Coláiste na Tríonóide, Baile Átha Cliath
Trinity College Dublin
Ollscoil Átha Cliath | The University of Dublin

WIND FIELD SIMULATION AND FLUID-STRUCTURE INTERACTION IN WIND FARMS

SUDIPTA LAL BASU

A thesis submitted in partial fulfilment of the requirements for the degree of Doctor Of Philosophy

in the

Department of Civil, Structural and Environmental Engineering
Trinity College Dublin
Ireland

June, 2023

Declaration

I declare that this thesis has not been submitted as an exercise for a degree at this or any other university and it is entirely my own work.

I agree to deposit this thesis in the University's open access institutional repository or allow the Library to do so on my behalf, subject to Irish Copyright Legislation and Trinity College Library conditions of use and acknowledgement.

I consent / do not consent to the examiner retaining a copy of the thesis beyond the examining period, should they so wish (EU GDPR May 2018).

Signature S.L.Basu

Sudipta Lal Basu
Civil, Struct. & Env. Engg.
Trinity College Dublin

Date 23/06/2023

Wind Field Simulation and Fluid-Structure Interaction in Wind Farms

Sudipta Lal Basu

1845073

Abstract

Wind energy as a renewable source has been a topic of immense interest over the past few decades. It is a well known fact that the wake structure influences the performance and operation of the wind turbines located further downstream. However, modelling the wind field as a whole along with the wind turbines involves complex geometrical models which require extensive computational resources, both in terms of memory and time. Thus, researchers have proposed different analytical models based on certain assumptions which, in some cases, have certain parameters involved which are difficult to obtain. This calls for an in-between approach such that the actual model can be built at a low cost and at the same time satisfying the theoretical requirements of the analytical models. There are certain mathematical approaches which can reduce this computational expense to a considerable extent and thus, certain special fluid-structure interaction (FSI) methods come into play. However, because of the size of a wind farm, the amount of data to be processed still remains substantial which can be easily handled computationally only if parallel numerical algorithms coupled with multi-core simulations are implemented.

A new model for laminar wind flow with vorticity and wake interaction has been proposed and its advantages over the potential flow model has been presented. This model has the ability to deal with both rotational and irrotational flow in 2D. This model coupled with a FSI method and parallel numerical algorithms has been used to simulate back-to-back actuator discs. The results were found to bear a nice resemblance with the analytical wind farm models with the added advantage of visualizing the velocity field under the light of vorticity. The added advantage is that unlike blade-resolved models which require modelling a complex mesh around the actuator disc, this model is simpler geometrically and as such is not expensive computationally. A linear spatial turbulence model has also been proposed based on Rapid Distortion Theory (RDT). The approach to develop the model uses Gaussian closure. The model has been simulated for a 2D steady fluid flow problem and the simulated results were found to be consistent. The laminar flow model with vorticity and wake interaction and the spatial turbulence model can form the basis of input to an individual wind turbine in a wind farm for subsequent analysis.

Summary

More and more nations are switching over to renewable sources of energy for a sustainable future. Wind energy is one such source which has gained considerable importance. In this thesis, a study on aerodynamics of wind farms has been carried out. Ideally, such simulations should be carried out for 3D unsteady flows. But, such models are complicated and computationally expensive. This calls for developing methods which can simplify these expensive models. The stepwise approach to achieve this objective would be to propose and test these models for 2D steady flow first before moving onto the more complicated 3D and unsteady flows. It can be understood that this entire extremely huge task. As such in this thesis only the aspects of the first step i.e. 2D steady flows are explored.

The work starts with an extensive literature review of existing simplified wind farm models. A shortcoming of these models is that they fail to address the effects Fluid-structure interaction (FSI) unless blade-resolved models are developed using commercial softwares. But again, these software models are expensive computationally which calls for looking into an approach which is something in-between. This is where special FSI methods come into play which provide the essential mathematical foundation to not only mitigate the complex software models but at the same time fulfilling the requirements of FSI. However, though using FSI reduces computational expense substantially compared to what the situation would have been with the blade-resolved models, for domains as large as wind farms extending over kilometers, amount of data getting processed still remains huge. As such, a review of parallel numerical algorithms and multi-core simulations is carried out as well to understand the way they work and how best these tools can be added to the advantage of the work carried out in this thesis.

The fluid flow usually consists of two main components, a mean component

and a turbulent component. Also, the flow is usually governed by some Partial Differential Equation (PDE). Potential flow model is one such governing PDE for irrotational flow. In this thesis, a new laminar flow model with constant vorticity or rather a governing PDE has been proposed for the mean velocity of the fluid which is more general and realistic and which takes care of both rotational and irrotational flow. Finite Difference Method (FDM) has been used to discretize the PDEs. This model is further developed in conjunction with one of the existing analytical wind farm models as well as incorporating the wake effects and FSI. The full potential of multi-core simulations and parallel numerical algorithms are used for this purpose. This new model has been used to simulate upto three back-to-back wind turbines modelled as actuator discs. The results so obtained have been found to bear a good resemblance with the theoretical values using the analytical wind farm model. A study with respect to the computational time has also been carried out and the reduction in computation time has been found to be quite significant, thereby signifying the proper implementation of multi-core simulation in the context of this new model.

For the turbulent part, a new Rapid Distortion Theory (RDT) based model has been proposed by modifying Reynold's Averaged Navier Stokes (RANS) and linearizing it using statistical approach. A Gaussian closure approach has been used for the solution. A large number of simulations has been carried out. In this case also, the simulations are done for 2D steady flows and the results obtained have been found to bear a good match with the gaussian nature of the proposed model. The laminar flow model with vorticity and wake interaction and the spatial turbulence model can form the basis of input to an individual wind turbine in a wind farm for subsequent analysis.

Acknowledgements

First of all, I am immensely grateful to my supervisors, Dr. Breiffni Fitzgerald and Prof. Biswajit Basu, Department of Civil, Structural and Environmental Engineering, Trinity College Dublin for their time and valuable suggestions which has guided me in proceeding with this challenging PhD work successfully. Their help in suggesting me the relevant resources and course works from time to time is deeply appreciated. I would like to acknowledge Dr. Kirk Soodhalter, School of Mathematics, Trinity College Dublin for his guidance and valuable suggestions in the application of parallel numerical algorithms and computational resource optimization which has formed the backbone of the numerical analysis carried out in this PhD work. I would also like to acknowledge the work of Dr.S.R.K. Nielsen, Department of Civil Engineering, Aalborg University, Denmark which has formed the base for the work on the turbulence carried out in this thesis. The time to time support I received from Dr. Darach Golden, Senior Support Scientist, Research IT, Trinity College Dublin has helped me a lot in learning MPI based programming. His suggestions and reference to resources whenever I got stuck is something I will never forget. I am thankful to Sustainable Energy Authority of Ireland (SEAI) for funding this research work without which the work itself would not have existed. Last but not the least, I am thankful to my wife, Suchana Datta, PhD student, University College Dublin for her help in teaching me certain concepts of Python which has gone a long way in successful completion of the codes to run numerical simulations. Finally, I am grateful to my father, Late Tripti Lal Basu and my mother, Shila Basu whose lessons and blessings have made me what I am today.

Contents

List of Figures

List of Tables

Acronyms

ABL Atmospheric Boundary Layer. v, 7, 11–13

AEP Annual Energy Production. 16

BEM Blade Element Momentum. 15

BiCGStab Bi-Conjugate Gradient Stable. 38–40

BLAS Basic Linear Algebra Subprograms. 6

CBC Cauchy Boundary Condition. 45

CBL convective boundary layer. v, 12

CFD Computational Fluid Dynamics. 12, 31

CG Conjugate Gradient. 38–40

CG-S Conjugate Gradient Squared. 38, 39

CSC Compressed Sparse Column. 38

CSR Compressed Sparse Row. 38

DBC Dirichlet Boundary Condition. 34, 45

DIIM Decomposed Immersed Interface Method. 23, 78

FDE Finite Difference Equation. 23, 32, 38

FDM Finite Difference Method. 31, 33

FEM Finite Element Method. 15, 23, 31, 33

FSI Fluid-structure interaction. i, 5, 7, 11, 18, 21, 24

FVM Finite Volume Method. 31, 33

GMRES Generalized Minimal Residual Method. 38

GPGPU General Purpose Graphics Processing Unit. i, 25, 26

GS Gauss-Seidel. 40

HAWT Horizontal Axis Wind Turbine. 2, 15

HPC High Performance Computing. 15, 25

IB Immersed Boundary Method. 22

IEC International Electrotechnical Commission. 3

IIM Immersed Interface Method. 23

JA Jacobi. 40

LAPACK Linear Algebra Package. 6

MDO Multidisciplinary Design Optimization. 19

MPI Message Passing Interface. i, 6, 25, 49

MSL Mean Sea Level. 13, 64

NBC Neumann Boundary Condition. 34, 45

NBL Nocturnal Boundary Layer. v, 12

NOAA National Oceanic and Atmospheric Administration. 1

NS Navier Stokes. 111

PBiCGStab Preconditioned Bi-Conjugate Gradient Stable. 40, 41, 48

PCG Preconditioned Conjugate Gradient. 40, 41, 48

PDE Partial Differential Equation. 22, 37

PDEs Partial Differential Equations. 7, 31

pdf probability density function. 109

PNA Parallel Numerical Algorithms. 18, 25

RANS Reynold's Averaged Navier Stokes. 111

RCT Rapidly Changing Turbulent. 20

RDT Rapid Distortion Theory. i, 3, 20, 107

SDG Sustainable Development Goals. 1

SEAI Sustainable Energy Authority of Ireland. 1

SGS Symmetric Gauss-Seidel. 48

SOR Successive Overrelaxation. 40

SSOR Symmetric Successive Overrelaxation. 40, 48

UN United Nations. 1

VAWT Vertical Axis Wind Turbine. 2

Notations

Table 1: Mathematical operators/symbols

Operator	Description
$\mathbf{a}$	bold character signifies vector / matrix
$< \mathbf{a}, \mathbf{b} >$	dot product of vectors $\mathbf{a}$ and $\mathbf{b}$
$\mathbf{a} \times \mathbf{b}$	cross product of vectors $\mathbf{a}$ and $\mathbf{b}$
$E[x]$ or $\overline{x}$	expectation / mean of x
x_j	component of a vector, x in direction, j
$f_x,\ f_{xx}, ...$	first, second, ... derivatives of function $f(x)$ w.r.t. x
$\|\|\mathbf{x}\|\|_2$	L2 norm of a vector, $\mathbf{x}$
$\lfloor x \rfloor$	floor value of $x, x \in \mathbb{R}$
$\sim$	order of magnitude
$\mathbb{R}$	real number
$\mathbb{N}$	natural number
δ_{ij}	Kronecker delta; $\delta_{ij} = \begin{cases} 1; & i = j \\ 0; & i \neq j \end{cases}$
$[a, b]$	range of all real numbers between lower limit, a and upper limit, b both inclusive
$[a, b] \times [c, d]$	all possible paired combinations of real numbers from the two ranges
∇^2	Laplace operator
$[\Phi]$	change (or jump) in value of a quantity, Φ at the point of discontinuity
$\|x\|$ or $\|\mathbf{x}\|$	Absolute value of a scalar, x or a vector $\mathbf{x}$
$\|\mathbf{A}\|$	Determinant value of square matrix, A

Table 2: Other Notations

Symbol	Description
A_i	actuator number i
a	axial induction factor
L, B, H	length (par. to X_1), breadth(par. to X_2), height(par. to X_3) of the overall domain
$\mathbf{M}^{-1}$	preconditioner
$P(\mathbf{X}, t)$	total pressure
$\overline{P}(\mathbf{X})$	mean pressure
$p'(\mathbf{X}, t)$	turbulent component of pressure
p, q, r	$-1/\triangle X_1^2, -1/\triangle X_2^2, -1/\triangle X_3^2$
s	$-2(p+q+r)$
t	time
$\overline{U}_\infty$	free stream velocity
$V_j(\mathbf{X}, t)$	total wind velocity component in direction, j
$\overline{V}_j(\mathbf{X})$	mean of the wind velocity vector
$v_j'(\mathbf{X}, t)$	turbulent component of wind velocity
$\tilde{v}_j$	pseudo-velocity
$\mathbf{X}$	Eulerian co-ordinate
$\triangle X_j$	none to node distance in direction, j
α	under-relaxation factor
ρ	density of air ($1.225 kg/m^3$)
ν	kinematic viscosity of air ($1.5 \times 10^{-5} m^2/s$)
$\boldsymbol{\omega}$	vorticity
φ	scalar potential
ϕ	dependent variable of a PDE
Ω	physical domain
Γ	interface between two domains
$\breve{\Theta}$	any quantity Θ non-dimensionalized w.r.t. some scale
$\boldsymbol{\mu}$	vector of means of n random variables
$\boldsymbol{\Sigma}$	variance-covariance matrix for n random variables

Chapter 1

Introduction

With the increasing human population, an ever increasing demand for energy, depletion in non-renewable energy sources like coal, petroleum, etc. and the pollution caused by these have led to increasing concerns worldwide. Rising global temperatures because of the pollutants and green house gases like carbon dioxide has resulted in melting of polar ice caps. National Oceanic and Atmospheric Administration (NOAA) published a report (Lindsey and Dahlman, 2021) which points out that the temperature of earth has been increasing at a rate of 0.08°C per decade since 1880. However, for the past 40 years, the rate of increase has almost doubled to 0.18°C. The year 2020 was recorded as the second warmest year as per data on NOAA record. Thus, over the past few decades, researchers have already started exploring renewable sources of energy, like solar, wind, hydro, tidal, etc. and understand them in greater depth. In other words, renewable sources are the future for clean, green and pollution-free energy and a way ahead for a sustainable future.

The importance of climate change and the increase in awareness worldwide can even be understood from the fact that United Nations (UN) in 2015 included 'Climate Action' (SDG-13) as part of its Sustainable Development Goals (SDG) and targets have been set for the member nations to work upon.

Ireland, as a member nation of UN has already started working on these targets which is clearly reflected in Sustainable Energy Authority of Ireland (SEAI) reports. As per *Energy in Ireland*, 2018 report published by SEAI, Ireland has been able to achieve about one-third of its energy production from renewable sources by

2017. This has resulted in a decrease of dependency on coal by about 21.2% and peat by about 6.4%. At the same time, energy production from renewable sources have increased by almost 19%. The carbon dioxide emission per kWh of energy is at an all time low of 437 g. In 2017, there has been a record 532 MW of wind energy installations and by end of 2020, Ireland had a total installation of 4.3 GW in wind energy sector, an increase in 180 MW compared to 2019 (*Energy in Ireland*, 2020 report). All this has resulted in saving €278 million worth of fossil fuel imports. From these statistics, it can be clearly seen that investment in wind energy sector is showing a phenomenal progress. This calls for a more detailed study of wind turbines and wind farms so that a better optimization of the energy output can be achieved. Hence, the present research focuses primarily on wind energy and wind farms in particular.

To date, almost all of the Irish wind farms are onshore. Setting up a wind farm requires considerable land area and land acquisitions are often faced with legal hurdles. Countries like China, Denmark, Germany, Netherlands, UK already have several commissioned offshore wind farms and many more are proposed. NSEnergy, 2021 states that as of 2020, top five countries of the world with the largest installed wind power capacity include China (288.32 GW), US (122.32 GW), Germany (62.85 GW), India (38.63 GW), Spain (27.24 GW). This progress in the worldwide scenario in setting up wind farms highlights their gaining popularity and at the same time satisfying the UN's SDG-13. As a result, researchers worldwide are putting in their best efforts to optimize the power production from the wind farms not only from the perspective of the aerodynamics of wind farms but also from the perspective of the controllers of the wind turbines and the design of the wind turbine itself.

Apart from the conventional Horizontal Axis Wind Turbine (HAWT) and Vertical Axis Wind Turbine (VAWT), various companies and research institutes are investing a lot into creating some innovative designs for wind turbines to cater to the small scale needs as well. A report by Gillespie, 2016 illustrates some such works, few of which can be seen in Figure 1.1 and Figure 1.2. Each wind tree (Figure 1.1) has the capacity to light 71 parking spaces which is equivalent to the energy requirement of an American Home for 4 months. Figure 1.2 shows a VAWT on top of of the decorative lights which is built and maintained by UGE with each turbine capable of producing 1 kWh to 1.5 kWh of energy. One can imagine that implementing renewable energy sources like wind to power such small scale applications can have

Figure 1.1: A wind tree installed at COP21 climate talks in Paris. (Photo Courtesy: New Wind)

Figure 1.2: Highway outside El Paso International Airport, Texas (Photo Courtesy: Vicki Scuri)

a considerable positive impact on climate change without compromising the energy demand.

Apart from these, concept of wind turbines integrated into a buildings architecture has also been put forward to meet the energy requirement of the buildings. This can be seen in some of the modern buildings like Strata SE1 in London, United Kingdom (Figure 1.3) and in Bahrain's World Trade Centre (Figure 1.4). However, such concepts are still work in progress and power generation from these integrated turbines are still not considered to be optimum.

1.1 Scope of work

All fluid flow problems are governed by the Navier-Stokes or Euler equations (refer Chapter 3) which are non-linear in nature. Numerically trying to solve these non-linear equations is not only computationally expensive but also may not lead to convergent results in some cases (e.g. for arbitrary initial and boundary conditions). This is particularly true in the situation when the domain is a large, wind field like the case where the wind farm as a whole is the main focus. The code of practice for wind turbines, IEC FDIS 61400-1, recommends the use of empirical stochastic turbulence models based on cross-spectral density functions or based on Rapid Distortion Theory (RDT) to model the turbulence. However, using empirical methods does not give the real picture. Since, RDT is a more realistic approach (as it is based on Navier-Stokes Equation), an effort will be made to look at this

Figure 1.3: Strata SE1, London (Photo Courtesy: treehugger.com; Frerk Meyer / Flickr / CC BY-SA 2.0)

Figure 1.4: Bahrain World Trade Centre (Photo Courtesy: buildinggreen.com; Ole Sangill, Norwin A/S)

linearization method closely in the present work.

It is worth noting that the effect of wake aerodynamics on the wind turbines installed farther downstream is yet not fully understood. The importance of wake aerodynamics can be understood by looking at Denmark's "Horns Rev 1" wind farm Stromsta, 2010 (refer Figure 1.5). It can be clearly seen from the figure that the wake aerodynamics of one wind turbine influences the other wind turbines installed downstream. This highlights the fact that the performance of a wind turbine depends not only on its own design but also on the the aerodynamic wake

Figure 1.5: Horns Rev 1. (Photo Courtesy: Recharge)

of the other wind turbines installed in its vicinity. This would mean that the wind velocity with which a turbine is working is going to be affected by the turbines installed upstream. Hence, the present research work will focus on not only the generation of spatial variation of turbulence in a large domain as mentioned but also on the Fluid-structure interaction (FSI) of the laminar flow wake with vorticity in a wind farm.

FSI models can be quite conveniently developed using commercial or open source softwares. However, because of the complications involved in aspects like modelling the geometry of the blades of the wind turbine, higher computational time and memory required to run a full scale CFD model, it is not always an optimal approach. A typical example could be a scenario when just a preliminary result is required to study the impact of different arrangements of wind turbines in a wind farm at a particular site on overall power output. Standard wind farm models do this job but they lack proper incorporation of FSI effects. An in-between approach is therefore required such that FSI effects can be conveniently incorporated alongside these standard wind farm models. This is where special FSI methodologies like Immersed Boundary (IB) Method, Immersed Interface Method (IIM), or their variants come into picture which can considerably reduce the computational resources required. These FSI methodologies have been known for the past few decades but their application to the field of wind farms have been extremely limited. However, these methodologies are based on Finite Differencing Method unlike commercial or open source softwares which are built on Finite Volume Method. This motivated the development of a computer program to incorporate one such FSI methodology (Decomposed Immersed Interface Method). It would be evident from the subsequent chapters that this method used in this research was originally proposed based on Gauss-Seidel iteration. However, when the domain is as large as that of a wind farm, other iterative algorithms like Biconjugate gradient (BiCG) or their variants perform much better. But, the way BiCG works, the algorithm could not be applied in a straightforward way to incorporate DIIM. Hence, in this work a modification has been proposed to the algorithm such that DIIM can be conveniently incorporated.

Dealing with such a large domain extending over kilometers requires extensive use of parallel numerical algorithms with multi-processor computation to achieve computational efficiency both in terms of memory and time. **Python3** has been chosen as the preferred language for the numerical simulations as it is free, has

object-oriented programming features and extremely powerful with a wide range of libraries to deal with any kind of data analysis. Thus, the simulation codes can be easily executed by anyone without any extra cost. Python libraries **numpy** (which itself is based on optimised linear algebra libraries like **Basic Linear Algebra Subprograms (BLAS) / Linear Algebra Package (LAPACK)**) and **mpi4py** (based on optimised C library for multi-processor simulation using **Message Passing Interface (MPI)**) coupled with parallel numerical techniques have been used extensively to achieve efficiency in simulations both in terms of memory and time.

1.1.1 Research Gaps and Challenges

Having realised the broad scope of the work, it is well understood that many challenges are posed and expected in the present work, few of which are mentioned below.

- Simulation of a domain extending over kilometers is a herculean task in itself demanding extensive computational effort.
- The boundary conditions are not well known and understood.
- Simulation of laminar flow in a domain with vorticity.
- The turbulent structure needs to be investigated in a mechanistic way.
- Numerically resolving the scales of turbulence in the context of wind farm as a whole.
- Integrating the turbulence and the wake effects in a wind farm.

1.1.2 Aims and Objectives

In the light of the previously mentioned scope of work and the challenges, this work will primarily aim at:-

1. Developing an understanding of iterative algorithms in a parallel setting and multi-core simulations including appropriate modifications to these algorithms to suit the needs of the current work.
2. Investigating the behaviour of a steady potential flow model to model the aerodynamics in the context of wind farm.
3. Propose an equation in two-dimension (2D) to substitute the potential flow

model and study the behaviour of laminar flow under constant vorticity.

4. Implementing FSI coupled with a new model for wind turbines modelled as actuator discs with appropriate boundary conditions to achieve a greater computational efficiency.
5. Proposing a modified version of pre-conditioned BiCGStab algorithm to accommodate DIIM.
6. Developing a RDT model to study the turbulence structure in Atmospheric Boundary Layer (ABL) using Gaussian closure.

1.1.3 Research Methodology

Evidently, in order to achieve the above objectives, study is required across a wide range of fields. Bearing this in mind, a step-by-step approach is needed. The task at hand is therefore broken down into the following series of steps:-

- Carry out an extensive literature review covering wind turbines and wind farms in order to understand the way different wind farm models work analytically and how the existing wind farm softwares deal with wind farm models including modelling the wind turbines.
- Understand fluid dynamics and behaviour of Partial Differential Equations (PDEs), both from analytical and numerical point of view, particularly for domains with a large number of nodes and how best these concepts can be applied to wind farms.
- Learn the concepts of parallel numerical algorithms, iterative algorithms and multi-core simulations by attending the relevant coursework.
- Investigate in detail the behaviour of steady potential flow model in 2D and propose a more general model in case the potential flow model does not work as expected.
- Modelling the discontinuity introduced into the computational domain by the wind turbines modelled as actuator discs using the concepts of FSI and couple it with the new proposed model and validate the performance of the model with respect to the existing analytical wind farm models.
- Carry out a 3D analysis of a wind farm with the assumption that a wind turbine is affected by the wake of another turbine which is directly along the direction of the wind flow and the wind turbines installed elsewhere in

the wind farm do not influence the aerodynamics of the wind turbines under consideration.

- Develop a RDT model for turbulence and examine the consistency of the results of the numerical simulation with respect to the assumptions made in developing the PDE.

1.1.4 Thesis Outline

Bearing in mind the aims and objectives presented above, this thesis is organised into five main chapters. To begin with in Chapter 2, an exhaustive literature review is carried out covering three major areas i.e. wind farms, fluid dynamics and parallel numerical algorithms.

In Chapter 3, the numerical analysis methods are discussed in detail. Thus, this chapter begins with an introduction to the fluid flow equations followed by the currently known discretization schemes which eventually lead to formulation of a linear system of equations. Subsequently, the data storage strategies and the way iterative algorithms use parallelism and how the same has been implemented in the present research work is discussed.

Chapter 4 begins with a discussion on the application of actuator disc theory in a wind turbine. The potential flow model has been analysed in detail from the perspective of current research work and improvements to the flow model has been suggested to take care of not only additional and more realistic cases but at the same time incorporate the effect of the wind turbine in the flow model in a more physically acceptable way.

Based on the discussion carried out in Chapter 4, in Chapter 5 a special FSI strategy has been introduced. The method is then used to model the FSI effects due to the presence of blades of the wind turbine on the fluid flow.

Finally, in Chapter 6, to generate the turbulence a RDT based model is proposed which uses Gaussian closure. Simulations are run for a large number of cases and the results have been presented. The chapter concludes with a detailed discussion on the proposed approach.

The thesis ends with Chapter 7 where the major findings of the work are

presented followed by a discussion on the possible future works.

Chapter 2

Literature review

Study of wind farms and aerodynamic simulation of the wind field requires knowledge of not only the atmospheric boundary layer and understanding the concepts of wind farm aerodynamics but it also involves studying numerical strategies for simulation and existing wind farm simulation tools and models and how they work. Since aerodynamics is one key aspect, understanding the wind farms from the perspective of fluid dynamics is essential. The wind turbines interact continuously with the surrounding air which is why Fluid-structure interaction (FSI) is yet another aspect which comes into picture. Now, if only a single wind turbine is to be studied, any normal sequential algorithm would have been enough. However, since wind farms are in focus, parallel numerical algorithms is yet another area that needs to be understood. As such, the literature review is organized into the following parts.

1. Wind energy
2. Fluid dynamics
3. Parallel numerical algorithms

2.1 Wind energy

Though work on wind turbines to generate electricity was done as early as the 19^{th} century, it was not until the late 1970s that the governments of countries like the US, UK, Germany and Sweden started funding wind energy projects (Burton et al., 2011). This section begins by discussing the influence of Atmospheric Boundary

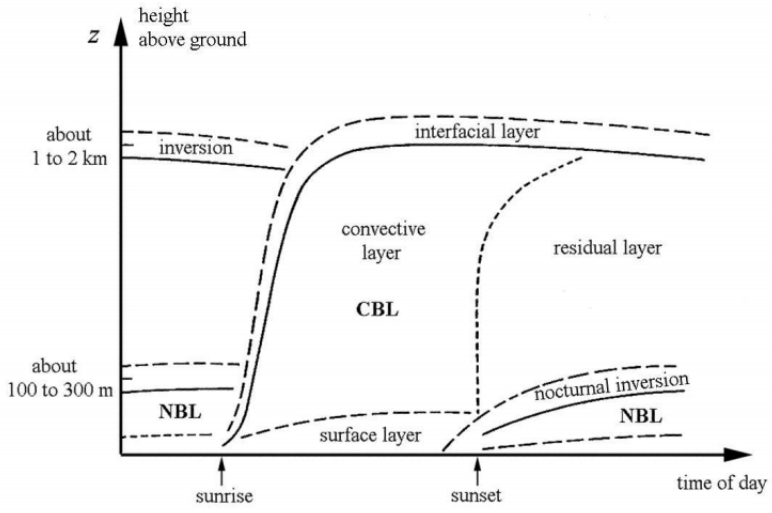


Figure 2.1: Typical evolution of the Atmospheric Boundary Layer over the course of a day over land and under clear skies. At sunrise, heating from below sets to a convective boundary layer, while at sunset heat loss to space terminates convection and creates a thin Nocturnal Boundary Layer (Garratt, 1992)

Layer (ABL) on wind turbines followed by how Computational Fluid Dynamics (CFD) has influenced wind turbine analysis over the past decades. And, finally, the existing wind farm models proposed by different researchers is presented.

2.1.1 Atmospheric Boundary Layer

A recent report by Weaver, 2017 shows that Germany has built the tallest onshore wind turbine with its hub at 178 m and the total height to the tip of the blade at 246.5 m. This wind farm is located in Gaildorf, Germany. According to Mines, 2018, General Electric is working on 260 m high offshore turbine which is still higher. This means the wind profile in the Atmospheric Boundary Layer (ABL) is going to have more prominent effect on the wind turbines. A typical figure of variation in ABL is given in Figure 2.1.

It can be seen from the figure that the variation in the wind velocity in ABL layer needs to be considered properly in wind energy calculations. One can notice the change in the boundary layer development at different times of the day. This thesis however, will not be focussed on the effect this kind of diurnal change will have on wind farms. Rather a typical mean wind profile will be considered for the purpose of simulations.

In the context of wind farms, the first and foremost information one needs is a mean wind profile prevalent in ABL. Over the years, meteorologists have proposed several models based on experimental data. For example, Wilson and Flesch, 2004 made an effort to provide a mean wind profile for the lower atmosphere such that a 'best fit' model can be obtained from the data available at hand. Peña et al., 2008 used measurements of the marine wind speed profile at a site located 18 km from the west coast of Denmark. They compared expressions for marine ABL with cup observations and concluded that wind speed gets over-predicted over 30 $m - 40$ m above Mean Sea Level (MSL) if the effect of boundary layer is ignored. Others like Richards and Norris, 2019 have discussed the problems associated with Horizontally Homogeneous Atmospheric Boundary Layer model in Computational Wind Engineering and proposed a series of steps for determining the velocity profile. They referred to the model proposed by Deaves and Harris, 1978 which uses a modified version of the Asymptotic Similarity Theory for the resultant mean velocity profile. Since their discussion is based from the perspective of computational wind engineering, the mean velocity profile proposed by them is implemented in the current research work. Another reason for choosing this velocity profile is that it is not dependent on any field observations and has been derived based on theoretical understanding of the ABL.

$$\overline{V}_1 = \frac{u_\tau}{\kappa'}\left(\ln\left(\frac{X_3}{z_0}\right) + 5.75\left(\frac{X_3}{h}\right) - 1.875\left(\frac{X_3}{h}\right)^2 - \frac{4}{3}\left(\frac{X_3}{h}\right)^3 + \frac{1}{4}\left(\frac{X_3}{h}\right)^4\right) \tag{2.1}$$

where,
$\overline{V}_1$ = mean wind velocity parallel to earth's surface,
κ' = von Karman's constant ≈ 0.4,
ϕ = latitude of interest = $53.3498°$ for Dublin,
Ω = rate of rotation of earth = $7.292 \times 10^{-5} rad/s$,
C_D = drag coefficient taken approximately as 0.0014 for sea (Stull, 2016),
u_τ = friction velocity = $M_{10}\sqrt{C_D} = M_{10}\sqrt{0.0014}$,
f = coriolis parameter = $2\Omega \sin(\phi) = 1.17 \times 10^{-4} rad/s$ (for Dublin),
h = gradient height = top of ABL = $u_\tau/6f$,
M_{10} = wind speed at height of 10 m (considered 10 m/s),
X_3 = height above MSL,
z_0 = ground roughness length (Davenport-Wieringa roughness length = 0.0002 m for sea (Stull, 2016)).
It is to be noted that the parameters are chosen assuming offshore wind farms so that the effect of terrain on the mean velocity profile can be neglected. Under

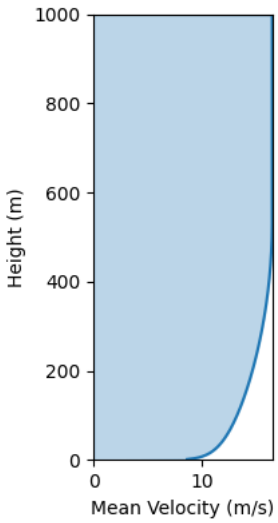


Figure 2.2: Typical Mean Velocity Profile

certain conditions, the log term in eq.2.1 could be dominant compared to the other terms. The authors have recommended to retain at least the first two terms upto 200 m instead of considering the log term only. It is to be noted that after the mean velocity reaches a maximum value ($\overline{V}_{1,max}$) at a height ($X_{3,max}$), the velocity is assumed to become constant and the flow is fully developed. $X_{3,max}$ can be obtained by differentiating eq.2.1 w.r.t. X_3,

$$\begin{aligned} \frac{\mathrm{d}\overline{V}_1}{\mathrm{d}X_3} &= \frac{u_\tau}{\kappa'}\left(\frac{1}{X_3} + \frac{5.75}{h} - \frac{3.750}{h^2}X_3 - \frac{4}{h^3}X_3^2 + \frac{X_3^3}{h^4}\right) \\ \Rightarrow 0 &= \left(1 + 5.75\frac{X_3}{h} - 3.75\frac{X_3^2}{h^2} - 4\frac{X_3^3}{h^3} + \frac{X_3^4}{h^4}\right) \\ \Rightarrow X_3 &= h(= X_{3,max}). \end{aligned} \tag{2.2}$$

Thus, $\overline{V}_{1,max} = (u_\tau/\kappa')(ln(h/z_0) + 67/24)$. A typical mean velocity profile obtained from the above equation and assuming $M_{10} = 10\ m/s$ is shown in Figure 2.2.

2.1.2 Wind Turbines

The concept of modelling wind turbine as an actuator disc has been in use over several decades now. Froude, 1889 proposed the actuator disc theory which is still the most simple and widely accepted theory in the field of wind energy. The theory works on the concept of an energy extracting actuator submerged in a fluid. Subsequently, other theories like the rotor disc theory and blade element momentum

theory (Lock et al., 1925) were proposed which take into account the effects of the turbine rotation as well. Glauert, 1926 proposed modifications to these theoretical methods to incorporate corrections like tip loss factor. With the advances made in CFD, from early 1990s researchers have started looking into energy extracting actuator disc from the perspective of numerical simulations. But, before looking into the models and the studies carried out for the wind farms as a whole, a look into the work done on a single wind turbine is essential. For instance, Voutsinas and Rados, 1995 worked on yawed operation of wind turbines. They carried out full 3D non-linear aero-elastic numerical investigation of Horizontal Axis Wind Turbine (HAWT) during yawed operation and used GENUVP code which is based on Helmholtz decomposition. They have used a numerical strategy based on time marching and the coupling of free-wake vortex particle model coupled with a 3D beam type structural model. Madsen, 1996 used Finite Element Method (FEM) to model the field surrounding a single wind turbine and compare the results with Blade Element Momentum (BEM) Theory. It was observed that tangential induced velocity varies greatly between CFD analysis and BEM results. Also, the turbulence mixing increases axial velocity in the wake as compared to steady value obtained by BEM and derived a linear analytical solution for actuator disc flow. Apart from these many other interesting works are carried out, few of which are discussed in Section 2.2.1.

2.1.3 Wind Farms

If one considers the spatial domain of a wind farm as a whole, the time and memory required for computation on such domains are extensive, which is why such studies require extensive use of supercomputers or clusters of several thousands of cores. Thus, High Performance Computing (HPC) comes into picture which is subsequently discussed in Section 2.3. It is a well known fact that the wake structure of the first installed wind turbine in a wind farm influences the performance of the wind turbines located further downstream. For this reason, apart from studying a single wind turbine, research on the wind farm as a whole is also of importance. Several models on performance of the wind turbines located downstream are based mostly on empirical models.

Churchfield, 2013 from NREL presented a review of wind turbine wake models where the models are categorised into four types depending on the purpose for

which they are to be used. These four types include models predicting power production and Annual Energy Production (AEP), loads, control strategies and basic physics. Out of these, the first category deals with steady state which is explored in detail in the current work. Author has highlighted the fact that these models are in no way each other's competitors and higher fidelity models can serve to improve the more simplistic lower fidelity or empirical models. In this thesis, this is what has been demonstrated in the later chapters where Jensen's model has been used alongside the proposed FSI methodology.

One of the earliest work is by Jensen, 1983 who proposed a top hat model for wake expansion and velocity deficit based on the law of conservation of momentum. Though experimental results suggested an entrainment constant of 0.07, Jensen suggested a value of 0.1 as the assumption that the velocity deficit of 1/3 times the free stream velocity just behind the actuator is fairly uncertain. Based on this model, Jensen presented a wake structure and energy outputs for 10 back-to-back wind turbines as well as turbines placed in a circle. Katic et al., 1986 also implemented Jensen's model to calculate the output from wind farms. Ainslie, 1988 used Eddy viscosity model with zero circumferential velocities and flowfield stationary with time. In this approach, the Navier-Stokes' equations were replaced with the thin shear layer approximation without any viscous term. The numerical solution for this model, though simple, showed a good resemblance with the experimental results but the behaviour in the near wake region was too complicated to handle.

Other models like Infinite Wind Farm Boundary Layer (IWFBL) model proposed by Frandsen, 1992 or another IWFBL model by Emeis and Frandsen, 1993 based on mixing-length theory exist, but some of their parameters are indeed difficult to determine. More recently, Frandsen et al., 2006 proposed a model for an array of wind turbines by splitting the wake into three regimes, the first is turbines' exposure to multiple wake flow linking the flow deficit with wake expansion, the next corresponds to wake expansion in the vertical direction only after wakes have merged and finally, flow in balance with boundary layer for an infinitely large wind farm.

It has been shown by Rathman et al., 2006 that in the case of large offshore wind farms extending over more than 5 rows, wake losses are significantly higher when compared to that of standard wind farm models. Schlez and Neubert, 2009 therefore presented some corrections to be applied to the wind farm models for

simplified wake modelling. This includes proposing a correction for ABL, using wind turbine density (i.e. number of wind turbines present in a particular sector) to judge if the correction is required and applying the correction only if row-to-row distance is closer than a threshold and downstream recovery of the wake. This is also illustrated in the work by Johnson et al., 2009 and Brower and Robinson, 2012 and is referred to as deep array correction. Larsen et al., 2007 proposed a wake meandering model for a single wind turbine. Though it was proposed for a single wind turbine, authors were hopeful of its applicability to wake interactions as well. They used a stochastic approach in their model because of the influence of large-scale vertical and lateral turbulence on wake transportation in ABL. The model showed a good resemblance with actual measurements. The authors claimed to have achieved a single unified wake model for turbine power and load unlike most other models. Subsequently, Larsen et al., 2013 applied this model to the offshore Egmond aan Zee wind farm in Netherlands which has a total of 36 V90-3MW wind turbines installed. Loads were compared by the authors for two different wind directions. Since the authors considered a turbine deep inside the array with influence from multiple wakes, they proposed a method to deal with multiple wake interaction using eddy viscosity model. The results have shown good resemblance with the field observations.

Bastankhah and Porté-Agel, 2014 pointed out that though commercial softwares like WindPRO (Thøgersen, 2005), WAsP (Barthelmie et al., 2006), WindSIM (Crasto et al., 2012) and few others use Jensen's model extensively, the model is not realistic and in fact, it is the mass conservation instead of conservation of momentum that the model implements. Hence, they proposed a Gaussian wake model and validated their results with Large Eddy Simulation (LES) and wind tunnel experiments of a miniature wind turbine as well as LES simulation of wake of a Vestas V80-2MW turbine. Ott and Nielsen, 2014 published a report on a linearized CFD model for wake called Fuga for offshore wind farms. The model is based on linearized RANS with simple turbulence closure. The model even includes the atmospheric stability and wake meandering effects. Faster solution is arrived at by using spectral analysis. Tian et al., 2015 developed a two-dimensional wake model to predict the velocity and turbulence distribution in the wake of a wind turbine. The approach is based on Jensen's model with the improvement that it incorporates a a cosine shape function in the cross-wind direction for determining the distribution of the wake deficit. The authors then compared the results of the model with wind tunnel

experiments, field data, LES and $k-\omega$ model and it was found that the model gave a good approximation of the far wake region. Sun and Yang, 2018 presented an analytical three-dimensional wake model by considering the wind speed variation along height. This model considers a Gaussian wake deficit. The authors carried out the validation of their proposed model by comparing their results with published wind tunnel experiments. Subsequently, the authors used their model for numerical investigation of the average wind speed of a single wind turbine (Sun and Yang, 2020) and claimed that the predictions from the model are practically acceptable.

Blondel and Cathelain, 2020 also presented a wake model but based on super Gaussian shape function. Other models like Gauss curl hybrid model proposed by King et al., 2021 incorporates the effects of wake steering in large wind farms. Few other models have also been proposed over time like those by Martinez-Tossas et al., 2019 who proposed a curled wake model for wind turbine under yaw condition and Nygaard et al., 2020 who presented two new models based on the Park model by including turbulence explicitly and another which considers blockage contribution from individual wind turbine to model the entire wind farm and hence, compute its energy output.

Physically, the presence of wind turbines in the computational domain i.e. the wind field complicates the scenario since FSI study is required to arrive at more realistic results for the wind field in the presence of wind turbines. It is to be noted that the basic purpose of these existing wind farm models is to arrive at a reasonable wake structure without going into the complexities of CFD which requires extensive computational resources and time. As a result, these models lack the important aspect of Fluid-structure interaction of the turbines with the wind which influences the wake structure as well. Having said that FSI methods need to be looked at in order to gain an understanding of their advantages over the more complicated computational models for wind farms. Further, as highlighted earlier, sequential algorithms are of little help for these large domains because of their high computational cost. Instead, these can be handled in a more time- and memory-efficient way by exploiting Parallel Numerical Algorithms (PNA).

2.2 Fluid dynamics

Fluid flow can be broadly categorised into laminar and turbulent flows. Turbulent flows are by far much more complicated and difficult to handle. Even solutions to simple flow problems can be non-trivial and might be difficult to model. Different commercial softwares are available like ANSYS, ABAQUS, etc. which can model flows like potential flow or the more common and perhaps the most vital Navier-Stokes equation. This section begins with a discussion on laminar flow and turbulence and rapid distortion theory followed by a brief review of the fluid structure interaction methodologies and the existing numerical algorithms to resolve pressure and velocities for the convection-diffusion problems.

2.2.1 Laminar flow model and vorticity

Researchers have tried to implement the potential flow models for analysing the wind fields. Xu and Sankar, 2000a presented a hybrid methodology wherein the field around wind turbine is modelled using Navier-Stokes and the remaining flow field is modelled with potential flow. Based on this study, Xu and Sankar, 2000b further presented a study on effects of turbulence models and transition models, rotor performance and the effect non-axial flow has on power generation. Palmiter and Katz, 2010 evaluated a potential flow model for propeller and wind turbine design. They used a 3D potential flow based unsteady panel code to model the flow over rotating blades. Their work primarily focused on two bladed turbine. Based on the simulation, they made an effort to arrive at an optimal shape for the propeller blades. Shane, 2011 worked on 'Potential Flow Modelling for Wind Turbines'. The work uses LibAero, a C++ based potential flow solver developed by Lawton and Crawford, 2012. The model uses strategies like n-body problem of physics and fast multipole method (by Greengard and Rokhlin, 1987) and is based on Helmholtz equation (eigenvalue problem for Laplace operator; $\nabla^2\phi = -k^2 f$). LibAero is aimed towards wind turbine performance prediction and for use in a Multidisciplinary Design Optimization (MDO) tool. It is based on Weissinger Lifting Line approximation and uses vortex particles, filaments and quadrilateral sheets for wake discretization.

van Kuik and Lignarolo, 2016 worked on potential flow solutions for actuator discs. Their model is based on Euler and continuity equations and included wake

expansion and pressure variation across an annulus. Further, they proposed correction to axially induced velocity. They have stressed the fact that most existing models assume uniform axial flow. However, they illustrated in their work it is the absolute velocity $|\mathbf{V}|$ which is uniform rather than the axial velocity, which is non-uniform. Based on these potential flow solutions, van Kuik et al., 2015 compared actuator disc and Joukowsky rotor flows and explored the need for a tip correction. However, a laminar flow model with vorticity and wake interaction is something which has not been explored in detail in the case of wind farms and this is an area which has been explored in this thesis.

2.2.2 Turbulence and Rapid Distortion Theory (RDT)

Rapid Distortion Theory (RDT) is a linearized approach used to calculate rapidly changing turbulent flows subjected to a different kind of distortions like large scale velocity gradients or presence of body forces, etc. It is a method based on linear analysis for calculating Rapidly Changing Turbulent flows.

Batchelor and Proudman, 1954 developed RDT for calculating distortions in turbulence structure when fluid flow is subjected to large-scale straining motions. Pearson, 1959 demonstrated how RDT might be useful to get an insight into the turbulent structure especially for shear flows. The subsequent works by Deissler, 1968, Townsend, 1980, Jeandel et al., 1978 demonstrated with the aid of experimental results that RDT can be applied to shear flows subjected to slowly changing turbulence as well. Batchelor, 1982 later detailed out the theory in the book "The Theory of Homogeneous Turbulence". Hunt and Carruthers, 1990 categorized the problems of turbulence into 3 classes:-

- Class I: Closed domain and deterministic boundary conditions.
- Class II: Open domain and statistical boundary conditions which can be further subdivided into.
    1. No turbulence outside domain
    2. Turbulence outside domain exists but mean flow is significant.
    3. Turbulence outside domain exists but mean flow is not significant.
- Class III: Initial conditions and changing boundary conditions

They presented how the application of RDT has extended to inhomogeneous turbulent flows and some of its mathematical restrictions have been overcome. From the perspective of the wind farms, as the turbulence exists outside the domain but the mean flow is significant, the scenario can be categorized as the case of Class II, Type 2.

Lee et al., 1990 demonstrated how RDT can be used to calculate the second order moments and deduce the characteristics of the eddy structures. Though not widely applied to the field of wind energy, it has found application in various other fields. For example, Brereton and Mankbadi, 1993 had applied RDT to developed unsteady wall-bounded flow and Chougule et al., 2017 has modelled atmospheric turbulence using RDT. Ainslie and Scott, 1990 had presented a review of principal mechanisms proposed for the generation of noise in wind turbines at European Wind Energy Conference EWEC, 1989 where application of RDT has been highlighted. Farr and Hancock, 2014 carried out wind tunnel studies of flow upstream of the rotor and suggested that the flow stagnation nullifies the amplification implied by RDT. Recently, Graham, 2017 used RDT on a horizontal axis wind turbine rotor as in wind turbine or in tidal-stream turbine. The work is interesting particularly because in that approach the velocity spectrum and the variance were calculated without assumptions like restrictions on the size of the longitudinal length scale of turbulence approaching the rotor. Mann et al., 2018 studied the changes in turbulence as wind approaches the rotor where they discuss the limitations of RDT for a single wind turbine. It can be seen that limited literature on the application of RDT in the context of wind farms is available and as such, this area needs to be explored in greater detail. Thus, in the final part of this thesis, an effort has been made to understand and implement RDT.

2.2.3 Fluid-structure interaction (FSI)

The issue with the fluid flow problems is that they start getting more complicated once there is a body immersed in the fluid domain just like the wind turbines in this thesis. From a mathematical point of view, the presence of a solid body introduces discontinuity in the flow domain and this, in turn, gives rise to FSI problem wherein the physical parameters of the fluid like the velocity and pressure are greatly affected. From the perspective of the immersed body, like the blades of the wind turbine, their structural behaviour is also impacted. Since the behaviour of the structure itself is

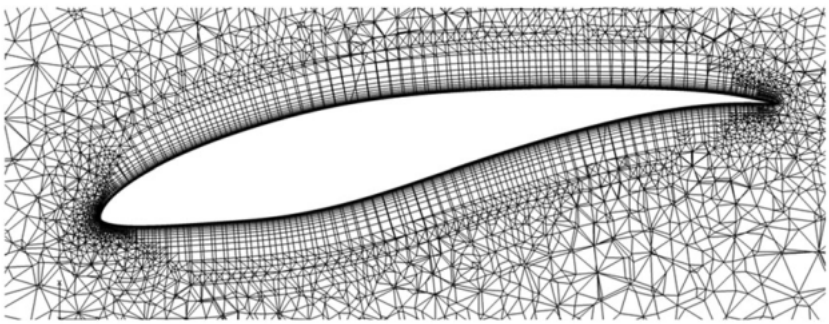


Figure 2.3: Unstructured mesh around a section of wind turbine blade, Mittal et al., 2016

not the main focus in this current thesis, the literature review carried out is from the point of view of the fluid only.

Commercial softwares or any of the standard solvers can take care of the fluid flow problems quite efficiently as long as there are no bodies immersed in the domain. With these solvers, one way is to model a mesh all around the structure as shown later in Figure 2.3. But, as can be understood and as discussed later in Section 2.3, these models are time consuming and their computational cost is significant. For this reason, Mittal and Iaccarino, 2005 have discussed the applications of special FSI approaches to various boundary conditions, and how such methods are effective in eliminating complex grids and lead to computationally efficient algorithms.

