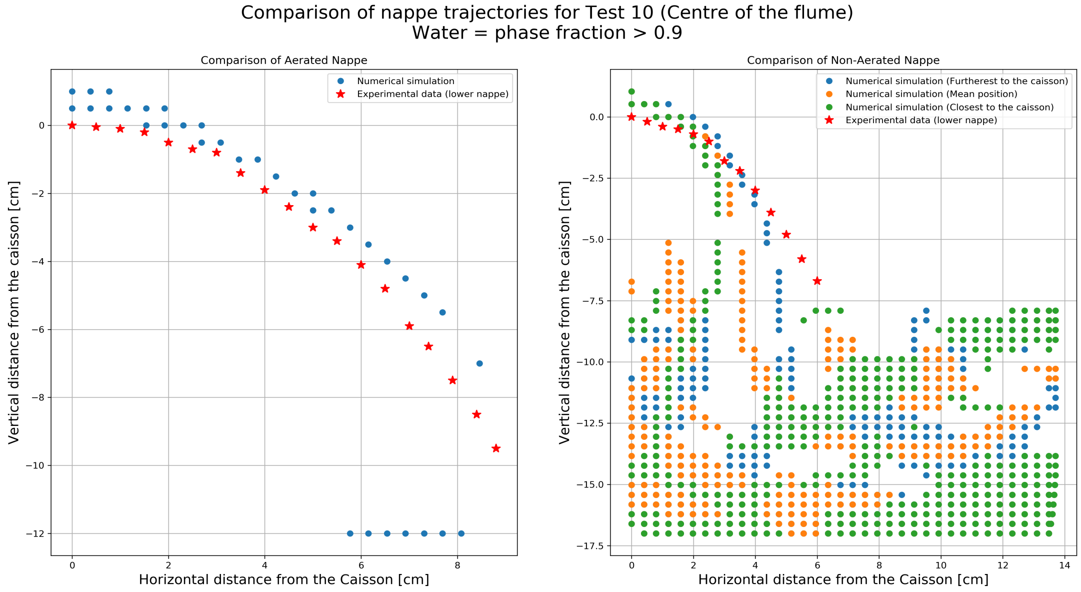


Figure 5.4: Test 10 nappe validation results with mesh size $Cl_{2.5}$

Figure 5.4 shows the validation results for the aerated and the non-aerated case for test 10 (Q = 0.0025 m^3/s). The nappe in this case does not flutter with a large amplitude. The image on the right side portrays the nappe behaviour for the three cases as labeled. The points at the bottom depict the tail water level for each case. Due to high numerical diffusion, the interface capture with this mesh size makes it difficult to obtain sufficient resolution in the nappe trajectory. In the aerated case, the nappe does not flutter. The nappe has been resolved to a good extent, however the mesh size is too coarse to resolve the overflow jet completely. As a result, a difference in the results can be seen in the numerical model. This motivates the need for finer mesh to adequately resolve the overflow jet trajectory.

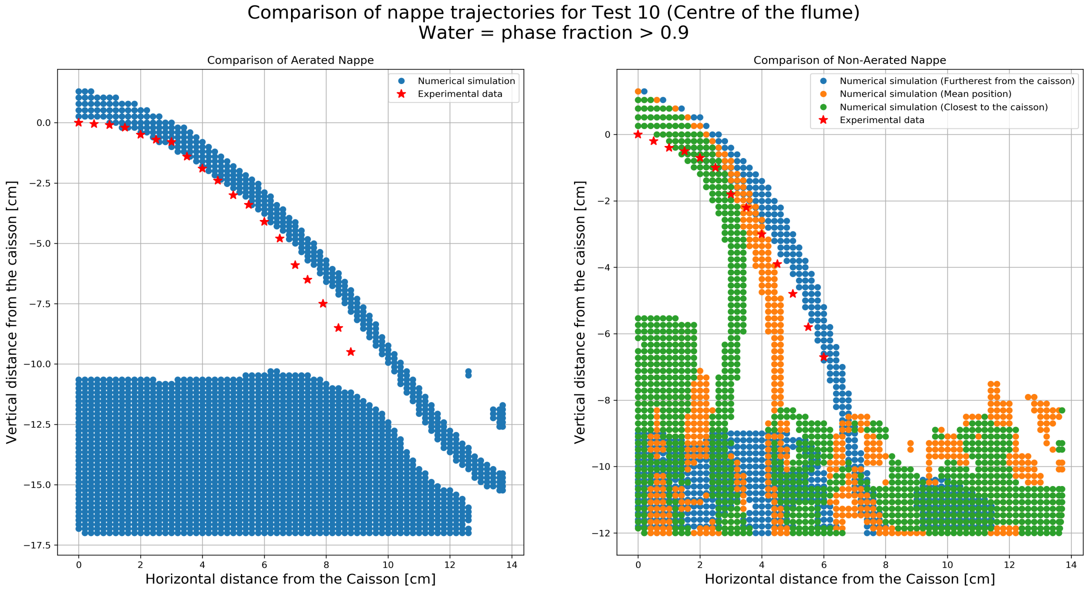


Figure 5.5: Test 10 nappe validation results with mesh size Cl_5

Using a finer mesh (Cl_5) certainly improves the overflow jet trajectory in the numerical model for test 10. This can be clearly seen in figure 5.5. The overflow jet behaviour for the non-aerated case with fine mesh is similar to that in the intermediate mesh ($Cl_{2.5}$). In contrast to the intermediate mesh, the fine mesh resolves the overflow jet trajectory and the size (thickness of the jet) adequately. It is also crucial to note the difference in the tail water level for the different non-aerated case overflow jet positions.

Test 3

Similar behaviour in the numerical results for test 3 (Q = 0.00405 m^3/s) are observed when compared to test 10 numerical results. Although the overflow jet trajectory is adequately resolved in this case (see figure 5.6), the size of the jet is not captured with sufficient detail. As a result, a fine mesh approach as proposed in the test 10 case was used to confirm the validity of the numerical model.

Figure 5.7 shows the results for fine mesh (Cl_5). It is clear to see that in the aerated case the numerical results follow the experimental results very well. The jet size and the trajectory are well captured in this numerical model. The non-aerated case shows high frequency and amplitude fluttering about its mean position in test 3. Tail water behaviour as seen in test 10 can also be seen in test 3. It is important to note the difference in the tail water levels for the aerated and the non-aerated case. This difference provides additional force which either stabilizes or destabilizes the caisson.

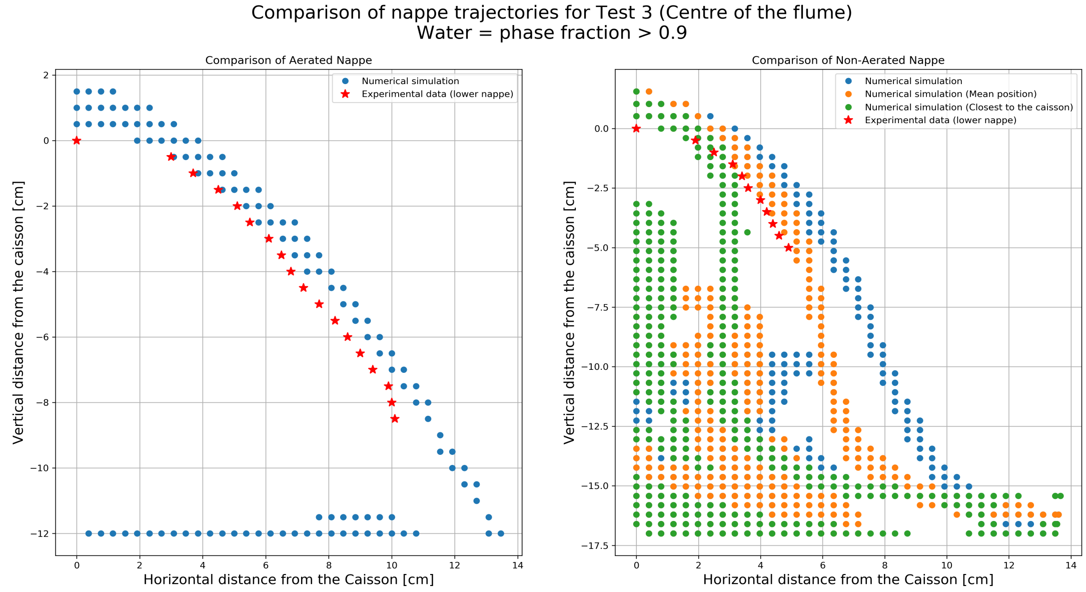


Figure 5.6: Test 3 nappe validation results with mesh size $Cl_{2.5}$

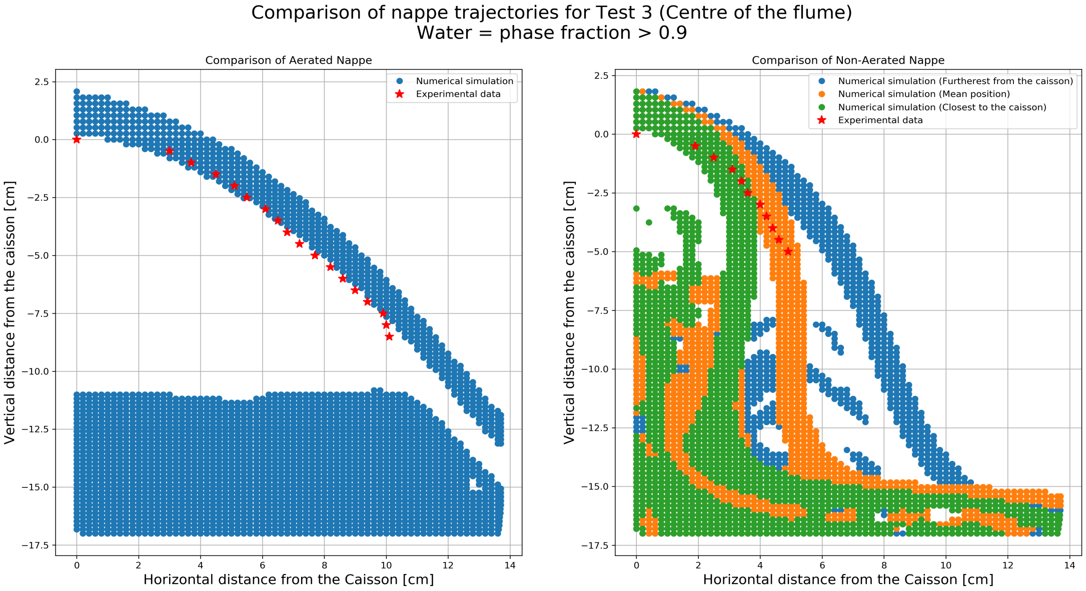


Figure 5.7: Test 3 nappe validation results with mesh size Cl_5

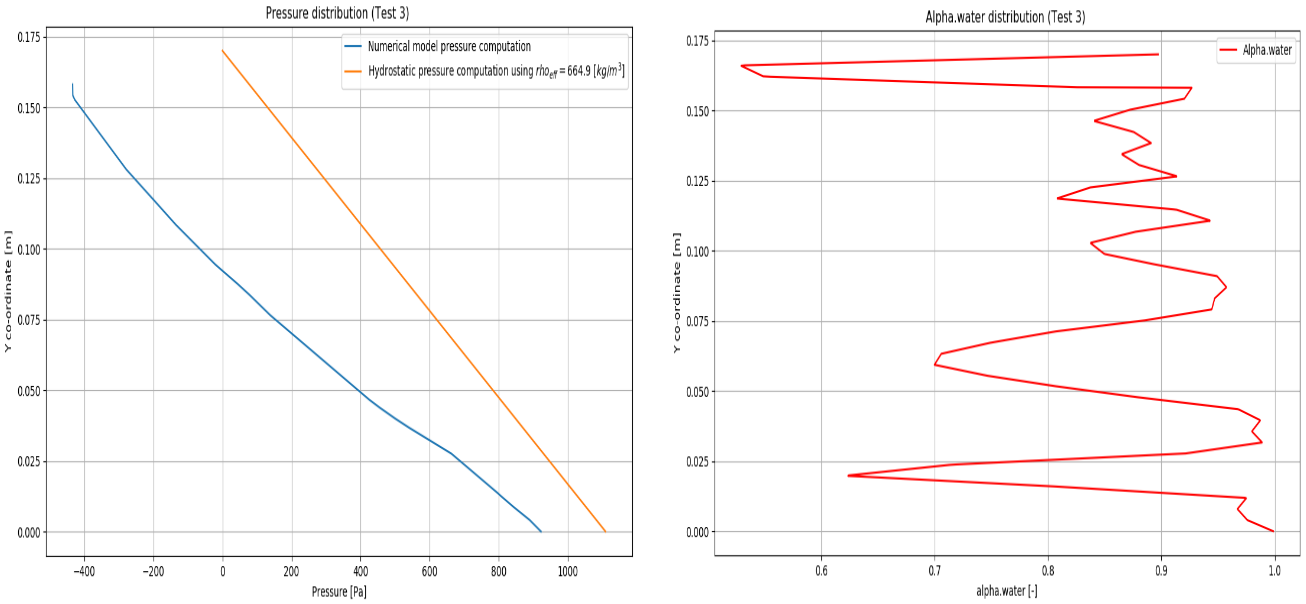
The validation of the non-aerated nappe in both the tests is sensitive to the time instance at which the nappe results are extracted. As a result, only the three positions as discussed above have been presented. The results of the numerical model are only stored every 0.5 [s]. This makes it difficult to correctly predict the position of the nappe in between the write-times (Time instance at which results are stored).

