A Consistent Numerical Method for Simulating Interfacial Turbulent Flows

by

Hanif Montazeri

A thesis submitted in conformity with the requirements for the degree of Doctor of Philosophy
Mechanical and Industrial Engineering
University of Toronto

© Copyright by Hanif Montazeri 2010
Abstract

A mathematically consistent algorithm for simulating interfacial turbulent flows is devised in this work. To minimize numerical errors for imposing dynamic boundary conditions at the interface locations, piezometric pressure is used to limit the effect of gravity forces in a flow field. Consequently, suitable and consistent numerical schemes are designed to accurately implement the new forms of interfacial forces. The proposed numerical methods are challenged for low Froude number flows which tend to trouble conventional algorithms. To capture the effect of turbulence on the interface, standard large eddy simulation techniques are reviewed and discussed. It is shown the standard filtered flow equations encounter numerical and mathematical inconsistencies. To remedy the irregularities of the conventional methods, a new framework for large eddy simulations is grounded. Purely mathematical models are derived and correlated with the conventionally more physical models. Semi implicit SIMPLE method is used to discretize the final flow equations. Taking advantage of the implicit feature of SIMPLE algorithm, an error correction technique is devised by which numerical cost of a turbulent simulation is substantially reduced. The entire framework is finally discussed toward simulating a turbulent interfacial flow.
Acknowledgments

Rumi’s book, Masnavi Manavi, was my very first companion when I came to Toronto to start my PhD program. Through his bright philosophical thoughts, I unconsciously began to study a mysterious path of life. At the time, Rumi’s heavenly world was nothing but an unachievable dream. I started my PhD project. Initially, it seemed straightforward, yet turned out to be far from it. The more I was working on it, the less I was obtaining results. The project was deliberately acting as a sharp knife to hollow my being shaping it as a cup. The cup was getting deeper and deeper everyday. It was only in the fourth year of the program that my mother with the careful guidance of her friend served the very first drops of the joyous divine wine in that very sore cup. This was what I had learnt four years earlier in Rumi’s book. The relief and joy could not be possible without the considerable sacrifices and hard work of these two ladies. From then, through the wisdom of my competent mother, I received so much blessing from many of my science and philosophy heroes: Ave Sina, Pythagoras, Archimedes, Nezami, Plato, Shakespeare, Hafez, Galileo Galilei, Ferdosi, Baba Taher, Mansur Hallaj, Khayam, Einstein, Mulla Sadra, Rhazes and Sadi. These heroes and some wonderful others left me such valuable heritages beyond my description.

In all these times, Professor J. Mostaghimi was the one who was patiently supervising this work. His supervision along with his smart comments and ideas indeed enriched the thesis. His ongoing support made this work possible to be concluded. Part of the work was cooperation with Dr. A. Pourmousa by which I developed Chapter 3. Similarly, some of the casual conversations with my colleague, A. Kashani, brought me some new ideas that improved some parts of the thesis. My favorite chapter of the thesis, Chapter 5, was carefully studied by Professor S. Madnia. Professor M. Bussmann helped me revising and improving Chapter 2 and 3. Professors P. E. Sullivan and S. Argyropoulos spent time reading and providing me constructive comments to improve the coherence of the thesis. I would like to show my appreciation to all the aforementioned individuals who have made great contributions to the quality of the work.

I would also like to express my sincere gratitude to my father, Dr. M. Montazeri, with his invaluable support, my adorable brothers, Ali, Majid and Vahid, and amazing family and friends.
I owe my deepest gratitude to my life and philosophy mentors, my mother, E. Ebrahim, and her brilliant friend.

Finally, it is difficult to forget mentioning enthusiastic teachers and professors who have contributed so much in the path of learning science. My third grade teacher Mr. Kazemian, my passionate math teachers Mr. Pezeshki and Mr. Ghaheri and my unexceptionally knowledgeable professor Dr. Noseir. I should also name Dr. S. Kazemzadeh, as he made available his kind support in a number of ways in finalizing and supervising my Master program.

Borrowing the symbolic style of Plato and Hafez, I start this thesis in the very same way my science and philosophy idols institute their work: “In the name of God the creature of the skies, earth, moon, sun and the entire universe”. 
# Table of Contents

Chapter 1: Introduction ........................................................................................................................................................................... 1

References ........................................................................................................................................................................................................... 3

Part I: Two-Phase Flow Modeling .......................................................................................................................................................... 6

Chapter 2: Consistent Mathematical and Numerical Description of Multiphase Flows ......................................................... 7

2.1. Introduction ........................................................................................................................................................................................................... 7

2.2. Navier-Stokes Equations for Two-Phase Flow ................................................................................................................................. 9

2.3. Estimating Surface Tension and Gravity Force .......................................................................................................................... 14

2.3.1. Surface Tension Force ............................................................................................................................................................... 15

2.3.2. Interfacial Gravity Force ............................................................................................................................................................... 18

2.4. Discretization of Pressure Gradient in Conjunction with Forcing Terms ...................................................................................... 21

2.5. Pressure-Velocity Coupling .............................................................................................................................................................. 25

2.6. The Level-Set Method (LS) ............................................................................................................................................................. 26

2.7. Outline of the numerical algorithm .................................................................................................................................................. 33

2.8. Results ........................................................................................................................................................................................................... 33

2.8.1. Static drop in equilibrium ...................................................................................................................................................... 34

2.8.2. Column of two phase flow ...................................................................................................................................................... 35

2.8.3. Rayleigh-Taylor Instability ...................................................................................................................................................... 41

2.8.4. Falling Water Droplet ...................................................................................................................................................... 45

References ........................................................................................................................................................................................................... 47

Appendix II-A ......................................................................................................................................................................................................... 51

Chapter 3: Sharp Implementation of Interfacial Forces ...................................................................................................................... 54

3.1. Introduction ........................................................................................................................................................................................................... 54

3.2. Governing Equations .............................................................................................................................................................. 56
3.3. Different methods for estimating sharp interfacial forces ........................................ 59
   3.3.1. Stair-step method .................................................................................................. 59
   3.3.2. Areal method ........................................................................................................ 66
3.4. Explanation for spurious currents for areal method .............................................. 74
   3.4.1. Numerical experiments ......................................................................................... 74
   3.4.2. Mechanical interpretation ..................................................................................... 78
3.5. Discussion and solutions ............................................................................................ 83
Appendix III-A ......................................................................................................................... 88
References ............................................................................................................................. 90

Part II: Turbulent Flows ............................................................................................................. 93
Chapter 4: Implicit Direct and Large Eddy Simulations of Turbulent Flows .................... 94
   4.1. Introduction .................................................................................................................. 94
   4.2. Governing equations .................................................................................................. 98
      4.2.1. Standard Filtered Flow Equations ....................................................................... 98
      4.2.2. Different Models .................................................................................................. 101
   4.3. Channel flow simulation ............................................................................................ 106
      4.3.1. Periodic boundary condition .............................................................................. 106
      4.3.2. Simulation results ................................................................................................. 108
      4.3.3. More accurate results .......................................................................................... 121
References ................................................................................................................................ 124
Chapter 5: A Consistent Large Eddy Simulation Formulation .................................. 129
   5.1. Introduction .................................................................................................................. 129
   5.2. Formal Filtering of Flow Equations .......................................................................... 130
   5.3. Filtering versus Resolving of an Equation ................................................................. 134
   5.4. Numerical experiments for “perfect LES” ................................................................. 146
5.5. Methods and Assumptions for Conventional LES .................................................. 150

5.6. New Perspective for Large Eddy Simulations ......................................................... 154

5.7. Modeling the Correction Error $C_1$ .............................................................................. 157

References ........................................................................................................................... 162

Part III: Turbulent Two-Phase Flows ........................................................................ 165

Chapter 6: Interfacial Turbulent Flow ........................................................................ 166

References ........................................................................................................................... 173

Chapter 7: Closure ....................................................................................................... 177
Chapter 1

Introduction

Simulation of interfacial turbulent flow is of great engineering interest. In addition to chaotic flow state, the existence of a distorted interface is the heart of difficulty in these forms of simulations. Although numerical capturing of interface is relatively well documented, simulation of single phase turbulent flows is still an ambiguous subject. Therefore, the existence of the second phase and its interactions through their interface worsens the difficulty for these types of simulations. These barriers have caused significant difficulties in the progress of this research field. The majority of research has focused on LES or DNS of two phase flows with small interfacial disturbances [4][6][7][9][10][11][12][13][14][17]. To author’s knowledge, there are few papers which aim to find suitable models for interfacial flows [7][13][18]. Inadequate experimental results and therefore insufficient insight of different turbulent length scales and time scales confine researchers to develop models for these types of flows. To the knowledge of author, the only modeling used in LES simulations is reducing the turbulent viscosity close to interface [11]. These viscosity modifications are basically the same wall functions which are conventionally used in RANS modeling. It is believed that the lighter fluid sees the heavier fluid as a solid wall. Perhaps this is the only modeling in large eddy simulation category. Some authors claim that LES simulation captures enough flow structure which removes the need of any further modeling or the current numerical schemes produce higher errors than subgrid interfacial terms [8]. This claim however is not supported. Furthermore, most papers in this class focus on low Reynolds number and consequently relatively small interface surfaces disturbances are involved.

As an alternative approach, RANS modeling allows less expensive simulations and usually less accurate results. This class of modeling, nevertheless, seems to be the only practical method to consider for high Reynolds numbers. By its definition, RANS, though, needs accurate modeling. Clearly, these models require deep understanding of turbulent structures and their interaction with the interface in the vicinity of boundaries. Provided enough experimental results,
an analytical analysis would be of great help to enlighten some of the ambiguity for the modeling of the flow regime [5][15][16][19]. Considering all these facts, RANS techniques have been rarely used for the simulation of interface. Shirani et al. [18] introduced the very first models for this group of simulations. They used Taylor length and time scales as the characteristic turbulent length and time scales for their representation of the flow behavior. Lack of detailed experimental results and also introducing many parameters in their models disallow them for any generalizations for their model.

In a nutshell, simulation of interfacial flows needs inexpensive methods and effective models. Inexpensive simulations means nothing except using RANS or at the most LES methods; and effective models can only be obtained by deep physical insights. In most above cited simulations, there are concerns regarding the effectiveness and accuracy of numerical methods used to simulate interfacial turbulent flows. For instance, existing of spurious currents due to surface tension or gravity force received little attention in most works. Also, close comparison between experimental results and numerical results is also a missing part in most of the above papers. In addition, modeling subgrid scales of turbulent interface is a crucial subject which needs to be extensively investigated.

A series of experimental papers by Walker et al. [20][21], Hong et al. [8] and Brocchini et al. [1][2][3] constitute our physical understanding of the phenomenon. Each of these series of papers aims to classify and establish an appropriate set of analytical equations by which a numerical method can be launched. Even though they have different perspectives for their methodology, they provide a complementary and profound understanding of different observations. Surface currents, turbulent kinetic energy and stress anisotropy are compared and presented for different Froude and Reynolds numbers in [8][20][21]. Brocchini et al. papers [1][2][3], on the other hand, focus on classifying different regimes of interface interactions. Introducing new time and length scales for different regimes, predicting flow state based on these turbulent scales as well as analytical formulation for these states are the main features of these series of papers.

In this research, inspired by these six papers, we first develop a consistent and accurate numerical framework for these types of simulations. The numerical framework has to have
several characteristics, in order to be useful for these types of applications. As mentioned earlier, existence of spurious currents at interface locations can considerably deteriorate accuracy of simulation, especially for low Froude numbers. Therefore, we devised a numerical algorithm which results in the least spurious currents. Having used SIMPLE algorithm, we have no concern regarding intermediate boundary conditions which have to be determined in two-step projection methods. Moreover, concerns regarding first-order accuracy in time for pressure due to the splitting of the convection-diffusion and projection steps have no room in SIMPLE type algorithm. We then focus on turbulent aspects of the simulations. We examined our numerical algorithm for direct numerical simulation of single phase to be sure of its accuracy and conservation properties. Then, several large eddy simulation, LES, models are implemented and tested for single phase flow. We also showed our implicit discretization scheme used in our SIMPLE algorithm allows us to increase the time steps for LES simulation. Having tested our computer code for standard LES, a new consistent framework for large eddy simulation is proposed. It is mathematically proven that the standard LES framework encounters numerical, mathematical and physical inconsistencies. In the consistent large eddy simulation, CLES, framework, it is our goal to avoid all these known inconsistencies.

This work is presented in three parts. In the first part, numerical techniques for accurate implementation of interfacial forces for two-phase flows are derived, implemented and tested. The second part specifically deals with turbulent simulations for single phase flows. In this part, first the implicit time integration feature of SIMPLE algorithm is challenged for its accuracy using the standard LES framework; then a detailed discussion on inconsistencies in the conventional LES framework is presented. Subsequently, the consistent large eddy simulation, CLES, is introduced. Finally, in the third part, insightful experimental and theoretical information and data are gathered and discussed toward the simulation of turbulent two-phase flows. Note mentioning, each of these parts are divided to chapters and each chapter is intended to be self-contained.

References


Part I

Two-Phase Flow Modeling
Chapter 2

Consistent Mathematical and Numerical Description of Multiphase Flows

2.1. Introduction

Mathematical description of the interface between two different fluids dates back to early nineteen century. However, accurate numerical modeling of such analytical descriptions has been challenging because of the discontinuity in material properties across the interface. Many different numerical schemes have been developed [1][4][10][13][16][19][41] to deal with this problem. One perhaps can distinguish two types of methods for including the effect of the discontinuity in material properties. The first group uses explicit forcing terms in multiphase flow equations to capture the effects of the discontinuity [4][10]; and the second group avoids any forcing term. The immersed interface method [21][23][24], the sharp interface method [29][46][47][48], the ghost fluid method [8][17][27] and the XFEM method [5][6][29] can be recognized as examples of this class. In these types of implementation, therefore, differential operators are defined so that they accommodate the interfacial jump conditions.

Both of these groups aimed to accurately capture interfacial effects. One of the problematic issues in including jump conditions is the development of spurious currents around the interface. Limited to special circumstances and conditions, Ghost Fluid Method (GFM) [40][44] and balanced force algorithm [10] eliminate such undesirable currents. GFM is based on an accurate mathematical modification of differential operators and it eliminates spurious currents if the curvature of the interface is assigned. Whereas the balanced force algorithm uses a forcing term in its formulation. Similar to GFM, it also eliminates spurious currents if the curvature is given. Unlike GFM, no clear mathematical or mechanical proof is provided for the accurate performance of the balanced force algorithm. The balanced force algorithm is basically architected within the framework of fractional step methods of solving Navier-Stokes equations and is suitable for both sharp and continuous implementation of surface tension force, SSF and CSF, respectively. In the algorithm, Volume-of-fluid (VOF) is the color function which
identifies different phases. Herrmann [13] used the level-set function instead of VOF method and accordingly modified the balanced-force algorithm. On the other hand, within the pressure-based methods (e.g. SIMPLE algorithm), the concept of staggered formulation [11] closely follows the concept of the balanced-force algorithm; however it fails to perfectly balance the pressure gradient and surface tension force.

A wide variety of flow problems involving incompressible fluid flow have been solved by pressure-based finite-volume techniques. The earliest finite-volume methods were formulated for sequential pressure-based solution algorithms, such as SIMPLE [34]. The robustness and efficiency of this class of techniques paved the path to expand the finite-volume methods to body-fitted and block-structured meshes [18][28][44]. Further developments for unstructured mesh [2][7][30][39], collocated and equal-order pressure-velocity storage[15][39][43], let these methods to be deployed in commercial software; and spread its use by industrial analysts and designers. Progress for the SIMPLE algorithm and its variants continued by introducing multigrid techniques to accelerate the convergence and expanding their use for compressible flows, see [22] [31] for example. An interested reader can find an extensive review of pressure-based methods in [1].

One of the distinctive features of SIMPLE-type algorithms is their capabilities to be used for both steady and unsteady state flows. This can be a useful feature for developing a generic computer program which is not limited to only unsteady flows. Even though a steady-state solution can be sought by allowing the computer code to reach a stationary solution, the fractional step method might not be an efficient method for this purpose. Besides, unlike fractional step methods, these types of solvers are not constrained by the size of time-step. As shown in [9][38][49] if the time step is too large, the fractional step method produces an error caused by operator splitting. Moreover, concerns regarding first-order accuracy in time for pressure due to the splitting of the convection-diffusion and projection steps [38][49], have no room in the pressure-based algorithms.

In this chapter, we present an accurate numerical technique for collocated variable arrangement within the framework of pressure-based flow algorithms. Since we use forcing term to include jump conditions at the interface, we derive suitable numerical scheme for implementing a generic forcing term within the collocated variable arrangement. We present a
proof for our technique by which the advantages and disadvantages of the method can be established (e.g. its application for non-uniform mesh). We show the balanced force algorithm is a specific form of our formulation. Considering the difficulties in implementing of forcing term, following [14], we incorporated the concept of piezometric pressure into the analytical formulation for two-phase flows. As a result, in addition to surface tension forces, gravity forces, which are transferred to interfacial locations, also constitute interfacial forces. Consequently suitable numerical schemes are developed in order to accurately implement the new interfacial forces within the arrangement of pressure-based flow algorithms (SIMPLE-type algorithms). We then validate and illustrate the advantages of the new methods through several test cases. Although we used pressure-based algorithms as the framework in this work, our numerical schemes and techniques are readily expandable for other frameworks such as fractional step methods.

The structure of the chapter is as follows. In Section 2.2, we construct implicit Navier-Stokes equations for two-phase flows in which gravity is included in the interfacial nodes. Besides, the flow equations are formatted to be consistent when level-set function is used. In Section 2.3 we describe the suitable strategies for applying surface tension and gravity forces. Then, proper implementation of the interfacial forces and pressure-velocity coupling schemes are presented in Section 2.4 and 2.5, respectively. In Section 2.6 the implementation of the level-set function and the method of reinitialization of level-set function are described; and finally we present different applications of interfacial forces in Section 2.8.

2.2. Navier-Stokes Equations for Two-Phase Flow

In this section, implicit governing equations for two-phase flows based on single-phase Navier-Stokes equations will be developed. Suitable governing equations for a two-phase flow are the Navier-Stokes equations written for variable fluid properties and applied across the entire flow domain including the interface. To simplify numerical implementation, it is common to combine these single-phase equations so that an implicit set of equations can be replaced with two separate sets of equations. Implicit equations for multiphase flows can be derived by different approaches. In this paper we develop our formulations following the method presented in [14].

Conservation of mass and momentum equations for non-constant property fluids are:
\[
\frac{\partial \rho}{\partial t} + \frac{\partial}{\partial x_i} (\rho U_i) = 0
\]  
(2-2-1)

\[
\frac{\partial}{\partial t} (\rho U_i) + \frac{\partial}{\partial x_j} (\rho U_j U_i) = -\frac{\partial \rho}{\partial x_i} + \frac{\partial}{\partial x_j} \left[ \mu \frac{\partial U_i}{\partial x_j} + \frac{\partial U_j}{\partial x_i} \right] + \rho g_i
\]  
(2-2-2)

where \( U_i, \mu, \rho \) and \( g_i \) represent velocity components, viscosity, density and gravity components, respectively.

A distance function, known as level-set function, can be defined on the entire domain so the zero level contour separates the two fluids:

\[
\phi(x_i, t) = 0
\]  
(2-2-3)

while negative and positive contours correspond to fluid 0 and fluid 1, respectively. A Heaviside function, \( \delta(x_i, t) \), is defined to keep track of the fluid properties at the point of interest, such as:

\[
\delta(x_i, t) = \begin{cases} 
1 & \text{if } \phi(x_i, t) < 0 \\
0 & \text{if } \phi(x_i, t) > 0
\end{cases}
\]  
(2-2-4)

Using chain rule, its spatial and time derivatives reads as:

\[
\frac{\partial \delta}{\partial x_i} = \frac{\partial \phi}{\partial x_i} \delta(\phi)
\]  
(2-2-5)

\[
\frac{\partial \delta}{\partial t} = \frac{\partial \phi}{\partial t} \delta(\phi)
\]  
(2-2-6)

where \( \delta \) is the Dirac delta function. Complimentary indicator function \( I_1 \), is defined as

\[
I_1 = 1 - I_0
\]  
(2-2-7)

by which the second fluid domain can be represented.

The two Heaviside functions, \( I_0 \) and \( I_1 \), allow us to explicitly separate different fluid domains for mass and momentum equations. This can be done by explicitly implementing the density, viscosity and pressure of each phase in the single phase flow equations. All properties and indicator functions which are indexed with 0 correspond to fluid 0 and the same holds for fluid 1. Inserting densities for the two fluids in continuity equation:
Taking derivatives and rearranging indicator functions and their derivatives; yields:

\[
\left[ \rho_0 \frac{\partial u_i}{\partial x_i} I_0 + \left( \rho_1 - \rho_0 \right) \left( \frac{\partial \phi}{\partial t} + U_i \frac{\partial \phi}{\partial x_i} \right) \right] = 0
\]  

(2-2-9)

In which the two indicator functions and the Dirac delta function establish three equations which apply in the two fluids and at their interface, respectively. Basically, the indicator functions and Dirac delta function divide the flow domain into three regions: fluid 1, fluid 0 and interface. In other words, in this formulation, there are three indicator functions which facilitate the derivation of all necessary equations for a two-phase flow. The first two brackets in Equation (2-2-9) are continuity equations for each of the fluids, shown in Equation (2-2-10), and the third bracket is the kinematic boundary condition for the interface. This kinematic boundary condition constitutes an equation for zero level-set function \( \phi(x_i, t) = 0 \) by which the interface can be tracked in time, Equation (2-2-11).

\[
\frac{\partial u_i}{\partial x_i} = 0
\]  

(2-2-10)

\[
\frac{\partial \phi}{\partial t} + U_i \frac{\partial \phi}{\partial x_i} = 0
\]  

(2-2-11)

Similarly, momentum equations can produce three sets of equations if fluid properties are explicitly substituted with their two-phase flow expressions:

\[
\frac{\partial}{\partial t} \left[ (\rho_0 I_0 + \rho_1 I_1) U_i \right] + \frac{\partial}{\partial x_j} \left[ (\rho_0 I_0 + \rho_1 I_1) U_j U_i \right] = -\frac{\partial}{\partial x_i} (p_0 I_0 + p_1 I_1) + \frac{\partial}{\partial x_j} \left[ (\mu_0 I_0 + \mu_1 I_1) \left( \frac{\partial U_i}{\partial x_j} + \frac{\partial U_j}{\partial x_i} \right) \right] + (\rho_0 I_0 + \rho_1 I_1) g_i
\]  

(2-2-12)

Taking derivatives and rearranging all terms:
\[
\left[ \frac{\partial}{\partial t} (\rho_0 U_i) + \frac{\partial}{\partial x_j} (\rho_0 U_j U_i) \right] I_0 + \left[ \frac{\partial}{\partial t} (\rho_1 U_i) + \frac{\partial}{\partial x_j} (\rho_1 U_j U_i) \right] I_1
= \left[ -\frac{\partial p}{\partial x_i} + \frac{\partial}{\partial x_j} \left[ \mu_0 \left( \frac{\partial U_i}{\partial x_j} + \frac{\partial U_j}{\partial x_i} \right) \right] + \rho g_i \right] I_0
+ \left[ -\frac{\partial p}{\partial x_i} + \frac{\partial}{\partial x_j} \left[ \mu_1 \left( \frac{\partial U_i}{\partial x_j} + \frac{\partial U_j}{\partial x_i} \right) \right] + \rho g_i \right] I_1
+ \left[ \mu_0 \left( \frac{\partial U_i}{\partial x_j} + \frac{\partial U_j}{\partial x_i} \right) - p_0 \delta_{ij} + p_1 \delta_{ij} - \mu_1 \left( \frac{\partial U_i}{\partial x_j} + \frac{\partial U_j}{\partial x_i} \right) \right] \frac{\partial \phi}{\partial x_j} \delta(\phi)
\]
(2-2-13)

or

\[
\left[ \frac{\partial}{\partial t} (\rho_0 U_i) + \frac{\partial}{\partial x_j} (\rho_0 U_j U_i) \right] I_0 + \left[ \frac{\partial}{\partial t} (\rho_1 U_i) + \frac{\partial}{\partial x_j} (\rho_1 U_j U_i) \right] I_1
= \left[ -\frac{\partial p}{\partial x_i} + \frac{\partial}{\partial x_j} \left[ \mu_0 \left( \frac{\partial U_i}{\partial x_j} + \frac{\partial U_j}{\partial x_i} \right) \right] + \rho g_i \right] I_0
+ \left[ -\frac{\partial p}{\partial x_i} + \frac{\partial}{\partial x_j} \left[ \mu_1 \left( \frac{\partial U_i}{\partial x_j} + \frac{\partial U_j}{\partial x_i} \right) \right] + \rho g_i \right] I_1
+ \left[ \mu_0 \left( \frac{\partial U_i}{\partial x_j} + \frac{\partial U_j}{\partial x_i} \right) - p_0 \delta_{ij} + p_1 \delta_{ij} - \mu_1 \left( \frac{\partial U_i}{\partial x_j} + \frac{\partial U_j}{\partial x_i} \right) \right] \frac{\partial \phi}{\partial x_j} \delta(\phi)
\]
(2-2-14)

where $\delta_{ij}$ is the Kronecker delta. Again the three indicator functions: $I_0$, $I_1$ and $\delta$ appeared in Equation (2-2-14) providing momentum equations for each phase and their dynamic boundary condition at their interface. Although Equations (2-2-13) and (2-2-14) are mathematically identical, as it will be shown, Equation (2-2-14) produces inaccurate results when it is discretized numerically. This will be discussed in Section 2.3.

Equation (2-2-13) can be decomposed into a set of momentum equations for fluid 0 and 1:

\[
\frac{\partial}{\partial t} (\rho U_i) + \frac{\partial}{\partial x_j} (\rho U_j U_i) = -\frac{\partial p}{\partial x_i} + \frac{\partial}{\partial x_j} \left[ \mu \left( \frac{\partial U_i}{\partial x_j} + \frac{\partial U_j}{\partial x_i} \right) \right] + \rho g_i
\]
(2-2-15)
or
\[
\rho \left( \frac{\partial U_i}{\partial t} + U_j \frac{\partial}{\partial x_j} (U_i) \right) = -\frac{\partial p}{\partial x_i} + \frac{\partial}{\partial x_j} \left[ \mu \left( \frac{\partial U_j}{\partial x_j} + \frac{\partial U_j}{\partial x_i} \right) \right] + \rho g_i
\]  
(2-2-16)

and their dynamic boundary condition:
\[
\mu_0 \left( \frac{\partial U_i}{\partial x_j} + \frac{\partial U_j}{\partial x_i} \right) - \rho_0 \delta_{ij} = -p_1 \delta_{ij} + \mu_1 \left( \frac{\partial U_i}{\partial x_j} + \frac{\partial U_j}{\partial x_i} \right)
\]  
(2-2-17)

So far, only continuum effects have been embodied in the foregoing derivations. Therefore, since surface tension force is not a continuum effect, the pressure jump which can arise owing to the effect of surface tension should be included differently in the derived equations. Under thermodynamic equilibrium, the pressure jump at the interface is proportional to the radius of curvature, \( k \), of the interface. This curvature can be calculated using the level-set function [17]:
\[
k = -\nabla \cdot \left( \frac{\nu \phi}{|\nabla \phi|} \right)
\]  
(2-2-18)

If the curvature is taken to be positive when its center lies in the region where \( I_0 \) is defined, the pressure jump is calculated as:
\[
\Delta p = p_1 - p_0 = \sigma k
\]  
(2-2-19)

known as the Young-Laplace equation [17].

According to [14] gravity force can be incorporated in boundary condition for a two-phase flow instead of being considered as a body force distributing on the entire fluid domain. This consideration can notably improve the result of a numerical method, especially for high density fluid or gravity dominated flows. When gravity force is applied as a body force in flow equations, it might overwhelm the other terms, particularly for small Froude number (See examples in Section 2.8). Following [14] piezometric pressure is defined as \( \hat{p} = p + \rho g y \), if \( g \) points in the opposite direction of \( y \)-direction; then pressure terms can be replaced with:
\[
p = \hat{p} - \rho g y
\]  
(2-2-20)

Incorporating piezometric pressure and surface tension force effect into the dynamic boundary condition, Equation (2-2-17) yields:
\[ \mu_0 \left( \frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right) - (\dot{p}_0 + \sigma \kappa - \rho_0 g y) \delta_{ij} = - (\dot{p}_1 - \rho_1 g y) \delta_{ij} + \mu_1 \left( \frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right) \]  \hspace{1cm} (2-21)

Recasting back this boundary condition in two-phase flow equations, the following explicit momentum equation accommodates the boundary condition and governs the fluid domain:

\[
\frac{\partial}{\partial t} \left[ (\rho_0 I_0 + \rho_1 l_1) U_i \right] + \frac{\partial}{\partial x_j} \left[ (\rho_0 I_0 + \rho_1 l_1) U_i U_j \right] \\
= - \frac{\partial}{\partial x_i} (\dot{p}_0 I_0 + \dot{p}_1 l_1) + \frac{\partial}{\partial x_j} \left[ (\mu_0 I_0 + \mu_1 l_1) \left( \frac{\partial U_i}{\partial x_j} + \frac{\partial U_j}{\partial x_i} \right) \right] \\
+ \left[ (\rho_1 - \rho_0) g y + \sigma \kappa \right] \frac{\partial I_0}{\partial x_i} 
\]

\hspace{1cm} (2-22)

This explicit equation is reformed as an implicit equation:

\[
\frac{\partial}{\partial t} (\rho U_i) + \frac{\partial}{\partial x_j} (\rho U_i U_j) = - \frac{\partial \dot{p}}{\partial x_i} + \frac{\partial}{\partial x_j} \left[ \mu \left( \frac{\partial U_i}{\partial x_j} + \frac{\partial U_j}{\partial x_i} \right) \right] + \left[ (\rho_1 - \rho_0) g y + \sigma \kappa \right] \frac{\partial I_0}{\partial x_i} 
\]

\hspace{1cm} (2-23)

In which

\[ \rho = \rho_0 I_0 + \rho_1 l_1 \]  \hspace{1cm} (2-24)

\[ \mu = \mu_0 I_0 + \mu_1 l_1 \]  \hspace{1cm} (2-25)

\[ \dot{p} = \dot{p}_0 I_0 + \dot{p}_1 l_1 \]  \hspace{1cm} (2-26)

In this set of equations, we avoid any kind of Dirac delta function to be consistent with our numerical discretization of level-set function which is detailed in the next section.