The phenomenon of FSI prompted researchers to devise different analytical and numerical approaches by taking into account both the physical nature of the fluid as well as the mathematical equations governing the fluid flow. Li, 2003 and Kumar and Joshi, 2012 have discussed some of these different approaches available to deal with these discontinuities in case of flows governed by the general elliptic Partial Differential Equation (PDE) (e.g. flows like potential flow problem in the context of fluid dynamics). To handle such discontinuities, methods like smoothing method for discontinuous coefficient where known smoothed functions are used and the method of harmonic averaging have also been implemented by researchers to handle the discontinuities. However, these latter two methods become more difficult to implement particularly in 3D. Shortcomings in these classical approaches prompted researchers to explore other different numerical strategies.

Peskin, 1977 proposed the Immersed Boundary Method (IB) to model cases

of fluid mechanics applied to biology problems. Subsequently, Peskin, 1981 presented the method in details in "Lectures on Mathematical Aspects of Physiology". The method was used to model blood flow in heart but later on the method was implemented in more general fluid flow applications. Works by Beyer, 1992, Arthurs et al., 1998, Bottino, 1998 are some interesting application of IB to fluid flow problems. Researchers have also parallelized IB which is reflected in the work of McQueen and Peskin, 1997. However, IB is only first order accurate and hence is not enough for many fluid dynamics problems. This prompted researchers to further explore the method and look for alternatives in order to make it second order accurate.

One significant contribution to devise a second order approach came from the work done by Li, 1994 who introduced Immersed Interface Method (IIM). The method was developed to solve PDEs numerically by using second order accurate Finite Difference Equation (FDE). Overtime several other variants of IIM have been developed by researchers. For example, Wiegmann and Bube, 2000 used an explicit strategy wherein instead of focussing on finding new coefficients for FDE, focus was mainly on the jumps of the dependent variable and its derivatives at the interface. Li and Lai, 2001 introduced the application of IIM to the incompressible Navier-Stokes Equations with singular forces. Berthelsen, 2004 introduced Decomposed Immersed Interface Method (DIIM) for solving two-dimensional variable coefficients elliptic equations on cartesian grid by introducing a correction term to the standard 5 point stencil of FDEs. In this thesis, this approach has been used, the reason for which is discussed in the subsequent chapters.

Meanwhile improvements were made to IB as well in order to increase its accuracy which is reflected in the works of Cortez and Minion, 2000 and Cortez et al., 2005. Subsequently Li and Ito, 2006 demonstrated the application of IIM to a wide range of PDEs beginning from one, two and three dimensional elliptic interface PDEs to Stokes and Navier-Stokes problems as well as parabolic interface PDEs. Some applications in the context of FEM and fourth order IIM have also been demonstrated which makes this method a suitable option for FSI problems. They also point out that though IB is simple and robust and research is undertaken on improving its accuracy, the method lacks in the proof of complete convergence. It is to be noted that IIM works by modifying the coefficient matrix of a linear system of equations in order to model FSI and it was found to perform quite well for different problems which can be seen in the works of Xu, 2008, Tan et al., 2009a and Tan

et al., 2009b. From the application point of view of FSI in wind farms, a recent effort has been made by Korobenko et al., 2017 to study FSI for two back-to-back wind turbines.

2.2.4 Fluid flow algorithms

One of the challenges in any fluid flow problem is calculating the pressure. Equations of continuity, equations for momentum and energy are known but any explicit equation related to pressure is not known. In fact, the pressure field in any fluid is coupled to the velocity field through the momentum equations and is one of major contributing factors towards the source term in the momentum equations. Further the velocity components in each direction of the momentum equations are coupled to each other. To overcome this difficulty Patankar and Spalding, 1972 proposed SIMPLE (Semi-Implicit Pressure Linked Equations) algorithm, an iterative algorithm wherein the discretized momentum equation containing the pressure term is substituted into the equation of continuity in order to obtain an equation for pressure. The velocity and pressure fields are then alternately solved. Correction factors are computed at each iteration step to correct the pressure and velocity fields until convergence is achieved. The issue with SIMPLE is that the pressure correction factor can correct velocities reasonably well but does not do a good job with correcting the pressure. Hence, Patankar, 1980 proposed SIMPLER (SIMPLE Revised) to overcome this issue wherein the pressure correction is not required at all. Instead, the pressure is computed from the guessed velocity fields initially and the velocity field generated at the end of each iteration. In this work, SIMPLER has been used to generate the velocity field. A detailed discussion on SIMPLER and how it has been applied is presented in Chapter 6.

Van Doormaal and Raithby, 1984 proposed yet another variant of SIMPLE called as SIMPLEC (SIMPLE Conistent) wherein the velocity correction equations omit lesser significant terms. Apart from these variants of SIMPLE, Issa, 1986 proposed another approach called PISO (Pressure Implicit with Splitting of Operators) which uses "one predictor step and two corrector steps". Issa et al., 1986 have shown that though PISO requires considerably more computational effort compared to SIMPLE, it is much more efficient and fast. Jang et al., 1986 and Versteeg and Malalasekera, 2007 in their work compared the performance of these algorithms. They found that SIMPLE is relatively straightforward but its other variants, though

involve more calculation per iteration, their approach reduces the overall computation time compared to SIMPLE. Further, SIMPLEC and PISO have been found to be as much as efficient as SIMPLER in certain cases but it has not yet been ascertained as to which algorithm is the most efficient. As can be understood from these discussions that CFD requires an understanding of various numerical strategies. However, as long as the domain to be dealt with is small, normal numerical strategies might just be enough. But, when one deals with a large domain (e.g. wind farms), another special area in terms of numerical strategies i.e. 'Parallel Numerical Algorithms' needs to be explored.

2.3 Parallel numerical algorithms

An algorithm can be considered efficient if it is able to achieve the end result without the requirement of excessive memory and without incurring heavy computation time. This calls for numerical algorithms to be developed in such a way that multiple steps of an iteration can be done at the same instance of time. This is where Parallel Numerical Algorithms (PNA) and High Performance Computing (HPC) come into picture. PNA had been in use for quite a long time now but their use has been limited in the field of wind farm simulations until late 1990s. These days more and more researchers are exploring the possibilities of implementing PNA for large scale wind farm simulations. With the advancement in computing technology and availibility of multiple core CPUs and interconnected clusters, PNA have reached a whole new level where multiple tasks of the same simulations are done in different CPU cores (or processors) at the same time instance. This in particular is useful for simulations of domains as large as wind farms where a great amount of data needs to be processed. Thus, discussion of PNA is incomplete without MPI and GPGPU, two vital tools which provide seamless interface to multi-core simulations. This section therefore looks into the work done by researchers which use these numerical and computational tools as well as discusses the existing wind farm simulation softwares which implement such algorithms.

2.3.1 Message Passing Interface (MPI)

Before 1990's, parallel codes were difficult to develop because of the widely varying computer architecture (Gropp et al., 1999). As a result, each research group

worldwide developed their own version of such parallel libraries in order to consider cross-platform portability across various computer architectures. Some of these projects include PVM (Sunderam, 1990), PICL (Grant and Dickens, 1993) , PARMACS (Calkin et al., 1994), to name a few. The most common applications of such parallel codes were in the domain of science and research. Hence, standardizing parallel computation was the need of the hour. In 1992, a supercomputing conference was held in Williamsberg Virginia (Walker, 1992) wherein a standard to develop such parallel codes was framed and MPI came into existence. Consequently, after receiving extensive reviews from researchers, the final standardized MPI version was released in a report by Walker and Dongarra, 1995.

MPI is a library originally written in two versions, one for C and one for FORTRAN. This library allows users to run the same program across multiple processes parallely. It also allows distributed computing by distributing the data across multiple computers (often referred to as nodes) or same computer with multiple cores. Subsequently, using these MPI standards, different research groups and companies developed their own MPI packages. MPICH by Argonne National Laboratory, US Department of Energy is one of the early such implementations of MPI. Other MPI projects include Open MPI, Intel MPI, Windows MPI to name a few. The MPI projects of chip manufacturers like Intel are designed to have better hardware compatibility of MPI with computers using Intel processors thereby increasing the performance of HPC. The work carried out in this thesis implements Open MPI which is maintained by a consortium of research institutes (e.g. Auburn University, University of Wisconsin, etc.), corporations (e.g. IBM, Intel, Broadcom, etc.) and various other independent researchers. As can be understood from the discussion above, MPI is particularly helpful in reducing computational time when the data to be processed is huge.

2.3.2 General Purpose Graphics Processing Unit (GPGPU)

The General Purpose Graphics Processing Unit is a relatively new development. There was a rapid increase in the performance of the processors like Intel and AMD in terms of clock frequency from GFLOPS (Giga floating-point operations per second) to TFLOPS (Tera floating-point operations per second) for more than two decades (Kirk and Hwu, 2013). However, this improvement slowed down since 2003 due to an increase in energy consumption and heat dissipation; thereby limiting

the increase in clock frequency. This prompted chip manufacturers to switch to designing multi-core CPUs to increase the CPU efficiency which in turn impacted the development of softwares greatly (Sutter and Larus, 2005).

Since 2003, microprocessor manufacturers have started using two main architectures (Hwu et al., 2008) viz. multi-core CPUs and GPUs.

In multi-core CPUs, the number of cores are increasing with each new generation and each core designed to handle operations sequentially. MPI relies on this architecture to achieve parallelism on multiple cores.

The second architecture is that of the GPUs (e.g. NVIDIA GTX) which itself is designed with multi-threading architecture with threads as large as 57,000 (NVIDIA 1080Ti with 3584cores and 16 threads) in order to achieve parallelism. NVIDIA's CUDA (Compute Unified Device Architecture) library enables to use the full functionality of the GPUs. As can be understood, because of GPUs ability to handle multiple threads gives it an upper hand in handling large parallel computations faster than MPI. However, a program with a fewer threads will undoubtedly perform faster on CPUs than on GPUs but numerically intensive operations perform much faster on GPUs. Thus, in 2007, NVIDIA's CUDA was designed to execute joint CPU-GPU operation to achieve maximum efficiency in numerical computations. More recent versions of commercial softwares like ANSYS Fluent, MATLAB have started incorporating CUDA library into their packages. However, in this thesis only MPI has been used to handle large CFD domains and CUDA has not been implemented.

2.3.3 PNA in wind farms

This ability of parallel numerical algorithms has been getting the attention of the researches for the past two decades and they have been coming up with different programs to analyse wind farms. PNA had been in use for quite sometime now. For example, Hussein and El-Shishiny, 2012 proposed a computational framework for wind farm simulation using distributed memory and massively parallel high performance computing platforms over micro-scale using RANS and Virtual Blade Model to model the wind turbines. The framework is able to run the models on supercomputers. Baez-Vidal et al., 2013 did LES simulation of wind farms with Actuator Line Method. They highlighted the fact that parallel computation of LES

with Wind Turbine Models (WTM) demand different domain decomposition and these do not necessarily coincide. They proposed a coupling strategy and simulated the wind turbine wake. However, though the parallelization was found to work properly, the simulations were not able to resolve the flows with enough accuracy.

With advances made in PNA and multi-core simulations, researchers even worked on several blade resolved models to capture FSI effects for wind turbines. In this approach, unstructured meshes, as shown in Figure 2.3 are used to model the fluid around the wind turbine blades. Works of Lavely et al., 2014, Mittal et al., 2016, Ehrich et al., 2018, to name a few, present blade-resolved models which in a way cater to FSI of single wind turbines. Kirby et al., 2019 pushed the boundaries for FSI even further and used blade resolved models for wind farm consisting of as many as 96 wind turbines and ran it on 44,928 cores. However, one can see that these blade resolved models, though are able to capture FSI, they require extensive computational resources because of the requirement to model the unstructured mesh around the blades which is extremely dense compared to the mesh size required for the wind field. This is where FSI methods discussed in Section 2.2.3 can come into play to reduce computational cost. But, even with FSI, for domains extending over several square kilometers of area like wind farms, PNAs are unavoidable.

This prompted researchers to further look into this area. For instance, Chand et al., 2010 developed CgWind based on PNA for LES of wind farms. It used matrix free multi-grid approach, compact discretizations and approximate factorizations. The work of Department of Mechanical Engineering, Danmarks Tekniske Universitet, Denmark and The Department of Wind Energy at RisøNational Laboratory in developing Ellipsys2D/3D deserves recognition. It used the concepts of PNA and "multiblock finite volume discretization of the incompressible Reynolds Averaged Navier-Stokes (RANS) equations in general curvilinear coordinates" (Sørensen, 2015). Ellipsys3D solver, however does not model the wind turbine in itself and just models the wind field based on RANS. The strength of PNA in handling large domains can be understood from the fact that Ivanell et al., 2008 used Ellipsys3D solver to carry out LES of NSE model in a wind farm containing upto nine wind turbines and with meshpoints as large as six million.

Similarly, softwares like SOWFA (Simulator fOr Wind Farm Applications) by National Renewable Energy Laboratory (NREL), USA is based upon a technical report by Sørensen and Shen, 2002 where wind turbine blades are discretized into

spanwise sections; lifts and drags are computed based on the incoming flow and these forces are again projected back into the flow for simulation. Others like PALM by Leibniz University Hannover, Germany (Maronga et al., 2015) incorporate advanced actuator disc approach by dividing the disc into concentric circles subjected to varying forces and is a project still under development. But, it can be seen that these models treat the fluid and structure separately to model the wind field. Codes like WRF-LES (by NCAR, USA) and SP-WIND (by KU-Leuven, Belgium) are also used which implement such algorithms for carrying out simulation of wind farms. It is worth noting that more recently researchers in wind energy domain are using these parallel numerical tools like Ellipsys3D, SOWFA, PALM, WRF-LES, SP-WIND extensively to understand the aerodynamics of wind farms. This clearly shows an increased importance of PNAs in the research arena.

Having discussed the existing works covering wind turbines, wind farm models, fluid dynamics and the parallel numerical algorithms, in the subsequent chapters the main work carried out in this thesis is presented.

Chapter 3

Application of parallel numerical algorithms to fluid flow

3.1 Introduction

Analytical solutions for many PDEs frequently encountered in engineering are still not known. However, the solutions for these equations are still needed to design the engineering system. For example, in the present research work the main focus is on the aerodynamics of the wind field in wind farms governed by the Navier-Stokes Equations, the analytical solution for which is not known. Regardless, a numerical solution is still required for engineering design of the wind turbines. This is where numerical analysis comes into picture which do not provide an exact solution but provide a result approximate enough for designing the system. The basic approach of a numerical analysis involves discretizing the physical domain considered for analysis in a way such that the numerical solution closely resembles the exact solution. However, once a domain has been discretized, its behaviour remains no longer the same as the continuous domain and hence, a careful investigation of the discretized system is required. The discussion here will be confined to the PDEs encountered in the field of CFD only. Discretization methods such as Finite Difference Method (FDM), Finite Element Method (FEM), Finite Volume Method (FVM) have their own unique approach to discretization and numerical computation. Commercial softwares like Ansys, Abaqus, etc. efficiently use these techniques to model the fluid problems. But, though these softwares cater to most engineering needs, they

cannot be used for more specific requirements particularly when it comes to the research arena. Bearing this fact in mind, Python3 has been used instead to suit the specific requirement of this work. As discussed earlier as the physical domain becomes more complex or large, the data to be processed also increases and hence, to optimize computational resources like CPU time and memory requirement, parallel computation is essential across multiple CPU cores. Hence, this chapter begins by discussing FDM and FVM and then moves onto the application of such parallel computation strategies to solve the Finite Difference Equation (FDE).

3.2 Fluid flow equations

Any fluid flow problem is governed by three conservation laws. For any Newtonian fluid of density ρ, velocity, $\mathbf{V}$, pressure, P, temperature, T and thermal conductivity, k, the PDEs governed by these conservation laws are:-

1. Conservation of mass (equation of continuity)

$$\frac{\partial \rho}{\partial t} + \nabla(\rho.\mathbf{V}) = 0 \tag{3.1}$$

2. Conservation of momentum (Navier-Stokes equation)

$$\frac{\partial(\rho V_j)}{\partial t} + \nabla(\rho V_j \mathbf{V}) = -\frac{\partial P}{\partial X_j} + \nabla(\mu \nabla V_j) + S_{M_j}; j \in \{1, 2, 3\} \tag{3.2}$$

3. Conservation of energy

$$\frac{\partial \rho i}{\partial t} + \nabla(\rho i \mathbf{V}) = -P\nabla V_j + \nabla(k\nabla T) + \mathbf{\Phi} + S_i \tag{3.3}$$

The terms S_{M_j} and S_i are the source terms and $\mathbf{\Phi}$ is the energy dissipation term and t indicates time and V_j is component of $\mathbf{V}$ in direction j. It can be observed from the structure of the above PDEs that they have a common pattern which for a general variable, Λ can be written in the form

$$\frac{\partial(\rho\Lambda)}{\partial t} + \nabla(\rho\Lambda\mathbf{V}) = \nabla(\Gamma\nabla\Lambda) + S_\Lambda. \tag{3.4}$$

These equations are referred to as the transport equations for the fluid property, Λ and in physical sense would mean

Rate of increase of Λ of fluid element + Net rate of flow of Λ out of fluid element = Rate of increase of Λ due to diffusion + Rate of increase of Λ due to sources

These equations form the backbone of any fluid flow problem.

3.3 Discretization methods

A look at the structure of the PDEs introduced in the previous section clearly indicates their complex nature and is indicative of the fact that an analytical solution of such equations is difficult to obtain. This is where discretization comes into picture which aims at writing down the differential operator as Taylor series expansion and solve those equations instead of these original PDEs. In this section, a brief description of Finite Difference Method and Finite Volume Method, two of the most common approaches used in the context of CFD has been introduced.

3.3.1 Finite Difference Method

The FDM, which is based on Taylor series has been implemented in the present work. Though FEM and FVM are much more structured compared to FDM, this approach is much useful when dealing with large domains like atmosphere where irregular grids do not govern the model. In the present work, since large rectangular domains in atmospheric boundary layer has been primarily dealt with, FDM has been chosen as the preferred discretization method. Also, as will be clear from the subsequent chapters, some of the methods that have been implemented have been formulated using FDM and whether or not they could be directly applied using FVM is yet to be explored.

Any real or complex function that is infinitely differentiable can be written in the form of an infinite series referred to as 'Taylor series'. Eq.3.5 gives the Taylor series for functions of one and two variables i.e. $f(x)$ and $g(x, y)$ respectively.

$$f(x \pm h_x) = f(x) \pm h_x f_x(x) + \frac{h_x^2}{2!} f_{xx}(x) \pm \frac{h_x^3}{3!} f_{xxx} + \frac{h_x^4}{4!} f_{xxxx} + ... \tag{3.5a}$$

$$g(x \pm h_x, y \pm h_y) = g(x, y) \pm h_x g_x(x, y) \pm h_y g_x(x, y) + \frac{h_x^2}{2!} g_{xx} + \frac{h_y^2}{2!} g_{yy} + h_x h_y g_{xy} \pm ... \tag{3.5b}$$

where, h_x and h_y are the distances between two adjacent points in the domain along

x and y directions respectively. If the higher order terms are neglected, the function can be considered to be a near approximation of the original function. From eq.3.5a, subtracting $f(x-h_x)$ from $f(x+h_x)$ gives

$$\begin{aligned} f(x+h_x) - f(x-h_x) &= 2h_x f_x(x) + \frac{h_x^3}{3} f_{xxx}(x) + ... \\ \Rightarrow f_x(x) &= \frac{f(x+h_x) - f(x-h_x)}{2h_x} + \frac{h_x^2}{6} f_{xxx}(x) + ... \\ \Rightarrow f_x(x) &\simeq \frac{f(x+h_x) - f(x-h_x)}{2h_x} + \mathcal{O}(h_x^2). \end{aligned} \tag{3.6}$$

Similarly, adding $f(x+h_x)$ from $f(x-h_x)$ gives

$$\begin{aligned} f(x+h_x) + f(x-h_x) &= 2f(x) + h_x^2 f_{xx}(x) + \frac{h_x^4}{12} f_{xxxx}(x) + ... \\ \Rightarrow f_{xx}(x) &= \frac{f(x+h_x) - 2f(x) + f(x-h_x)}{h_x^2} + \frac{h_x^2}{12} f_{xxxx}(x) + ... \\ \Rightarrow f_{xx}(x) &\simeq \frac{f(x+h_x) - 2f(x) + f(x-h_x)}{h_x^2} + \mathcal{O}(h_x^2). \end{aligned} \tag{3.7}$$

Thus, ignoring the higher order terms, the first and the second order derivative of function $f(x)$ at a point x can be expressed in terms of its adjacent neighbouring points and it is second order accurate. This approach is referred to as **Central Difference** in FDM wherein the value of the derivative at a certain point is dependent on its immediate surrounding points only.

In general, for any fluid flow problem considered, the information is always available for the boundary nodes. If the value of the dependent variable is known at these nodes, it is referred to as Dirichlet Boundary Condition (DBC). However, in some cases it may so happen that instead of the values at these boundary nodes, the slope of the dependent variable at the boundary is defined which is referred to as Neumann Boundary Condition (NBC)). This is where the **One-sided Difference** (*Forward Difference and Backward Difference*) scheme comes into picture. The second order forward and backward difference equations (Hanifi, 2010) for the first derivative of $f(x)$ is given by eq.3.8a and eq.3.8b respectively.

$$f_x(x) \simeq \frac{-3f(x) + 4f(x+h_x) - f(x+2h_x)}{2h_x} + \mathcal{O}(h_x^2) \tag{3.8a}$$

$$f_x(x) \simeq \frac{3f(x) - 4f(x-h_x) + f(x-2h_x)}{2h_x} + \mathcal{O}(h_x^2) \tag{3.8b}$$

In a similar manner, the derivatives of function $g(x, y)$ in 2D can be expressed as

$$g_x(x,y) \simeq \frac{g(x+h_x,y) - g(x-h_x,y)}{2h_x} + \mathcal{O}(h_x^2) \text{ (Central Difference)} \tag{3.9a}$$

$$g_x(x,y) \simeq \frac{-3g(x,y) + 4g(x+h_x,y) - g(x+2h_x,y)}{2h_x} + \mathcal{O}(h_x^2) \text{ (Forward Difference)} \tag{3.9b}$$

$$g_x(x,y) \simeq \frac{3g(x,y) - 4g(x-h_x,y) + g(x-2h_x,y)}{2h_x} + \mathcal{O}(h_x^2) \text{ (Backward Difference)} \tag{3.9c}$$

$$g_y(x,y) \simeq \frac{g(x,y+h_y) - g(x,y-h_y)}{2h_y} + \mathcal{O}(h_y^2) \text{ (Central Difference)} \tag{3.9d}$$

$$g_y(x,y) \simeq \frac{-3g(x,y) + 4g(x,y+h_y) - g(x,y+2h_y)}{2h_y} + \mathcal{O}(h_y^2) \text{ (Forward Difference)} \tag{3.9e}$$

$$g_y(x,y) \simeq \frac{3g(x,y) - 4g(x,y-h_y) + g(x,y-2h_y)}{2h_y} + \mathcal{O}(h_y^2) \text{ (Backward Difference)} \tag{3.9f}$$

$$g_{xx}(x,y) \simeq \frac{g(x+h_x,y) - 2g(x,y) + g(x-h_x,y)}{h_x^2} + \mathcal{O}(h_x^2) \tag{3.9g}$$

$$g_{yy}(x,y) \simeq \frac{g(x,y+h_y) - 2g(x,y) + g(x,y-h_y)}{h_y^2} + \mathcal{O}(h_y^2). \tag{3.9h}$$

Unless otherwise specified, the FDM based derivations presented in this work are all based on the above mentioned second order accurate expressions.

3.3.2 Finite Volume Method

As pointed out in Section 3.3.1, even though the work carried out uses FDM primarily, an insight into FVM is required as well since some of the algorithms commonly used in the context of FVM are required in the current research and as such are needed to be modified to suit FDM. The finite volume method is one of the most widely accepted methods for CFD simulations of fluid flow problems. A brief description of how FVM works is provided in this section. The methodology is based on conservation laws introduced in Section 3.2 as these laws govern the PDEs encountered in any fluid dynamics problem. This method divides the entire physical domain getting analysed into a mesh referred to as **cells** within which the equations are to be analysed. Unlike FDM which uses Taylor series to discretize the PDE

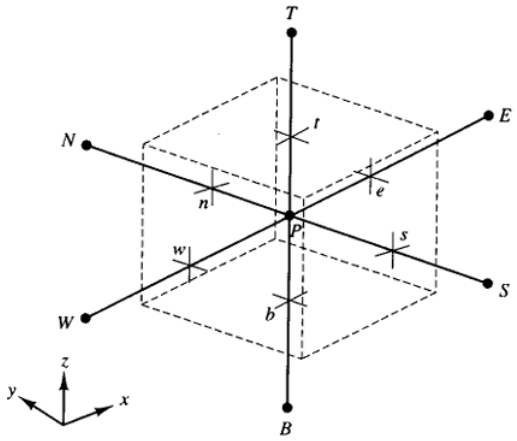


Figure 3.1: A cell in three dimensions and neighbouring nodes [Adapted from Versteeg and Malalasekera, 2007]

itself, FVM works on integration over Control Volume (CV) of these cells as shown below

$$\int_{CV} \frac{\partial(\rho\Lambda)}{\partial t} dV_o + \int_{CV} \nabla(\rho\Lambda\mathbf{V}) dV_o = \int_{CV} \nabla(\Gamma\nabla\Lambda) dV_o + \int_{CV} S_\Lambda dV_o. \tag{3.10}$$

A typical control volume is shown in Figure 3.1. Node P indicates the node where Λ is analysed. E, W, N, S, T, B indicate nodes and e, w, n, s, t, b indicates face of the cell towards east, west, north, south, top and bottom respectively. Subsequent reference and discussion related to FVM refer to the notations used in this figure.

Gauss's divergence theorem for a vector, $\mathbf{a}$ states that

$$\int_{CV} \nabla.\mathbf{a} dV_o = \int_A \mathbf{n}.\mathbf{a} dA. \tag{3.11}$$

Hence, the above equation can be rewritten as

$$\frac{\partial}{\partial t}\left(\int_{CV} (\rho\Lambda) dV_o\right) + \int_A \mathbf{n}.(\rho\Lambda\mathbf{V}) dA = \int_A \mathbf{n}.(\Gamma\nabla\Lambda) dA + \int_{CV} S_\Lambda dV_o \tag{3.12}$$

and for steady state problems, the equation becomes

$$\int_A \mathbf{n}.(\rho\Lambda\mathbf{V}) dA = \int_A \mathbf{n}.(\Gamma\nabla\Lambda) dA + \int_{CV} S_\Lambda dV_o \tag{3.13}$$

For a steady flow problem, in physical sense, w.r.t. a small cell with node centred at P (refer Figure 3.1) the integration terms of the equation for east-west direction

would mean computing the difference in flux across the east and west faces of the cell i.e.

$$\int_A \mathbf{n}.(\rho\Lambda\mathbf{V})dA = (\rho A\Lambda\mathbf{V})_e - (\rho A\Lambda\mathbf{V})_w \tag{3.14}$$

Putting this in eq.3.13,

$$\int_A \mathbf{n}.(\Gamma\nabla\Lambda)dA + \int_{\triangle CV} S_\Lambda dV_o = \left(\Gamma A\frac{\partial\Lambda}{\partial x}\right)_e - \left(\Gamma A\frac{\partial\Lambda}{\partial x}\right)_w + \overline{S}\triangle V_0 \tag{3.15}$$

where, $\triangle V_0$ is the volume of a cell and $\overline{S}$ is the average value of source over $\triangle V_0$. For unsteady flows, integration over time is required additionally.

$$\begin{aligned}\int_{CV}\int_t^{t+\triangle t}\frac{\partial(\rho\Lambda)}{\partial t}dtdV_o + \int_t^{t+\triangle t}\int_A \mathbf{n}.(\rho\Lambda\mathbf{V})dAdt = \\ \int_t^{t+\triangle t}\int_A \mathbf{n}.(\Gamma\nabla\Lambda)dAdt + \int_t^{t+\triangle t}\int_{CV} S_\Lambda dV_o dt\end{aligned} \tag{3.16}$$

The first term of the equation when integrated over time physically means evaluating the difference in magnitude of $\rho\Lambda$ between time instants t and $t+\triangle t$

$$\int_{CV}\int_t^{t+\triangle t}\frac{\partial(\rho\Lambda)}{\partial t}dtdV_o = (\rho\Lambda)_P\triangle V_0 - (\rho\Lambda^0)_P\triangle V_0 \tag{3.17}$$

where, Λ^0 is the value of Λ at time, t. As far as the remaining terms are concerned, integration over time could be done w.r.t. to time, t or with respect to time, $t+\triangle t$. Alternately, value w.r.t. some weight on value at time instants t and $t+\triangle t$ can also be used to evaluate these integrals. From the final forms of equations presented above, it can be seen that there are still terms like $\partial\Lambda/\partial x$. At this stage, Taylor series come into play and depending on the nature of the problem, appropriate finite differencing scheme is incorporated to evaluate such terms. From these discussions, it is evident that both FDM and FVM use Taylor series to solve the transport equations but in FDM, the PDEs are discretized straightaway whereas in FVM, integration is performed over a control volume before incorporating the finite difference equations.

3.4 Solving linear systems

From the previous section, it is understood that the partial derivative terms of any linear PDE can be conveniently expressed as a set of algebraic equations for each

and every node inside the physical domain using the FDE. In matrix form, the resulting set of linear equations can be expressed in the form

$$\mathbf{Ax} = \mathbf{b} \tag{3.18}$$

where, $\mathbf{A}$ is the coefficient matrix of size $n \times n$, $\mathbf{x} \in \mathbb{R}^n$ is the vector of unknowns to be determined (i.e. the value of the dependent variable at all the internal nodes), $\mathbf{b} \in \mathbb{R}^n$ is a known vector and n is the total number of internal nodes. An element of $\mathbf{A}$ can be represented by $\{a_{ij} : a_{ij} \in \mathbb{R}\}$ where i represents the row index and j represents the column index and $i, j \in \{0, 1, 2, .., n-1\}$. It is evident that for domains with a large number of node points, solution to the above mentioned system using matrix inversion is not a feasible way. Additionally, for most fluid flow problems, $\mathbf{A}$ is extremely sparse because the Taylor series expansion at any particular node inside the domain considers dependency on one or two adjacent nodes only and not on all adjacent nodes. Thus, $\mathbf{A}$ is usually ill-conditioned and as such, the matrix inversion might often destabilize the numerical solution. Further, if the elements of A is stored in memory contiguously, quite a large amount of space will be wasted to store the zero elements. This is where memory management and iterative algorithms come into picture.

3.4.1 Memory management and iterative solvers

In order to avoid storing the zero entries of the linear systems, two popular approaches are in use viz. Compressed Sparse Row (CSR) and Compressed Sparse Column (CSC). In CSR format, only the non-zero elements of $\mathbf{A}$, are stored in a contiguous real array, AA in row major order. Along with that two more integer arrays, JA and IA are defined. JA stores the column index of the non-zero entries in $\mathbf{A}$ and IA is an array of $n + 1$ elements such that it contains the pointers to the beginning of each row. CSC format follows a similar pattern but in column major order. Besides these, other formats like Ellpack-Itpack format are also quite common and often used.

The properties of the coefficient matrix $\mathbf{A}$ determine the kind of iterative solver or the numerical scheme that will be best suited for the problem in hand. Some of these solvers include Conjugate Gradient (CG), Bi-Conjugate Gradient Stable (BiCGStab), Generalized Minimal Residual Method (GMRES), Conjugate

Gradient Squared (CG-S), etc. For instance, if **A** is symmetric, iterative methods like Conjugate Gradient (CG) can be implemented and if assymetric, Bi-Conjugate Gradient Stable (BiCGStab) can be used. GMRES is known to be extensively memory consuming compared to the other algorithms. As such, the work carried out in this thesis primarily depends on CG and BiCGStab and its variants. The steps for BiCGStab are given below in Algorithm 1.

Algorithm 1 Bi-Conjugate Gradient Stable (BiCGStab) (van der Vorst, 1992)

Assume initial trial value for vector $\mathbf{x}$, say $\mathbf{x}_0$.
$\mathbf{r}_0 = \mathbf{b} - \mathbf{A}\mathbf{x}_0$.
Choose an arbitrary vector $\mathbf{r}^*$ such that $< \mathbf{r}^*, \mathbf{r}_0 > \neq 0$, e.g., $\mathbf{r}^* \leftarrow \mathbf{r}_0$.
$\rho_0 = \alpha = \omega_0 = 1$.
$\boldsymbol{\nu}_0 = \mathbf{p}_0 = 0$.
for $j = 1, 2, 3, \ldots$ **do**
  $\rho_j \leftarrow < \mathbf{r}^*, \mathbf{r}_{j-1} >$; $\beta \leftarrow (\rho_j / \rho_{j-1})(\alpha / \omega_{j-1})$
  $\mathbf{p}_j \leftarrow \mathbf{r}_{j-1} + \beta(\mathbf{p}_{j-1} - \omega_{j-1}\boldsymbol{\nu}_{j-1})$
  $\boldsymbol{\nu}_j \leftarrow \mathbf{A}\mathbf{p}_j$
  $\alpha \leftarrow \rho_j / < \mathbf{r}^*, \boldsymbol{\nu}_j >$
  $\mathbf{s} \leftarrow \mathbf{r}_{j-1} - \alpha\boldsymbol{\nu}_j$
  $\mathbf{t} \leftarrow \mathbf{A}\mathbf{z}$
  $\omega_j \leftarrow < \mathbf{t}, \mathbf{s} > / < \mathbf{t}, \mathbf{t} >$
  $\mathbf{x}_j \leftarrow \mathbf{x}_{j-1} + \alpha\mathbf{p}_j + \omega_j\mathbf{z}$; if $\mathbf{x}_j$ accurate enough then quit
  $\mathbf{r}_j = \mathbf{s} - \omega_j\mathbf{t}$
end for

The interesting thing about these iterative algorithms is that the matrices are never formed explicitly and the calculation steps are done not by matrix-matrix or matrix-vector operations but rather by solving the expression corresponding to the unknowns. This itself saves a huge amount of memory. An important point to note about these algorithms is that they work by splitting **A** into 3 other matrices $\mathbf{D}, -\mathbf{E}, -\mathbf{F}$ where, **D** constitutes the diagonal elements only and -**E** and -**F** represent strictly the lower and upper part respectively (refer Figure 3.2). The significance of this partitioning will become clear in Section 3.4.2 where the concept on preconditioner has been introduced.

3.4.2 Preconditioner

Iterative solvers lack robustness when compared to direct solvers and though they are able to handle large systems, this shortcoming hinders their performance. This is

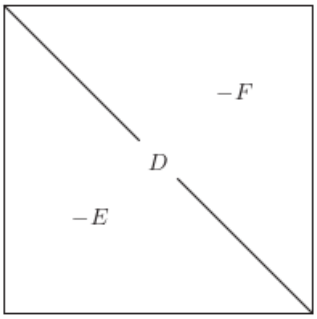


Figure 3.2: Initial partitioning of $\mathbf{A}$

where precondtioning comes in. Preconditioning is a technique wherein the original linear system is transformed into an equivalent linear system which is relatively easier to solve. Consider a preconditioner, $\mathbf{M}$ of form $\mathbf{M} = \mathbf{M}_1\mathbf{M}_2 \approx \mathbf{A}$. Transforming eq.3.18 with a preconditioner would mean solving an equivalent system

$$\tilde{\mathbf{A}}\tilde{\mathbf{x}} = \tilde{\mathbf{b}} \tag{3.19}$$

where, $\tilde{\mathbf{A}} = \mathbf{M}_1^{-1}\mathbf{A}\mathbf{M_2}^{-1}$, $\tilde{\mathbf{x}} = \mathbf{M}_2 x$ and $\tilde{\mathbf{b}} = \mathbf{M}_1^{-1}\mathbf{b}$. This is called preconditioning from both sides. However, if $\mathbf{M}_1 = \mathbf{I}$, we get a right preconditioned equation and if $\mathbf{M}_2 = \mathbf{I}$, we get a left preconditioned equation. Some of the common types of preconditioners are Jacobi (JA), Gauss-Seidel (GS), Successive Overrelaxation (SOR) and Symmetric Successive Overrelaxation (SSOR). These are represented by:

$$\begin{aligned} \mathbf{M}_{JA} &= \mathbf{D} \\ \mathbf{M}_{GS} &= \mathbf{D} - \mathbf{E} \\ \mathbf{M}_{SOR} &= \frac{1}{\omega}(\mathbf{D} - \omega\mathbf{E}) \\ \mathbf{M}_{SSOR} &= \frac{1}{\omega(2-\omega)}(\mathbf{D} - \omega\mathbf{E})\mathbf{D}^{-1}(\mathbf{D} - \omega\mathbf{F}) \end{aligned} \tag{3.20}$$

In the current work, the right preconditioned system has been implemented for reasons which will be explained in relevant sections at a later stage.

Preconditioners can also be designed to suit the requirement of the problem and can be far more complicated involving integral calculations as well. Based on this concept of preconditioning, the different iterative algorithms like CG and BiCGStab are also modified to have their preconditioned versions as well (known as Preconditioned Conjugate Gradient (PCG) if $\mathbf{A}$ ia symmetric and Preconditioned

Bi-Conjugate Gradient Stable (PBiCGStab) if assymetric. The numerical solution to the flow problems presented in this thesis implement PCG and PBiCGStab. These algorithms are presented below in Algorithm 2 and Algorithm 3.

Algorithm 2 Preconditioned Conjugate Gradient (PCG) (Saad, 2003)

Assume initial trial value for vector $\mathbf{x}$, say $\mathbf{x}_0$.
$\mathbf{r}_0 \leftarrow \mathbf{b} - \mathbf{A}\mathbf{x}_0$.
$\mathbf{z}_0 \leftarrow \mathbf{M}^{-1}\mathbf{r}_0$, $\mathbf{p}_0 \leftarrow \mathbf{z}_0$
for j = 0, 1, 2, ... **do**
  $\alpha_j = < \mathbf{r}_j, \mathbf{z}_j > / < \mathbf{A}\mathbf{p}_j, \mathbf{p}_j >$
  $\mathbf{x}_{j+1} \leftarrow \mathbf{x}_j + \alpha_j \mathbf{p}_j$; if $\mathbf{x}_j$ accurate enough then quit
  $\mathbf{r}_{j+1} \leftarrow \mathbf{r}_j - \alpha_j \mathbf{A}\mathbf{p}_j$
  $\mathbf{z}_j + 1 \leftarrow \mathbf{M}^{-1}\mathbf{r}_{j+1}$
  $\beta_j \leftarrow < \mathbf{r}_{j+1}, \mathbf{z}_{j+1} > / < \mathbf{r}_j, \mathbf{z}_j >$
  $\mathbf{p}_{j+1} \leftarrow \mathbf{z}_{j+1} + \beta_j \mathbf{p}_j$
end for

Algorithm 3 Preconditioned Bi-Conjugate Gradient Stable (PBiCGStab) (van der Vorst, 1992)

Assume initial trial value for vector $\mathbf{x}$, say $\mathbf{x}_0$.
$\mathbf{r}_0 \leftarrow \mathbf{b} - \mathbf{A}\mathbf{x}_0$.
Choose an arbitrary vector $\mathbf{r}^*$ such that $< \mathbf{r}^*, \mathbf{r}_0 > \neq 0$, e.g., $\mathbf{r}^* \leftarrow \mathbf{r}_0$
$\rho_0 = \alpha = \omega_0 = 1$
$\boldsymbol{\nu}_0 = \mathbf{p}_0 = 0$
for j = 1, 2, 3, ... **do**
  $\rho_j \leftarrow < \mathbf{r}^*, \mathbf{r}_{j-1} >$; $\beta = (\rho_j/\rho_{j-1})(\alpha/\omega_{j-1})$
  $\mathbf{p}_j \leftarrow \mathbf{r}_{j-1} + \beta(\mathbf{p}_{j-1} - \omega_{j-1}\boldsymbol{\nu}_{j-1})$
  $\mathbf{y} \leftarrow \mathbf{M}^{-1}\mathbf{p}_j$
  $\boldsymbol{\nu}_j \leftarrow \mathbf{A}\mathbf{y}$
  $\alpha \leftarrow \rho_j / < \mathbf{r}^*, \boldsymbol{\nu}_j >$
  $\mathbf{s} \leftarrow \mathbf{r}_{j-1} - \alpha\boldsymbol{\nu}_i$
  $\mathbf{z} \leftarrow \mathbf{M}^{-1}\mathbf{s}$
  $\mathbf{t} \leftarrow \mathbf{A}\mathbf{z}$
  $\omega_j \leftarrow < \mathbf{M}_1^{-1}\mathbf{t}, \mathbf{M}_1^{-1}\mathbf{s} > / < \mathbf{M}_1^{-1}\mathbf{t}, \mathbf{M}_1^{-1}\mathbf{t} >$
  $\mathbf{x}_j \leftarrow \mathbf{x}_{j-1} + \alpha\mathbf{y} + \omega_j\mathbf{z}$; if $\mathbf{x}_j$ accurate enough then quit
  $\mathbf{r}_j \leftarrow \mathbf{s} - \omega_i\mathbf{t}$
end for

3.4.3 Matrix reorderings

Reordering the matrices is a common practice in implementing parallelism. The primary reason for reordering the matrix is to decouple the nodes so that the values of the nodes with the same colour i.e. the nodes independent of each other can be updated simultaneously. Matrix reorderings include different strategies like level-set orderings, independent set orderings, multi-colour orderings. This thesis implements the multi-coloring ordering strategy. In multi-colour ordering, it is required to be ensured that no two adjacent nodes of a graph have the same colour and that the minimum number of colours is used to colour the graph. Greedy multi-colouring algorithm (refer Algorithm 4) is one such approach. Based on this multi-colouring

Algorithm 4 Greedy multicoloring algorithm (Saad, 2003)

for j = 1, 2, ..., n **do**
  Set color(i) $\leftarrow 0$
end for
for j = 1, 2, ..., n **do**
  Set color(i) $\leftarrow min\{k > 0 | k \neq Color(j), \forall j \in Adj(i)\}$
end for

ordering scheme, one of the most common ordering scheme is an ordering with just two colours i.e. the red-black colouring scheme which is discussed in detail in Section 3.5.1.

3.4.4 Multi-core processing

In Chapter 2, it has been pointed out that Open MPI has been used in the present research work. In this section, few details of MPI which have been implemented in this research work are discussed. Open MPI project (Gabriel et al., 2004) defines MPI as

"Written by the MPI Forum (a large committee comprised of a cross-section between industry and research representatives), MPI is a standardized API typically used for parallel and/or distributed computing".

Open MPI is an open source which merges the features of three well known projects viz. FT-MPI from the University of Tennessee, LA-MPI from Los Alamos National Laboratory and LAM/MPI from Indiana University and contributions from the PACX-MPI team at the University of Stuttgart.

The parallel computation codes developed for the simulations carried out in this thesis use the python library **mpi4py** which offers a wide range of MPI implementations including Open MPI, MPICH, etc. In this thesis, mpi4py has been built on Open MPI version 3.1.6. The advantage is that the program can be run on cluster of nodes and not only on a single computer with multiple cores. At this stage, it is essential to introduce, few terms that are used quite often in the context of MPI.

1. size: It is the total number of cores running the parallel code, say PS. e.g. if there are 4 interconnected nodes in a cluster each with 6 cores, then the total available cores for parallel processing is $4 \times 6 = 24$. Out of this if only 12 cores are utilized, then the size of the MPI will be 12
2. rank: It is the identification number of the core starting from $0, 1, 2, ..., PS-1$ (also referred to as processor $P_0, P_1, P_2, ..., P_{PS-1}$), assigned by the operating system.
3. communicator: The communicator holds a group of processes which can communicate with one another.

MPI provides a wide range of various other functionalities. However, the five major operations are used in the current thesis to carry out a fast and efficient multi-core simulations include:

1. MPI_Bcast: Broadcast data across all the processors.
2. MPI_Send: Send data from processor P_i to processor P_j.
3. MPI_Recv: Receive data from processors P_i to processors P_j.
4. MPI_Scatter: Scatter data equally across all the processors from rank j (usually 0). In case a very large matrix, $\mathbf{M}$ of dimension $n \times m$ is required to be processed parallely for some operation, the scatter operation will split $\mathbf{M}$ into equal blocks of matrices of dimensions $n/PS \times m$ which will then be processed by each available core at the same instance of time, thereby achieving a faster computation time. However, as can be observed, there can be two conditions i.e. $1.\lfloor n/PS \rfloor = n/PS$; $2.\lfloor n/PS \rfloor \neq n/PS$. For the first case, data can be easily distributed equally across all the cores. But, for the second case, $\mathbf{M}$ is padded with a zero matrix, $\mathbf{M}_0$ of dimension $n_0 \times m$ such that $\lfloor (n + n_0)/PS \rfloor = (n_0 + n)/PS$ and thus, scatter operation can be done seamlessly.

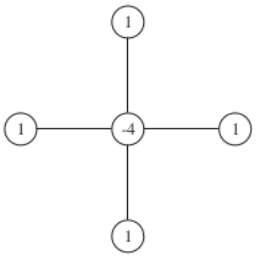


Figure 3.3: Typical structure of a five point stencil. The value of -4 is valid only if the grid distances are the same in both directions.

5. MPI_Gather: This operation gathers data scattered across all the cores and stores them in processor of rank j and is inverse of the scatter operation. As can be understood, in order to retrieve the original matrix, the zero padding done during the scatter operation needs to be removed.

3.5 Application to elliptic PDEs

Now that memory management, iterative solvers and multi-core simulations have been introduced, the next task is to look into how these concepts can be used in FDE for elliptic PDEs. A typical non-homogeneous elliptic PDE (or Poisson's equation) in Einstein notation can be written in the form

$$\frac{\partial^2 \phi}{\partial X_i \partial X_i} = -b(X_1, X_2, X_3); (X_1, X_2, X_3) \in \Omega \tag{3.21}$$

where, Ω represents the physical domain, $X_j; j \in \{1, 2, 3\}$ indicates the coordinates along X_1, X_2, X_3 axes respectively and $-b(X_1, X_2, X_3)$ is the source term.

Let $p = -\frac{1}{\triangle X_1^2}$, $q = -\frac{1}{\triangle X_2^2}$, $r = -\frac{1}{\triangle X_3^2}$, $s = -2(p+q+r)$ where, $\triangle X_1$, $\triangle X_2$, $\triangle X_3$ are the node to node spacing in X_1, X_2 and X_3 direction respectively. In 2D, say in $X_1 - X_3$ plane, the second order accurate central difference scheme (eq.3.9g and eq.3.9h) results in a 5 point stencil (refer Figure 3.3) when used to discretize eq.3.21. The discretized equation takes the form

$$b_{i,j} - s\phi_{i,j} - p(\phi_{i+1,j} + \phi_{i-1,j}) - r(\phi_{i,j+1} + \phi_{i,j-1}) = 0. \tag{3.22}$$

where, at a typical internal node denoted by (i, j) having coordinate (X_1, X_3), the value of source term is $-b(X_1, X_3) = -b_{i,j}$, the value of dependent variable is $\phi_{i,j}$ and

the values of dependent variables at nodes on left, right, bottom and top of node at (i, j) are $\phi_{i-1,j}, \phi_{i+1,j}, \phi_{i,j-1}, \phi_{i,j+1}$ respectively. Hoffman, 2001 has shown that the finite difference approximation of such an elliptic PDE is consistent with the original PDE. Like any other PDE, the solution of the aforementioned system depends on the nature of the boundary conditions, which can be Dirichlet (DBC), Neumann (NBC), Cauchy (CBC) or Robin (RBC). If all the boundary conditions are DBC, the aforementioned system remains unaffected since the values of the dependent variable, ϕ are known at the boundary. The presence of NBC or CBC however presents a different story altogether. It modifies eq.3.22 for the penultimate nodes (i.e. the nodes adjacent to the boundary) because instead of ϕ, the directional derivative normal to the boundary is a known parameter at the boundary. To address this scenario, two approaches can be used to frame the equations for the penultimate nodes viz.

1. one sided differencing
2. central differencing considering ghost nodes

A detailed discussion and the equations for the penultimate nodes based on these two approaches are presented in Appendix A.1.