The numerical results with intermediate mesh ($Cl_{2.5}$) and fine mesh (Cl_5) follow the experimental results very well in both the test cases for aerated and non-aerated cases. As seen in the above section, the overflow jet resolution highly depends on the grid size used. In order to avoid grid effect, the final computations have been carried out using the fine mesh (Cl_5). In conclusion, the physics of the overflow jet over a caisson can be studied numerically using OpenFOAM's `interFoam` solver and $k - \epsilon$ turbulence model with a 3-D numerical model.

5.3 Pressure distribution in the tail water

One of the key assumptions in the study carried out by S. Mudiyanselage (2017) was the hydro-static assumption. The pressure inside the tail water under the cavity was assumed to be hydro-static. This assumption was used to derive the force acting on the caisson and other analytical results in the study carried out by S. Mudiyanselage (Mudiyanselage 2017). As a result, verification of this pressure distribution is a key result with regards to the applicability of the analytical method. The numerical model results have been used to verify this assumption.

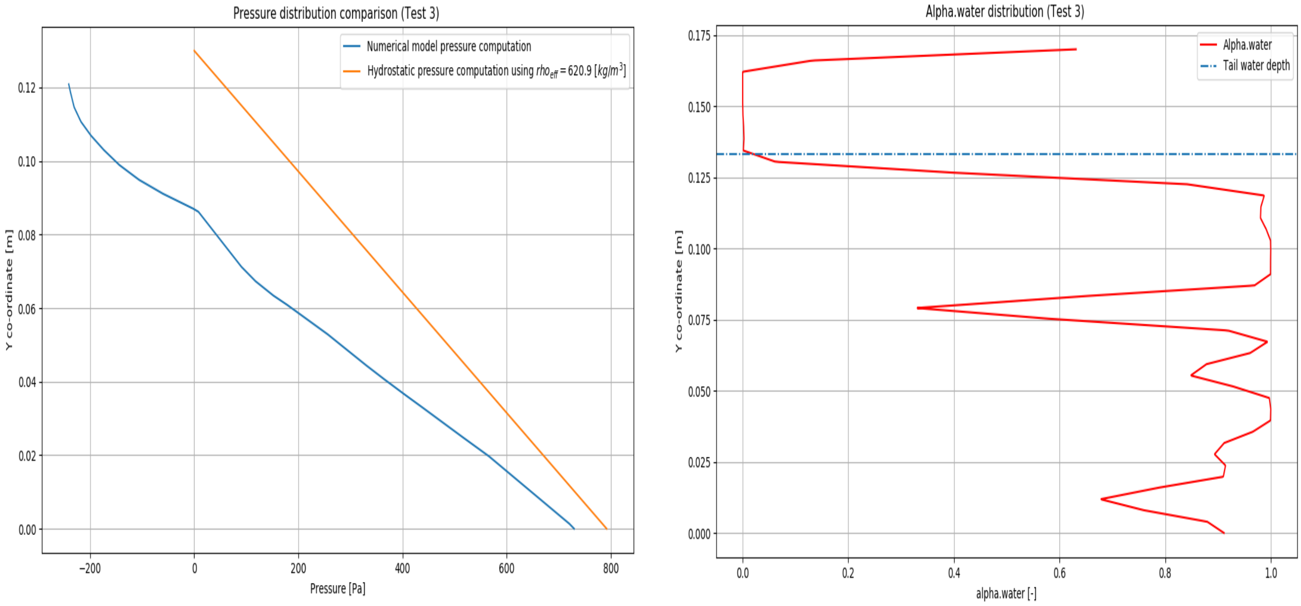
The pressure distribution in the tail water under the cavity has a deviation from the expected linear hydrostatic pressure distribution. As the alpha.water fraction is reduced, the effective density inside the tail water is also reduced. Using the reduced alpha.water fraction to compute hydrostatic pressure results in a pressure profile as shown in figures below (Only Test 3). The average alpha.water fraction has been considered from section 5.5.



(a) Pressure distribution downstream of the caisson.

(b) Alpha.water distribution downstream of the caisson.

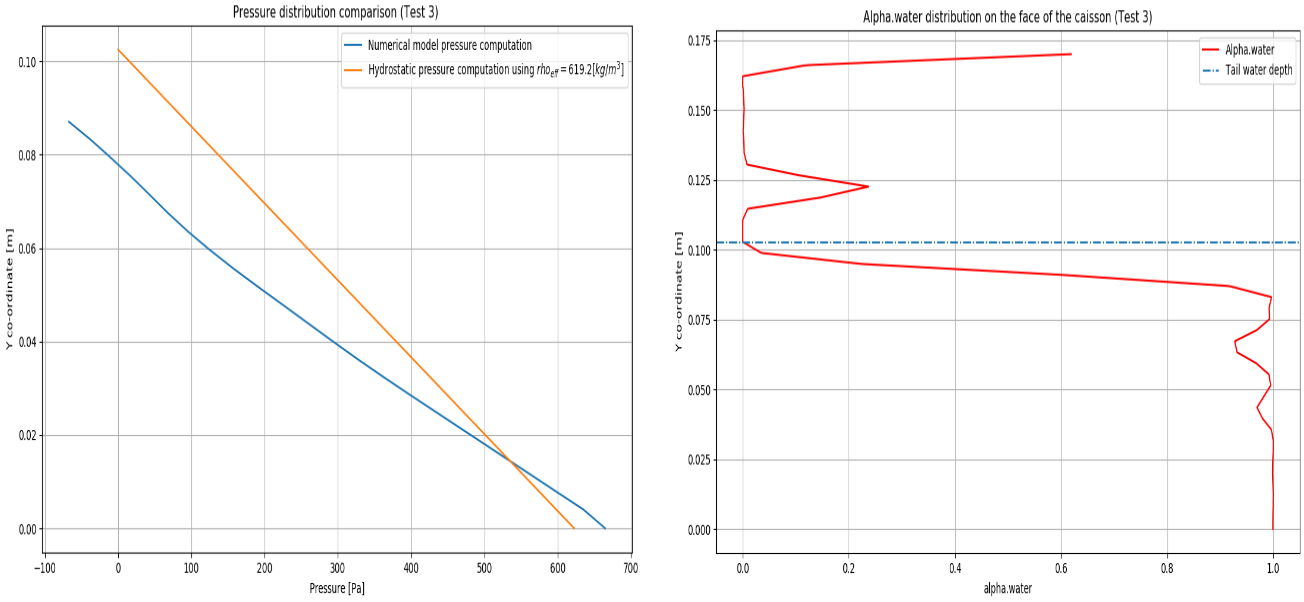
Figure 5.8: Pressure and air.water distribution downstream of the caisson. Overflow jet closest to the caisson.



(a) Pressure distribution downstream of the caisson.

(b) Alpha.water distribution downstream of the caisson.

Figure 5.9: Pressure and alpha.water distribution downstream of the caisson. Overflow jet mean position.



(a) Pressure distribution downstream of the caisson.

(b) Alpha.water distribution downstream of the caisson.

Figure 5.10: Pressure and alpha.water distribution downstream of the caisson. Overflow jet furthest from the caisson.

As seen in the figures, the pressure distribution is sensitive on the following parameters :

- Instance at which the results are extracted
- Write interval at which results are available (0.5 [s] in this case)

It is not clear if the alpha.water change or the dynamic pressure contributes to this difference in pressure behaviour. This drastic behaviour of the alpha.water fraction was not anticipated to be dealt during the computations. As a result, no log file can be generated to observe a time series behavior. Hence, due to the limited scope and time constraints of this study, a conclusion about the pressure distribution cannot be made with the current data from the numerical model.

5.4 Force on the caisson

The force on the caisson has a typical time series signature for the aerated and the non-aerated case. The force on the caisson is calculated as presented in listing A.7 (Line 65-77). The basic principle behind the force computation is that OpenFOAM applies a static equilibrium and then computes the force on the given boundary. A simple schematic which represents the crucial forces in the system can be seen in figure 5.11.

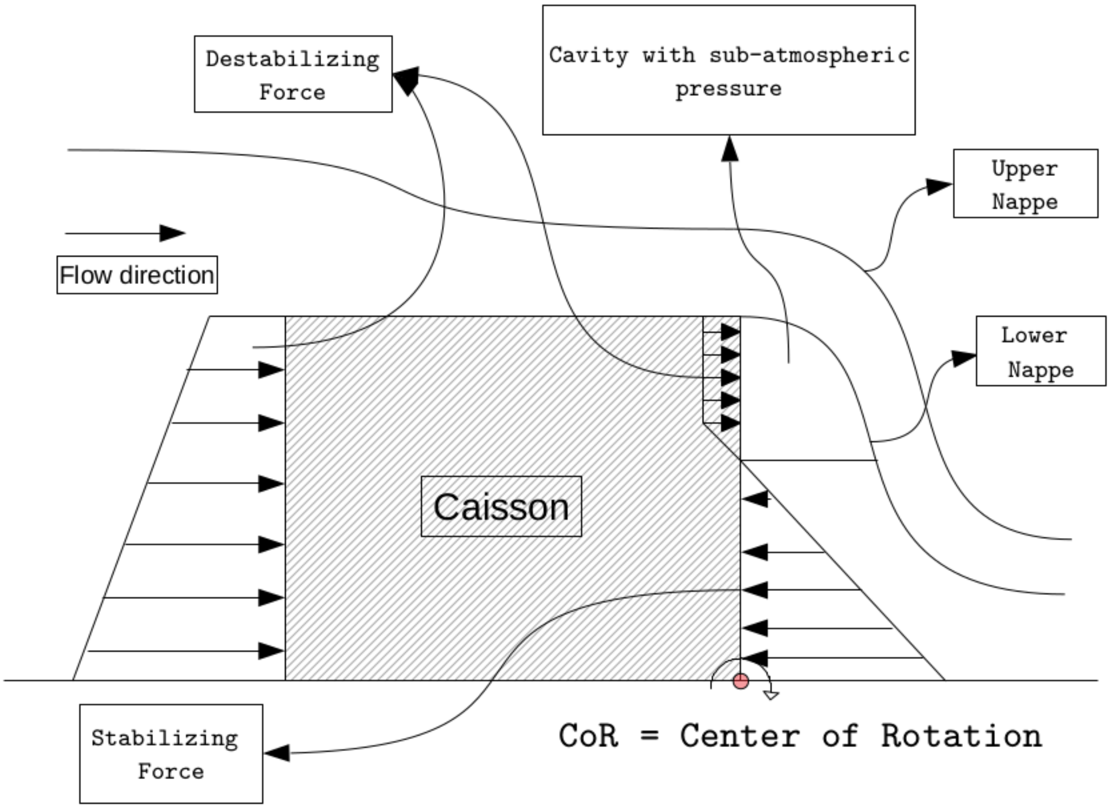


Figure 5.11: Highly simplified sketch of the forces acting on the caisson during over-topping

Using this idea, the force computations in the numerical model are as presented below.