### 2.3. Estimating Surface Tension and Gravity Force

In the previous section, the analytical formulation of two-phase flow equations and level-set function were explained. Utilizing these formulations, in this section, the proper implementation of interfacial body force, \([ (\rho_1 - \rho_0) g y + \sigma \kappa \] \( \frac{\partial I_0}{\partial x_i} \), is described. Interfacial body forces basically
embody the pressure difference due to the curvature of an interface and surface tension, or to the density variation across the interface which results in interfacial gravity force.

Generally, interfacial forces can be implemented either as “sharp” or “continuous” forces. Sharp surface force, SSF, refers to a numerical method that surface force is implemented on cells in which interface passes through. In SSF method, the surface tension force is non-zero only at faces which the distance function, $\varphi$, changes sign. Whereas, continuum surface force, CSF, aims to distribute this sharp force on several cells normal to the interface which leads to a smoother transition layer. According to Equation (2-2-23), there are two different types of interfacial body forces which must numerically implement i.e. surface tension force and gravitational force. Each of these forces can be implemented sharply or continuously.

![Diagram of an interface crossing over a meshed numerical domain](image)

Figure 1. An interface crossing over a meshed numerical domain, $h$ is interfacial thickness and $A(\vec{x}_s)$ represents interface location.

### 2.3.1. Surface Tension Force

The surface force per unit interfacial area reads as:

$$\vec{F}_s = \sigma \kappa \hat{n}$$  \hspace{1cm} (2-3-1)
Hence, total force applied on the interface, \( A(x_s) \) shown in Figure 1, is a surface integral over the interface area:

\[
\int \int_{A(x_s)} \vec{F}_s dA
\]

(2-3-2)

This surface force can be converted to a volume force by equating the total force applied on the surface to the equivalent force applied on the volume capturing the area, then:

\[
\int A \vec{F}_s dA = \int \int_V \vec{F}_v dV
\]

(2-3-3)

In which volume \( V(x_s) \) surrounds the surface area, \( A(x_s) \). This volume, \( V(x_s) \), is constructed by normal expansion of interface area as much as \( h/2 \) and \(-h/2\) normal to the interface. Following Brackbill at el. [1], a mollified color function, \( \tilde{c}(x_s) \), introduces a smooth transition region by convolving a color function \( c(x_s) \) and an interpolating function, \( \zeta(x_s) \):

\[
\tilde{c}(x) = \int \int_V c(x')\zeta(x' - x) dV
\]

(2-3-4)

Using this function, it is proved that:

\[
\int A \vec{F}_s dA = \lim_{h \to 0} \int \int_V \sigma\kappa \frac{\tilde{c}(x)}{[c]} dV
\]

(3-5)

\([c]\) is the difference between color function values in two phases. Comparing Equation (2-3-3) and (3-5) therefore, for a narrow enough transition region \( \vec{F}_v \) is estimated as:

\[
\vec{F}_v = \sigma\kappa \frac{\tilde{c}(x)}{[c]}
\]

(2-3-6)

Note that the integrands, \( \vec{F}_s \) and \( \vec{F}_v \), constitute SSF and CSF method for surface tension force, respectively. Equation (2-3-6) is similar to what was derived in Equation (2-2-23), which can be seen as:

\[
\vec{F}_v = \sigma\kappa \nabla I_0 \quad ; \quad \vec{F}_v = ([f_v]_x, [f_v]_y, [f_v]_z)
\]

(2-3-7)

The step function \( I_0 \) can be smeared-out such as[45]:
where ε establishes the region on which the interface is spread. As a result the volume force \( \vec{F}_V \) is reformulated as:

\[
\vec{F}_V = \sigma \kappa \nabla H(\varphi)
\]  \hspace{1cm} (2-3-9)

It is a common practice in level-set formulation to analytically take the derivative of this equation—since analytical value for \( H(\varphi) \) is available—and convert the volume force to:

\[
\vec{F}_V = \sigma \kappa \delta_H(\varphi)
\]  \hspace{1cm} (2-3-10)

where \( \delta_H(\varphi) \) is directly derived from Equation (2-3-8):

\[
\delta_H(\varphi) = \begin{cases} 
0 & \varphi < -\varepsilon \\
\frac{1}{2\varepsilon} \cos \left( \frac{\pi \varphi}{\varepsilon} \right) & -\varepsilon \leq \varphi \leq \varepsilon \\
0 & \varphi > \varepsilon
\end{cases}
\]  \hspace{1cm} (2-3-11)

However, as also mentioned in [13], Equation (2-3-10) is unable to exactly balance the numerical value for interfacial force in the vicinity of the interface. This can be shown, if the interfacial force is estimated at a cell face. For example for face “e” in Figure 2, using Equation (2-3-9), the interfacial force reads as:

\[
\left\{ [f_V]_X \right\}_e = \sigma \kappa_e \frac{H(\varphi_E) - H(\varphi_P)}{\Delta x}
\]  \hspace{1cm} (2-3-12)

Whereas using Equation (2-3-10) results in:

\[
\left\{ [f_V]_X \right\}_e = \sigma \kappa_e \left. \delta_H(\varphi) \right|_e
\]  \hspace{1cm} (2-3-13)

Equations (3-12) and (3-13) are equal if \( \left. \delta_H(\varphi) \right|_e \) is equal to \( \frac{H(\varphi_E) - H(\varphi_P)}{\Delta x} \); which is not necessarily the case. It is reported in [13] that Equation (2-3-13) produces inconsistent results when used in conjunction with the balanced force algorithm; however no reason is mentioned for this inconsistency. Even though one might argue that Equation (2-3-12) produces consistent result because of the similarity of numerical operator for calculating the gradient of step function and pressure, we will show in the second part of this paper that it is not the case. We will prove
in that part of paper that the crude estimation of surface forces in Equation (2-3-12) helps this equation to be consistent with Cartesian grid discretization.

![Figure 2. A schematic representation of three control volumes](image)

For SSF method one can implement the following logical condition to implement the sharp surface tension force:

\[
\text{If } \varphi(E) \cdot \varphi(P) < 0 \text{ then } \left[ f_s \right]_e = \frac{\sigma \kappa P}{\Delta x} \text{Sign} \left( \frac{-d\varphi}{dx} \right) \]  

(2-3-13)

this logical condition discerns sign change in level-set function and accordingly surface tension force is calculated at the control volume face where the sign change happens. This way of estimating surface force is equivalent to the method presented in [10].

### 2.3.2. Interfacial Gravity Force

Similar to surface tension force, gravity force appeared in the interfacial term in Equation (2-2-23):

\[
\left[ (\rho_1 - \rho_0)g y + \sigma \kappa \right] \frac{\partial l_0}{\partial x_i} \]  

(2-3-14)

and it can be implemented with the same strategy. In addition to variable elevation term, “y”, the difference in fluid densities and earth’s gravity constitute the gravity term in this equation. “y” which determines the elevation from a reference level must be calculated at the interface locations. Therefore, continuous or sharp implementation of the gravity force must start with calculating the elevation variable. For the sake of brevity, continuous implementation of the interfacial forces in which gravity effects are also included is named CIF, an abbreviation for “Continuous Interfacial Forces” and similarly “Sharp Interfacial Forces” is called SIF in this paper.
As it is shown in Figure 3, assume node “B” resides inside the marginal distance where the interfacial forces should be smeared out. Line “ee” is a perpendicular line to the shown interface and it passes through node “B”. The altitude of point “B” is indicated by \( y_B \) and “d” is the vertical distance between node “B” and the closest point to “B” at the interface. In the CIF method the altitude, “y”, in the interfacial body force:

\[
[(\rho_1 - \rho_0) g y] \frac{\partial l_s}{\partial x_i} \tag{2-3-15}
\]

can vary in the marginal region around the interface. For instance, in a line perpendicular to the interface, like “ee”, different points obtain different altitude. As a result, there are two options to estimate the altitude of any interfacial node such as node “B”: considering the actual altitude of node “B” or considering the altitude of the closest point to node “B” on the interface, point “A” in Figure 3. Using the latter approach, the altitude of point “A”, which is referred as effective altitude, is assigned to all marginal points along line “ee” within interface thickness. This effective altitude allows us to smear out a unique and constant value along line “ee” rather than a variable altitude for the interfacial nodes. The aim of this effective altitude is assigning a uniform pressure jump to the fluid domain. If actual altitude is used for the body force, the total pressure jump along the interface considerably depends on the thickness of the marginal region around interface, as different nodes within interface thickness have different altitudes. Dependence of the pressure jump to the interface thickness is the main disadvantage of using the actual altitude which can be avoided using the described effective altitude. One should note that gravitational forces might substantially be larger than surface tension forces, for instance for water-air interface, slight variation in altitude are multiplied by \( 10^4 \) which brings about a considerable variation in pressure jump.

In this work, the closest point at the interface is found and its altitude is considered for the desired location. For the sake of simplicity, the interfacial gravity force for node “B” is calculated, first by calculating the effective altitude using this expression:

\[
y_{\text{effective}} = y_B - \varphi(\vec{x}) n_y(\vec{x}) \bigg|_{\vec{x} = \vec{x}_B} + d \tag{2-3-16}
\]
where distance function is multiplied to the vertical component of unit normal vector, \( \vec{n}(\vec{x}) \) which is defined as:

\[
\vec{n}(\vec{x}) = \left( n_x(\vec{x}), n_y(\vec{x}), n_z(\vec{x}) \right)
\]  

(2-3-17)

In this way, the volumetric gravity force for CIF method for node “B” reads as:

\[
\vec{F}_V\big|_{\vec{x}=\vec{x}_B} = \left[ (\rho_1 - \rho_0)g \cdot y_{\text{effective}} \right] \frac{\partial I_0}{\partial x_i}
\]  

(2-3-18)

Node “B” therefore has two different volumetric forces in x and y directions (for 2D case). Even though we used node “B” for demonstrating the method, the volumetric forces can be calculated at the desired locations such as control volume faces. For instance, the gravity body force for face “e” in Figure 2 is calculated as:

\[
\{[f_v]_e\} = (\rho_1 - \rho_0) \cdot g \cdot y_{\text{effective}} \cdot \frac{H(\varphi_E) - H(\varphi_P)}{\Delta x}
\]  

(2-3-19)

The gravity force can also be implemented sharply at the interface locations. This is very similar to what was presented for surface tension force. A logical condition verifies if the gravity force should be applied on the desired place, face “e” in Figure 2:

\[
\text{If } \varphi(E) \cdot \varphi(P) < 0 \text{ then } [f_x]_e = \frac{(\rho_1 - \rho_0) \cdot g \cdot y_{\text{effective}}}{\Delta x} \cdot \text{Sign} \left( -\frac{d\varphi}{dx}\big|_{x_E} \right)
\]  

(2-3-20)

for SIF; Likewise, the other components of gravity force can be calculated on the other faces.
2.4. Discretization of Pressure Gradient in Conjunction with Forcing Terms

In the previous section, numerical techniques to estimate the interfacial term were detailed:

\[
F_I = \left[(\rho_1 - \rho_0)gy + \sigma \kappa\right] \frac{\partial l_0}{\partial x_i}
\]

(2-4-1)

where \( F_I \) represents the volumetric force estimated around the interface. We now need to derive a method by which these volumetric forces accurately link to pressure gradients around interface, in order to capture the desired pressure jump. However, we want to derive generic schemes for implementing forcing terms in flow equations. Therefore, we schematically represent the pressure gradient term and forcing term in the RHS of Equation (2-2-23) as:

\[
- \frac{\partial \hat{p}}{\partial x_i} + \left[(\rho_1 - \rho_0)gy + \sigma \kappa\right] \frac{\partial l_0}{\partial x_i} \equiv - \frac{\partial \hat{p}}{\partial x_i} + F_I
\]

(2-4-2)

and now we intend to find a suitable discretization for this part of the momentum equation. Note that we consider \( F_I \) as a symbol of a general forcing term with no restriction on its type or its location. It can be any kind of force such as electromagnetic, gravity or interfacial force which can be either applied on the entire domain or a part of the domain.
Numerical strategy for capturing interfacial forces considerably depends on the way pressure gradients are calculated. In our formulation, since the Navier-Stokes equations are discretized at cell centers, collocated arrangement, we must incorporate the forcing terms so that pressure gradient terms can accurately embrace them. As a result, all these forces are estimated on control volume faces. Using this strategy, for a one-dimensional case, forcing terms are represented as $F_e$ and $F_w$ for east and west faces respectively, Figure 4. In the absence of viscosity, applying Newton’s second law for the control volume surrounded between “P” and “E” yields:

$$\rho_e A \frac{d\rho e}{dt} + F_e \Delta x A - \rho_e A \frac{d u_e}{dt}$$

in which density of face “e” is assumed to dominate in the control volume shown with dashed lines in Figure 4. “A” is the area of each face which is assumed to be the same for both faces and $\Delta x$ is equidistance space between each two nodes. Similarly, for the control volume between “P” and “W”, linear momentum equation reads as:

$$\rho_w A \frac{d\rho w}{dt} + F_w \Delta x A - \rho_w A \frac{d u_w}{dt}$$

Figure 4. A schematic representation of three control volumes and their forcing terms estimated on cell faces

If in Equations (4-1) and (4-2) the face areas are canceled and the rest of the terms are recast as:

$$\begin{align*}
\frac{\rho_e \Delta x}{\rho_e} \frac{d u_e}{dt} + F_e \rho_e &= \frac{d u_e}{dt} \\
\frac{\rho_w \Delta x}{\rho_w} \frac{d u_w}{dt} + F_w \rho_w &= \frac{d u_w}{dt}
\end{align*}$$

Now, each side of these equations can be summed and averaged, giving:
\[
\frac{1}{2} \left( \frac{\dot{p}_e - \dot{p}_E}{\rho_e \Delta x} + F_e \right) + \left( \frac{\dot{p}_w - \dot{p}_P}{\rho_w \Delta x} + F_w \right) = \frac{1}{2} \left( \frac{du_e}{dt} + \frac{du_w}{dt} \right) \tag{2-4-6}
\]

in which the pressure subtraction terms can be interpreted as estimation of pressure gradient on “e” and “w” faces, therefore:

\[
\frac{1}{2} \left[ \left( -\frac{1}{\rho_e} \frac{d \rho}{dx} \right)|_e + F_e \right] + \left( \frac{1}{\rho_w} \frac{d \rho}{dx} \bigg|_w + F_w \right) = \frac{1}{2} \left( \frac{du_e}{dt} + \frac{du_w}{dt} \right) \tag{2-4-7}
\]

or

\[
\frac{1}{2} \left[ \left( -\frac{1}{\rho} \frac{d \rho}{dx} + \frac{F}{\rho} \right)|_e + \left( -\frac{1}{\rho} \frac{d \rho}{dx} + \frac{F}{\rho} \right)|_w \right] = \frac{1}{2} \left( \frac{du}{dt}\bigg|_e + \frac{du}{dt}\bigg|_w \right) \tag{2-4-8}
\]

If we summarize the LHS of Equation (2-4-8) as:

\[
\frac{1}{2} \left[ \left( -\frac{1}{\rho} \frac{d \rho}{dx} + \frac{F}{\rho} \right)|_e + \left( -\frac{1}{\rho} \frac{d \rho}{dx} + \frac{F}{\rho} \right)|_w \right] \equiv \left[ -\frac{1}{\rho} \frac{d \rho}{dx} + \frac{F}{\rho} \right]_{f \rightarrow c} \tag{2-4-9}
\]

where symbol “f \rightarrow c” denotes simple averaging from control volume faces to its center. Then using this symbol, Equation (2-4-6) reads as:

\[
\left[ -\frac{1}{\rho} \frac{d \rho}{dx} + \frac{F}{\rho} \right]_{f \rightarrow c} = \frac{1}{2} \left( \frac{du}{dt}\bigg|_e + \frac{du}{dt}\bigg|_w \right) \tag{2-4-10}
\]

Subsequently, we assume the average accelerations of faces “e” and “w” can be replaced with the acceleration of node “P” which is located in the center of the middle control volume “c” (the symbol of control volume center), then:

\[
\left[ -\frac{1}{\rho} \frac{d \rho}{dx} + \frac{F}{\rho} \right]_{f \rightarrow c} = \frac{du}{dt}\bigg|_c \tag{2-4-11}
\]

This is the key equation which should be incorporated in the Navier-Stokes to balance forcing terms with pressure gradients. Equation (2-4-11) is a numerically consistent relation which should be considered in the vicinity of interface. The above derivation holds in the absence of viscosity. In order to incorporate viscosity effect in the formulation, we assume cell-centered discretization method for viscosity term around interface is sufficient and no special treatment for viscosity term is needed. As a result, the effect of viscosity on the central control volume follows as:
\[
\left[ -\frac{1}{\rho} \frac{d\hat{p}}{dx} + \frac{F}{\rho} \right]_{f \rightarrow c} + \frac{\mu}{\rho} \nabla^2 \mathbf{u} \bigg|_c = \frac{du}{dt} \bigg|_c
\]  

(2-4-12)

comparing this discretization with one-dimensional form of Equation (2-2-23):

\[
\frac{D\mathbf{u}}{Dt} = -\frac{1}{\rho} \frac{d\hat{p}}{dx} + \frac{\mu}{\rho} \nabla^2 \mathbf{u} + \frac{F}{\rho}
\]  

(2-4-13)

therefore, Equation (2-4-12) is a discretization of Equation (2-2-23) written for the central control volume in Figure 4. Hence, following Equation (2-4-12), in our formulation momentum equation is discretized as:

\[
\frac{D(\rho \mathbf{u}_i)}{Dt} \bigg|_c = \frac{\partial}{\partial x_i} \left[ \mu \left( \frac{\partial \mathbf{u}_i}{\partial x_i} + \frac{\partial \mathbf{u}_j}{\partial x_j} \right) \right] + \rho_c \left[ -\frac{1}{\rho} \frac{d\hat{p}}{dx_i} + \frac{F_i}{\rho} \right]_{f \rightarrow c}
\]  

(2-4-14)

where \( F_i \) represents different directions of the volumetric force \( F \). The last term on the RHS of this equation is similar to what is presented in [10] as a proper interpolation scheme, so-called balanced-force algorithm. It will be shown that using this discretization along with appropriate pressure-velocity coupling formulation can eliminate spurious currents, see Section 2.8.

Alternatively, the analytical one-dimensional Navier-Stokes equation for the cell center “c” might be discretized as (see Equation (2-4-13)):

\[
\frac{D\mathbf{u}}{Dt} \bigg|_c = -\frac{1}{\rho} \frac{d\hat{p}}{dx} \bigg|_c + \frac{\mu}{\rho} \nabla^2 \mathbf{u} \bigg|_c + \frac{F}{\rho_c}
\]  

(2-4-15)

if the interfacial force is estimated at the cell center. This is the equation which is conventionally used in many numerical methods. Having proved Equation (2-4-12), it is clear this simple discretization consequences in spurious currents all around interface; see[10][16]. This is mainly due to the lack of balance between pressure gradient and the interfacial force in a discretized space.

We derived all the equations based on the assumption of having uniform mesh in the computational domain. In the way we derived Equation (2-4-14), we strictly used this assumption for averaging the acceleration and other terms. Having shown the proof for Equation (2-4-14), one can see that process of converting Equation (2-4-8) to Equation (2-4-11) is not possible but for uniform mesh. This is shown in Appendix II-A that the above discretization
encounters difficulties for non-uniform mesh. In the appendix, we prove that the similar methods (such as the balanced-force algorithm) are not readily expandable to non-uniform mesh.

2.5. Pressure-Velocity Coupling

It is well-known that collocated arrangement of variables on a numerical grid encounters pressure-velocity coupling problem which may create oscillations in the pressure field. Increasing accuracy for calculating face velocities using either pressure gradients or higher order estimations using Taylor series are common approaches to resolve the difficulty [20]. It is shown in [9] that compact central-difference approximations of pressure gradients adds a correction to the interpolated velocity that is proportional to the third derivative of the pressure multiplied by $(\Delta x)^2/4$. As described by Rhie and Chow in [43], the method can be briefly explained by considering the discretized momentum equation in SIMPLE algorithm as:

$$a_p^u u_p = \sum a_{nb}^u u_{nb} + b$$ (2-5-1)

$a_p^u$ and $a_{nb}^u$ are constant coefficients derived after discretizing Navier-Stokes equations [9][33]. To estimate face velocities, the compact central-difference approximations at the cell faces can be used as [9]:

$$u_e = \langle u_c \rangle_{c \rightarrow f} + \left[ \frac{\psi_p}{a_p^u} \right]_{f \rightarrow c} - \frac{\psi_p}{a_p^u}$$ (2-5-2)

in which the momentum coefficient $a_p^u$ is also interpolated all through the procedure. This expression which is designed for SIMPLE-type algorithm uses no time step and therefore it adds no restriction for the type of numerical solution. Hence, it can be used for both steady and unsteady solutions. This property is mainly desired for pressure-based algorithms in which both steady and unsteady flows can be solved. We now need to extent this concept for the discussed interfacial body forces and make sure these interpolations are accurately integrated to the whole algorithm.

Francois et al. [10] suggests that instead of Equation (2-5-2) the following equation can be used to estimate face velocities:
where the superscripts “n” and “n+1” represent the previous and current time step. Clearly the above equation is inspired by the Equation (2-4-11) which suggests pressure gradients divided to density must be carried along with interfacial body forces divided by density. This formulation is used in two-step projection method and since basically two-step projection method is designed for unsteady solutions, the above equation suits the flow algorithm well. However, the appeared time step, $\Delta t$, in this formulation prevents its use in pressure-based algorithms such as SIMPLE which tend to be useful for both steady and unsteady flows. Besides, we encountered pressure oscillations even when Equation (2-5-3) was implemented in our SIMPLE algorithm for an unsteady solution. The oscillation occurred mainly due to the absence of $a_p^u$ in Equation (2-5-3), whereas in Equation (2-5-2) $a_p^u$ is carried along the interpolation procedure and therefore it enhances the pressure-velocity coupling accuracy proportional to the third derivative of the pressure multiplied by $(\Delta x)^2/4$. As a result to be consistent with Equations (5-2) and (4-11) we used the following formulation in our SIMPLE algorithm:

$$u_e = (u_c)_{c\rightarrow f} + \frac{\rho_f}{\rho} \left( \frac{1}{a_p^u} \left( \nabla p - \frac{F}{\rho} \right) \right)_{c\rightarrow f} - \frac{\Delta t}{\rho f_{n+1}} (\nabla p^n - F^{n+1})_f$$

(2-5-4)

In this expression, there is no time step which allows the algorithm to be universally used in both steady and unsteady flows. Besides, both $a_p^u$ and interfacial forces are interpolated from faces to cell-centers which accommodate both equations (4-11) and (5-2). However, this formulation needs density of faces which should be properly interpolated from cell centers. Our numerical experiments demonstrated that linear interpolations for face densities become unstable for high density ratios, whereas harmonic averaging considerably improves the instability problem.

2.6. The Level-Set Method (LS)

We employ a standard Level-set algorithm to track the interface. The method uses fifth order accurate scheme which is so called HJ-WENO [17]. The level-set method makes use of a signed continuous function which allows the prediction of the free surface, normal to the surface and
curvature. Briefly, in this method [32], the interface is traced by the LS function $\varphi$ which for a domain $\Omega$ (Figure 5) is defined as a signed distance to the boundary $\partial \Omega$.

$$|\varphi(\vec{x}, t)| = \min(|\vec{x} - \vec{x}_I|) \quad \vec{x}_I \in \partial \Omega \quad (2-6-1)$$

Figure 5. A representation of the level-set function for a numerical domain, $\partial \Omega$.

Implying that $\varphi(\vec{x}, t) = 0$ on the boundary where $\vec{x} \in \partial \Omega$. Thus

$$\varphi(\vec{x}, t) = \begin{cases} 
< 0 & \vec{x} \in \Omega^- \\
= 0 & \vec{x} \in \partial \Omega \\
> 0 & \vec{x} \in \Omega^+ 
\end{cases} \quad (2-6-2)$$

The location of the interface ($\varphi(\vec{x}, t) = 0$) is determined by Equation (2-2-11):

$$\frac{\partial \varphi}{\partial t} + U_i \frac{\partial \varphi}{\partial x_i} = 0$$

There are a number of methods to discretize the convective terms of this equation. However, practical experience [32] suggests that level-set method is sensitive to the accuracy of spatial discretization, implying that schemes of high order accuracy (third-order or higher) must be used. The unit normal vector and curvature at any point on the liquid interface are calculated from the LS function as follows (details can be found in [32]):

$$\hat{n} = \frac{\vec{v}_\varphi}{|\vec{v}_\varphi|} \quad k = -\nabla \cdot \left( \frac{\vec{v}_\varphi}{|\vec{v}_\varphi|} \right) \quad (2-6-3)$$

As the interface evolves in time, $\varphi$ loses its character to be a distance function [32]. Thus to keep $\varphi$ a signed distance function, a technique, known as reinitialization, should be applied. Sussman, Smereka and Osher [45] suggested the following equation to reinitialize the distance function:
\[
\frac{\partial \varphi}{\partial \tau} + S(\varphi_0)(|\nabla \varphi| - 1) = 0 \tag{2-6-4}
\]

where \(S(\varphi_0)\) is a sign function taken as 1 in \(\Omega^+\), -1 in \(\Omega^-\) and 0 on the interface and \(\varphi_0\) indicates the initial value of \(\varphi\) before solving the reinitialization equation. To smear out the numerical values near interface this sign function is suggested to be of the following form [45]:

\[
S(\varphi_0) = \frac{\varphi_0}{\sqrt{\varphi_0^2 + \Delta x^2}} \tag{2-6-5}
\]

where \(\Delta x\) is spatial increment in the discretized domain. To discretize \(S(\varphi_0)|\nabla \varphi|\), Godunov’s method is recommended through the following formulation [32]:

\[
\varphi_\chi^2 \approx \text{Max}(\text{Max}(\varphi_\chi^-, 0)^2, \text{Min}(\varphi_\chi^+, 0)^2) \quad \text{if} \quad S(\varphi_0) > 0
\]

\[
\varphi_\chi^2 \approx \text{Max}(\text{Min}(\varphi_\chi^-, 0)^2, \text{Max}(\varphi_\chi^+, 0)^2) \quad \text{if} \quad S(\varphi_0) < 0 \tag{2-6-6}
\]

The distance function is reinitialized using HJ-WENO scheme which is fifth order accurate in space. The reinitialization equation can be discretized in time using simple Euler time advancement:

\[
\varphi^{n+1} = \varphi^n + \Delta \tau \ S(\varphi_0)[1 - \sqrt{\varphi_\chi^2 + \varphi_y^2 + \varphi_z^2}] \tag{2-6-7}
\]

or by using Runge–Kutta schemes either TVD [32] or non-TVD formats [12]. For all the simulation in this work, we used fourth-order non-TVD Runge–Kutta schemes to take advantage of its higher accuracy:

\[
\varphi^{n+1} = \varphi^n + \frac{\Delta \tau}{6} [k_1 + 2k_2 + 2k_3 + k_4]
\]

\[
k_1 = S(\varphi_0)[1 - |\nabla (\varphi^n)|]
\]

\[
k_2 = S(\varphi_0) \left[1 - |\nabla (\varphi^n + \frac{\Delta \tau}{2} k_1)|\right]
\]

\[
k_3 = S(\varphi_0) \left[1 - |\nabla (\varphi^n + \frac{\Delta \tau}{2} k_2)|\right]
\]

\[
k_4 = S(\varphi_0) \left[1 - |\nabla (\varphi^n + \Delta \tau k_3)|\right] \tag{2-6-8}
\]

Results of the Hamilton–Jacobi equation for reinitialization are very sensitive to the setting used in the solution procedure, namely the number of reinitialization steps performed per unit of time, and the number and size of false time steps used to solve the reinitialization equation. This is pointed out in several articles such as Peng et al. [35] and GÓMEZ et al. [12]. In this work, the pseduo time step, \(\Delta \tau\), is taken to be numerically equal to half of the grid size and to reduce computational costs the reinitialization subroutine was called after 3 real time
steps. Besides, a limit on the maximum number of false time steps is placed which cannot be exceeded even when reinitialization fails to enforce this condition:

$$1 - |\nabla \phi_{i,j,k}| \leq \text{Error}_{\text{Max}}$$  

(2-6-9)

Additionally, if an updated distance function finds more error than its previous values, then that updated field is disregarded. Clearly the choice of these parameters affects the final results as it is shown in P. Gómez et al. [12].