A linear system of equations of form

$$\mathbf{A\Phi} = \mathbf{b}' \tag{3.23}$$

is thus obtained where, $\mathbf{A}$ (dimension of $n \times n$) is the coefficient matrix with positive diagonal entries (except in certain special cases), $\mathbf{\Phi}$ and $\mathbf{b}'$ are column vectors of dimension $n \times 1$, n is the number of discrete points considered inside the boundary where the value of ϕ needs to be determined. In the five point stencil, since the central node is dependent only on its adjacent nodes, the matrix $\mathbf{A}$ is extremely sparse and poorly conditioned. Upon introducing the boundary conditions, the known right hand side vector $\mathbf{b}'$ has contribution from the boundary conditions as well, particularly for the penultimate nodes and thus takes the form

$$\mathbf{b}' = \mathbf{b} - \mathbf{b}_{DBC} - \mathbf{b}_{NBC} \tag{3.24}$$

where, $\mathbf{b}$ contains the value of the source term as defined in eq.3.22, $\mathbf{b_{DBC}}$ is the value of ϕ at the boundary multiplied with p or r in case of DBC and $\mathbf{b_{NBC}}$ are

the terms under the column "Known value" in the tables of Appendix A.1. The form of $\mathbf{b_{DBC}}$ and $\mathbf{b_{NBC}}$ is such that they will have all entries as zero except for the equations corresponding to the penultimate nodes. It is to be noted that when boundary conditions are all Dirichlet, $\mathbf{A}$ remains symmetrical. However, if instead of all DBC, if there is DBC at some boundaries and NBC or CBC at other boundaries, the coefficient matrix $\mathbf{A}$ becomes asymmetric because the equation structure as shown in eq.3.22 changes to that shown in Appendix A.1. Also, it is interesting to note that if all the boundary conditions are Neumann, $\mathbf{A}$ becomes singular implying that it can have infinitely many solutions. This behaviour is expected because it means the constant term in the function ϕ is not known at any of the boundaries and thus instead of an unique solution multiple solutions will exist. Hence, it is essential to have at least one boundary condition as Dirichlet to arrive at an unique solution for the linear system.

3.5.1 Red-Black colouring scheme

The **red-black colouring scheme** (Saad, 2003) is a type of multi-colouring ordering which is useful to write numerically parallel algorithms when the central node is dependent on the adjacent nodes only. This section discusses in detail this scheme and how it can be useful to handle a large domain.

It is evident from the structure of eq.3.22 that the value of $\phi_{i,j}$ is dependent on its immediately adjacent nodes, $\phi_{i-1,j}, \phi_{i+1,j}, \phi_{i,j-1}, \phi_{i,j+1}$. Bearing this fact in mind and following Algorithm 4 the discretized domain as shown in Figure 3.4 is obtained for a 2D domain. This means that a red node is independent of all other red nodes and a black node is independent of all other black nodes. Now, suppose a quantity ϕ needs to be computed at any red node. At j^{th} step of iteration of the iterative algorithms like Algorithm 2 or Algorithm 3, ϕ can be computed for all red nodes in one shot.

For instance, consider nodes 5 and 15 in Figure 3.4. If sequential algorithms were used i.e. where reordering has not been implemented, node 15 will have to wait for computation of node 5 to be completed. In other words, all the nodes remain coupled to one another and hence, 5 and 15 could not be updated simultaneously. However, with red-black reordering, the equations are decoupled. After all the values of red nodes are computed, values of all the black nodes are computed in one

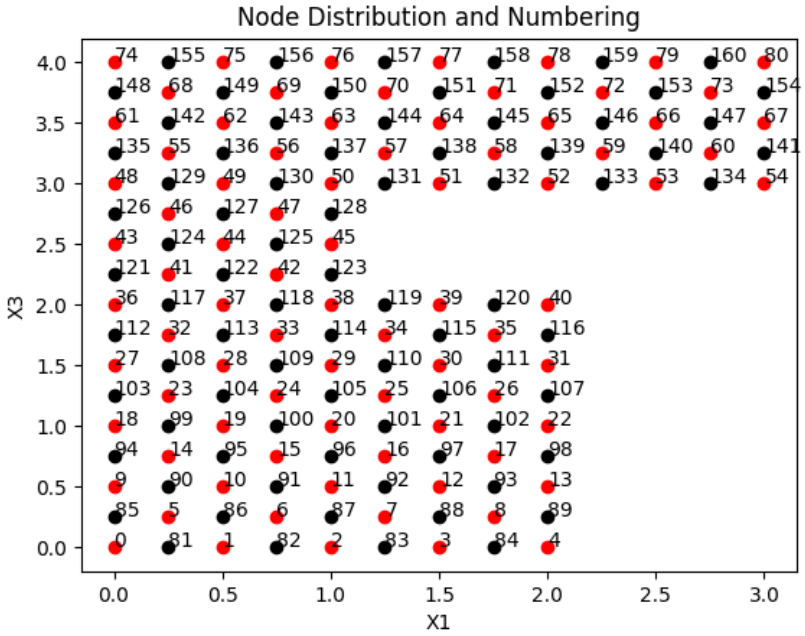


Figure 3.4: Red-black coloring scheme in a 2D domain

go. This process is repeated for every step of the iteration till the solution finally converges.

3.5.2 Setting up the linear system

Using the concepts of red-black coloring scheme introduced in Section 3.5.1 and using the partitioning scheme presented earlier in Figure 3.2, the matrix structure of eq.3.23 can be split into blocks of smaller matrices which in can be expressed in the form

$$\mathbf{A\Phi} = \begin{pmatrix} \mathbf{D}_{red} & \mathbf{F} \\ \mathbf{E} & \mathbf{D}_{black} \end{pmatrix} \begin{pmatrix} \mathbf{\Phi}_{red} \\ \mathbf{\Phi}_{black} \end{pmatrix} = \begin{pmatrix} \mathbf{b'}_{red} \\ \mathbf{b'}_{black} \end{pmatrix} \tag{3.25}$$

where,

$$\mathbf{D} = \begin{pmatrix} \mathbf{D}_{red} & \mathbf{0} \\ \mathbf{0} & \mathbf{D}_{black}, \end{pmatrix}, \mathbf{E} = \begin{pmatrix} \mathbf{0} & \mathbf{0} \\ \mathbf{E} & \mathbf{0} \end{pmatrix}, \mathbf{F} = \begin{pmatrix} \mathbf{0} & \mathbf{F} \\ \mathbf{0} & \mathbf{0} \end{pmatrix}, \mathbf{0} = \text{zero matrix}$$

It is to be noted that matrices $\mathbf{E}$ and $\mathbf{F}$ are padded with zero matrices for compatibility of the dimensions such that $\mathbf{A} = \mathbf{D} + \mathbf{E} + \mathbf{F}$. In linear algebra and numerical analysis, this approach of splitting a large matrix into blocks of smaller matrices is

an efficient way of matrix operation requiring less computational time. As a matter of fact, BLAS depends greatly on such block operations for handling large matrices. As discussed in Section 3.4.2, a good preconditioner is required to make the solution of linear system eq.3.23 converge faster. For the purpose of the work carried out on elliptic PDEs in this thesis, this linear system uses PCG if all the boundaries have DBC and PBiCGStab if at least one boundary has NBC.

SSOR preconditioner, with $\omega = 1$ (not to be confused with vorticity, $\boldsymbol{\omega}$) has been implemented (Saad, 2003). With $\omega = 1$, SSOR preconditioner reduces to Symmetric Gauss-Seidel (SGS) as shown in eq.3.26.

$$\begin{aligned}
&\mathbf{M}_{SSOR} = (\mathbf{D} + \omega\mathbf{E})\mathbf{D}^{-1}(\mathbf{D} + \omega\mathbf{F}) \\
\Rightarrow &\mathbf{M}_{SGS} = (\mathbf{D} + \mathbf{E})\mathbf{D}^{-1}(\mathbf{D} + \mathbf{F}) \\
\Rightarrow &\mathbf{M}_{SGS} = \underbrace{(\mathbf{I} + \mathbf{E}\mathbf{D}^{-1})}_{\mathbf{L}}\underbrace{(\mathbf{D} + \mathbf{F})}_{\mathbf{U}} \\
\Rightarrow &\mathbf{M}^{-1} = (\mathbf{D} + \mathbf{F})^{-1}(\mathbf{I} + \mathbf{E}\mathbf{D}^{-1})^{-1} = \mathbf{U}^{-1}\mathbf{L}^{-1}
\end{aligned} \tag{3.26}$$

Simplifying lower triangular matrix, $\mathbf{L}$ and upper triangular matrix, $\mathbf{U}$, the following equations are obtained.

$$\begin{aligned}
\mathbf{L} = \mathbf{I} + \mathbf{E}\mathbf{D}^{-1} &= \begin{pmatrix} \mathbf{I}_{nr\times nr} & \mathbf{0}_{nr\times nb} \\ \mathbf{0}_{nb\times nr} & \mathbf{I}_{nb\times nb} \end{pmatrix} + \begin{pmatrix} \mathbf{0}_{nr\times nr} & \mathbf{0}_{nr\times nb} \\ \mathbf{E}_{nb\times nr} & \mathbf{0}_{nb\times nb} \end{pmatrix} \begin{pmatrix} [\mathbf{D}_{red}^{-1}]_{nr\times nr} & \mathbf{0}_{nr\times nb} \\ \mathbf{0}_{nb\times nr} & [\mathbf{D}_{black}^{-1}]_{nb\times nb} \end{pmatrix} \\
&= \begin{pmatrix} \mathbf{I}_{nr\times nr} & \mathbf{0}_{nr\times nb} \\ [\mathbf{E}\mathbf{D}_{red}^{-1}]_{nb\times nr} & \mathbf{I}_{nb\times nb} \end{pmatrix}
\end{aligned} \tag{3.27}$$

$$\begin{aligned}
\mathbf{U} = \mathbf{D} + \mathbf{F} &= \begin{pmatrix} [\mathbf{D}_{red}]_{nr\times nr} & \mathbf{0}_{nr\times nb} \\ \mathbf{0}_{nb\times nr} & [\mathbf{D}_{black}]_{nb\times nb} \end{pmatrix} + \begin{pmatrix} \mathbf{0}_{nr\times nr} & \mathbf{F}_{nr\times nb} \\ \mathbf{0}_{nb\times nr} & \mathbf{0}_{nb\times nb} \end{pmatrix} \\
&= \begin{pmatrix} [\mathbf{D}_{red}]_{nr\times nr} & \mathbf{F}_{nr\times nb} \\ \mathbf{0}_{nb\times nr} & [\mathbf{D}_{black}]_{nb\times nb} \end{pmatrix}
\end{aligned} \tag{3.28}$$

The inverse of matrices $\mathbf{L}$ and $\mathbf{U}$ can be easily computed as

$$\mathbf{L}^{-1} = \begin{pmatrix} \mathbf{I}_{nr\times nr} & \mathbf{0}_{nr\times nb} \\ -[\mathbf{E}\mathbf{D}_{red}^{-1}]_{nb\times nr} & \mathbf{I}_{nb\times nb} \end{pmatrix}; \mathbf{U}^{-1} = \begin{pmatrix} [\mathbf{D}_{red}^{-1}]_{nr\times nr} & -[\mathbf{D}_{red}^{-1}\mathbf{F}\mathbf{D}_{black}^{-1}]_{nr\times nb} \\ \mathbf{0}_{nb\times nr} & [\mathbf{D}_{black}^{-1}]_{nb\times nb} \end{pmatrix}. \tag{3.29}$$

In the above equations, suffixes $r \times r$, $r \times b$, $b \times r$, $b \times b$ denote the dimension

of the matrices where, r, b refer to the total number of the red and black nodes inside the domain (excluding the boundary nodes). As pointed out earlier, since the matrices are not formed explicitly, while implementing the preconditioner to any of the iterative algorithms, it is applied in two steps i.e. first by computing $\mathbf{w} = \mathbf{L}^{-1}\mathbf{A}$ and then by computing $\mathbf{z} = \mathbf{U}^{-1}\mathbf{w}$.

3.5.3 Multi-core simulations

Now that the linear system and the preconditioner is defined, the next task is to carry out the multi-core simulations. This is achieved by splitting the matrix operation across all processors. It can be observed that the steps in the iterative algorithms include several matrix operations like computing the residue, multiplication of different vectors with the preconditioner, $\mathbf{M}^{-1}$ or the coefficient matrix A. One can imagine that if the dimension of $\mathbf{\Phi}$ is substantial, each and every such matrix operation would incur significant computational cost. This is where multi-core simulations using MPI comes into play which can curtail the computational time significantly. For instance, when the residual for red nodes are computed i.e. the operation $[\mathbf{b'}_{red} - \mathbf{D}_{red}\mathbf{\Phi}_{red} - \mathbf{F}\mathbf{\Phi}_{black}]_{nr\times 1}$ is performed, MPI_Scatter can be done across PS cores. This means instead of performing a single operation for a vector of dimension $nr \times 1$ in a single processor, the operation is now equally split into blocks of vectors each of dimension $(nr + n_0)/PS \times 1$ and is computed by each of the total PS cores available at the same instance of time.

Thus, using MPI, the algorithm steps for all red nodes (or black nodes) are not processed sequentially but parallelly. From the above discussion, it is evident that if a single processor was used, it would have taken time, t to process one iteration. But, with multi-core simulations, for PS processors, every iteration step will now take approximate time of t/PS instead. A detailed analysis of the gain in speed is presented later in Chapter 5.

3.6 Application to other PDEs

Other forms of PDEs like parabolic or hyperbolic can be approached in a similar manner. However, since the physical nature of such PDEs are dependent on the direction of propagation, one-sided differencing schemes are a better representation of these PDEs. With Algorithm 4, it can be shown that these kind of PDEs would

require three or more colors. and hence, the red-black coloring scheme can no longer be implemented. For the purpose of the work carried out in this thesis, this aspect with more than two colours has not been explored in greater detail. As such, in this thesis, when PDEs of form other than elliptic are encountered, preconditioners have not been used and instead Algorithm 1 has been used for multi-core simulation using the approach mentioned in Section 3.5.3.

Chapter 4

Modelling laminar flow with vorticity

Much work on the theories for the analysis of wind turbines had been done. With the availability and easy access to more powerful computational resources, CFD has become extensively popular. It is not unknown that these CFD models can become extremely complicated and difficult to handle for domains as large as wind farms. Hence, there is still scope for further improvement to these CFD models. The reason for discussing the concepts of parallel numerical analysis and multi-core simulations in the previous chapter will become clear from their applications in this and the subsequent chapters.

But first, before going into these details further, a couple of classical concepts need to be introduced in the context of wind farms and fluid dynamics. As such this chapter begins with a brief introduction to the classical actuator disc theory and the potential flow problem before moving ahead into more intricate details and the new proposed model.

4.1 Actuator Disc Theory

Froude, 1889 proposed the actuator disc theory which is still the most simple and widely accepted theory in the field of wind energy. The theory works on the concept of an energy extracting actuator submerged in a fluid (refer Figure 4.1 and

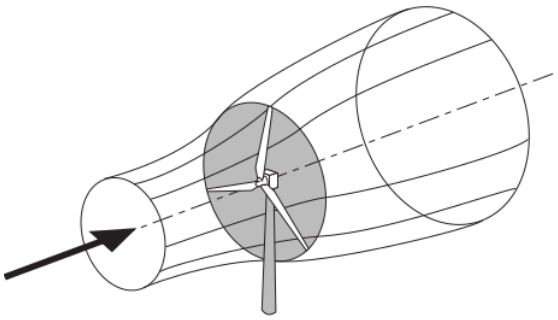


Figure 4.1: The energy extracting stream-tube of a wind turbine (Adapted from Burton et al., 2011)

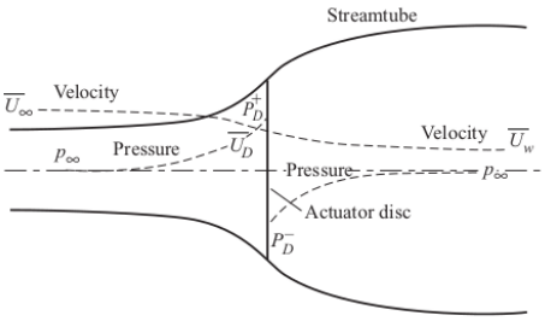


Figure 4.2: An energy extracting actuator disc and stream-tube (Adapted from Burton et al., 2011)

Figure 4.2).

The reason for assuming an expanded wake in these figures is to maintain the law of conservation of mass due to a reduction in the velocity of the wind since the kinetic energy has been extracted from it by the actuator disc. It is also assumed that there is no inflow into or outflow from the domain either from the top or the bottom boundary. In other words, the vertical component of velocity on both the top and the bottom edges of the domain is 0. Further, the presence of the actuator disc itself in the fluid, introduces a velocity drop in the free stream velocity which is given by $-a\overline{U}_\infty$, where a is referred to as the **axial induction factor** and $\overline{U}_\infty$ is the mean free stream velocity. Accordingly, the mean velocity at the face of the actuator changes to

$$\overline{U}_D = \overline{U}_\infty(1-a). \tag{4.1}$$

At this stage, it is important to note that the discussion carried out in this chapter is confined to the mean velocity only, the reason being the actuator disc theory considers the streamtube which implies the existence of laminar flow. Thus, the fluctuating or the turbulent component for the time being is considered to be 0 but a detailed discussion on this will be taken up later in the final chapter of this thesis. In Figure 4.2, applying Bernoulli's equation at a section upstream and downstream of the actuator gives

$$\frac{1}{2}\overline{U}_\infty^2 + \frac{p_\infty}{\rho} = \frac{1}{2}\overline{U}_D^2 + \frac{p_D^+}{\rho} \tag{4.2}$$

$$\frac{1}{2}\overline{U}_W^2 + \frac{p_\infty}{\rho} = \frac{1}{2}\overline{U}_D^2 + \frac{p_D^-}{\rho} \tag{4.3}$$

where, ρ is the density of the fluid, p_∞ is the free stream pressure, $\overline{U}_D$ and $\overline{U}_W$ are

the mean velocities at the disc location and at the wake respectively and p_D^+ and p_D^- are the pressures at the face and just behind the actuator. Subtracting these equations gives the pressure difference across the disc i.e.

$$(p_D^+ - p_D^-) = \frac{1}{2}\rho(\overline{U}_\infty^2 - \overline{U}_W^2). \tag{4.4}$$

The mass flow rate is the same everywhere along the stream tube and thus

$$\rho A_\infty \overline{U}_\infty = \rho A_D \overline{U}_D = \rho A_W \overline{U}_W$$

where, A_∞ = area of the streamtube far upstream, A_D = area swept by the actuator disc, A_W = area of the expanded streamtube behind the actuator. There is a change in momentum as the free stream velocity changes from $\overline{U}_\infty$ to $\overline{U}_W$ which is given by

$$\text{Rate of change of momentum} = (\overline{U}_\infty - \overline{U}_W)\rho A_D \overline{U}_D \tag{4.5}$$

The thrust, T or the force acting on the disc is the difference in pressure between the disc faces multiplied by the area of the disc i.e.

$$\begin{aligned} T = (p_D^+ - p_D^-)A_D &= (\text{Rate of change of momentum})A_D \\ &= (\overline{U}_\infty - \overline{U}_W)\rho A_D \overline{U}_D \\ &= (\overline{U}_\infty - \overline{U}_W)\rho A_D \overline{U}_\infty (1-a). \end{aligned} \tag{4.6}$$

Substituting the pressure difference from eq.4.4 in eq.4.6, the wake velocity can be obtained as

$$\begin{aligned} \frac{1}{2}(\overline{U}_\infty + \overline{U}_W) &= \overline{U}_\infty (1-a) \\ \Rightarrow \overline{U}_W &= (1-2a)\overline{U}_\infty. \end{aligned} \tag{4.7}$$

Substituting $\overline{U}_W$ back into eq.4.6,

$$T = 2\rho A_D \overline{U}_\infty^2 a(1-a). \tag{4.8}$$

Power can be obtained from the thrust using the equation

$$P = T\overline{U}_D = 2\rho A_D \overline{U}_\infty^3 a(1-a)^2. \tag{4.9}$$

The power coefficient is then defined as

$$C_P = \frac{P}{\frac{1}{2}\rho \overline{U}_\infty^3 A_D} = 4a(1-a)^2. \tag{4.10}$$

The maximum value of C_P can be obtained by setting $dC_p/da = 0$ which gives $a = 1/3$. Thus, $C_{P,max} = 0.593$ is obtained which is commonly referred to as **Lanchester-Betz** limit.
The thrust, T can be non-dimensionalized to give the thrust coefficient, C_T which is defined by

$$\begin{aligned} C_T &= \frac{T}{\frac{1}{2}\rho \overline{U}_\infty^2 A_D} \\ \Rightarrow C_T &= 4a(1-a). \end{aligned} \tag{4.11}$$

It is to be noted that though wind turbines physically consist of two or more blades which rotate about the hub in order to extract the kinetic energy from the wind, none of the equations or derivations shown above takes into the account rotation or even the presence of blades in the extraction of energy from the winds. This is where theories like the rotor disc theory or blade element theory comes into picture which considers the effect of rotation of wind turbine blades and consider the effect of lift and drag forces in order to evaluate the performance of the wind turbine as a whole. However, in this chapter the discussion has been confined strictly to the actuator disc theory only because the main focus of this thesis is on steady state analysis. As such, effects of rotation will not come into picture.

4.2 The Potential Flow Problem

Any fluid flow parameter always has a mean and a turbulent component. However, before looking into these aspects in greater detail, in the context of wind turbines, a study is required into the relatively simpler flow models. One such flow is the potential flow model for flows with zero vorticity where the velocity components are expressed in terms of the scalar potential field. The reason for introducing the potential flow at this stage is its implementation in the context of wind farms in past works which is evident from the research work discussed in the literature review earlier. Therefore, in this section the relevance of the potential flow model in the context of the cases studies undertaken here has been discussed.

From the concepts of fluid dynamics, it is known that potential flow is irrotational and directional derivatives of the potential function φ provide the velocity field in the relevant direction. The velocity component parallel to a co-ordinate axis, X_j in terms of φ is defined by $\overline{V}_j = \varphi_{X_j} = \partial\varphi/\partial X_j; j \in \{1,2,3\}$. The sign convention followed in this thesis is shown in Figure 4.3 . For a 2D irrotational flow, say for instance in $X_1 - X_3$ plane, the vorticity, $\boldsymbol{\omega}$ given by the curl of the velocity field becomes 0 i.e.

$$\boldsymbol{\omega} = \boldsymbol{\nabla} \times \overline{\mathbf{V}} = \left(\frac{\partial \overline{V}_3}{\partial X_1} - \frac{\partial \overline{V}_1}{\partial X_3}\right)\mathbf{j} = 0\mathbf{j} \tag{4.12}$$

where, $\mathbf{i}, \mathbf{j}, \mathbf{k}$ are the unit vectors parallel to X_1, X_2 and X_3 axes respectively. Potential can also be defined in terms of a line integral of the the velocity vector (Waters, 2014) i.e.

$$\varphi(X_1, X_3, t) = \varphi_0(t) + \int_{\mathbf{0}}^{\mathbf{x}} \overline{\mathbf{V}}.d\mathbf{X} = \varphi_0(t) + \int_0^x (\overline{V}_1 dX_1 + \overline{V}_3 dX_3) \tag{4.13}$$

where, $\varphi_0(t)$ is arbitrary (constant for steady potential flow). The simulations carried out in this work are for steady state flow and henceforth, any function referred to is considered to be independent of time, t unless noted otherwise. The path of integration does not govern the value of φ which can easily be proved using Green's Theorem.

$$\begin{aligned} &\oint_C \overline{\mathbf{V}}.d\mathbf{S} = \iint_S (\boldsymbol{\nabla} \times \overline{\mathbf{V}}).\mathbf{n}dA; \mathbf{n}\text{: unit outward vector normal to area dA} \\ \Rightarrow &\oint_C (\overline{V}_1 dX_1 + \overline{V}_3 dX_3) = \iint_S -\left(\frac{\partial \overline{V}_3}{\partial X_1} - \frac{\partial \overline{V}_1}{\partial X_3}\right) dX_1 dX_3 = 0 \end{aligned} \tag{4.14}$$

where, S is the area enclosed by the curve C.

It is known that the potential flow problem is of the form of a homogeneous elliptic PDE (or more popularly a Laplace equation), which in Einstein notation can be represented by

$$\frac{\partial^2 \varphi}{\partial X_i \partial X_i} = 0 \tag{4.15}$$

where, $X_i; i \in \{1,2,3\}$ refers to the axis parallel to the hub of the wind turbine, perpendicular to the hub in horizontal direction and parallel to the wind turbine tower respectively.

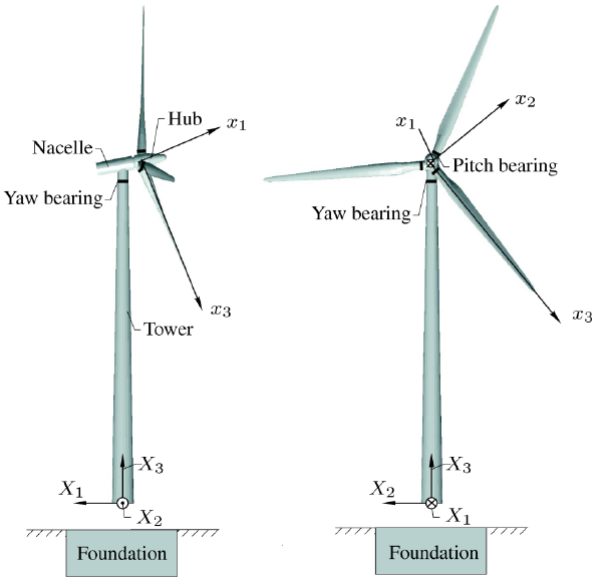


Figure 4.3: Co-ordinate system of three bladed wind turbine (Nielsen, 2017). Simulations presented in this thesis use similar convention but the origin is considered at the upstream side of the overall domain.

4.2.1 Simulation model

For a CFD simulation like the case for the wind farms, ideally a 3D model is best suited. The prime objective of this thesis is to investigate the aerodynamics of a wind farm which inherently signifies the presence of a large physical domain and hence a greater amount of data. However, as the domain becomes larger, greater numerical challenges arise and the simulations become more computationally intensive. Hence, this case needs to be investigated in the light of existing well established analytical models first. But, most of these analytical models are based on 2D analysis. As such, 2D cases are chosen to start with the investigation of these large domains.

The domain can be considered to be a rectangle subjected to a shear flow in atmospheric boundary layer. The origin of the domain can be considered to be at the bottom left corner. Using potential flow model, the first task is to simulate a shear flow and study its effect on wind turbines assuming that the original shear profile is regained after few actuator diameters of the last installed wind turbine. In this thesis, the shear wind profile given by eq.2.1 observed in atmospheric boundary layer has been considered. In the following sections, a detailed analysis of the potential flow model eq.4.15 for the present scenario is undertaken.

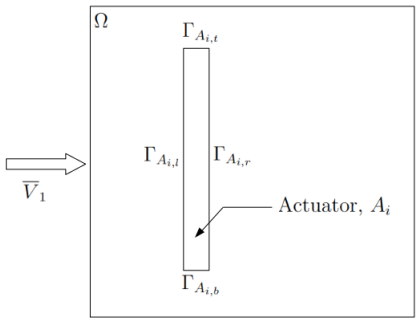


Figure 4.4: Schematic diagram of an actuator disc submerged in a fluid domain

4.2.2 Detailed analysis of the model

In Chapter 3, a detailed discussion on the structure of the discretized Laplace equation has been presented. In this section, the application of that discretized system in the context of the potential flow is undertaken. It is to be noted that the numerical scheme works smoothly as long as the discretized physical domain is continuous. However, introducing the actuator discs into the domain makes it discontinuous which poses certain numerical challenges. Rather than the aerodynamics of the wind field and performance of the wind farms, these challenges are first looked into. But, before moving onto discussing these aspects in greater details, few terms need to be defined.

Let A_i be an actuator submerged in a fluid domain, Ω containing multiple actuators (refer Figure 4.4 where $i \in \{0, 1, 2, ...\}$ denotes the actuator number. Each actuator is bounded by four boundaries denoted by $\Gamma_{A_{i,l}}$, $\Gamma_{A_{i,r}}$, $\Gamma_{A_{i,b}}$, $\Gamma_{A_{i,t}}$ on left, right, bottom and top respectively. The domain is subjected to a mean shear flow of $\overline{V}_1$. The challenges beginning from the geometrical model of the actuator disc to the nature of the boundary conditions and the mathematical nature of the potential flow model itself are discussed below.

1. **Geometrical model of the actuator disc and internal boundary conditions**
   The actuator disc can be modelled in quite a few ways.
   (a) One convenient option to model the actuator disc is to create an interior opening in the domain with height = diameter of the disc and thickness = average width of blade and assign NBC i.e. $\overline{V}_1 = \varphi_{X_1}$ on $\Gamma_{A_{i,l}}$ and $\Gamma_{A_{i,r}}$ and $\overline{V}_3 = \varphi_{X_3}$ on $\Gamma_{A_{i,b}}$ and $\Gamma_{A_{i,t}}$. The values of φ_{X_1} can be assumed

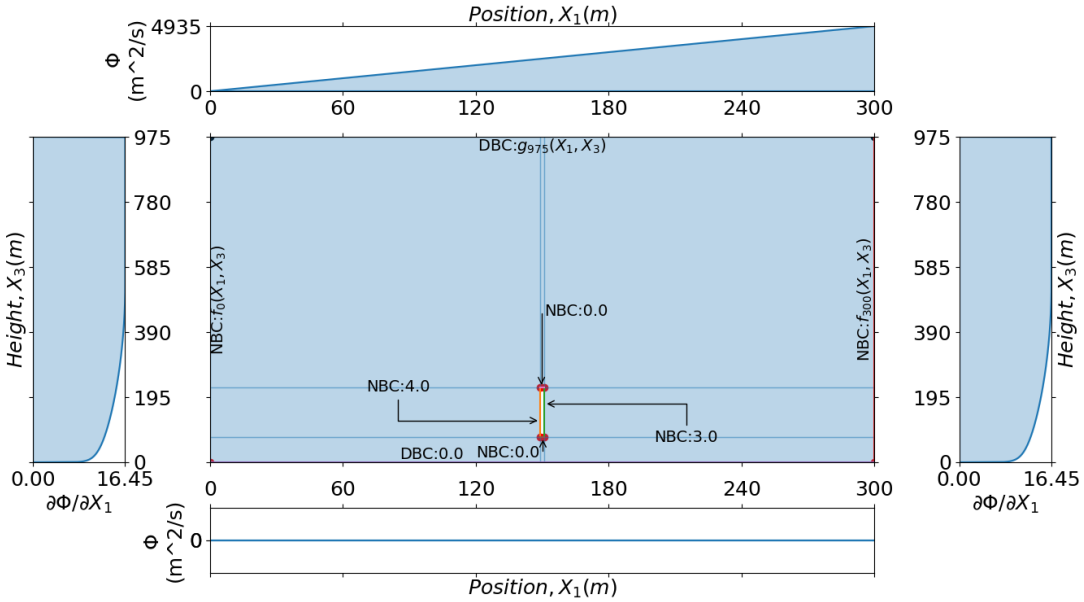


Figure 4.5: Geometrical model with potential, φ as dependent variable

considering a reasonable value for axial induction factor, a. However, φ_{X_3} has to be based on some assumption. An example of this scenario can be seen Figure 4.5 where a single actuator disc submerged in a domain Ω of size 300 $m \times 975$ m is considered with $\varphi_{X_1} = 4$ m/s on $\Gamma_{A_{i,l}}$ and $\varphi_{X_1} = 3$ m/s on $\Gamma_{A_{i,r}}$.

(b) Another option could be to model the actuator as a solid disc of zero thickness with φ_{X_1} and φ_{X_3} in both directions at the grid points. Just like in the previous option, in this case also, φ_{X_3} has to be based on some assumption. But, in this case, another complication arises. Since, the actuator thickness is zero, an abrupt change in the value of φ_{X_1} occurs as the velocity just in front of the actuator and just behind the actuator are different as per the actuator disc theory. Numerically, this would require some sort of smoothening function to mitigate which is altogether a different aspect and requires an in-depth investigation from mathematical perspective.

Nonetheless both these options give rise to an internal closed boundary with all boundaries having NBC. As pointed out earlier in Section 3.5 that as soon as closed boundary with all NBC is encountered, the solution starts to diverge. This requires modelling the discontinuity introduced by the actuator as a FSI problem or alternately ensure that φ is known on at least one interior bound-

ary. This would mean for every actuator within the domain φ is needed to be defined for at least one of its edge in the form of some arbitrary constant or a function. However, the problem is that the relation between the values of φ for each of these actuators is unknown and cannot be determined beforehand.

2. **External boundary conditions**

   As discussed earlier, the actuator disc theory assumes no inflow or outflow from the top and the bottom boundaries. Hence, at first glance, the following boundary conditions for the external boundaries might seem obvious.

   (a) Bottom boundary: $\overline{V}_1\big|_{X_3=0} = \varphi_{X_1} = 0; \overline{V}_3\big|_{X_3=0} = \varphi_{X_3} = 0$
   Integrating both the above equation gives the value of φ

$$\varphi = \int \varphi_{X_1} dX_1 = \int 0 dX_1 = f(X_3)$$
$$\varphi = \int \varphi_{X_3} dX_3 = \int 0 dX_3 = f(X_1)$$

   which is possible if and only if $\varphi = f(X_3) = f(X_1) = k$, where $k =$ constant i.e. DBC can be specified for φ

   (b) Top boundary: The top boundary can be assigned boundary conditions in two ways:

      i. Assigning NBC with $\varphi_{X_3}\big|_{X_3=H} = 0$; where $H =$ overall height of the fluid domain considered.

      ii. Additionally, if it is assumed that at such a far off height, $\overline{V}_1$ ideally remains the same throughout i.e. $\varphi_{X_1}\big|_{X_3=H} = \overline{V}_{1,H}$ (where $\overline{V}_{1,H}$ is the mean velocity at height H), then just like the bottom boundary DBC can be defined in this case as well.

$$\varphi = \int \varphi_{X_1} dX_1 = \int \overline{V}_{1,H} dX_1 = \overline{V}_{1,H} X_1 + f(X_3)$$
$$\varphi = \int \varphi_{X_3} dX_3 = \int 0 dX_3 = f(X_1)$$

      Equating the above two equations give $\varphi = \overline{V}_{1,H} X_1$

   If for top boundary, NBC is assumed, there is no concern. But, if DBC is considered for the top boundary and which is the more accurate condition, the issue is that the relation between $\varphi = k$ (i.e. DBC for bottom boundary) and $\varphi = \overline{V}_{1,H} X_1$ is not known.

   (c) Left and right boundaries: For these two boundaries, since the mean

velocity profile is known, the boundary conditions can be NBC with $\varphi_{X_1}\big|_{X_1=0} = \varphi_{X_1}\big|_{X_1=B} = \overline{V}_1(X_3)$, where B = overall width of the domain.

Now, even if the boundary conditions are assigned as stated above, few other problems still need to be addressed.

(a) Assigning DBC = k at bottom boundary does not ensure that numerical simulation would yield $\overline{V}_3\big|_{X_3=0} = 0$. One way around this is to impose NBC = 0 additionally. In other words, CBC needs to be provided at the bottom boundary.

(b) Similarly, assigning NBC = 0 at top boundary does not ensure $\overline{V}_1\big|_{X_3=H} = \overline{V}_{1,H}$. It just ensures $\overline{V}_3\big|_{X_3=H} = 0$. The only way to propagate both the information $\overline{V}_1 = \varphi_{X_1} = \overline{V}_{1,H}$ and $\overline{V}_3 = \varphi_{X_3} = 0$ into the overall domain numerically is to impose CBC just like for bottom boundary. However, this would required DBC to be defined as well and as discussed earlier, the relation between constants $\overline{V}_{1,H}$ and k is not known.

(c) Finally, the effect of the velocity components tangential to the domain boundary or in other words, the gradient of φ parallel to the boundary has no effect on the solution of the velocity field.

3. **Compatibility at corner nodes**

With only one parameter φ governing the values of both $\overline{V}_1$ and $\overline{V}_3$, another issue is the complexity in satisfying the compatibility at the corner nodes of the domain. The values of φ assumed at the boundary nodes should be such that the value for $\overline{V}_1$ w.r.t. both boundaries forming the corner should also satisfy the requirement of $\overline{V}_3$.

In Figure 4.6 consider a corner node, say node 0. It is known that $\overline{V}_1 = \overline{V}_3 = 0$ at $X_3 = 0$, and hence it needs to be ensured that nodes $0, 18, 1, 21, 6$ have exactly same value of φ (only then by using eq.3.9b and eq.3.9e, $\overline{V}_1 = \overline{V}_3 = 0$ can be ensured). Same criteria is required w.r.t. Node 20 as well. Similarly, for corner node 33, to avoid inflow from top boundary $\overline{V}_3 = 0$ needs to be ensured which means, φ has to be the same for nodes $27, 12, 33$. Same can be said for corner node 17. It is difficult to ensure that all these criteria are satisfied at the same time and numerically these would introduce great many constraints in the problem definition.

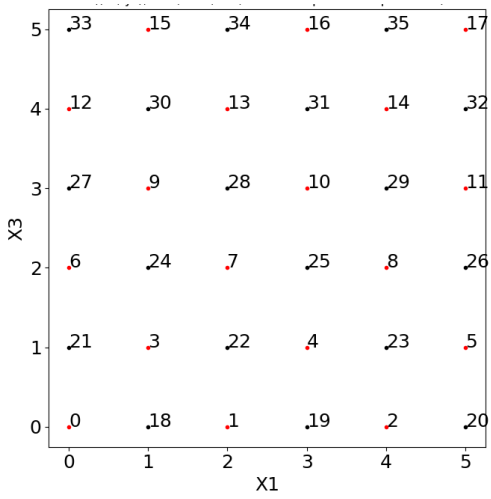


Figure 4.6: Node Distribution

4.3 Model reformulation

Keeping in view the issues presented in the previous section, the potential flow problem needs to be re-framed. The idea presented in this section is influenced by Basu, 2016. The equation of continuity in fluid dynamics which is based on the law of conservation of mass is given by

$$\frac{\partial \overline{V}_1}{\partial X_1} + \frac{\partial \overline{V}_3}{\partial X_3} = 0. \tag{4.16}$$

The vorticity, $\boldsymbol{\omega}$ of fluid in 2D is defined by

$$\boldsymbol{\omega} = \boldsymbol{\omega}(X_1, X_3) = \left(\frac{\partial \overline{V}_3}{\partial X_1} - \frac{\partial \overline{V}_1}{\partial X_3} \right) \mathbf{j}. \tag{4.17}$$

Differentiating the equation of continuity w.r.t. X_1 and the equation for vorticity w.r.t. X_3, the following equations are obtained.

$$\frac{\partial^2 \overline{V}_1}{\partial X_1^2} + \frac{\partial^2 \overline{V}_3}{\partial X_1 \partial X_3} = 0; \quad \frac{\partial^2 \overline{V}_3}{\partial X_3 \partial X_1} - \frac{\partial^2 \overline{V}_1}{\partial X_3^2} = \boldsymbol{\omega}_{X_3} \tag{4.18}$$

Subtracting these two equations, a PDE for $\overline{V}_1$ is obtained.

$$\frac{\partial^2 \overline{V}_1}{\partial X_1^2} + \frac{\partial^2 \overline{V}_1}{\partial X_3^2} = -\boldsymbol{\omega}_{X_3} \tag{4.19}$$

Similarly, differentiating the equation of continuity w.r.t. X_3 and the equation for vorticity w.r.t. X_1, the following equations are obtained.

$$\frac{\partial^2 \overline{V}_1}{\partial X_3 \partial X_1} + \frac{\partial^2 \overline{V}_3}{\partial X_3^2} = 0; \quad \frac{\partial^2 \overline{V}_3}{\partial X_1^2} - \frac{\partial^2 \overline{V}_1}{\partial X_1 \partial X_3} = \boldsymbol{\omega}_{X_1} \tag{4.20}$$

Again, adding these two equations, a PDE for $\overline{V}_3$ is obtained.

$$\frac{\partial^2 \overline{V}_3}{\partial X_1^2} + \frac{\partial^2 \overline{V}_3}{\partial X_3^2} = \boldsymbol{\omega}_{X_1} \tag{4.21}$$

Thus, two elliptic PDEs, with $\overline{V}_1$ and $\overline{V}_3$ as dependent variables are obtained. This new formulation overcomes all the issues highlighted in Section 4.2.2. The advantages of this new formulation are

1. **Geometrical model actuator disc**
   Since, two elliptic PDEs are now available with $\overline{V}_1$ and $\overline{V}_3$ as dependent variables, there is no need to consider NBC at the boundaries because the velocity functions are known at the boundaries straightaway and hence, they can be assigned as DBC resulting in a symmetric coefficient matrix, **A**. Symmetric solvers like PCG can therefore be implemented which are relatively faster in case when just the velocity field needs to be simulated in the absence of the actuators or when the velocity field in the vicinity of the actuator is known.
2. **Boundary conditions**
   Unlike the case with the potential flow model, where some arbitrary value for potential is required to be imposed in order to start the simulations and where CBC is needed to satisfy the criteria for both $\overline{V}_1$ and $\overline{V}_3$ at the boundaries resulting in more complications and making the system more rigid, the current formulation is far more flexible. Also, there is no need to think about the gradient of the dependent variable parallel to the boundaries as it has no physical implication as such.
3. **Compatibility at corner nodes**
   Since two separate PDEs are getting solved and DBC are known at the boundaries straightaway, there is no need to check for compatibility at the corner nodes as such.
4. **A more general model**
   In most practical fluid flow problems, vorticity is inevitable. The cross-section

of the blades of a wind turbine is in the shape of an aerofoil with the chord length gradually decreasing from the root towards the tip. Aerofoils operate on the principal of pressure difference between the top and bottom surfaces with the lift force acting at the bottom and the drag force at the top. However, at the tip of the aerofoil, leakages occur since air flows from lower to the upper side resulting in an increase in tangential velocity. As a result, a continuous sheet of vortices is formed in the wake behind the wind turbine blade. Helmholtz's second theorem states that a vortex line cannot end in a fluid; it must extend to the boundaries of the fluid or form a closed path. The closed path of these vortices forms far downstream as per Prandtl's lifting line theory and eventually these vortices breakdown.

As can be understood, the presence of the aerofoil induces the formation of vortices in the flow field. In other words, the interaction between the air (fluid) and the aerofoil (structure) is responsible for the vortex formation. However, modelling the aerofoil itself or the blade is computationally expensive. The model presented in this work is an attempt to inculcate this fluid-structure interaction by using an approach which is computationally less intensive compared to the standard blade resolved CFD models and without the need for modelling the actual aerofoil but at the same time including its effects by applying appropriate boundary conditions at the interface of the actuator disc and the fluid domain. Hence, this formulation can be applied to a wider range of problems unlike the potential flow which is more specific and applicable for irrotational flow only. As a matter of fact, the current formulation can even address the case of potential flow if $\boldsymbol{\omega}(X_1, X_3) = 0$.

4.3.1 Numerical analysis

Having laid the foundation for the equations of the mean velocity required to describe the full flow field, the next task is to run the simulations for a large domain governed by these newly formulated set of equations. To start with, the flow field for $\overline{V}_1$ is computed first and the behaviour of the model and the numerical simulation is discussed. The reason for choosing $\overline{V}_1$ is that the energy extracted from the wind is primarily dependent on this component of the velocity. In this section, the basis of defining the geometry for the simulation has been discussed first followed by the boundary conditions needed to run the simulation and finally the results of

the simulation carried out has been presented.

4.3.1.1 Geometry

As pointed out in literature review, the overall heights of wind turbines have reached as high as 260 m and blade lengths as large as 75 m. Bearing this fact in mind, the diameter of the actuator discs in this simulation are considered to be 150 m with hub height at 150 m above MSL. For a typical 5 MW wind turbine, at the widest section, the blade can be as large as 4 m (Resor, 2013). Since the blade profile varies throughout its entire length, it can be argued that an average value for the actuator thickness should be chosen for the numerical model. However, it is to be noted that the exact behaviour of the fluid in the immediate vicinity of the actuator is unpredictable. As such, choosing the thickness on a higher side seems to be a good option to bypass this region of uncertainty from the numerical results. Accordingly, the thickness of the discs considered for the purpose of the simulations undertaken is fixed at 4 m. It is to be noted that the disc rotation and its effect in creating a counter-rotating wake is not considered since focus is on the steady state behaviour.

It is a normal practice to position the wind turbines at a distance varying from three to five times the actuator diameter. Thus, centre to centre distance between the discs have been considered to be $5 \times 150 = 750\ m$. Also, the disturbances induced to velocity field because of the presence of the discs are usually considered to be greatly reduced after four-five actuator diameters. Bearing this fact in mind, the downstream domain length beyond the centre of the last actuator disc is considered to be $5 \times 150 = 750\ m$. An additional domain length of $3 \times 150 = 450\ m$ is considered upstream from the centre of the first actuator disc. Thus, the total domain length, L = upstream length + (no. of discs - 1) x c/c distance of disc + downstream length $= 450 + (3 - 1) \times 750 + 750 = 2700\ m$.

In a similar manner, the domain height above the top of the actuator disc is considered to be $5 \times 150 = 750\ m$. Thus, the overall domain height, H = hub height + actuator radius + additional top height $= 150 + 75 + 750 = 975\ m$ The node to node distance has been considered as $\triangle X_1 = 1\ m$ and $\triangle X_3 = 2\ m$. Since, $\triangle X_3$ has been assumed as 2 m, H has been considered as 976 m instead to suit the requirement of the grid spacing. A finer grid or a still larger domain could also be considered. However, for the time being the simulation is done just to highlight some

of the key aspects of the present approach of geometrical modelling by considering the actuator disc as a solid disc.

4.3.1.2 Boundary conditions

In any system defined by differential equations, it is the boundary condition which governs the final solution by directing the generalised solution of the differential equation towards a unique result. The importance of applying proper boundary conditions can also be understood from the fact that two different fluid flow problem can have different solutions even though they are governed by the exact same set of PDEs if their boundary conditions are different. This section discusses the boundary conditions considered for the simulations.

Three back-to-back actuators are considered with the the values of $\overline{V}_1$ defined both at $\Gamma_{A_{i,l}}$ and $\Gamma_{A_{i,r}}; i \in \{0, 1, 2\}$. For the time being, $NBC = 0$ is assumed for $\Gamma_{A_{i,b}}$ and $\Gamma_{A_{i,t}}$. Wind is assumed to blow along the direction of positive X_1 axis. The mean velocity profile as defined by eq.2.1 is assigned at the left inlet. Assuming that this original mean wind profile is regained at the exit of the domain, the right outlet is assigned the same velocity profile. As far as the bottom boundary is concerned, the velocity value has been set as 0 and the velocity at the boundary has been assumed to be the constant throughout.The domain with these boundary conditions is shown in Figure 4.7).

4.3.2 Results

Simulations are now carried out successively first for one, then for two and then for three back-to-back actuators. As highlighted earlier in Chapter 3, elliptic PDEs are unconditionally stable. Hence, as expected, the iterative algorithms converge at a relatively stable pace. In all the three cases, the solutions converged quite nicely within only a few iterations even for a domain with such a large number of nodes. This is reflected clearly in the plot of the residual shown in Figure 4.8.