Test 10

The force on the caisson in test 10 is as seen in figure 5.12. It is interesting to note that the overflow jet has small amplitude force fluctuations which correspond to the nappe fluttering as discussed before.

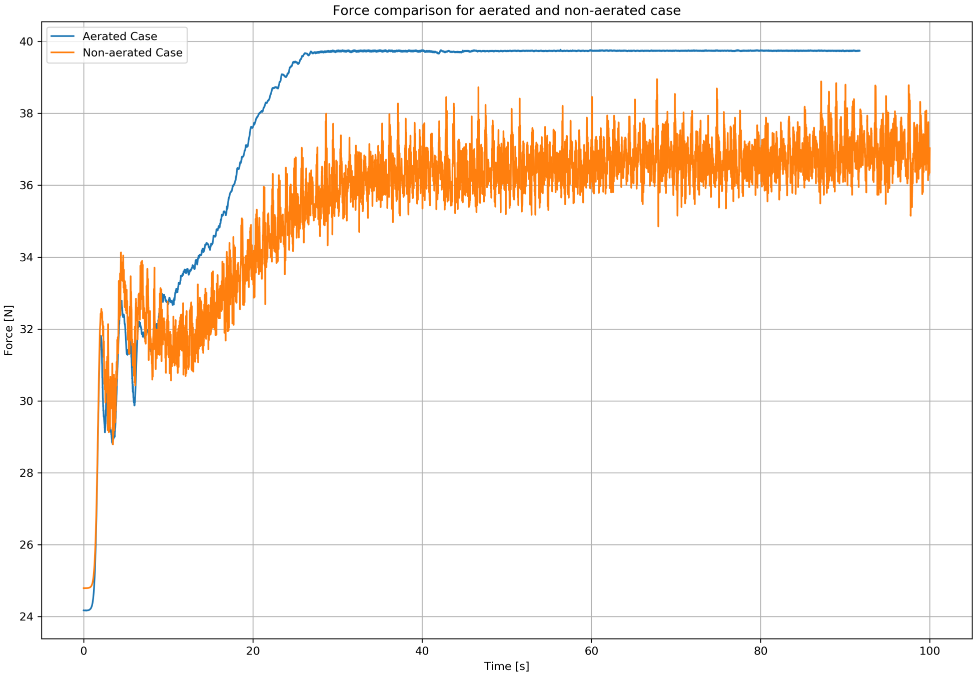


Figure 5.12: Comparison of force on the caisson for the two cases under investigation

The results in test 10 show that the non-aerated case is stable in comparison the aerated case. This maybe due to the change in tail water level as a function of time in the non-aerated case. In any case the results obtained go against the initial hypothesis of caisson stability in case of non-aeration of the overflow jet.

Test 3

The force on the caisson in test 3 is as seen in figure 5.13. Initially, both the cases behave in a similar fashion. However, the non-aerated case starts to develop fluttering due to air entrainment from the cavity. As opposed to test 10, the amplitude of oscillations/fluttering is relatively high. As a result, the caisson feels a higher force in case of the non-aerated overflow jet. The discharge in this case is higher in comparison to test 10, which results in the impinging velocity at the tail water being higher. Due to this higher momentum, the air entrainment rate is also higher in test 3.

The nappe trajectory is less curved (or further away from the caisson) when compared to test 10. As the air from the cavity is entrained, the nappe moves closer to the caisson wall (See figure 5.3). The air entrainment occurs at a faster rate, resulting in faster movement of the overflow jet back and forth. This explains the high peaks in the force felt by the caisson during this fluttering process. The tail water change also contributes to the additional force. In figures 5.5 and 5.7, it is clear to see the tail water behaviour.

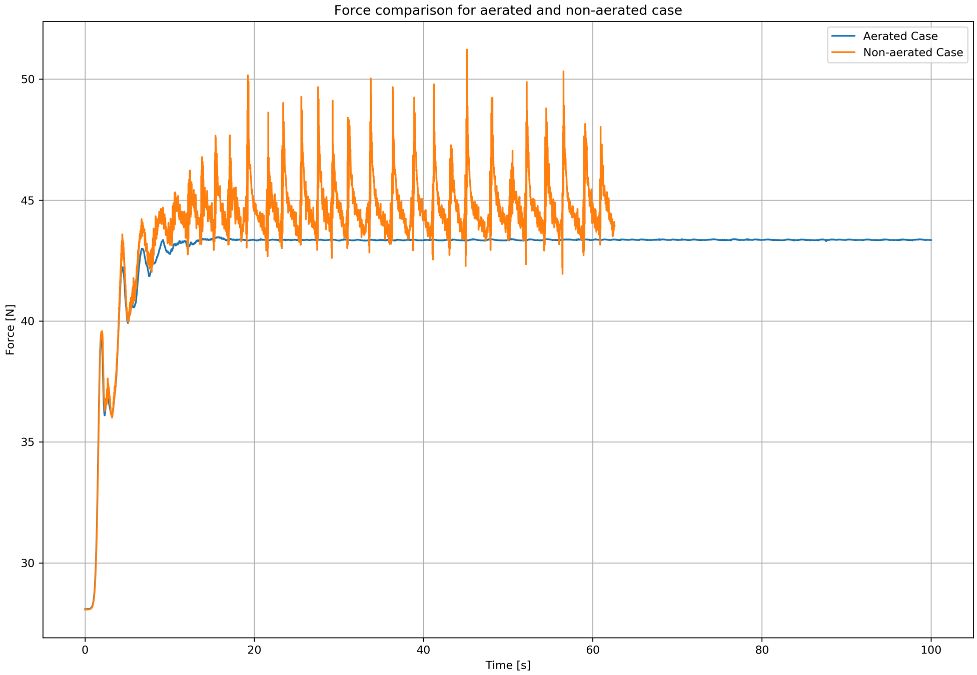


Figure 5.13: Comparison of force on the caisson for the two cases under investigation

NOTE : Due to computational limitations, the non-aerated case simulation has been cut-off at about 62 [s]

Additional force on the caisson

As seen in test 3, non-aeration can lead to additional force on the caisson. The maximum additional force in the two cases considered have been presented in table 5.2. The additional force is defined by equation 5.1.

$$F_{ad} = F_{na} - F_a \tag{5.1}$$

where, F_{ad} [N] is the additional force, F_{na} [N] is the maximum force in the non-aerated case and F_a [N] is the maximum force in the aerated case.

This maximum force on the caisson can be considered as the design condition.

Table 5.2: Maximum force on the caisson for different cases considered in this study

Test	Aerated [N]	Non-Aerated [N]	Additional force [N]
Test 10 (Q = 0.0025 $[m^3/s]$)	39.77	38.95	-0.82
Test 3 (Q = 0.00405 $[m^3/s]$)	43.47	51.23	7.76

The tail water level difference in test 10 is the cause for the negative additional force. This means that the force on the caisson is higher in the aerated case. For test 3, the additional force is about 17.85% of the total force in the aerated case. This is a key result in terms of the stability of the caisson.

5.5 Alpha.water fraction under the cavity

The tail water provides a stabilizing force for the caisson. This makes it imperative to discuss the effective density in the tail water under the cavity. The effective alpha.water for each case for various positions has been presented in table 5.3. The tail water rises and falls depending on how much air is entrained from the cavity. As a result, to have consistent comparison, a box as shown in figure 5.14 was used to compute the average alpha.water fraction.

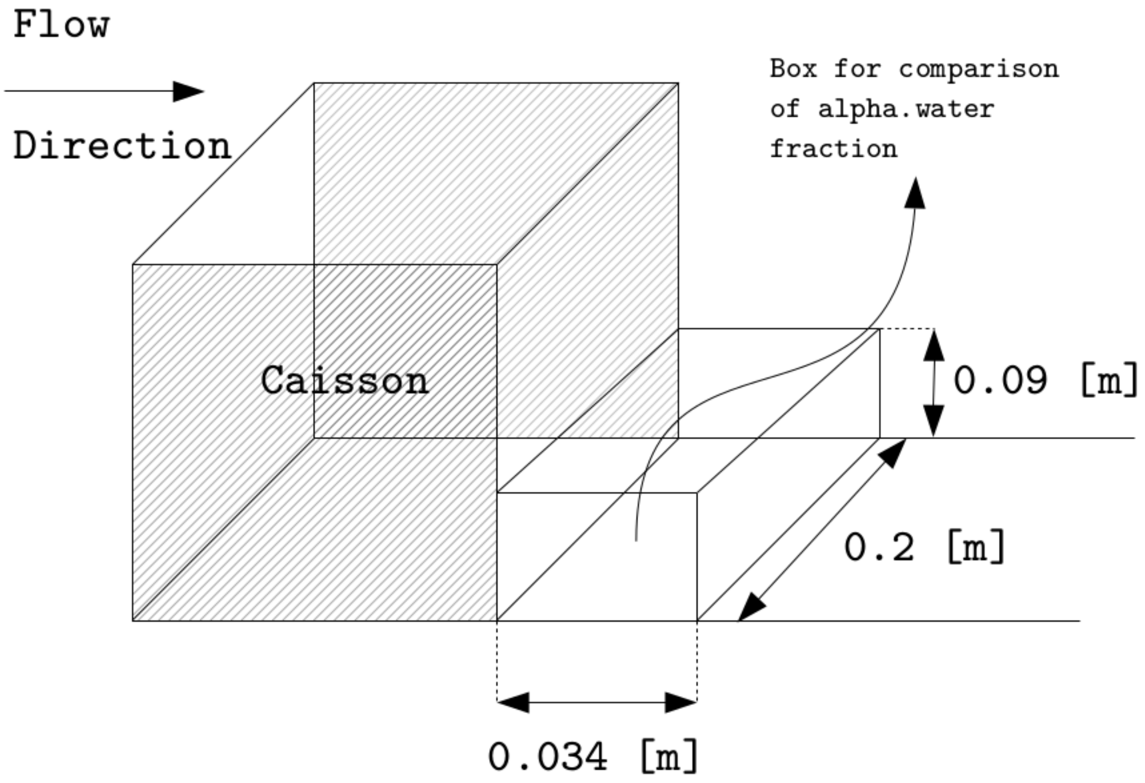


Figure 5.14: Schematic for alpha.water computation block in the domain

Table 5.3: Effective alpha.water under the cavity

Position of the nappe	Test 10 (α_w)	Test 3 (α_w)
Closest to the caisson	0.6547	0.6649
Mean position	0.8398	0.6209
Furthest from the caisson	0.6433	0.6192

The effective density for each case can be computed using equation 5.2.

$$\rho_{eff} = \alpha_w \cdot \rho_{water} \tag{5.2}$$

where,
ρ_{eff} is the effective density $[kg/m^3]$ α_w is the alpha.water fraction [-] and ρ_{water} is the density of water $\sim$ 1000

$[kg/m^3]$.

The density/alpha.water fraction is not constant during the entire overflow period. The tail water behaviour under the cavity is highly dynamic and depends on various factors such as the sub-atmospheric pressure in the cavity, rate of air-entrainment from the cavity, nappe trajectory and the tail water depth itself. However, no significant conclusion can be drawn from the current numerical model with regards to the distribution of alpha.water fraction under the cavity. The results are pretty inconsistent for various positions of the nappe. This is because the alpha.water fraction is sensitive to the time instance at which the data block is extracted for analysis.

5.6 Velocity distribution over the caisson

The flow over the caisson was assumed to follow the broad-crested weir type flow. Validation of this flow regime was already done by S. Mudiyanselage (Mudiyanselage 2017). However, no description of the velocity distribution over the depth was provided. As a result, figure 5.16 presents the velocity distribution over the caisson ends for both the tests. The distance at which the plots are seen have been converted to dimensionless units for simplicity. The definition of the distance can be seen in figure 5.15.