To validate the level set implementation, two standard test cases were considered. In the first test case, a circle of radius of .2 is positioned at (0.0, 0.65) in a $2 \times 2$ box subjected to a rotating flow field:

$$u = y$$

$$v = -x$$

The origin of the coordinate system is chosen at the centre of the box. The test case was first discretized on $150 \times 150$ nodes and advanced in time using $\Delta t = 2\pi/2400$ and then discretized on $250 \times 250$ nodes and advanced in time using $\Delta t = 2\pi/3600$. Table 1 and Figure 6 show the results of these tests. It is shown in the table the initial and final areas of the disk and the mass loss after one full rotation.

<table>
<thead>
<tr>
<th>Mesh Resolution</th>
<th>Initial Area</th>
<th>Final Area</th>
<th>Mass loss %</th>
</tr>
</thead>
<tbody>
<tr>
<td>$150 \times 150$</td>
<td>0.12573697</td>
<td>0.12653932</td>
<td>-0.63811731</td>
</tr>
<tr>
<td>$250 \times 250$</td>
<td>0.12568957</td>
<td>0.12543300</td>
<td>-0.20413393</td>
</tr>
</tbody>
</table>
The second test case is a vortex test. A circle of radius .15 is initially located at (0.5, 0.75) in a $1 \times 1$ domain. The advection uses the following velocity field till $t=1$:

$$u = \sin^2(\pi x) \sin (\pi y)$$

$$v = -\sin^2(\pi y) \sin (\pi x)$$

Then the simulation continues with the negative of the above velocity field till $t=2$ to return the circle to its initial position. To test the effect of mesh resolution, the simulation was carried on $100 \times 100$ and $250 \times 250$ computational nodes using $\Delta t = 2/2000$ and $\Delta t = 2/6000$ respectively. The satisfactory results are presented in Table 2, Figure 7 and Figure 8 [41].

<table>
<thead>
<tr>
<th>Mesh Resolution</th>
<th>Initial Area</th>
<th>Final Area</th>
<th>Mass loss %</th>
</tr>
</thead>
<tbody>
<tr>
<td>$100 \times 100$</td>
<td>0.07072798</td>
<td>0.06827780</td>
<td>-3.46423073</td>
</tr>
<tr>
<td>$250 \times 250$</td>
<td>0.07069257</td>
<td>0.07040941</td>
<td>-0.40054994</td>
</tr>
</tbody>
</table>
Figure 7. Result of the vortex test using 100×100 computational nodes
Figure 8. Result of the vortex test using 250×250 computational nodes
2.7. Outline of the numerical algorithm

An outline of the overall iterative solution procedure at each time step is presented below (Equations are simplified for X-direction):

1. Calculate the surface tension forces (details are given in Section 2.3)
   For CIF (using Equation (2-3-12) and Equation (2-3-19)):
   \[
   \{[f_{v_{x}}]_{e}\} = \left[\sigma_{e} + (\rho_{1} - \rho_{0}) * g * y_{\text{effective}} \right] * \frac{H(\varphi_{E}) - H(\varphi_{P})}{\Delta x}
   \]
   For SIF (using Equation (2-3-13) and Equation (2-3-20)):
   \[
   \text{If } \varphi(E) * \varphi(P) < 0 \text{ then } \{f_{x}\}_{e} = \left[\sigma_{e} \Delta x + \frac{(\rho_{1} - \rho_{0}) * g * y_{\text{effective}}}{\Delta x} \right] \text{Sign} \left(-\frac{d\varphi}{dx}\right)
   \]

2. Solve the discretized momentum equations (details are given in Section 2.2 and 2.4)
   \[
   \frac{\partial}{\partial t}(\rho_{c} U_{1}) + \frac{\partial}{\partial x_{i}}(\rho_{c} U_{j} U_{1}) = \frac{\partial}{\partial x_{j}} \left[ \mu_{c} \left( \frac{\partial U_{1}}{\partial x_{j}} + \frac{\partial U_{j}}{\partial x_{1}} \right) \right] + \rho_{c} \left[ \frac{1}{\rho} \frac{d\rho}{dx_{1}} C_{1} + F_{1} \right]_{f \rightarrow c}
   \]

3. Solve the discretized pressure equations by calculating the cell-face velocities using the momentum interpolation scheme (details are given in Section 2.5)
   \[
   u_{c} = \langle u_{c} \rangle_{c \rightarrow f} + \rho_{f} \left[ \frac{1}{a_{p}^{u}} \left( \nabla p - \frac{F}{\rho} \right) \right]_{f \rightarrow c} - \rho_{f} \left[ \frac{1}{a_{p}^{u}} \left( \nabla p - \frac{F}{\rho} \right) \right]_{f}
   \]

4. Repeat steps 2 and 3 until the solution converges for each time step

5. Update the level-set function, \( \varphi \), (details are given in Section 2.6) and using the updated field, calculate density and viscosity for the entire flow field using Equation (2-2-24) and (2-2-25)

6. Estimate the curvature and normal vectors using Equation (2-6-3); return to Step 1 for the next time step.

2.8. Results

This section presents a range of test cases to illustrate the ability of the proposed methods. All the test cases presented in this section have low or zero Froude number. Low Froude number examples are chosen, since numerical errors are comparable with their small characteristic velocity. First, stationary water droplet for a 2D case is presented to validate the proposed
discretization schemes for surface tension force. Then the implementation of piezometric pressure methods, CIF and SIF, are tested for a stationary column of a two-phase flow. Finally, in order to demonstrate the capability of the piezometric pressure concept the results of a Rayleigh–Taylor instability test and a falling drop are presented.

2.8.1. Static drop in equilibrium

In this section, we illustrate the application of Equation (2-2-23) in conjunction with the proposed pressure correction routine, Equation (2-5-4) to solve a static drop. For sake of comparison, we use a similar configuration presented in [10]. A drop with radius $R=2$ is positioned in a square domain having side length of 8 units. The surface tension coefficient is taken to be 72 and the density inside the drop is 1 while the outside density is varied to create different density ratios. The exact pressure jump is then:

$$ (\Delta P)_{exact} = \sigma \kappa = 72 \times \frac{1}{2} = 36 $$

The simulation was carried out on uniform mesh size, $h$, and mesh resolution for the presented case is $R/h=10$. Since we are able to solve simulations both in steady and unsteady state, we solved this case as a steady state case. As well, the unsteady solution converged to steady state results. Figure 9 shows the results for CSF and SSF implementation of surface tension force in the absence of gravity force. Both cases attain the exact pressure jump with machine zero error in pressure jump and spurious currents. Pressure jump here means the maximum pressure minus the minimum pressure on the entire domain. Similar results, exact pressure jump with no spurious current, were achieved using different resolutions $R/h=5, 10, 20$ and $40$ for different density ratios ranging from 1 to $10^4$. 
2.8.2. Column of two phase flow

It was shown that piezometric pressure can be incorporated into flow equations for two phase fluids. In this test case, the piezometric pressure formulation, summarized in Equation (2-2-23), is tested and its advantages in compare to normal implementation of gravity force are explained.
Since the pressure term appears in the interfacial part of Equation (2-2-23), it can be implemented sharply (SIF) or continuously (CIF). To validate the model described in Section 2.3, a column of a heavy fluid is pressurized by a lighter fluid, Figure 10. A domain with side length of one unit is equally filled with a lighter phase with density 1 and a heavier phase with density 1000 distributed on a uniform meshes of $15 \times 60$, Figure 10(a). The test case is first solved using regular implementation of gravity as cell-centered body force on the entire domain (see Equation (2-4-15)) and then the piezometric pressure is introduced to the solution. Figure 10(b) shows the results of a numerical solution when gravity force is applied on the entire domain as a body force. As a result, linear pressure distribution and pressure jump due to variable density is obtained. However, as explained in Section 2.4, since in cell-centered body force formulation, Equation (2-4-15), the accurate balance between pressure gradients and interfacial forces is ignored, spurious currents as large as .011 appear close to the interface. Consequently, numerical pressure differs 1.5% from its exact value:

$$
\left( \rho_1 \frac{g H}{2} + \rho_0 \frac{g H}{2} \right) = (1000 + 1) \times 9.81 \times .5 = 4909.905
$$

where $H$ is the height of the domain. The inaccuracy decreases, if mesh resolution increases. For instance, for a very fine mesh resolution (240 nodes in the vertical direction) spurious currents decrease to .0017 and pressure error is less than .01%. However, it still can be significant error for low Froude number flow simulations which are involved with small characteristic velocity. This test case demonstrates an existing error if one avoids piezometric pressure and applies gravity as a cell centered body force on the entire domain (conventional method). Even though the error decreases with increasing the number of mesh, it considerably increases the computational cost.

Similarly, the same field is numerically solved using the piezometric pressure analysis, Figure 10(c-d). Since gravity effects are only applied at the interface locations, there should be no pressure gradient within each phase. However, density variation at the interface creates a pressure jump. Generally, the gravitational pressure jump acts similar to the surface tension pressure jump. Since we set the ground level at the bottom of the domain, from Equation (2-2-23), the exact pressure jump must be:

$$
(\rho_1 - \rho_0)g y = (1000 - 1) \times 9.81 \times .5 = 4900.095
$$
This is almost the same pressure jump for both CIF and SIF solutions shown in Figure 10. Clearly, there are no spurious velocities for these two solutions as a result of using Equation (2-4-14). Uniform pressure distribution in each of the phases can be seen for both CIF and SIF. Due to the extrapolation scheme described in Section 2.3 to find the exact location of the interface, the CIF method predicts the same pressure jump as SIF. If the location of the interface was not assigned as elevation term for all the interfacial nodes, the CIF-pressure jump differed from its exact value proportional to the interface thickness.

Figure 10. Numerical solutions of a stationary two-phase flow with density ratio of 1000/1 on a uniform mesh net are presented. Gravity force is implemented in three different methods: b) as a cell-centered body force on the entire domain, c) SIF and d) CIF. Their exact pressure jumps are: b) 4909.905, c) 4900.095, d) 4900.095. The spurious currents appeared in the solution of case (b) causes 1.5 % error in predicting pressure jump, whereas the two other solutions for SIF and CIF encounter no significant errors.

Drawbacks of conventional implementation of gravity force, cell-centered formulation, are not limited to two-phase flows. The implementation even fails if non-uniform mesh is used for single-phase flows. Figure 11 compares the numerical results for a single-phase flow of density 1000 using two different meshes. Figure 11(a) illustrates the results obtained using uniform mesh and Figure 11(b) shows the inaccurate results acquired with non-uniform mesh. We used algebraic expansion with a factor of 1.02. One can prove that cell-centered formulation,
Equation (2-4-14), in conjunction with uniform mesh for single-phase flow recovers Equation (2-4-15) whereas it does not hold for non-uniform mesh; therefore, spurious currents appear for the case of non-uniform mesh.

Figure 11. Numerical solutions of a stationary single-phase flow using conventional implementation of gravity as a cell-centered body force: a) Uniform mesh and b) non-uniform mesh. Density is 1000. Inaccurate balance between pressure gradients and volumetric-body forces causes spurious current ($\sim 10^{-7}$) for non-uniform mesh.

Another illustrative example is comparing the solution of regular implementation of gravity and piezometric pressure in filling out a small tank with a small flux of water. A column of two-phase fluids, water and air, is numerically solved when a boundary velocity of .1 is assigned at the inlet, Figure 12. Viscosities of both fluids are ignored and slip boundary condition is imposed on walls. We used a uniform mesh of $25 \times 55$ to resolve the domain. The domain once was solved using the regular implementation of gravity, Figure 13 (a), and then it was solved using the piezometric pressure version, Figure 13 (b). Since the spurious currents of the conventional implementation of gravity are comparable with the characteristics velocity, the simulation fails as soon as the accumulated errors overcome the inlet velocity. The asymmetric 5th order HJ WENO scheme is responsible for the asymmetric result in this case. This can be further tested if the inlet velocity is increased to .35. In that case, since the inlet velocity
overcomes the accumulated spurious velocities, the normal implementation of gravity encounters no instability, Figure 14.

Figure 12. Schematic of the small tank filled with water and air.
Figure 13. Filling out a small tank with water inlet velocity as .1; a) using standard implementation of gravity b) using CIF method. Interface locations are shown for different times.

Figure 14. Filling out a small tank with water inlet velocity as .35 using standard implementation of gravity. Interface locations are shown for different times.
One should note, since the ground level is an arbitrary point, the interfacial force, $\Delta \rho gy$, can be considerably reduced if a suitable point is chosen. For instance, in Figure 11 if the ground level were chosen at the interface location, there would be no pressure jump if the flow stays stationary. Therefore, in that case, the gravity force comes to picture only if some parts of the interface displace from its current position. As a result, a two-phase flow can be efficiently simulated with minimum amount of body force exerted only on the deviated part of the interface. Generally, considering the ground level at the interface is helpful, if the interface has limited displacements from its original location or its displacement is around a stationary location.

### 2.8.3. Rayleigh-Taylor Instability

To demonstrate the performance of the proposed method for implementing gravity, the well-known Rayleigh-Taylor instability is a suitable test case. This test case has been presented in literature using different methods (Bell and Marcus [3]; Popinet and Zaleski [37]; Puckett et al. [40]; Gomez et al. [12]). We repeat the same test case similar to what was presented in Gomez et al. [12]. A heavy fluid $\rho_0 = 1.225$ is placed above a light fluid $\rho_1 = 0.1694$ and both fluids have the same viscosity $\mu_0 = 0.00313$. The actual test case has a domain of size $1 \times 4$, however, due to symmetry of the case, the computational domain is chosen to be the half of the physical domain, $0.5 \times 4$. The separating interface is located in the middle of the domain and initially disturbed by a cosine function with amplitude of 0.05:

$$y(x) = -ACos\left(\frac{2\pi x}{l}\right) \quad ; \quad A = 0.05, \; l = 1$$

Symmetry boundary conditions were applied at the lateral boundaries and free-slip conditions were imposed for the upper and lower walls. The gravity constant is set to $g = 9.81$. For these simulations, we considered a constant time step for the entire time integration, $\Delta t = 2.5 \times 10^{-4}$ [13].

For the piezometric pressure case, we used CIF implementation of gravity where gravity and interfacial forces were smeared-out on 1.5 cell size off the interface, Figure 15. Similar to [12], we set the maximum number of pseudo steps to 4 and $Error_{Max} = 1$. The computational domain was evenly resolved using $64 \times 512$ nodes in X and Y directions, respectively. Figure
15 shows the results of this simulation, which are in agreement with the results obtained using the standard level-set method in [12].

It is shown in [12] that mesh resolution, pseudo time steps, Error\_Max and the number of reinitialization considerably affects the results of level-set function for this simulation. Both level-set function and reinitialization equations are very sensitive to their methods of discretization (including resolution) and variable settings.

Having explained difficulties for comparison, to compare the piezometric and regular implementation of pressure, two different cases with the same computational settings are simulated. For reinitialization, the maximum number of false steps is set to 4 and we considered $\text{Error}_{\text{Max}} = 1$ [12] according to Equation (2-6-9). The computational domain is resolved once on $32 \times 256$ and then $64 \times 512$ nodes in X and Y directions, respectively. Figure 15 and Figure 16 compare the results in each case. As it is shown, the solutions these two methods are more similar when the resolution increases. The reason can be sought in the inherent error in normal implementation of gravity. We showed in Section 2.8.2 that spurious currents due to gravity decrease as the mesh resolution increases. That is why more discrepancies are observed in Figure 16, as higher spurious currents deteriorate the results of normal gravity. We also noticed that the number iterations needed for convergence in each time step is considerably higher for regular implementation of gravity. For some time steps the convergence cost almost 10 times more than using piezometric pressure implementation. Existing of body forces on the entire domain in addition to permanent spurious currents at the interface locations are possible reason for higher computational cost.

The effect of resolution also can be seen in Figure 16 and Figure 17 which follows the same pattern demonstrated in [12]. It is shown in [12] that higher mesh resolution improves the results of the level-set method; therefore Figure 15 and Figure 17 have higher accuracies.
Figure 15. Results obtained for Rayleigh-Taylor Instability test using piezometric formulation for pressure. CIF method was considered for implementation of the interfacial. Resolution is $64 \times 512$. The shown counters are 0-level Set functions at: a) 0, b) .4, c) .6, d) .8, e) .9 seconds.
Figure 16. Comparing the results of Rayleigh-Taylor Instability test for regular implementation of gravity force with the piezometric pressure concept. Resolution is 32×256. For piezometric pressure, the interfacial forces were applied using CIF method. The dashed line indicates the 0 level set functions for regular gravity and the solid lines for the piezometric pressure formulation for: a) 0, b) .4, c) .6, d) .8, e) .9 seconds.
Figure 17. Comparing the results of Rayleigh-Taylor Instability test for regular implementation of gravity force with the piezometric pressure concept. Resolution is 64×512. For piezometric pressure, the interfacial forces were applied using CIF method. The dashed line indicates the 0 level set functions for regular gravity and the solid lines for the piezometric pressure formulation for: a) 0, b) .4, c) .6, d) .8, e) .9 seconds.

2.8.4. Falling Water Droplet

A small water droplet surrounded in air, similar to the test case presented in [17], is simulated in this section. We used CIF implementation of gravity where interfacial forces were smeared-out on 3 cell size off the interface. The water droplet with radius 1/300 and of density $\rho_0 = 1000$ is surrounded in air of density $\rho_1 = 1.226$ in a rectangular domain of .02 wide and .06 high which is resolved on $50 \times 150$ mesh net. Density was smeared-out on 3 cell size off the interface and constant time step $2.5 \times 10^{-4}$ seconds was set for the entire simulation. Surface tension
coefficient between air and water is 0.0728 and gravity constant was set as 9.81. The simulation was first carried out by considering water viscosity \( \mu_0 = 1137 \times 10^{-6} \) and air viscosity \( \mu_1 = 178 \times 10^{-7} \) and then both viscosities were set to zero to compare average gravitational acceleration with the gravity constant. Results shown in Figure 18 are comparing two different methods for implementation of interfacial forces. We showed in Section 2.8.2 that normal implementation of gravity for two-phase flows encounters permanent error. As it is seen in Figure 18, the very same error decelerates the falling droplet when gravity is applied on the entire domain. The average acceleration in inviscid simulation for time 0.05 seconds is 9.798 and 9.760 for piezometric and regular pressure implementation methods, respectively. Considering the small size of droplet and small velocity values in the absent of viscosity, piezometric method gained more accurate acceleration than regular implementation of gravity forces.

Although CIF implementation of gravity for physical cases is quite successful and straightforward, SIF implementation methods encounter more complicated issues. Further studies are presented in the next chapter. In that part, we will also be able to prove why delta function produces inconsistent results when used in CIF formulation.
Figure 18. Results obtained for a falling water droplet using piezometric formulation and regular implementation of gravity for pressure shown with solid line and dashed-dotted lines, respectively. CIF method was considered for implementation of the interface. The shown counters are 0-level Set functions at four different times: 0, .0541, .0751 and .0951 seconds.

Reference


Appendix II-A

In Figure I a non-uniform mesh is depicted so that the middle control volume is larger than the east control volume. It is common for single phase flows to calculate the pressure gradient at node “P” as:

\[
\left. \frac{d\hat{p}}{dx} \right|_P = \frac{\hat{p}_e - \hat{p}_w}{x_e - x_w} = \frac{1}{x_e - x_w} \left\{ \eta_e\hat{p}_E + (1 - \eta_e)\hat{p}_P - \frac{\eta_w\hat{p}_E + (1 - \eta_w)\hat{p}_P}{\hat{\rho}} \right\} \tag{A1}
\]

where

\[
\eta_e = \frac{x_e - x_p}{x_e - x_w}; \quad \eta_w = \frac{x_p - x_w}{x_p - x_E}
\tag{A2}
\]

Therefore, the pressure gradient is basically an uneven combination of pressure values on three nodes “W”, “P” and “E”. Now, if we calculate the pressure gradient for single phase flow with the same methodology we used in Section 4 for uniform mesh, the pressure gradient reads as:

\[
\left. \frac{d\hat{p}}{dx} \right|_P = \frac{1}{2} \left[ \left. \frac{d\hat{p}}{dx} \right|_e + \left. \frac{d\hat{p}}{dx} \right|_w \right] = \frac{1}{2} \left[ \left( \frac{\hat{p}_p - \hat{p}_E}{\hat{\rho}E(x_E - x_p)} \right) + \left( \frac{\hat{p}_w - \hat{p}_P}{\hat{\rho}W(x_p - x_W)} \right) \right] \tag{A3}
\]

Clearly this pressure gradient is not equivalent to what was derived in Equation (A1). Equation (A3) equals to Equation (A1) only if the distances between the three shown nodes are the same. In other word, Equation (A3) provides the same pressure gradient for node “P” independent of the location of east or west faces of the central control volume. This means changing location of any virtual face γ makes no difference in the final outcome of Equation (A3) whereas it can considerably change the final value of Equation (A1), especially if the pressure profile is nonlinear. Having compared the difference between these two equations for single phase flow, we can readily realize that Equation (A3) is not a suitable expression for two phase flow as well. Equation (A1) uses pressure of control volume faces in order to find pressure gradient at cell center. However, in our method (and also in the balanced-force algorithm) pressure gradient at cell center must be estimated using pressure gradient at cell faces. If we estimate the desired pressure gradient using the same method we used for uniform mesh:

\[
\left. \frac{d\hat{p}}{\rho dx} \right|_P = \frac{1}{2} \left[ \left. \frac{d\hat{p}}{\rho dx} \right|_e + \left. \frac{d\hat{p}}{\rho dx} \right|_w \right] = \frac{1}{2} \left[ \left( \frac{\hat{p}_p - \hat{p}_E}{\hat{\rho}E(x_E - x_p)} \right) + \left( \frac{\hat{p}_w - \hat{p}_P}{\hat{\rho}W(x_p - x_W)} \right) \right] \tag{A4}
\]
due to density variation in the numerical domain, Equation (A4) returns considerably different pressure gradient depending on the location of the virtual face $\gamma$ while the pressure difference between node “P” and “E” remains constant. In another word, Equation (A4) estimates the pressure gradient independent of the location of face. As a result, the available numerical schemes such as the balanced-force algorithm, developed for uniform mesh, are unable to balance interfacial forces for a non-uniform meshed domain. Our numerical experiments proved that the more non-uniform mesh is used, the more spurious currents occur around interface locations.

![Figure I. A schematic representation of three control volumes and their interfacial forces estimated on cell faces for a non-uniform node distribution.](image)

All in all, there are two drawbacks for methods similar to the balanced-force algorithm in collocated variable arrangements. Firstly, the concept of non-uniform mesh is difficult to be accommodated in these methods, since it needs unique pressure gradient calculation. Secondly, the interfacial forces must be calculated at control volume faces instead of control volume centers. The latter difficulty can cause inconsistency if the interfacial force is not smooth enough in some control volumes, loosing as a result some interface locality desirable for the algorithm.

To demonstrate the effect of non-uniform mesh, we simulate the same static drop described in Section 2.8. The only difference here is using non-uniform mesh with linear expansion of 1.03 in the vertical direction, shown in Figure II. We showed in Section 2.8 no spurious currents arise if uniform mesh is used for a static drop. Even though the degree of non-uniformity is comparably small, the spurious currents appear around interface, assuring as a result the above mathematical description of the effect of non-uniform mesh.
Figure II. Static drop resolved on $41 \times 82$ mesh net ($R/h=10$ and $R/h\approx20$ respectively) is simulated. The vertical resolution is decreasing with a linear expansion of 1.03. As a result of the non-uniform mesh, spurious currents (in the order of $10^{-7}$) arise around the interface.
Chapter 3

Sharp Implementation of Interfacial Forces

3.1. Introduction

In the previous chapter, piezometric pressure concept [10] was numerically incorporated in two-phase flow equations. Based on that formulation, continuous implementation of interfacial forces, CIF, was introduced and examined for several test cases. Being more complicated, further elaboration on sharp implementation of surface forces, SIF, was left to be discussed in this chapter.

As explained in the last chapter, sharp interfacial forces included both the effects of surface tension forces and gravity forces. As a result, in implicit two-phase flow equations a forcing term was emerged which was formulated utilizing a Delta function. The Delta function, therefore, needs to be accurately discretized at the interface locations.

The most commonly used method for discretizing Dirac delta function in flow equations is, so called, continuum method. In continuum methods the exact Dirac delta function is basically replaced with its regularized form. Methods such as Continuum Surface Force (CSF) of Brackbill et al. [2] is of the most famous methods in this category. A number of papers can be found in literature on the regularization of Dirac delta function and Heaviside function [1][3][14][19][20][21][22]. Tornberg and Engquist [19] analyzed the regularization of Dirac delta function of the singularities and extent their use for multi-dimensions. They also showed the use of distance functions in level-set methods to extend one dimensional regularization to higher dimensions may produce O(1) error. Having proved such error for level set methods; in [3] two methods are presented for constructing consistent approximations to Dirac delta function to be conveniently used in conjunction with level set methods. They showed that their approximation is at least first-order accurate. More recently, in [14] a second-order accurate method for discretizing the Heaviside and the Dirac delta functions on irregular domains is presented.
An alternative approach for discretizing the Direct delta function is using sharp methods similar to Ghost Fluid Method (GFM) [4][11][13], Sharp Surface Force (SSF) method [6][7], Sharp method of [18] and Areal method [16]. All these methods except the areal method are basically following the same idea as in the Ghost Fluid Method. A fundamental assumption among these groups of methods is that the free surface is oriented either vertically or horizontally. That is why in this paper we refer to these types of methods as Stair-Step methods. Unlike the GFM based method, areal method avoids such assumption by including the actual orientation of the free surface.

Since in this chapter, we aim to devise a suitable method for sharp interfacial forces, SIF, we examine both of these groups of methods. Among GFM based methods, we use the Sharp Surface Force (SSF) presented in [6] and its results will be compared with areal method presented in [16].

The SSF method [6] (Thereafter stair-step method) yields satisfactory results when it is used for sharply discretizing surface tension force. The essence of the method can be summarized as ignoring small variation in surface tension force for sake of considering the dominant variation. In other words, if surface tension force is considerably larger in one direction than the other direction, the smaller component of the force is ignored. Following the balanced force algorithm of [6] and implementing the estimated forces in staggered manner, the method converges to exact pressure jump for a stationary drop if curvature is assigned. The areal method which was first introduced in [16] was entirely implemented in collocated manner. The idea behind this method, unlike stair-step method, is to consider all the components of interfacial forces in all directions without considering any preferable direction. It was shown in [16] that considerably smoother pressure jump takes place in the vicinity of interfacial locations. Since areal method was designed and implemented for fully collocated numerical algorithm, surface forces were estimated at cell centers [16]. Therefore, the method needs to be adapted to the numerical framework we devised in the first part of this paper, as it requires estimation of interfacial forces at cell faces instead of cell centers.

In this chapter, we first briefly review necessary equations for two-phase flow equations in Section 3.2. In Section 3.3 we discuss both stair-step method and areal method and
demonstrate their advantages and disadvantages through some test cases. However, the detailed study on each of these methods and their solution strategies are described in Section 3.4. Besides, in this section we will be able to explain the difference between regularization of Delta function using regulized Heaviside function and its exact transform of the level set scalar. This clarifies why implementing regulized Delta function transform of the level set scalar leads to spurious current when used in CIF formulation explained in the first part of the paper. The paper will be concluded with some insights in Section 3.5 and a remedy for sharp implementation of interfacial forces. We tested the proposed method for a falling water droplet and Rayleigh-Taylor instability test case.

3.2. Governing Equations

Governing equations for two-phase flow are basically two different Navier-Stokes equations written for each phase of flow with dynamic boundary conditions at their interface. In this chapter we use the same equations derived in the last chapter where implicit Navier-Stokes equations in conjunction with level-set function govern the entire domain. To keep track of the fluid properties, a Heaviside function, \( I_0(x_i,t) \) such as:

\[
I_0(x_i,t) = \begin{cases} 
1 & \text{if } \varphi(x_i,t) < 0 \\
0 & \text{if } \varphi(x_i,t) > 0 
\end{cases}
\]  

is defined in which \( \varphi(x_i,t) \) is a distance function famous as Level-Set function [15][17] and is tracked with its time derivative:

\[
\frac{\partial \varphi}{\partial t} + U_i \frac{\partial \varphi}{\partial x_i} = 0
\]  

In addition complimentary indicator function, \( I_1 \), is defined as

\[
I_1 = 1 - I_0
\]  

by which the second fluid domain can be represented, if \( I_0 \) represents the first fluid. By means of these auxiliary functions, implicit form of Navier-Stokes equations and continuity equation can be derived as:
\[ \begin{align*}
\frac{\partial u_i}{\partial x_i} &= 0 \\
\frac{\partial}{\partial t}(\rho u_i) + \frac{\partial}{\partial x_j}(\rho u_j u_i) &= -\frac{\partial p}{\partial x_i} + \frac{\partial}{\partial x_j}
\left[ \mu \left( \frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right) \right] + \left[ (\rho_1 - \rho_0) g y + \sigma \kappa \right] \frac{\partial \eta}{\partial x_i}
\end{align*} \] (3-2-4)

in which

\[ \begin{align*}
\rho &= \rho_0 I_0 + \rho_1 I_1 \\
\mu &= \mu_0 I_0 + \mu_1 I_1 \\
\hat{p} &= \hat{p}_0 I_0 + \hat{p}_1 I_1
\end{align*} \] (3-2-5)

and \( \hat{p} \) is piezometric pressure which eliminates the necessity of introducing gravity force on the entire domain and confining it to interfacial nodes [10]:

\[ \hat{p} = p + \rho g y \] (3-2-6)

\( p \) is the conventional mechanical pressure appeared in the original single phase flow equations. As a result of employing the piezometric pressure, \( \hat{p} \), interfacial boundary conditions forms as, see Equation (2-5):

\[ \left[ (\rho_1 - \rho_0) g y + \sigma \kappa \right] \] (3-2-7)

wherein the first term enforces gravity effect at the interface and the second term imposes pressure jump due to the curvature of the interface. The pressure jump in the vicinity of interface is proportional to the radius of curvature, \( k \), of the interface. Curvature and unit normal vector for an interface can be estimated using any color function such as the level-set function [15]:

\[ k = -\nabla \left( \frac{\nabla \phi}{|\nabla \phi|} \right) \quad \hat{n} = \frac{\nabla \phi}{|\nabla \phi|} \] (3-2-8)

If the curvature is taken to be positive when its center lies in the region where \( I_0 \) is defined, the pressure jump is calculated as [12]:

\[ \Delta p = \sigma \kappa \] (3-2-9)
As it was presented in the last chapter, surface tension forces must be estimated at locations in which exact pressure gradient can be calculated using the available pressure information. This means, for instance, for a collocated arrangement, surface tension forces should be estimated on control volume faces, since pressure equation is discretized on cell centers and therefore the pressure gradients can be exactly calculated on cell faces.