From Figure 4.9, an interesting thing can be observed i.e. there is zone in the wake of the actuator where the values of $\overline{V}_1$ is extremely low and it can be seen to extend over most of the area. The values are seen to be reasonable for the nodes which are nearer to the outlet boundary. This can be attributed to the fact that the

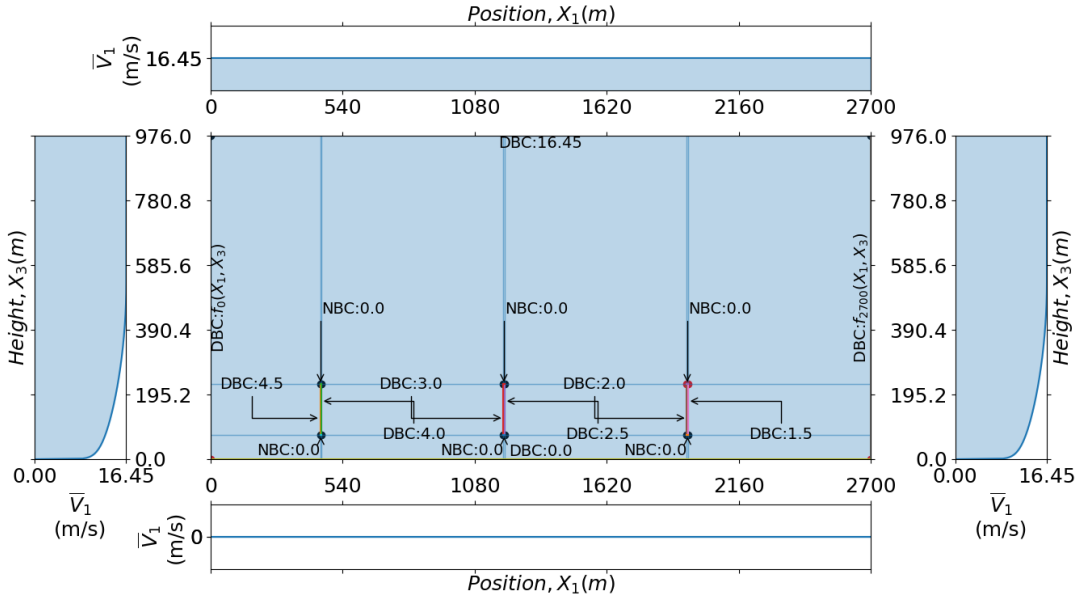


Figure 4.7: Domain and boundary conditions for the newly formulated model

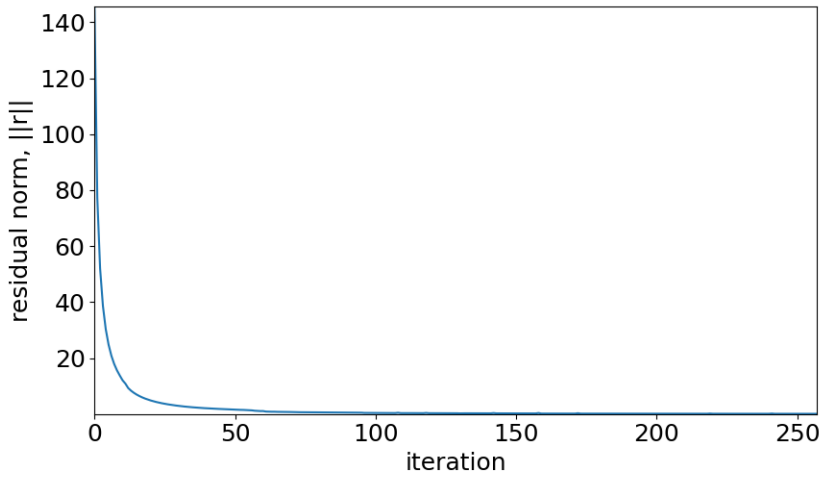


Figure 4.8: Plot of residual norm with increase in iterations for simulation with 3 actuators

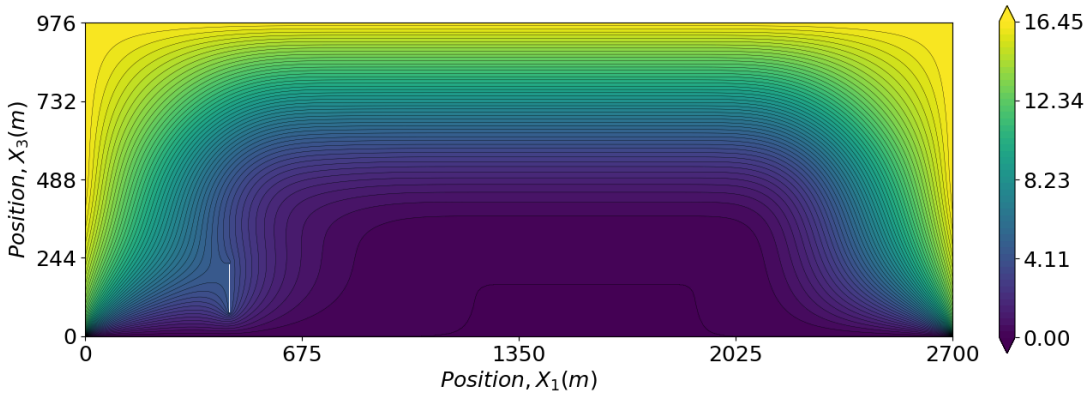


Figure 4.9: Contour plot of $V_1(m/s)$ with 1 actuator disc

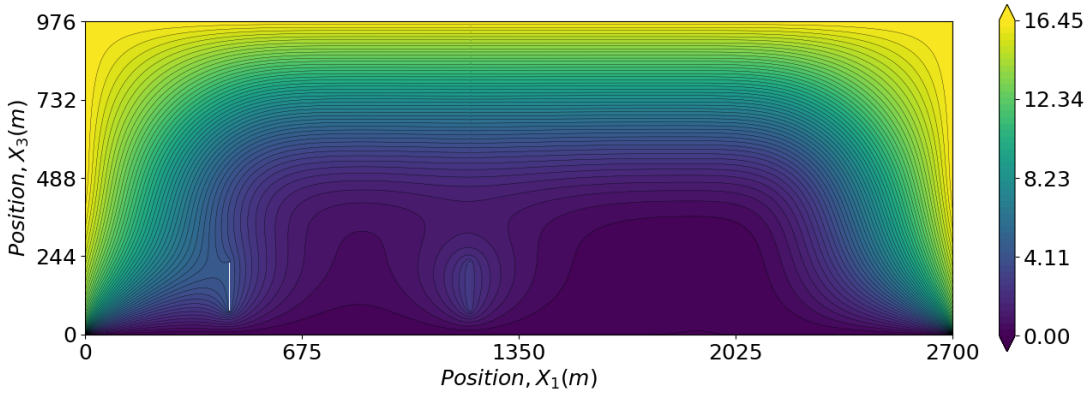


Figure 4.10: Contour plot of $V_1(m/s)$ with 2 actuator discs

actuator disc has been modelled as a solid disc. This behaviour is also evident for the case with two (Figure 4.10) and three actuator discs (Figure 4.11) as well wherein the zones of low velocity can be observed in the wake of the downstream actuators as well. Apart from the regions far away from the actuator in X_3 direction, the regions of non-zero velocity can be observed only for a local region in the vicinity of the downstream actuators which is obvious given the fact that $\overline{V}_1$ (which is forcefully imposed as some arbitrary value) is defined as a boundary condition for these actuators.

Another issue needs to be pointed out. In Figure 4.12, the boundary of the actuator is indicated by the blue line. As per the boundary conditions assumed, DBC is assumed at $\Gamma_{A_{i,l}}$ and NBC is assumed at $\Gamma_{A_{i,t}}$. This would mean node 5435 needs to satisfy two criteria i.e. DBC w.r.t. $\Gamma_{A_{i,l}}$ and NBC w.r.t. $\Gamma_{A_{i,t}}$. Ideally, if

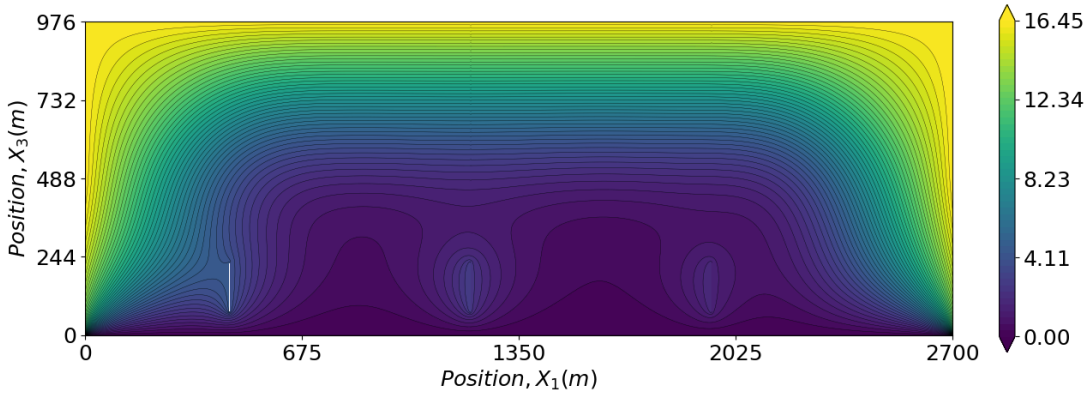


Figure 4.11: Contour plot of $V_1(m/s)$ with 3 actuator discs

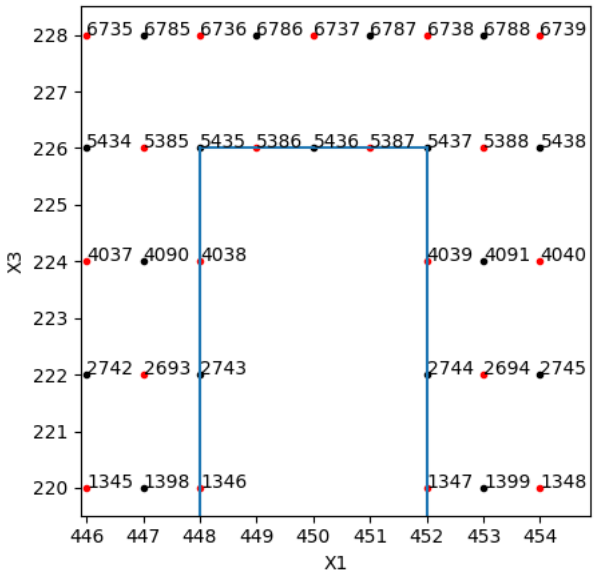


Figure 4.12: Node layout near the top of an actuator

central differencing is used for NBC, this would mean that values at nodes 4038 and 6736 should be the same. Similarly, nodes 4039 and 6738 should also have the same values w.r.t. corner node 5437. With the current setting, it is difficult to ensure that both these conditions are satisfied at the same time. This highlights the fact that DBC needs to be assigned at all the 4 faces of the actuator which is practically not feasible.

With respect to Figure 4.12, another contradiction in the choice of equations for numerical analysis can be observed. Nodes $N_i = \{5386, 5436, 5387\}$, where NBC is defined can be included as part of the equation system getting solved by considering ghost nodes and central differencing as presented in Appendix A. However, this approach is physically consistent only if the behaviour of the ghost nodes and these set of boundary nodes, N_i are governed by the same PDE or rather the same physical model. But, that is not the case in the present approach of modelling the actuator where our assumption is that the behaviour of the zone contained within the boundaries of the actuators is unknown or that the actuator is a solid disc. As such, using one-sided differencing is the only option to tackle the boundary conditions at these nodes. Now, if one-sided differencing is chosen, issue arises with respect to the corner nodes 5435 and 5437 which should ideally be governed by central differencing scheme because it is dependent on nodes which are guided by central differencing schemes. Numerically, for nodes on the same boundary, $\Gamma_{A_{i,t}}$ should all be governed by either central differencing or one-sided differencing. However, as explained corner nodes are getting governed by central differencing and the other nodes by one-sided differencing. This is yet another reason why such physically inconsistent results are observed in terms of extensive zones of low magnitude velocities in the wake of the actuators. From the observations and the points highlighted above, it is evident that this kind of approach to model the computation domain would require DBC to be defined at all the interfaces of the actuators which is practically not feasible.

4.4 Discussions

In this chapter, a brief introduction to the classical actuator disc theory and the potential flow model has been presented. Issues with the 2D potential flow model using FDM in modelling the ABL in the presence of the actuators has been investigated. The potential flow model has been found to be quite problematic in its ability to handle the model accurately. The challenges faced by the potential flow

model has been addressed by introducing a new flow model which also has the form of a Poisson's equation and which not only takes care of the case of potential flow but can also handle any type of fluid flow where vorticity is known. Numerically, this new model does not pose any additional challenge as being elliptic, it is also unconditionally stable.

It is evident from the results of the numerical simulations presented earlier in Section 4.3.2 that the way the actuators have been modelled and the boundary conditions assigned, the velocity profile along all of its bounding surfaces has to be known beforehand which is practically not feasible. As such a more realistic systematic assessment for the $\overline{V}_1$ is required to arrive at a reasonable value.

Besides, in reality the actuator disc is not a solid body but rather a porous object through which the fluid passes. So, even though a converged velocity field is obtained, the results are not a true representation of the physical scenario. Alternately, even if the geometric model of the actuator is visualized as if it is not a solid disc but rather a zone where the behaviour of the velocity is unknown and omitting this zone from the analysis, it cannot be denied that the velocity field within this unknown zone will influence the velocity field getting simulated.

In either case, the approach presented in modelling the actuator is not accurate. Ideally, the interfaces or the surfaces $\Gamma_{A_{i,l}}, \Gamma_{A_{i,b}}, \Gamma_{A_{i,r}}, \Gamma_{A_{i,t}}$ bounding the actuator enclose a zone wherein the behaviour of the fluid is different from that of the fluid outside this boundary. This scenario needs to be taken care of in order to make the numerical model more realistic and consistent with the actuator disc theory. It is not unknown that the solution of two different fluid flow problems might be governed by the same set of equations but their solution might be altogether different depending on the boundary conditions assigned to them. The current case can be visualized in a similar manner wherein the zone outside the bounding surface of the actuator is governed by one set of boundary conditions and the zones confined within each individual actuator (three in the present case) is governed by three other set of boundary conditions. In other words, four different sets of boundary conditions govern the four different zones of the fluid domain but have a common governing PDE with each zone interacting with each other such that all the four sets of boundary conditions and the governing PDE are satisfied together.

From the above discussions, it is imminent that geometrical modelling alone

does not suffice to achieve the required outcome and instead something is required to be done in the numerical approach itself. From the perspective of fluids, each and every zone bounded by the actuator is a zone of discontinuity. As pointed out earlier in Chapter 2 creating a complex mesh around the actuator blades is of course one way but then these are computationally expensive. This is where special FSI methodologies come into picture which are ideally designed to take care of these kind of discontinuities in the fluid domain. In the next chapter, this aspect has been addressed and the numerical simulation for the new formulation presented in this chapter has been carried out in the light of the FSI strategies.

Chapter 5

Fluid-structure interaction for back-to-back actuators

From the discussions in the previous chapter, it is evident that modelling the actuator disc is not a straightforward task. The region bounded within the boundaries of the actuator or rather a zone in and around the vicinity of the actuator behaves quite differently with respect to the rest of the domain and influences the velocity field of the overall domain. As such, the aerodynamics of such discontinuous fluid domains are required to be looked into. This is where FSI comes into picture. Of course, as discussed earlier in Chapter 2, FSI can be carried out by using blade-resolved models but then these models are computationally expensive.

A full three-dimensional (3D) CFD model of the entire wind farm which might consist of hundreds of turbines is computationally expensive both in terms of processing time as well as the memory and the number of processors required. During the preliminary engineering analysis and design stage when detailed site data is not available, investing in such computational resource is not worthwhile. Nonetheless one or more arrangement of the wind turbines in the wind farm is still required to be investigated in order to demonstrate the pros and cons of each arrangement from the perspective of power production. This is where the analytical two-dimensional (2D) models mentioned earlier in Chapter 2 become useful. The primary aim of these models is that during this preliminary analysis stage, they give an insight into the approximate power output of a wind farm without the need to develop the full 3D CFD model. An interesting fact about the analytical models

is that none of them include the effect of the presence of blades. Therefore, the current work aims at proposing an approach so that these models can be further refined to include the effect, the blades might have on the flow field, so that the preliminary estimate of power production from a wind farm using these models are more realistic. The present work is intended to achieve the same by not going into complexities of the aero-elasticity of the blades (Korobenko et al., 2017; Sayed et al., 2019), which of course provide useful information, but rather trying to include its effect geometrically by considering a zone where the behaviour of wind is different compared to the rest of the domain.

The objective of the work carried out in this chapter is to introduce a hybrid approach of using the wind farm models coupled with these FSI strategies in order to arrive at a more realistic velocity field for the new formulation presented in the previous chapter at a relatively low computational cost. However, before implementing FSI algorithms for simulating the wind farms, few concepts relevant to the work done need to be introduced. This chapter, therefore begins with the discussion of one of the earliest and the most basic wind farm models i.e. Jensen's model and the modifications proposed to it, followed by FSI algorithms before carrying out the simulations for back-to-back actuator discs.

5.1 Jensen's model and modifications

In Chapter 2, a detailed review of different wind farm models proposed by researchers have been presented. However, Jensen's model is discussed here as this model has been chosen for the present work. As will be clear from the upcoming discussions that FSI algorithms require certain parameters to be defined which can be easily done with the aid of these wind farm models, thereby making the velocity fields obtained using these models more realistic compared to the strictly analytical results obtained using these theoretical models wherein FSI effects are not getting considered.

Like most fluid flow problems, the ABL also has a mean and turbulent component. The work is based on the fact that the mean velocity in ABL is predominantly responsible for the power production from wind turbines and the contribution from turbulence when compared to the mean velocity is not that much significant. However, turbulence does affect the fatigue life performance of the blades, which however

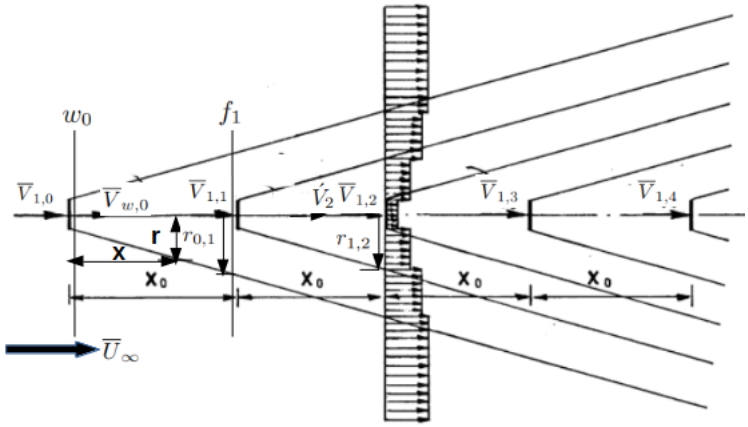


Figure 5.1: Overlapping wakes for back to back actuators as proposed by Jensen (Adapted from Jensen, 1983 with modifications)

is not the focus here. The Reynolds number in the present case is very large i.e. $Re = vD/\nu \sim 10^2 \times 10/10^{-5} = 10^8$ and as such viscosity has no role to play unless the scales of turbulence to be resolved are small enough to investigate the local effects on the blades for detailed analysis. Jensen's model does not consider the effects of either vorticity or viscosity and is based on the mean velocity of ABL.

Though the model was initially proposed for actuators of equal radius by Jensen, the concept can be extended to actuators of varying radius. It is to be borne in mind that the primary objective of this chapter is to study the new formulation proposed in the previous chapter using FSI strategies and check its performance with respect to the original Jensen's model of wind farms with actuators of equal radii. As such, testing the model with respect to a field with varying actuator radii has not been taken up in the current work.

At this stage, few notations are required to be introduced. The radius of each actuators is denoted by r_0. Here, r denotes the overall radius of the expanded wake at any typical section and $r_{i,j}$ denotes radius of the wake at actuator j generated by actuator i. For any typical actuator A_i; $i = \{0, 1, 2, ...\}$, f_i and w_i represent sections just in front and behind the actuator and $\overline{V}_{1,i}$, $\overline{V}_{1,i,d}$ and $\overline{V}_{w,i}$ represent velocity upstream, at and downstream of actuator respectively. $\acute{V}_i$ is the weighted velocity of air entrained from the previous overlapping wakes before approaching A_i. $[\overline{V}_1]_{f,i}$, $[\overline{V}_1]_{w,i}$, $[\overline{V}_1]_{t,i}$ and $[\overline{V}_1]_{b,i}$ represent the jump in velocity or the change in velocity at the front ($\Gamma_{A_{i,l}}$), back ($\Gamma_{A_{i,r}}$), top ($\Gamma_{A_{i,t}}$) and bottom ($\Gamma_{A_{i,b}}$) interface of the wind and the actuator respectively.

Consider the case presented in Figure 5.1 where back-to-back actuators are shown with overlapping wake effects. Jensen used the law of conservation of momentum and proposed a linear wake expansion model for the velocity deficit assuming $r = r_0 + \alpha x$, where, $\alpha(= 0.1)$ is an entrainment constant. Thus, when applied at w_0 and f_1, one obtains;

$$\pi r_{0,1}^2 \overline{V}_{1,1} = \pi r_0^2 \overline{V}_{w,0} + \pi(r_{0,1}^2 - r_0^2)\overline{U}_\infty \tag{5.1}$$

where, $\overline{U}_\infty$ is the mean free stream velocity outside the expanded wake, and $r_{0,1} = r_0 + \alpha x_0$. From eq.4.7, it is known that $\overline{V}_{w,0} = (1 - 2a)\overline{U}_\infty$; where, a is the axial induction factor. Substituting these values of $r_{0,1}$ and $\overline{V}_{w,0}$, $\overline{V}_{1,1}$ in above equation,

$$\begin{aligned}
& r_{0,1}^2 \overline{V}_{1,1} = r_0^2 \overline{V}_{w,0} + (r_{0,1}^2 - r_0^2)\overline{U}_\infty \\
\Rightarrow & \frac{\overline{V}_{1,1}}{\overline{U}_\infty} = \left(\frac{r_0}{r_{0,1}}\right)^2 \frac{\overline{V}_{w,0}}{\overline{U}_\infty} + \left[1 - \left(\frac{r_0}{r_{0,1}}\right)^2\right] \\
\Rightarrow & \frac{\overline{V}_{1,1}}{\overline{U}_\infty} = 1 - k\left[1 - \frac{\overline{V}_{w,0}}{\overline{U}_\infty}\right] \\
\Rightarrow & \frac{\overline{V}_{1,1}}{\overline{U}_\infty} = 1 - k\left[1 - (1 - 2a)\frac{\overline{U}_\infty}{\overline{U}_\infty}\right]
\end{aligned} \tag{5.2}$$

where $k = \left(\frac{r_0}{r_{i,i+1}}\right)^2 = \left(\frac{r_0}{r_0 + \alpha x_0}\right)^2 ; i \in \{0, 1, ...\}$. As stated in the previous chapter, the optimum value of $a = 1/3$ enables an actuator to achieve its maximum power output. So, Jensen based the derivations for the value of $a = 1/3$ i.e. $\overline{V}_{w,0} = \overline{U}_\infty/3$. In other words, the velocity in the wake is one-third of the freestream velocity creating the thrust on the disc. Experimental measurements by Høistrup, 1983 when used to calibrate α produced a value of 0.070. However, Jensen considered $\alpha = 0.1$ as the assumption $\overline{V}_{w,0} = \overline{U}_\infty/3$ is fairly uncertain. In general, for any value of $a = \{a : 0 < a \leq 1/3, a \in \mathbb{R}\}$, eq.5.2 simplifies to

$$\frac{\overline{V}_{1,1}}{\overline{U}_\infty} = 1 - 2ak. \tag{5.3}$$

A similar approach by considering sections w_1 and f_2 would give

$$\pi r_{1,2}^2 \overline{V}_{1,2} = \pi r_0^2 \overline{V}_{w,1} + \pi(r_{1,2}^2 - r_0^2)\acute{V}_2 \tag{5.4}$$

Jensen claimed that $\acute{V}_2 \approx \overline{U}_\infty$ since the spacing between the actuators is usually

large and $\acute{V}_2$ is $a(r_0/\alpha x_0)^2\overline{U}_\infty$ less than $\overline{U}_\infty$; a factor which can be neglected. Thus, the above equation can be written in a manner similar to eq.5.2

$$\begin{aligned} r_{1,2}^2\overline{V}_{1,2} &= r_0^2\overline{V}_{w,1} + (r_{1,2}^2 - r_0^2)\overline{U}_\infty \\ \Rightarrow \frac{\overline{V}_{1,2}}{\overline{U}_\infty} &= 1 - k\left[1 - \frac{\overline{V}_{w,1}}{\overline{U}_\infty}\right] \\ \Rightarrow \frac{\overline{V}_{1,2}}{\overline{U}_\infty} &= 1 - k\left[1 - (1-2a)\frac{\overline{V}_{1,1}}{\overline{U}_\infty}\right] \end{aligned} \tag{5.5}$$

Proceeding in a similar manner, the governing equation for A_i by considering sections w_{i-1} and f_i

$$\pi r_{i-1,i}^2\overline{V}_{1,i} = \pi r_0^2\overline{V}_{w,i-1} + \pi(r_{1,i}^2 - r_0^2)\acute{V}_i \tag{5.6}$$

from which, for A_n, the velocity at the face of the actuator comes out to be

$$\frac{\overline{V}_{1,i}}{\overline{U}_\infty} = 1 - k\left[1 - \frac{\overline{V}_{w,i-1}}{\overline{U}_\infty}\right] = 1 - k\left[1 - (1-2a)\frac{\overline{V}_{1,i-1}}{\overline{U}_\infty}\right] \tag{5.7}$$

The value of a can vary from actuator to actuator. However, in the study carried out in this chapter, the impact of the variation in the value of a has not been taken up and it is assumed that all the actuators have achieved the maximum power output with $a = 1/3$. Now, if the spacing reduces Jensen's model cannot be applied straightaway as the assumption $\acute{V}_i \approx \overline{U}_\infty$ is no longer valid and needs to be modified. For this case, it is reasonable to assume $\acute{V}_{i+1} \approx \overline{V}_{1,i}$ just behind A_i. From eq.5.3, it can be observed that the free stream velocity field on the downstream actuator is $1 - 2ak$ times the free stream velocity field of the upstream actuator. In a similar way, it can be considered that with respect to A_{i+1} onwards, the free stream velocity with respect to A_i is changed from $\overline{U}_\infty$ to $\overline{V}_{1,i}$.

$$\overline{V}_{1,i+1}/\overline{V}_{1,i} = 1 - 2ak. \tag{5.8}$$

This implies, for $n+1$ closely spaced back-to-back actuators in a wind farm,

$$\overline{V}_{1,n+1} = (1-2ak)\overline{V}_{1,n} = (1-2ak)(1-2ak)\overline{V}_{1,n-1} = ... = (1-2ak)^n\overline{U}_\infty. \tag{5.9}$$

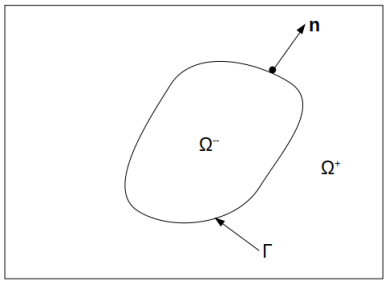


Figure 5.2: Irregular interface Γ dividing overall domain Ω into Ω^+ and Ω^-

5.2 Decomposed Immersed Interface Method

Berthelsen, 2004 proposed Decomposed Immersed Interface Method (DIIM) for elliptic PDEs of form

$$\frac{\partial(\beta\phi_{X_1})}{\partial X_1} + \frac{\partial(\beta\phi_{X_3})}{\partial X_3} = -b(X_1, X_3); (X_1, X_3) \in \Omega \tag{5.10}$$

where, ϕ is the dependent variable and the 2D domain, Ω is defined by $[c, d] \times [e, f]$. Coefficients $\beta = \beta(X_1, X_3)$ and the source term, $-b(X_1, X_3)$ are continuous and smooth on each subdomain enclosed by a boundary. For instance, in Figure 5.2, the values of $\beta(X_1, X_3)$ and $b(X_1, X_3)$ might be different for Ω^+ and Ω^-. From, the equation structure of eq.4.19 and eq.4.21, it is evident that $\beta(X_1, X_3) = 1$ for the simulations undertaken. As such, the discussion here will be confined for this specific value of β and the impact it will have if its value were different at each subdomain will not be looked into. For grid nodes spaced equally, the node to node distance can be defined by $\triangle X_1 = (d - c)/(M - 1), \triangle X_3 = (f - e)/(N - 1)$, where M, N are the number of grid points along axes X_1 and X_3 respectively. Thus,

$$\begin{aligned} X_1 &= c + i \triangle X_1; 0 \leq i < M \\ X_3 &= e + j \triangle X_3; 0 \leq j < N \end{aligned} \tag{5.11}$$

Berthelsen demonstrated the strategy for smaller domains with discontinuities and claimed that it can be extended to other forms of PDEs as well. However, the problems presented in Berthelsen's work were confined to fluids with a single discontinuity and the method has not been explored in greater detail. Hence, the performance of DIIM with multiple discontinuities in the context of a relatively larger domain is validated first before moving on to a domain as large as that of a wind farm. It is to be noted that the IIM strategies (including DIIM) are formulated

for domains discretized using Finite Difference Method (FDM).

5.2.1 A brief introduction

In this section, a brief description of the DIIM strategy has been presented, some of the terms and notations used in the context has been explained and the equations involved are presented before analysing it in greater detail are highlighted. The method is based on a prior knowledge of two main conditions i.e. the change in value of ϕ and the change in the derivative of ϕ along the normal, $\mathbf{n}$ passing through the point of intersection of the gridlines and the fluid-structure interface, Γ (refer Figure 5.2). These changes, often referred to as 'jump' are defined by

$$[\phi] = \lim_{(X_1,X_3)\to\Gamma^+} \phi(X_1, X_3) - \lim_{(X_1,X_3)\to\Gamma^-} \phi(X_1, X_3) = w(X_1, X_3) \tag{5.12}$$

$$[\phi_n] = \lim_{(X_1,X_3)\to\Gamma^+} \phi_n(X_1, X_3) - \lim_{(X_1,X_3)\to\Gamma^-} \phi_n(X_1, X_3) = v(X_1, X_3) \tag{5.13}$$

where, $[\phi]$ = jump in value of ϕ, $[\phi_n]$ = jump in value of derivative in direction $\mathbf{n}$. With reference to Figure 5.2, it can be seen that Γ divides the overall domain, Ω into sub-domains Ω^+ and Ω^- which refer to the region outside and inside respectively of the region enclosed by Γ. This method relies on level set function $\xi = \pm d$, where d is the shortest distance of an irregular node from Γ.

In general, IIM strategies classify the nodes inside a domain as regular and irregular. A node is classified as regular if all its dependent nodes are on the same side of Γ. For instance, with respect to the standard 5-point stencil for elliptic PDEs, if a node lying in Ω^+ lying on the left side of Ω^- and very near to Γ is considered, then its right side node will naturally lie inside Ω^-. As such, this node will be classified as irregular. Similarly, if hyperbolic or parabolic PDEs are looked at, which do not depend on 5-point stencil or even for elliptic PDEs with higher order differencing, a node, N_i will be classified as regular only if the FDE framed for N_i is dependent on a set of surrounding nodes, S such that $\{N_i, S\} \in \Omega^+$ or $\{N_i, S\} \in \Omega^-$ and irregular if $\{N_i \in \Omega^-, S \in \Omega^+\}$ and $\{N_i \in \Omega^+, S \in \Omega^-\}$.

For eq.5.10 and corresponding to $\beta = 1$, the finite difference equation using the standard 5 point stencil can be written as

$$\frac{\phi_{i+1,j} - 2\phi_{i,j} + \phi_{i-1,j}}{\triangle X_1^2} + \frac{\phi_{i,j+1} - 2\phi_{i,j} + \phi_{i,j-1}}{\triangle X_3^2} = -b_{i,j} - C_{i,j}$$

$$\Rightarrow b_{i,j} + C_{i,j} - s\phi_{i,j} - p(\phi_{i+1,j} + \phi_{i-1,j}) - r(\phi_{i,j+1} + \phi_{i,j-1}) = 0. \tag{5.14}$$

It can be observed that unlike structure of eq.3.22, there is an additional term, $C_{i,j}$ in eq.5.14. $C_{i,j}$ is a correction term which DIIM introduces for the irregular nodes at every stage of an iterative algorithm. $C_{i,j}$ in turn consists of components from both directions and is defined by

$$C_{i,j} = C_{i,j}^{X_1} + C_{i,j}^{X_3} \tag{5.15}$$

The term $C_{i,j}^{X_k}; k \in \{1,3\}$ is defined by

$$C_{i,j}^{X_k} = S_\phi \left[\frac{C_1}{\triangle X_k^2} + \frac{C_2}{\triangle X_k} \right] \tag{5.16}$$

where,

$$S_\phi = \begin{cases} -1 & ,\xi_i < 0 \\ 1 & ,\xi_i \geq 0 \end{cases} \tag{5.17}$$

and

$$C_1^{X_k} = \begin{cases} [\phi] - \lambda[\phi_{X_k}]\acute{a} \triangle X_k + \frac{1}{2}[\phi_{X_k X_k}]\acute{a}^2 \triangle X_k^2, & \text{if } (\xi_i \geq 0 \text{ and } 0 \leq \acute{a} < 1/2) \\ & \text{or } (\xi_i < 0 \text{ and } 0 < \acute{a} \leq 1/2) \\ [\phi] + \lambda[\phi_{X_k}](1-\acute{a}) \triangle X_k + \frac{1}{2}[\phi_{X_k X_k}](1-\acute{a})^2 \triangle X_k^2, & \text{if } (\xi_i \geq 0 \text{ and } 1/2 \leq \acute{a} < 1) \\ & \text{or } (\xi_i < 0 \text{ and } 1/2 < \acute{a} \leq 1) \end{cases} \tag{5.18}$$

$$C_2^{X_k} = \begin{cases} \lambda[\phi_{X_k}] + \frac{1}{2}[\phi_{X_k X_k}](1 - 2\acute{a}) \triangle X_k, & \text{if } (\phi_i \geq 0 \text{ and } 0 \leq \acute{a} < 1/2) \\ & \text{or } (\phi_i < 0 \text{ and } 0 < \acute{a} \leq 1/2) \\ 0, \text{otherwise} \end{cases} \tag{5.19}$$

The parameters λ and $\acute{a}$ are defined as

- If Γ lies between $X_{i,j}$ and $X_{i+1,j}$ or $X_{i,j}$ and $X_{i,j+1}$,
  $\lambda = 1, \acute{a} = \xi_i/(\xi_i - \xi_{i+1})$
- If Γ lies between $X_{i-1,j}$ and $X_{i,j}$ or $X_{i,j-1}$ and $X_{i,j}$,
  $\lambda = -1, \acute{a} = \xi_i/(\xi_i - \xi_{i-1})$

where, $\xi_i = \pm d$ is the level set function, d = shortest distance from an irregular node to Γ. It is to be noted that correction $C_2^{X_k}$ is required only if the flux is discontinuous at Γ. The approximation of the correction term, $C_{i,j}$ needs to be

under-relaxed in order to ensure the numerical stability. This is done by introducing an under-relaxation parameter, α.

$$C_{i,j}^{l} = C_{i,j}^{l-1} - \alpha(C_{i,j}^{l-1} - C_{i,j}^{new}) \tag{5.20}$$

where, l and $l-1$ denote the step number of iteration. The simulations presented in this thesis use α within the range $0.01 - 0.05$.

Eq.5.14 can be expressed as a linear system just like eq.3.23 with the additional term $\mathbf{C}$ added to $\mathbf{b}'$, where $\mathbf{C}$ is a vector of $C_{i,j}$ for irregular nodes and 0 for regular nodes. Advantage of DIIM over other IIM methods is that eq.5.14 has the same coefficient matrix as eq.3.23. In other words, the equation for a central node still remains coupled to the immediate surrounding nodes only and not on any additional nodes. This implies that the multicoloring algorithm to achieve parallelism i.e. red-black ordering can still be applied to this new system. However, unlike eq.3.23, $\mathbf{C}$ makes the vector $\mathbf{b}'$ solution dependent at every iteration step. Evidently, iterative algorithms discussed in Chapter 3 cannot be applied straightaway. In the subsequent section, the modifications and choice of algorithm is therefore investigated in greater detail.

5.2.2 Choice of algorithm

As the system matrix is symmetric and positive-definite, the choice of algorithms would normally be the method of conjugate gradients (Hestenes and Stiefel, 1952). However, DIIM introduces a stabilizing correction term $\mathbf{C}$ which changes at each iteration. This did not pose any additional algorithmic challenges in Berthelsen, 2004 because the authors embed the strategy in a stationary iterative method such as Gauss-Seidel. The correction is thus seen as a modification of the correction at the present iteration with no wider algorithmic consequences.

Conversely, the method of conjugate gradients exploits the fact that the residual at each step is the direction of steepest descent for the error, and this direction is then orthogonalized against the most recent residuals to avoid backtracking. This is disrupted when a new DIIM correction is applied to the residual at each iteration. More fundamentally, the application of the correction terms induces an iteration which is for a non-symmetric system, for which conjugate gradients is no longer appropriate.

One way to see this is to interpret the correction to the residual, $\mathbf{r} = \mathbf{b}' - \mathbf{A}\mathbf{\Phi} + \mathbf{C}$ as the result of the application of an unknown, varying right-preconditioner; i.e., $\mathbf{r} = \mathbf{M}^{-1}(\mathbf{b}' - \mathbf{A}\mathbf{\Phi})$. This is what is known as a *flexible preconditioner*. Embedding flexible preconditioning in a conjugate gradients iteration produces a method known as flexible conjugate gradients (Notay, 2000). However, it has been pointed out that the use of the flexible preconditioner interferes with naturally-arising short recurrence and optimality of conjugate gradients because the iteration becomes one for an implicitly-defined non-symmetric, right-preconditioned matrix. Thus, conjugate gradients is replaced with a solver for non-symmetric problems that is compatible with flexible preconditioning to accommodate the correction. The short-recurrence right-preconditioned BiCGStab method van der Vorst, 1992 is chosen, which has been shown to be compatible with flexible preconditioning Vogel, 2007.

Thus, effectively a right PBiCGStab is implemented to carry out the iterations. If the actuator was not present, with this algorithm, the residual, $\mathbf{r} = \mathbf{b}' - \mathbf{A}\mathbf{\Phi}$ would have been updated at the end of each iteration step with the updated vector $\mathbf{\Phi}$. However, to incorporate correction to the irregular nodes, the residual is modified to $\mathbf{r} = \mathbf{b}' - \mathbf{A}\mathbf{\Phi} - \mathbf{C}^{l-1} + \mathbf{C}^{l}$ instead, where l denotes the iteration step.

5.2.3 Validation for single internal discontinuity

The first task is to validate whether parallelization and the approach for residual correction for PBiCGStab using DIIM gives acceptable results. To do so, a commonly studied elliptic PDE problem, $\nabla^2\phi = 0$ (Li and Ito, 2006, Wiegmann and Bube, 2000, Berthelsen, 2004) with $[\phi] = 0$ and $[\phi_n] = 2$ in a square domain $\Omega, [-1\ m, 1\ m] \times [-1\ m, 1\ m]$ and mesh size of 80×80 is considered. The exact solution of this problem is given by eq.5.21. Figure 5.3 show the boundary conditions assigned to the domain. The circle indicates Γ bounding a region within which the behaviour of the fluid is different with respect to the rest of the domain.

$$\phi(X_1, X_3) = \begin{cases} 1 & , X_1^2 + X_3^2 < \frac{1}{4} \\ 1 + \log\left(2\sqrt{X_1^2 + X_3^2}\right) & , X_1^2 + X_3^2 \geq \frac{1}{4} \end{cases} . \tag{5.21}$$

At this stage, it is essential to highlight the way Γ is treated in the numerical simulation. DIIM is based on the interpolation carried out along the normals drawn

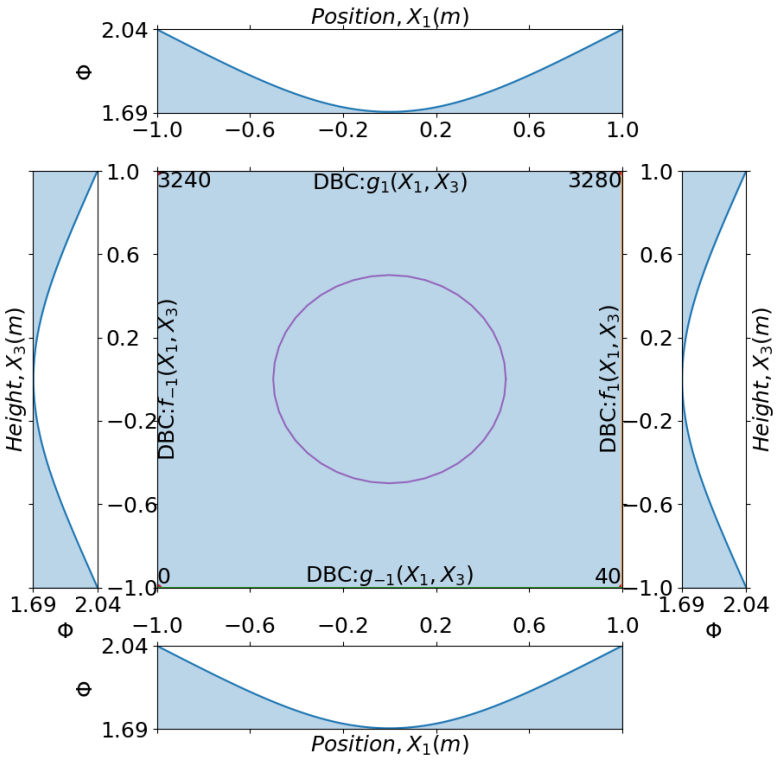


Figure 5.3: Domain with boundary conditions for the DIIM validation problem with single internal discontinuity

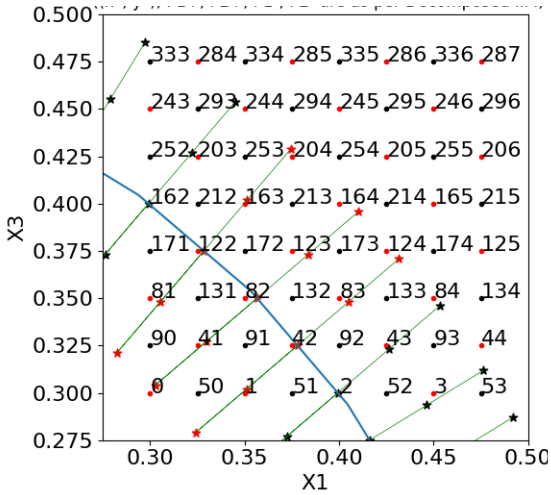


Figure 5.4: Normal at the interface with interpolation nodes to compute $C_{i,j}^{X_1}$

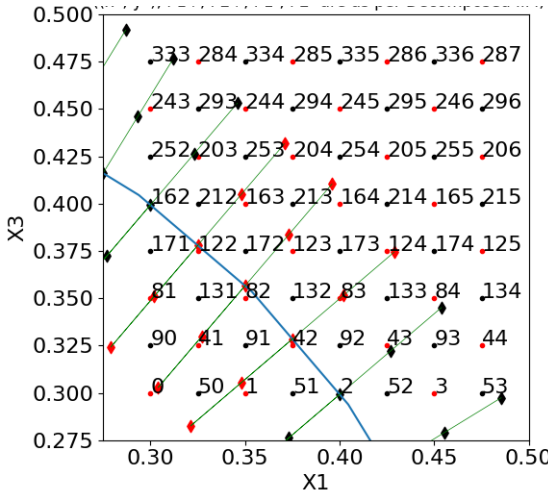


Figure 5.5: Normal at the interface with interpolation nodes to compute $C_{i,j}^{X_3}$

at the interfaces which are indicated by the green lines as shown in Figure 5.4 and Figure 5.5. For instance, in order to compute $C_{82}^{X_1}$, following steps are carried out:-

1. Since, focus is on correction along X_1, the coordinate of the point of intersection, (X_1^*, X_3^*) of gridline parallel to X_1 and passing through 82 and Γ is determined first.
2. The normal, $\mathbf{n}$ passing through (X_1^*, X_3^*) is now determined.
3. Two points are chosen on each side of Γ along the normal i.e. $\{P_1^+, P_2^+\} \in \Omega^+$ and $\{P_1^-, P_2^-\} \in \Omega^-$. For interpolation to work properly, it is essential that these points are positioned in a way such that all the nodes surrounding them are on the same side of Γ. In Figure 5.4, these points are denoted by red $*$ as 82 is a red node.
4. Berthelsen suggested that a second order Lagrangian polynomial of order two is sufficient enough to achieve the desired accuracy. As such, it is vital that $\{P_1^+, P_2^+\}$ or $\{P_1^-, P_2^-\}$ are not surrounded by the same set of grid nodes or else the second order Lagrangian polynomial cannot be formulated. Thus, two polynomials are obtained, one w.r.t. $\{P_1^+, P_2^+\}$ and another w.r.t. $\{P_1^-, P_2^-\}$.
5. The values, $U_1^+, U_2^+, U_1^-, U_2^-$ at $P_1^+, P_2^+, P_1^-, P_2^-$ are computed using these polynomials. Differentiating these two polynomials gives the normal flux which are then utilized to compute U_*^+ and U_*^- i.e. value at (X_1^*, X_3^*) w.r.t. Ω^+ and Ω^- respectively.
6. Finally, these values are used to approximate the jumps along X_1.

In a similar manner, $C_{82}^{X_3}$ can be computed but this time considering the gridline parallel to X_3 passing through node 82 and following a similar set of steps.

Figure 5.6 shows the numerical solution of this problem computed using the modified residual correction with PBiCGStab as explained in Section 5.2.2. The circular interface is modelled approximately as a polygon of 40 sides circumscribed by the circle. As such, two scenarios might arise viz.

1. the normal passes through the side of this polygon: In this case, a line perpendicular to side of the polygon is assumed to be the normal. However, ideally as per the problem statement, the normal should have been the line joining the centre of the circle and Γ. But, since instead of a circle, an equivalent approximate polygon is considered, the normal is chosen as such.

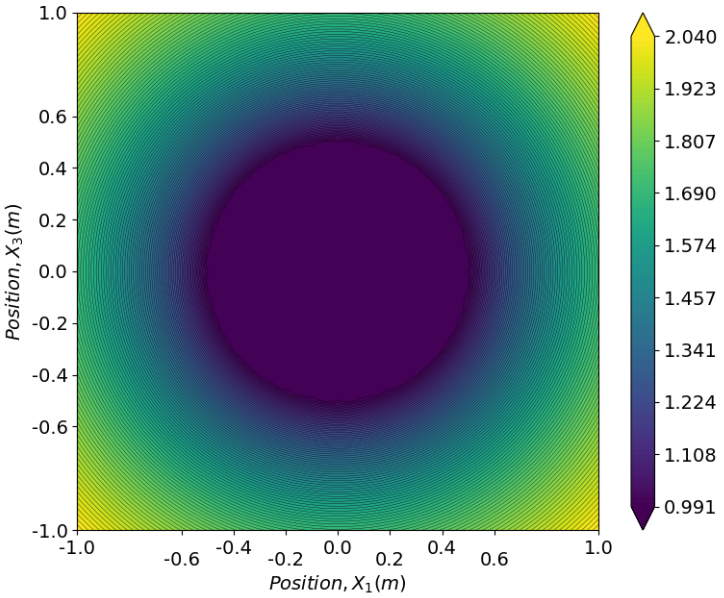


Figure 5.6: Numerical solution for validation

2. the normal passes through the vertex of the polygon: In this scenario, the line joining the centre of the circle and the vertex is considered as the normal.

For the purpose of this simulation, choice of $\alpha = 0.01$ is found to be approppriate for the solution to converge. It is observed that the maximum deviation at a node from the exact solution is only 1.7% which indicates that the modifications proposed to PBiCGStab in order to couple it with DIIM has excellent performance.

5.2.3.1 Convergence and grid refinement analysis

Next a convergence and grid refinement analysis for the problem is carried out before proceeding with the actuator disc. The purpose of this check is to ascertain whether PBiCGStab performs satisfactorily for varying mesh sizes. To check this, the criterion for the iterations to stop is fixed at $OC_\Phi \leq 10^{-13}$ where,

$$OC_\Phi \sim \frac{\left|\,||\mathbf{\Phi}^{l-1}||_2 - ||\mathbf{\Phi}^{l}||_2\,\right|}{||\mathbf{\Phi}^{l-1}||_2} \times 100;\ l = \text{iteration no.}$$

Table 5.1 presents the values of OC_Φ with increasing mesh sizes and various values of α. It can be clearly seen from the table that the value of α determines

Table 5.1: Grid Refinement Analysis

Mesh Size ($\Delta \boldsymbol{X_i}$ (m))	$\boldsymbol{\alpha}$	$\boldsymbol{OC_\phi}$
40×40 (0.0500)	0.01	10^{-13}
80×80 (0.0250)	0.01	10^{-14}
120×120 (0.0167)	0.01	Diverges
	$0.0105 - 0.03$	Converges then Diverges
	> 0.03	Diverges
160×160 (0.0125)	0.01	Diverges
	0.011	Converges then diverges
	> 0.012	10^{-13}
200×200 (0.0100)	0.02	Converges then diverges
	0.03	Converges then diverges
	> 0.04	10^{-13}

the nature of convergence of the solution. Also, the number of iterations taken is dependent on α. Thus, choosing α judiciously is important to ascertain the convergence of the solution. For some cases, it is observed that the solution converges to the correct solution but then starts diverging again. Possible explanation for this could be the excitation of the stable solution by the correction factor as the number of iterations are increased. Thus, instead of OC_Φ, some other parameter needs to be used to decide the point at which the algorithm needs to be terminated in order to get an acceptable solution.