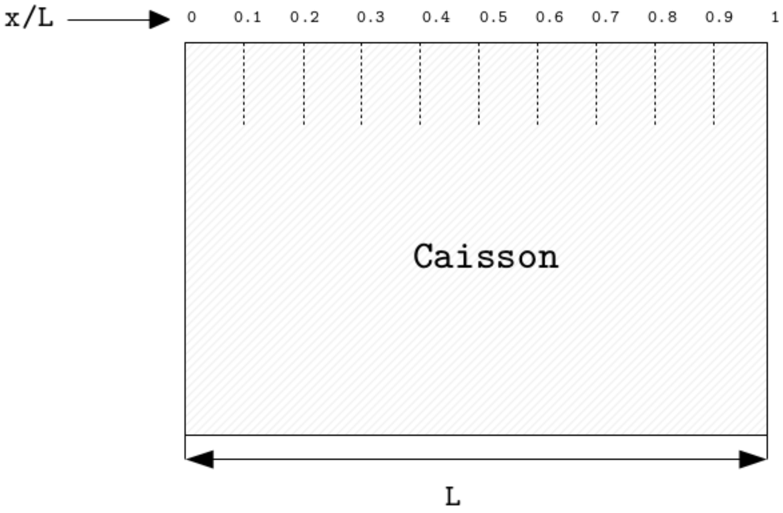
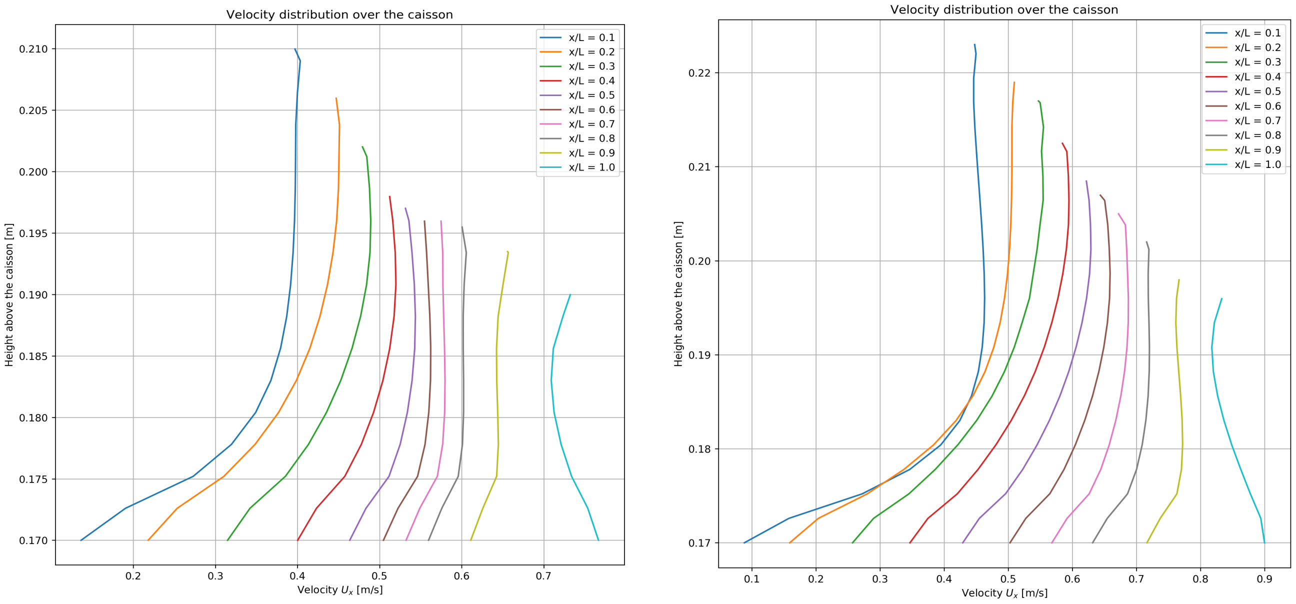


Figure 5.15: Locations where velocity profile has been presented. The distance has been scaled with respect to the length of the caisson.

Figure 5.16 shows the velocity distribution for both the test cases investigated in this study. The velocity near the brink/end of the caisson is higher near the wall. This observation is in line with the studies carried out previously (Felder & Chanson 2012). This also explains the behaviour of the overflow jet. As the velocity near the wall is higher, the jet simply shoots past the caisson. This also explains why the overflow jet does not have a different behaviour near the wall and inside the core of the jet. The velocity profile undergoes a change between $x/L = 0.9$ and $x/L = 1.0$. Velocity near the bottom becomes higher than that in the rest of the water column/jet. This flattening of the velocity profile along the caisson is quite pronounced in the two test cases presented in this study.



(a) Velocity distribution along the caisson top wall for test 10

(b) Velocity distribution along the caisson top wall for test 3

Figure 5.16: Velocity distribution for the numerical model.

Chapter 6

Conclusions and Recommendations

"Once you have eliminated the impossible, whatever remains, however improbable, must be the truth"

- Spock

This section will provide the conclusions for the current research work. A detailed discussion to answer the research questions has been presented in the section below.

6.1 Conclusions

- **Can OpenFOAM be used to simulate turbulent flows that occur during over-topping of a breakwater?**

  OpenFOAM proves to be a good numerical tool to model flow over the caisson breakwater. The validation results by S. Mudiyanselage (Mudiyanselage 2017) and the current study have provided insights into the modeling process and numerical model set-up to obtain sufficiently accurate results without expending vast computational resources. However, the results from the numerical results are quite sensitive to the grid size used in the numerical model. Using higher order schemes and finer grid can result in adequate resolution of the overflow jet. Resolution of the overflow jet is one of the major challenges in using the `interFoam` solver.

- **What kind of simplifications are required in order to have adequate resolution and at the same time limit the computational resources utilized?**

  It was clear that the 3-dimensional physics of the overflow jet cannot be studied using a 2-dimensional model. Previous studies (Mudiyanselage 2017) tried to implement a 2D numerical approach in order to model the flow over the caisson in the non-aerated case. However, as seen in figure 6.1 the overflow jet hugs the caisson wall and never leaves the surface. This is clearly due to the absence of any physical means for the air to enter back into the cavity in a 2D model. As a result, the flow over a caisson cannot be modeled as a 2D process.

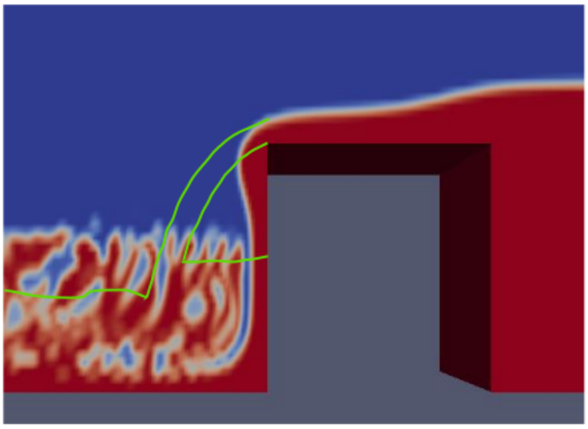


Figure 6.1: 2D numerical model studies carried out by S. Mudiyanselage (Mudiyanselage 2017). The image shows the overflow jet hugs the caisson wall.

In order to correctly model the physics of the overflow jet, a 3-dimensional model with a fine grid has to be used to capture the liquid-air interface. In comparison to the 2-D model, the 3-D model can cater for the physics of overflow jet contraction in the 3-dimensions. This resolution allows the free breakup of the jet at the sides allowing air-entrainment into the cavity. This cannot be achieved in the 2-D model. In order to limit computational expense, a simplified process as described in figure A.3 can be followed.

- **Does the standard $k - \epsilon$ turbulence model perform adequately?**

The standard $k - \epsilon$ model as discussed in the current study performs adequately. However, there are certain model deficiencies (especially in separated flows) viz highly diffusive, stability issues and applicability in separated flows, that need to be addressed in terms of the turbulence model effects.

- **How does the force on the caisson compare in the aerated and non-aerated jet case?**

Non-aeration of the overflow jet results in fluttering of the jet about its mean position. The cycles of aeration and non-aeration can introduce momentum changes resulting additional forces on the caisson. The physical process by which this flutter occurs has already been discussed in section 5.2.2. In test case 3 ($Q = 0.00405$ $[m^3/s]$) the additional force (maximum) was about 17.85% of the force felt by the caisson in the aerated case. However, in test 10 ($Q = 0.0025$ $[m^3/s]$) the force on the caisson was higher in the aerated case in comparison to the non-aerated case. This can be attributed to the difference in the tail water level and the reduced effective density of the tail water under the cavity. To recapitulate, there is a definitive relationship between non-aeration and additional force on the caisson.

With regards to the stability of the caisson, a static analysis also would not be sufficient. The fluttering of the overflow jet produces large momentum changes which does not lead to a constant load on the caisson. This cyclic load on the caisson can prove detrimental to the stability of the caisson. As a result, the amplitude at which the force oscillates and frequency are crucial from the design perspective.

- **Provide an explanation of the aeration mechanism of the cavity under the nappe.**

The aeration mechanism under the cavity has been discussed in section 5.2.2. To summarize the discussion, a step by step guide to cavity aeration is listed below.

1. Air is entrained from the cavity and overflow jet moves closer to the caisson wall.
2. Tail water pulled upwards as the cavity grows smaller in size.
3. Overflow jet increases curvature and starts breaking at the sides of the flume. Simultaneously the overflow jet develops a curvature along the width of the flume (z co-ordinate).
4. After sufficient air in entrained from the cavity, the overflow jet cannot sustain the curvature (due to high incoming horizontal momentum) and breaks at the sides and draws in air as a result of this breaking.
5. As air has been entrained back into the cavity, the overflow jet resumes its initial trajectory and starts the cycle of fluttering.

Similar behaviour can be observed at the pressure sensor 2. Figure 6.2 illustrates this growth/build up of pressure and then sudden collapse. This behaviour is observed due to the tail water moving up and down. Pressure sensor 2 readily captures this movement of tail water (Test 3).

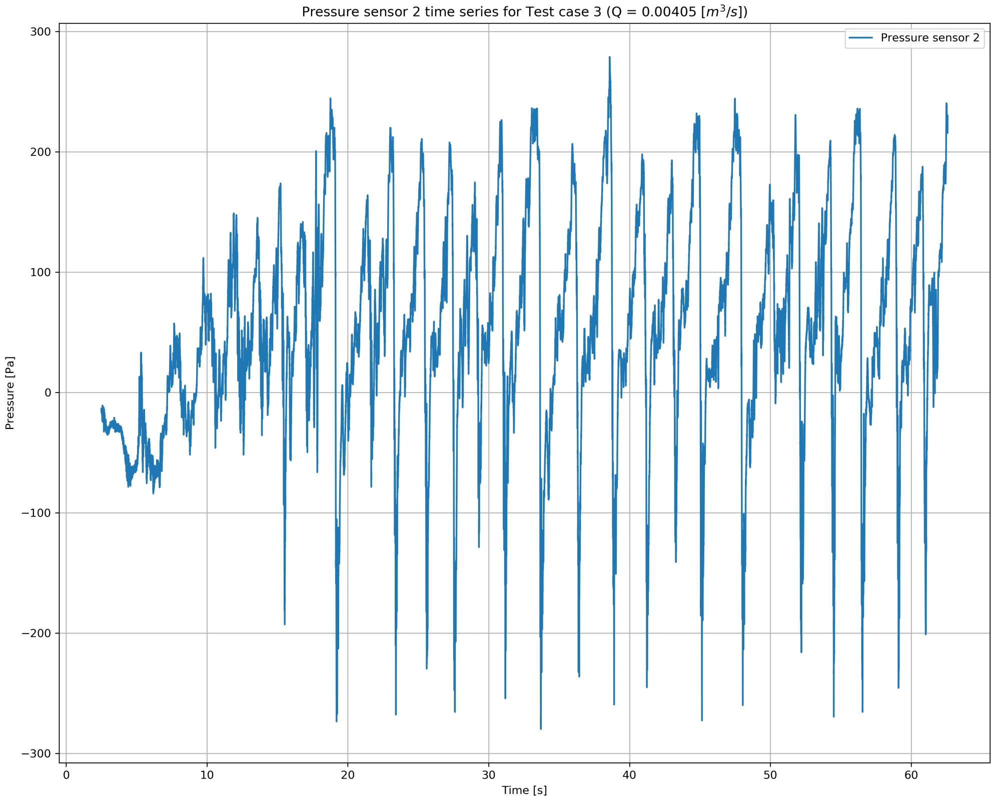


Figure 6.2: Pressure sensor 2 time series for test case 3.