![Figure 1. A schematic representation of three control volumes and their interfacial forces estimated on cell faces](image)

Figure 1 represents three control volumes and their interfacial forces along X-direction. The following equations exactly balance pressure gradient and surface force for any halfway control volume, shown with dashed lines in Figure 1:

\[
\begin{align*}
\frac{\dot{p}_P - \dot{p}_E}{\rho_e \Delta x} + \frac{F_e}{\rho_e} &= \frac{dV_e}{dt} \\
\frac{\dot{p}_W - \dot{p}_P}{\rho_w \Delta x} + \frac{F_w}{\rho_w} &= \frac{dV_w}{dt}
\end{align*}
\]  

As it is seen in this set of equations, surface force must be estimated at control volume faces. This requirement mandates to find X-direction forces on perpendicular faces to X-axis and find Y-direction forces in perpendicular faces to Y-axis. Clearly, for example, X-direction forces are estimated independently from Y-direction forces in a control volume while both forces will eventually be considered as body forces for the same control volume. Being obliged to calculate surface forces at cell faces means, for a collocated algorithm, finding surface forces at the cell centers violates the rule and will result in spurious currents.
3.3. Different methods for estimating sharp interfacial forces

The surface tension force which appeared in the implicit momentum equation, Equation (3-2-5):

\[ \sigma \frac{\partial \rho}{\partial x_i} \]  

(3-3-1)

can be implemented either as a sharp force or as several forces embodying the sharp force at a interface location. As stated earlier, we aim to merely elaborate sharp techniques in this paper. In our formulation, since we consider gravity as an interfacial force, we need to expand sharp surface tension force techniques to sharp interfacial force, SIF, techniques such that

\[ \left( \rho_1 - \rho_0 \right) g y + \sigma \frac{\partial \rho}{\partial x} \]  

(3-3-2)

constitutes the imposed dynamic boundary condition. Clearly, the methods for imposing this new term follows the same logic as sharp surface tension force methods, however, special care must be taken to ensure the stability of the methods. In this section, therefore, two different methods for sharp interfacial force, SIF, are detailed to investigate their characteristics.

3.3.1. Stair-step method

Stair-step method or SSF [6] refers to an implementation method in which one direction might be preferred to another direction. In other words, the dominant variations are considered the actual effect of the function in the price of ignoring the smaller variations. The method which is based on GFM formulation in [11] is derived by separately applying flow equations for each phase. Without loss of generality, by ignoring viscosity in flow equations:

\[
\begin{align*}
\left[ \frac{\partial u}{\partial t} \right]^L &= \left[ -\frac{\partial p}{\rho \partial x} \right]^L \\
\left[ \frac{\partial u}{\partial t} \right]^G &= \left[ -\frac{\partial p}{\rho \partial x} \right]^G
\end{align*}
\]  

(3-3-3)

Considering vertically oriented interface, Figure 2, and using the continuity condition at the interface:
Then the pressure jump condition requires that

$$p_i^L - p_i^G = -\sigma \kappa_i$$  \hspace{1cm} (3-3-5)$$

Including this condition and considering Equation (3-3-4):

$$\left[ \frac{\partial u}{\partial t} \right]^L = \left[ \frac{\partial u}{\partial t} \right]^G = \left[ -\frac{\partial p}{\rho \partial x} \right]^L = \left[ -\frac{\partial p}{\rho \partial x} \right]^G = -\frac{p_{i+1} - p_i}{\rho_{i+1/2}^L} - \sigma \kappa_i \frac{1}{\rho_{i+1/2}^L}$$  \hspace{1cm} (3-3-6)$$

This therefore means, in the implicit form of flow equation for two phase flow, the pressure term should be replaced with

$$-\frac{\partial p}{\rho \partial x} \rightarrow -\frac{p_{i+1} - p_i}{\rho_{i+1/2}^L} - \sigma \kappa_i \frac{1}{\rho_{i+1/2}^L}$$  \hspace{1cm} (3-3-7)$$

Or the surface tension term can be formulated as an external force in flow equations. The crucial assumption in the above formulation is the stair-step interface configuration for deriving Equation (3-3-4). Considering non stair-step like interface is not straightforward in this formulation. One should note the above formulation can be readily derived if Direct delta function is regulized as:
\[ \sigma \kappa \delta = \sigma \kappa \frac{1}{\Delta x} = \frac{\sigma \kappa \Delta y \Delta z}{\Delta x \Delta y \Delta z} = \frac{\sigma \kappa A_1}{\Delta \nu} \quad (3-3-8) \]

where \( A_1 \) is the area of the enclosed interface in the control volume. This means GFM method basically finds a volumetric force for one of the face in the interfacial control volume. Using the proof in Chapter 2, one can find similar discretization to the above formulation for this volumetric force, meaning that SSF method is inherently CSF method.

This method can be conveniently used for both SSF and SIF methods, since it uses a simple numerical arrangement. However, the more uneven a function is, the more uneven the results would be. The stair-step method suggests discretizing the interfacial surface force as (for \( X \)-direction):

\[
\left[ (\rho_1 - \rho_0) g y + \sigma \kappa \right] \frac{\partial l_0}{\partial x} \quad \text{Converts to} \quad \frac{(\rho_1 - \rho_0) g y + \sigma \kappa}{\Delta x} \text{Sign}\left(-\frac{d\varphi}{dx}\right) \quad (3-3-9)
\]

if there is a sign change in the level-set function of any two neighboring nodes in \( X \)-direction [5]. For instance, if interfacial force in \( X \)-direction of a cell face is desired then:

\[
\text{If } [\varphi(i,j) \times \varphi(i+1,j)] \approx 0 \quad \rightarrow \quad \left\{ \begin{array}{l}
[f_x]_{i+\frac{1}{2},j,k} = \frac{(\rho_1 - \rho_0) g y + \sigma \kappa}{\Delta x} \text{Sign}\left(-\frac{d\varphi}{dx}\right) \\
[f_y]_{i+\frac{1}{2},j,k} = 0
\end{array} \right. \quad (3-3-10)
\]

and the same applies for other directions. The sign function appeared in the discretization formula delineates the direction of the surface force. Figure 3 shows how the stair-step method estimates interfacial forces for cell faces. As it is shown in this figure, there is only a sign change in level-set function of node 1 and 2; therefore, an appropriate force needs to be applied at their common face. However, since there is no sign change for any two other nodes, the rest of cell faces obtain no force.
It is clearly seen that this method considers no force for all other control volumes while the interface passes through all the shown control volumes. The shown interface in Figure 3 is a sharp interface which might not happen in many applications. However, as the mesh resolution increases in a simulation, the possibility of arising such sharp interfaces increases.

Stair-step method can be easily implemented in our numerical algorithm. As an example for such implementation, a static drop with radius $R=2$ is positioned in a cubic domain having side length of eight units. The surface tension coefficient it taken to be 72 and the density inside the drop is 1 while the outside density is 1000. The exact pressure jump is then:

$$ (\Delta P)_{Exact} = \sigma \kappa = 72 \times \frac{1}{2} = 36 $$

The simulation was carried out on uniform mesh size, $h$, and mesh resolution for the presented case is $R/h=10$, see the previous chapter for simulation details. Figure 4 shows the results of the implementation of surface tension force in the absent of gravity force. The exact pressure jump with machine zero error in pressure jump and spurious currents was attained. Pressure jump here means the maximum pressure minus the minimum pressure on the entire domain. This test case is a standard test case which was also simulated in Chapter 2.
Figure 4. Sharp implementation of surface tension force using stair-step method for a static drop with mesh resolution of $R/h=10$ and density ratio of 1000; left: pressure contours, right: Pressure distribution along X-axis.

No spurious currents and exact pressure jump makes this simulation satisfactory. However, the effect of stair-step method is clearly projected in Figure 4(a) where jagged pressure contours happen instead of round pressure contours. It is not difficult to predict that the stair-step method can lead to a failure in certain types of simulations. To demonstrate one of these types of failures, a falling droplet is simulated. A droplet with radius $.1 \text{ m}$ and of density $\rho_1 = 1.225 \text{ kg/m}^3$ is surrounded in a light density fluid of density $\rho_1 = .1694 \text{ kg/m}^3$ in a rectangular domain of $.5 \text{ m}$ wide and $1.5 \text{ m}$ high. Time step was set $2.5 \times 10^{-4}$ second and surface tension coefficient as $3.13 \times 10^{-3} \text{ kgm}^{-1}\text{s}^{-1}$ [7]. The zero potential level is set the same as the initial location of the droplet center. Figure 5 shows the results of the simulation before the droplet disappeared in the lighter flow.
Figure 5. Results obtained for a falling droplet surrounded in a light density fluid for different time: a) 0, b) .1, c) .15 seconds.

The unphysical results prove the failure of the numerical method. The reason can be sought in uneven distribution of surface force around interface. Noteworthy, the asymmetric results are because of using the asymmetric 5\textsuperscript{th} order HJWENO [11] in level set calculations. Since in SIF methods gravity effects are also included in surface forces, the accurate distribution of surface forces are more crucial than in SSF methods in which surface forces were only encountered with surface tension forces. For this simulation, for example, the maximum effect of gravity force is

\[ \Delta \rho gy = (1.225 - .1694) \times 9.81 \times (1.3 - 1.2) = 1.035 \]

whereas the effect of surface tension force is
\[ \sigma \kappa = 3.13 \times 10^{-3} \times \frac{1}{0.1} = 3.13 \times 10^{-2} \]

which explains the inherent difference between SIF and SSF terms. Larger term and larger gradient in SIF terms mandates avoiding stair-step method in these types of simulations. To the best of our investigations, inaccurate force distributions around interface pinch the interface in the locations where surface forces obtain higher gradients. Our numerical experiments for a similar case but higher density values (~1000 for the heavy fluid), in which gravity forces considerably increases, confirmed an earlier failure than the above test case proving the lack of consistency in the numerical method. Figure 6 demonstrates nodal forces (arithmetically averaged from faces) for the first time step of this simulation. Rapid change in direction and magnitude of interfacial forces are clearly exposed in this figure.
Figure 6. Estimated interfacial force distribution using stair-step method for a droplet in a light density fluid at the first time step of the simulation; a) the entire domain b) the magnified droplet.

Briefly, although stair-step method might be a suitable technique for SSF methods, it is not a reliable method for SIF methods. To find a remedy for this drawback, we now examine areal method proposed by Pourmousa et al. [16] as an alternative method for sharp implementation of surface tension force.

### 3.3.2. Areal method

Pourmousa et al. [16] suggested a method for collocated flow algorithm [5] by which interfacial forces are smoothly distributed over interfacial control volumes. In their method, since interfacial forces needed to be estimated at control volume centers, they were able to consider the total interfacial force effect. In other words, in this method, in contrast to stair-step method, there is no preferred direction for interfacial force and therefore, at any location, all force directions are considered.
Briefly, in this method the enclosed surface area, \( A_e \), within a control volume is first estimated and, based on the area, the surface force is calculated for the control volume. Then, the surface force is converted to a volumetric force for the desired control volume. To determine which control volumes should obtain force, a new logical condition is used to test whether any part of an interface passes through the control volume. The logical condition uses the distance between control volume center and the closest point at the interface - which is basically the value of level-set function - and compares it with the dimensions of the control volume. Figure 7 shows a control volume through which an interface passes. Using the same methodology presented in [16] the enclosed area within the control volume, \( A_e \), can be calculated, therefore, the surface force reads as:

Surface Force \( \vec{F}_s = \left( \rho_1 - \rho_0 \right) g y + \sigma x A_e \vec{n} \) \hspace{1cm} (3-3-11)

in which \( \vec{n} \) is the unit normal vector at the interface:

\( \vec{n} = (\eta_x, \eta_y, \eta_z) \) \hspace{1cm} (3-3-12)

then the surface force should be converted to a volumetric force \([2]\) which can be used in discretized form of flow equations

Volumetric Force \( \vec{F}_v = \frac{\left( \rho_1 - \rho_0 \right) g y + \sigma x A_e \vec{n}}{\Delta x \Delta y \Delta z} \) \hspace{1cm} (3-3-13)

As a result, the SIF method can be summarized as:

\[
\begin{align*}
If \ |\varphi(i, j)| < \frac{1}{2} \sqrt{(\Delta x)^2 + (\Delta y)^2} & \rightarrow \begin{cases} 
[F_Y]_x (i, j, k) = \frac{\left( \rho_1 - \rho_0 \right) g y + \sigma x A_e n_x}{\Delta x \Delta y \Delta z} \\
[F_Y]_y (i, j, k) = \frac{\left( \rho_1 - \rho_0 \right) g y + \sigma x A_e n_y}{\Delta x \Delta y \Delta z}
\end{cases}
\end{align*}
\] (3-3-14)
To compare the result of this type of implementation and the one of stair-step method, similar geometry to Figure 3 is considered in Figure 8. It can be seen instead of a single force in X-direction in Figure 3, there are several forces in X direction.

An illustrative example is shown in Figure 9 to compare the two methods of SIF. In this figure, gray control volumes indicate that an interfacial force in Y-direction is applied at those control volumes. In Figure 9(a), the logical condition of stair-step method is used to determine whether a control volume obtains any interfacial force. We used the logical condition to check if there is a sign change between cell faces of a control volume. It is shown increasing resolution slightly improves the uneven surface force implementation.
On the other hand, Figure 9(b) demonstrates a continuous graph of gray cells confirming the areal method establishes an even distribution of interfacial force. Figure 9(b) shows that increasing mesh resolution only improves the results. It will be shown that this continuous implementation of interfacial force is crucial for certain simulations where stair-step method fails to converge to a solution.
Figure 9. Comparing different methods for implementing interfacial forces. Colored cells indicate cells in which body force in Y-direction would be applied using: (a) Stair-step method (forces are applied on the faces where there is a sign change), (b) Areal method. In each case, resolution effect is tested.

As mentioned earlier, areal method was originally designed to calculate surface forces at nodal locations. The method aims to calculate surface forces at the cell center and then the nodal force is decomposed to find forces in each direction. Conversely, our numerical algorithm obliges us to estimate the interfacial forces at cell faces rather than cell centers. As a result, a strategy is needed to find face forces using areal method to be consistent with our algorithm.
Figure 10 shows a schematic representation of an interface crossing over a uniform mesh net. The X and Y-directions of surface forces must be estimated on the vertical and horizontal faces, respectively. Therefore, to estimate these forces, we introduce auxiliary control volumes. For example to estimate the X-direction force, $F_e$, in the vertical face between P and E, an auxiliary control volume is constructed between these two nodes; consequently the center of this auxiliary control volume lies exactly at the cell face between node “P” and “E”. As a result, the interfacial force is calculated at the face and its X-direction force is retained for this face. Similarly to estimate the Y-direction force for the horizontal face between node “P” and “N”, the auxiliary control volume between nodes “P” and “N” allows finding the surface force at this face and therefore the Y-direction of the force is retained. Using this strategy, we make sure to accurately calculate surface forces at the cell faces. Not to mention, as described earlier, interpolation from other locations to find surface forces at certain location is not permitted. This removes the option of estimating surface forces at cell centers and interpolating them to calculate forces for the cell faces. Our numerical experiments also proved that interpolation techniques are not useful.

![Figure 10. Mesh stencil is shown in vicinity of an interface. Numerical equations are discretized on the mesh stencil shown by solid lines and the auxiliary control volumes shown by dashed lines are used to estimate interfacial forces in different directions.](image)

We are now able to test the areal method in our numerical framework. The same drop solved in previous section is solved using areal method and the results are shown in Figure 11. Pressure jump- averaged pressure for cells $r \leq R/2$ minus averaged pressure for cells $r \geq 3R/2$
- is 35.85 and spurious currents are around 10^{-5}. Even though all the interfacial forces were estimated at face centers, spurious currents still occur around the interface. This means estimating surface forces at cell faces is not the sufficient requirement to eliminate spurious currents.

Figure 11. Sharp implementation of surface tension force using Areal method for a static drop with mesh resolution of R/h=10 and density ratio of 1000; left: pressure contours, right: Pressure distribution along X-axis.

To further investigate the effect of the areal method, the unsuccessful simulation of a falling droplet in the previous section is repeated here using the areal method. The successful results are shown in Figure 12 for different times.
The reason for the successful simulation can be sought by looking at smoothly distributed interfacial forces around interface, shown in Figure 13. Comparing force distributions of the stair-step method shown in Figure 6 and the one of areal method reveals difference between these two methods. Even though the areal method failed to eliminate spurious currents for a static droplet, it converged to an accurate solution for a test case which stair-step method is unable to capture. Not to mention, the spurious currents are usually very small which might not even be considered as a considerable source of error for many physical simulations.
3.4. Explanation for spurious currents for areal method

Despite the success of the areal method, it results in spurious current when it is implemented in our numerical algorithm, whereas the more crude method, stair-step method, results in no spurious currents for a static droplet. This made us to conduct some numerical experiments to investigate what other requirements must be met to assure a sharp interfacial model obtain no spurious currents.

3.4.1. Numerical experiments

An inclined virtual free surface is positioned in the corner of a rectangular box, Figure 14. A constant surface tension coefficient is assigned on the entire numerical domain ensuring a constant pressure jump across the stationary interface. This virtual test case allows us to convert...
a two-dimensional problem to a one-dimensional problem, as the direction of the surface force is perpendicular to the surface. If the equation of the inclined surface is:

\[ ax + by + c = 0 \]

then the distance between any point in the domain from the line would be:

\[ d = \frac{|ax_0 + by_0 + c|}{\sqrt{a^2 + b^2}} \]

and therefore a distance function can be found as

\[ \varphi(x_0, y_0) = \frac{ax_0 + by_0 + c}{\sqrt{a^2 + b^2}} \]

Surface tension coefficient is set to be 10 and a constant curvature \( \kappa = 1 \) is assigned to the inclined surface. To avoid any density ratio effect on simulations, a constant density is set for the entire domain. We now test three different cases.

**a) 45 degree inclined line on a uniform mesh**

The virtual interface is situated 45 degree and the numerical domain is meshed uniformly in both X and Y-directions, Figure 14(a). Figure 14(b) shows the obtained pressure jump across the interface. Surprisingly, no spurious currents occur in this simulation. For sake of comparison, interfacial forces are also depicted in Figure 14(c). Since interfacial forces are estimated on cell faces, therefore in Figure 14(c) these forces are arithmetically averaged to be shown on the cell centers (referred as nodal forces in the figure).
Figure 14. An inclined virtual interface is situated 45 degree on a uniform mesh causing a constant pressure jump; a) configuration of the interface and mesh net b) pressure jump across the interface c) Nodal forces are shown for a magnified part of the numerical domain

b) 30 degree inclined line on a uniform mesh

In this case, the virtual interface is tilted to 30 degree on a uniform mesh net, Figure 15(a). Even though in the previous case no spurious currents happened, in Figure 15(b) spurious current as large as $10^{-5}$ occurs. Figure 15(c) shows chaotic nodal forces for this case.
Figure 15. An inclined virtual interface is situated 30 degree on a uniform mesh causing a constant pressure jump; a) configuration of the interface and mesh net b) pressure jump across the interface and the resulted spurious currents d) Nodal forces are shown for a magnified part of the numerical domain

\[ \Delta x = 2 \Delta y. \]

This case tests the results of a similar interface to previous test case on a non-uniform mesh net. Since the interface is tilted 30 degree, the Y-direction is resolved two times more than X-direction. Surprisingly, spurious currents are eliminated and nodal forces are again well oriented perpendicular to the surface, Figure 16(b) and (c).
Consequently, it seems under certain conditions, the areal method arrives to an accurate solution such as in case (a) and (c). These conditions are to be found in the next section.

3.4.2. Mechanical interpretation

We began our analysis by comparing the stair-step method and the areal method. In Figure 17 a circular interface has been observed by both methods. Since the stair-step method uses the logical condition described in Equation (3-3-10), therefore the method maps the actual interface to a set of perpendicular lines which are orthogonal to the mesh net, Figure 17(a). In other words, the stair-step method estimates exactly the same surface forces for both the circular interface and...
the step-like interface. On the other hand, because of the logical condition in the areal method, Equation (3-3-14), it converts the circular interface to a set of small lines within each control volume, Figure 17(b).

Figure 17. Comparing areal method and stair-step method. a) Stair-step method reads a physical interface as perpendicular lines to mesh net b) Areal method reads a physical interface as small line segments within each control volume
We now estimate interfacial forces for a cell face in a control volume. In Figure 18 an interface crossing a control volume is shown. As it is demonstrated in the figure, stair-step method reads the interface as a straight line perpendicular to the line connecting node “P” and “E”. Therefore, the surface force is perpendicular to the interface and exactly passing through the center of the face of the control volume. Using the areal method, however, the interfacial force is perpendicular to the mapped interface read by this method. Since a force can be moved along its direction, we can therefore transfer the interfacial force until it intersects with the control volume face. Consequently, the X-component of the force is retained as the force applied on this face. As it is shown in the figure, for this case, the interfacial force does not pass through the center of the face while in our numerical algorithm we have to consider it as a force applied at the face center. This displacement from another location to face center is only permitted if the corresponding moment is also included into flow equations. As a result of ignoring these moments, the spurious currents occur when areal method is implemented. Clearly for a uniform mesh only if the angle of interface with X-direction is 45 degree, the interfacial forces exactly pass through the face center. One can easily examine this interpretation for case (b) and (c) described in the previous section.
Figure 18. Estimating the interfacial force for a face of a control volume, shown on top, using two different models: stair-step method and areal method.

Therefore, the reason that stair-step method obtains no spurious current is that all forces estimated in this method exactly pass the face centers; whereas it is only the case for areal method if the interface makes 45 degree angle with Cartesian co-ordinate(for a uniform mesh net). To demonstrate this, a static drop with very small surface tension force is simulated on a uniform mesh using the areal method. The results shown in Figure 19 demonstrate that for angles close to 45 degree there is no spurious currents while as the surface angle deviates from this optimum angle the more spurious currents occur. The above explanation can also be seen through the requirement of continuity equation:
\[
\left[\frac{\nabla p}{\rho}\right]^L \cdot \hat{n} = \left[\frac{\nabla p}{\rho}\right]^G \cdot \hat{n} \Rightarrow \left[\frac{\partial p}{\partial \varphi_x} n_x + \frac{\partial p}{\partial \varphi_y} n_y + \frac{\partial p}{\partial \varphi_z} n_z\right]^L \left[\frac{\partial p}{\partial \varphi_x} n_x + \frac{\partial p}{\partial \varphi_y} n_y + \frac{\partial p}{\partial \varphi_z} n_z\right]^G
\]

One can prove that the above equation holds for both case (a) and (c) whereas it does not hold for case (b).

Figure 19. Pressure contours and occurred spurious currents for a stationary droplet showing there are no spurious currents around 45 degrees. The pressure jump is .03 and spurious currents around \(10^{-8}\).

Using the same analogies one can understand why using the exact delta function in CSF method results in spurious current; while estimating the derivative of step function produces no spurious current [9]. Due to the accuracy of delta function, similar to areal method, it smears out the interface perpendicular to the interface and therefore similar problems like in areal method occur. While the numerical derivative of step function smears out the interface similar to the stair step method and therefore no spurious velocity occur. In Appendix A, we present simulations accrediting this argument.
3.5. Discussion and solutions

In the previous sections we showed the stair-step is an unstable method for sharp implementation of interfacial forces, SIF. Areal method, however, was used as an alternative method for stair step method and its successful simulation was presented. Unlike stair-step method, the areal method ignores no forces in any direction in order to simplify the estimation method. The method works quite well as far as all forces are estimated at the same location. Nevertheless, estimating surface forces in different locations and transferring their effects to cell center brings about some inconsistencies. One can show for co-located arrangement and uniform mesh, areal method produces an error proportional to interfacial term, numerical time step and the deviation from 45 degree:

\[ \Delta E \propto \left( (\rho_1 - \rho_0)g + \sigma \right) \mid \tan(\alpha) - \tan(45) \mid \Delta t \]

Therefore, for instance, we were unable to simulate Rayleigh-Taylor instability using this method. In Rayleigh-Taylor instability test, the initial angles are far from the optimum angle for a uniform mesh (45 degree), and therefore, the simulation was encountered instabilities. For any simulation in which the above error is significant, areal method encounters instabilities. Considering the success of areal method and failure of stair step method, a final remedy for SIF techniques would be a fully-collocated method in which interfacial forces can be calculated at the same location. This might suggest staggered variable arrangement; nevertheless, further research should be carried out. This, therefore, suggests that numerical methods in which interfacial forces must be estimated in different locations are not attractive methods for sharp implementation of interfacial forces.
As a remedy to these difficulties, a more stable and versatile method is proposed here for SIF techniques. In both stair-step method and areal method, the effects of interfacial forces are confined only to one of the face in each direction. However, our numerical experiments show that if we expand the effect of interfacial forces to a control volume, we are able to resolve the instabilities issues. Therefore, we smear out an interface on $0.25\Delta x$, see Figure 20, then use similar formulation used in CIF:

$$\{[f_y]_x\}_e = \left[\sigma \kappa_e + \left(\rho_1 - \rho_0\right) \cdot g \cdot y_{\text{effective}}\right] \cdot \frac{H(\phi_E) - H(\phi_P)}{\Delta x}$$

$f_y$ is the volumetric force in $X$ direction estimated on the east face of a control volume, see Figure 1. $H(\phi_E)$ and $H(\phi_P)$ are smeared step function for nodes “E” and “P”. Similarly, other forces for different faces can be estimated. In this way, the effects of interfacial forces are limited only to one control volume in each direction.

To test this method, the standard Rayleigh-Taylor instability described in the previous chapter is the first test case. The test was carried out on $64 \times 512$ uniform mesh. Its successful results are shown in Figure 21. The second test case is a falling drop moving under gravity effect.
Figure 20. A circular interface is expanded $.25\Delta x$ off its location.
A small water droplet surrounded in air, similar to the test case presented in chapter 2, is simulated in this section. The water droplet with radius 1/300 and of density $\rho_0 = 1000$ is surrounded in air of density $\rho_1 = 1.226$ in a rectangular domain of .02 wide and .03 high which is resolved on $50 \times 150$ mesh net. Constant time step $10^{-4}$ is set for the entire simulation. Surface tension coefficient between air and water is .0728. Gravity constant was set as 9.81 and
viscosity for both fluids was set to zero. Results shown in Figure 22 demonstrate the stable performance of the method.

Figure 22. Results obtained for a falling water droplet. The shown counters are 0-level Set functions at: a) 0, b) .02, c) .04, d) .06 seconds.
Appendix III-A

In the previous Chapter we mentioned that using exact smeared delta function in flow equations causes inconsistency for CIF techniques. An example of such inconsistency can be seen in simulation of a static droplet which was previously discussed in this paper, Figure I. In the first part, we showed if instead of delta function the derivative of step function is used no spurious current occur around interface.

Figure I. Simulation of a static drop using delta function in CIF. Spurious currents as large as 2.34 × 10⁻⁶ occur around interface when exact curvature is assigned.

We discussed in this paper that the reason for this spurious currents is keeping the sense of angle when delta function is used. To demonstrate that, we devise two test cases similar to what we described in Section 3.4. We integrate delta function in the formulation of CIF technique and solve each case, Figure II. One can see that the very same pattern happened for areal method, takes place here. There are no spurious currents for 45 degree while they appear for 30 degree inclined slope. Similar to areal method, when smeared delta function is used, there is no correlation between interfacial forces between two different faces of a control volume.
Similar to areal method, Delta function basically smears out the interface perpendicular to it, and therefore, it keeps the sense of angle. For instance in Figure III, the dashed line in one of the iso-surfaces of a smeared delta function. Clearly, since its normal distance to interface is the same for all points along this iso-surface, the value of delta function is also the same for all the points along it. As seen in this figure, this iso-surface does not pass through point “n”, as the normal distance of point “n” is different than the normal distance of point “e”. Consequently, when interfacial forces are estimated at points “e” and “n”, they use different values for delta function, whereas when derivative of step function is used, its inaccurate values map the interface very similar to stair-step method. However, if a 45 degree interface was considered in Figure III, the very same iso-surface which passes thorough “e”, would pass through point “n”. That is why there is no spurious current for case (a) in Figure II.
Figure II. Application of smeared delta function in formulation of CIF. a) A 45 degree interface resolved on uniform mesh; b) A 30 degree interface is resolved on uniform mesh

Figure III. Schematic of a control volume and an interface. The interface is shown with solid line and the dashed line shows an iso-surface of its delta function

Reference


Part II

Turbulent Flows
Chapter 4

Implicit Direct and Large Eddy Simulations of Turbulent Flows

4.1. Introduction

Direct and large eddy simulations of turbulent flows have been developed for a number of applications using different methods and techniques. These methods are characterized with their spatial and time discretization methods, numerical schemes and flow algorithms. Due to sensitive nature of turbulent simulations, researchers have been studying the effects of different techniques in variety of frameworks such as spectral method, finite difference, finite element and finite volume.