It is observed that L_2 norm of residual vector, $||\mathbf{r}^l||_2$ does not always approach zero even when OC_Φ has approached zero. This happens because the vector $\mathbf{b}'$ is non-stationary and is solution dependent. Thus, instead of OC_Φ, iteration is based on minimization of the value of $||\mathbf{r}^l||_2$. For this validation problem, criterion for iteration to stop is modified to $||\mathbf{r}^l||_2 \leq 0.1(\approx 0)$. Table 5.2 gives the optimal values of α correct to three decimal places with this new criterion. It can be seen that scenarios for solutions diverging, after converging to the correct solutions, are eliminated. It is observed that in some cases $\mathbf{C}^l$ causes $||\mathbf{r}^l||_2$ to remain nearly the same over several iterations but much away from 0 which is indicative of very slow convergence. Also in certain cases, as with the mesh size of 120×120, it is observed that $||\mathbf{r}^l||_2$ stabilizes near a value which is way away from zero. In this case, from Figure 5.7, it can be seen that the values for nodes inside Ω^- has not fully developed. But, if the iterations are allowed to continue, the solution diverges. The reason for this could be attributed to two possible reasons. Firstly, as pointed out by Berthelsen, one drawback of DIIM is that in certain rare scenarios the values estimated at Γ become too inaccurate. This could probably be one such

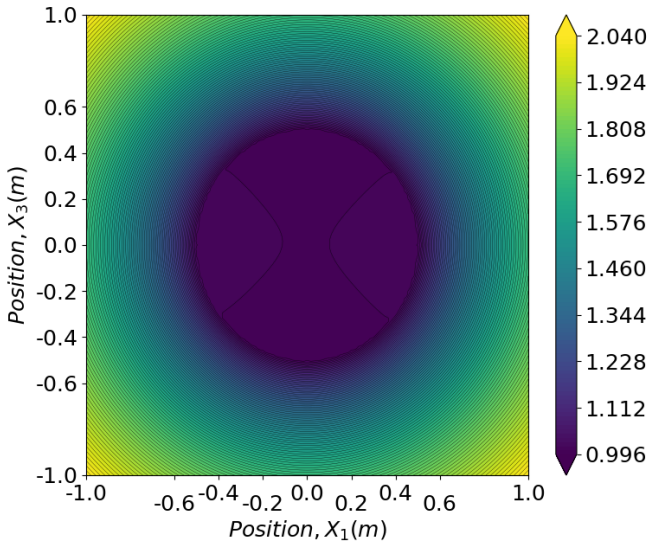


Figure 5.7: Contour plot showing solution not getting developed properly inside Ω^- even after 600 iterations for mesh size of 120×120

Table 5.2: Optimal Values of α

Mesh size	$\Delta \boldsymbol{X_i}$ (m)	$\boldsymbol{\alpha}$
40×40	0.0500	0.01
80×80	0.0250	0.01
120×120	0.0167	$\|\|\mathbf{r}^l\|\|_2$ never approaches 0
160×160	0.0125	0.011
200×200	0.0100	0.018

case. Secondly, the way the circular interface is modelled i.e. modelling it as a polygon instead of an exact circle makes the normals with respect to the polygon slightly different from what they would have been if the actual circle was modelled. However, regardless of this shortfall, the approach is still accurate enough for most of the cases.

5.2.3.2 Performance of PNA

The next important aspect is to investigate the computational efficiency of the algorithm. Table 5.3 shows the time taken per iteration when the iterative algorithm is run serially and on 2, 4 and 6 cores. It is evident that the number of unknowns increase, there is a reduction in the processing time. Of course using the maximum possible number of cores does not essentially ascertain lower processing time when the number of unknowns are less. This can be seen from the case of mesh size of

Table 5.3: Analysis of Computational Time

Mesh Size (ΔX_i)	**Interior Nodes**	**Time (in sec) taken per iteration for**			
		1 core	2 cores	4 cores	6 cores
40×40 (0.0500)	1.521×10^3	$0.20 - 0.22$	$0.20 - 0.22$	$0.20 - 0.22$	$0.20 - 0.22$
80×80 (0.0250)	6.241×10^3	$0.38 - 0.40$	$0.43 - 0.44$	$0.47 - 0.49$	$0.48 - 0.51$
120×120 (0.0167)	1.416×10^4	$0.81 - 0.86$	$0.76 - 0.79$	$0.83 - 0.85$	$0.86 - 0.89$
160×160 (0.0125)	2.528×10^4	$1.08 - 1.11$	$1.10 - 1.14$	$1.06 - 1.10$	$1.10 - 1.12$
200×200 (0.0100)	3.960×10^4	$1.38 - 1.45$	$1.38 - 1.40$	$1.33 - 1.38$	$1.42 - 1.48$
AD case: 2700×975 (1)	2.628×10^6	$9.75 - 10.43$	$5.82 - 6.28$	$4.15 - 4.20$	$3.67 - 3.83$

Time taken is for a Intel® Core™i9-8950HK CPU @ 2.90GHz × 12 processor.

80×80 where using more cores in fact increases the processing time because of an increase in the number of communication between processors relative to the number of equations in the linear system. Similarly, for mesh size of 160×160 and 200×200, 4 cores are seen to be optimal. From Chapter 4, it can be seen that a typical size of the domain for a wind farm considered for this thesis is as large as 2700×975. With the size of discretization adopted the number of nodes is as large as 10^6 Evidently for each and every matrix operation carried out, this operation would incur a significant computation time. It is observed that the processing time is reduced significantly with 6 cores when such a great deal of data is dealt with. Compared to a sequential process , there is almost a reduction of 60% in the time taken per iteration when 6 cores are used. One can imagine how much impact this can have in total time taken for the solution to converge. A plot showing the comparison of CPU time is given in Figure 5.8.

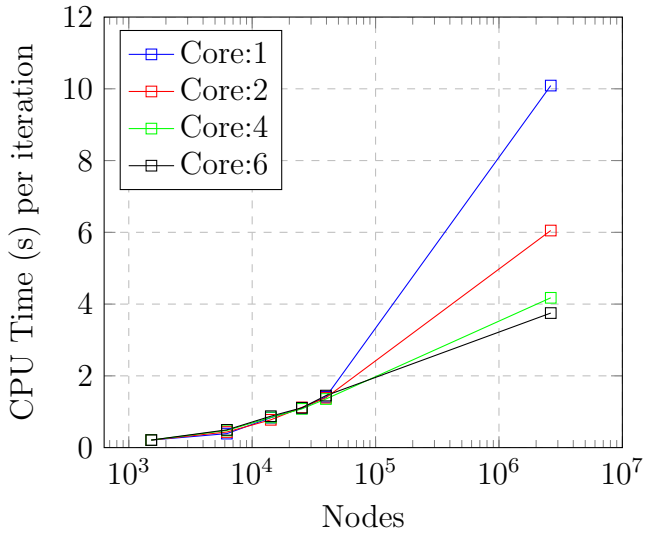


Figure 5.8: Performance comparison for multi-cores

5.2.4 Validation for multiple internal discontinuities

Berthelsen, 2004 demonstrated DIIM for single internal discontinuities only. In the previous section, the numerical results obtained by modifying PBiCGStab for such a case has been validated with respect to analytical results. However, if the approach is to be applied for wind farms, a scenario for multiple internal discontinuities needs to be investigated as well. The usual approach to test the stability and performance of an iterative algorithm is to examine it at two levels viz.

1. Case-1: smaller value of the jumps
2. Case-2: larger value of the jumps

The results of the numerical simulation can be validated with certainty if the analytical solution to the PDE is known beforehand. An analytical solution for ϕ is given by

$$\phi = (c_1 e^{pX_1} + c_2 e^{-pX_1})(c_3 \cos pX_3 + c_4 \sin pX_3) \tag{5.22}$$

where, c_1, c_2, c_3, c_4 are constants is one of the many functions which satisfy eq.5.10 for the case when $b(X_1, X_3) = 0$ (Grewal, 2012). Accordingly, the following analytical function satisfying eq.5.10 is chosen.

$$\phi(X_1, X_3) = \begin{cases} (0.5e^{kX_1} + 1.5e^{-kX_1})(2\cos kX_3 + 7\sin kX_3) & , \{X_1, X_3\} \in \Omega^+ \\ k_i' & , \{X_1, X_3\} \in \Omega_i^- \end{cases} \tag{5.23}$$

where, $k = 0.009$ and Ω_i^- denotes a typical zone of discontinuity inside the global domain. Therefore, the jumps as required by DIIM come out to be

$$\begin{aligned} [\phi]_{\Omega_i^-} &= (0.5e^{kX_1} + 1.5e^{-kX_1})(2\cos kX_3 + 7\sin kX_3) - k_i' \\ [\phi_n]_{\Omega_i^-, X_1} &= k(0.5e^{kX_1} - 1.5e^{-kX_1})(2\cos kX_3 + 7\sin kX_3) \\ [\phi_n]_{\Omega_i^-, X_3} &= k(0.5e^{kX_1} + 1.5e^{-kX_1})(-2\sin kX_3 + 7\cos kX_3) \end{aligned}$$

where, $[\phi]_{\Omega_i^-}$ is the jump in value of ϕ along the interface, $[\phi_n]_{\Omega_i^-, X_1}$ and $[\phi_n]_{\Omega_i^-, X_3}$ are the jumps in the derivatives in the direction normal to the vertical and the horizontal interface i.e. along X_1 and X_3 direction respectively for Ω_i^-,

A domain, Ω of size 150 $m \times 50$ m is considered with node to node distance

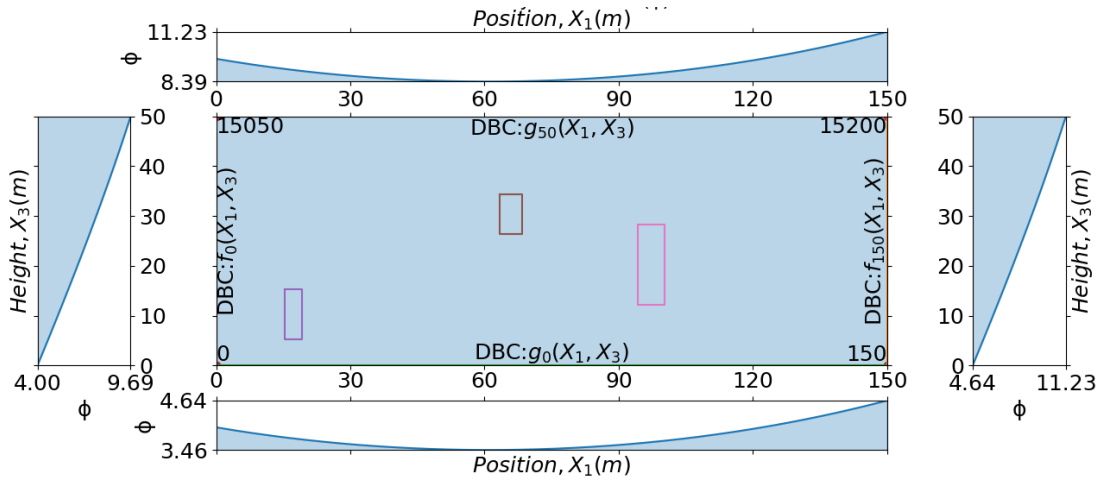


Figure 5.9: Domain with multiple discontinuities

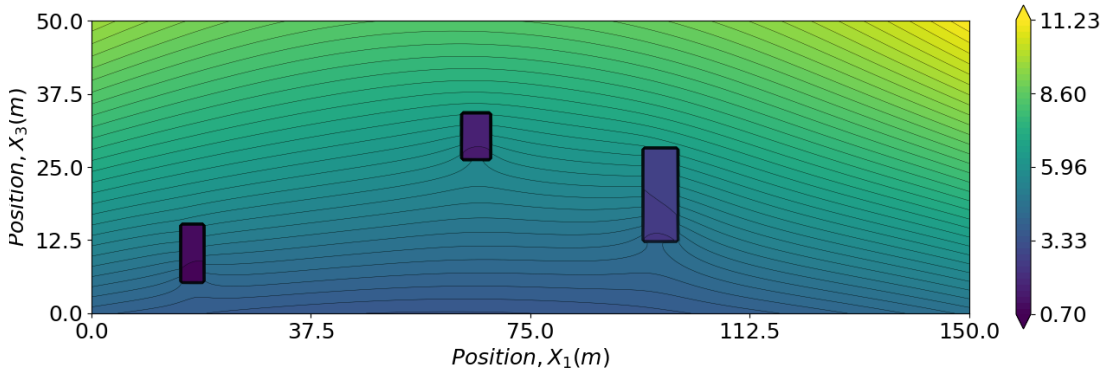


Figure 5.10: Contour plot of numerical values of ϕ for a domain with multiple interior discontinuities and with smaller jump values

of 0.5 m in both the directions and DBC as defined by eq.5.23 is assigned at the external boundaries (refer Figure 5.9). The three coloured rectangles shown inside Ω are the regions of discontinuities denoted by $\Omega_i^-; i = \{0, 1, 2\}$. The internal discontinuities, Ω_i^- are considered at arbitrary locations within the overall domain in order to understand if the spatial distribution of the discontinuities have any adverse numerical implication.

Case-1: To test this case, k_i' is considered as $\{1, 2, 3\}$ for the regions $\Omega_i^-; i = \{0, 1, 2\}$ respectively. From Figure 5.10 and the results presented in Table 5.4, it can be observed that the numerical results show an excellent match with the analytical values. The solution also converged well within 350 iterations which is not substantial and which is expected given the fact that the value of the jumps are small.

Table 5.4: Case-1: Values at a few selected coordinates

Region	X_1	X_3	Theoretical	Numerical
Ω^+	10	3.5	4.2569	4.2149
	10	45.5	8.8643	8.8535
	125	3.5	4.4991	4.5044
	125	45.5	9.3686	9.3831
Ω_0^-	15.5	10.5	1	0.9965
	18	13.5	1	1.0023
Ω_1^-	63.5	30	2	1.6602
	67	32.5	2	1.7173
Ω_2^-	95	24	3	2.6890
	97	17.5	3	2.5886

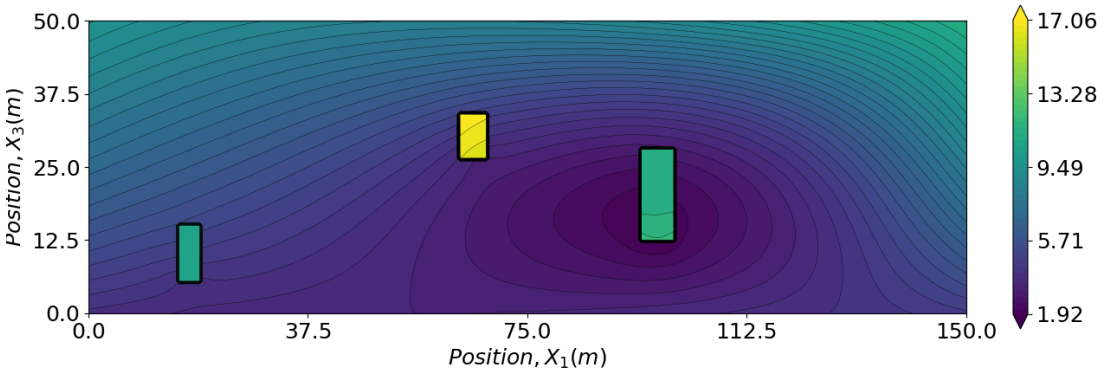


Figure 5.11: Contour plot of numerical values of ϕ for a domain with multiple interior discontinuities and with larger jump values

Case-2: For this case, k_i' is considered as $\{11, 19, 15\}$ for the regions $\Omega_i^-; i = \{0, 1, 2\}$ respectively. From Figure 5.11 and the results presented in Table 5.5, it can be observed that the numerical results are a bit away from the analytical values but they are not too far off implying the validity of DIIM in such a case as well. Also, unlike Case-1, the number of iterations are as large as 2500 in this case which is expected because of a larger difference in the values at the interface.

These two cases clearly suggest that DIIM can perform well in a parallel setting for larger domains with multiple discontinuities using the modification proposed to the residual of PBiCGStab. As such, it is appropriate to say that the approach can be implemented to simulate multiple actuators using the new model proposed in Chapter 4.

Table 5.5: Case-2: Values at a few selected coordinates

Region	X_1	X_3	Theoretical	Numerical
Ω^+	10	3.5	4.2569	4.1851
	10	45.5	8.8643	8.8163
	125	3.5	4.4991	3.9383
	125	45.5	9.3686	8.6598
Ω_0^-	15.5	10.5	11	10.8076
	18	13.5	11	10.7308
Ω_1^-	63.5	30	19	16.7266
	67	32.5	19	16.7345
Ω_2^-	95	24	15	11.2049
	97	17.5	15	11.4500

5.3 Simulation of back-to-back actuator discs

In this section, the simulations carried out for back-to-back actuators is presented. The challenge is that since the wind profile at the face of the downstream actuators are not known, the simulation for the three actuators could not be carried out directly. Jensen's model considers $\overline{U}_\infty$ = constant. However, in the simulations carried out, the focus is on a shear flow i.e. $\overline{U}_\infty = \overline{V}_1(X_3)$ as defined by eq.2.1. The value of $a = 1/3$ is considered throughout. Though both $\overline{V}_1$ and $\overline{V}_3$ represent the full flow field, in this work, the horizontal velocity component, $\overline{V}_1$ has only been simulated, as the power output from an energy extracting actuator disc is dependent mainly on this component.

5.3.1 Geometry

Though the approach is valid for multiple actuators, in this thesis, the study for upto three back-to-back actuator discs each of diameter, $\varnothing = 150\ m$ with a hub height of 149.3 m is presented. The spacing between adjacent actuators is considered to be $5\varnothing$. A zone of $3\varnothing$ is considered in front of the first actuator to allow for the free flow of the wind towards the actuator. An additional zone of $5\varnothing$ is considered downstream of the last actuator following which it is assumed that the initial velocity profile is regained. The height of the domain is fixed by considering a zone of $5\varnothing$ above the topmost point of the disc. Accordingly, the domain size of $2700\ m \times 975\ m$ is considered. Node to node spacing along the X_1 and X_3 axes are considered to be $\triangle X_1 = 1\ m$ and $\triangle X_3 = 2\ m$ respectively, which is a fine enough resolution for such a large domain. As can be seen, the total number of nodes is close to a million.

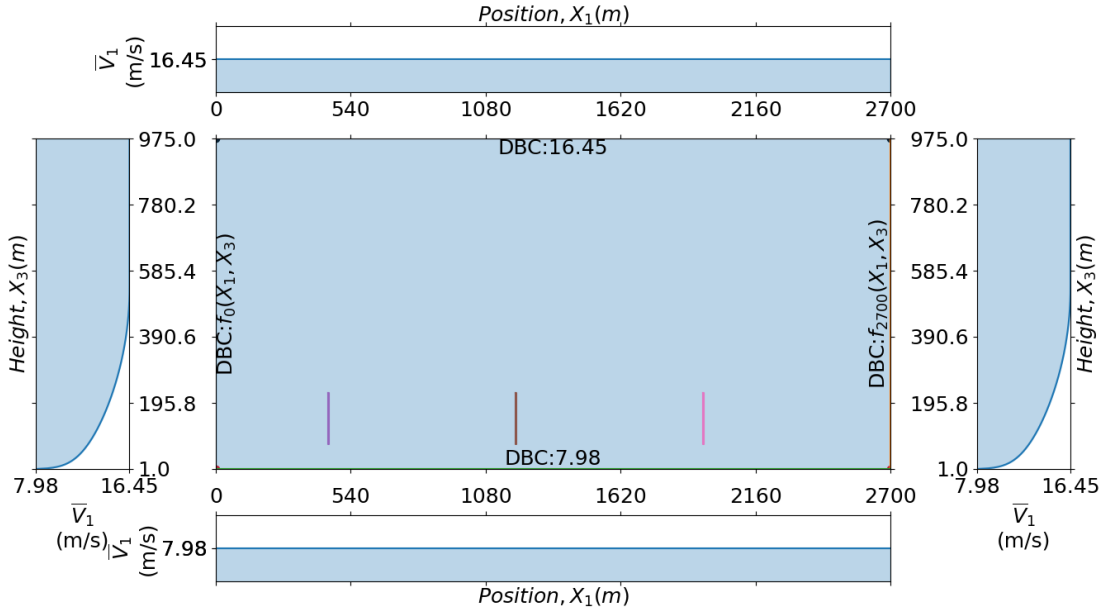


Figure 5.12: Domain showing the positions of the actuator discs and assumed boundary conditions

It is to be noted that the velocity profile changes sign as $X_3 \to 0\ m$ with a discontinuity at $X_3 = 0\ m$. One way to circumvent this issue is to assume the values of $\overline{V}_1$ near to the surface as 0. However, doing so would introduce a discontinuity in the boundary condition which in turn might incur instabilities in the numerical simulation. Instead, the model can be developed ignoring this small zone, the effect of which is practically negligible in such a large domain. Figure 5.12 shows the boundary conditions for the simulations presented in this work. The left and right boundaries of the domain are assigned a velocity as defined by eq.2.1. The bottom and top boundaries are assumed to have constant velocities throughout for $X_3 = 1\ m$ and $X_3 = 975\ m$ respectively obtained by using eq.2.1. Thus, the domain, Ω is defined by the region $[0\ m, 2700\ m] \times [1\ m, 975\ m]$.

5.3.2 Defining jumps

From the perspective of DIIM, the wind turbines can be considered to be the regions of discontinuities in the domain of the wind bounded by the interface, Γ where the change in velocity occurs. It has been discussed earlier in Section 4.3.1.1 that the root width of blades of such large turbines can range from 3m to 4m. Thus, a zone of width of the order of the root width of the blade is considered within which the free

stream velocity changes as defined by eq.5.24 and eq.5.25. The boundary defined by a rectangle of height equal to the diameter of the actuator and width equal to the root width of the blade with its centre located at the hub height from the MSL defines Γ. It is assumed that there is no change in the derivative of the $\overline{V}_1$ in a direction normal to Γ. The jumps for A_i can hence be defined as

1. Case 1: Actuators are spaced far enough such that $\acute{V_i} \approx \overline{U}_\infty$

$$\begin{aligned} [\overline{V}_1]_{f,i} &= \overline{U}_\infty - \overline{V}_{1,i,d} = a\overline{U}_\infty; \\ [\overline{V}_1]_{w,i} &= \overline{V}_{w,i} - \overline{V}_{1,i,d} = -a\overline{U}_\infty \end{aligned} \tag{5.24}$$

2. Case 2: A_{i-1} and A_i are spaced such that $\acute{V_i} \approx \overline{V}_{1,i-1}$

$$\begin{aligned} [\overline{V}_1]_{f,i} &= \overline{V}_{1,i} - \overline{V}_{1,i,d} = a\overline{V}_{1,i} = a(1-2ak)^{i-1}\overline{U}_\infty \\ [\overline{V}_1]_{w,i} &= \overline{V}_{w,i} - \overline{V}_{1,i,d} = -a\overline{V}_{1,i} = -a(1-2ak)^{i-1}\overline{U}_\infty \end{aligned} \tag{5.25}$$

In both the cases $[\overline{V}_1]_{t,i} = [\overline{V}_1]_{b,i} = 0$. As will be seen from the simulation results that this approach of using Jensen's model to evaluate the jumps show a good resemblance with the theoretical values.

5.3.3 Convergence criteria

Iterative approaches require appropriate stopping critera to both detect when the solution has been resolved to a required level of accuracy and also to prevent an onset of divergent behavior due to issues of stability or floating-point arithmetic. For many iterative methods, there are worst-case bounds on the number of iterations based on eigenvalue distribution in the case of a normal matrix (which includes the case of, e.g., symmetric matrices) and additionally non-orthogonality of the eigenvectors in the case of non-normal matrices.

The simulations use BiCGStab method which combines two different approaches (BiCG and restarted GMRES) and is still amenable to some of this analysis. However, the DIIM approach introduces adjustments to the residual which can be interpreted as the action of a dynamically induced implicit "flexible" preconditioner. Thus, an implicit or explicit representation of the system matrix is not available with which to apriori analyze expected convergence behavior or stability. A

more heuristic approach is adopted by studying the residual to ascertain if sufficient convergence has been achieved and to detect any divergence due to instability.

Accordingly, the following criteria has been set for convergence.

- Criteria 1: The value of residual norm, $||\mathbf{r}||_2 \sim 10^{-2}$.
- Criteria 2: The relative value of $||\mathbf{r}||_2/||\mathbf{\Phi}||_2 \sim 10^{-5}$.

where, $||\mathbf{r}||_2$ and $||\mathbf{\Phi}||_2$ are the L2 norms of the residual and the solution vectors respectively. Some supplemental iterations (approx. $100 - 200$) are performed, and it is observed that the residual has stabilized when these criteria are satisfied.

One additional issue one must be cognizant of is that these convergence criteria are related to the discretized problem of algebraic equations which approximate the physical model in question. The convergence tolerances therefore also should not be more stringent than the error induced by the process of modeling reality or the discretization process.

5.3.4 Results

The results of the simulations carried out for back-to-back actuators are now looked into. The challenge is that since the velocity profile at the face of the downstream actuators are not known, the simulation for the three actuators could not be carried out straight away. Jensen's model considers $\overline{U}_\infty$ = constant. But, one of the most important aspect of this work is to demonstrate that DIIM can be applied to any general case and need not be confined to any specific or constant value of $\overline{U}_\infty$. As such, in the simulations carried out, the primary focus is on a shear flow i.e. $\overline{U}_\infty = \overline{V}_1(X_3)$ as defined by eq.2.1, one which is more realistic in ABL. As mentioned earlier, $a = 1/3$ is considered throughout all the simulations.

5.3.4.1 Single actuator

First the simulation is run for a single actuator located at $X_1 = 448.3m$. From eq.5.24, the jumps conditions are

$$[\overline{V}_1]_{f,0} = \overline{U}_\infty - \overline{V}_{1,0,d} = \overline{U}_\infty - (1-a)\overline{U}_\infty$$

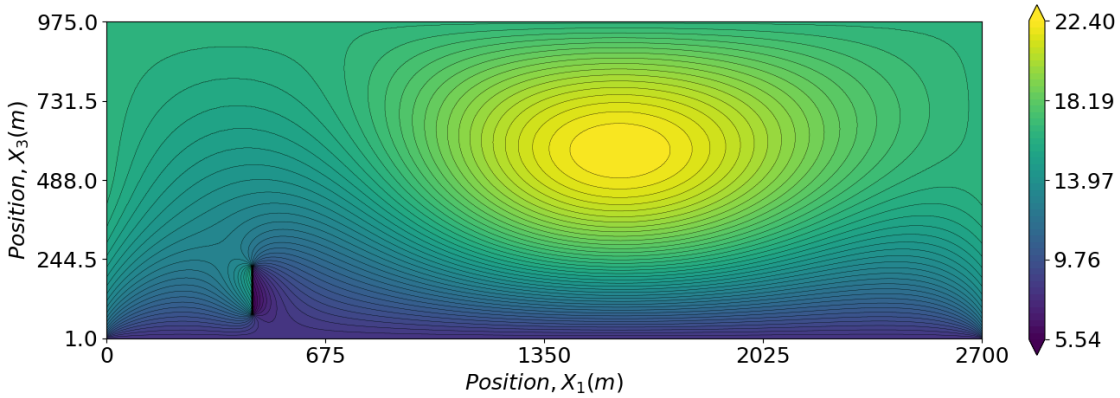


Figure 5.13: Plot of $\overline{V}_1$ (m/s) considering A_0 at $X_1 = 448.3$ m

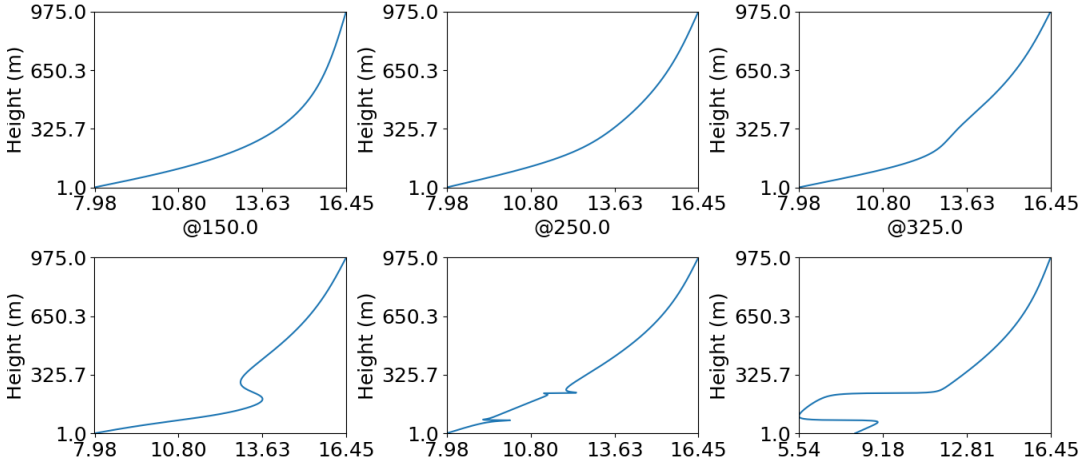


Figure 5.14: Change in velocity profile both upstream and downstream of A_0

$$
\begin{aligned}
&= a\overline{U}_\infty = 1/3\overline{U}_\infty(X_3) \\
[\overline{V}_1]_{w,0} = \overline{V}_{w,0} - \overline{V}_{1,0,d} &= (1-2a)\overline{U}_\infty - (1-a)\overline{U}_\infty \\
&= -a\overline{U}_\infty = -1/3\overline{U}_\infty(X_3). \qquad (5.26)
\end{aligned}
$$

The plot of the velocity field incorporating the effects of FSI is shown in Figure 5.13.

Figure 5.14 shows that the shear profile of the wind is maintained for a reasonable distance. As expected, this profile starts getting affected approximately from $X_1 = 325m$ and the effect becomes more prominent closer to the actuator. At $X_1 = 449m$, that is just after the jump at the interface occurs, drop in the velocity is observed as expected. The value of the velocity averaged over the height, after the drop is around $10m/s$ which is nearly equal to the theoretical value of $\overline{V}_{1,0} = (1-a)\overline{U}_\infty = (2/3)(1/2r_0)\int_{74.3}^{224.3}\overline{V}_1(X_3)dX_3 = 9.28m/s$. Similarly, the average velocity

at the wake i.e. at $X_1 = 453m$ is observed to be approximately $5.5m/s$ which is nearly equal to the average theoretical value of $4.64m/s$. These results validate the approach of DIIM to model FSI in the context of the classical actuator disc theory.

However, an interesting fact is observed from Figure 5.13 i.e. the presence of stationary points in the velocity field and the formation of vortex. Such stationary points are generated by the presence of vorticity. In fact, these points are going to affect the amount of energy extracted from the downstream actuators, a parameter not considered by the existing wind farm models.

5.3.4.2 Two actuators

For the case with more than one actuator, the classical actuator disc theory cannot be applied straightaway as $\overline{U}_\infty = \overline{V}_1(X_3)$ is no longer valid. This is where Jensen's model can be put to use. At this stage, two parameters need to be validated first:-

1. Average value of $\overline{V}_1$ obtained numerically in the wake of A_0 compared with Jensen's model
2. $\acute{V}_1 \rightarrow \overline{U}_\infty$ as the distance from actuator A_0 increases

For this purpose, numerical integration using Simpson's rule was carried out to determine the average value of the velocity within the expanded wake at $X_1 = \{650\ m, 950\ m, 1198\ m, 1300\ m, 1500\ m, 1900\ m\}$. The coordinates of the lower and upper limits of the expanded wake i.e. the limits within which the numerical integration is carried out to determine the average velocity is shown in Table 5.6.

Table 5.6: Average value of velocity as distance from actuator increases

$X_1(m)$	r	$X_{3,l}$	$X_{3,u}$	Average velocity (m/s)		
				$\overline{V}_J$	$\overline{V}_{JC}$	Numerical
650	95	54.3	244.3	8.10	–	9.95
950	125	24.3	274.3	10.54	–	11.50
1198	149.8	−0.5	299.1	11.55	11.57	12.73
1300	160	−10.7	309.3	11.84	12.25	13.19
1500	180	−30.7	329.3	12.26	13.40	14.02
1900	220	−70.7	369.3	12.80	13.40	14.00

$X_{3,l}; X_{3,u}$: Lower and upper X_3 coordinate of wake as proposed by Jensen.

Jensen's work is based on a freely expanding wake. So, for larger dis-

tances from actuator, $X_{3,l}$ tends to become negative which is physically inconsistent. Hence, a correction to the average velocity obtained from Jensen's model is required. This is done by conserving the mass within the expanded wake.

$$2r\rho\overline{V}_J = \rho X_{3,u}\overline{V}_{JC} \Rightarrow \overline{V}_{JC} = 2r\overline{V}_J/X_{3,u} \tag{5.27}$$

where, ρ is the density of air, $\overline{V}_J$ is the velocity for freely expanding wake, $\overline{V}_{JC}$ is the corrected velocity. As $\overline{V}_{JC} \rightarrow \overline{U}_\infty$, this correction is no longer required as it can be assumed that the wake effect has died out and the original velocity has been regained. From the results presented in Table 5.6, it can be observed that

1. average value of $\overline{V}_1$ in the wake of A_0 computed numerically shows a good resemblance with the values obtained from Jensen's model.
2. as the distance from the actuator increases, the value of the average velocity increases such that at $X_1 = 1500\ m$, the value ($14.02\ m/s$) nearly reaches the average value of $\overline{U}_\infty = 13.40\ m/s$, thus, validating the fact that Jensen's model can be considered to assign the jumps for A_1.

Next, two cases are investigated such that A_1 is placed at a distance where

1. average value of $\acute{V}_1 \approx \overline{U}_\infty$;
2. average value of $\acute{V}_1 \approx \overline{V}_{1,1}$

Case-1: A_1 **at** $X_3 = 1500.3\ m$

For this case, the two actuators are considered to be placed far apart from each other such that the wake effect of A_0 on A_1 has minimized to a great extent. Figure 5.15 shows the velocity field for two actuators considering $a = 1/3$ for both A_0 and A_1. A_0 is placed at $X_1 = 1500.3\ m$ considering the fact that $\overline{V}_{1,1} \approx \overline{U}_\infty$, an assumption consistent with Jensen's model for downstream actuators. The numerical simulations show a nice convergence.

Case-2: A_1 **at** $X_3 = 1198.3m$

In this case, when A_1 lies within the wake of A_0, the axial induction factor needs to be examined. It is observed that for $a = 1/3$ that the convergence of the numerical solution is only upto the order of 10^{-4} and the residual does not stabilize. This means that it might not be possible to have $a = 1/3$ inside the wake and the power

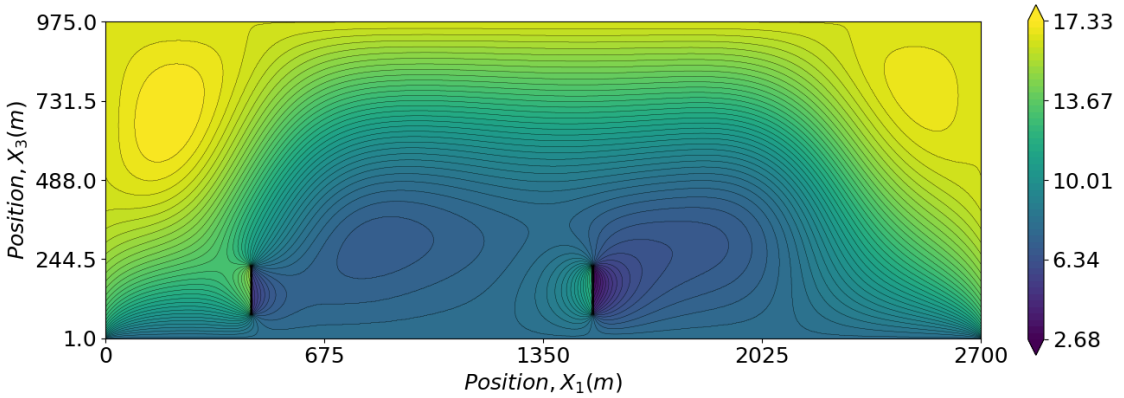


Figure 5.15: Plot of $\overline{V}_1$ (m/s) considering A_0 at $X_1 = 448.3$ m and A_1 at $X_1 = 1500.3$ m

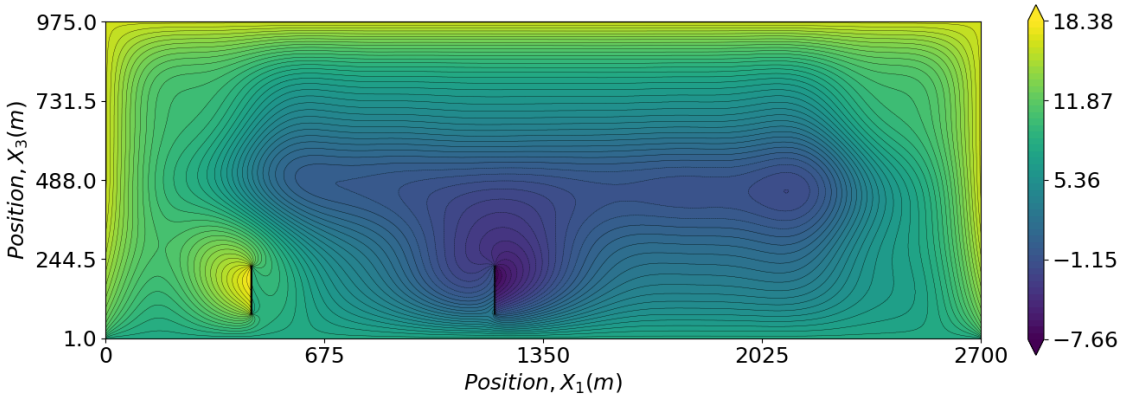


Figure 5.16: Plot of $\overline{V}_1$ (m/s) considering A_0 at $X_1 = 448.3$ m and A_1 at $X_1 = 1198.3$ m

production will be even less. Accordingly, a needs to be modified to an acceptable value. Eq.5.25 gives this value of modified axial induction factor as $a(1-2ak)^{2-1} = 0.27$. Contour plot for this modified value of a is shown in Figure 5.16. As expected, the residual stabilizes with this value of a.

Now when this case is compared with respect to Case-1, it can be clearly seen that zones of low magnitudes of velocity are prevalent and the values of velocity have even become negative implying a change in direction of the flow path.

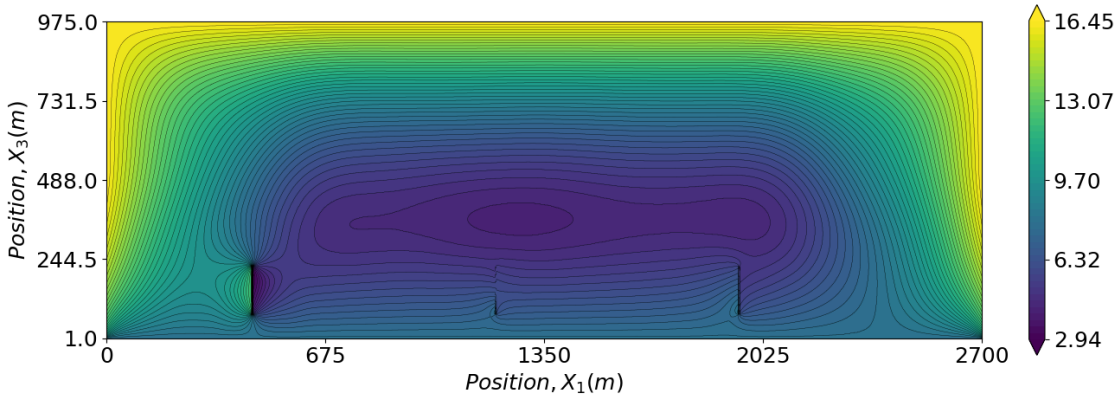


Figure 5.17: Plot of $\overline{V}_1$ (m/s) considering A_0 at $X_1 = 448.3\ m$, A_1 at $X_1 = 1198.3\ m$ and A_2 at $X_1 = 1948.3\ m$

5.3.4.3 Three actuators

Now that the applicability of both classical actuator disc theory and Jensen's model in the context of FSI has been established, the next step is to model a case with three back-to-back actuators in order to check the performance of DIIM for a greater number of actuator discs. As per eq.5.25, the modified axial induction factors for this case are 0.27 for A_1 and 0.23 for A_2. A stable solution is obtained as expected. Zones of low velocity are observed above A_1 and A_2 actuators signifying deficit of wind energy locally.

5.3.4.4 Comparison with FLORIS

The NREL's FLOw Redirection and Induction in Steady- state (FLORIS) model (NREL, 2021) describes the steady-state properties of wakes in wind farms. It was developed to optimize wind turbine control settings and turbine positions, taking into account the effect of wakes on downstream turbines. To optimize the control settings, the model includes the influence of pitch, rotor speed, and yaw settings on the wake's steady-state speed and direction. It is presented in detail in Gebraad et al., 2016. FLORIS's standard wake model is basically an extension of the Jensen model. In this section, the results of implementing DIIM with respect to a wind farm model modelled using FLORIS is compared.

Figure 5.18 shows the wake structure for each of the four cases dealt with in the previous sections. It can be clearly seen that none of the plots show the effect

of vorticity. The reason being, though Jensen's model provides an outline of the expected wake structure, it does not consider the impact vorticity has on the velocity field in any way i.e. whether the flow is rotational or irrotational is not reflected in the velocity field. Further, the model itself does not take into account the effect the presence of blades will have on the velocity field. The proposed approach, on the other hand is able to capture these details by using eq.4.19 to incorporate vorticity coupled with the simplified FSI approach (DIIM) to incorporate the effect of the presence of the geometry of the blades for preliminary analysis. This can clearly be observed by comparing Figure 5.13, Figure 5.15, Figure 5.16, Figure 5.17 with Figure 5.18-I, II, II, IV respectively.

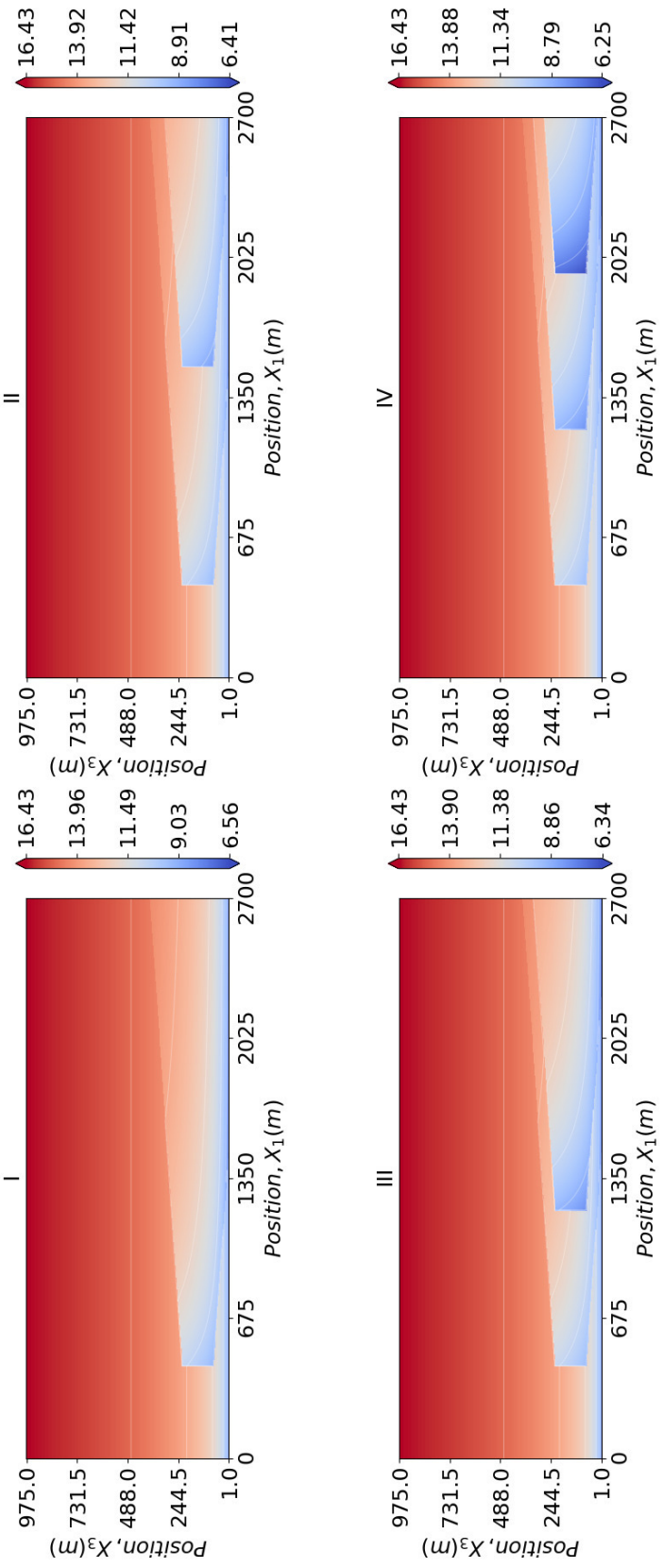


Figure 5.18: Plot of $\overline{V}_1$ (m/s) using Jensen's model in FLORIS

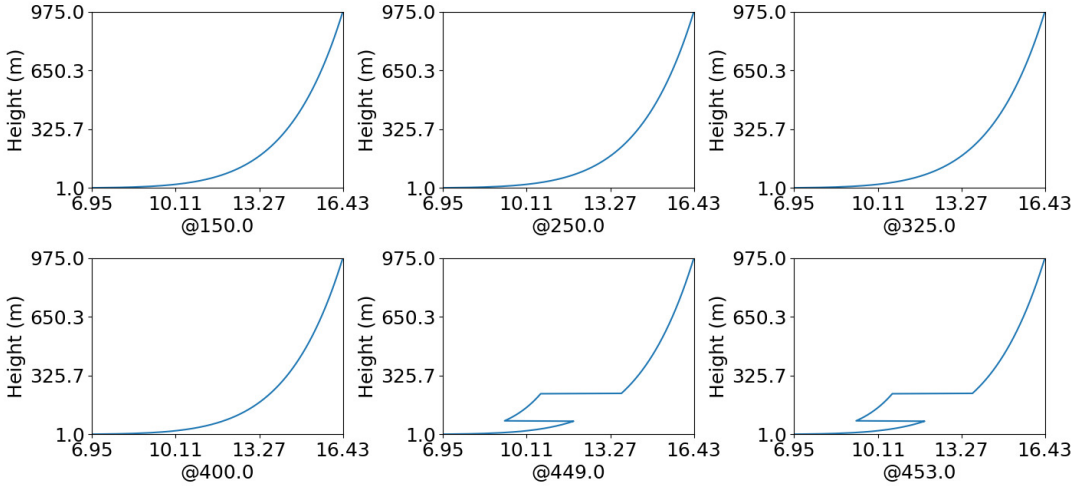


Figure 5.19: Change in velocity profile (m/s) both upstream and downstream of A_0 (using FLORIS)

The change in velocity profile for the case with one actuator is now compared. The change in velocity profile using DIIM both upstream and downstream of the actuator, A_0 is shown in Figure 5.14. Compared to Figure 5.19 generated using FLORIS, a significant change in the profile is observed. For instance, the effect the actuator has on the upstream velocity at $X_1 = 400\ m$ is clearly not reflected when FSI effects are ignored in FLORIS model. Further, the velocity profile just upstream and downstream (i.e. at $X_1 = 449\ m$ and at $X_1 = 453\ m$ respectively) of the actuator are seen to be identical in FLORIS. However, because of the effects of vorticity and FSI, the magnitude of velocity and hence, the thrust at $X_1 = 449\ m$ is observed to be on the higher side compared to the case when these effects are ignored.

5.4 Discussions

In this chapter, a formulation for FSI simulation in two dimensional wind fields over a domain which is large in size, by adapting the Decomposed Immersed Interface Method has been proposed. In Chapter 4, it was seen that the way the actuator discs were modelled resulted in extensive zones of low velocity which was unrealistic and physically impossible. Now, from the contour plots in this chapter which implement DIIM, it is clear that the FSI approach is working and more realistic results are obtained and the issue of extensive zones of low velocity has been resolved. It is important to note that the locations of the actuators are chosen as fractional

numbers like $448.3\ m, 1500.3\ m, 1198.3\ m, 1948.3\ m$. The reason for this is to ensure that the interfaces do not coincide with the grid lines.