- **Provide a general relationship between the upstream caisson head (H_d) and the additional force (F_{ad}) generated due to non-aerated cavity under the nappe.**

A basic relationship between overflow discharge and the additional force on the caisson can be seen in figure 6.3. The overflow discharge can quite easily be converted to a H_d-Force relationship. However, for sake of simplicity it has not been done here. Due to limited scope of the project and time constraint, the data obtained is not sufficient to draw sound conclusion about the relationship between overflow discharge/head and additional force on the caisson.

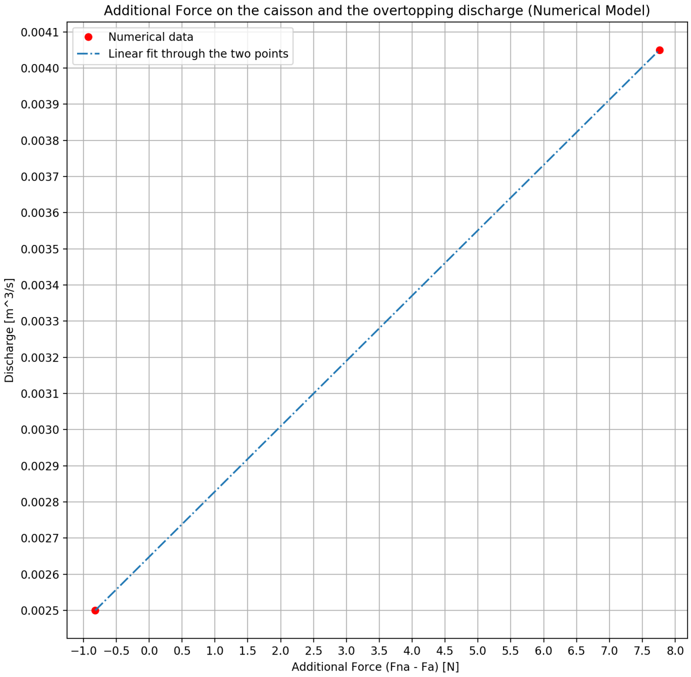


Figure 6.3: Discharge-Additional force relationship as observed in the numerical model studies.

6.2 Recommendations

In this study, the numerical model (CFD) was used to replicate the experimental studies performed earlier. This numerical model provided some keen insights in the flow behaviour and expected dynamics of the over-topping of a breakwater. However, there have been certain limitations with respect to the research carried out using this framework and they have been discussed below.

The current study only performs comparison between two test cases. Due to lack of numerical data, an authentic conclusion about the behaviour of the additional force with respect to the non-aeration cannot be made. In order to quantify this relationship, large number of test case comparisons with varying overflow discharge can be assessed.

The force on the caisson is quite sensitive to the tail water depth under the cavity. Controlling the behaviour of the tail water in the aerated case is not that straightforward. As a result, additional investigation with respect to the tail water levels can be done to assess its sensitivity.

The numerical simulations are quite sensitive to the number of cores used. It was observed that switching from 40 computing cores to 20 changed the results quite a bit. The OpenFOAM version used in this study was OpenFOAM-4.1, no bugs about `mpirun` were found during the investigation. As a result, additional research about sensitivity to parallel computing and the effect on the results can be done.

Applicability of the standard $k - \epsilon$ turbulence model can be questioned for problems involving flow separation. As a result, investigation with respect to the different turbulence closures and/or Large Eddy Simulations can provide detailed information about the behaviour of the overflow jet. Especially in case of lower discharges the behaviour of the overflow jet is drastically different as the jet does not completely break.

Bibliography

1. Arikawa, T. *et al.* Failure mechanism of the Kamaishi breakwater due to the great east Japan earthquake tsunami. *Coastal Engineering* (2012).
2. Bricker, J. D., Takagi, H. & Mitsui, J. Turbulence model effects on VOF analysis of breakwater overtopping during the 2011 Great East Japan Tsunami. *Proceedings of the 2013 IAHR World Congress* **10153** (2013).
3. CFD-Online. *Turbulence intensity* `https://www.cfd-online.com/Wiki/Turbulence_intensity`. [Online Accessed, 10th August 2018].
4. Felder, S. & Chanson, H. Free-Surface Profiles, Velocity and Pressure Distributions on a Broad-Crested Weir: A Physical Study. *Journal of irrigation and drainage engineering* (2012).
5. Furbo, E. *Evaluation of RANS turbulence models for flow problems with significant impact of boundary layers* (2010).
6. Greenshields, C. J. OpenFOAM programmers guide. *OpenFOAM Foundation Ltd.* (2015).
7. Hunter, J. C., Crawley, A. W., Petrie, M., Yang, J. E. & Aragón, T. J. *Local Public Health System Response to the Tsunami Threat in Coastal California following the Tōhoku Earthquake* `http://currents.plos.org/disasters/article/local-public-health-system-response-to-the-tsunami-threat-in-coastal-california-following-the-tohoku-earthquake/`. 2012.
8. Hydraulics. CIVE2400 Fluid Mechanics Section 2: Open Channel Hydraulics. *Online* [**Online, Accessed 09-August-2018**] (2018).
9. Kajishima, T. & Taira, K. *Computational Fluid Dynamics : Incompressible Turbulent Flows* doi:`10.1007/978-3-319-45304-0` (Springer International Publishing, Switzerland, 2017).
10. Kolmogorov, A. N. A refinement of previous hypotheses concerning the local structure of turbulence in a viscous incompressible fluid at high Reynolds number. *Journal of Fluid Mechanics* **13(01)** (1962).
11. Lesieur, M. *Turbulence in Fluids* (Springer, 2008).
12. Moin, P. & Mahesh, K. Direct numerical simulation: A Tool in turbulence research. *Annual Review Fluid Mechanics* **30** (1998).
13. Mudiyanselage, S. D. *Effect of Nappe Non-aeration on Caisson Sliding Force during Tsunami Breakwater Overtopping* (2017).
14. Nieuwstadt, F. T., Boersma, B. J. & Westerweel, J. *Turbulence: Introduction to Theory and applications of turbulent flows* (Springer International Publishing,Switzerland, 2016).
15. Onishi, N. *Japan Revives a Sea Barrier That Failed to Hold* ”`https://www.nytimes.com/2011/11/03/world/asia/japan-revives-a-sea-barrier-that-failed-to-hold.html,`”. [Online, Accessed 10-June-2018]. 2011.

16. OpenFOAM & Community. *OpenFOAM, the opensource CFD toolbox* `https://www.openfoam.com`. [Online, Accessed 10-June-2018]. 2018.

17. Patil, A., Mudiyanselage, S. D., Bricker, J. D., Uijttewaal, W. & Keetels, G. Effect of overflow nappe non-aeration on Tsunami breakwater failure. *ICCE.* [Draft version submitted] (2018).

18. Pope, S. *Turbulent Flows* (Cambridge University Press, 2013).

19. Roache, P. J. Quantification of uncertainty in Computational Fluid Dynamics. *Annual Review of Fluid Mechanics* **29** (1997).

20. Stro, J. N. *Large-Eddy Simulation and Turbulent Energy Cascade: A brief overview of ideas and concepts* `http://ffden-2.phys.uaf.edu/647fall2013_web.dir/j_stroh/intro.html`. 2013.

21. Tomas, P. P. *CFD of multiphase pipe flow: A comparison of solvers* `https://repository.tudelft.nl/`. 2016.

22. Van Leer, B. Towards the ultimate conservative difference scheme. V. A second-order sequel to Godunov's method. *Journal of Computational Physics* **32(1)** (1979).

23. Weller, H. G. *A new approach to VOF-based interface capturing methods for incompressible and compressible flow* 2008.

Appendix A

Details of the case setup and files

This chapter will discuss in detail the case setup in OpenFOAM. A familiarity to the Linux bash shell environment has been assumed in the current presentation of the case setup. This chapter will mainly focus on details about OpenFOAM and will not discuss navigation through the bash shell or other related details. See the following hyperlink "`http://www.ee.surrey.ac.uk/Teaching/Unix/`" for more information about linux/UNIX related information .

Before the case set-up procedure can be initiated, it is worth noting that the choice of the solver can affect the steps that will be discussed below. As certain solvers can vary in the each (or a few) step(s), the current description only deals with the set-up of `interFoam` solver.

A.1 Pre-Processing

This part of the case setup consists of the following steps:

1. Mesh generation, Mesh visualization, Mesh quality check.
2. Setting the boundary and initial conditions.
3. Specification of the turbulence models to be solved, physical properties or the fluids, frame of reference and acceleration in the system etc.

A.1.1 Mesh generation

There are two (possibly more) ways to generate a mesh/grid on which the field variables will be computed. As the geometry is simple, `blockMesh` will be used to generate the mesh. The geometry of the current case can be seen in figure A.1.

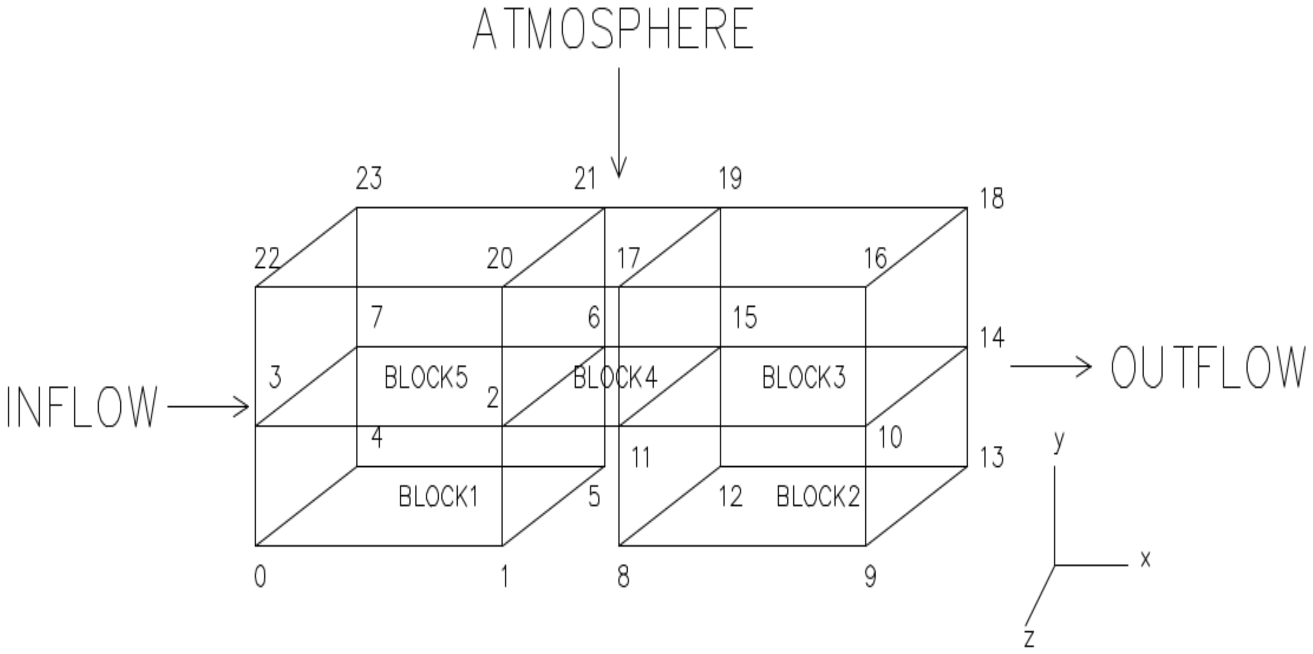


Figure A.1: Basic blockMesh geometry with points specification. The generic boundary conditions for the model have also been specified in the above figure. The co-ordinates for the points can be see in the `blockMeshDict` file.