Despite the higher accuracy of spectral methods, finite difference and finite volume methods are becoming more popular for direct and large eddy simulations. An advantage of these techniques lies in the ease with which they can be applied to complex geometries. To the best of our knowledge, finite difference methods emerged earlier and has received more attention than finite volume methods, since implementing higher order numerical schemes is easier for these types of methods. Besides, using midpoint-rule approximation of the surface integral in finite volume methods always defaults it to second-order accuracy. Therefore, they seem less promising for high-order accuracy simulations. Finite volume techniques, however, benefit from its conservative properties which can be considerably important both for large and direct simulation of turbulent flows.

Rai and Moin presented an extensive study for the reliability of finite difference methods for direct simulation of turbulent flows in [46]. They compared a kinetic-energy-conserving type of central difference with a high-order-accurate upwind scheme. Despite the dissipative nature of upwind schemes, they recommended high-order-accurate upwind scheme for direct simulations of turbulent flows when finite-difference schemes are used. On the other hand, similar studies for
large eddy simulations revealed different perspectives [32][33][53]. Tafti [53] suggested conservative central difference discretization as a suitable scheme for LES simulation because it maintains the integrity of the resolved spectrum better than high-order upwind methods. Whereas, 5th order non-conservative upwinding discretization of convective terms and 2nd order central difference discretization for pressure equation were recommended for DNS. Furthermore, Tafti also found little evidence to justify the computational cost of high-order treatment of gradient operators. The paper also provided evidences which using high-order conservative treatment of convection terms, in spite of the second order finite volume operator exhibits high order accuracy. Najjar and Tafti [33] also found that the dynamic procedure in implementing dynamic turbulent models senses the dissipative nature of the non-conservative upwind biased approximation and returns a lower turbulent viscosity than with a second order central difference approximation. They also indicated that the inclusion of the dynamic model or any subgrid scale model with finite-difference approximations will not always result in better predictions of the flow field.

As stated earlier, finite volume simulations of turbulent flows have received less attention than finite difference techniques, some of the works using finite volume technique can be found in [26][29][48][58][59][51][11]. Smagorinsky-based models were evaluated in the framework of Finite-Volume by Majander and Siikonen [26]. They found that none of the tested models improves the results of low Reynolds number simulation for any grid level compared to the calculations with no model; however, at the higher Reynolds number the subgrid-scale models stabilize the computation. They analyzed the accuracy of a resolved field and realized that the discretization error overwhelms the subgrid term at $Re_b = 2800$ in the most part of the computational domain. Error-assessment of central finite-volume discretizations in large-eddy simulation was studied in [29]. They examined second and fourth order central difference discretization for convective and viscous terms (including subgrid-scale models), denoted as 2 and 4 respectively. They used different combinations for convective and viscous terms represented as 2-2, 2-4, 4-4 and 4-2. They observed that use of ‘equal-order’ finite-volume methods (2–2 and 4–4) leads to better results in comparison to mixed-order combinations (2-4 and 4-2). 2–2 scheme generally obtains the smallest errors at coarser resolutions, closely followed by the 4–4 scheme. They concluded that, at coarse resolutions, the asymptotic error behavior as expressed by the order of the spatial discretization is not a suitable indicator for the
total error in large-eddy simulations. Xu et al. [57] propose a finite volume method with compact fourth order accuracy scheme for large eddy simulation (LES), showing that the higher accurate finite volume method for LES is proved to be a promising numerical method. In [59] the feasibility of an optimal finite-volume large-eddy simulation (LES) model for isotropic turbulence was assessed. They used a stochastic estimate of the fluxes in a finite-volume framework to approximate the ideal LES. Furthermore, the variational multiscale (VMS) approach is applied to the Smagorinsky model and developed within the context of unstructured finite-volume solver in [48].

Explicit numerical algorithms have been extensively used both for direct and large eddy simulations, DNS and LES respectively. Perhaps the most common algorithm used for this group of simulations is the fractional step method of Kim and Moin [20]. Explicit algorithms are justified both for DNS and LES, since small time steps are required to accurately resolve flow details. However, it was shown in [7] that time step of explicit methods is quite stringent for such simulations. Kim et al. [21] used $0.0676\nu/u_r^2$ as the computational time step for their explicit algorithm of simulating channel flow, where $\nu$ is kinematic viscosity and $u_r$ is wall-shear velocity. Whereas, Choi and Moin [7] obtained successful simulations even when they used $1.2\nu/u_r^2$ as the time step for their fully implicit algorithm. They showed that this time step is close to the half of Kolmogorov time scale in their simulation of turbulent channel flow. Furthermore, Hou and Mahesh [16] overcome the acoustic time-scale limit in their implicit algorithm for direct and large eddy simulation of compressible turbulent flows. Moreover, Park [35] studied the effects of time-integration method using PISO Algorithm. The standard PISO algorithm is basically a modification of SIMPLE type algorithm for unsteady simulation of a flow field and uses EULER implicit scheme for time integration. Seeking second order accuracy in time, Park [35] modified the algorithm and compared its simulation results with the fractional step method. It was demonstrated that the modified PISO algorithm successfully predicts turbulent flows while using time steps considerably higher than time step restriction of explicit methods. In addition, it was shown if the first order implicit EULER scheme is used, the time step has to be smaller than $2\nu/u_r^2$. Consequently, it seems expensive implicit algorithms can be justifiable for simulation of direct and large eddy simulations.
Although both the PISO and fractional-step methods are designed for unsteady simulations, due to the splitting of the convection-diffusion and projection steps, they have an error proportional to square root of time step, $\Delta t^2$ [12]. Therefore, they are best suitable for explicit time integration rather than implicit time integration. This splitting operator has also brought about concerns regarding first-order accuracy in time for pressure [44][56]. Not to mention, these types of methods encounter difficulties when the boundary velocity is discontinuous, as demonstrated in [44].

Considering all these, one might look for an alternative robust algorithm which avoids these errors and also is suitable for implicit time integration expandable for higher order accuracy. Perhaps, SIMPLE-type algorithms [39] are suitable candidates satisfying all these criteria. SIMPLE-type algorithms, unlike fractional step methods, do not produce the splitting error when large time steps are used and also higher order accuracy in time for pressure can be readily achieved. Besides, concerns regarding suitable boundary conditions for intermediate velocity field as well as frustrations due to discontinuity of velocity field have no room in these types of algorithms. Hence, they seem to be a promising approach for DNS and particularly for LES simulation of complex geometries and generic flow fields.

Over years, SIMPLE-type algorithms have been extensively used and developed for different types of flow simulation [27][49][19][4][45][9][28][25][17], but received considerably less attention for direct and large eddy simulations [32][52]. In addition to multigrid techniques [22][31][34][52], recently, new techniques have been proposed to accelerate the convergence and increase the accuracy of these types of methods for unsteady simulations [18][52]. An interested reader can find an extensive review of these types of methods in [1].

In general, within the framework of finite difference or finite volume methods, the above review and discussion can be summarized as:

i) Implicit time integration methods considerably reduce computational time for DNS and LES simulations [7][35].

ii) High-order upwinding discretization of convective terms and low order central difference discretization of other operations are optimized schemes for DNS simulations [53][46].
iii) Equal-order central difference schemes, 2-2 and 4-4, are recommended for LES simulations, with little ambition for higher order schemes [53][29].

iv) There are evidences that second order finite volume operator does not prevent obtaining higher order accuracy for turbulent flows [53].

v) SIMPLE-type algorithms are not encountered with the problems caused by the splitting operator in fractional step methods, hence, higher accuracy and also more generic formulation is expected from these types of algorithms.

All these conclusions convince us to assess the performance of the standard SIMPLE algorithm, as it has the potential to be a suitable ground for including each of these statements. Despite the conventional perception of SIMPE-type algorithms, i.e. being unsuitable for LES and DNS simulations, their inherent characteristics promise successful simulations. Perhaps expensive implicit time integration and difficulties for expanding their accuracies in the framework of finite volume method have been the main reasons which have made researchers reluctant to use these types of algorithms. However, the above study does not support these barriers. To the best of our knowledge, these algorithms have received little attention in literature for simulations of direct and large eddy simulations.

Considering all these, we therefore, in this chapter evaluate the performance of the standard SIMPLE algorithm, as it has not been particularly studied. As recommended in literature, we first use second order central difference discretization for all operators and then we will test third order QUICK scheme for convection terms. We also evaluate the effect of different turbulent models for LES simulations. Moreover, the effects of time step on the simulations are also presented both for LES and DNS simulations. We take advantage of conservative properties of control volume framework, in order to discretize the flow equations.

### 4.2. Governing equations

#### 4.2.1. Standard Filtered Flow Equations

We start with a space-time convolution filter operation $L$ and assume an unbounded domain. The filter $L$ is linear and its operation is defined by [24][14][47]:
\[ \tilde{f}(x, t) = L(f)(x, t) = \int_{-\infty}^{+\infty} dt' \int_{-\infty}^{+\infty} G(x-x', t-t')f(x', t')dx' = G \ast f \] (4.2.1)

where

\[ \int_{-\infty}^{+\infty} dt' \int_{-\infty}^{+\infty} G(x-x', t-t')f(x', t')dx' = 1 \] (4.2.2)

Convolution filters verify the following commutative properties:

\[ \overline{\partial_i f} = \partial_i \tilde{f} \quad ; \quad \overline{\partial_t f} = \partial_t \tilde{f} \quad i = 1, 2, 3 \] (4.2.3)

Where \( \partial_t \) and \( \partial_i \) denote time and special derivatives. If we define the commutator \([f, g]\) of two operators \( f \) and \( g \) as:

\[ [f, g] \Phi = f \circ g(\Phi) - g \circ f(\Phi) \] (4.2.4)

Then we can express the commutative properties in Equation (4.1.3) as:

\[ [\mathcal{L}, \partial_t](f) = 0 \quad ; \quad [\mathcal{L}, \partial_i](f) = 0 \quad i = 1, 2, 3 \] (4.2.5)

It may be shown that a linear integral filter commutates with partial differentiation if the kernel is a function of \( x - x' \) only, i.e., of convolution type [14].

Developing LES for complicated flow simulations requires using non-uniform filters to overcome geometrical and dynamical complexities. In contrast to the convolution filters, non-uniform filters do not necessarily commutate with partial differentiation. An extended filter \( L \) in one spatial dimension is defined by:

\[ \tilde{f}(x, t) = \int_{x-D_-(x)}^{x+D_+(x)} \frac{G(x, \xi)}{\Delta(\xi)} f(x, \xi) d\xi \] (4.2.6)

and filter-width is defined as:

\[ \Delta = \Delta_+ + \Delta_- \] (4.2.7)

where \( \Delta_+, \Delta_- \) denote the upper and lower bounding functions. Applying this filter may bring new terms in the flow equations that increase the complexity of simulation since:
Hence, filtering the flow equation using non-uniform filters yields [14]:

\[
\partial_t \bar{u}_i + \partial_j (\bar{u}_i \bar{u}_j) + \partial_i \bar{p} - \frac{1}{Re} \partial_{jj} \bar{u}_i \]

\[
= -\partial_j \left( [L, \Pi](u_i, u_j) \right) - \left[ [L, \partial_i](p) - \frac{1}{Re} [L, \partial_{jj}](u_i) + [L, \partial_i] \left( \Pi(u_i, u_j) \right) \right]
\]

(4.2.9)

Where

\[
\Pi(u_i, u_j) = u_i u_j
\]

(4.2.10)

The first term on the right hand side of the equation is the turbulent stress tensor. The remaining unclosed terms so called ‘commutator errors’ are usually neglected in most LES simulations [47]. On the other hand, applying a convolution type filter to flow equations, filtered Navier-Stokes equations considerably simplifies the above flow formulation:

\[
\partial_t \bar{u}_i + \partial_j (\bar{u}_i \bar{u}_j) + \partial_i \bar{p} - \frac{1}{Re} \partial_{jj} \bar{u}_i = 0
\]

(4.2.11)

If we introduce \( \tau_{ij} \), the turbulent stress tensor as [14]:

\[
\tau_{ij} = \bar{u}_i \bar{u}_j - \bar{u}_i \bar{u}_j = L \left( \Pi(u_i, u_j) \right) - \Pi \left( L(u_i), L(u_j) \right) = [L, \Pi](u_i, u_j)
\]

(4.2.12)

in which the commutator \([f, g]\) of two operators \( f \) and \( g \) applied to the dummy variable is defined as:

\[
[f, g] = f \circ g(\Phi) - g \circ f(\Phi)
\]

(4.2.13)

Then the flow equation reads as:

\[
\partial_t \bar{u}_i + \partial_j (\bar{u}_i \bar{u}_j) + \partial_i \bar{p} - \frac{1}{Re} \partial_{jj} \bar{u}_i + \partial_j \tau_{ij} = 0
\]

(4.2.14)

Turbulent stress tensor \( \tau_{ij} \) or its dynamic consequence \( \partial_j \tau_{ij} \) should be modeled based on resolved velocity field. These models so called ‘subgrid’ models primarily involve structures
smaller than the filter-width $\Delta$. In most present-day simulation studies $\Delta$ is directly related to grid spacing, $h$, e.g., $\Delta=2h$. However, for most common turbulent models, $\Delta$ appears as a result of considering it as a typical length scale. The core of LES is constituted based on capturing all motions larger than $\Delta$, therefore, it is quite understandable to have this length scale as part of modeling. In some models, however, such as similarity models this length scale is not explicitly used, yet its actual effect appears in an explicit filtering procedure.

Considering physical concept of filtering in LES, the filters must be local to filter out local disturbances in their domain. The most commonly used filter functions are [14][47]:

**Top-hat Filter**

$$G_{th} = \frac{1}{l}(H(z + l/2) - H(z - l/2)) = \begin{cases} \frac{1}{l} & \text{if } |z| < \frac{l}{2} \\ 0 & \text{Otherwise} \end{cases} \tag{4.2.15}$$

**Gaussian filter**

$$G_g = \left(\frac{a}{\pi l^2}\right)^{1/2} \exp\left[-\left(\frac{a z^2}{l^2}\right)\right] \tag{4.2.16}$$

**Spectral cut-off filter**

$$G_{Co}(x - \xi) = \frac{1}{l} \frac{\sin(\pi(x - \xi)/l)}{\pi(x - \xi)/l} \tag{4.2.17}$$

### 4.2.2. Different Models

Employing eddy-viscosity concept, Smagorinsky model is considered as one the very first modeling procedure for LES. Based on molecular viscosity, this model adds additional viscosity to flow equation. If a Reynolds-type averaging operator is used, the turbulent stress tensor can be expressed as:

$$\tau_{ij} = (\overline{u_i} + u_i')(\overline{u_j} + u_j') - \overline{u_i u_j}$$

$$= (\overline{u_i u_j} - \overline{u_i} \overline{u_j}) + (\overline{u_i u_j} + \overline{u_i' u_j'}) + \overline{u_i' u_j} = \overline{u_i u_j} \tag{4.2.18}$$
Although the LES filtering doesn’t strictly holds Reynolds properties, it is believed that the philosophy behind this model is relevant enough to capture subgrid scales motions. Assuming $\Delta$ as a suitable length scale and $\Delta|\mathbf{S}|$ as velocity scale, turbulent viscosity reads as:

$$\nu_t \approx (\bar{u})(\bar{I}) \approx (\Delta|\mathbf{S}|)(\Delta)$$

$$S_{ij} = \partial_i u_j + \partial_j u_i \quad ; \quad S = .5S_{ij}S_{ij}$$

$$\nu_t = (C\Delta|\mathbf{S}|)\Delta = (C_s\Delta)^2|\mathbf{S}|$$ (4.2.19)

Consequently,

$$\partial_t \bar{u}_i + \partial_j (\bar{u}_i \bar{u}_j) + \partial_i \bar{p} - \left[ \frac{1}{Re} + \nu_t \right] \partial_{jj} \bar{u}_i = 0 \quad ; \quad \partial_i \bar{u}_i = 0$$ (4.2.20)

$$\nu_t = (C_s\Delta)^2|\mathbf{S}|$$ (4.2.21)

$C_s$ is called Smagorinsky constant. Over years, various values for $C_s$ have been suggested. For homogeneous isotropic, for instance, turbulence $C_s$ is suggested as .17 [14] yet in most application .1 is used [24][14]. This constant can have significant effect on flow simulation. Over predicted value damps all turbulent fluctuations resulting in laminar solution for the flow field.

Although the Smagorinsky model is widely used, it doesn’t properly capture the instantaneous structures arising in the turbulent stresses. A poor correlation coefficient between Smagorinsky model and actual turbulent stress of only .3 is observed in several flows. Numerically, this model is too dissipative and needs careful attention to obtain accurate results [47].

Similarity models are one of the alternatives for eddy-viscosity models. These types of models are designed based on the assumption of similarity between turbulent flow field structures for different length scales. Similarity models arise by applying the turbulent stress definition $\tau_{ij} = [L, \Pi]$ directly to the resolved velocity components. Filter width $\Delta$ doesn’t appear explicitly in this formulation. In this context, assumptions such as alignment of turbulent stress tensor and rate of stress tensor $S_{ij}$ are not considered. The similarity model can be formulated in terms of filtered velocities [14][5]:
\[ m^B_{ij} = [L, \Pi]\left( \overline{u_i} , \overline{u_j} \right) = \overline{u_iu_j} - \overline{\hat{u}_i\hat{u}_j} \quad (4.2.22) \]

In a more general form it can also be represented as:

\[ m^B_{ij} = [L_1, \Pi]\left( L_2(\overline{u_i}) , L_2(\overline{u_j}) \right) \quad (4.2.23) \]

\( L_1, L_2 \) can be any two filter operators. Similarity models have shown significant increase in correlation coefficient with actual turbulent stress tensor. This has been reported in the range .6 to .9 [14].

Combining eddy-viscosity and similarity model concepts is generally called mixed models. These mixed models basically aim to find the most optimum values for model coefficients by allowing them to vary in space and time. A well known identity, Germano’s Identity, congregates the formulation for these models. This identity expresses how two types of filters correlate and commutate with the operator \( \Pi \) [14]:

\[ [L_1L_2, \Pi] = [L_1, \Pi]L_2 + L_1[L_2, \Pi] \quad (4.2.24) \]

Using Germano’s identity:

\[ \frac{[L_1L_2, \Pi]}{T_{ij}} = \frac{[L_1, \Pi]L_2 + L_1[L_2, \Pi]}{\hat{T}_{ij}} \quad (4.2.25) \]

\[ T_{ij} - \hat{T}_{ij} = R_{ij} \]

\[ T_{ij} = \overline{\hat{u}_i\hat{u}_j} - \overline{\hat{u}_i\hat{u}_j} \]

\[ R_{ij} = \overline{\hat{u}_i\hat{u}_j} - \overline{\hat{u}_i\hat{u}_j} \quad (4.2.26) \]
Here, \( (\cdot) \) denotes LES-filter with width \( \Delta \) and \( (\cdot) \) is, so called, ‘test’ filter with width \( \hat{\Delta} \). The width of the combined filter \( (\cdot) \) can be defined exactly for two Gaussian filters as \( \Delta^2 = \hat{\Delta}^2 + \Delta^2 \); this relation can be optimal for other filters such as Top hot filter. In this type of modeling only the ratio \( \Delta / \hat{\Delta} \) needs to be adjusted which is commonly set equal to 2. It is supposed to consider two suitable models for each of \( T_{ij}, \hat{T}_{ij} \). If we consider both of the “mixed” type:

\[
m_{ij} = a_{ij} + c b_{ij} \quad ; \quad a_{ij} = a_{ij}(\bar{u}) \quad b_{ij} = b_{ij}(\bar{u})
\]

\[
M_{ij} = A_{ij} + \mathcal{C} B_{ij} \quad ; \quad A_{ij} = a_{ij}(\hat{\bar{u}}) \quad B_{ij} = b_{ij}(\hat{\bar{u}})
\]  \hspace{1cm} (4.2.27)  \hspace{1cm} (4.2.28)

Where \( a_{ij}, b_{ij} \) are basic models, e.g., Smagorinsky model. Essential Assumption \( \mathcal{C} \approx c \) implies the similarity between the basic level filter and combined level filter. By substituting it back into Germano’s identity:

\[
M_{ij} - \hat{M}_{ij} = R_{ij}
\]

\[
\frac{(A_{ij} + \hat{a}_{ij}) + c (B_{ij} + \hat{b}_{ij})}{\mathcal{Y}_{ij}} = R_{ij}
\]

\[
\Rightarrow c \beta_{ij} = R_{ij} - \mathcal{Y}_{ij}
\]  \hspace{1cm} (4.2.29)  \hspace{1cm} (4.2.30)  \hspace{1cm} (4.2.31)

Using optimization for finding minimum error for:
\(\varepsilon(c) = \langle 5\{(R_{ij} - Y_{ij}) - c\beta_{ij}\}^2 \rangle \) \hspace{1cm} (4.2.32)

Yields to:

\[ c = \frac{\langle (R_{ij} - Y_{ij})\beta_{ij} \rangle}{\langle \beta_{ij} \rangle} \] \hspace{1cm} (4.2.33)

In which \(\langle \rangle\) usually defines an averaging operator on homogeneous direction.

Dynamic eddy-viscosity model can be formulated, if the basic model is chosen as Smagorinsky model, [14][47][42]:

\[
\begin{align*}
\tau_{ij} - \frac{\delta_{ij}}{3} \tau_{kk} &\approx m_{ij} = -2c\tilde{\Delta}^2 |\bar{S}|\bar{S}_{ij} \\
T_{ij} - \frac{\delta_{ij}}{3} T_{kk} &\approx M_{ij} = -2\bar{C}\tilde{\Delta}^2 |\bar{S}|\bar{S}_{ij} \\
\end{align*}
\] \hspace{1cm} (4.2.34)

\[
\begin{align*}
T_{ij} &= \bar{u}_i \bar{u}_j - \bar{u}_i \bar{u}_j \\
\tau_{ij} &= \bar{u}_i \bar{u}_j - \bar{u}_i \bar{u}_j \\
\end{align*}
\] \hspace{1cm} (4.2.35)

\[ R_{ij} = T_{ij} - \hat{r}_{ij} \Rightarrow R_{ij} = \bar{u}_i \bar{u}_j - \bar{u}_i \bar{u}_j \] \hspace{1cm} (4.2.36)

This means in general form \(Y_{ij} = 0\), therefore:

\[ \beta_{ij} = B_{ij} + \hat{b}_{ij} = \tilde{\Delta}^2 |\bar{S}| \bar{S}_{ij} - \bar{C}\tilde{\Delta}^2 |\bar{S}|\bar{S}_{ij} \] \hspace{1cm} (4.2.37)

\[ c = \frac{\langle R_{ij}\beta_{ij} \rangle}{\langle \beta_{ij} \rangle} \] \hspace{1cm} (4.2.38)

Hence, the final formulation finds the following form:

\[
\begin{align*}
\tau_{ij} - \frac{\delta_{ij}}{3} \tau_{kk} &\approx m_{ij} = -2c\tilde{\Delta}^2 |\bar{S}|\bar{S}_{ij} \\
\partial_t \bar{u}_i + \partial_j (\bar{u}_i \bar{u}_j) + \partial_i \bar{p} - \frac{1}{\text{Re}} \partial_{jj} \bar{u}_i + \partial_j \tau_{ij} &= 0 \\
\end{align*}
\] \hspace{1cm} (4.2.39)
In this paper, to implement the test filter the following numerical structure was employed with the trapezoidal rule [59]:

\[
\ddot{\tilde{\rho}}_{i,j,k} = .25(\ddot{\tilde{\rho}}_{i+1,j,k} + 2\ddot{\tilde{\rho}}_{i,j,k} + \ddot{\tilde{\rho}}_{i-1,j,k})
\]

\[
\dddot{\tilde{\rho}}_{i,j,k} = .25(\dddot{\tilde{\rho}}_{i,j+1,k} + 2\dddot{\tilde{\rho}}_{i,j,k} + \dddot{\tilde{\rho}}_{i,j-1,k})
\]

\[
\ddot{\tilde{\rho}}_{i,j,k} = .25(\ddot{\tilde{\rho}}_{i,j,k+1} + 2\ddot{\tilde{\rho}}_{i,j,k} + \ddot{\tilde{\rho}}_{i,j,k-1})
\] (4.2.40)

4.3. Channel flow simulation

4.3.1. Periodic boundary condition

A symmetric channel flow was simulated using periodic boundary condition in two homogenous direction X and Z.

![Figure 3. Periodic boundary condition for channel flow](image)

Periodic boundary condition is usually used to produce fully developed regime in a limited numerical space [38]. All parameters are considered periodically constant except pressure, since
it drives the flow. Hence, pressure should be reformulated such that it retains the same property.
Considering $L_x$ as the channel length in X direction, we can write:

$$p(x,y) - p(x + L_x, y) = p(x + L_x, y) - p(x + 2L_x, y) = \cdots \quad (4.3.1)$$

By Defining

$$\frac{p(x,y) - p(x + L_x, y)}{L_x} = \beta_x \quad (4.3.2)$$

pressure can be expressed as:

$$p(x,y) = -\beta_x x + P(x, y) \quad (4.3.3)$$

in which $P(x, y)$ indicates the periodic pressure hence:

$$P(x, y) = P(x + L_x, y) = P(x + 2L_x, y) = \cdots \quad (4.3.4)$$

As a result, the Navier-Stokes equations read as

$$\begin{aligned}
\left\{ \partial_t u_i + \partial_j (u_i u_j) = -\partial_i P + \frac{1}{Re} \partial_{ij} u_i \\
\psi(x^+, y) = \psi(x^+ + L_x, y) \quad \psi = u, v, P
\end{aligned} \quad (4.3.5)$$

For span wise direction $\beta_z$ is taken as zero, then:

$$\psi(x^+, y) = \psi(x^+ + L_z, y) \quad \psi = u, v, P \quad (4.3.6)$$

In order to determine $\beta_x$, one can balance inlet and outlet pressure with wall shear stress $\tau_w$:

$$p_{in}[L_yL_z] = p_{out}[L_yL_z] + 2\tau_w[L_xL_z] \quad (4.3.7)$$

Therefore:

$$\frac{p_{in} - p_{out}}{L_x} = \frac{\tau_w}{\delta} = \beta_x \quad (4.3.8)$$

If friction velocity turbulent Reynolds number is defined as

$$u_T = \sqrt{\frac{\tau_w}{\rho}} ; \quad Re_T = \frac{\rho u_T \delta}{\mu} \quad (4.3.9)$$

Then $\beta_x$ can be expressed as:
\[ \beta_x = \frac{\rho u_x^2}{\delta} \]  

(4.3.10)

or

\[ \beta_x = \frac{\mu^2 Re^2}{\rho \delta^3} \]  

(4.3.11)

### 4.3.2. Simulation results

Both Smagorinsky and Dynamic Smagorinsky model were used as the subgrid models for the channel flow simulation. Initial velocity profile was randomly perturbed using cosine function. Then flow equations were integrated in time by means of implicit Euler method. Time averaging was performed after the statistically steady state reached. Turbulent Reynolds number is about 180, based on the turbulent wall-shear velocity and the channel half-width \( \delta \). The channel size is chosen as \( \pi \times .289\pi \) in streamwise and spanwise [7]. The half-width is chosen as 1 and computational nodes in this direction are distributed using tangent hyperbolic distribution [30]:

\[ y_j = \frac{1}{2} \tanh \left[ \left( -1 + \frac{2(j-1)}{N_J-1} \right) \tanh^{-1} \alpha \right] \quad j = 1, 2, ..., N_J \]  

(4.3.12)

The transformation parameter \( \alpha \) can vary between 0 and 1. The higher \( \alpha \) the more nodes close to walls will be distributed. For simulations in this paper we used \( \alpha = .98346 \). Uniform meshes \( 16 \times 32 \) are used in the streamwise and spanwise directions providing uniform spacings of \( \Delta x^+ \approx 35 \) and \( \Delta z^+ \approx 5 \). Flow equations were co-locatedly discretized using finite volume method and the standard SIMPLE algorithm.