Also, the application of FSI to a large domain with wind turbines modelled as actuator discs using the concepts of parallel numerical algorithms and Message Passing Interface to achieve a fast and memory efficient simulation for a wind field with constant vorticity has been demonstrated. The advantage of using multi-core parallel process to achieve significantly lower runtime when compared to a sequential process has also been presented.

Further, the simulation is done for a more realistic physical model and is not based on any empirical model or assumptions. Unlike empirical models, the approach used in these simulations can be used to simulate a practical wind energy related problem more realistically. Different IIM strategies work on modifying the coefficient matrix. However, Decomposed IIM works by correcting the residual at each iteration and hence is convenient to apply with most existing iterative algorithms. An attempt has been made to optimize the under-relaxation parameter for a large wind field subjected to shear flow but it was found that the under-relaxation parameter is case dependent as with most fluid dynamics problems. The under-relaxation factor is found to be within the range of $0.01 - 0.05$ for all the simulations.

In this work, an FSI strategy using DIIM in the presence of vorticity and demonstrated how this can be implemented to simplify complex CFD wind farm models has been demonstrated. Modifications to Jensen's model have been proposed to compute the appropriate jump conditions for the purpose of FSI analysis when wake-wake interaction is not pre-dominant. The values of the velocity field has been computed theoretically using the actuator disc theory and modifications to Jensen's model and it is found that the results obtained numerically using FSI strategies are in close resemblance with these theoretical values. Presence of stationary points in the fluid are detected in the contour plots signifying the presence of vorticity and the impact of FSI. This fact also illustrates the importance of FSI in the wind farm models and how this interaction influences the generation of vortices. The analytical wind farm models alone would not have been able to predict these vortices. Though the simulations are carried out for a fixed value of axial induction factor, the approach can still be used for cases where the axial induction factor varies from turbine to turbine. It is expected that other wind farm models like the Gaussian

models can be similarly implemented alongwith DIIM.

Chapter 6

A linearized model of turbulence using Rapid Distortion Theory

6.1 Introduction

In the chapters discussed so far, the aerodynamics of the ABL was discussed from the perspective of mean velocity only and turbulence was not taken in to account. However, turbulence is also an important aspect which cannot simply be ignored and is a common phenomenon associated with almost every fluid flow problem. As such in this chapter, turbulence has been investigated and a new model using Rapid Distortion Theory (RDT) has been proposed. But, before proceeding, a few concepts need to be highlighted. Velocity and pressure are two important parameters which govern a multi-physics problem like aerodynamics. Although other parameters like temperature also affect the behaviour of fluids, this thesis is centred around steady state analysis of aerodynamics of wind farms. As such diurnal fluctuations in temperature do not come into play. Since turbulence introduces fluctuations in these physical parameters of the fluid, it has to be treated differently. Thus, any physical parameter, Φ of fluid can be visualized in terms of a mean and a fluctuating component which is commonly referred to as *Reynolds Decomposition* i.e.

$$\Phi = \overline{\Phi} + \Phi'. \tag{6.1}$$

The issue with turbulence is that because of its disorderly nature, even if the same experimental setup is used, the realization of the fields is going to be different every

time. This is where ergodicity comes into picture. Ideally, this would mean that the mean referred to here is the ergodic mean considered over a large set of realizations of the experiment. Thus, some sort of averaging is required, be it spatial or temporal. Reynolds, 1895 used certain averaging conditions not all of which were formulated correctly. But, based on Reynolds's work, Monin and Yaglom, 1971 stated that the following five relations known as *Reynolds conditions* must be satisfied.

$$\overline{f+g} = \overline{f} + \overline{g} \tag{6.2a}$$

$$\overline{af} = a\overline{f}; a = \text{constant} \tag{6.2b}$$

$$\overline{a} = a; a = \text{constant} \tag{6.2c}$$

$$\overline{\frac{\partial f}{\partial s}} = \frac{\partial \overline{f}}{\partial s}; s \text{ is } X_1, X_2, X_3, t \tag{6.2d}$$

$$\overline{\overline{f}g} = \overline{f}\overline{g} \tag{6.2e}$$

where, $f = f(X_1, X_2, X_3, t) = \overline{f} + f'$ and $g = g(X_1, X_2, X_3, t) = \overline{g} + g'$ and $\{f, g\} \in \Phi$. These conditions form the basis of the derivations that have been undertaken in this chapter. Using these equations, it is easy to show that the following relations hold true.

$$\overline{\overline{f}} = \overline{\overline{f}.1} = \overline{f} \text{ (using eq.6.2e)} \tag{6.3a}$$

$$\overline{f'} = \overline{f - \overline{f}} = \overline{f} - \overline{\overline{f}} = \overline{f} - \overline{f} = 0 \text{ (using eq.6.2a and eq.6.3a)} \tag{6.3b}$$

$$\overline{\overline{f}\overline{g}} = \overline{f}\ \overline{g} \text{ (using eq.6.2e)} \tag{6.3c}$$

$$\overline{\overline{f}g'} = \overline{f}\ \overline{g'} = 0 \text{ (using eq.6.2e and eq.6.3b)} \tag{6.3d}$$

From the above expressions, it is evident that the mean of the fluctuating component, Φ' is always 0. An important result that can be deduced from these averaging conditions and one which is used quite frequently is the covariance of f and g.

$$\begin{aligned}\overline{fg} = \overline{(\overline{f} + f')(\overline{g} + g')} &= \overline{\overline{f}\ \overline{g} + f'g' + \overline{f}g' + \overline{g}f'} \\ &= \overline{\overline{f}\ \overline{g}} + \overline{f'g'} + \overline{\overline{f}g'} + \overline{\overline{g}f'} \\ &= \overline{f}\ \overline{g} + \overline{f'g'}\end{aligned} \tag{6.4}$$

6.2 Multivariate random fields and moments

As is evident from the ergodicity of the turbulence, its nature is best described by statistical measures rather than by some deterministic quantity. As such, few concepts from the perspective of statistical mechanics need to be highlighted first before moving on to the derivations pertaining to the proposed model. Determining the precise statistical specification for any random field is cumbersome and difficult to estimate. For this reason, practically studies are restricted to simpler statistical parameters which describe some particular statistical property of the flow. **Moments** of the probability distribution is one such parameter. For N jointly distributed random variables, $u_i; i \in \{0 \leq i \leq N\}$ with probability density function (pdf), $p(u_1, u_2, ...u_N)$, the moment is defined as

$$\begin{aligned} B_{k_1 k_2 ... k_N} &= \overline{u_1^{k_1} u_2^{k_2} ... u_N^{k_N}} \\ &= \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} ... \int_{-\infty}^{\infty} u_1^{k_1} u_2^{k_2} ... u_N^{k_N} p(u_1, u_2, ...u_N) du_1 du_2 ... du_N \end{aligned} \tag{6.5}$$

where, $k_i \in \mathbb{N}$ and order of the moment $= \sum k_i$. Thus, the mean can be visualized as the moment of first order. However, the central moment is another moment that is used quite often and is defined by

$$b_{k_1 k_2 ... k_N} = \overline{(u_1 - \overline{u}_1)^{k_1} (u_2 - \overline{u}_2)^{k_2} ... (u_N - \overline{u}_N)^{k_N}} \tag{6.6}$$

The second order moment b_2 is referred to as the **variance**, $\sigma_{u_i}^2$ and its square root σ_{u_i} is referred to as the **standard deviation**. Similarly, the second order moment of two variables u_j and u_k defined by $b_{11} = \sigma_{u_j u_k} = \overline{(u_j - \overline{u}_j)(u_k - \overline{u}_k)}$ is referred to as the **covariance**.

Gaussian random fields are one of the most common observed in nature. For this reason, gaussian distribution is considered to model the turbulence. For N jointly distributed random variables, $u_i; i \in \{0 \leq i \leq N\}$ the gaussian probability density function (pdf) is defined by

$$\begin{aligned} p(u_1, u_2, ..., u_N) &= \frac{1}{\sqrt{(2\phi)^N . |\boldsymbol{\Sigma}|}} \exp\left\{ -\frac{1}{2} (\mathbf{U} - \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1} (\mathbf{U} - \boldsymbol{\mu}) \right\} \\ &= \frac{1}{\sqrt{(2\phi)^N . |\boldsymbol{\Sigma}|}} \exp\left\{ -\frac{1}{2} \sum_{j,k=1}^{N} g_{jk} (u_j - a_j)(u_k - a_k) \right\} \end{aligned} \tag{6.7}$$

where, $\{a_j, g_{jk}\} \in \mathbb{R}$, $\mathbf{U} = [u_1\ u_2\ ...\ u_N]^T$ and $\boldsymbol{\mu} = [\mu_1\ \mu_2\ ...\ \mu_N]^T$ are vectors of random variables, u_i and their respective means, $\mathbf{\Sigma}$ is the variance-covariance matrix of size $N \times N$ of the random variables and $|\mathbf{\Sigma}|$ denotes its determinant value. From the concepts of linear algebra, it is known that the determinant value of a matrix is the product of its eigen values i.e. if $\lambda_i; i \in \{0 \leq i \leq N\}$ are the eigen values of $\mathbf{\Sigma}$, then

$$|\mathbf{\Sigma}| = \prod_{i=1}^{N} \lambda_i \Rightarrow |\mathbf{\Sigma}|^{1/2} = \prod_{i=1}^{N} \lambda_i^{1/2}$$

It is evident from the above expression that $\lambda_i > 0$ as $p \in \mathbb{R}$ and hence, $\mathbf{\Sigma}$ should be positive semi-definite. It can be easily seen that all central moments of odd order are 0. Isserlis, 1918 deduced the central moments of even order for jointly distributed gaussian variables, $w_i; \{i : i \in \mathbb{N}, 1 \leq i \leq 2K\}$ (out of which some might be identical) with mean 0 and presented a general rule;

$$\overline{w_1 w_2 ... w_{2K}} = \sum \overline{w_{i_1} w_{i_2}}\ \overline{w_{i_3} w_{i_4}}\ ...\ \overline{w_{i_{2K-1}} w_{i_{2K}}} \tag{6.8}$$

In other words, the even moment can be expressed as the sum of the product of the covariances of all possible pairs of $2K$ random variables. Thus, the fourth order moment can be expressed as

$$b_{1111} = \overline{w_1 w_2 w_3 w_4} = \overline{w_1 w_2}\ \overline{w_3 w_4} + \overline{w_1 w_3}\ \overline{w_2 w_4} + \overline{w_1 w_4}\ \overline{w_2 w_3} \tag{6.9}$$

It has been shown earlier in eq.6.3b that mean of the turbulent components is 0. Thus, using the above equation, the fourth order moment of turbulent components can be expressed as

$$\overline{u_1' u_2' u_3' u_4'} = \overline{u_1' u_2'}\ \overline{u_3' u_4'} + \overline{u_1' u_3'}\ \overline{u_2' u_4'} + \overline{u_1' u_4'}\ \overline{u_2' u_3'} \tag{6.10}$$

6.3 A RDT based model

It is not unknown that non-linear PDEs like the Navier-Stokes equations which define the fluid flow consume considerable computational time. This is why linearization like RDT comes into picture. In this section, equations for the turbulent components have been derived using Reynolds conditions introduced in the previous section. Air can be considered as an incompressible, homogeneous Newtonian fluid having total velocity field $\mathbf{V_j}$. From equation of continuity of an incompressible

fluid,

$$\frac{\partial V_j}{\partial X_j} = 0 \tag{6.11}$$

Applying Reynold's averaging (Monin and Yaglom, 1971) to the the above equation;

$$\frac{\partial \overline{V}_j}{\partial X_j} = 0 \tag{6.12}$$

Eq.6.11 can be re-written as

$$\frac{\partial \overline{V}_j}{\partial X_j} + \frac{\partial v'_j}{\partial X_j} = 0 \tag{6.13}$$

Subtracting eq.6.12 from eq.6.13;

$$\frac{\partial v'_j}{\partial X_j} = 0 \tag{6.14}$$

Ignoring body forces, the well known Navier Stokes equations are given by:

$$\frac{\partial V_j}{\partial t} + \underbrace{V_k \frac{\partial V_j}{\partial X_k}}_{\text{Advection}} = \underbrace{-\frac{1}{\rho}.\frac{\partial P}{\partial X_j}}_{\text{Source}} + \underbrace{\nu \frac{\partial^2 V_j}{\partial X_k \partial X_k}}_{\text{Diffusion}} \tag{6.15}$$

eq.6.15 is also a form of convection-diffusion equation where the advection, source and diffusion are as shown. However, unlike a standard convection-diffusion equation, there is no separate equation for source term. Using eq.6.11, the advection term of eq.6.15 can be modified and written as,

$$\underbrace{\frac{\partial V_j}{\partial t}}_{\text{Term-1}} + \underbrace{\frac{\partial (V_j V_k)}{\partial X_k}}_{\text{Term-2}} = \underbrace{-\frac{1}{\rho}.\frac{\partial P}{\partial X_j}}_{\text{Term-3}} + \underbrace{\nu \frac{\partial^2 V_j}{\partial X_k \partial X_k}}_{\text{Term-4}} \tag{6.16}$$

Taking mean of eq.6.15, the Reynold's Averaged Navier Stokes equations are obtained:

$$\underbrace{\frac{\partial \overline{V}_j}{\partial t}}_{\text{Term-1}} + \underbrace{\frac{\partial (\overline{V}_j \overline{V}_k + \overline{v'_j v'_k})}{\partial X_k}}_{\text{Term-2}} = \underbrace{-\frac{1}{\rho}.\frac{\partial \overline{P}}{\partial X_j}}_{\text{Term-3}} + \underbrace{\nu \frac{\partial^2 \overline{V}_j}{\partial X_k \partial X_k}}_{\text{Term-4}} \tag{6.17}$$

Subtracting eq.6.17 from eq.6.16the following simplification can be obtained:
Term-1:

$$\frac{\partial (V_j - \overline{V}_j)}{\partial t} = \frac{\partial v'_j}{\partial t}$$

Term-2:

$$\frac{\partial[V_jV_k-(\overline{V}_j\overline{V}_k+\overline{v_j'v_k'})]}{\partial X_k}=\frac{\partial[\overline{V}_kv_j'+\overline{V}_jv_k'+v_j'v_k'-\overline{v_j'v_k'}]}{\partial X_k}$$

Term-3:

$$\frac{\partial(P-\overline{P})}{\partial X_j}=\frac{\partial p'}{\partial X_j}$$

Term-4:

$$\frac{\partial^2(V_j-\overline{V}_j)}{\partial X_k\partial X_k}=\frac{\partial^2 v_j'}{\partial X_k\partial X_k}$$

Thus, the following equation is obtained

$$\frac{\partial v_j'}{\partial t}+\frac{\partial}{\partial X_k}(\underbrace{\overline{V}_kv_j'+\overline{V}_jv_k'}_{\substack{\text{Linear or}\\ \text{Rapid Terms}}}+\underbrace{v_j'v_k'-\overline{v_j'v_k'}}_{\substack{\text{Non-Linear or}\\ \text{Slow Terms}}})=-\frac{1}{\rho}.\frac{\partial p}{\partial X_j}+\nu\frac{\partial^2 v_j'}{\partial X_k\partial X_k} \tag{6.18}$$

Let $[\overline{V}]=[\overline{V}_1\ \overline{V}_2\ \overline{V}_3]^T=[\overline{V}_1(X_3)\ 0\ 0]^T$ and $[v]=[v_1'\ v_2'\ v_3']^T$.
The **Linear** or **Rapid** terms of the first equation can be written as

$$\frac{\partial(\overline{V}_kv_j'+\overline{V}_jv_k')}{\partial X_k}=\Bigg[\underbrace{\left(\frac{\partial(\overline{V}_1v_j')}{\partial X_1}+\frac{\partial(\overline{V}_2v_j')}{\partial X_2}+\frac{\partial(\overline{V}_3v_j')}{\partial X_3}\right)}_{\text{Term-L1}}+\underbrace{\left(\frac{\partial(\overline{V}_jv_1')}{\partial X_1}+\frac{\partial(\overline{V}_jv_2')}{\partial X_2}+\frac{\partial(\overline{V}_jv_3')}{\partial X_3}\right)}_{\text{Term-L2}}\Bigg] \tag{6.19}$$

Since $\overline{V}_2=0$ and $\overline{V}_3=0$, Term-L1 reduces to

$$\frac{\partial(\overline{V}_1v_j')}{\partial X_1}=v_j'\frac{\partial\overline{V}_1}{\partial X_1}+\overline{V}_1\frac{\partial v_j'}{\partial X_1}=\overline{V}_1\frac{\partial v_j'}{\partial X_1}$$

(As $\overline{V}_1$ is a function of X_3 only)

Term-L2 can be modified as below:-

1. For j=1,

$$\frac{\partial(\overline{V}_1v_1')}{\partial X_1}+\frac{\partial(\overline{V}_1v_2')}{\partial X_2}+\frac{\partial(\overline{V}_1v_3')}{\partial X_3}=\overline{V}_1\Bigg[\underbrace{\frac{\partial v_1'}{\partial X_1}+\frac{\partial v_2'}{\partial X_2}+\frac{\partial v_3'}{\partial X_3}}_{\text{=0 as per eq.6.14}}\Bigg]+v_3'\frac{\partial\overline{V}_1}{\partial X_3}=v_3'\frac{\partial\overline{V}_1}{\partial X_3}$$

(As $\overline{V}_1$ is a function of X_3 only, the partial derivatives w.r.t. X_1 and X_2 become 0)

2. For $j = 2$ & $j = 3$, this term vanishes as $\overline{V}_2 = 0$ and $\overline{V}_3 = 0$

The **Non-Linear** or **Slow** terms can be written as

$$\frac{\partial(v_j'v_k' - \overline{v_j'v_k'})}{\partial X_k} = \left[\underbrace{\frac{\partial(v_j'v_1')}{\partial X_1} + \frac{\partial(v_j'v_2')}{\partial X_2} + \frac{\partial(v_j'v_3')}{\partial X_3}}_{\text{Term-NL1}} - \left(\underbrace{\frac{\partial\overline{v_j'v_1'}}{\partial X_1} + \frac{\partial\overline{v_j'v_2'}}{\partial X_2} + \frac{\partial\overline{v_j'v_3'}}{\partial X_3}}_{\text{Term-NL2}}\right)\right] \tag{6.20}$$

Term-NL1 can be simplified further using eq.6.14

$$v_1'\frac{\partial v_j'}{\partial X_1} + v_2'\frac{\partial v_j'}{\partial X_2} + v_3'\frac{\partial v_j'}{\partial X_3} + v_j'\left(\underbrace{\frac{\partial v_1'}{\partial X_1} + \frac{\partial v_2'}{\partial X_2} + \frac{\partial v_3'}{\partial X_3}}_{=0 \text{ as per eq.6.14}}\right) = v_1'\frac{\partial v_j'}{\partial X_1} + v_2'\frac{\partial v_j'}{\partial X_2} + v_3'\frac{\partial v_j'}{\partial X_3}$$

In a similar way, Term-NL2 can be written as

$$\frac{\partial\overline{v_j'v_1'}}{\partial X_1} + \frac{\partial\overline{v_j'v_2'}}{\partial X_2} + \frac{\partial\overline{v_j'v_3'}}{\partial X_3} = \overline{\frac{\partial(v_j'v_1')}{\partial X_1} + \frac{\partial(v_j'v_2')}{\partial X_2} + \frac{\partial(v_j'v_3')}{\partial X_3}} = \overline{v_1'\frac{\partial v_j'}{\partial X_1} + v_2'\frac{\partial v_j'}{\partial X_2} + v_3'\frac{\partial v_j'}{\partial X_3}}$$

In tensor notation,

$$\frac{\partial v_j'}{\partial t} + \overline{V}_1\frac{\partial v_j'}{\partial X_1} + v_3'\frac{\mathrm{d}\overline{V}_j}{\mathrm{d}X_3}\delta_{j1} + v_k'\frac{\partial v_j'}{\partial X_k} - \overline{v_k'\frac{\partial v_j'}{\partial X_k}} = -\frac{1}{\rho}\frac{\partial p'}{\partial X_j} + \nu\frac{\partial^2 v_j'}{\partial X_k\partial X_k} \tag{6.21}$$

The pressure fluctuation, p' can be obtained by taking divergence of both sides (i.e. using operator $\partial/\partial X_l$)

$$\frac{\partial}{\partial X_l}\left[\frac{\partial v_j'}{\partial t} + \overline{V}_1\frac{\partial v_j'}{\partial X_1} + v_3'\frac{\partial\overline{V}_j}{\partial X_3} + v_k'\frac{\partial v_j'}{\partial X_k} - \overline{v_k'\frac{\partial v_j'}{\partial X_k}}\right] = \frac{\partial}{\partial X_l}\left[-\frac{1}{\rho}\frac{\partial p'}{\partial X_j} + \nu\frac{\partial^2 v_j'}{\partial X_k\partial X_k}\right]$$

$$\Rightarrow \frac{\partial}{\partial t}\frac{\partial v_j'}{\partial X_l} + \overline{V}_1\frac{\partial}{\partial X_1}\frac{\partial v_j'}{\partial X_l} + \frac{\partial\overline{V}_1}{\partial X_l}\frac{\partial v_j'}{\partial X_1} + v_3'\frac{\partial}{\partial X_3}\frac{\partial\overline{V}_j}{\partial X_l} + \frac{\partial v_3'}{\partial X_l}\frac{\partial\overline{V}_j}{\partial X_3} + v_k'\frac{\partial}{\partial X_k}\frac{\partial v_j'}{\partial X_l} +$$
$$\frac{\partial v_k'}{\partial X_l}\frac{\partial v_j'}{\partial X_k} - \left(\overline{\frac{\partial v_k'}{\partial X_l}\frac{\partial v_j'}{\partial X_k} + v_k'\frac{\partial}{\partial X_k}\frac{\partial v_j'}{\partial X_l}}\right) = -\frac{1}{\rho}\frac{\partial^2 p'}{\partial X_l\partial X_j} + \nu\frac{\partial^2}{\partial X_k\partial X_k}\frac{\partial v_j'}{\partial X_l}$$

Putting $l = j = 1, 2, 3$ and adding the three equations, the first, second, fourth, sixth and last term on left hand side and second term on right hand side become 0.

$$\begin{aligned}\frac{\partial^2 p'}{\partial X_j \partial X_j} &= -\rho\left(\frac{\partial \overline{V}_1}{\partial X_j}\frac{\partial v'_j}{\partial X_1} + \frac{\partial v'_3}{\partial X_j}\frac{\partial \overline{V}_j}{\partial X_3} + \frac{\partial v'_k}{\partial X_j}\frac{\partial v'_j}{\partial X_k} - \overline{\frac{\partial v'_k}{\partial X_j}\frac{\partial v'_j}{\partial X_k}}\right) \\ &= -\rho\left(2\frac{\mathrm{d}\overline{V}_1}{\mathrm{d}X_3}\frac{\partial v'_3}{\partial X_1} + \frac{\partial v'_k}{\partial X_j}\frac{\partial v'_j}{\partial X_k} - \overline{\frac{\partial v'_k}{\partial X_j}\frac{\partial v'_j}{\partial X_k}}\right)\end{aligned} \tag{6.22}$$

6.3.1 Linearization

For linearization, the non-linear term can be approximated as

$$(v'_j v'_k - \overline{v'_j v'_k}) \simeq \kappa(\overline{V}_k v'_j + \overline{V}_j v'_k).\kappa > 0 \tag{6.23}$$

Therefore, eq.6.18 can be written as

$$\frac{\partial v'_j}{\partial t} + (1+\kappa)\frac{\partial}{\partial X_k}(\overline{V}_k v'_j + \overline{V}_j v'_k) = -\frac{1}{\rho}.\frac{\partial p'}{\partial X_j} + \nu\frac{\partial^2 v'_j}{\partial X_k \partial X_k} \tag{6.24}$$

which can be simplified to

$$\boxed{\frac{\partial v'_j}{\partial t} + (1+\kappa)\left(\overline{V}_k\frac{\partial v'_j}{\partial X_k} + v'_k\frac{\partial \overline{V}_j}{\partial X_k}\right) = -\frac{1}{\rho}.\frac{\partial p'}{\partial X_j} + \nu\frac{\partial^2 v'_j}{\partial X_k \partial X_k}} \tag{6.25}$$

and equation for pressure becomes

$$\boxed{\frac{\partial^2 p'}{\partial X_j \partial X_j} = -\rho(1+\kappa)\left(\frac{\partial \overline{V_k}}{\partial X_j}\frac{\partial v'_j}{\partial X_k} + \frac{\partial \overline{V}_j}{\partial X_k}\frac{\partial v'_k}{\partial X_j}\right)} \tag{6.26}$$

κ can be determined so that the variances of the left and right hand sides become equal. The following criteria equates the sum of the variances of all components of the turbulence with the same weight.

$$\overline{(v'_j v'_k - \overline{v'_j v'_k})(v'_j v'_k - \overline{v'_j v'_k})} = \kappa^2\overline{(\overline{V}_k v'_j + \overline{V}_j v'_k)(\overline{V}_k v'_j + \overline{V}_j v'_k)} \tag{6.27}$$

The left hand side of the above equation can be written as:

$$\overline{(v'_j v'_k - \overline{v'_j v'_k})(v'_j v'_k - \overline{v'_j v'_k})} = \overline{v'_j v'_k v'_j v'_k - 2v'_j v'_k.\overline{v'_j v'_k} + \overline{v'_j v'_k}.\overline{v'_j v'_k}}$$

$$
\begin{aligned}
&= \overline{v'_j v'_k v'_j v'_k} - 2\overline{v'_j v'_k}.\overline{v'_j v'_k} + \overline{\overline{v'_j v'_k}.\overline{v'_j v'_k}} \\
&= \overline{v'_j v'_k v'_j v'_k} - 2\overline{v'_j v'_k}.\overline{v'_j v'_k} + \overline{v'_j v'_k}.\overline{v'_j v'_k} \\
&= \overline{v'_j v'_k v'_j v'_k} - \overline{v'_j v'_k}.\overline{v'_j v'_k}
\end{aligned}
$$

If v'_j and v'_k are considered to follow joint gaussian pdf, then as per eq.6.10, the fourth order moment i.e. the first term of the equation gets simplified to

$$
\overline{v'_j v'_k v'_j v'_k} = \overline{v'_j v'_k}.\overline{v'_j v'_k} + \overline{v'_j v'_j}.\overline{v'_k v'_k} + \overline{v'_j v'_k}.\overline{v'_k v'_j} = 2\overline{v'_j v'_k}.\overline{v'_j v'_k} + \overline{v'_j v'_j}.\overline{v'_k v'_k}
$$

Thus,

$$
\overline{(v'_j v'_k - \overline{v'_j v'_k})(v'_j v'_k - \overline{v'_j v'_k})} = \overline{v'_j v'_k}.\overline{v'_j v'_k} + \overline{v'_j v'_j}.\overline{v'_k v'_k}
$$

The right hand side of the above equation can be simplified as follows:

$$
\overline{(\overline{V}_k v'_j + \overline{V}_j v'_k)(\overline{V}_k v'_j + \overline{V}_j v'_k)} = \overline{\overline{V}_k v'_j \overline{V}_k v'_j + \overline{V}_j v'_k \overline{V}_j v'_k + 2\overline{V}_k v'_j.\overline{V}_j v'_k}
$$

From the subscripts, it can be observed that the first two terms are identical i.e. $\overline{V}_k v'_j \overline{V}_k v'_j = \overline{V}_j v'_k \overline{V}_j v'_k = \overline{V}_m \overline{V}_m v'_l v'_l$. Using similar subscripts, the last term can be written as $\overline{V}_l \overline{V}_m v'_l v'_m$. Using eq.6.2, the above expression simplifies to

$$
\begin{aligned}
\overline{\overline{V}_k v'_j \overline{V}_k v'_j + \overline{V}_j v'_k \overline{V}_j v'_k + 2\overline{V}_k v'_j.\overline{V}_j v'_k} &= \overline{2(\overline{V}_m \overline{V}_m v'_l v'_l + \overline{V}_m \overline{V}_l v'_l v'_m)} \\
&= \overline{2\overline{V}_m(\overline{V}_m v'_l v'_l + \overline{V}_l v'_l v'_m)} \\
&= 2\overline{\overline{V}_m}(\overline{\overline{V}_m}\,\overline{v'_l v'_l} + \overline{\overline{V}_l}\,\overline{v'_l v'_m}) \\
&= 2\overline{V}_m(\overline{V}_m \overline{v'_l v'_l} + \overline{V}_l \overline{v'_l v'_m})
\end{aligned}
$$

Thus,

$$
\begin{aligned}
&\overline{(v'_j v'_k - \overline{v'_j v'_k})(v'_j v'_k - \overline{v'_j v'_k})} = \kappa^2 \overline{(\overline{V}_k v'_j + \overline{V}_j v'_k)(\overline{V}_k v'_j + \overline{V}_j v'_k)} \\
\Rightarrow &\overline{v'_j v'_k}.\overline{v'_j v'_k} + \overline{v'_j v'_j}.\overline{v'_k v'_k} = 2\kappa^2 \overline{V}_m(\overline{V}_m \overline{v'_l v'_l} + \overline{V}_l \overline{v'_l v'_m}) \\
\Rightarrow &\kappa = \sqrt{\frac{\overline{v'_j v'_j}.\overline{v'_k v'_k} + \overline{v'_j v'_k}.\overline{v'_j v'_k}}{2\overline{V}_m(\overline{V}_m.\overline{v'_l v'_l} + \overline{V}_l.\overline{v'_l v'_m})}}
\end{aligned}
\tag{6.28}
$$

With respect to eq.6.21, the linearized form is

$$
\boxed{\frac{\partial v'_j}{\partial t} + (1+\kappa)\left(\overline{V}_1 \frac{\partial v'_j}{\partial X_1} + v'_3 \delta_{j1} \frac{\mathrm{d}\overline{V}_j}{\mathrm{d}X_3}\right) = -\frac{1}{\rho}.\frac{\partial p}{\partial X_j} + \nu \frac{\partial^2 v'_j}{\partial X_k \partial X_k}} \tag{6.29}
$$

And for pressure, eq.6.22 becomes

$$\boxed{\frac{\partial^2 p}{\partial X_j \partial X_j} = -\rho(1+\kappa)\left(\frac{\mathrm{d}\overline{V}_1}{\mathrm{d}X_j}\frac{\partial v_j'}{\partial X_1}\delta_{j3} + \frac{\mathrm{d}\overline{V}_1}{\mathrm{d}X_3}\frac{\partial v_3'}{\partial X_1}\delta_{j1}\right)} \tag{6.30}$$

6.4 Non-Dimensionalizing

An usual approach to run numerical simulations on these sort of fluid flow equations is by non-dimensionalizing the fields so that one does not have to worry about the units or the effect they might have in terms of the order of magnitude of their values in numerical simulation. This can be done by using the scaling relations

$$\breve{v}_j = \frac{v_j' - V_r}{V_s}; \breve{X}_j = \frac{X_j - X_r}{X_s}; \breve{t} = \frac{t - t_r}{t_s}; \breve{p} = \frac{p' - P_r}{P_s}$$

where, the terms $\breve{.}$ denote the dimensionless velocity, distance, time and pressure terms and the terms with the subscripts r and s denote the reference values and the corresponding scale factors respectively. Once the length and velocity scales are defines, the time scale need not be defined explicitly and can instead be derived from these by using the relation

$$t_s = \frac{X_s}{V_s}$$

Before moving onto non-dimensionalizing the equations derived in the previous section, the derivative operators need to be expressed in terms of non-dimensionalized quantities as well. The partial derivative of any two quantities denoted by Θ and Φ with respect to their non-dimensional counterpart $\breve{\Theta}$ and $\breve{\Phi}$ and having a scale factor of Θ_s and Φ_s can be written in the form

$$\partial\Theta = \Theta_s \partial\breve{\Theta};\ \partial\Phi = \Phi_s \partial\breve{\Phi}.$$

Hence, the following operator can be obtained

$$\frac{\partial}{\partial\Phi} = \frac{1}{\Phi_s}.\frac{\partial}{\partial\breve{\Phi}}.$$

Differentiating again,

$$\frac{\partial^2}{\partial\Phi^2} = \frac{\partial}{\partial\Phi}\left[\frac{\partial}{\partial\Phi}\right] = \frac{1}{\Phi_s}.\frac{\partial}{\partial\breve{\Phi}}\left[\frac{1}{\Phi_s}.\frac{\partial}{\partial\breve{\Phi}}\right] = \frac{1}{\Phi_s^2}.\frac{\partial^2}{\partial\breve{\Phi}^2}.$$

Accordingly, for derivative of order n, the following derivative operator is obtained.

$$\frac{\partial^n}{\partial \Phi^n} = \frac{1}{\Phi_s^n}.\frac{\partial^n}{\partial \breve{\Phi}^n}$$

Thus, n^{th} order derivative of Θ w.r.t. Φ in terms of their non-dimensionalized counterpart can be written as

$$\frac{\partial^n \Theta_j}{\partial \Phi^n} = \frac{\Theta_s}{\Phi_s^n}\frac{\partial^n \breve{\Theta}_j}{\partial \breve{\Phi}^n}$$

Setting the reference values as 0 and using the expression eq.2.1 and eq.2.2, $\overline{V}_1$ and $d\overline{V}_1/dX_3$ can be non-dimensionalized as

$$\breve{V}_1 = \frac{\overline{V}_1}{V_s} = \frac{u_\tau}{V_s \kappa'}\left(\ln\left(\frac{X_3}{z_0}\right) + 5.75\left(\frac{X_3}{h}\right) - 1.875\left(\frac{X_3}{h}\right)^2 - \frac{4}{3}\left(\frac{X_3}{h}\right)^3 + \frac{1}{4}\left(\frac{X_3}{h}\right)^4\right) \tag{6.31}$$

$$\frac{\mathrm{d}\overline{V}_1}{\mathrm{d}X_3} = \frac{V_s}{X_s}\frac{\mathrm{d}\breve{V}_1}{\mathrm{d}\breve{X}_3} = \frac{u_\tau}{\kappa'}\left(\frac{1}{X_3} + \frac{5.75}{h} - \frac{3.750}{h^2}X_3 - \frac{4}{h^3}X_3^2 + \frac{X_3^3}{h^4}\right) \tag{6.32}$$

Using this approach, the equation of continuity; eq.6.14 can be non-dimensionalized as

$$\frac{\partial v_j}{\partial X_j} = 0 \Rightarrow \frac{V_s}{X_s}\frac{\partial \breve{v}_j}{\partial \breve{X}_j} = 0 \Rightarrow \frac{\partial \breve{v}_j}{\partial \breve{X}_j} = 0 \tag{6.33}$$

Similarly, the momentum equations, eq.6.21 can be non-dimensionalized to

$$\frac{V_s^2}{X_s}\left(\frac{\partial \breve{v}_j}{\partial \breve{t}} + \breve{V}_1\frac{\partial \breve{v}_j}{\partial \breve{X}_1} + \breve{v}_3\frac{\mathrm{d}\breve{V}_j}{\mathrm{d}\breve{X}_3}\delta_{j1} + \breve{v}_k\frac{\partial \breve{v}_j}{\partial \breve{X}_k} - \overline{\breve{v}_k\frac{\partial \breve{v}_j}{\partial \breve{X}_k}}\right) = -\frac{1}{\rho}\frac{P_s}{X_s}\frac{\partial \breve{p}}{\partial \breve{X}_j} + \nu\frac{V_s}{X_s^2}\frac{\partial^2 \breve{v}_j}{\partial \breve{X}_k \partial \breve{X}_k} \tag{6.34}$$

Dividing throughout by $\nu V_s/X_s^2$, the following equation is obtained:-

$$\begin{aligned}\underbrace{\frac{V_s X_s}{\nu}}_{\mathrm{Re}}\left(\frac{\partial \breve{v}_j}{\partial \breve{t}} + \breve{V}_1\frac{\partial \breve{v}_j}{\partial \breve{X}_1} + \breve{v}_3\frac{\mathrm{d}\breve{V}_j}{\mathrm{d}\breve{X}_3}\delta_{j1} + \breve{v}_k\frac{\partial \breve{v}_j}{\partial \breve{X}_k} - \overline{\breve{v}_k\frac{\partial \breve{v}_j}{\partial \breve{X}_k}}\right) &= -\frac{P_s X_s}{\mu V_s}\frac{\partial \breve{p}}{\partial \breve{X}_j} + \frac{\partial^2 \breve{v}_j}{\partial \breve{X}_k \partial \breve{X}_k} \\ \Rightarrow \left(\frac{\partial \breve{v}_j}{\partial \breve{t}} + \breve{V}_1\frac{\partial \breve{v}_j}{\partial \breve{X}_1} + \breve{v}_3\frac{\mathrm{d}\breve{V}_j}{\mathrm{d}\breve{X}_3}\delta_{j1} + \breve{v}_k\frac{\partial \breve{v}_j}{\partial \breve{X}_k} - \overline{\breve{v}_k\frac{\partial \breve{v}_j}{\partial \breve{X}_k}}\right) &= -\frac{P_s}{\rho V_s^2}\frac{\partial \breve{p}}{\partial \breve{X}_j} + \frac{1}{Re}\frac{\partial^2 \breve{v}_j}{\partial \breve{X}_k \partial \breve{X}_k}\end{aligned} \tag{6.35}$$

where, Reynolds number, $Re = V_s X_s/\nu$. Accordingly, the linearised form of momentum equations can be written as

$$\left(\frac{\partial \breve{v}_j}{\partial \breve{t}} + (1+\kappa)\left(\breve{V}_1 \frac{\partial \breve{v}_j}{\partial \breve{X}_1} + \breve{v}_3 \frac{\mathrm{d}\breve{V}_j}{\mathrm{d}\breve{X}_3}\delta_{j1}\right)\right) = -\frac{P_s}{\rho V_s^2}\frac{\partial \breve{p}}{\partial \breve{X}_j} + \frac{1}{Re}\frac{\partial^2 \breve{v}_j}{\partial \breve{X}_k \partial \breve{X}_k} \tag{6.36}$$

From this final form of the equation, it would be appropriate to set the pressure scale P_s as ρV_s^2 rather than assuming some arbitrary value. A suitable choice of the scaling factors X_s and V_s is required to execute the numerical simulation effectively. A detailed discussion on the same is done subsequently in Section 6.7.

6.5 Discretization

From Chapter 3, it is known that any derivative can be expressed either by one-sided differencing scheme or by central differencing scheme. However, the choice of the scheme depends on the nature of the problem itself. In this section, a discussion on the choice of the appropriate differencing schemes for the equations of continuity and momentum are presented.

6.5.1 Continuity equation

In the case of fluids, an important physical aspect is its direction of flow and the same should be reflected in the FDE approximation of its governing PDEs. Numerically it can be shown that the central differencing scheme cannot be used to express the derivatives in the equation of continuity, eq.6.14 in order to reflect the direction of flow but one-sided differencing schemes can. Thus, one-sided first order differencing scheme is implemented which is often referred to as the first order upwind scheme.

$$\frac{\partial v_j'}{\partial X_j} = 0 \Rightarrow \frac{v_1'^t - v_{1_{l-1}}'^t}{\triangle X_1} + \frac{v_2'^t - v_{2_{m-1}}'^t}{\triangle X_2} + \frac{v_3'^t - v_{3_{n-1}}'^t}{\triangle X_3} = 0 \tag{6.37}$$

It can be argued that a higher order differencing or at least a second-order differencing acheme might have been more accurate. However, many exsisting fluid flow algorithms use first order differencing for continuity equations. Also, the main task at hand is to check the performance of the proposed RDT model in light of the already existing algorithms and not to improve upon its accuracy. It can be noted that all the terms in the above equations are considered at time instance t. The

impact of t will be clear when the discussion on the momentum equations are done in Section 6.5.2.

6.5.2 Momentum equations

In this section, the FDE of eq.6.29 is examined closely. Let the fluctuating component of Φ in j^{th} direction at a node (l, m, n) in space and at time instance, t be denoted by Φ_j^t. Similarly, for neighbouring nodes, suffixes are added only for the node whose co-ordinate changes w.r.t. (l, m, n). For example, the parameters at $(l-1, m, n)$, $(l-2, m, n)$, $(l, m-1, n)$ are denoted by Φ_{jl-1}^t, Φ_{jl-2}^t, Φ_{jm-1}^t.

The momentum equation consists of derivatives for both velocity and pressure. The nature of the pressure is that it is not dependent on direction. Thus, a central differencing approach can be used for the first order pressure derivative in space and it is second order accurate.

An interesting thing to note in the momentum equation is the presence of a derivative with respect to time. Central differencing for this term does not make any sense as time always moves forward. Naturally the corresponding FDE of such a derivative will use forward differencing which in turn will have a term at time t and another at time $t+1$. This is referred to as 'forward marching' in time. A second or a higher order differencing in time which would have resulted in dependency on time step $t-1$ is not required because the result at time step t already has the effect of time step $t-1$ temporally.

Now the question arises whether the remaining terms are to be considered at time instance t or at time instance $t+1$. This is where two approaches to formulate the FDE come into picture viz. implicit and explicit. The implicit approach considers all the terms at time step $t+1$ i.e.

$$\frac{\partial v_j'}{\partial t} + (1+\kappa)\left(\overline{V}_1 \frac{\partial v_j'}{\partial X_1} + v_3' \delta_{j1} \frac{\mathrm{d}\overline{V}_j}{\mathrm{d}X_3}\right) = -\frac{1}{\rho}.\frac{\partial p'}{\partial X_j} + \nu \frac{\partial^2 v_j'}{\partial X_j \partial X_j}$$

$$\Rightarrow \frac{v_j'^{t+1} - v_j'^{t}}{\triangle t} + (1+\kappa)\left(\overline{V}_1 \frac{v_j'^{t+1} - v_{jl-1}'^{t+1}}{\triangle X_1} + v_3'^{t+1} \delta_{j1} \overline{V}_{jX_3}\right) =$$

$$-\frac{1}{2\rho \triangle X_j}[(p_{l+1}'^{t+1} - p_{l-1}'^{t+1})\delta_{j1} + (p_{m+1}'^{t+1} - p_{m-1}'^{t+1})\delta_{j2} + (p_{n+1}'^{t+1} - p_{n-1}'^{t+1})\delta_{j3}].$$

Evidently when the quantities are computed at time step $t+1$, except for $v_j'^{t}$, all

the other quantities are unknown. This makes the equations more complicated and highly coupled with the quantities computed at time step t. It is known that this approach is unconditionally stable. However, numerically, such intensively coupled equations would incur greater computational time for a single iteration of any iterative algorithm used to solve these equations. This is where the explicit approach comes into picture.

In explicit approach, the approximated FDE for the spatial derivatives in the above equation are considered at time step t instead of time step $t+1$ i.e.

$$\begin{aligned}
&\frac{\partial v'_j}{\partial t} + (1+\kappa)\left(\overline{V}_1\frac{\partial v'_j}{\partial X_1} + v'_3\delta_{j1}\frac{\mathrm{d}\overline{V}_j}{\mathrm{d}X_3}\right) = -\frac{1}{\rho}.\frac{\partial p'}{\partial X_j} + \nu\frac{\partial^2 v'_j}{\partial X_j \partial X_j} \\
\Rightarrow &\frac{v_j'^{t+1} - v_j'^t}{\triangle t} + (1+\kappa)\left(\overline{V}_1\frac{v_j'^t - v_{j_{l-1}}'^t}{\triangle X_1} + v_3'^t\delta_{j1}\overline{V}_{j_{X_3}}\right) = \\
&-\frac{1}{2\rho \triangle X_j}[(p_{l+1}'^t - p_{l-1}'^t)\delta_{j1} + (p_{m+1}'^t - p_{m-1}'^t)\delta_{j2} + (p_{n+1}'^t - p_{n-1}'^t)\delta_{j3}] \\
&-\nu[p(v_{j_{l+1}}'^t + v_{j_{l-1}}'^t) + q(v_{j_{m+1}}'^t + v_{j_{m-1}}'^t) + r(v_{j_{n+1}}'^t + v_{j_{n-1}}'^t) + sv_j'^t] \\
\Rightarrow &(v_j'^{t+1} - v_j'^t) + (1+\kappa)\triangle t\left(\overline{V}_1\frac{v_j'^t - v_{j_{l-1}}'^t}{\triangle X_1} + v_3'^t\delta_{j1}\overline{V}_{j_{X_3}}\right) = \\
&-\frac{\triangle t}{2\rho \triangle X_j}.[(p_{l+1}'^t - p_{l-1}'^t)\delta_{j1} + (p_{m+1}'^t - p_{m-1}'^t)\delta_{j2} + (p_{n+1}'^t - p_{n-1}'^t)\delta_{j3}] \\
&-\nu \triangle t[p(v_{j_{l+1}}'^t + v_{j_{l-1}}'^t) + q(v_{j_{m+1}}'^t + v_{j_{m-1}}'^t) + r(v_{j_{n+1}}'^t + v_{j_{n-1}}'^t) + sv_j'^t]. \qquad (6.38)
\end{aligned}$$

Clearly this approach reduces the coupling between the nodes both spatially and temporally because all the terms in the FDE are now known and $v_j'^{t+1}$ is the only term that is computed from the historical data. As such, compared to the implicit approach the computational cost also reduces. But, this reduced coupling comes at the cost of stability and thus requires a stability analysis. In case of inviscid fluids, the above PDE becomes hyperbolic and can be written as

$$\begin{aligned}
&(v_j'^{t+1} - v_j'^t) + \zeta(v_j'^t - v_{j_{l-1}}'^t) + (1+\kappa)\triangle t v_3'^t\delta_{j1}\overline{V}_{j_{X_3}} = \\
&-\frac{\triangle t}{2\rho \triangle X_j}.[(p_{l+1}'^t - p_{l-1}'^t)\delta_{j1} + (p_{m+1}'^t - p_{m-1}'^t)\delta_{j2} + (p_{n+1}'^t - p_{n-1}'^t)\delta_{j3}]. \qquad (6.39)
\end{aligned}$$

The above equation can be rearranged such that the velocity, $v_j'^{t+1}$ can be expressed

in terms of the quantities at the previous time step. which can be rearranged as

$$\begin{aligned} v_j'^{t+1} =&(1-\zeta)v_j'^t + \zeta v_{jl-1}'^t - (1+\kappa)\triangle t v_3'^t \delta_{j1}\overline{V}_{j_{X_3}} \\ &- \frac{\triangle t}{2\rho \triangle X_j}.[(p_{l+1}'^t - p_{l-1}'^t)\delta_{j1} + (p_{m+1}'^t - p_{m-1}'^t)\delta_{j2} \\ &+ (p_{n+1}'^t - p_{n-1}'^t)\delta_{j3}] \end{aligned} \tag{6.40}$$

where, convection number, $\zeta = (1+\kappa)\overline{V}_1\frac{\triangle t}{\triangle X_1}$. As per Courant–Friedrichs–Lewy (CFL) condition, to ensure stability of the discretized equation, $\zeta \leq 1$. But, if steady-state flow is considered, this criteria does not come into picture as the term $\frac{\partial v_j'}{\partial t}$ becomes 0. As only steady-state flow is considered in the current thesis, a detailed investigation into CFL criteria is not undertaken.

6.6 SIMPLER Algorithm

Since only steady state flow is considered, for clarity the superscript t is henceforth removed. Accordingly, the discussion and the equations presented in this section are in line with steady state form of the non-dimensionalized equation eq.6.35. As discussed earlier, Patankar and Spalding, 1972 first introduced SIMPLE (Semi implicit pressure linked) algorithm. Though SIMPLER has been implemented in the present work, an insight into SIMPLE is required beforehand so that the reason for using SIMPLER can be established.