The following `blockMeshDict` file has been used to generate the mesh in **Test_10_non-aerated**.

NOTE : The setup for aerated case includes an additional block which has not been included in the case below.

Listing A.1: blockMeshDict file

```
/*---------------------------------*- C++ -*-----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  4.1                                   |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    object      blockMeshDict;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

convertToMeters 0.01; //Conversion from [m] to [cm] scale


vertices    // Definition of points as described in figure
(

    (0 0 10) //0
    (163 0 10) //1
    (163 17 10) //2
    (0 17 10) //3

    (0 0 0) //4
    (163 0 0) //5
    (163 17 0) //6
    (0 17 0) //7

    //BLOCK 2
    (178.3 0 10) //8
    (228.3 0 10) //9
    (228.3 17 10) //10
    (178.3 17 10) //11

    (178.3 0 0) //12
    (228.3 0 0) //13
    (228.3 17 0) //14
    (178.3 17 0) //15

    //BLOCK 3
    (228.3 30 10) //16
    (178.3 30 10) //17

    (228.3 30 0) //18
    (178.3 30 0) //19

    //BLOCK 4
    (163 30 10) //10

    (163 30 0) //21

    //BLOCK 5
    (0 30 10) //22

    (0 30 0) //23

);

blocks      // Definition  of  blocks  as  described  in  figure
(
    //The block  definition  should be such that the volume formed is  positive  ( ref .  figure  and  definition )
    //  ( Points  in  block ) (x    y    z)   expansionRatio  (x y z)
    hex (0 1 2 3 4 5 6 7) (816  85 50) simpleGrading (1 1 1)        //Block 1
    hex (8 9 10 11 12 13 14 15) (250 85 50) simpleGrading (1 1 1) //Block 2
    hex (11 10 16 17 15 14 18 19) (250 50 50) simpleGrading (1 1 1) //Block 3
    hex (2 11 17 20 6 15 19 21) (85 50 50) simpleGrading (1 1 1) //Block 4
    hex (3 2 20 22 7 6 21 23) (816 50 50) simpleGrading (1 1 1) //Block 5
);

edges
(
);

boundary //Definition  of  patches
(
    inlet
    {
        type patch;
        faces
        (
            (3 7 23 22)
            (0 4 7 3)
        );
    }

    outlet
    {
        type patch;
        faces
        (
            (9 13 14 10)
            (10 14 18 16)
        );
    }

    atmosphere
    {
        type patch;
        faces
        (
            (22 20 21 23)
            (20 17 19 21)
            (17 16 18 19)
        );
    }

    breakwater
    {
    type wall;
    faces
    (
            (1 5 6 2)
            (2 11 15 6)
            (8 12 15 11)
    );
    }

    walls
    {
        type wall;
        faces
        (
            (0 1 2 3)
            (8 9 10 11)
            (11 10 16 17)
            (2 11 17 20)
            (3 2 20 22)
            (0 1 5 4)
            (8 9 13 12)
        );
    }

    symmetry
    {
    type    symmetryPlane;
    faces
    (
            (4 5 6 7)
            (12 13 14 15)
            (15 14 18 19)
            (6 15 19 21)
            (7 6 21 23)
    );
    }
);

mergePatchPairs
(
);

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

A.1.2 Mesh check and visualisation

In order to ascertain that the generated mesh conforms to the computational requirements an OpenFOAM utility called `checkMesh` can be invoked. The checkMesh.log file can be generated by simply using the below routine on the terminal

```
user@OpenFOAM:~ $ checkMesh > checkMesh.log
```

Listing A.2: checkMesh log file

```
/*---------------------------------------------------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  4.1                                   |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
Build  : 4.1
Exec   : checkMesh
Date   : Aug 22 2018
Time   : 13:55:22
Host   : "Akshay-PC"
PID    : 7146
Case   : /home/akshay/Desktop/test10na
nProcs : 1
sigFpe : Enabling floating point exception trapping (FOAM_SIGFPE).
fileModificationChecking : Monitoring run-time modified files using timeStampMaster
allowSystemOperations : Allowing user-supplied system call operations

// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //
Create time

Create polyMesh for time = 0

Time = 0

Mesh stats
    points:           7626132
    faces:            22440710
    internal faces:   22007290
    cells:            7408000
    faces per cell:   6
    boundary patches: 6
    point zones:      0
    face zones:       0
    cell zones:       0

Overall number of cells of each type:
    hexahedra:     7408000
    prisms:        0
    wedges:        0
    pyramids:      0
    tet wedges:    0
    tetrahedra:    0
    polyhedra:     0
```

```

Checking topology...
    Boundary definition OK.
    Cell to face addressing OK.
    Point usage OK.
    Upper triangular ordering OK.
    Face vertices OK.
    Number of regions: 1 (OK).

Checking patch topology for multiply connected surfaces...
    Patch                   Faces    Points   Surface topology
     inlet                  6750     6936     ok (non-closed singly connected)
    outlet                  6750     6936     ok (non-closed singly connected)
    atmosphere              57550    58752    ok (non-closed singly connected)
    breakwater              12750    13056    ok (non-closed singly connected)
    walls                   201460   202932   ok (non-closed singly connected)
    symmetry                148160   149532   ok (non-closed singly connected)

Checking geometry...
    Overall domain bounding box (0 0 0) (2.283 0.3 0.1)
    Mesh has 3 geometric (non-empty/wedge) directions (1 1 1)
    Mesh has 3 solution (non-empty) directions (1 1 1)
    Boundary openness (-6.41946e-19 -1.19114e-14 -8.80343e-13) OK.
    Max cell openness = 1.63091e-16 OK.
    Max aspect ratio = 1.44444 OK.
    Minimum face area = 3.6e-06. Maximum face area = 5.2e-06. Face area magnitudes OK.
    Min volume = 7.9902e-09. Max volume = 1.04e-08. Total volume = 0.065889. Cell volumes OK.
    Mesh non-orthogonality Max: 0 average: 0
    Non-orthogonality check OK.
    Face pyramids OK.
    Max skewness = 1.33227e-12 OK.
    Coupled point location match (average 0) OK.

Mesh OK.

End
```

After it is clear that the mesh conforms to the minimum requirements, a visual check can be done using Paraview. In order to visualize (import) the geometry into paraview the following procedure can be used.

```
user@OpenFOAM:~ $ touch foam.foam
user@OpenFOAM:~ $ paraview

                         OR

user@OpenFOAM:~ $ paraFoam
```

A.1.3 Setting boundary and initial conditions

Depending on the solver and turbulence model used, the boundary conditions can vary. The list of parameters which require specification in this particular case are listed in table A.1.

Table A.1: Boundary conditions required for `interFoam` solver using $k - \epsilon$ turbulence model with two phases

Parameter	**Description**
alpha.water	Definition of the water phase fraction boundary conditions inside the domain
alpha.air	Definition of the air phase fraction boundary conditions inside the domain
U	Velocity field boundary definitions
p_rgh	Pressure field boundary definitions
k	Turbulent kinetic energy boundary definitions
epsilon	Turbulent kinetic energy dissipation boundary definitions
nut	Turbulent viscosity boundary definitions

An example of the boundary condition file can be seen in the listing A.3.

Listing A.3: Boundary condition for velocity field

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  4.1                                   |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       volVectorField;
    object      U;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

// Including the initial conditions file in 0/ folder
#include        "include/initialConditions"

//Dimensions of the vector
dimensions      [0 1 -1 0 0 0 0];

// Setting all internal cells with 0 velocity
internalField   uniform (0 0 0);

boundaryField
{
    inlet
    {
        type            variableHeightFlowRateInletVelocity;
        flowRate        $inletFlowRate;
        alpha           alpha.water;
        value           uniform (0 0 0);
    }

    outlet
    {
        type            zeroGradient;
    }


    walls
    {
        type            noSlip;
    }

    breakwater
    {
        type            noSlip;
    }

    atmosphere
    {
        type            pressureInletOutletVelocity;
        value           uniform (0 0 0);
    }

    symmetry
    {
        type            symmetryPlane;
    }
}

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

The initial conditions can be set through the `setFields` utility. This file initializes a scalar or vector field inside the computational domain with the specified value in the `setFieldsDict`. In this particular case, the initial setting is done as described in listing A.4.

Listing A.4: `setFields` system file for initialization of initial conditions

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  4.1                                   |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    object      setFieldsDict;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

// Initialising the internal cells as all air phase fraction
defaultFieldValues
(
    volScalarFieldValue alpha.air 0
);