We test SIMPLE algorithm for a series of DNS and LES solutions using different time steps for time integration and two different schemes for discretizing convection terms. In Table 1 different test cases are named and their characteristics are listed. For each test case, plane-averaged wall-shear rates \( \partial u / \partial y |_{\omega} \delta / U \) and root-mean-square turbulent statistics are monitored and shown in Figures 3 to 5. \( U \) is centerline velocity. It can be shown that the exact nondimensional wall-shear rates \( \partial u / \partial y |_{\omega} \delta / U \) for any laminar flow in channel flow is 2. Therefore, if a numerical algorithm is able to maintain the unstable modes in turbulent flow, its averaged wall-shear rates fluctuate over time and do not converge to 2. As it is shown two of the test cases LES 4 and LES 5 converge to laminar solutions.
LES 4 shows that standard SIMPLE algorithm with Euler time integration converges to laminar solution if the computational time step in wall units $\Delta t \nu / u_r^2$ exceeds 0.2. This is 3 times higher than what Kim et al. [21] used, .067, in their explicit simulation. However, the laminarization starts earlier than implicit algorithm of Choi and Moin [7], as in their algorithm laminarization starts after $1.2 \nu / u_r^2$. This might give a clue for the question they raised. Having used second order accurate Crank-Nicolson scheme, in their paper [7], they noticed that the laminarization begins quite earlier than using Kolmogorov time scale as a time step. Kolmogorov time scale for this channel flow is about 2.4 wall units while the laminarization occurs for any time step larger than 1.2. They were suspicious of amplification factor of Crank-Nicolson scheme, yet they ruled out this possibility by testing the result using the backward Euler scheme. Considering even earlier laminarization when we used first order accurate Euler scheme, the laminarization might be attributed to low accuracy of time integration in both methods. Perhaps third or fourth order time advancement scheme allow implicit algorithms to converge to a turbulent solution if comparable time steps to Kolmogorov are used. However, the numerical cost for these higher accurate schemes must be carefully studied.
Table 1. Different test case conventions and their numerical specifications.

<table>
<thead>
<tr>
<th>Test Case Convention</th>
<th>Type of Modeling</th>
<th>Numerical Resolution</th>
<th>Time Steps $\Delta t u^2 / v$</th>
<th>Convection Discretization Scheme</th>
<th>Other Descriptions</th>
</tr>
</thead>
<tbody>
<tr>
<td>DNS</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>$4^{th}$ order Central Difference</td>
<td>Results from Ref. [1]</td>
</tr>
<tr>
<td>DNS 1</td>
<td>---</td>
<td>$16 \times 129 \times 32$</td>
<td>.004</td>
<td>$2^{nd}$ order Central Difference</td>
<td></td>
</tr>
<tr>
<td>DNS 2</td>
<td>---</td>
<td>$16 \times 129 \times 32$</td>
<td>.04</td>
<td>$2^{nd}$ order Central Difference</td>
<td></td>
</tr>
<tr>
<td>LES 1</td>
<td>No Model</td>
<td>$16 \times 66 \times 32$</td>
<td>.004</td>
<td>$2^{nd}$ order Central Difference</td>
<td></td>
</tr>
<tr>
<td>LES 2</td>
<td>No Model</td>
<td>$16 \times 66 \times 32$</td>
<td>.1</td>
<td>$2^{nd}$ order Central Difference</td>
<td></td>
</tr>
<tr>
<td>LES 3</td>
<td>No Model</td>
<td>$16 \times 66 \times 32$</td>
<td>.2</td>
<td>$2^{nd}$ order Central Difference</td>
<td></td>
</tr>
<tr>
<td>LES 4</td>
<td>No Model</td>
<td>$16 \times 66 \times 32$</td>
<td>.4</td>
<td>$2^{nd}$ order Central Difference</td>
<td></td>
</tr>
<tr>
<td>LES 5</td>
<td>No Model</td>
<td>$16 \times 66 \times 32$</td>
<td>.004</td>
<td>$3^{rd}$ order QUICK</td>
<td></td>
</tr>
<tr>
<td>LES 6</td>
<td>Smagorinsky</td>
<td>$16 \times 66 \times 32$</td>
<td>.04</td>
<td>$2^{nd}$ order Central Difference</td>
<td></td>
</tr>
<tr>
<td>LES 7</td>
<td>Dynamic Smagorinsky</td>
<td>$16 \times 66 \times 32$</td>
<td>.04</td>
<td>$2^{nd}$ order Central Difference</td>
<td></td>
</tr>
</tbody>
</table>

As mentioned above, LES 5 also converges to a laminar solution, Figure 5. The test case uses third-order accurate Quadratic Upwind Interpolation for Convective Kinematics or so called QUICK. The laminarization for this test case can be attributed to its upwinding characteristics. It is well known that upwinding schemes are dissipative in nature; therefore, LES 5 test case
disqualifies using third-order accurate upwinding schemes when spatial resolutions are comparable to this case.

Figure 3. Nondimensional velocity gradient at the wall of channel flow versus nondimensional time is shown. Second order accurate central difference scheme is used for convection terms. Time step was set as $10^{-4}$ and 129 nodes were used in the wall-normal direction.
Figure 4. The effect of 4 different time steps on nondimensional velocity gradient at the wall of channel flow versus nondimensional time is shown. In these test cases (LES 1, LES 2, LES 3 and LES 4) second order accurate central difference scheme is used for convection terms and 66 nodes were used in the wall-normal direction. Due to large time step, LES 4 converges to laminar flow solution.

Figure 5. Nondimensional velocity gradient at the wall of channel flow versus nondimensional time is shown. Third order accurate QUICK scheme is used for convection terms. Time step was set as $10^{-4}$ and 66 nodes were used in the wall-normal direction.
To study the effect of time step three test cases are designed. LES 1, LES 2 and LES 3 use three different time steps to illustrate the effect of time integration on turbulent statistics. Figure 6 shows that, the smaller the time steps are, the more accurate results are obtained. However, little difference can be seen for mean velocity profile and Reynolds shear stresses for these three test cases. Similar results are shown in Figure 7 for different DNS solutions. The effect of time steps on the results stems from both inaccuracy in time integration and capturing larger flow structures. Large time steps are not able to capture small fluctuations in flow field; therefore, small perturbations are eradicated from results. Nevertheless, the effect of time accuracy on the results should not be ignored. Using more accurate schemes can considerably improve the results [7]. The balance between the size of time step and the accuracy of time integration methods is similar to the balance between spatial spacing and the accuracy of spatial schemes. A suitable balance can be estimated using Fourier analysis of different schemes. An interested reader is referred to study the concept of modified wave number in [12][40]. Figure 8 shows three different iso-surfaces for normal component of velocity for part of the channel. This figure shows the coexistence of different values of normal velocity. Obviously, no such iso-surfaces could be drawn if the flow field became laminar, as normal component of velocity would be zero on the entire flow field.

![Diagram](image_url)
Figure 6. The effect time steps on root-mean-square of different turbulent statistics. In these test cases (LES 1, LES 2 and LES 3) second order accurate central difference scheme is used for convection terms and 66 nodes were used in the wall-normal direction.
Figure 7. The effect time steps on root-mean-square of different turbulent statistics. In these test cases (DNS 1, DNS 2) second order accurate central difference scheme is used for convection terms and 66 nodes were used in the wall-normal direction.

Figure 8. Three different iso-surfaces -.4, -.038 and .157 for normal component of velocity.

We furthered the investigating of the performance of SIMPLE algorithm by introducing subgird models. Smagorinsky and dynamic Smagorinsky models are used to simulate the same channel flow. The results are shown in Figure 10. LES 6 and LES 7 represent the results of using Smagorinsky and dynamic Smagorinsky models, respectively. Low performance of both models
is clear in these figures, especially Smagorinsky model. The added turbulent viscosity, shown in Figure 10 by both models does not improve the result of the simulation, yet dynamic Smagorinsky model has clear superiority. These results are aligned with the results of other references such as [26].
Figure 9. Root-mean-square of different turbulent statistics is compared using Smagorinsky and dynamic Smagorinsky models, LES 6 and LES 7. Their results are also compared with LES1 and DNS. In these test cases (LES 1, LES 6 and LES 7) second order accurate central difference scheme is used for convection terms and 66 nodes were used in the wall-normal direction.
4.3.3. More accurate results

It was shown in the previous section that implicit algorithm allows us to use larger time steps to simulate turbulent flows. However, it was shown both in Figure 6 and Figure 7 that less accurate solutions are obtained when larger time steps are used. In this section we aim to devise a method by which more accurate solutions can be obtained. We basically follow the same methodology as in Richardson’s extrapolation technique [7], yet it is customized for this particular application. Assume we have estimated a turbulent quantity, $\theta$, using a particular time discretization method, therefore:

$$
\theta_{\text{Exact Solution}} = \theta(\Delta t) + E(\Delta t) \quad (4.3.12)
$$

If two different time steps are used to estimate $\theta$, then:

$$
\theta = \theta(\Delta t_1) + E(\Delta t_1) \quad (4.3.13)
$$

$$
\theta = \theta(\Delta t_2) + E(\Delta t_2) \quad (4.3.14)
$$
Having an estimate for the time discretization method, one can approximate error and improve the solution. For instance, in standard SIMPLE algorithm, numerical time integration method has an error of:

\[ \text{(4.3.15)} \]

If we assume \( \text{, then a better estimation for } \theta \text{ can be found using equation (4.3.15):} \]

\[ \text{(4.3.16)} \]

We call this more accurate estimation as “combined solution”. To illustrate the effect of this improvement, we use the solutions of LES 2 as \( \text{ and the solutions for LES 3 as } \). Combining these two solutions using equation (4.3.16), results in Figure 11 are obtained. The combined solution closely follows the solutions of LES 1 which uses a considerably smaller time step. To compare the cost of finding a solution for LES 1 and the combined solution, we assume we need to integrate flow equations for 100 non-dimensional time. Therefore, for LES 1, LES 2 and LES 3 we need 25000, 1000 and 500 realizations, respectively. This means the combined solution needs only 1500 (1000+500) realizations which is 16.6 times less than the number of realizations for LES 1. In this way, not only an accurate solution was obtained, but also the cost of time integration was considerably reduced. The combined solution method is particularly attractive for implicit method in which large time steps can be used.

![Figure 11](image-url)
Figure 11. Finding more accurate solutions using previous solutions. The “combined solutions” are obtained by using LES 2 and LES 3.

References


Chapter 5

A Consistent Large Eddy Simulation Formulation

5.1. Introduction

Turbulent flow, as its name stands, consists of chaotic interactions of small fluid elements called eddies. These so called eddies are characterized by their length and time scale [39]. Derived from dimensional analysis, Kolmogorov length and time scale are believed to be the smallest characteristics of turbulent flows [39]. Therefore, to find an accurate solution for a turbulent flow field, the three-dimensional, time-dependent Navier-Stokes equations are finely discretized both in time and space to capture all different time and length scales including the Kolmogorov scales. Such an approach is referred to as direct numerical simulation (DNS) [14]. DNS aims to capture the smallest dynamically-existing eddies in a turbulent flow. Knowing the smallest length and time scales of a turbulent field scale as $Re^{-\frac{3}{4}}$ and $Re^{-\frac{1}{2}}$ [39] respectively, it is not difficult to understand that direct numerical simulations of many practical flow fields are not feasible. Generally, the total number of points for DNS in three dimensions is estimated to be proportional to $Re^{\frac{9}{4}}$ [33]. Here Reynolds number is based on an integral scale of the flow.

An alternative method, LES, targets larger length and consequently, time scales of flow to overcome the DNS computational cost [14][36]. Formally, LES equations are derived by applying a low-pass filter [36] [34] [24] to flow equations to remove smaller motions than a characteristics length scale. This length scale is assumed to play a key role in these types of simulation. Generally, since different mathematical filters do not necessarily commutate with spatial derivatives and the nonlinear terms, new terms may appear in flow equations. Hence, similar to most turbulence modeling methods, closure problem emerges for solving these equations. To resolve the closure problem, in most actual LES simulations, many of these unknown terms are first eliminated and the remaining terms are modeled [1][5][14][28][30][32][33][36]. Many different techniques have been developed to model these presumably more influenced terms using the resolved variables [3][6][7][13][14][15][18][26][29][36][37][43]. Mathematically, the characteristic length scale and the shape of the low-pass filter have to
determine how the closure problem should be delineated. However, this is an absent concept in present-day models [14][36].

Conventionally, the main objective of introducing a low-pass filter is the intuitive similarity of the act of filtering operator and the results of resolving an equation on a coarse mesh. Intuitively the act of filtering is considered as “removing the small spatial scales” [17] and since in LES simulations inadequate spatial resolution is incapable to resolve small scales, therefore their results are assumed to be similar. This intuitive similarity is the basis of all LES simulation in which the concept of filtering is used. Note worthy, in implicit LES simulation, so-called ILES [36], or Two level Simulation (TLS) approach[20][21], for instance, the filtering process is disregarded and therefore this assumption has no room in these types of simulations.

In this paper, we intend to question whether the intuitive similarity of filtering and coarse resolving of an equation is also mathematically supported. We first amplify the rule of filtering using a simple ODE equation by which different characteristics of filtering operator is described. We often deliberately intensify the effect of filtering to provide a clear insight of the rule of filtering on an equation. We then introduce some mathematical tools and operators to facilitate our discussion toward questioning the conventional filtered Navier-Stokes equations. We borrow some numerical results and arguments presented in literature to enrich our study. Using the mathematical tools and the numerical evidences, inconsistencies and problems in the conventional LES simulations are explained and discussed. Some of these conceptual questions were also raised by Pope as “Ten Questions” to LES of turbulent flows [35], yet with a different perspective. Finally, a new consistent framework for large eddy simulations is introduced. The new framework explains and justifies the inconsistent observations in conventional LES framework and paves the path for future more reliable simulations.

5.2. Formal Filtering of Flow Equations

As it was mentioned earlier, LES equations are formally obtained by applying a low pass filter to the Navier-Stokes equations. We therefore start with a space-time convolution filter operation $\mathcal{L}$ which is applied on an unbounded domain. The filter $\mathcal{L}$ is linear and its operation is defined by

$$
\bar{u}(x, t) = \mathcal{L}(u)(x, t) = \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} G(x - x', t - t')u(x', t')dx' = G * u \quad (5.2.1)
$$
where
\[ f^{+\infty}_{-\infty} dt^{'} \int^{+\infty}_{-\infty} G(x-x^{'}, t-t^{'}) u(x^{'}, t^{'}) dx^{'} = 1 \] (5.2.2)

It may be shown that a linear integral filter commutates with partial differentiation if and only if it is a convolution operation [14]. Therefore convolution filters verify the following commutative properties:
\[ \partial_t f = \partial_t \tilde{f} \quad ; \quad \partial_i f = \partial_i \tilde{f} \quad i = 1,2,3 \] (5.2.3)

Where \( \partial_t \) and \( \partial_i \) denote time and special derivatives. If we define the commutator \([f, g]\) of two operators \( f \) and \( g \) as:
\[ [f, g] \Phi = f \circ g(\Phi) - g \circ f(\Phi) \] (5.2.4)

Then we can express the commutative properties in Equation (5.1.3) as:
\[ [\mathcal{L}, \partial_t](u) = 0 \quad ; \quad [\mathcal{L}, \partial_i](u) = 0 \quad i = 1,2,3 \] (5.2.5)

As a result, since convolution filters commutate with partial differentiation, filtered Navier-Stokes equations find the following form[14][36]:
\[ \partial_t \bar{u}_i + \partial_j (\bar{u}_j \bar{u}_i) + \partial_i \bar{p} - \frac{1}{Re} \partial_j \bar{j} \bar{u}_i = 0 \] (5.2.6)

or
\[ \partial_t \bar{u}_i + \partial_j (\bar{u}_j \bar{u}_i) + \partial_i \bar{p} - \frac{1}{Re} \partial_j \bar{j} \bar{u}_i + \partial_j \tau_{ij} = 0 \] (5.2.7)

in which:
\[ \tau_{ij} = \bar{u}_i \bar{u}_j - \bar{u}_i \bar{u}_j \] (5.2.8)

Similarly, continuity equation reads as:
\[ \partial_i (\bar{u}_i) = 0 \] (5.2.9)

Turbulent stress tensor \( \tau_{ij} \) or its dynamic consequence \( \partial_j \tau_{ij} \) should be therefore modeled based on the resolved velocity filed, \( \bar{u}_i \).
In contrast to convolution filters, non-uniform filters do not necessarily commute with partial differentiation. Developing LES for complicated simulations requires these types of filters to overcome geometrical and dynamical complexities. An extended filter in one spatial dimension is defined as:

$$\tilde{u}(x, t) = \int_{x-\Delta_-(x)}^{x+\Delta_+(x)} \frac{g(\xi)}{\Delta(x)} u(\xi, t) d\xi$$  \hspace{1cm} (5.2.10)

$\Delta_+(x)$ and $\Delta_-(x)$ are donated as the upper and lower bounding functions where filter-width is defined as

$$\Delta(x) = \Delta_+(x) + \Delta_-(x)$$  \hspace{1cm} (5.2.11)

Consequently, non-commuting property of this filter is expressed as:

$$\partial_x(u) \neq \partial_x(\tilde{u}) \quad \text{or} \quad \llbracket \mathcal{L}, \partial_x \rrbracket(u) \neq 0$$  \hspace{1cm} (5.2.12)

Hence, filtering the flow equations using non-uniform filters (or any spatially non-commutator filter) yields [14][36]:

$$\partial_t \tilde{u}_i + \partial_j(\tilde{u}_i \tilde{u}_j) + \partial_i \tilde{p} - \frac{1}{Re} \partial_{jj} \tilde{u}_i = -\partial_j \left( \left[ \mathcal{L}, \Pi \right](u_i, u_j) \right) - \left[ \mathcal{L}, \partial_j \right](p) - \frac{1}{Re} \left[ \mathcal{L}, \partial_{jj} \right](u_i) + \left[ \mathcal{L}, \partial_j \right] \left( \Pi(u_i, u_j) \right)$$  \hspace{1cm} (5.2.13)

$$\partial_i(\tilde{u}_i) = - \left[ \mathcal{L}, \partial_i \right](u_i)$$  \hspace{1cm} (5.2.14)

$\Pi$ is the product operator defined as:

$$\Pi(f, g) = fg$$  \hspace{1cm} (5.2.15)

The first term on the RHS of Equation (5.2.13) is the turbulent stress tensor which also appeared in the uniform filtered equation. The remaining unclosed terms in Equation (5.2.13) and the RHS of Equation (5.2.14) are known as ‘commutator errors’ [14][36]. It is interesting to note that filtered continuity equation is not divergence-free if commutation errors arise. Even though there are a number of modeling techniques for the turbulent stress tensor, ‘commutator errors’ usually escape from modeling and therefore they are neglected [36][9].
To complete the formal formulation of large eddy simulation, we briefly introduce three classical filters for these types of simulations. If \( l \) denotes a length parameter, the kernel for Top-hat, Gaussian and cut-off filter for mono-dimensional case read as [14][36][34]:

1- Top-hat Filter

\[
G_{Th}(x - \xi) = \begin{cases} 
\frac{1}{l} & \text{if } |x - \xi| < \frac{l}{2} \\
0 & \text{Otherwise} 
\end{cases} 
\]  

(5.2.16)

2- Gaussian filter

\[
G_{Ga}(x - \xi) = \left(\frac{\alpha}{\pi l^2}\right)^{1/2} \exp \left[-\left(\frac{\alpha(x-\xi)^2}{l^2}\right)\right] 
\]  

(5.2.17)

3- Spectral cut-off filter

\[
G_{Co}(x - \xi) = \frac{1}{l} \frac{\sin(\pi(x-\xi)/l)}{\pi(x-\xi)/l} 
\]  

(5.2.18)

Parameter \( \alpha \) is a constant and generally taken to be 6 (~2\( \pi \)). Since these kernels are merely mathematical expressions, therefore, there is no formal restriction on the length parameter, \( l \). On the other hand, since formally in large eddy simulation localized filters are desired, effective filter width \( \Delta_{ef} \) can be defined. To define filter width, the Fourier transfer of kernel \( G \) is defined as

\[
H(k) = \int_{-\infty}^{+\infty} G(x)e^{-ikx} \, dx 
\]  

(5.2.19)

then the filter width can be found as [14]:

\[
\frac{1}{\Delta_{ef}} = \frac{1}{2\pi} \int_{-\infty}^{+\infty} |H(k)|^2 \, dk 
\]  

(5.2.20)

or alternatively it is found such as

\[
\Delta_{ef}^2 = \int_{-\infty}^{+\infty} z^2 G(z) \, dz 
\]  

(5.2.21)

It is also common to define the filter width as the length of the interval around the origin in which the filter-kernel \( G \) is non-zero [14].
In the sequel, all these filters can be easily expanded for higher dimensions. Denoting \( L_i \) as one dimensional filter operation, three dimensional filters can be represented as:

\[
L(u) = L_1 \circ L_2 \circ L_3 (u)
\]  

(5.2.22)

5.3. Filtering versus Resolving of an Equation

In the previous section the formal basis of LES equations were presented. It was shown that applying a low-pass filter adds new unclosed terms into flow equations and therefore, the filtered Navier-Stokes and continuity equations considerably differ from their original forms. In this section, we aim to compare the solution and properties of a filtered equation and its original equation. It is of our interest to visualize the effect of filtering on the solution of an ODE (or a PDE). We try to find out how one should interpret the numerical solution of a filtered equation. In order to provide a clear picture, we use simple equations instead of considerably more complicated flow equations.

We begin our study by illustrating the effect of using different filters and different length parameters, \( l \), on a function. A random function \( F(x) \) is defined as:

\[
F(x) = (x - 2)^3 - x^4 + .5\sin(200x) + 2\sin(10x) + .6\sin(50x) + 15
\]  

(5.3.1)

Two different filter kernels, Top hat and Gaussian, are used to numerically filter \( F(x) \); and in order to show the effect of length parameter, we used \( l = .1 \) and \( l = .9 \) for each filters. The filtering results in four distinct answers for \( F(x) \) which are shown Figure 1.
Figure 1. The effect of applying different filters and different length parameters.

Therefore, unlike $F(x)$, $\overline{F(x)}$ does not have a unique value at point $x$. This means, one should strictly associate $\overline{F(x)}$ with its filter characteristics. In order to explicitly distinguish different answers for $\overline{F(x)}$, we use a new format to show a filtered function. For instance, the filtered value of $F(x)$ using Top hat kernel and length parameter $l = .1$ is denoted as:

$$\overline{F(x)}|_{l=.1}^{Top \ hat}$$

As a result, inequalities such as:

$$\overline{F(x)}|_{l=.1}^{Top \ hat} \neq \overline{F(x)}|_{l=.9}^{Top \ hat}$$

or

$$\overline{F(x)}|_{l=.9}^{Top \ hat} \neq \overline{F(x)}|_{l=.9}^{Gaussian}$$

can be exclusively expressed.

It is worthy to emphasis that there is no formal restriction on the choice of the length parameter. Although using different length scales might result in very different forms of a function, no error should be considered for any length parameter. For example, if we use $l = 2$ to filter $F(x)$, the results loosely follow the details of the original function, see Figure 2.
However, low correlation of the filtered function and the original function should not be interpreted as a mathematical error; it is merely part of characteristics of filtering operator.

Considerably different filtered values for a unique function $F(x)$ in the above example clearly suggest that the properties of filtering operator cannot be dismissed. In other words, the symbol $\overline{F(x)}$ corresponds to infinite possible answers (which can be very different) unless the characteristics of a filter is specified, $\overline{F(x)}\big|_{l=\alpha}^{\text{Filter type}}$. Moreover, the choice of length width (and also the type of filter) does not introduce inaccuracies or errors to the solution of a filtering operator, i.e. all different solutions of a filtering operator are equally accurate independent of how they correlate with the original function.

Another illustrative example is finding a numerical solution of a filtered ODE. This example is merely presented to provide an insightful comparison of a numerical solution of an ODE (or PDE) and its filtered form. We aim to demonstrate the effect of filtering on the form of an equation and also its answer. This demonstration is planned here, as we found no evidence in literature of sharp distinction of an ODE (or PDE) equation and its filtered form. Perhaps, the intuitive understanding of low-pass filter (i.e. removing the small spatial scales), has considerably contributed in the matter. Consider the following ODE and its exact solution:
\[
\begin{align*}
\begin{cases}
y' - y = 2x - x^2 + \delta(\omega \cos(\omega x) - \sin(\omega x)) \\
y(0) = 0
\end{cases}
\end{align*}
\]  
(5.3.5)

Exact Solution: \( y = x^2 + \delta \sin(\omega x) \)

We now apply a Top hat filter with constant length parameter of \( \Delta \), therefore the filter for dummy function \( f \) is defined as:

\[
\bar{f}^{\text{Top hat}}_{|t=\Delta}(x, t) = \int_{-\infty}^{+\infty} G(x - z)f(z)dz = \int_{\frac{x-\Delta}{2}}^{\frac{x+\Delta}{2}} f(z)dz
\]  
(5.3.6)

Applying this filter on the above set of equations and using the commutating properties of Top hat filter:

\[
\begin{align*}
\begin{cases}
\bar{y}^{\text{Top hat}}_{|t=\Delta} - \bar{y}^{\text{Top hat}}_{|t=\Delta} = \int_{\frac{x-\Delta}{2}}^{\frac{x+\Delta}{2}} [2z - z^2 + \delta(\omega \cos(\omega z) - \sin(\omega z))] \frac{1}{\Delta} dz \\
\bar{y}^{\text{Top hat}}_{|t=\Delta}(0) = \frac{\Delta^2}{12}
\end{cases}
\end{align*}
\]  
(5.3.7)

Exact Solution: \( \bar{y}^{\text{Top hat}}_{|t=\Delta} = \int_{\frac{x-\Delta}{2}}^{\frac{x+\Delta}{2}} [z^2 + \delta \sin(\omega z)] \frac{1}{\Delta} dz \)

and consequently the following set of equations can be derived:

\[
\begin{align*}
\begin{cases}
\bar{y}^{\text{Top hat}}_{|t=\Delta} - \bar{y}^{\text{Top hat}}_{|t=\Delta} = 2x - x^2 - \frac{\Delta^2}{12} + \frac{2\delta \sin(\frac{\omega \Delta}{2})}{\Delta} \left( \cos(\omega x) - \frac{\sin(\omega x)}{\omega} \right) \\
\bar{y}^{\text{Top hat}}_{|t=\Delta}(0) = \frac{\Delta^2}{12}
\end{cases}
\end{align*}
\]  
(5.3.8)

Exact Solution: \( \bar{y}^{\text{Top hat}}_{|t=\Delta} = x^2 + \frac{\Delta^2}{12} + \frac{2\delta}{\omega \Delta} \sin \left( \frac{\omega \Delta}{2} \right) \sin(\omega x) \)

If similar to standard LES formulation we define scalar \( \tau \) as:

\[
\tau(x, \Delta) = \left[ -\frac{\Delta^2}{12} + \frac{2\delta \sin(\frac{\omega \Delta}{2})}{\Delta} \left( \cos(\omega x) - \frac{\sin(\omega x)}{\omega} \right) \right] - \left[ \delta(\omega \cos(\omega x) - \sin(\omega x)) \right] \]  
(5.3.9)

then Equation (5.3.8) can be formulated as

\[
\begin{align*}
\begin{cases}
\bar{y}^{\text{Top hat}}_{|t=\Delta} - \bar{y}^{\text{Top hat}}_{|t=\Delta} = 2x - x^2 + \delta(\omega \cos(\omega x) - \sin(\omega x)) + \tau(x, \Delta) \\
\bar{y}^{\text{Top hat}}_{|t=\Delta}(0) = \frac{\Delta^2}{12}
\end{cases}
\end{align*}
\]  
(5.3.10)

Exact Solution: \( \bar{y}^{\text{Top hat}}_{|t=\Delta} = x^2 + \frac{\Delta^2}{12} + \frac{2\delta}{\omega \Delta} \sin \left( \frac{\omega \Delta}{2} \right) \sin(\omega x) \)
It can be simply proved that:

$$\lim_{\Delta \to 0} \tau(x, \Delta) = 0$$  \hspace{1cm} (5.3.11)

We now know an ODE equation, Equation (5.3.5), and its equivalent filtered form, Equation (5.3.10), and their exact solutions. Several observations can be made by comparing these two sets of equations. The effect of top hat filter on the ODE equation can be seen by comparing the LHS of (3.5) and (3.10). The difference tends to zero if $\Delta$ leans to zero; conversely, the difference becomes significant if $\Delta$ is not chosen small enough. Noteworthy, choosing large $\Delta$ does not introduce any inaccuracy or inconsistency into the solution of the filtered ODE, as filtering is a mathematical operator not a physical operator. Moreover, although the solution of ODE in Equation (5.3.5) is unique, the filtered ODE in Equation (5.3.10) has countless number of solutions which can be produced by varying the value of $\Delta$. It is quite interesting to note that all these solutions are equally accurate and no mathematical preference can be considered for each of them. At this point, we want to stress that filtering is a purely mathematical operator and one should not be confused with its physical interpretation. Relating the effect of a purely mathematical operator to a physical event needs careful attention.

Having analytically compared the difference of the ODE and its filtered form, let us compare them in a numerical space. Assume we are interested to numerically find the solution of the ODE in Equation (5.3.5) (compare it with finding numerical solution for Navier-Stokes equations) in the following form:

$$\begin{cases} 
y' - y = 2x - x^2 + \delta (\omega \cos(\omega x) - \sin(\omega x)) \\
\omega = 100 \; ; \; \delta = .1 \\
y(0) = 0 \end{cases} \Rightarrow y(x) =? \hspace{1cm} (5.3.12)$$

Since it is a simple ODE, one can easily numerically expand the derivative term $y'$ and rearrange the equation in the following form:

$$y^{n+1} = [2x - x^2 + A(\delta, \omega) + y^n] \Delta x + y^n \; ; \; y^0 = 0 \hspace{1cm} (5.3.13)$$

where

$$A(\delta, \omega) = \delta (\omega \cos(\omega x) - \sin(\omega x)) \hspace{1cm} (5.3.14)$$
and the superscript “n” denotes the value of $y(n\Delta x)$ and $\Delta x$ is constant mesh spacing. Therefore, the more nodes we use, the more accurate the solution would be. The results of using different resolutions for solving Equation (5.3.12) are shown in Figure 3. It is shown that the numerical discretization of Equation (5.3.12) converges to the exact solution if enough mesh is used (compare it with DNS of flow equations). However, using inadequate number of mesh brings about inaccuracies in the numerical solution, as it is seen when 100 and 50 nodes are used in Figure 3. For a given numerical scheme, the accuracy of the solution merely relies on the mesh resolution, and therefore, by increasing mesh resolution the solution converges toward the exact solution (A property which is known in most standard LES simulations). An important observation is that, unlike filtering, resolving an equation on inadequate mesh introduces errors into the final solution. This means resolving an equation is tied with error.