SIMPLE begins by guessing the initial velocity fields; $\breve{v}_j^*$ and pressure field $\breve{p}^*$. The discretized momentum equations are then solved using these guessed values. Subsequently, using pressure correction equations, the pressure correction, p'' at each node is obtained. Using these values of p'', the values of velocity correction, v_j'' are obtained. The new corrected velocity and pressure fields are then obtained by using

$$\begin{aligned} \breve{v}_j &= \breve{v}_j^* + v_j'' \\ \breve{p}' &= \breve{p}^* + p'' \end{aligned} \tag{6.41}$$

The newly generated velocity and pressure fields are subsequently utilized in the next step of the iteration to further refine the values of v'' and p''. The steps are repeated until a converged solution is obtained. The steps to be carried out are outlined

in Algorithm 5. α_{pc} is the under-relaxation parameter needed to ensure numerical

Algorithm 5 SIMPLE

Input: $\breve{v}_j^*$, $\breve{p}^*$
while i=1,2,3,... **do**
  Solve the momentum equations to obtain $\breve{v}_j^*$
  Solve for p''
  $\breve{p}_{new} = \breve{p}^* + \alpha_{pc} p''$
  Compute v_j''
  $\breve{v}_{j,new} = \breve{v}_j^* + v_j''$
  if converged **then**
    break
  else
    $\breve{p}^* = \breve{p}_{new}$
    $\breve{v}_j^* = \breve{v}_{j,new}$
  end if
end while

stability. The momentum and pressure correction equations can be solved using any of the suitable iterative solvers presented earlier in Chapter 3. BiCGStab has been used as the preferred solver in this work. Additionally, the under-relaxation for the momentum equations is done by using the relation $\breve{v}_{j,new} = \alpha \breve{v}_j + (1-\alpha)\breve{v}_j^{n-1}$, where $\breve{v}_j^{n-1}$ is the value of $\breve{v}_j$ at end of iteration-$n-1$, $\breve{v}_j$ is the computed value from iteration n without any under-relaxation and $\breve{v}_{j,new}$ is the final under-relaxed value at the end of iteration-n of each iteration step of BiCGStab.

The issue with SIMPLE algorithm is that if the pressure field guessed initially is too far off, the solution will take much longer to converge. This motivated Patankar, 1980 to revise SIMPLE algorithm and propose SIMPLER. Just like SIMPLE, the momentum equations are also under-relaxed in this case. However, it is interesting to note that the pressure field is computed from the velocity fields and as such there is no chance that the pressure field will be far off from the actual value. Hence, the pressure field is not required to be corrected.

Comparing the two algorithms, one can note that BiCGStab needs to be run twice for every iteration of SIMPLE but thrice in the case of SIMPLER. At first glance, it might therefore seem that SIMPLER requires more computation time for every iteration and must not be preferred but then SIMPLER is much more efficient as the pressure field is not just any random field but values obtained from assumed velocity field and though the computation time per iteration is higher, the overall

Algorithm 6 SIMPLER

Input: $\breve{v}_j^*$
while i=1,2,3,... **do**
  Compute the pseudo-velocities, $\tilde{v}_j$
  Solve for $\breve{p}$
  $\breve{p}^* = \breve{p}$
  Solve the momentum equations to obtain $\breve{v}_j^*$
  Solve for p''
  Compute v_j''
  $\breve{v}_{j,new} = \breve{v}_j^* + v_j''$
  if converged **then**
    break
  else
    $\breve{v}_j^* = \breve{v}_{j,new}$
  end if
end while

time required to reach a converged solution for both pressure and velocity field is less in SIMPLER. As such SIMPLER has been used as the preferred algorithm. Of course SIMPLEC or PISO could also have been used but as highlighted earlier the performance of these algorithms are flow dependent and it is hard to justify which algorithm (SIMPLER/SIMPLEC/PISO) would perform better in the current scenario.

6.6.1 Staggered grid applied to FVM

Before moving onto the derivations, one aspect that FVM implements while using SIMPLE or its variants is the use of staggered grid. From the perspective of FVM, it is interesting to note that if the values of the scalar quantities like pressure are read at the same node as the velocity values, the true nature of the scalar field is not captured properly. This can be illustrated in the case of a pressure field shown in Figure 6.1.

The pressure gradient term for node P is given by

$$\frac{\partial P}{\partial x} = \frac{P_e - P_w}{\delta x} = \frac{\frac{P_E+P_P}{2} - \frac{P_P+P_W}{2}}{\delta x} = \frac{P_E - P_W}{2\delta x} = 100 - 100 = 0$$

This result gives an impression that there is an absence of any pressure gradient in the vicinity of node P but that is not the case. As such, the numerical analysis

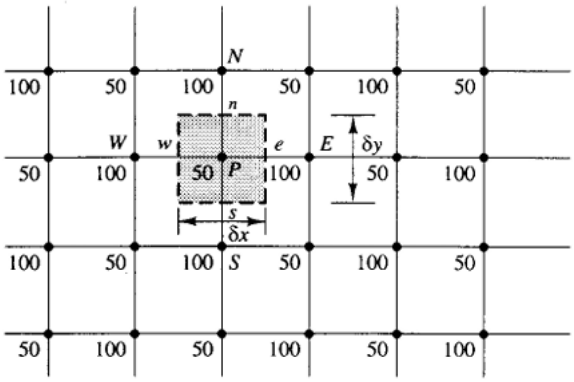


Figure 6.1: Pressure field in a 2D problem [Adapted from Versteeg and Malalasekera, 2007]

in this scenario generates an incorrect impression of the pressure gradient. To overcome this issue, Harlow and Welch, 1965 proposed the idea of a staggered grid. The idea behind staggered grid is to compute the scalar components like pressure, temperature, etc. at node points and velocity components on a grid staggered with respect to the grid capturing scalar components (i.e. at cell faces). A typical staggered grid is shown in Figure 6.2.

The pressure values are located at the intersection of the firm I and J grid lines (e.g. point P) and the velocity values are computed at the intersection of a firm line and a dotted line. Since, FDM is used throughout this work, linear interpolation done to obtain the value of P_e or P_w i.e. $P_e = (P_E + P_P)/2$ is not required to be done as will become clear from the derivations carried out in Section 6.6.2. As such a staggered grid is not required to be considered in the present scenario. There is of course another importance of staggered grid. In certain cases, if the velocity boundary conditions are known, then with staggered grid, the pressure boundary conditions need not be defined and instead can be derived from the velocity values of two adjacent cells.

6.6.2 Formulating equations for SIMPLER

It is to be noted that equations for computing pseudo-velocity or any of the corrections like pressure or velocity correction equations for SIMPLER have been mostly used in the context of FVM. Since, in the present work FDM has been implemented, the equations for SIMPLER need to be modified to suit FDM. Hence, in this sec-

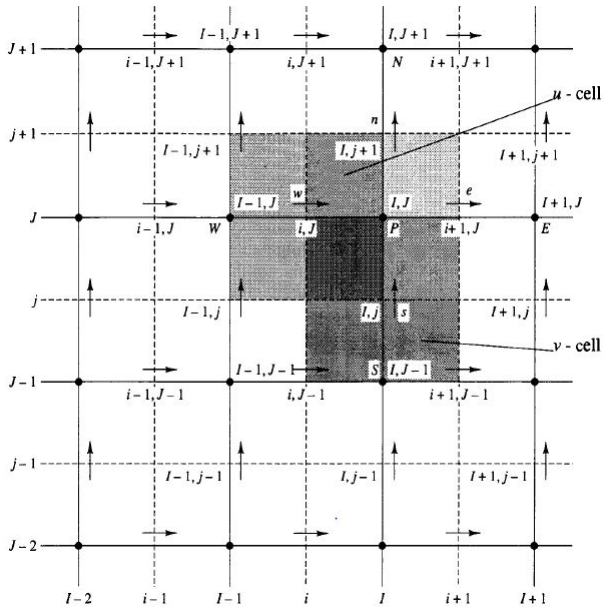


Figure 6.2: Staggered Grid System in a 2D problem [Adapted from Versteeg and Malalasekera, 2007]

tion, a step-by-step derivation of the equations required for running SIMPLER has been derived for steady state problem.

The momentum equations can be written as

$$
\begin{aligned}
&(1+\kappa)\left(\breve{V}_1\frac{\breve{v}_j-\breve{v}_{j_{l-1}}}{\triangle\breve{X}_1}+\breve{v}_3\delta_{j1}\breve{V}_{j_{\breve{X}_3}}\right)=\\
&-\frac{P_s}{2\rho V_s^2\triangle\breve{X}_j}[(\breve{p}_{l+1}-\breve{p}_{l-1})\delta_{j1}+(\breve{p}_{m+1}-\breve{p}_{m-1})\delta_{j2}+(\breve{p}_{n+1}-\breve{p}_{n-1})\delta_{j3}]\\
&-\frac{1}{Re}[p(\breve{v}_{j_{l+1}}+\breve{v}_{j_{l-1}})+q(\breve{v}_{j_{m+1}}+\breve{v}_{j_{m-1}})+r(\breve{v}_{j_{n+1}}+\breve{v}_{j_{n-1}})+s\breve{v}_j]\\
\Rightarrow&a(\breve{v}_j-\breve{v}_{j_{l-1}})-b_j=\\
&-\frac{P_s}{2\rho V_s^2\triangle\breve{X}_j}.[(\breve{p}_{l+1}-\breve{p}_{l-1})\delta_{j1}+(\breve{p}_{m+1}-\breve{p}_{m-1})\delta_{j2}+(\breve{p}_{n+1}-\breve{p}_{n-1})\delta_{j3}]\\
&-\frac{1}{Re}[p(\breve{v}_{j_{l+1}}+\breve{v}_{j_{l-1}})+q(\breve{v}_{j_{m+1}}+\breve{v}_{j_{m-1}})+r(\breve{v}_{j_{n+1}}+\breve{v}_{j_{n-1}})+s\breve{v}_j]\\
\Rightarrow&\breve{v}_j=\breve{v}_{j_{l-1}}+b_j/a\\
&-\frac{P_s}{2a\rho V_s^2\triangle\breve{X}_j}.[(\breve{p}_{l+1}-\breve{p}_{l-1})\delta_{j1}+(\breve{p}_{m+1}-\breve{p}_{m-1})\delta_{j2}+(\breve{p}_{n+1}-\breve{p}_{n-1})\delta_{j3}]\\
&-\frac{1}{a.Re}[p(\breve{v}_{j_{l+1}}+\breve{v}_{j_{l-1}})+q(\breve{v}_{j_{m+1}}+\breve{v}_{j_{m-1}})+r(\breve{v}_{j_{n+1}}+\breve{v}_{j_{n-1}})+s\breve{v}_j]
\end{aligned}
$$

$$\Rightarrow \left(1+\frac{s}{a.Re}\right)\breve{v}_j = \Big(-\frac{p}{Re.a}\breve{v}_{j_{l+1}} + \left(1-\frac{p}{Re.a}\right)\breve{v}_{j_{l-1}} - \frac{q}{Re.a}(\breve{v}_{j_{m+1}}+\breve{v}_{j_{m-1}})$$
$$-\frac{r}{Re.a}(\breve{v}_{j_{n+1}}+\breve{v}_{j_{n-1}}) + b_j/a\Big) +$$
$$\frac{d_j}{2}.[(\breve{p}_{l+1}-\breve{p}_{l-1})\delta_{j1} + (\breve{p}_{m+1}-\breve{p}_{m-1})\delta_{j2} + (\breve{p}_{n+1}-\breve{p}_{n-1})\delta_{j3}]$$
$$\Rightarrow \breve{v}_j = \Big(-\frac{p}{Re.a}\breve{v}_{j_{l+1}} + \left(1-\frac{p}{Re.a}\right)\breve{v}_{j_{l-1}} - \frac{q}{Re.a}(\breve{v}_{j_{m+1}}+\breve{v}_{j_{m-1}})$$
$$-\frac{r}{Re.a}(\breve{v}_{j_{n+1}}+\breve{v}_{j_{n-1}}) + b_j/a\Big)/D +$$
$$\frac{d_j}{2D}.[(\breve{p}_{l+1}-\breve{p}_{l-1})\delta_{j1} + (\breve{p}_{m+1}-\breve{p}_{m-1})\delta_{j2} + (\breve{p}_{n+1}-\breve{p}_{n-1})\delta_{j3}] \tag{6.42}$$

where,
$a = (1+\kappa)\breve{V}_1/\triangle\breve{X}_1$; $b_j = -(1+\kappa)\breve{v}_3\delta_{j1}\breve{V}_{j_{X_3}}$; $d_j = -P_s/(V_s^2\rho a\triangle\breve{X}_j)$; $D = \left(1+\frac{s}{a.Re}\right)$.
The terms p, q, r, s are the same as defined in previous chapters in the context of Laplace operator.

From the above equation for $\breve{v}_j$, it can also be seen that the term b_j is non-zero only if $j = 1$. Since, b_j is dependent on $\breve{v}_3$, it is essential that its value is known before the set of equations are solved for $\breve{v}_1$. Accordingly, the sequence of solving the momentum equations while using SIMPLER should be set as $\breve{v}_3$, $\breve{v}_2$ and $\breve{v}_1$. If looked at closely, it can be seen that the momentum equations is of the form

$$\boxed{\breve{v}_j = \sum \frac{a_{nb}\breve{v}_{j_{nb}}}{D} + \frac{d_j}{2D}.[(\breve{p}_{l+1}-\breve{p}_{l-1})\delta_{j1} + (\breve{p}_{m+1}-\breve{p}_{m-1})\delta_{j2} + (\breve{p}_{n+1}-\breve{p}_{n-1})\delta_{j3}]} \tag{6.43}$$

In the context of SIMPLER, the term $\tilde{v}_j = \sum a_{nb}\breve{v}_{nb}/D$ is referred to as pseudo-velocity where the subscript nb indicates neighbouring nodes i.e. $\breve{v}_{j_{nb}}$ is the velocity of the neighbouring nodes and a_{nb} is the coefficient.

The next step is obtaining the pressure equation. The discretized equation for pressure can be obtained by putting the above obtained expressions of the velocity components in the equation of continuity.

$$\frac{\breve{v}_1-\breve{v}_{1_{l-1}}}{\triangle\breve{X}_1} + \frac{\breve{v}_2-\breve{v}_{2_{m-1}}}{\triangle\breve{X}_2} + \frac{\breve{v}_3-\breve{v}_{3_{n-1}}}{\triangle\breve{X}_3} = 0$$
$$\Rightarrow \quad ((\tilde{v}_1 + \tfrac{d_1}{2D}(\breve{p}_{l+1}-\breve{p}_{l-1})) - (\tilde{v}_{1_{l-1}} + \tfrac{d_{1_{l-1}}}{2D_{l-1}}(\breve{p}-\breve{p}_{l-2})))/\triangle\breve{X}_1 +$$
$$((\tilde{v}_2 + \tfrac{d_2}{2D}(\breve{p}_{m+1}-\breve{p}_{m-1})) - (\tilde{v}_{2_{m-1}} + \tfrac{d_{2_{m-1}}}{2D_{m-1}}(\breve{p}-\breve{p}_{m-2})))/\triangle\breve{X}_2 +$$
$$((\tilde{v}_3 + \tfrac{d_3}{2D}(\breve{p}_{n+1}-\breve{p}_{n-1})) - (\tilde{v}_{3_{n-1}} + \tfrac{d_{3_{n-1}}}{2D_{n-1}}(\breve{p}-\breve{p}_{n-2})))/\triangle\breve{X}_3 = 0$$

Thus,

$$\boxed{\begin{aligned}&(\tilde{a}_{1_{l-1}}+\tilde{a}_{2_{m-1}}+\tilde{a}_{3_{n-1}})\breve{p}=\\&\tilde{a}_1(\breve{p}_{l+1}-\breve{p}_{l-1})+\tilde{a}_2(\breve{p}_{m+1}-\breve{p}_{m-1})+\tilde{a}_3(\breve{p}_{n+1}-\breve{p}_{n-1})+\\&\tilde{a}_{1_{l-1}}\breve{p}_{l-2}+\tilde{a}_{2_{m-1}}\breve{p}_{m-2}+\tilde{a}_{3_{n-1}}\breve{p}_{n-2}-2\tilde{b}\end{aligned}} \tag{6.44}$$

where, $\tilde{a}_j=\frac{d_j}{D\triangle\breve{X}_j}$; $\tilde{b}=-\frac{\tilde{v}_1-\tilde{v}_{1_{l-1}}}{\triangle\breve{X}_1}-\frac{\tilde{v}_2-\tilde{v}_{2_{m-1}}}{\triangle\breve{X}_2}-\frac{\tilde{v}_3-\tilde{v}_{3_{n-1}}}{\triangle\breve{X}_3}$

Since SIMPLER begins by guessing an initial velocity field, $\breve{v}^*$ and computing a guessed pressure field, $\breve{p}^*$ using the pressure equation, the momentum equations can be written as

$$\breve{v}_j^*=\sum\frac{a_{nb}\breve{v}_{j_{nb}}^*}{D}+\frac{d_j}{2D}.[(\breve{p}_{l+1}^*-\breve{p}_{l-1}^*)\delta_{j1}+(\breve{p}_{m+1}^*-\breve{p}_{m-1}^*)\delta_{j2}+(\breve{p}_{n+1}^*-\breve{p}_{n-1}^*)\delta_{j3}] \tag{6.45}$$

Subtracting eq.6.45 from eq.6.43;

$$\begin{aligned}&(\breve{v}_j-\breve{v}_j^*)=\sum\frac{a_{nb}(\breve{v}_{j_{nb}}-\breve{v}_{j_{nb}}^*)}{D}+\frac{d_j}{2D}.[((\breve{p}_{l+1}-\breve{p}_{l+1}^*)-(\breve{p}_{l-1}-\breve{p}_{l-1}^*))\delta_{j1}+\\&((\breve{p}_{m+1}-\breve{p}_{m+1}^*)-(\breve{p}_{m-1}-\breve{p}_{m-1}^*))\delta_{j2}+((\breve{p}_{n+1}-\breve{p}_{n+1}^*)-(\breve{p}_{n-1}-\breve{p}_{n-1}^*))\delta_{j3}]\\&\Rightarrow v_j''=\sum\frac{a_{nb}v_{j_{nb}}''}{D}+\frac{d_j}{2D}.[(p_{l+1}''-p_{l-1}'')\delta_{j1}+(p_{m+1}''-p_{m-1}'')\delta_{j2}+(p_{n+1}''-p_{n-1}'')\delta_{j3}]\end{aligned}$$

Dropping the first term of RHS gives the equation for velocity correction i.e.

$$v_j''=\frac{d_j}{2D}.[(p_{l+1}''-p_{l-1}'')\delta_{j1}+(p_{m+1}''-p_{m-1}'')\delta_{j2}+(p_{n+1}''-p_{n-1}'')\delta_{j3}] \tag{6.46}$$

This step forms the backbone of SIMPLER. Had this term been retained, it would have brought in the velocity corrections of the adjacent nodes which in turn are again dependent on pressure correction of neighbouring nodes and as such the equation would have become implicit. Dropping this term bypasses this issue making the equation partially implicit and hence the word semi-implicit was used by Patankar. The corrected velocity can therefore be written as

$$\breve{v}_j=\breve{v}_j^*+v_j''=\breve{v}_j^*+\frac{d_j}{2D}.[(p_{l+1}''-p_{l-1}'')\delta_{j1}+(p_{m+1}''-p_{m-1}'')\delta_{j2}+(p_{n+1}''-p_{n-1}'')\delta_{j3}] \tag{6.47}$$

The corrected velocity should also satisfy the continuity equation. Substituting this in continuity equation,

$$\frac{\breve{v}_1-\breve{v}_{1_{l-1}}}{\triangle\breve{X}_1}+\frac{\breve{v}_2-\breve{v}_{2_{m-1}}}{\triangle\breve{X}_2}+\frac{\breve{v}_3-\breve{v}_{3_{n-1}}}{\triangle\breve{X}_3}=0$$

$$\Rightarrow((v_1^* + \frac{d_1}{2D}(p''_{l+1} - p''_{l-1})) - (v^*_{1_{l-1}} + \frac{d_{1_{l-1}}}{2D}(p'' - p''_{l-2})))/ \triangle \breve{X}_1 +$$
$$((v_2^* + \frac{d_2}{2D}(p''_{m+1} - p''_{m-1})) - (v^*_{2_{m-1}} + \frac{d_{2_{m-1}}}{2D}(p'' - p''_{m-2})))/ \triangle \breve{X}_2 +$$
$$((v_3^* + \frac{d_3}{2D}(p''_{n+1} - p''_{n-1})) - (v^*_{3_{n-1}} + \frac{d_{3_{n-1}}}{2D}(p'' - p''_{n-2})))/ \triangle \breve{X}_3 = 0$$

Thus,

$$\boxed{\begin{aligned}&(\tilde{a}_{1_{l-1}} + \tilde{a}_{2_{m-1}} + \tilde{a}_{3_{n-1}})p'' = \\ &\tilde{a}_1(p''_{l+1} - p''_{l-1}) + \tilde{a}_2(p''_{m+1} - p''_{m-1}) + \tilde{a}_3(p''_{n+1} - p''_{n-1}) + \\ &\tilde{a}_{1_{l-1}}p''_{l-2} + \tilde{a}_{2_{m-1}}p''_{m-2} + \tilde{a}_{3_{n-1}}p''_{n-2} - 2b''\end{aligned}} \tag{6.48}$$

where, $b'' = -\frac{v_1^* - v^*_{1_{l-1}}}{\triangle \breve{X}_1} - \frac{v_2^* - v^*_{2_{m-1}}}{\triangle \breve{X}_2} - \frac{v_3^* - v^*_{3_{n-1}}}{\triangle \breve{X}_3}$ and $v_j^* = \breve{v}_j$ for boundary nodes.

With these basic discretized equations now in place, an important point needs to be highlighted. Since the derivations above are based on FDM, even though the equation structures match with those of FVM, they are not exactly identical. For instance, the term b'' in FVM approach reflects the flux and hence, the area of the cell face comes into picture in the expression of b'' but in this case it is not so.

6.6.3 Boundaries

Now that the basic equations for SIMPLER are established, the next step is to have a closer look at the equations for the nodes on and next to the boundaries.

6.6.3.1 Nodes next to inlet boundaries

In eq.6.44, it can be observed that the value of pressure at any interior node is dependent on the pressure of one node downstream and two nodes upstream. If the penultimate nodes next to the inlet boundaries (i.e. nodes immediately next to the boundaries) are considered, information on only one node upstream i.e. the boundary node is available. So, for these nodes, the pressure equation needs to be modified to overcome this issue.

Consider for example Figure 6.3. In this figure, the boundaries at $X_1 = 0$ and $X_3 = 1$ are the inlets. Nodes $10, 19$ lie next to the inlet boundary perpendicular

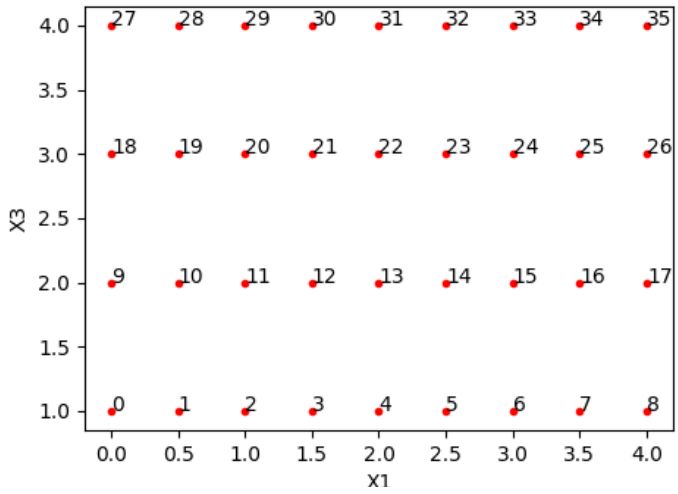


Figure 6.3: Nodes in a 2D plane extracted from a larger 3D domain

to direction of flow of $\breve{v}_1$. If $\breve{v}_1$ at this boundary is known, the pressure equation can be reformulated as

$$
\begin{aligned}
& \tfrac{\breve{v}_1-\breve{v}_{1_{l-1}}}{\triangle X_1}+\tfrac{\breve{v}_2-\breve{v}_{2_{m-1}}}{\triangle X_2}+\tfrac{\breve{v}_3-\breve{v}_{3_{n-1}}}{\triangle X_3}=0 \\
\Rightarrow\ & ((\tilde{v}_1+\tfrac{d_1}{2D}(\breve{p}_{l+1}-\breve{p}_{l-1}))-\breve{v}_{1_{l-1}})/\triangle X_1+ \\
& ((\tilde{v}_2+\tfrac{d_2}{2D}(\breve{p}_{m+1}-\breve{p}_{m-1}))-(\tilde{v}_{2_{m-1}}+\tfrac{d_{2_{m-1}}}{2D}(\breve{p}-\breve{p}_{m-2})))/\triangle X_2+ \\
& ((\tilde{v}_3+\tfrac{d_3}{2D}(\breve{p}_{n+1}-\breve{p}_{n-1}))-(\tilde{v}_{3_{n-1}}+\tfrac{d_{3_{n-1}}}{2D}(\breve{p}-\breve{p}_{n-2})))/\triangle X_3=0 \\
\Rightarrow\ & (\tilde{a}_{2_{m-1}}+\tilde{a}_{3_{n-1}})\breve{p}=\tilde{a}_1(\breve{p}_{l+1}-\breve{p}_{l-1})+\tilde{a}_2(\breve{p}_{m+1}-\breve{p}_{m-1})+\tilde{a}_3(\breve{p}_{n+1}-\breve{p}_{n-1})+ \\
& \tilde{a}_{2_{m-1}}\breve{p}_{m-2}+\tilde{a}_{3_{n-1}}\breve{p}_{n-2}-2\tilde{b}
\end{aligned}
$$

where, $\tilde{b}=-\frac{\tilde{v}_1-\breve{v}_{1_{l-1}}}{\triangle\breve{X}_1}-\frac{\tilde{v}_2-\tilde{v}_{2_{m-1}}}{\triangle\breve{X}_2}-\frac{\tilde{v}_3-\tilde{v}_{3_{n-1}}}{\triangle\breve{X}_3}$.
In this case, it can seen from the first term of the expression for $\tilde{b}$ i.e. $-\frac{\tilde{v}_1-\breve{v}_{1_{l-1}}}{\triangle X_1}$ that for boundary nodes, $\tilde{v}_1=\breve{v}_1$. Similarly, it can be shown that for inlet boundaries perpendicular to $\breve{v}_2$ and $\breve{v}_3$, $\tilde{v}_2=\breve{v}_2$ and $\tilde{v}_3=\breve{v}_3$ respectively. Further the terms $\tilde{a}_{1_{l-1}}$, $\tilde{a}_{2_{m-1}}$ and $\tilde{a}_{3_{n-1}}$ vanish for nodes next to inlet boundaries perpendicular to X_1, X_2 and X_3 axes respectively. In other words, the equation of pressure at the nodes next to inlet boundary perpendicular to X_k can be written as

$$
\boxed{
\begin{aligned}
& (\tilde{a}_{1_{l-1}}\epsilon_{k1}+\tilde{a}_{2_{m-1}}\epsilon_{k2}+\tilde{a}_{3_{n-1}}\epsilon_{k3})\breve{p}= \\
& \tilde{a}_1(\breve{p}_{l+1}-\breve{p}_{l-1})+\tilde{a}_2(\breve{p}_{m+1}-\breve{p}_{m-1})+\tilde{a}_3(\breve{p}_{n+1}-\breve{p}_{n-1})+ \\
& \tilde{a}_{1_{l-1}}\breve{p}_{l-2}\epsilon_{k1}+\tilde{a}_{2_{m-1}}\breve{p}_{m-2}\epsilon_{k2}+\tilde{a}_{3_{n-1}}\breve{p}_{n-2}\epsilon_{k3}-2\tilde{b}
\end{aligned}
} \tag{6.49}
$$

where, $\epsilon_{kj}=1-\delta_{kj}$. Similarly, the pressure correction equation for these nodes also

Table 6.1: Values of $\tilde{b}$ for nodes next to inlets perpendicular to X_k

k	$\tilde{b}$	b''
1	$-\left[\frac{\tilde{v}_1-\breve{v}_{1_{l-1}}}{\triangle\breve{X}_1}+\frac{\tilde{v}_2-\tilde{v}_{2_{m-1}}}{\triangle\breve{X}_2}+\frac{\tilde{v}_3-\tilde{v}_{3_{n-1}}}{\triangle\breve{X}_3}\right]$	$-\left[\frac{v_1^*-\breve{v}_{1_{l-1}}}{\triangle\breve{X}_1}+\frac{v_2^*-v_{2_{m-1}}^*}{\triangle\breve{X}_2}+\frac{v_3^*-v_{3_{n-1}}^*}{\triangle\breve{X}_3}\right]$
2	$-\left[\frac{\tilde{v}_1-\tilde{v}_{1_{l-1}}}{\triangle\breve{X}_1}+\frac{\tilde{v}_2-\breve{v}_{2_{m-1}}}{\triangle\breve{X}_2}+\frac{\tilde{v}_3-\tilde{v}_{3_{n-1}}}{\triangle\breve{X}_3}\right]$	$-\left[\frac{v_1^*-v_{1_{l-1}}^*}{\triangle\breve{X}_1}+\frac{v_2^*-\breve{v}_{2_{m-1}}}{\triangle\breve{X}_2}+\frac{v_3^*-v_{3_{n-1}}^*}{\triangle\breve{X}_3}\right]$
3	$-\left[\frac{\tilde{v}_1-\tilde{v}_{1_{l-1}}}{\triangle\breve{X}_1}+\frac{\tilde{v}_2-\tilde{v}_{2_{m-1}}}{\triangle\breve{X}_2}+\frac{\tilde{v}_3-\breve{v}_{3_{n-1}}}{\triangle\breve{X}_3}\right]$	$-\left[\frac{v_1^*-v_{1_{l-1}}^*}{\triangle\breve{X}_1}+\frac{v_2^*-v_{2_{m-1}}^*}{\triangle\breve{X}_2}+\frac{v_3^*-\breve{v}_{3_{n-1}}}{\triangle\breve{X}_3}\right]$
1,2	$-\left[\frac{\tilde{v}_1-\breve{v}_{1_{l-1}}}{\triangle\breve{X}_1}+\frac{\tilde{v}_2-\breve{v}_{2_{m-1}}}{\triangle\breve{X}_2}+\frac{\tilde{v}_3-\tilde{v}_{3_{n-1}}}{\triangle\breve{X}_3}\right]$	$-\left[\frac{v_1^*-\breve{v}_{1_{l-1}}}{\triangle\breve{X}_1}+\frac{v_2^*-\breve{v}_{2_{m-1}}}{\triangle\breve{X}_2}+\frac{v_3^*-v_{3_{n-1}}^*}{\triangle\breve{X}_3}\right]$
2,3	$-\left[\frac{\tilde{v}_1-\tilde{v}_{1_{l-1}}}{\triangle\breve{X}_1}+\frac{\tilde{v}_2-\breve{v}_{2_{m-1}}}{\triangle\breve{X}_2}+\frac{\tilde{v}_3-\breve{v}_{3_{n-1}}}{\triangle\breve{X}_3}\right]$	$-\left[\frac{v_1^*-v_{1_{l-1}}^*}{\triangle\breve{X}_1}+\frac{v_2^*-\breve{v}_{2_{m-1}}}{\triangle\breve{X}_2}+\frac{v_3^*-\breve{v}_{3_{n-1}}}{\triangle\breve{X}_3}\right]$
1,3	$-\left[\frac{\tilde{v}_1-\breve{v}_{1_{l-1}}}{\triangle\breve{X}_1}+\frac{\tilde{v}_2-\tilde{v}_{2_{m-1}}}{\triangle\breve{X}_2}+\frac{\tilde{v}_3-\tilde{v}_{3_{n-1}}}{\triangle\breve{X}_3}\right]$	$-\left[\frac{v_1^*-\breve{v}_{1_{l-1}}}{\triangle\breve{X}_1}+\frac{v_2^*-v_{2_{m-1}}^*}{\triangle\breve{X}_2}+\frac{v_3^*-v_{3_{n-1}}^*}{\triangle\breve{X}_3}\right]$
1,2,3	$-\left[\frac{\tilde{v}_1-\breve{v}_{1_{l-1}}}{\triangle\breve{X}_1}+\frac{\tilde{v}_2-\breve{v}_{2_{m-1}}}{\triangle\breve{X}_2}+\frac{\tilde{v}_3-\breve{v}_{3_{n-1}}}{\triangle\breve{X}_3}\right]$	$-\left[\frac{v_1^*-\breve{v}_{1_{l-1}}}{\triangle\breve{X}_1}+\frac{v_2^*-\breve{v}_{2_{m-1}}}{\triangle\breve{X}_2}+\frac{v_3^*-\breve{v}_{3_{n-1}}}{\triangle\breve{X}_3}\right]$

gets modified to

$$\boxed{\begin{aligned}&(\tilde{a}_{1_{l-1}}\epsilon_{k1}+\tilde{a}_{2_{m-1}}\epsilon_{k2}+\tilde{a}_{3_{n-1}}\epsilon_{k3})p''=\\&\tilde{a}_1(p''_{l+1}-p''_{l-1})+\tilde{a}_2(p''_{m+1}-p''_{m-1})+\tilde{a}_3(p''_{n+1}-p''_{n-1})+\\&\tilde{a}_{1_{l-1}}p''_{l-2}\epsilon_{k1}+\tilde{a}_{2_{m-1}}p''_{m-2}\epsilon_{k2}+\tilde{a}_{3_{n-1}}p''_{n-2}\epsilon_{k3}-2b''\end{aligned}} \tag{6.50}$$

It is to be noted the nodes which have more than one surrounding node lying on inlet boundaries will have the influence from all of these boundaries. For instance, for node 10, the surrounding node 9 lies on X_1 and 1 lies on X_3. So, in this case the pressure equation will become

$$\tilde{a}_{2_{m-1}}\breve{p}=\tilde{a}_1(\breve{p}_{l+1}-\breve{p}_{l-1})+\tilde{a}_2(\breve{p}_{m+1}-\breve{p}_{m-1})+\tilde{a}_3(\breve{p}_{n+1}-\breve{p}_{n-1})+\tilde{a}_{2_{m-1}}\breve{p}_{m-2}-2\tilde{b}$$

Table 6.1 shows the value of $\tilde{b}$ and b'' for these nodes. An interesting point to observe is that the node which has its $l-1/m-1/n-1$ all lying on inlets will have the diagonal entry in coefficient matrix A as 0. If Figure 6.3 is visualised as not being part of a 3D domain but a 2D domain in its entirety, then the scenario for node

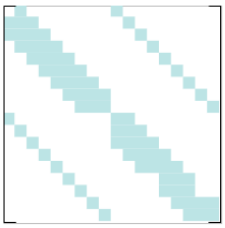


Figure 6.4: Structure of coefficient matrix for pressure equation

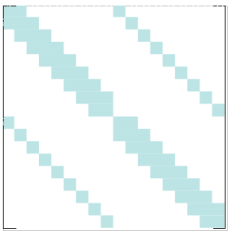


Figure 6.5: Structure of coefficient matrix for velocity equation

10, will be similar i.e. the coefficient of pressure will be 0. However, even though the diagonal entry is 0, it does not have any effect on the numerical performance of BiCGStab and it is still possible to obtain a non-zero value for pressure at this node because it is coupled to other interior nodes as well. Validation of the results for this kind of matrix structure is carried out and the results have been found to match with the theoretical results (refer Appendix B.1). Typical structure of the coefficient matrix, **A** for pressure and velocity equations has been shown in Figure 6.4 and Figure 6.5. The blue and white space in this structure denote non-zero and zero entries respectively.

6.7 Analysis of the discretized equations

Having established the FDEs, the next step is to analyze the behaviour of the equations. Since v'_j is random, an obvious choice for v'_j and p' at the boundaries of the domain would be to assign gaussian data since such data often occur in nature. Yet another reason for choosing gaussian data is many of its statistical properties are known and well established.

As stated earlier, since the mean of the fluctuating component is always 0, the velocity and pressure fields at the boundaries can be assumed to be $v'_j \sim \mathcal{N}(0, \sigma^2_{v_j})$ and $p'_j \sim \mathcal{N}(0, \sigma^2_p)$ where, $\sigma^2_{v_j}$ and σ^2_p are the variances for velocity and pressure fields respectively. The magnitude of the gaussian data generated for velocity components or the pressure at the boundaries depend on these variances and hence, their values could be extremely small or extremely large. Choosing an appropriate scale factor is therefore important so that the numerical analysis can be carried out without excessive loss due to truncation of the floating points.

6.7.1 Impact of under-relaxation factor

It is a known fact that choosing an extremely small value for α might cause the solution to converge extremely slowly. On the other hand, choosing a large value for α often might cause the iterative algorithms to oscillate and even diverge. The choice of α is therefore not only problem dependent but also depends on the magnitude of the variable getting solved. Since, parameters like velocity and pressure are under-relaxed using

$$\Phi_{new} = \alpha\Phi^n + (1-\alpha)\Phi^{n-1}$$

a careful examination of how α influences the progress of the solution is essential. If $\alpha = 1, \Phi_{new} = \Phi^n$ and if $\alpha = 0, \Phi_{new} = \Phi^{n-1}$. As the solution reaches convergence, $\Phi^n \approx \Phi^{n-1}$ and thus, $\Phi_{new} \approx \Phi^n$. However, modifying the value of Φ at the end of every iteration mathematically means modifying the coefficient matrix. For instance, from the under-relaxed value of $\breve{v}_j$;

$$\breve{v}_j^n = \frac{\breve{v}_{new}}{\alpha_{v_j}} - \frac{(1-\alpha_{v_j})}{\alpha_{v_j}}\breve{v}_j^{n-1}$$

Substituting this value in the discretized equation for velocity viz. eq.6.43;

$$\begin{aligned} D\breve{v}_j &= \sum a_{nb}\breve{v}_{j_{nb}} + \frac{d_j}{2}.[(\breve{p}_{l+1} - \breve{p}_{l-1})\delta_{j1} + (\breve{p}_{m+1} - \breve{p}_{m-1})\delta_{j2} + (\breve{p}_{n+1} - \breve{p}_{n-1})\delta_{j3}] \\ \Rightarrow \frac{D\breve{v}_{j,new}}{\alpha_{v_j}} &= \sum a_{nb}\breve{v}_{j_{nb}} + \frac{d_j}{2}.[(\breve{p}_{l+1} - \breve{p}_{l-1})\delta_{j1} + (\breve{p}_{m+1} - \breve{p}_{m-1})\delta_{j2} + (\breve{p}_{n+1} - \breve{p}_{n-1})\delta_{j3}] + \\ & \quad \frac{D(1-\alpha_{v_j})}{\alpha_{v_j}}\breve{v}_j^{n-1} \end{aligned}$$

This change in turn influences the pressure equation as well. Numerically, if the range of values of Φ is small (like in our case where focus is on turbulence), this might result in a solution which is erratic even though the residual approaches a value close to zero. Appendix B.1 gives one such example where under-relaxation parameter has been implemented in BiCGStab.

6.7.2 Scale Factors

As discussed earlier, there are three main scale factors viz. the length scale, X_s, the velocity scale, V_s and the pressure scale, P_s. From a preliminary observation of the structure of eq.6.36, it can be envisaged that if the diffusion term is to have

any impact on the equation, it would be possible for very low values of Reynolds number. To achieve the same, the velocity and length scales need to be set to very low values primarily because ν for air is very less ($\simeq 1.5 \times 10^{-5} m^2/s$). If the V_s is set to a large value, then it would require an extremely small X_s which means a finer grid and hence, greater computational time. Alternately, if X_s is set to a large value, it would mean V_s is required to be set to a very small value which would then mean a large value for $\breve{v}_j$ and still larger value for $\overline{V}_j$. Such large scaled values might incur numerical instabilities due to floating point truncation errors. Hence, a balance in choice of the scale needs to be established so that the numerical analysis can be carried out with a reasonable grid size and avoiding any major computational expense without any excessive floating point truncation error.

6.7.2.1 Length Scale

Since the turbulent component is dealt with here, an obvious choice for X_s would be the grid size rather than the domain size. A finer grid size is expected to resolve the small scales of turbulence. Since at this stage, the problem is visualized from the perspective of numerical simulation, the first task is to fix the parameters necessary for proper functioning of the iterative algorithm (BiCGStab) rather than doing a grid refinement analysis. Accordingly, the following criteria can be set for the simulations

$$X_s = \min\{\triangle X_1, \triangle X_2, \triangle X_3\} \text{ or } X_s = \max\{\triangle X_1, \triangle X_2, \triangle X_3\}$$

The basic principle of any discretization strategy is that the aspect ratios of any discretized physical domain are not too far off which means the order of node to node distance in any direction are consistent i.e. $\triangle X_1 \sim \triangle X_2 \sim \triangle X_3$. A reasonable grid to grid distance for the proposed linear model is of the order of 10^0. Hence, it is understandable that $\breve{X}_j \sim 10^{-1}$ or at most $\breve{X}_j \sim 10$ and thus, $10^{-2} < \{\{p, q, r, s\} \sim 1/\triangle \breve{X}_j^2\} < 10^2$.

6.7.2.2 Velocity Scale

From Reynolds decomposition, there are two parts to any of the velocity components, mean, $\overline{V}_j$ and turbulent, v'_j. If eq.6.29 is looked at, it can be seen that the equation is governed by both of these components. Hence, the order of magnitude

of both of these components need to be investigated separately. Theoretically, three scenarios might arise;

1. $v_j' \sim \overline{V}_j$: If for some value of σ_{v_j}, the order of magnitude of v_j' is same as that of $\overline{V}_j$ i.e. $v_j' \sim \overline{V}_j$, fixing a velocity scale is not an issue and a scale $V_s = \min\{\overline{V}_j, v_j'\}$ or $V_s = \max\{\overline{V}_j, v_j'\}$ might be adequate.
2. $v_j' \gg \overline{V}_j$: This scenario might not arise practically and hence is not required to be taken care of in the simulations.
3. $v_j' \ll \overline{V}_j$: This is the most realistic scenario which can actually occur in ABL and is therefore dealt with in detail in this section.

The order of magnitude values of $\overline{V}_1$ is first looked at. The worst cyclonic storms have the maximum wind speed ranging from $200-300km/h$ i.e. $50-80m/s$. From the perspective of numerical analysis, this means practically even in the worst case scenarios $\overline{V}_j \sim 10$ and never reaches a value of 10^2.

If small values of σ_{v_j} say, 0.1 are to be considered, the gaussian data generated will have most values centred near 0. It is worth noting that most random numbers generated using a computer are in fact pseudo-random numbers and not true random numbers. In our case, numpy library of python is used to generate these random numbers which uses the *Ziggurat algorithm*. It is observed that for a large set of random numbers generated, values as low as 10^{-5} were obtained. One can imagine if true random numbers were used or if still larger random data were generated, chances are the order of these values might be even less. For the time being, the implications of the presence of such small values considering the lowest possible value of v_j' to be of the order of 10^{-5} is discussed.

Implications of setting $V_s \sim \overline{V}_1$: If V_s is set to a value consistent with the order of $\overline{V}_1$ i.e. $V_s \sim 10$, this would mean $Re = V_s X_s/\nu \sim 10.10^0/10^{-5} \sim 10^6$. Also, as shown belolw, $\breve{v}_j$ for extremely small values of v_j' will diminish further.

$$\breve{v}_j = \frac{v_j'}{V_s} \sim \frac{10^{-5}}{10} \sim 10^{-6}$$

Since computers can handle only fixed amount of floating point data, this would result in truncation of number of digits after decimal. For instance, consider a number 0.12345678912345678. Dividing by scale factor of 10 should theoretically return

0.012345678912345678 but instead a computer with 64bit floating point number returns 0.012345678912345679 which is slightly different from the original value. If eq.6.42 is looked at from the perspective of iterative algorithms, it is not difficult to notice that simulating such low values of $\breve{v}_j$ might cause the iterations to reach the residual limit even before the actual numerical solution is achieved. Further, if the coefficients of surrounding nodes are examined, almost all of them have terms like $p/(Re.a) \sim p.10^{-6}$. Depending on grid size, even if p reaches value of 10^2, the value of these coefficients are still small thereby implying that the velocity of all the surrounding nodes except $l-1$ node (i.e. the node immediately downstream of the central node in X_1 direction) are loosely coupled to the velocity component of the centre node and hence have little or no effect on its value.

Since $P_s = \rho V_s^2 \sim 10^2$, this means $\breve{p} = p'/P_s$ will behave just like $\breve{v}_j$ i.e. the values will reduce further resulting in truncation of floating points.

Implications of setting $V_s \sim v'_j$: The lowest possible value of v'_j can be 0. Of course, it cannot be set as the scale factor. Instead, any other non-zero value of v'_j can be chosen as the scale factor. If V_s is set to a value as low as 10^{-5}, this would mean Re in this case will be very low i.e. $Re = V_s X_s/\nu \sim 10^{-5}.X_s/10^{-5} \sim X_s$. Re therefore becomes approximately of the order of the length scale. The finer the grid the lower will be the value of Re. This might give an initial impression that choosing such a value of V_s will start bringing in the effect of diffusion in the simulations. But, then when the behaviour of the remaining parameters is investigated, it does not turn out to be so.

The order of magnitude of non-dimensionalized mean velocity, $\breve{V}_1$ becomes

$$\breve{V}_1 = \frac{\overline{V}_1}{V_s} \sim \frac{10}{10^{-5}} \sim 10^6$$

It can be seen this value becomes extremely large. This in turn would mean the parameter, a will be of order 10^6. Situation will complicate further if $\triangle\breve{X}_1 < 1$ in which case $a = (1+\kappa)\breve{V}_1/\triangle\breve{X}_1 \sim 10^6/10^{-1} \sim 10^7$ thereby making it even larger. As a result, parameters like d_j which have a in denominator will become exceedingly small thereby decoupling or loosely coupling the pressure of the surrounding nodes from velocity which will go against the basic physics of the problem. Just like in the previous case, presence of $Re.a \sim X_s.10^6$ in the denominator of the coefficients of velocities of surrounding nodes highlights the dependence of $\breve{v}_j$ primarily on the

$\breve{v}_{j_l-1}$.

Even $\breve{p}$ is also adversely effected. $P_s \sim V_s^2 \sim 10^{-10}$. Thus, $\breve{p} = p'/P_s \sim 10^{10}$ which is an extremely large value. If the discretized momentum equation is looked at, such large values of pressure is not that much critical since it eventually gets multiplied with d_j which is again extremely small. Therefore, computationally it is not an issue. Problem arises when the pressure equation is getting solved. Since, pressure values are of order 10^{10}, there will be excessive loss of floating point data when BiCGStab is used to solve the equation which is not desirable.

Another problem arises when the pressure correction equation is solved. Because of its very nature, the pressure correction values arrived at the end of an iteration of SIMPLER is also exceedingly large. Usually, α_{pc} controls the value of pressure correction. But, if pressure corrections have such large values, setting an appropriate value for α_{pc} becomes difficult. In such a case, even if a very small value for α_{pc} is considered, it might still not be enough to ensure the numerical stability of SIMPLER.

In Table 6.2 and Table 6.3, a detailed analysis of the order of magnitudes of the PDE variables and other parameters of the model depending on the choice of the scale factors are presented. It is evident that in any case i.e. whether $\overline{V}_1$ or v'_j is chosen as our preferred scale, the effect of the diffusion term does not play any major role unless an extremely fine mesh is used which is not desirable for most practical purposes. Further, from the discussions earlier, it can be observed that choosing $V_s \sim v'_j$ has more cons than pros. As such, for any arbitrary mean velocity profile, the possible options for the numerical analysis is by adopting

$$V_s = \max\{\overline{V}_1(X_3)\} \text{ or } V_s = \min\{\overline{V}_1(X_3)\}$$

It has already been mentioned earlier that scaling the small turbulent values with respect to this scale causes some loss in floating point data. It is to be noted that $V_s \in \mathbb{R}$. Upon scaling by a real number, some further floating point error might affect the scaled values. This might not have been a major issue if data dealt with were not small in magnitude i.e. $\lfloor |v'_j| \rfloor > 0$ or $\lfloor |p'_j| \rfloor > 0$ because in that case accuracy of upto three or four digits after decimal might have been acceptable for most practical purposes. However, for small values of σ_v or σ_p like in the present case makes the turbulent components exceedingly small in some cases and $\lfloor v'_j \rfloor = 0$

Table 6.2: Order of magnitudes of different variables

V_s	P_s	X_s	Re	X_j	$\breve{X}_j$	v_j'	$\breve{v}_j$	$\overline{V}_1$	$\breve{V}_1$	$\overline{V}_{1,X_3}$	$\breve{V}_{1,\breve{X}_3}$	p'	$\breve{p}$
1	2	-5 -4 -3 -2 -1 0 1 2	1 2 3 4 5 6 7 8	-5 -4 -3 -2 -1 0 1 2	0	-5	-6	1	0	-2	-8 -7 -6 -5 -4 -3 -2 -1	-5	-7
-5	-10	-5 -4 -3 -2 -1 0 1 2	-5 -4 -3 -2 -1 0 1 2	-5 -4 -3 -2 -1 0 1 2	0	-5	0	1	6	-2	-2 -1 0 1 2 3 4 5	-5	5

or $\lfloor p_j' \rfloor = 0$. In order to have a control over the scaled floating point data, scale factor is chosen in terms of powers of 10. Hence, the scale factors are modified to

$$V_s = 10^{\lfloor \log_{10} \max\{\overline{V}_1(X_3)\} \rfloor} \text{ or } V_s = 10^{\lfloor \log_{10} \min\{\overline{V}_1(X_3)\} \rfloor}$$

where, $\lfloor . \rfloor$ denotes floor function. The reason for choosing the floor value is to ensure that additional floating point errors do not enter into the scaled values and disrupt the gaussian nature of the bounday data.