```

```
//Patch the water phase fraction region
regions
(
    boxToCell //create a box using the extents in (x y z) direction respectively
    {
        box (0 0 0) (1.63 0.17 0.2);

    // Specify the alpha.water phase inside the bounding box
        fieldValues
        (
            volScalarFieldValue alpha.water 1
        );
    }

    boxToCell
    {
        box (1.783 0 0) (2.283 0.064 0.2);

        fieldValues
        (
            volScalarFieldValue alpha.water 1
        );
    }

);

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

Using the above file, the initial conditions for this particular case can be seen in figure A.2. To visualize the geometry, the same procedure as used above holds.

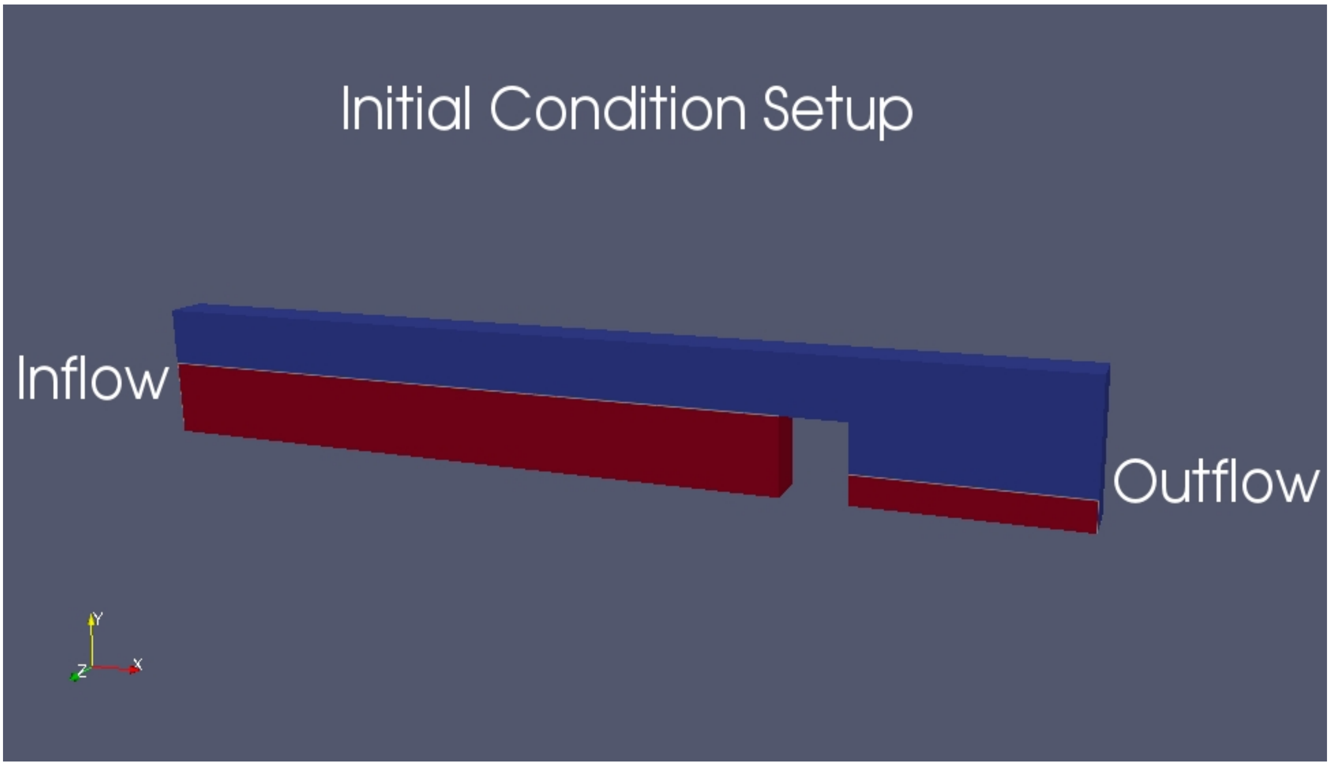


Figure A.2: Initial conditions generated using `setFields` file

Estimation of turbulent kinetic energy and dissipation rate

In order to improve convergence of the closure equations for k and ϵ, an estimation at the inlet for the two have to be provided. The values for each case differ depending on the inlet discharge ($Q[m^3/s]$) specified. The procedure to estimate the values for k and ϵ in this study has been provided in the following section.

Turbulent Intensity

The estimation for turbulent intensity (I) can be done using fully developed pipe flow analogy (*Turbulence intensity*). Using equation A.1, a first estimate for the turbulence intensity can be made. In this case the value for Re_{dh} is taken to be about 15000.

$$I \sim 0.16 Re_{dh}^{-1/8} \tag{A.1}$$

For most of the cases in this study the Reynolds number does not vary that much. As a result, the value for turbulent intensity has been considered to be the same for the two cases under consideration. Using this formulation the value for turbulent intensity $I \sim 0.16 \cdot (15000)^{-1/8} \sim 0.0481\%$. Using this information, the estimation for k and ϵ can be made as discussed below.

Estimation for k and ϵ

Using equation A.2 and A.3 the first estimate for turbulence related values in the numerical model can be estimated (Greenshields 2015). These equations are valid for isotropic turbulence. As the production in the buffer layer is isotropic this estimate holds in this case.

$$k = \frac{3}{2} \cdot (I|U_{ref}|)^2 \tag{A.2}$$

$$\epsilon = \frac{C_{\mu}^{0.75} k^{1.5}}{L_{ref}} \tag{A.3}$$

where, U_{ref} is the reference velocity $\sim \mathcal{O}(10)[m/s]$, C_{mu} is the model constant $= 0.09$, L_{ref} is the reference length scale $\sim 0.2[m]$. Using these values, the estimation for k and ϵ are provided in the table below.

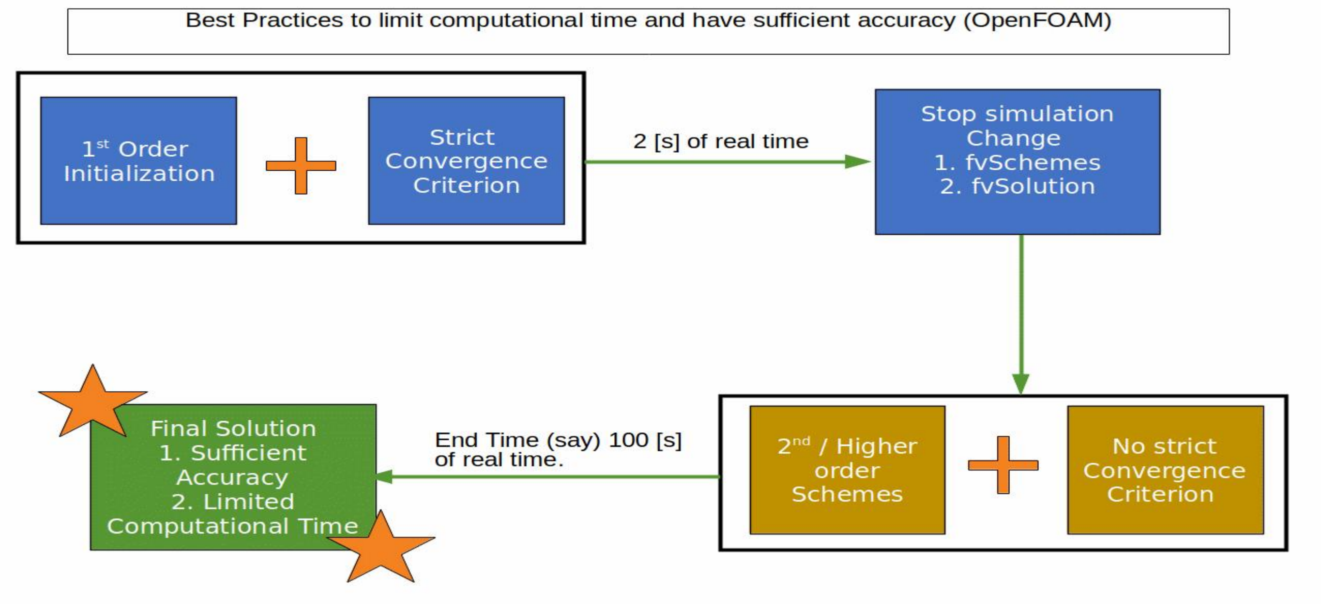
Table A.2: Turbulence properties initiated at the inlet face of the domain

Variable	Value
Turbulent Intensity (I)	0.0481 %
Turbulent Kinetic Energy (k)	0.347041 $[m^2/s^2]$
Turbulent Kinetic Energy Dissipation Rate (ϵ)	0.1679 $[m^2/s^3]$

A.1.4 Other pre-processing steps

In order to have sufficient numerical accuracy and limit computational resources the following procedure is recommended (see figure A.3). using this procedure, the numerical solutions can be optimized to a certain extent. The case details have been included in this sub section.

Figure A.3: Best practice to limit computational resources and have sufficient accuracy in the numerical simulation


Best Practices to limit computational time and have sufficient accuracy (OpenFOAM)
1st Order Initialization
+
Strict Convergence Criterion
2 [s] of real time
Stop simulation Change
1. fvSchemes
2. fvSolution
2nd / Higher order Schemes
+
No strict Convergence Criterion
End Time (say) 100 [s] of real time.
Final Solution
1. Sufficient Accuracy
2. Limited Computational Time

Discretisation schemes setup using fvSchemes

The discretisation schemes for various terms that are solved in the numerical model are described in the `fvSchemes` input file. The current simulation is listed as below.

Listing A.5: `fvSchemes` input file

```
/*---------------------------------*- C++ -*-----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version :  4.1                                  |
|   \\  /    A nd           | Web:       www.OpenFOAM.org                     |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    location    "system";
    object      fvSchemes;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

//The simulation is initialised using 1st order time marching scheme
//After a stable solution is obtained , a higher order scheme is employed
//Similar procedure is used for other discretisations
ddtSchemes
{
    default         Euler;
//     default     CrankNicolson 1;  //Higher order time discretisation
}

gradSchemes
{
    default         Gauss linear;
//   default       cellMDLimited leastSquares 1; // Diffusion limiting scheme
}

divSchemes //The comments below are the switches made to the computation after 2[s] of real time simulation
{
    div(rhoPhi,U)  Gauss linear;
    //div(rhoPhi,U) Gauss limitedLinearV 1;
    div(phi,alpha)  Gauss vanLeer;
    div(phirb,alpha) Gauss linear;
    //div(phirb,alpha) Gauss vanLeer;
    div(phi,k)       Gauss upwind;
    //div(phi,k)     Gauss linear;
    div(phi,epsilon) Gauss upwind;
    //div(phi,epsilon)  Gauss linear;
    div(((rho*nuEff)*dev2(T(grad(U))))) Gauss linear;
}

laplacianSchemes
{
    default         Gauss linear corrected;
}

interpolationSchemes
{
    default         linear;
}

snGradSchemes
{
    default         corrected;
}

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

Finite volume solution techniques using fvSolution

In the previous section the discretisation schemes were discussed. This section deals with the solution methods used to solve the system of equations. The fvSolutions file listing can be seen below.

Listing A.6: `fvSolution` input file

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  4.1                                   |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    object      fvSolution;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

//For initial 2 [s] of simulation
// 1. The convergence criterion is strict
// 2. The numerical schemes selected are low order (1st/2nd order)

//The initial convergence criteria may involve higher computational time ( initially )

solvers
{
    alpha.water
    {
        nAlphaCorr      1;
        nAlphaSubCycles 3; //CrankNicolson does not support sub-cycling procedure ( set = 1)
        cAlpha          1;
    }

    pcorr
    {
        solver          PCG; //Other solvers can be utilised for pressure equation
        preconditioner  DIC;
        tolerance       1e-10;
        relTol          0.1;
    }

    p_rgh
    {
        solver          PCG;
        preconditioner  DIC;
        tolerance       1e-07;
        relTol          0.05;
    }

// The iterative solver can be optimised by putting all the effort into the last iteration <parameter>Final
// As a result , (pcorr/p_rgh/U/k/epsilon)Final has a higher iteration
    pcorrFinal
    {
        $p_rgh;
    tolerance   1e-12; // Comment out this line after initial solution is obtained
        relTol          0.01; // Comment out this line after initial solution is obtained
    }

    p_rghFinal
    {
        $p_rgh;
    tolerance   1e-10; // Comment out this line after initial solution is obtained
        relTol          0.01; // Comment out this line after initial solution is obtained
    }

    "(U|k|epsilon)"
    {
        solver          smoothSolver;
        smoother        symGaussSeidel;
        tolerance       1e-8;
        relTol          0.1;
    }

    "(U|k|epsilon)Final"
    {
        $U;
    tolerance   1e-10;// Comment out this line after initial solution is obtained
        relTol          0.01;// Comment out this line after initial solution is obtained
    }
}

PIMPLE
{
    momentumPredictor no;
    nCorrectors      3;
    nNonOrthogonalCorrectors 1; //Minimum number of correctors is 1 for solution accuray
//     nOuterCorrectors   2; //  OuterCorrectors not needed in this case , however can be switched on depending on the solvers
}


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

Control of the simulation using controlDict file

The final listing provided is the controlDict file. This file controls the simulation read, write and other possible functions which are implemented in the run time.

Listing A.7: `controlDict` input file

```
/*--------------------------------*- C++ -*----------------------------------*\
| =========                 |                                                 |
| \\      /  F ield         | OpenFOAM: The Open Source CFD Toolbox           |
|  \\    /   O peration     | Version:  4.1                                   |
|   \\  /    A nd           | Web:      www.OpenFOAM.org                      |
|    \\/     M anipulation  |                                                 |
\*---------------------------------------------------------------------------*/
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    location    "system";
    object      controlDict;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

application     interFoam;  //OpenFOAM solver used

startFrom       latestTime; //This allows starting the simulation from the last time step written

startTime       0;

stopAt          endTime;

endTime         100;

deltaT          0.0005;     //Time step size

writeControl    adjustableRunTime; //This allows changes to be made while the simulation is running

 writeInterval   0.5;  //Dont forget to change the force output in force function below.

purgeWrite      0;

writeFormat      ascii ;  // File format of the data written

writePrecision  6;

writeCompression uncompressed;

timeFormat      general;

timePrecision   6;

runTimeModifiable yes;

adjustTimeStep on; //Dynamic updating of the time step to limit Courant number

maxCo            0.5;     //Flow Courant number limit
maxAlphaCo       0.4;

maxDeltaT        1;  //Maximum time step size

//Adding forces in the computation
libs
```

```
(
    "libforces.so"  //OpenFOAM library for computing forces
    "libutilityFunctionObjects.so"  //OpenFOAM library for other function objects
    "libsampling.so"     //OpenFOAM library for probes
);

functions
{
    FORCE
    {
        patches      (breakwater);   //Patch for which force is to be computed
        CofR         (178.3 0 10);   //Centre of rotation
        outputControl   timeStep;
        outputInterval   1;
        type           forces;
        pName        p;
        UName        U;
        rho       rhoInf;
        rhoInf        1000;   // Reference value for density
        log       true;
    }


//OpenFOAM only outputs the residuals for the values which are computed
    residuals
    {
        type residuals;
        enabled      true;
        writeControl     timeStep;
        writeInterval     1;
        fields
        (
            p_rgh
            k
            epsilon
        );
    }

//Probe for pressure measurement
    probe
    {
        type probes;
        name probe;
        fields
        (
            p    // Field   specification   which is to be logged
        );
        probeLocations
        (
            (1.783 0.027 0.1)    //Probe 1
            (1.783 0.075 0.1)    //Probe 2
            (1.783 0.14 0.1)     //Probe 3
        );
    }
};


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

Appendix B

Data analysis using BASH shell and Python scripts

A variety of different tools have been used to carry out the validation procedure. The most relevant tools which have been used are listed in the table below.

Table B.1: List of main tools along with the corresponding function

Tools Used	Purpose
Paraview	Extracting data blocks of interest and visualizations
BASH shell	Modification of log files
Python scripting	Data analysis, computations, visualizations
Gnuplot	Plotting log files

B.0.1 Preparing the log files

The fastest way to prepare the log files for post-processing is using BASH shell tools. A sample log file for the force can be seen in the listing below.

Listing B.1: forces.dat log file generated