![Figure 3. Resolving an ODE using different mesh resolution.](image)

Similarly, the numerical solution of the filtered ODE equation:

$$\begin{align*}
\left. \tilde{y} \right|_{i=\Delta}^{\text{Top hat}} - \left. \tilde{y} \right|_{i=\Delta}^{\text{Top hat}} &= 2x - x^2 + A(\delta, \omega) + \tau(x, \Delta) \\
\omega &= 100 \quad \delta = .1 \\
\left. \tilde{y} \right|_{i=\Delta}^{\text{Top hat}} (0) &= \frac{\Delta^2}{12}
\end{align*}$$

$$\Rightarrow \left. \tilde{y} \right|_{i=\Delta}^{\text{Top hat}} =? \quad (5.3.15)$$
can be solved using:

\[
[\tilde{y}_{i=\Delta}^{Top\ hat}]^{n+1} = \left[ 2x - x^2 + A(\delta, \omega) + \tau(x, \Delta) + [\tilde{y}_{i=\Delta}^{Top\ hat}]^n \right] \Delta x + [\tilde{y}_{i=\Delta}^{Top\ hat}]^n \quad (5.3.16)
\]

Since the filtered equation does not have unique answer, we need to specify the length parameter to distinguish the solutions. To illustrate the effect of length parameter, we use these three constant length \( \Delta = 0.1, \Delta = 1 \) and \( \Delta = 3 \). Figure 4 shows three different solutions for the filtered ODE using different resolutions and their comparison with the exact solutions. In the figure, also, the original function \( y(x) \) is shown with a black solid line. Several observations can be made:

a. Depending on the filter characteristics, the solution of the filtered ODE can be considerably different. For instance, the exact solutions of \( \Delta = 0.1 \) and \( \Delta = 3 \) in Figure 4 are considerably dissimilar. It is crucial to note that although these exact solutions are quite different, both of them are free of any error.

b. The solution of the filtered ODE (even the exact solution) might show little correlation with the original function (Compare the solution of \( \Delta = 3 \) in Figure 4 with the original function).

c. The numerical solutions do not converge to the original function \( y(x) \); they all converge to their exact filtered solutions which are different than the original function.

d. Although different exact solutions for filtered ODE encounter no error, their numerically resolved solutions gain errors proportional to the inverse of the number of nodes.

e. Numerical solution of the filtered ODE converges quite differently toward its exact solution than the original ODE (compare the resolved solutions for \( N=100 \) in Figure 3 and Figure 4).
Figure 4. Solution of the filtered ODE for different length parameter: a) $\Delta = 0.1$, b) $\Delta = 1$ and c) $\Delta = 3$.

Solid blue lines are the exact solutions for the corresponding filter length parameter. The black solid line shows the original function which was filtered.
One should note that $\tau(x, \Delta)$ is responsible for the entire discrepancy between the behavior the filtered ODE and the original ODE. Clearly, the above observations indicate that this term has a key rule in the solution of an ODE and it cannot be ignored unless the length parameter is very small. Besides, it is difficult to associate any generic physical property to $\tau(x, \Delta)$, as it was derived by solely mathematical operator which can be considerably capricious. For instance, $\tau(x, \Delta)$ in the above example is shown in Figure 5 for two different $\Delta$. It is difficult, if not impossible, to claim that these two different curves are representing the very same physical quantity or phenomenon (turbulent dissipation for example). If we claim so, we should be able to justify the arbitrariness of the type of filter and its length parameter. Note, a physical phenomenon does not change its value, properties and characteristics based on what we use in our mathematical space. However, one might accurately design a mathematical operator so that it resembles a physical quantity. All these observations and conclusions recommend a careful interpretation of a filtered equation.

![Figure 5](image.png)

Figure 5. $\tau(x, \Delta)$ is plotted for two different length parameters: $\Delta=.01$ and $\Delta=3$
Finally, it is essential to consider whether discrete derivative operators used in resolving an equation have a low-pass filtering effect. For example, is there any similarity between the numerical solutions of Equation (5.3.12) in Figure 3 and the exact solutions of Equation (5.3.15) in Figure 4 (which are also compared in Figure 6)? It is quite clear in the figures that resolving the original ODE equation has no similarity with its filtered forms, as it should not have. Despite the common description of the effect of filtering, the rule of filtering is to “include” the effects of small scales rather than “removing” the effect of small scales. The smoothen results of filtered solutions in Figure 6 is not because of removing the effects of small scales, it is because of summing all the effects of smaller scales than filter length scale and representing them in a summed form. That is why by increasing the filter length parameter the filtered solutions in Figure 6 are shifted upward. On the other hand, inadequate numerical resolution for resolving the original function fails to see the small perturbations of the function and therefore results in inaccurate answers. Therefore, resolving an equation should be sharply distinguished with filtering an equation. The errors which are introduced in resolving an equation should not be confused with the effect of a filtering process. Filtering is an errorless operator, whereas resolving operator introduces errors into solution. To demonstrate it more mathematically, we denote $\mathcal{R}$ as a resolving operator and we use $f(x, y(x))$ to represent an ODE. For example, the ODE in Equation (5.3.5), can be written as:

$$f(x, y) = y' - y - 2x + x^2 - A(\delta, \omega) = 0 \quad (5.3.17)$$
Then resolving operator $\mathcal{R}$ is defined as the effect of discretization on $f(x,y)$ to resolve $f(x,y)$ on $n=N$ nodes. Equation (5.3.13), for instance, can be summarized as:

$$\mathcal{R}_{n=N}\{f(x,y)\} = y^{n+1} - [2x - x^2 + A(\delta, \omega) + y^n]\Delta x - y^n = 0 \quad (5.3.18)$$

Clearly, the properties of the resolving operator $\mathcal{R}$ are influenced by the choice of discretization schemes. As stated earlier, unlike filtering operator, resolving operator $\mathcal{R}$ is associated with a systematic error which is expressed as:

$$y(x)|_{Exact} - y(x)|_{Resolved} = \text{Error}(N, \text{Discretization schemes}) \quad (5.3.19)$$

If filter operator $\mathcal{L}$ is applied on $f(x,y)$:

$$\overline{f(x,y)}|_{l=\Delta}^{\text{Filter type}} = \mathcal{L}[f(x,y)] \quad (5.3.20)$$

Then the differentiation of filtering operator and resolving operator can be explicitly expressed as:

$$\mathcal{R}_{n=N}\{f(x,y)\} \neq \mathcal{L}[f(x,y)] \quad (5.3.21)$$

Although the above expression seems trivial, the effects of these two distinct operators might intuitively seem equivalent [17][9]. For example, the effect of discretization schemes for resolving an equation might be considered similar to the effect of filtering the exact solution on the same mesh net, which is not necessarily true. Accepting such statement corresponds to linking the associated error in resolving operator $\mathcal{R}$ to the effect of the errorless filtering operator $\mathcal{L}$, see Figure 6. Noteworthy, resolving a filtered equation (i.e. $\mathcal{R} \circ \mathcal{L}[f]$) should not be compared with resolving the original equation (i.e. $\mathcal{R}[f]$); as such comparison is irrelevant to the context.

Having introduced resolving operator $\mathcal{R}$ and filter operator $\mathcal{L}$, they can be used to considerably simplify some of our equations. For instance, Equation (5.3.16) can be conveniently expressed as:

$$\mathcal{R} \circ \mathcal{L}(f) = \mathcal{R}_{n=N}\{\mathcal{L}[f(x,y)]\} \quad (5.3.22)$$

Moreover, to facilitate future discussions, let us re-arrange and re-express Equation (5.3.16):
\[ R_{n=N}\{L[f(x, y)]\} = \left[ \tilde{y}_{l=\Delta}^{Top\ hat} \right]^{n+1} - \left[ 2x - x^2 + A(\delta, \omega) + \tau(x, \Delta) + \left[ \tilde{y}_{l=\Delta}^{Top\ hat} \right]^{n} \right] \Delta x - \left[ \tilde{y}_{l=\Delta}^{Top\ hat} \right]^{n} = 0 \]  

(5.3.23)

considering Equation (5.3.13):

\[ R_{n=N}\{f(x, y)\} = y^{n+1} - [2x - x^2 + A(\delta, \omega) + y^n] \Delta x - y^n = 0 \]  

(5.3.24)

Then by explicitly differentiating the rule of filtering as:

\[ R_{n=N}\{L[f(x, y)]\} = \left( \left[ \tilde{y}_{l=\Delta}^{Top\ hat} \right]^{n+1} - \left[ 2x - x^2 + A(\delta, \omega) + \left[ \tilde{y}_{l=\Delta}^{Top\ hat} \right]^{n} \right] \Delta x - \left[ \tilde{y}_{l=\Delta}^{Top\ hat} \right]^{n} \right) - \{\tau(x, \Delta) \Delta x \} \]  

(3.25)

Or in a more mathematical form:

\[ R_{n=N}\{L[f(x, y)]\} = R_{n=N}\{f(x, \Delta)\} + R_{n=N}\{\tau(x, \Delta)\} \]  

(5.3.26)

In this form, therefore, filtered equations are arranged so that they keep the original form of equations plus an extra term to encapsulate all the explicit effects of the filtering operator. This is the preferred form of an equation which is usually used in most LES formulations. One should note the above form of equation has a critical property:

\[ \lim_{\Delta \to 0} \left[ R_{n=N}\{f(x, \Delta)\} + R_{n=N}\{\tau(x, \Delta)\} \right] = R_{n=N}\{f(x, y)\} \]  

(5.3.27)

Which shows that if the effect of filtering operator (i.e. \( \tau(x, \Delta) \)) is negligible (or erroneously ignored or underestimated), the solution of the resolving the filtered equation converges to the solution of resolving the original function. As a result, \( \tau(x, \Delta) \) plays a key rule in discerning the solution of a filtered equation and the solution of the original equation. It is important to note that most low-pass filters hold the following property:

\[ \lim_{\Delta \to 0}(\tau(x, \Delta)) = 0 \]  

(5.3.28)
and consequently:
\[
\lim_{\Delta \to 0} \bar{y} = y \quad (5.3.29)
\]

Therefore, the following points should be considered:

- If \( \Delta \) is very small, then \( \bar{y} \) closely follows \( y \). As a result, the filtered equation has no practical use. The same holds if \( \tau(x, \Delta) \) is incorrectly estimated small. In that case the role of filtering is basically ignored and the solution converges toward the original solution.
- If \( \Delta \) is large enough to provide some practical uses, then \( \tau(x, \Delta) \) must be accurately estimated, unless the solution of \( \bar{y} \) would have neither a meaningful relation with the actual \( \bar{y} \) nor a correlation with the original solution \( y \).
- For large values of \( \Delta \), one must always consider the fact that \( \bar{y} \) might poorly correlate with the details of the original solution \( y \), for instance see Figure 4 (c). This raises a question: what is the use of such solution?

We will further discuss these issues in the LES framework.

### 5.4. Numerical experiments for “perfect LES”

Following previous section, we continue to present more numerical experiments in this section. We borrow suitable numerical experiments done by other researchers within the conventional large eddy simulation framework. However, we employ these numerical experiments with a different objective than their original initiatives.

In order to skip modeling, a non-practical but insightful numerical method called ‘perfect LES’ is utilized in [9][10][16][11][38]. ‘Perfect LES’ is basically established based on beforehand knowledge of the exact solution (which is DNS solution). It, therefore, enables us to examine the unknown or unclosed terms which must be modeled.

The general form of filtered Navier-Stokes equations was presented in Equation (5.2.13) where turbulent stress tensor and commutator errors were introduced as:

\[
-\partial_j \left( [L, \Pi](u_i, u_j) \right) - \left( [L, \partial_j](p) - \frac{1}{Re} [L, \partial_j](u_i) + [L, \partial_j]\left(\Pi(u_i, u_j)\right) \right) \quad (5.4.1)
\]
These new terms were emerged due to the filtering of flow equations and their strong dependence on the filtering operator is clearly projected in the structure of the above expression. As discussed in the previous section, each of these terms must be carefully modeled based on the characteristics of the filtering operator; unless they filtered equation would not accurately represent the flow solution. Authors in [40] were interested to conduct a numerical experiment to demonstrate the effect of filtering operator on each of these terms. However, they confined their numerical experiments to calculate the flux of turbulent stress tensor and convective flux commutator error:

$$\partial_j (\tau_{ij}) = \partial_j \left( [\mathcal{L}, \Pi] (u_i, u_j) \right) = \partial_j (\overline{u_i u_j} - \overline{\bar{u}_i \bar{u}_j}) \quad (5.4.2)$$

$$C_j (u_i u_j) = [\mathcal{L}, \partial_j] \left( \Pi (u_i, u_j) \right) \quad (5.4.3)$$

They first used order of magnitude estimation for subgrid terms for symmetric second-order, skewed and higher-order filters. It was shown that the use of higher-order filters allows additional control over the size of the subgrid terms, however the dominant scaling of both the flux of turbulent stress tensor and convective flux commutator error with $\Delta$ and its derivatives is formally identical if filter-order is higher than 2. For first-order filter, they observed that the commutator error is formally larger than the flux of turbulent stress tensor. They suggest: “if in a certain flow the turbulent stress contributions require explicit modeling, one should also consider incorporating explicit modeling for the commutator errors. It appears to make little sense to model one of the subgrid terms and ignore the other class of subgrid contributions which are formally of equal order of magnitude”. As a result, a priori analysis of the subgird terms was considered using DNS solution of a turbulent flow in a temporal mixing layer [41], in which the governing equations were solved in a cubic geometry of side $\ell$. In order to build different filters, filter width in $x_1$ and $x_3$ directions were kept constant while the $x_2$ direction were allowed to vary.

$$\Delta_1 = \Delta_3 = \Delta_r = \ell / 16 \quad ; \quad \Delta_2 (x_2) = \Delta_r \left( 1 - \alpha e^{-\left( \beta x_2 / \ell \right)^2} \right) \quad (5.4.4)$$

$\alpha$ and $\beta$ are two parameters by which different filters can be defined. In addition, shift-parameter $\alpha$ was introduced by which top-hat filter could be skewed. Non-zero values of $\alpha$ introduces
skewness to the filter, see [8] for further details. To compare different terms, they used $L_2$ norm which is defined as:

$$\|\mathcal{F}\|^2 = \frac{1}{|\Omega|} \int_{\Omega} \mathcal{F}(x)^2 \, dx$$

(5.4.5)

Among several rest cases they presented in their paper, we only borrow two figures. Figure 7 and Figure 8 illustrate some of the effects of filtering operator on subgrid terms. Clear obedience of subgrid terms from different filters can be observed in these two figures. In Figure 7 the $L_2$ norm of the commutator error in the streamwise direction at time $t=60$ for six different filters is presented. It is not surprising to observe one order of magnitude difference for the $L_2$ norm of the commutator error. The commutator error term emerged purely because of filtering operator; hence, logically its value has to strongly rely on the type of filter. In Figure 8 the $L_2$ norms of the SGS fluxes and the commutator error in the streamwise direction at time $t=60$ are presented for 4 different asymmetric top-hat filters. Similar observations seen in this figure indicates the importance the type of filtering and also the effect of the commutator error. An interested reader finds a comprehensive analysis on some more test cases in the article.

Figure 7 The $L_2$ norm of the commutator error in the streamwise direction at $t=60$: (a) $\alpha=1/4$ (dashed), $1/2$ (dotted), and $3/4$ (solid) with $\beta$ fixed to 10; (b) $\alpha$ fixed to $3/4$ and $\beta=5$ (dash-dotted), 10 (solid), 30 (dashed), and 60 (dotted)[40].
The authors finally concluded that: “skewness of a filter, which is often unavoidable, can have a strong influence on the size of the commutator error relative to the SGS flux. Moreover, sufficiently rapid variations in the local filter width may induce local situations in which the commutator error is no longer negligible compared to the SGS fluxes. Finally, the use of higher-order filters does not offer an independent control over the size of either type of subgrid terms. Hence, in various situations or for the sake of retaining appropriate efficiency in LES of complex flows, one has to resort to explicitly developing specific models for the commutator error”. Similar results and discussion can be found in [2][8][9][19], yet with different motivations. For sake of brevity we avoid repeating their results here.

All of these results and conclusions basically support our discussion in the previous section that one should strictly include filtering operator characters into account for using any filtered equation. One should note that in the above results only one of the commutator errors were studied, it is not difficult to predict that including other terms will intensify the effects of commutator errors. Now that we have seen strong ties between filtering operator and new terms in actual flow simulations, in the next section, we aim to study the ‘real’ LES equations.
5.5. Methods and Assumptions for Conventional LES

In this section we focus on describing the conventional governing equations for large eddy simulations. In order to facilitate the discussion, we define Navier-Stokes operator as:

\[ \mathcal{N}S(u_i, p) = \partial_t u_i + \partial_j (u_i u_j) + \partial_i p - \frac{1}{Re} \partial_j u_i \]  

(5.5.1)

Therefore, the flow equations can be written as:

\[ \mathcal{N}S(u_i, p) = 0 \]  

(5.5.2)

and to be more explicit, in this section we specify the type of filter and its length parameter whenever filtering operator is used:

\[ \mathcal{L}^{F_T}_\Delta \{ f(x, y) \} = f(x, y) \Big|_{\mathcal{L}^{F_T}_\Delta} \]  

(5.5.3)

Now, if the filtering operator \( \mathcal{L}^{F_T}_\Delta \) is applied on Navier-Stokes equations:

\[ \mathcal{L}^{F_T}_\Delta \circ \mathcal{N}S = \mathcal{L}^{F_T}_\Delta \{ \mathcal{N}S(u_i, p) \} \]  

(5.5.4)

then using Equation (5.2.13) the result is summarized as:

\[ \mathcal{N}S(\bar{u}_i |_{\mathcal{L}^{F_T}_\Delta}, \bar{p} |_{\mathcal{L}^{F_T}_\Delta}) = -\partial_j \left( \mathcal{L}^{F_T}_\Delta, \Pi \right) (u_i, u_j) \]

\[ - \left[ \mathcal{L}^{F_T}_\Delta, \partial_i \right] (p) - \frac{1}{Re} \left[ \mathcal{L}^{F_T}_\Delta, \partial_j \right] (u_i) + \left[ \mathcal{L}^{F_T}_\Delta, \partial_j \right] \left( \Pi (u_i, u_j) \right) \]  

(5.5.5)

Then, the filtered equation can be further summarized if commutator errors and turbulent stress tensor are shown as

\[ \mathcal{C}E_i |_{\mathcal{L}^{F_T}_\Delta}(p, u_i) = \left[ \mathcal{L}^{F_T}_\Delta, \partial_i \right] (p) - \frac{1}{Re} \left[ \mathcal{L}^{F_T}_\Delta, \partial_j \right] (u_i) + \left[ \mathcal{L}^{F_T}_\Delta, \partial_j \right] \left( \Pi (u_i, u_j) \right) \]  

(5.5.6)

\[ \tau_{ij} |_{\mathcal{L}^{F_T}_\Delta}(u_i) = \left[ \mathcal{L}^{F_T}_\Delta, \Pi \right] (u_i, u_j) \]  

(5.5.7)

Finally, the filtered Navier-Stokes equations are expressed in this form
\[ N\mathcal{S}(\bar{u}_i|_\Delta^{F,T}, \bar{p}|_\Delta^{F,T}) = -\tau_{ij}|_\Delta^{F,T}(u_i) - C\mathcal{E}_i|_\Delta^{F,T}(p, u_i) \] (5.5.8)

So far, no assumption or simplification has been made. Clearly the RHS of this equation is unclosed and needs to be modeled or estimated as a function of known variables \( \bar{u}_i \) and \( \bar{p} \). Conventionally, it is quite common to ignore the effect of commutation errors \( C\mathcal{E}_i|_\Delta^{F,T}(p, u_i) \), as there have been little knowledge about their modeling \[14\][33][36][32][1][5][30][40][19][2], therefore the filtered equations reads as:

\[ N\mathcal{S}(\bar{u}_i|_\Delta^{F,T}, \bar{p}|_\Delta^{F,T}) = -\tau_{ij}|_\Delta^{F,T}(u_i) \] (5.5.9)

Moreover, in most (and strictly speaking all) present-day models for turbulent stress tensor, the type of filtering operator and its characteristics is disregarded \[14\][36]. Most LES models are inherently physical models which are spiced by some filter-like parameters such as \( \Delta \). Since these physical models are not derived from the same filtering operator which is applied to the flow equations, they are usually involved with uncertainties and difficulties to define their filter-like parameters and symbols \[36\]. Therefore, the basic form of most conventional models are designed independent of the type filter, yet secondary filtering procedures might be used in some modeling procedures such as dynamic models \[14\][15]. One should note it is difficult to argue that the secondary filtering routines represent the type of filtering operator as they are merely used to estimate some parts or components of a physical model such as their coefficients. For the case of similarity models, not only justification of the type of filtering is difficult but also the length parameter is absent in these types of modeling. Generally speaking, subgrid models are largely independent of the specific choice of the filter \[14\][36]. Assuming, LES models include, at least, some kind of effect of \( \Delta \), then the flow equations read as:

\[ N\mathcal{S}(\bar{u}_i|_\Delta, \bar{p}|_\Delta) = -\tau_{ij}|_\Delta(u_i) \] (5.5.10)

And by ignoring the effect the secondary filtering routines, this means:

\[ \tau_{ij}|^{Top \ hat}_\Delta = \tau_{ij}|^{Gaussian}_\Delta = \tau_{ij}|^{cut-off}_\Delta = \tau_{ij}|^{Any \ random \ Filter}_\Delta \] (5.5.11)

Therefore, in the conventional framework of LES, all strong ties of filtering operator are ignored. Consequently, considering Equation (5.5.10), the following identity is inevitable:
\[
\bar{u}_i |_{\Delta}^{Top\, hat} = \bar{u}_i |_{\Delta}^{Gaussian} = \bar{u}_i |_{\Delta}^{cut-off} = \bar{u}_i |_{\Delta}^{Any\, random\, Filter} \quad (5.5.12)
\]

That is in sharp contrast of what were presented in previous sections. All mathematical explanations in Section 5.3 and numerical experiments using “perfect LES” make it difficult to believe that Equation (5.5.10) has any correlation with Equation (5.5.9). As stated in Section 5.3 ignoring or misrepresenting filtered terms is equivalent to removing the filtering features from the equations and Equation (5.5.12) clearly manifests that the rule of filtering is eliminated from the solution of filtered equations.

Contradictory facts about the solution of Equation (5.5.10) can be listed as:

- Actual LES solutions are independent of the type of filtering operator
- Despite the importance of commutation errors, as shown in the perfect LES, they are ignored in calculations of actual LES simulations. Although they can be extremely important if very non-uniformed or skewed filtered are used, there is little evidence in the actual simulations that ignoring them causes any significant problem! Successful LES simulations using non-uniformed or skewed filtered can be found almost in any reported LES simulation which ignored the commutation errors, for instance [14][36][30][31].
- As demonstrated in Section 5.3, since filtering is an errorless operator, any length parameter \( \Delta \) is acceptable and it has to lead to a solution, nonetheless not a unique solution. However, it is well-known that using large length parameters \( \Delta \) leads to no turbulent solution (Laminar solution) in conventional LES template.

In spite of all these contradictory facts and crude estimations for Equation (5.5.9), this question might rise why these unrealistic estimations eventually leads to successful LES simulations. It is not difficult to find a reason for these successful solutions which are basically built on inconsistent realities in the conventional LES template. Recalling the mathematical tools in Section 5.3, a more generic form of resolving operator \( R \), 3 dimensional resolving operator \( R^{1,2,3} \) using \( N_1, N_2 \) and \( N_3 \) nodes in 1, 2 and 3 dimensions is defined as \( R^{3}_{n=N_1,N_2,N_3} \), therefore if it is applied on Equation (5.5.10):

\[
R^{1,2,3}_{n=N_1,N_2,N_3} \left\{ \mathcal{NS} (\bar{u}_i |_{\Delta}, \bar{p}|_{\Delta}) + \tau_{ij} |_{\Delta} (u_i) \right\} = R^{1,2,3}_{n=N_1,N_2,N_3} \left\{ \mathcal{NS} (\bar{u}_i |_{\Delta}, \bar{p}|_{\Delta}) \right\} + R^{1,2,3}_{n=N_1,N_2,N_3} \left\{ \tau_{ij} |_{\Delta} (u_i) \right\}
\]
As a result, if the effect of $R_{n=N_1,N_2,N_3}^{1,2,3} \{ \tau_{ij} \mid_{\Delta} (u_i) \}$ is considered small (which is the case for all conventional LES subgird models [36])

$$\lim_{R_{n=N_1,N_2,N_3}^{1,2,3} \{ \tau_{ij} \mid_{\Delta} (u_i) \} \to 0} R_{n=N_1,N_2,N_3}^{1,2,3} \left\{ \mathcal{N} \mathcal{S}(\bar{u}_i \mid_{\Delta}, \bar{p} \mid_{\Delta}) + \tau_{ij} \mid_{\Delta} (u_i) \right\} = R_{n=N_1,N_2,N_3}^{1,2,3} \{ \mathcal{N} \mathcal{S}(u_i, p) \}$$

(5.5.14)

therefore the numerical solution of conventional LES template converges to the numerical solution of the original Navier-Stokes equations (Compare it with the interpretation of Equation (3.27)). This means, the conventional LES template mainly solves the original flow equations rather than filtered equations! Therefore, the solution of conventional LES template should not be interpreted as the solution of the filtered flow equations. Note, as stated before, mathematically speaking, there is no reason that one should confine the effect of filtering small in order to consider small values for $\tau_{ij} \mid_{\Delta} (u_i)$, however it is not the case in the conventional LES simulations. There is little flexibility for the size of $\Delta$ and any large enough $\Delta$ makes the simulation converged to a laminar solution.

Considering all these facts, nor the mathematical establishment of Equation (5.5.10) neither the numerical solutions of this equation provide any signature of the filtering operator characteristics. The success of Equation (5.5.10) merely relies on the smart decomposition of non-linear terms in flow equations such that the original form of Navier-Stokes stays intact. The additional and usually positive viscosity added by subgrid models is mainly based on physical understanding of turbulent flows interactions rather than including the filtering effects. Conceptually, since the unclosed terms are emerged from filtering operator, which is a purely mathematical operator, they tend to represent the filter properties rather than representing a physical quantity. As a result, modeling such terms using physical intuition and perceptive, which is the dominant strategy, seems conceptually incompatible. As a result, the conventional template is unable to satisfy basic mathematical principles of a filtered solution. For instance:

- Why the effect of subgrid terms has to be small unless there would be no turbulent solution for such simulations
Why ignoring important terms such as commentator errors leads to successful results

How independent modeling of subgrid terms from the filtering operator guarantees convergence of the resultant equation to a filtered solution

Being convinced of little or no existence of filtering effects in actual LES template, in the next section we provide a new perspective on large eddy simulations.

5.6. New Perspective for Large Eddy Simulations

It was shown that so called filtered flow equations used in the standard large eddy simulations encounters conceptual and mathematical inconsistencies. In addition, the numerical results and conclusions in Section 5.3 provide no motivation for using filtering operator in governing equations. We explained and showed the effect of filtering operator “is not” removing the effect of small scales, even though intuitive understanding suggests a contrary opinion. Also, as indicated in the final conclusion of Section 5.3 neither small nor large filtering length parameter \( \Delta \) demonstrates any practical benefit. Consequently, we propose a different approach for resolving large eddies in a turbulent flow by which we would be able to explain some of the inconsistencies observed in the conventional LES framework.