```
# Forces
# CofR       : (1.783000e+02 0.000000e+00 1.000000e+01)
# Time [s]       forces(pressure viscous porous) [N]
0.00445106       ((9.862425e+00 6.251166e-02 0.000000e+00) (2.280809e-06 -2.272840e-06 -1.471035e-07) (0.000000e+00 0.000000e+00 0.000000e+00))
0.0154435        ((9.862421e+00 6.251349e-02 0.000000e+00) (9.220468e-07 -9.026096e-07 -4.750481e-08) (0.000000e+00 0.000000e+00 0.000000e+00))
0.0397389        ((9.862418e+00 6.251356e-02 0.000000e+00) (1.774547e-07 -2.697144e-07 2.199168e-08) (0.000000e+00 0.000000e+00 0.000000e+00))
0.0768997        ((9.862410e+00 6.251231e-02 0.000000e+00) (2.353187e-07 -2.023420e-07 1.407941e-07) (0.000000e+00 0.000000e+00 0.000000e+00))

Time      | pressure [x]   | pressure [y]   | pressure [z]   | viscous [x]   |   viscous [y]   |   viscous [z]   |   POROUS
```

Looking at the above log file, it is clear that the only force which is of any interest is the pressure force in the x direction. Carrying out file manipulations in python can be quite difficult depending on the analysis library used. In order to supply python a clean set of files for analysis, a BASH shell routine as seen in the block below can be utilized.

```
cat forces.dat | cut -d "(" -f 3 | grep [0..9] | cut -d " " -f 1 > force

cat forces.dat | cut -d "(" -f 1 | grep [0..9] > time
```

The above lines of bash shell code supply two files viz. `force` and `time`. Using `python`, the trend plot can be easily generated using the two files. Similar shell routine can be used to dissect the `probe` log file (which contains the pressure value at a certain [x,y,z] co-ordinate).

B.0.2 Gnuplot script to plot the log files

Gnuplot provides the fastest way to generate plots without having to leave the terminal window (Unless python scripts are invoked from the terminal). In this section the Gnuplot script will be used to produce simple plots for log files. The listing below (B.2) represents a simple script to generate plots. The bash script can be converted into an executable script using the following command in the terminal window.

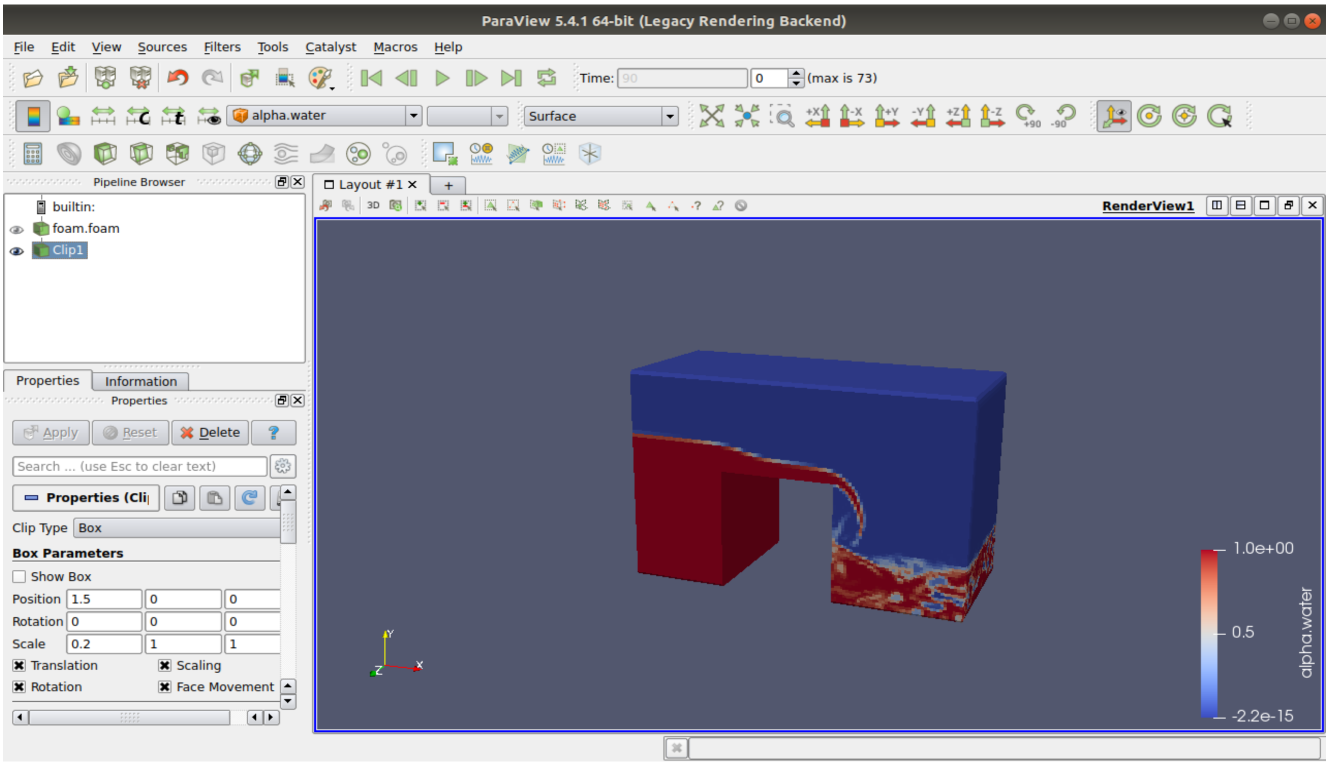
```
chmod +x filename
```

Listing B.2: Gnuplot script for generating plots for log files

```
#!/usr/bin/gnuplot   #This line is the ’BANG’ line which loads the gnuplot program



#Set terminal type as PNG with the image size 1080 x 1080 pixels
set terminal png size 1080,1080

#Set the name of the output file
set output ’ForceTimeSeries’

#Set the x and y labels
set xlabel ’Time [s]’
set ylabel ’Force [N]’

#Set the title of the plot
set title ’Force on the Caisson’

#Set the x and y tics
set ytics 5
set xtics 5

#Set the grid
set grid

#Main line which plots the data/log files
plot ’<name of the file>’ with lines
```

B.0.3 Extracting data block using `Paraview`

In order to carry out the nappe validation, only the downstream end of the caisson is required. As a result, only a small block of the entire domain is to be imported into python. A simple way to do this data is to extract only a small part of the data block using paraview. Paraview allows the option of saving the data in a `.csv` format. This makes it easier to carry out the data analysis. Figure B.1 shows one way of extracting the block in paraview. The extent of the geometry can be changed based on the requirement of the data.


ParaView 5.4.1 64-bit (Legacy Rendering Backend)
File Edit View Sources Filters Tools Catalyst Macros Help
Pipeline Browser
Properties Information
Apply Reset Delete ?
Search ... (use Esc to clear text)
Layout #1
RenderView1
3D
Y
X
Z
1.0e+00
0.5
-2.2e-15
alpha.water

Time: 90 | 0 (max is 73)

alpha.water | Surface

builtin:
foam.foam
Clip1

Properties (Clip1)

Clip Type Box

Box Parameters

□ Show Box

Position	1.5	0	0
Rotation	0	0	0
Scale	0.2	1	1

☒ Translation ☒ Scaling

☒ Rotation ☒ Face Movement

Figure B.1: Clipping the data block into smaller data block for analysis

B.0.4 Python validation routine (script)

Once the planar data is extracted from paraview and exported to .csv file format, the following script can be used to generate the data validation plots. It is important to tweak a few parameters for certain data comparison cases. Similar scripts have been used to generate other plots. For sake of brevity the scripts have not been provided here.

Listing B.3: Nappe validation script

```
#Before the script is initialised , it is important to have a look at the data file
#This overview can help to write a cleaner script without utilising too much memory

#Importing the required libraries
import numpy as np                  #Numpy library used for computations
import matplotlib.pyplot as plt        # Matplotlib library used to produce plots
import pandas as pd                  # Pandas library to support data files in .csv format

#Importing the Pressure file
Aerated_Nappe = pd.read_csv('data/t10_a_im_c0.csv')
Non_Aerated_nappe = pd.read_csv('data/t10_na_im_c0.csv')

#Slicing the data file
#Setting the basic data variables as present in the data file structure

#Aerated Nappe Case
x_Aerated = Aerated_Nappe['Points:0']
y_Aerated = Aerated_Nappe['Points:1']
# p_Aerated = Aerated_Nappe['p']
water_Aerated = Aerated_Nappe['alpha.water']

#Non-Aerated Nappe Case
x_Non_Aerated = Non_Aerated_nappe['Points:0']
y_Non_Aerated = Non_Aerated_nappe['Points:1']
# p_Non_Aerated = Non_Aerated_nappe['p']
water_Non_Aerated = Non_Aerated_nappe['alpha.water']


#Saving the point only where the alpha value is > 0.9
```

```

#Aerated Case

# #Setting up empty lists
X_a = []
Y_a = []

#Looping through all the points in the data
for i in range(len(x_Aerated)):
    if (water_Aerated[i] > 0.9 ):    #Check if alpha.water > 0.9
#If true save x and y co-ordinates
        X_a.append(x_Aerated[i])
        Y_a.append(y_Aerated[i])


#Non-Aerated Case

#Setting up empty lists
X_na = []
Y_na = []

#Looping through all the points in the data
for i in range(len(x_Non_Aerated)):
    if (water_Non_Aerated[i] > 0.9):    #Check if alpha.water > 0.9
#If true save x and y co-ordinates
        X_na.append(x_Non_Aerated[i])
        Y_na.append(y_Non_Aerated[i])


#Experimental Data

#TEST_10
#Aerated Case Data
X_A_data = np.array ([..........])    #Input the data separated with a ','
Y_A_data = -np.array ([..........])   #Input the data separated with a ','


#Non-Aerated Case Data

X_NA_data = np.array ([.........])    #Input the data separated with a ','
Y_NA_data = -np.array ([..........])  #Input the data separated with a ','

#Plotting the results

plt.subplots(2,2, figsize =(20,10))
plt.suptitle(”Comparison of nappe trajectories for Test 10 (Centre of the flume)\nWater = phase fraction > 0.9”,fontsize=20)
plt.subplot(121)

#Aerated Nappe

plt.plot(X_dim_a,Y_dim_a,'o',label=”Numerical simulation”)
plt.plot(X_A_data,Y_A_data,'r*',markersize=10,label=”Experimental data”)
plt.title(”Comparison of Aerated Nappe”)
plt.ylabel(”Vertical distance from the caisson [cm]”,fontsize=15)
plt.xlabel(”Horizontal distance from the Caisson [cm]”,fontsize=15)
plt.legend()
plt.grid();

plt.subplot(122)

#Non-Aerated Nappe

plt.plot(X_dim_na,Y_dim_na,'o',label=”Numerical simulation”)
plt.plot(X_NA_data,Y_NA_data,'r*',markersize=10,label=”Experimental data”)
plt.title(”Comparison of Non-Aerated Nappe”)
plt.ylabel(”Vertical distance from the caisson [cm]”,fontsize=15)
plt.xlabel(”Horizontal distance from the Caisson [cm]”,fontsize=15)
plt.legend()
plt.grid();

plt.savefig(”NAME”,dpi=300)  #Save the figure with NAME and choose the resolution by changing dpi setting
```