Assume the exact solution of a turbulent flow is obtained using infinite number of nodes in all three dimensions, then we donate the exact solution as \( \mathcal{R}_{\infty}^{1,2,3} \{ \mathcal{N} \mathcal{S}(u_t, p) \} \) which is defined as:

\[
\lim_{n \to \infty} \mathcal{R}_{n}^{1,2,3} \{ \mathcal{N} \mathcal{S}(u_t, p) \} = \mathcal{R}_{\infty}^{1,2,3} \{ \mathcal{N} \mathcal{S}(u_t, p) \} \tag{5.6.1}
\]

On the other hand, if resolving operator uses finite number of mesh \( n_t \) in order to capture larger scale of motions in a flow then, similar to Equation (5.3.19), we can write:

\[
\mathcal{R}_{\infty}^{1,2,3} \{ \mathcal{N} \mathcal{S}(u_t, p) \} - \mathcal{R}_{n_t=N_1,N_2,N_3}^{1,2,3} \{ \mathcal{N} \mathcal{S}(u_t, p) \} = \text{Error}_{|_{u_t \text{ or } p}(n_t, Discretization Schemes)} \tag{5.6.2}
\]

The error term is obtained since resolving operator is associated with a systemic error. Without losing generality, we avoid presenting the error of \( p \) in the rest of manuscript. Clearly, since \( n_t < \infty \), the solution of \( \mathcal{R}_{n_t=N_1,N_2,N_3}^{1,2,3} \{ \mathcal{N} \mathcal{S}(u_t, p) \} \) only includes the effect of scales larger than the local
mesh spacing which ‘intuitively’ is the effect of large eddies in a turbulent flow. However, one might be interested to improve this solution by including the effect of unresolved scales. To do so, we rearrange Equation (5.6.2) as:

\[
\mathcal{R}^{1,2,3}_{\infty}\{NS(u_i,p)\} = \mathcal{R}^{1,2,3}_{n_1=N_1,N_2,N_3}\{NS(u_i,p)\} + Error|_{u_i}(n_1,\text{Discretization Schemes})
\]

(5.6.3)

Therefore, adding the exact value of discretization error retains the exact solution for of the turbulent flow, nevertheless it is not determined. Consequently, any appropriate modeling for this term can partially retrieve the exact solution. This means:

\[
\mathcal{R}^{1,2,3}_{\infty}\{NS(u_i,p)\} \approx \mathcal{R}^{1,2,3}_{n_1=N_1,N_2,N_3}\{NS(u_i,p)\} + \text{Modeling}\{Error|_{u_i}(n_1,\text{Discr. Schemes})\}
\]

(5.6.4)

Or

\[
\mathcal{R}^{1,2,3}_{\infty}\{NS(u_i,p)\} \approx \mathcal{R}^{1,2,3}_{n_1=N_1,N_2,N_3}\{NS(u_i,p)\} + C_i(n_1,\text{Discr. Schemes})
\]

(5.6.5)

which \(C_i\) denotes suitable modeling of error for the i’th direction of the momentum equation. Hence, different methodologies can be used to model this error. Clearly the modeling has to consider the discretization error as the primary source of error. Nevertheless, the model might be designed based on physical understanding of small scale turbulent interactions, as their effects have overlooked due to large mesh spacing. In this formulation, including the effect of unresolved eddies are well justified, as it is the exact consequence of the lack of numerical resolution. It is interesting that most LES models which were developed in the conventional LES template can be easily modified to this formulation, since they are mainly base on representing physical effect of unresolved scales rather than filtering effect. Another appealing point in the above formulation is its ability to predict where in flow more correction should be considered. For example, if at some part of flow simulation, the mesh resolution is comparable to Kolmogorov length scale, no correction should be added at those parts. Therefore, the correction would be proportional to resolution (instead of using undetermined length parameter for a filter \(\Delta\)). For instance, for a channel flow in which refined mesh close to walls and coarse mesh around
center line is used, less or no correction is needed close to walls. This justifies why in conventional LES models one should often use wall functions close the wall. Worth noting, justifying wall functions within the context of conventional LES is difficult, as more commentator errors appear in that formulation close to walls. In addition, the ambiguity of defining a suitable length parameter for different models has no room in the new formulation. This is because, the error merely appear due to the lack of resolution. Therefore, the most suitable length scale for modeling is the mesh spacing which can be variable in different parts of simulation. Moreover, unlike the standard template, this variation of mesh resolution adds no serious difficulty for modeling the error term. This readily means skewed, non-uniform mesh can be conveniently utilized without being worried about its commutative errors. It is now clear why ignoring the commutative errors in the standard LES template leads to no serious difficulties. The new formulation clearly permits using any type of mesh configuration as long as they are enough to resolve the instabilities in the turbulent flow. Low mesh resolution, therefore, prohibited the growth of instabilities and as a result, the simulation converges to laminar solution.

Considering the above explanation, the new formulation is not encountered with the inconsistencies explained in the conventional LES framework. Perhaps the most important features can be summarized:

- Unlike the standard LES, non-uniform or skewed mesh can be conveniently used without introducing any new error except the resolving operator error
- Mesh spacing can be used as appropriate and justifiable length parameter for modeling purposes. This removes the difficulty of defining a filter length parameter in the standard template, as any definition could not be mathematically justified.
- Using physical based models is logical, since they represent the effect of smaller scale than the mesh resolution. Unlike the standard template, they are not responsible to project the effect of the “mathematical” filtering operator.
- Modeling in the new framework is based on the discretization errors. These errors can be determined using the schemes which are used in the resolving operator. This means for different discretization, different models can be developed. As a result, modeling procedure finds some clear mathematical foundation. This is a different approach than the
standard template in which modeling had to be done based on unavailable DNS data and its relation with the LES mesh configuration.

- There is no mathematical inconsistency in this framework, as opposed to the standard LES.

5.7. Modeling the Correction Error $C_i$

In this section we aim to model the correction error appeared in Equation (5.6.5). We mainly intend to introduce a modeling procedure for $C_i$ rather than introducing the best modeling technique. As part of our modeling, we will use some hypotheses which to our knowledge are reasonable assumptions. Clearly, varieties of models can be produced based on different hypotheses and procedures.

Perhaps the most popular methodology for modeling the effect of small eddies in a turbulent flow is the analogy of momentum transfer caused by turbulent eddies with the momentum transfer caused by molecular diffusion. This hypothesis is often referred as Boussinesq eddy viscosity assumption and it was first formulated by Boussinesq in 1877 [42]. This model basically defines a new viscosity, called turbulent viscosity, using a suitable length scale $\ell$ and velocity scale $u$. The eddy-viscosity assumption is the basis of many famous models such as Smagorinsky model [26][37]. Considering the theoretical and numerical studies in [4][11][22][25][44][45], in our modeling, forward energy cascade is recognized as the dominant mechanism being responsible of the effect of unresolved scales. Therefore, the act of subgrid scales on the resolved scales is considered strictly as dissipative action. This means the turbulent viscosity attains no negative values. Besides, we assume that if at any part of a numerical domain the mesh resolution is comparable to DNS resolution, the correction term in Equation (5.6.5) is zero. Consequently, we define the length scale for our eddy-viscosity model as:

$$
\hat{l}_i = \begin{cases} 
0 & \text{if } \delta x_i < \frac{1}{4} \ell_{\text{Kolmogorov}} \\
\delta x_i & \text{if } \delta x_i \geq \frac{1}{4} \ell_{\text{Kolmogorov}}
\end{cases}
$$

(5.6.6)

in which $\ell_{\text{Kolmogorov}}$ and $\delta x_i$ denote the local Kolmogorov length scale and mesh spacing in the $i$’th direction. The ‘$\frac{1}{4}$’ factor in the above definition is based on resolving a Kolmogorov sized
eddy using 4 nodes which is assumed to be enough for a DNS simulation. Using strain rate
tensor $S_{ij}$ and the above length scale we define velocity scale $\hat{u}_i$ as

$$\hat{u}_i = \hat{I}_i |S| \quad ; \quad S_{ij} = \partial_j u_i + \partial_i u_j \quad ; \quad S = .5S_{ij}S_{ij} \quad (5.6.7)$$

Therefore, turbulent viscosity in the i’th direction is defined as:

$$\nu_i = c\hat{u}_{(i)}\hat{I}_{(i)} = c(\hat{I}_{(i)}|S|)\hat{I}_{(i)} = c\hat{I}^2_i |S| \quad (5.6.8)$$

We identify this viscosity as the entire act of small scales on the resolved scales. So far, we have merely rely on our physical understating of interaction of unresolved and resolved scales, we have not yet included the effect discretization in the modeling of the correction term $C_i$.

In order to include the effect of discretization, for sake of simplicity, we ignore all linear
terms of Navier-Stokes; as a result we study the effect of discretization on this form of flow
equation:

$$\partial_t u + u\partial_x u + v\partial_y u + w\partial_z u = 0 \quad (5.6.9)$$

Clearly, there are a number of methods to discretize this equation. If for example, we use
upwinding to discretize $\partial_x u$ then the first spatial term reads as:

$$u\partial_x u = u_i \frac{1}{\Delta x} (u_i - u_{i-1}) \quad (5.6.10)$$

where i indicates the i’th node in X direction. Then using Taylor series:

$$u_{i-1} = u_i - \Delta x \partial_x u_i + \frac{1}{2} \Delta x^2 \partial_{xx} u_i + \ldots \quad (5.6.11)$$

$$\frac{1}{\Delta x} (u_i - u_{i-1}) = \partial_x u_i - \frac{1}{2} \Delta x \partial_{xx} u_i + \ldots \quad (5.6.12)$$

Which means the first-order upwind approximation of $\partial_x u$ is basically act as the RHS of
Equation (5.6.12). Similar expressions can be derived for $v\partial_y u$ and $w\partial_z u$; after some
rearrangements, the following equation is obtained by substituting these types of approximations:

$$\partial_t u_i + u_i \left[ \partial_x u_i - \frac{1}{2} \Delta x \partial_{xx} u_i \right] + v_i \left[ \partial_y u_i - \frac{1}{2} \Delta y \partial_{yy} u_i \right] + +w_i \left[ \partial_z u_i - \frac{1}{2} \Delta z \partial_{zz} u_i \right] = 0$$
We are now able to reform this numerically discretized equation into an equivalent analytical equation which is conventionally called modified equation for Equation (5.6.9):

\[
\frac{\partial f}{\partial t} + u_j \frac{\partial f}{\partial x_i} = \lambda_j D_{ij}
\]

(5.6.15)

\[
\lambda_j = \frac{1}{2} \delta x_{(j)} u_{(j)} \quad ; \quad D_{ij} \equiv \delta y_{(j)} \delta z_{(i)}
\]

(5.6.16)

here “i” and “j” are tensor indexes. Comparing Equation (5.6.15) and Equation (5.6.9) the dissipative nature of first-order upwinding is summarized as \(\lambda_j D_{ij}\). \(\lambda_j\) basically plays a similar role as viscosity coefficient, yet anisotropic coefficient. On the other hand, if Equation (5.6.9) is discretized using central difference scheme, similar to the above procedure, the approximation for special derivatives read as:

\[
\frac{1}{2\Delta x} (u_{i+1} - u_{i-1}) = \partial_x u_i + \frac{1}{3} \Delta x^2 \partial_{xxx} u_i
\]

(5.6.17)

And therefore, the leading error is dispersive instead of dissipative. This means central difference schemes introduce almost no dissipation into the solution. Obviously, each of these two schemes and any other scheme have a unique effect on the correction term in Equation (5.6.5) and therefore they must be modeled differently. To clarify this, we limit our modeling technique to the above two schemes. Using the results of eddy-viscosity model in Equation (5.6.8) and considering the effect of each discretization scheme the correction term in Equation (5.6.5) is approximated as

\[
C_i = \begin{cases} 
\partial_j \left( V_{(i)} \partial_j u_{(i)} \right) & \text{For Centeral Difference} \\
\text{Max} \left\{ \partial_j \left( V_{(i)} \partial_j u_{(i)} \right) - \lambda_j D_{ij} , 0 \right\} & \text{For Upwinding}
\end{cases}
\]

(5.6.18)

In modeling \(C_i\) the effect of discretization is directly projected in the final expression. In the above procedure, the physical understanding of interaction of different turbulent length scales...
has been included considering the effect of discretization schemes [20]. This means, the basic Smagorinsky model has to be implemented differently in low accurate finite-volume framework and high accurate spectral framework, for instance. This is aligned with the numerical experiments in conventional LES framework, as turbulent viscosity models are implemented independent of the types of discretization schemes. For example, Smagorinsky-based models were evaluated in the framework of finite-volume by Majander and Siikonen [26]. They found that none of the tested models improves the results of low Reynolds number simulation for any grid level compared to the calculations with no model. Najjar and Tafti [28] also indicated that the inclusion of the dynamic model or any subgrid scale model with finite-difference approximations will not always result in better predictions of the flow field.

As stated earlier, the above procedure was merely presented to provide an insight for modeling techniques. The new straightforward template of LES simulation is compared with the conventional LES templates in Diagram 1. It is important to note that implementing the new template is very similar with the standard LES implementation. However, it is not involved with any complication caused by applying a low-pass filter to equations. This readily means, any mesh configuration and distribution can be consistently used. We leave numerical experiments of implementing these types of modeling for future work.
Comparing LES Templates

**Standard LES**

\[ NS(u_i, p) = 0 \]

Filtering

\[ \mathcal{L}^{f,T}_\Delta \{ NS(u_i, p) \} \]

\[ NS(\bar{u}_i |^F,T_\Delta, \bar{p}|^F,T_\Delta) = -\tau_{ij} |^F,T_\Delta (u_i) - \mathcal{C} \varepsilon_i |^F,T_\Delta (p, u_i) \]

Disregarding unknown terms and effects

\[ NS(\bar{u}_i |_\Delta, \bar{p}|_\Delta) = -\tau_{ij} |_\Delta (u_i) \]

Modeling subgrid term using physical understanding of turbulent interactions and independent of the choice of filter

Resolving the final closed equations

\[ \mathcal{R}^{1,2,3}_{n=N_1,N_2,N_3} \{ NS(\bar{u}_i, \bar{p}) + \tau_{ij}(\bar{u}_i) \} \]

**Proposed LES**

\[ NS(u_i, p) = 0 \]

Applying Resolving Operator

\[ \mathcal{R}^{1,2,3}_{n=N_1,N_2,N_3} \{ NS(u_i, p) \} + \text{Modeling}\{\text{Error}(n_1, \text{Discr. Sche.})\} \]

Resolving

\[ \mathcal{R}^{1,2,3}_{n=N_1,N_2,N_3} \{ NS(u_i, p) \} + C_i(n_p, \text{Discr. Schemes}) \]

Final Solution in Numerical Space

Diagram 1 Comparing and presenting the standard and proposed LES templates
References


Part III
Turbulent Two-Phase Flows
Chapter 6

Interfacial Turbulent Flow

Wide range of free-surface deformations occur when a free surface is deformed by fluid motions. Similar to any flow regime, the existence of turbulence considerably affect these types of deformations. There are different sources for this turbulence such as breaking waves at the interface or preexisted turbulence from submerged jets or boundary layers. The effect of turbulent structures on the free surface is restricted by surface tension which tends to be influential on small length scales and gravity which has a dominant effect on large scales [4][7]. Many aspects of the turbulent flows and the transport mechanisms, e.g. generation and interaction processes at the interface and its roles are not clear yet. Turbulence is highly anisotropic at the interface, due to the surface tension and gravity forces restrictions; hence predominately parallel turbulent fluctuations to the surface. Surface deformation and vortices near the free surface generate vorticity normal to the free surface [29] [48]. Moving boundary, existence of vorticity normal to the interface, complicated and rapid deformation, ligament formation, breakup and merging of the interface are some aspects of turbulent interfacial flows that make this type of flows different from the other turbulent flows.

Basically, the accurate prediction of interface properties, such as velocity and location is a complicated topic which needs much research. Although some analytical studies have been conducted to generate better model, it needs more attempts to do so. At a glance, analytical formulation, built on a surface-moving coordinate frame, can predict the vorticity components and its flux from an arbitrary interface. These formulas are function of velocity field, interface curvature and fluid properties [47]. Among reviews of turbulence at a free surface or closely related topics, are those of Sarpkaya [37], Tsai & Yue [43], Melville [31] and [42].

Vorticity which is continuously created from the interface plays a key rule in the characteristics of interface. In most cases, the process of creation, advection and diffusion of vorticity is highly unsteady and three dimensional. The vorticity creation from an interface may
fall into a more general subject which is the vorticity dynamics on boundaries. This boundary can be a rigid wall, or an interface. The dynamics of vorticity was first studied for rigid-wall type of boundary (Lighthill [21]) and it was further developed by Wu and Wu [48] for an arbitrarily curved moving rigid surface. It has long been known that on a free surface there is a viscous boundary layer [20][46], hence the physics of vorticity creation from a rigid wall reappears in the interfacial vorticity dynamics. Despite the common characters of the vorticity creation from a rigid wall and an interface, there exist some fundamental differences between the two cases. The driving mechanism for the interfacial vorticity field is the continuity of tangential surface stress across the interface rather than that of tangent velocity. Using this difference, it can be shown why inviscid solution or potential flow works well when applied for water waves theory. Clearly, inviscid solution by no means can be justified for solid wall boundaries. It can also be shown that vortex lines can attach to a fluid-fluid interface, whereas they cannot attach at a no-slip boundary on a rigid wall.

Being described by irrotational flow, water waves which are the typical motion of a free surface can not be considered as part of a turbulent flow. Yet if they break, they are a potent source of turbulence. Also, in resonant interaction theory for water wave generation, it is shown that similarity of time and length scale of weak turbulence might generate water waves [33][41]. When turbulence is not strong, a passive response to the turbulent fluctuation of pressure can be seen on a weekly deformed free surface. However, for moderate strength turbulence, surface waves might strongly damped if their wavelength is similar to, or less than the dominant turbulence length scales. As mentioned earlier, interaction of vorticity and free surface plays a key role for all these deformations. For example, vertically oriented vortices can cause depression at the surface. Scars, which can be common feature where turbulence disturbs an almost a flat surface, are manifestation of the interaction of patches of vertical flow with a free surface [28][37]. Scars are cusp- or corner- like notches of the water surface.

A descriptive and interesting study is presented in [4][5][3]. In their study they classified different regimes for interface states using Froude and Weber numbers. They used descriptive terminologies in order to explain different complicated situations. Moderately coherent and discrete volumes of fluid that move upward to the surface disturbing, moving parallel or perhaps falling back on it was called `blobs'. These blobs can form the boundary of the turbulence, and
thus might be represented by the major 'billows' at the edge of an active turbulent region. Assigning a length scale, $L$, and overall velocity $q$, with a representative kinetic energy density $= \frac{1}{2} q^2$, they quantified these blobs. It was supposed $L$ and $q$ are independent parameters and turbulence happens if Reynolds number is $Re = \frac{qL}{v} > 100$. To include the effect of gravity, the corresponding specific energy densities, $gL$ and $\frac{1}{2} q^2$ were used to form turbulent Froude number $Fr = \frac{q}{\sqrt{2gL}}$. Low Froude number, gravity dominated flow, results in nearly flat surface, yet strong surface displacement for shorter length scales of around $\frac{q^2}{2g}$ is also predicted. Specific surface energy is defined as $S = \frac{T}{\rho}$ where $T$ is surface tension and $\rho$ is density to quantify the effect of surface tension. As a result, Weber number is defined as $= \frac{q^2L}{2S}$. Little or no disturbances to the surface are the result of a Low Weber number; and conversely, disintegration of surface into drops might be associated with high Weber number.

Since both $Fr$ and $We$ affect the state of interface, $L - q$ diagram [4] is divided into four different regions using critical values for these two non dimensional numbers: $Fr_c$ and $We_c$.

\[
q = \sqrt{2Fr_cgL} \Rightarrow \log(q) = \xi(Fr_c) + \frac{1}{2} \log(L)
\]

\[
q = \sqrt{\frac{2We_cS}{L}} \Rightarrow \log(q) = \xi(We_c) - \frac{1}{2} \log(L)
\]

where $\xi$ and $\xi$ are constant numbers. In the diagram line $Re = 100$ is also shown with long dashed. Below and to the left of this line viscosity has a dominating effect on the evolution of any possible turbulence generation. A shaded area in the diagram indicates the range of variations between surface that is no longer smooth or possibly broken because of turbulent flow, yet the area can not be determined with precise values for $Fr$ and $We$. Further details regarding the critical values can be found in [4].
Region 0, weak turbulence: both the Froude number and the Weber number of the turbulence are small. There is little or no surface disturbance at this quadrant of the \((L, q)\)-diagram. Multitude length, time and velocity scales are needed to represent the flow, as time and

Figure 4. Diagram of the \((L, q)\)-plane for water and the corresponding descriptions of the flow regimes [4].
length scales similar to those satisfying the linear surface wave description relation is likely to occur. Important studies have been conducted such as Gibson & Rodi [11], Rashidi et al. [34], Handler et al. [12], Borue et al. [2], Walker et al. [44], Walker, Leighton & Garzanos [45] and Shen et al. [38] on the interaction between turbulence and such a free surface.

**Region 1, surface tension dominated turbulence:** Large Froude number and small Weber number results in smooth rounded surfaces. The dominant effect of surface tension maintains the cohesion of the liquid and therefore, gravity fails to keep the surface flat. Relatively small-scale turbulence usually occurs such as length scales of the order 1 cm and less for water. For turbulence below the critical region there is not much wave generation and the surface is likely to be knobbly. In other experiments such as [32], intense turbulence with very small length scales comparable to breaker is reported. Small breaking waves so called Microbreakers are likely to be seen in this region. Microbreakers are known to play a key rule for heat and mass transfer between ocean and atmosphere. Slightly strong wind on water surface can result in these Small breaking waves.

**Region 2: Very strong turbulence:** Both gravity and surface tension fail to keep the cohesion of the flow giving rise to breaking up into drops and bubbles. In this regime, any strongly generated turbulent eddy is not restrained by the inertia of surrounding non-turbulent liquid. Maintaining its speed, any blob of liquid erupts the surface when it approaches to the surface. However, for weak turbulence, the vertical velocity fluctuations $w$ approach zero at the free surface. This means, in strong turbulence $w$ can be expected to be larger than the horizontal fluctuations, $u$ and $v$, as they are more constrained by the inertia of the surrounding liquid. That is why different regimes have to be modeled differently.

**Region 3, Gravity-dominated turbulence:** Low Froude number and large Weber number results in flat of nearly flat surfaces. Gravity dominated turbulence is the most common state which applies to almost all terrestrial water bodies with flow: streams, rivers, seas and oceans. Staying away from splashing region, Linearized boundary conditions can be considered for this regime. At this region, turbulent has adequate energy to disturb the interface, but only at length scales which are smaller than the main turbulent eddies. This can be seen, if a horizontal line corresponding to relatively constant turbulent kinetic energy is drawn in the diagram. Along this
line, a point in region 3 when leads to smaller values of $L$ often can reach the boundary with the splashing region manifesting the effect of turbulent eddies on the interface. Waves, vortex dimples and scars are the most characteristic features of these smaller regions. These can give rise to localized surface breaking, and hence initiate to the turbulent flow aspects.

Figure 5. Different flow regimes and interface topologies for water-air interface
The advancement in computer technology and numerical methods has made it possible to conduct numerical simulation of turbulent interfacial flows. However, due to the complexity of the simulations, almost all simulations to date either simplify the simulations or impose limiting criteria. For example, simulation presented in [1][10][15][22][40] focused on one phase modeling using simplified water-wave theories. Other simulations such as Lombardi et al. [27] and Liu et al. [26] considered a flat interface to study a countercurrent gas-liquid turbulent flow. The extension of Fulgosi et al. [9] to a deformable interface with a focus on effects of small-amplitude wave on gas side turbulence was restricted to using boundary fitting method which cannot be extended to strong topological changes of the interface. Therefore they used $We = 4.8 \times 10^{-3}$ and $Fr = 8.7 \times 10^{-5}$ in order to limit the elevation amplitude and steepness of the free surface. Moreover, simulations presented in [2][12] [19] [38] [45] avoid the significance of the effect of air side, hence, their result is valid if interaction between air and water is not significant such as low wind situations. Similar idealized studies which closely address this subject can be found both for DNS and LES simulations [8][6][13][14][16][17][30][18] [23][24][25][35][39]. Although one might expect substantial subgrid modeling in large eddy simulations, yet this goal still needs considerable progress in the field. Relatively simple modeling concepts such as damping viscosity close to the interface have been introduced while other interfacial terms are escaped from modeling [24]. Perhaps methods and modeling concepts used and derived for RANS equations [3][14][39] might help for future subgrid models in LES simulations. Clearly, insightful DNS studies such as [15] can considerably help to find suitable models. As mentioned before and also shown in [15], subgrid models merely based on wall boundary layer or free-slip surface layer might not be sufficient to capture coherent structures at the interface, even though there are similarities between them.

We in this work have developed a consistent numerical method in which numerical errors due to spurious currents have been minimized. Spurious currents which occur both because of surface tension forces and gravity forces might introduce substantial errors into simulation. Although it has received less attention in simulation in this field, numerical errors due to spurious currents can be in the same order of magnitude as the characteristic velocity of a simulation. Introducing piezometric pressure and accurate numerical techniques to implement both surface tension forces and gravity forces are aimed to avoid these types of errors. Moreover, using Level Set function to track the interface allows more sophisticated simulations. All these
establish a ground for accurate DNS or LES simulations. Using distance function in the Level Set function paves the path for complicated subgrid models in which distance from the interface is crucial information.

References


Chapter 7

Closure

In modeling turbulent two-fluid flows accurate capturing of the effects of surface tension forces, gravity forces and turbulent structures pose particular challenges. Numerical errors due to inaccurate implementations of interfacial forces result in nonphysical parasitic or spurious currents, which can dominate important physical effects. Existence of turbulent structures aggravates the difficulties in these types of simulations. Although direct numerical simulation techniques, DNS, provide the most accurate solutions for turbulent flows, their substantial numerical cost disqualifies them to be considered as feasible methods. Hence, less accurate techniques, large eddy simulation methods, LES, can be considered as the alternative to the highly expensive DNS solutions. In this thesis, we developed several numerical schemes to improve the accuracy and reliability of a turbulent two-phase flow simulation. The numerical schemes were structured so that in each chapter a particular issue was addressed.

In chapter 2, we presented and developed Navier-Stokes equations for variable properties flows consistent with proper implementation of forcing terms in conjunction with Level-set function. Piezometric pressure concept was used and introduced in both analytical and numerical formulations of a multi-phase flow. It was explained and demonstrated why piezometric pressure improves a numerical solution of a two-phase flow, especially for low Froude number flows. Several numerical techniques and schemes such as estimation of continuous interfacial forces, CIF, sharp interfacial forces, SIF, and suitable pressure-velocity coupling scheme were developed in order to be compatible with pressure-based algorithms. Several test cases: static drop in equilibrium, stationary column of a two phase flow, Rayleigh-Taylor instability and falling water droplet were solved and their results were satisfactory.

In chapter 3, sharp implementation of interfacial forces was discussed. Different methods were examined, in order to find a reliable technique. It was shown that GFM based methods for estimating sharp surface forces encounter instabilities when it is used for implementing interfacial forces which includes gravity effects. Alternatively, the areal method was another
method which was tested. Using the enclosed area within a control volume and normal unit vectors at the desired location, the method accurately calculates a sharp force for a desired location. As a result of this accuracy, we showed the method improved stability of our numerical algorithm. However, areal method cannot be recognized as a versatile method, as it encounters compatibility problems with staggered estimation of interfacial forces. We found the cause of occurring spurious currents for areal method through which we then explained why using regularized Direct delta function (instead of regularized Heaviside function) in flow equations produce spurious currents. All these suggest future research to find a solution to take advantage of accurate force estimation of the areal method and regularized Direct delta function. The current formulations and methods are not able to accommodate these more accurate techniques. Finally, we introduced an alternative method to sharply implement interfacial forces. The method was tested for Raleigh-Taylor instability and a falling water droplet. The successful results assure the reliability of the method.

In Chapter 4, the standard SIMPLE algorithm is tested for simulation of turbulent channel flow. The algorithm, which is discretized using control volume method, consequences satisfactory results both for direct and large eddy simulation when second order central difference scheme is used for convective terms. However, when third order accurate QUICK scheme was used, no turbulent solution was obtained. The dissipative nature of QUICK scheme prohibits large eddy simulation to maintain unstable modes alive in the flow field.

We took advantage of implicit time integration scheme of SIMPLE algorithm to increase the size of time steps. Even though first order Euler method is used, turbulent solution is obtained with time steps three times larger than explicit methods. Comparing with the results of other references, laminarization is considerably affected by the accuracy of time integration method. First and second order time advancement methods does not allow time step as large as Kolmogorov time scale. Therefore, it would be an interesting test using third or higher order implicit algorithm to study whether a turbulent simulation can be obtained using time steps comparable to Kolmogorov time scale.

We found utilizing Smagorinsky or Dynamic Smagorinsky models has negative effects on the simulation of turbulent channel flow; yet dynamic Smagorinsky considerably works better
than Smagorinsky model. Clearly, this conclusion is confined to the test case presented in the chapter and its spatial resolution.

Due to their discretization method, SIMPLE-type algorithms are promising algorithms for general purpose simulations. Their implicit time integration scheme allows more stability especially when fine mesh is used. Considering stingy time step restriction of explicit methods, larger time steps which can be used in these types of algorithms might considerably facilitate simulations. For instance, we showed how accurate solutions, combined solutions, can be obtained with considerably less computational cost. Moreover, splitting operator errors, first order accuracy for pressure term as well as problems regarding discontinuous velocity field have no room in these types of algorithms.

In chapter 5, we proved and demonstrated the solution of the standard LES equations or so called filtered Navier-Stokes equations should be carefully interpreted, as it might not represent what it is claimed to present. The shown mathematical and numerical evidence question the authentication of the many simplifications and modeling methods which are widely used in the standard LES simulations. We found no good reason for introducing the filtering operator for simulation of large scales of a turbulent flow. Therefore, we constituted a new template for LES without using filtering operator. The new approach helped us to explain some contradictory facts in the conventional LES methods. The implementation of the new template is similar to standard LES methods, yet it is free of mathematical inconsistencies. We leave implementation of the new method for future work.

In chapter 6, we reviewed experimental and some analytical analysis for the characteristics of turbulent interfaces. It was discussed how one can categorize the state of interface using $Fr$ and $We$ numbers. Different regimes for the state of interface were introduced. For each case, dominant length and time scales were approximated. These analytical and experimental insights can be used for future numerical simulations.