6.8 Simulation of RDT model

In this section, results of the numerical analysis for the proposed model has been presented. A 2D domain, Ω of size $[0\ m, 50\ m] \times [1\ m, 51\ m]$ is considered. The grid spacing of $\triangle X_1 = 1\ m$ and $\triangle X_3 = 2\ m$ is chosen. For the purpose of boundary conditions, $\sigma^2_{v_1'} = 0.01, \sigma^2_{v_3'} = 0.01, \sigma_{v_1' v_3'} = -0.005$ is considered. A negative value for $\sigma_{v_1' v_3'}$ is assumed because these two parameters are known to exhibit negative correlation. The variance-covariance matrix can therefore be written as

$$\Sigma = \begin{pmatrix} 0.01 & -0.005 \\ -0.005 & 0.01 \end{pmatrix}$$

Table 6.3: Order of magnitudes of different parameters

V_s	P_s	X_s	p,q,r,s	a	$\frac{p}{Re.a}$	b_j	d_j	D	$\tilde{a}$	$\tilde{b}, b''$
1	2	-5	0	0	-1	-14	0	0	0	-6
		-4			-2	-13				
		-3			-3	-12				
		-2			-4	-11				
		-1			-5	-10				
		0			-6	-9				
		1			-7	-8				
		2			-8	-7				
-5	-10	-5	0	6	-1	-2	-6	0	-6	0
		-4			-2	-1				
		-3			-3	0				
		-2			-4	1				
		-1			-5	2				
		0			-6	3				
		1			-7	4				
		2			-8	5				

A set of pseudo-random numbers is generated and applied as velocity boundary conditions. From eq.6.28, it can be observed that $\kappa \sim \left[\frac{v'_j}{\overline{V}_m}\right]^2$. As $\overline{V}_m \gg v'_m \Rightarrow 0 < \kappa < 1$. Thus, a value of $\kappa = 0.1$ is assumed for the time being. It is to be noted that the purpose of the current simulations is to check the behaviour of these linearized equations rather than checking the validity of the assumed variances of the turbulent quantities or the value of κ for that matter. Boundary conditions for pressure fluctuations are assumed as gaussian as well i.e. $p' \sim \mathcal{N}(0, 0.01)$.

To get a proper insight into the behaviour of the model and to ensure ergodicity, the correct approach would be to run a large number of simulations keeping the $\overline{V}_1$ but random v'_j for each simulation. To achieve this, the simulation is run 500 times such that for every simulation, the boundary conditions assigned are random and as such do not bear resemblance with one another. A typical set of boundary conditions for v'_1, v'_3, p' for one such realization is shown in Figure 6.6 to Figure 6.8.

6.8.1 Convergence criteria

An under-relaxation factor of 0.05 is found to be suitable for the current simulations. Since, the magnitude of the turbulent components itself is small, choosing an appropriate convergence criteria is vital. It is evident from the way SIMPLER works that

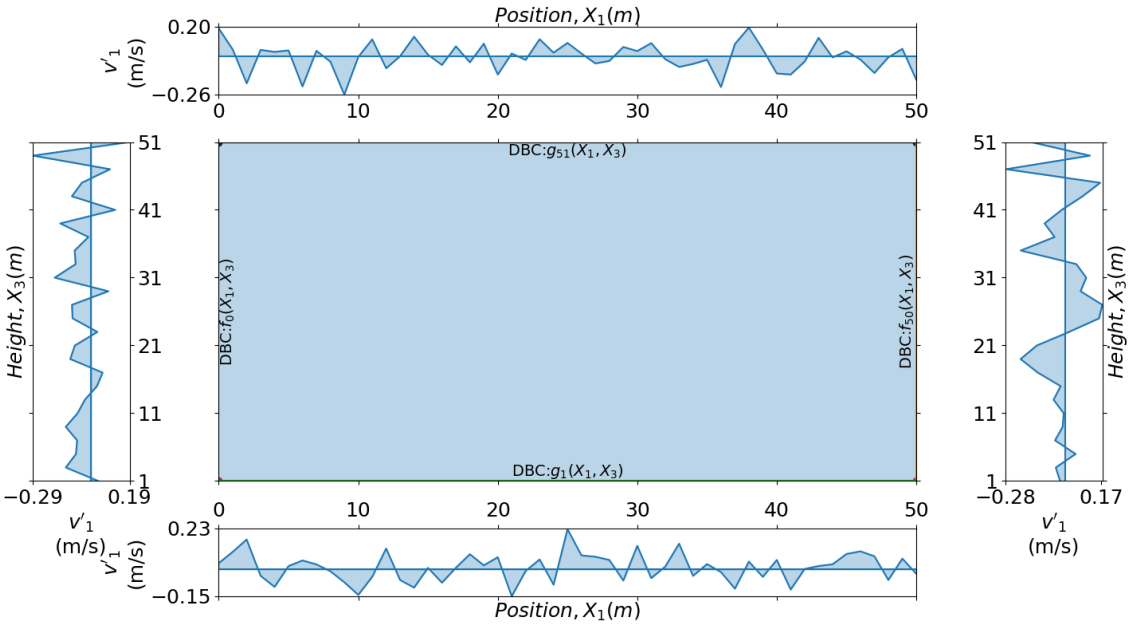


Figure 6.6: Boundary conditions for v'_1

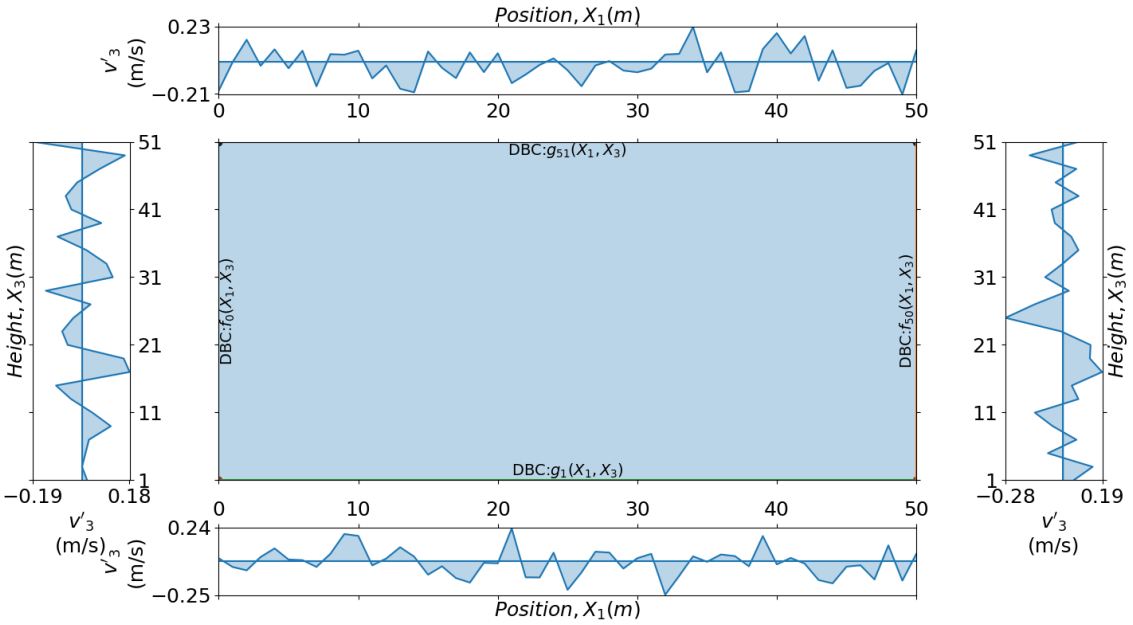


Figure 6.7: Boundary conditions for v'_3

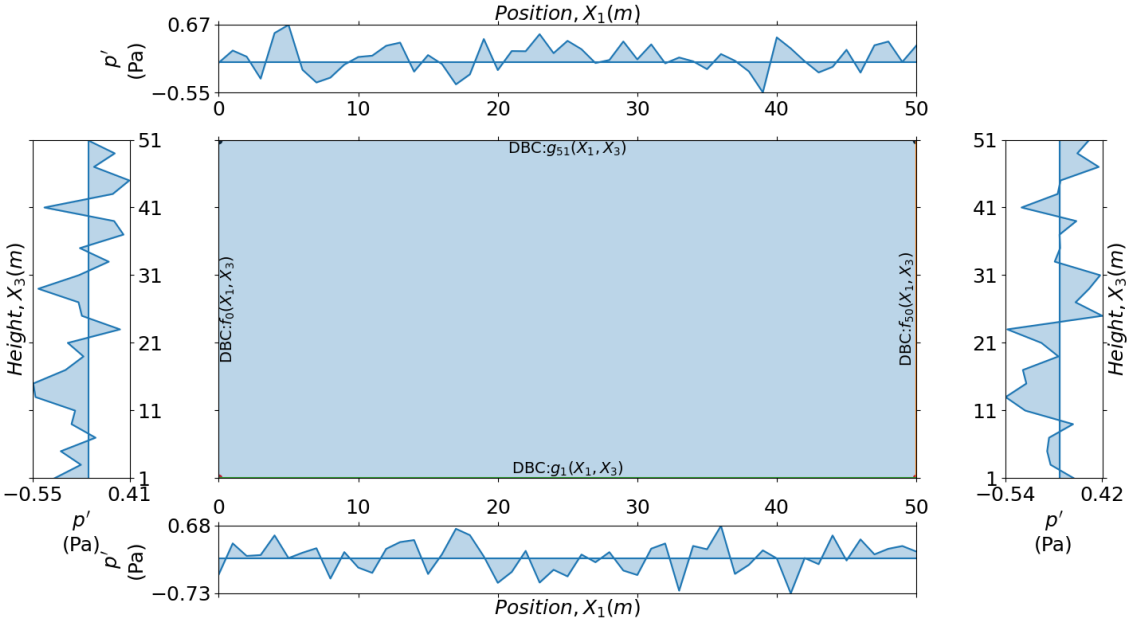


Figure 6.8: Boundary conditions for p'

for every iteration of SIMPLER, there are four set of iterations running BiCGStab for computing v_1', v_3', p' and p''. As such, the convergence criteria is required to be set at two levels, one for BiCGStab and another for iterations of SIMPLER itself. Accordingly, the following criteria has been set for these iterations;

1. for BiCGStab, iteration terminates when the norm of the residual vector, $||\mathbf{r}||_2 < 0.01$
2. for SIMPLER, iteration terminates when

$$\left| \frac{||\breve{\mathbf{\Phi}}||_2^l - ||\breve{\mathbf{\Phi}}||_2^{l-1}}{||\breve{\mathbf{\Phi}}||_2^{l-1}} \right| \times 100 < 5$$

where, $\breve{\mathbf{\Phi}} = \{\breve{\mathbf{v}}_1, \breve{\mathbf{v}}_3, \breve{\mathbf{p}}\}$, $\breve{\mathbf{v}}_1, \breve{\mathbf{v}}_3, \breve{\mathbf{p}}$ are column vectors consisting of $\breve{v}_1, \breve{v}_3, \breve{p}$ respectively at all internal nodes and l = iteration no. of SIMPLER

6.8.2 Results

Since, the simulations are carried out for an ensemble of experiments with random boundary conditions, the behaviour of the proposed model is best visualized by plotting histograms. For this purpose, data from two sets of nodes are chosen to be analyzed viz.

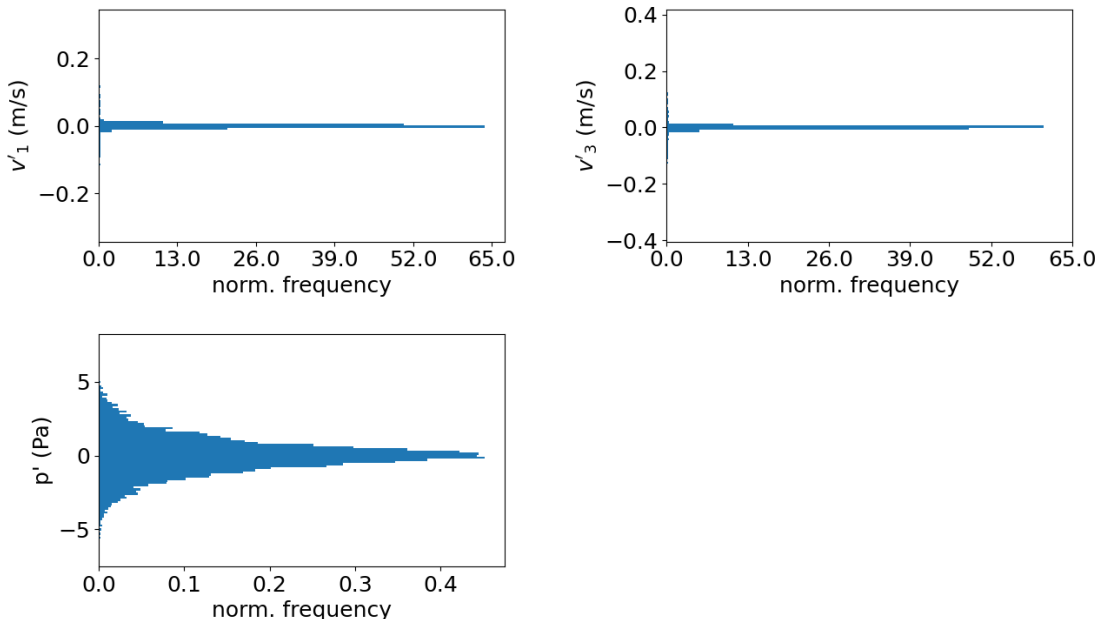


Figure 6.9: Distribution along $X_1 = 1\ m$

1. along the nodes next to the boundaries: $X_1 = \{1\ m, 49\ m\}$ and $X_3 = \{3\ m, 49\ m\}$. The reason for choosing these set of nodes is that the form of the discretized equations for these nodes are different from any other interior nodes as seen in Section 6.6.3.
2. along any internal set of nodes: for this purpose a set of centrally located nodes is chosen i.e. $X_1 = \{25\ m\}$ and $X_3 = \{25\ m\}$

The histograms are shown in Figure 6.9 to Figure 6.14. The histograms are normalized such that the total area under the histograms equals to unity. It can be observed that the mean of the distribution is always near to 0 for all the three physical parameters i.e. v'_1, v'_3, p'. This justifies that the way the equations have been discretized and SIMPLER algorithm has been implemented does not change the intended behaviour of the model and the dependent physical parameters.

The ensemble mean, averaged over the number of experiments along a typical internal line of nodes is shown in Figure 6.15 and Figure 6.16. As stated earlier, the mean of the turbulent quantities should be 0. The simulations can be said to work properly if this value of the mean is achieved. It can be observed from the results that the ensemble mean is near to 0 and not too far off which is in line with the behaviour expected of the model.

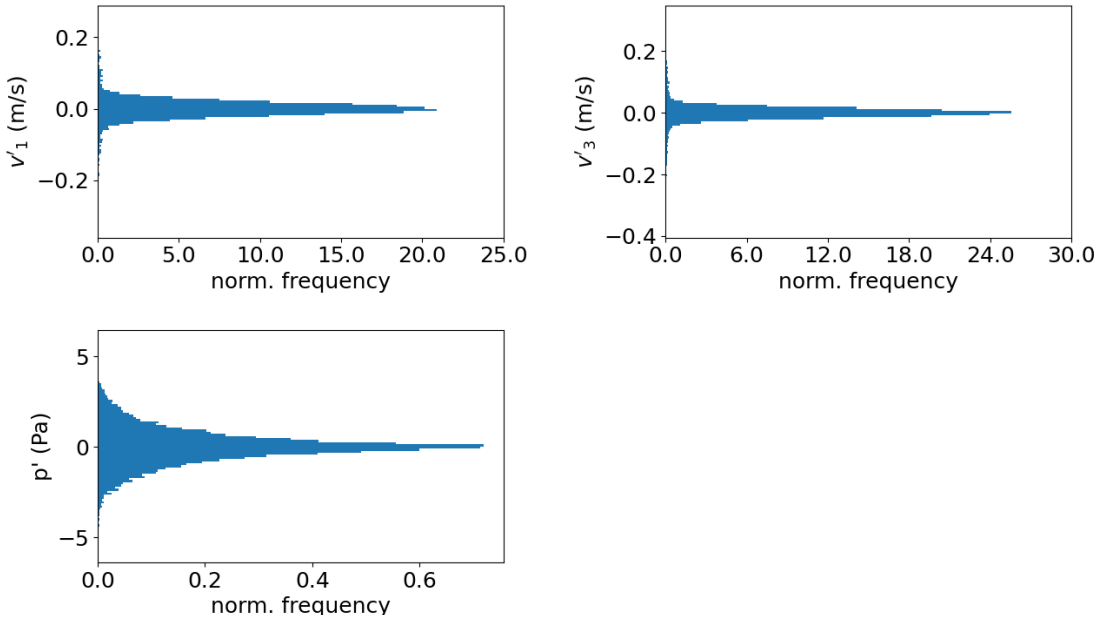


Figure 6.10: Distribution along $X_1 = 25\ m$

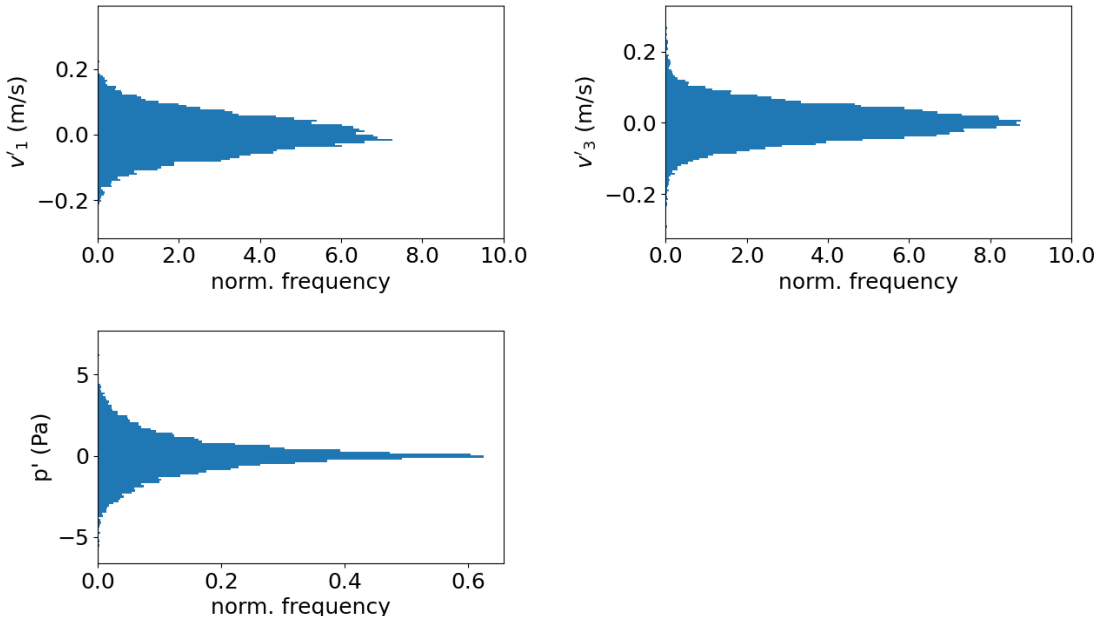


Figure 6.11: Distribution along $X_1 = 49\ m$

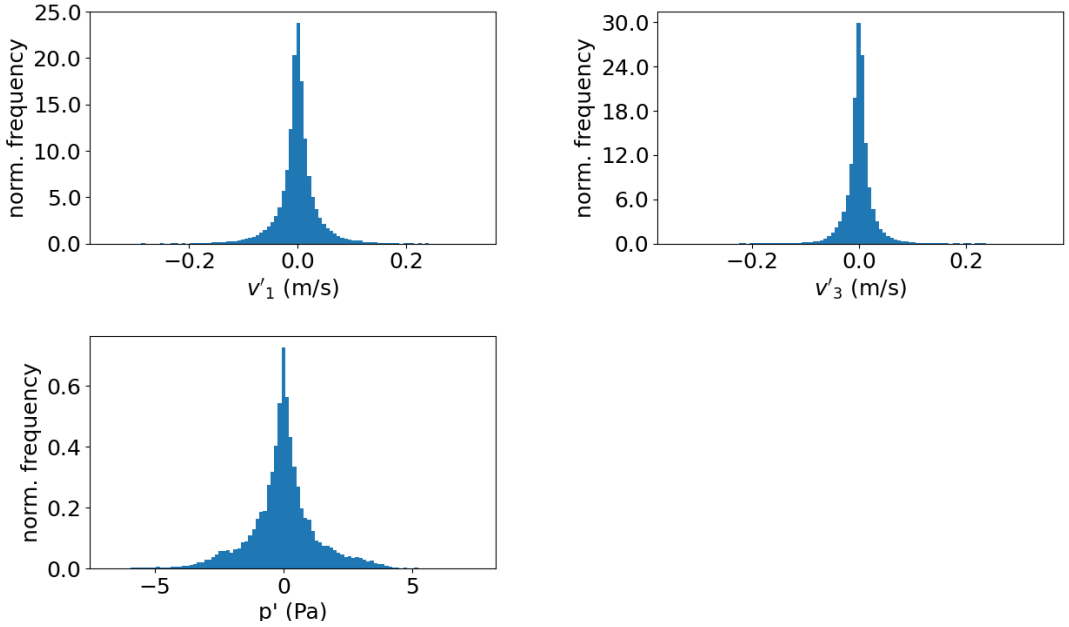


Figure 6.12: Distribution along $X_3 = 3\ m$

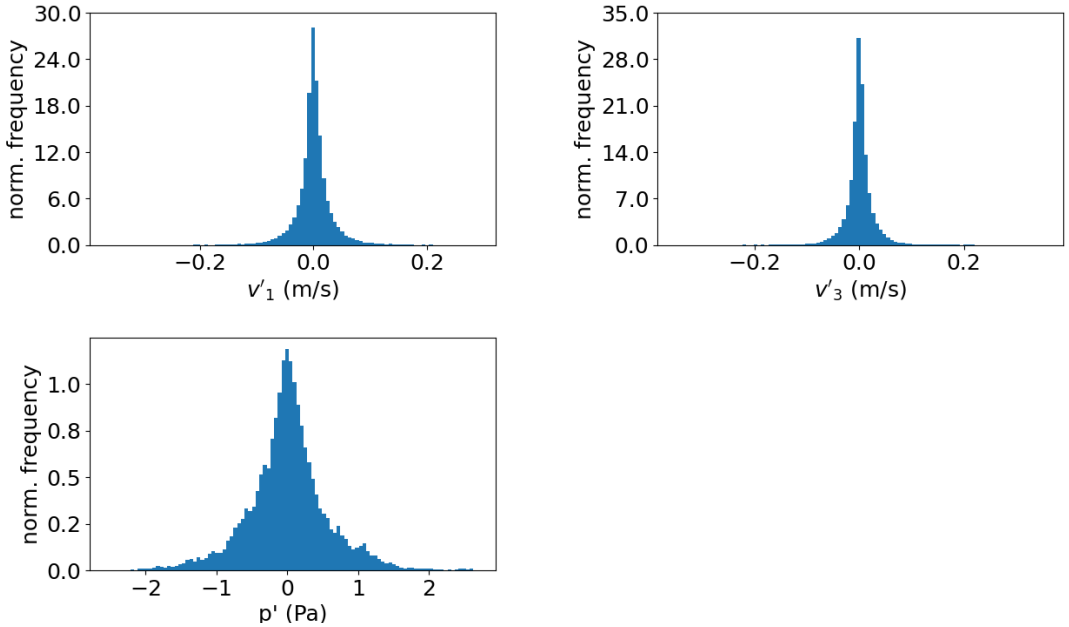


Figure 6.13: Distribution along $X_3 = 25\ m$

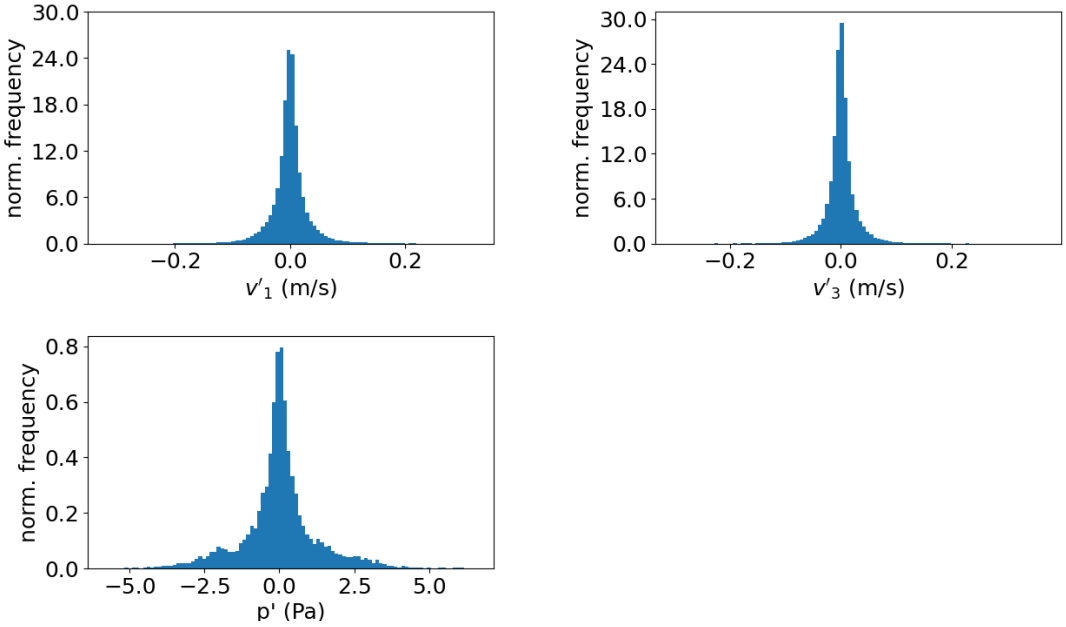


Figure 6.14: Distribution along $X_3 = 49\ m$

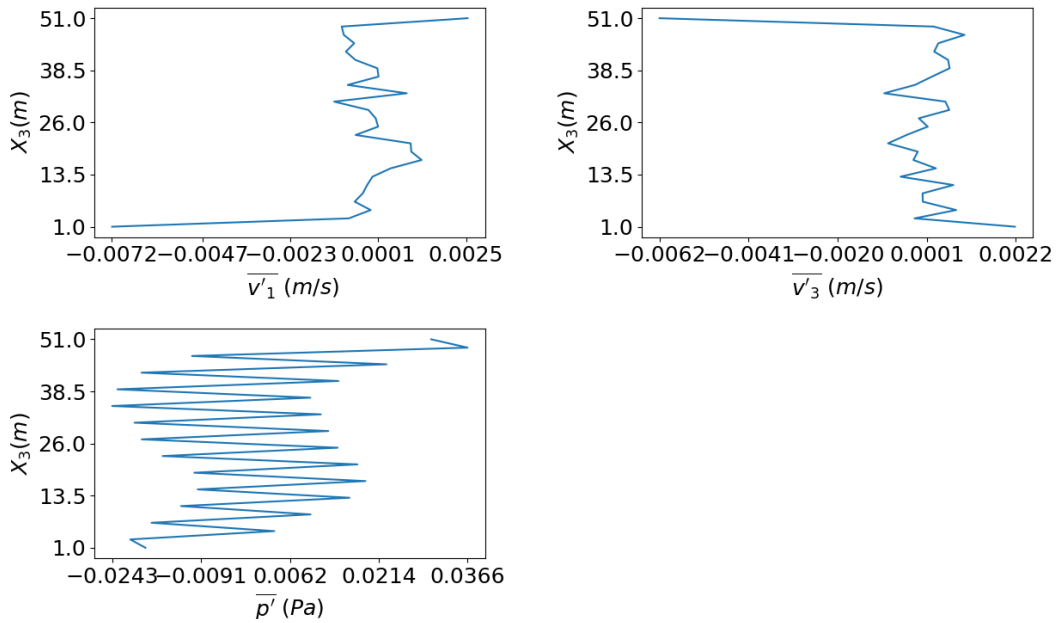


Figure 6.15: Mean along $X_1 = 25\ m$

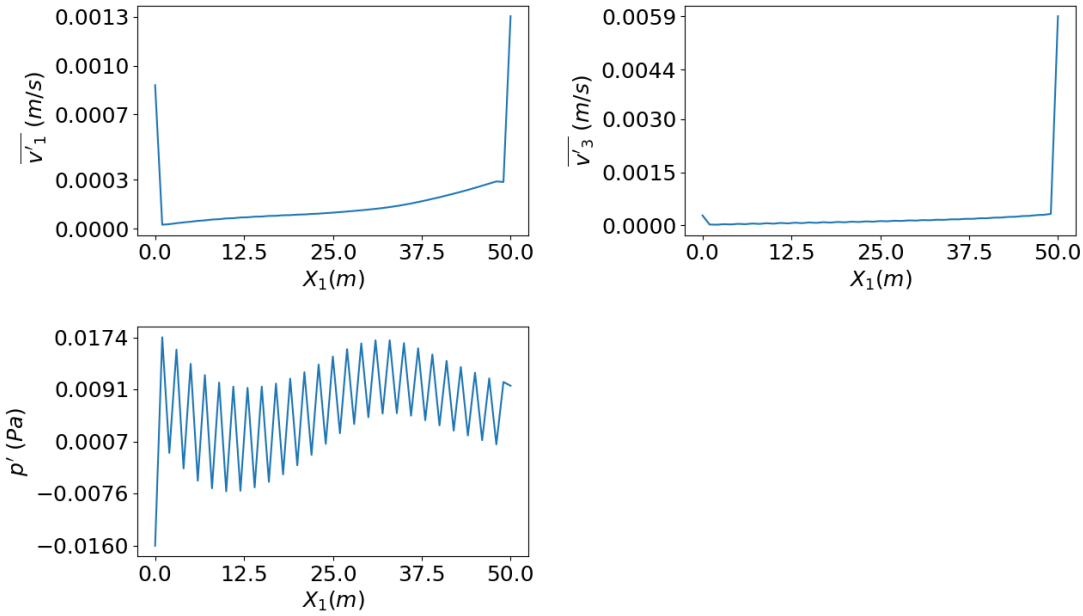


Figure 6.16: Mean along $X_3 = 25\ m$

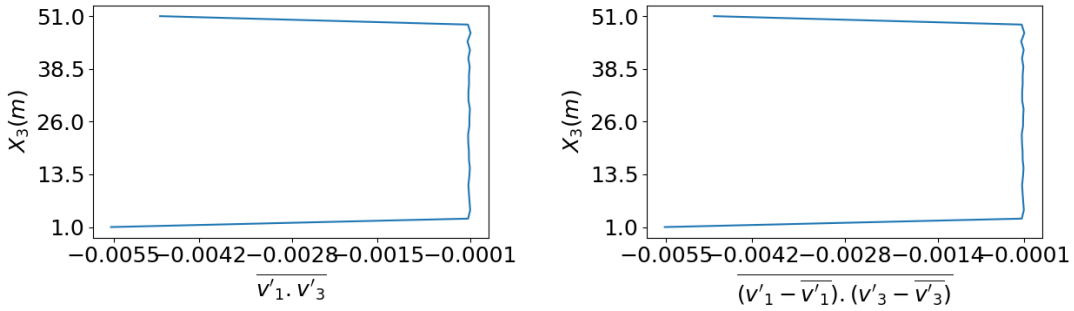


Figure 6.17: covariance along $X_1 = 25\ m$

Apart from the mean the nature of the covariance of v_1' and v_3' also needs to be verified. The results of the numerical simulation should exhibit a negative correlation between these two parameters. Figure 6.17 and Figure 6.18 clearly show that the covariance between these two parameters is negative. Additionally, in Figure 6.19 results of 4 simulations chosen randomly from the 500 experiments are presented. v_1' and v_3' clearly exhibit a negative correlation for each of this individual simulation.

Continuity check is carried out at the end of every iteration of SIMPLER. It is found that continuity equation when multiplied with the appropriate scale factor i.e. V_s/X_s (refer eq.6.33) is satisfied for all the internal nodes. This justifies that the

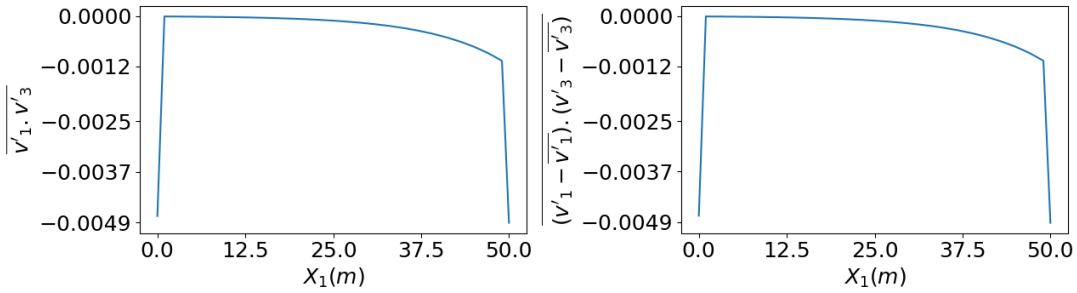


Figure 6.18: Covariance along $X_3 = 25\ m$

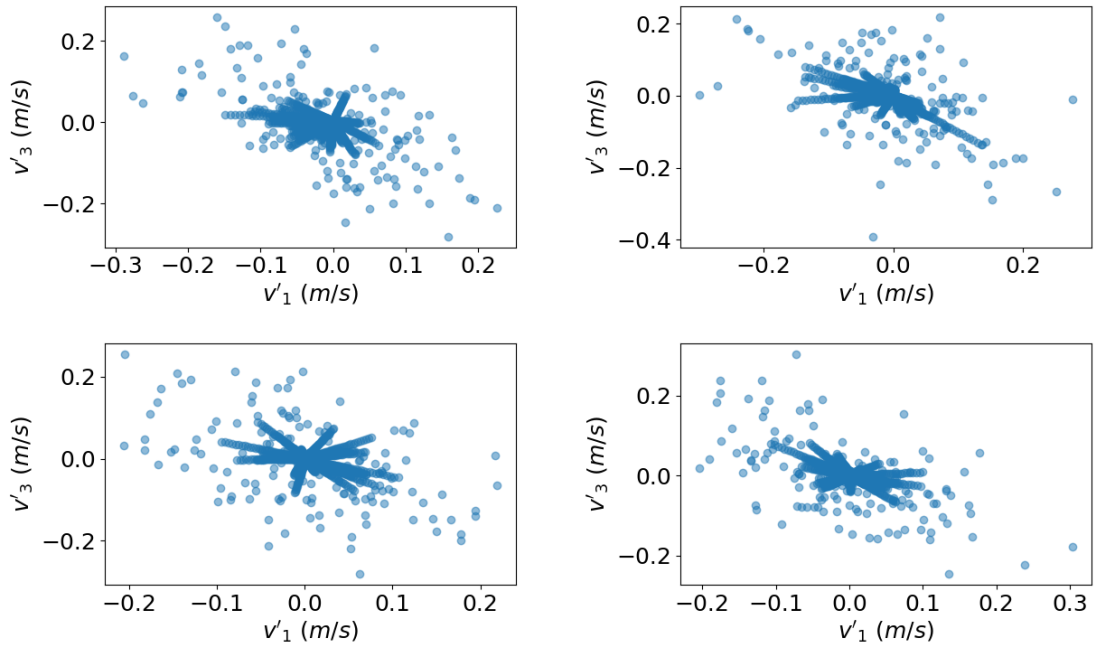


Figure 6.19: Nature of covariance for 4 simulations chosen randomly out of the total 500 simulations

choice of the convergence criteria set above is accurate enough for these simulations.

6.9 Discussions

In this chapter, a linearized model for turbulence has been presented. The model generates turbulent flow with characterization (mean, variance) and can be used to simulate turbulent time series at a desired loaction in the wind farm using the wind spectrum. It is to be noted that the model is rendered invalid if $\overline{V}_1 = 0$. The effect of this is that both the components of the linearized term i.e. $\breve{V}_1 \frac{\partial \breve{v}_j}{\partial \breve{X}_1}$ and $\breve{v}_3 \frac{\mathrm{d}\breve{V}_j}{\mathrm{d}\breve{X}_3} \delta_{j1}$ in eq.6.36 will become 0 which in turn would imply the directional derivative of the turbulent component of velocity will have no effect on the flow whatsoever. This sort of behaviour is physically inconsistent and unexpected. As such, the proposed model cannot be used for such cases where the mean of the total velocity is 0.

Simulations have been carried out for this new proposed RDT model and the results are seen to behave in the expected way both in terms of the gaussian nature of the assumed model as well as the mean and correlation between the velocity components. However, the model needs further refinement in quite a few aspects. This includes computing a proper value for κ which for the time being has been assumed arbitrarily based on the relative order of magnitude of the mean and turbulent components of velocity. However, the appropriate value for the same needs to be computed statistically instead of assuming it heuristically.

An important validation of this model can be carried out by simulating the actual non-linear form of the equation as well i.e. using SIMPLER with eq.6.21 and comparing the behaviour of the results obtained and presented above for the linearized equation. A grid refinement analysis just like the one performed for DIIM in Chapter 5 would be useful to get a better insight into the numerical stability and the choice of scales required.

For any numerical simulation, an important test in terms of the stability and convergence of the discretized equation and the algorithm implemented is by simulating the equations for low magnitudes of boundary values. If the solution diverges, the algorithms adopted or the discretized equations used can be said to be numerically inconsistent. In the simulations presented above, the magnitudes of the values at the boundaries are low and the solutions converge nicely as per the

convergence criteria set for each of the 500 simulations. As such, the discretized equations can be said to have passed this test.

The domain currently analysed is relatively smaller in size and has less number of nodes in it. Numerical complications might arise if a relatively larger domain like the one considered in Chapter 5 is taken up for study. A larger domain would mean greater number of iterations and hence, greater chance of floating point truncation error at each step of iteration both at the level of BiCGStab as well as at the level of SIMPLER. For low values of the velocity and pressure fluctuations at the boundary, it would be interesting to see if there is an excessive loss of floating point data and whether it causes the path of the conjugate gradient to deviate away from the solution and if such scenario arises how to mitigate the same numerically by appropriately modifying the discretized structure of the equation without causing it to deviate away from the actual PDE model.

This model is currently implemented for the aerodynamics of ABL only. The behaviour of this model for other flow types which have turbulence has not been investigated. Even for ABL, to be able to implement it in the context of wind farms, the boundary conditions or the way the actuator disc is to be modelled is needed to be looked into greater detail, a study which has not yet been undertaken. An option for modelling the same would be to frame the proposed linear model in line with the IIM strategy which has been used in Chapter 5 and which has been used by researchers in the context of Navier-Stokes equation as well.

Chapter 7

Conclusions and future work

7.1 General summary

The focus of the present thesis is fluid-structure interaction in wind farms. A detailed review of different wind farm models has been presented. For the purpose of studying the aerodynamics, research on different fluid dynamics aspects including fluid-structure interaction, turbulence and numerical algorithms have been undertaken. Further, to be able to handle large scale computational data, a review of parallel numerical algorithms, message passing interface and GPGPU has been considered including their application in CFD analysis of wind farms. A detailed discussion on FDM and FVM has been presented. The way these methods can be implemented in a parallel setting for elliptic or any other form of PDE has been discussed in detail using strategies like multi-colouring algorithms and these concepts have been used for simulating 2D wind fields.

7.2 Main findings

Based on the research undertaken, it can be concluded that

1. Though potential flow model is still a widely accepted model in many wind farm studies, the model proposed in this thesis works with mean velocity components as the dependent variables in elliptic equations which is much more realistic as it can consider the effects of vorticity. Not only this, the

proposed model provides much more flexibility in defining the 2D geometric model of wind farms and in a domain with complex geometry.

2. The classical actuator disc theory does provide the approximate power produced from a single wind turbine. However, since it does not consider the effects of the blades of the wind turbine the velocity profile in the wake of the actuator is a gross approximation.
3. In a similar way, the analytical wind farm models also do not take into account the presence of blades again. This introduces error in velocity profile in the wake of the first actuator. This error assimilates resulting in still further deviation from the actual values. Though these models have an important role to play during the analysis in preliminary engineering stage, it cannot be denied that such crude approximations might result in quite erroneous results.
4. The approach proposed in this thesis i.e. the methodologies like DIIM offers a convenient way to implement FSI in the context of 2D simulations and helps in getting way more realistic results as it considers the effects of the presence of blades in the wind field. Additionally, this approach requires relatively less computational resources to carry out CFD analysis compared to the analysis with blade flexibility yet providing reasonable accuracy in terms of the wake generated in the wind field.
5. From the perspective of numerical analysis, DIIM for elliptic PDEs have been solved earlier by researchers by using Gauss-Seidel iterations. But, when it comes to simulate a large linear algebraic system, iterative algorithms like PBiCGStab are much more efficient. The approach of DIIM in modifying the residue at the end of each iteration is an important step which is not in line with the way iterative algorithms work. Thus, in this research a modification has been proposed to the algorithm such that the change in residue at the end of each iteration can be conveniently accommodated. This modification has also been shown to perform quite nicely in this present work, thereby broadening the spectrum of the kind of iterative algorithms which can be used for implementing DIIM.
6. The work carried out on turbulence modelling implements Gaussian closure to linearize Navier Stokes equations using Rapid Distortion Theory. The nature of the proposed PDE is non-conventional from the perspective of fluid dynamics. It is therefore essential to study the numerical behaviour of the FDE approximation of the proposed PDE. For this reason, in this thesis the

system in steady state is studied first for numerical stability. Graphical plots of the simulated results reflect the gaussian nature with the mean and variance of the simulated results for pressure and velocity components consistent with the original equation. However, a more robust approach would be to use statistical tests like Kolmogorov-Smirnov test or Shapiro-Wilk test that can be used to further ascertain the nature of the distribution of the numerical results.

Though the PDE have turbulent components of the velocity as the dependent variable, turbulence is a time-dependent phenomenon and as such the numerical analysis of the steady state form of the proposed equation does not fully represent the true nature of the turbulence. Now that the behaviour of the FDE approximation of the PDE has been ensured, the unsteady state model can be simulated. In case of one-dimensional flow, the probability density function defining the turbulent component usually has a simple form. However, the statistics of the flow becomes more complex when the flow becomes unsteady with two or three dimensions because the joint probability distribution of the turbulent components come into picture which do not have a simple form. This is where Fourier transform, and characteristic functions come into play. A spectral analysis can be carried out to determine the frequency content of the velocity field which can then be used to test the statistics of the flow.

7. The laminar flow model with vorticity and wake interaction and the spatial turbulence model can form the basis of input to an individual wind turbine in a wind farm for subsequent analysis.

7.3 Future work

There are still certain areas which can be explored further in future in order to improve the models and develop a better understanding into their physical behaviour Some of these aspects are:

1. The vorticity considered in this study is assumed to be constant for analysing the scenarios. Future work with other vortex functions can also be conducted and the behaviour of the model can be studied in order to better understand and demonstrate its behaviour.

2. Fluid-structure interaction method like DIIM has been used to model the actuators which are assumed as rectangular zones wherein the behaviours of the fluid is assumed to be different from the overall global domain. But, this assumption of the interface as a rectangle is purely heuristic and requires further refinement. The correct definition of the interface incorporating the blade behaviour and its effect on the wind would be yet another aspect which can be investigated in a greater depth. Further, a study of DIIM with respect to other analytical wind farm models is another research which can be taken up in future.
3. FVM used for SIMPLER algorithms usually use a grid staggered with respect to velocity grids where the value of the pressure is computed. A similar approach can be tried later for the RDT model proposed with FDM. At present, all the fluctuations in velocity and pressure boundary conditions are assumed. As such, this does no correlate the velocity and pressure in any way. An investigation into obtaining a more realistic data by correlating these two physical parameters would improve the model further, a work that can be later looked into in greater detail.
4. It is worth noting that from the perspective of numerical simulation, the proposed RDT can be parallelized further by using the greedy multi-coloring algorithm. However, the number of colours is more than two and whether or not it will really help in achieving a significant benefit in computation time would be something worth working with. A preconditioned version of BiCGStab can also be looked into just like the way it has been done for elliptic PDEs.

Appendix A

FDE for elliptic PDEs

A.1 Finite Difference Equations

In this appendix, the finite difference equations for the nodes inside a 2D domain governed by elliptic PDEs are presented. The coefficients of the linear system of equations of unknown variables using the standard 5-point stencil (Figure 3.3) in 2D is known. This standard case is referred to as Case-1 (eq.3.22) in this appendix. However, if the nodes are just next to the boundary nodes, the coefficients of the linear system changes depending on the nature of the boundary conditions (Dirichlet or Neumann). In this appendix, the equations for all possible cases of nodes adjacent to the domain boundary, depending on the status of their surrounding boundary nodes are presented. It is to be noted that the cases discussed below are w.r.t. $X_1 - X_3$ plane only and hence, the notations X_1 (refers to horizontal axis) and X_3 (refers to vertical axis) have been used as appropriate. In case any other plane is chosen, everything will remain the same and only horizontal (X_1) and vertical (X_3) axes notation need to be replaced with the horizontal and vertical axis of the chosen plane.

In Figure A.1, a 2D domain of dimensions $[0, 5] \times [0, 4]$ is considered with NBC at $X_1 = 0$ and $X_3 = 1$. At the remaining two boundaries viz. $X_1 = 5$ and $X_3 = 4$, it is assumed that DBC are known. The nodes indicated by ● are the nodes which form part of the domain including the boundary and those indicated with $\times$ indicate the ghost nodes, the meaning and significance of which is discussed later.

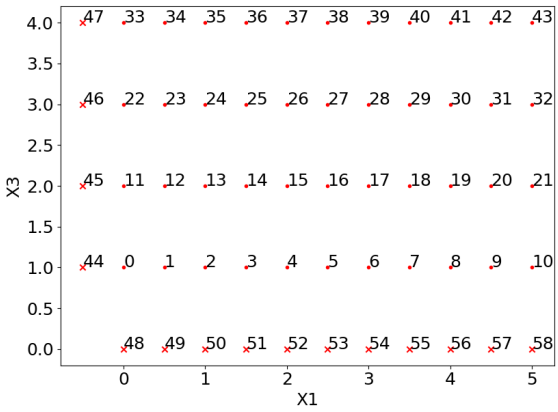


Figure A.1: A typical arrangement of nodes in a 2D domain showing ghost nodes

This section provides the values of coefficients in the coefficient matrix **A** and the right hand side term for the nodes adjacent to the boundary nodes. Here, $cent, bo, le, ri, to$ refers to central, bottom, left, right, top nodes respectively w.r.t. a standard 5-point stencil. Suffix 'D' and 'N' refer to the Dirichlet and Neumann boundary conditions respectively at the bo, le, ri, to nodes. The cases presented below are for penultimate nodes where the equation to be solved gets modified as discussed in Section 3.3.2. Two approaches have been presented here.

A.1.1 One sided differencing

A penultimate node next to the bottom boundary defined by NBC is governed by the following equation

$$N_{bo} = \phi_{X_3}|_{bo} = \frac{-3\phi_{bo} + 4\phi_{cent} - \phi_{to}}{2 \triangle X_3} \Rightarrow -3\phi_{bo} + 4\phi_{cent} - \phi_{to} - 2N_{bo} \triangle X_3 = 0$$

Additionally, the flow is also governed by the equation (i.e. central difference equation for elliptic PDE)

$$b - s\phi_{cent} - p(\phi_{ri} + \phi_{le}) - r(\phi_{to} + \phi_{bo}) = 0$$

Since, ϕ_b is not known, it can be eliminated from these two equations to obtain the following equation for $\phi_{i,j}$

$$(4r + 3s)\phi_{cent} + 2q\phi_{to} + 3p(\phi_{le} + \phi_{ri}) + (-3b - 2r \triangle X_3 N_b) = 0$$

Table A.1: FDEs (one or more surrounding boundary nodes have NBC)

Case	Desc.	Coefficients of					Known value
		ϕ_{cent}	ϕ_{bo}	ϕ_{le}	ϕ_{ri}	ϕ_{to}	
51/71	bo_N	$4r+3s$	$-$	$3p$	$3p$	$2r$	$-3b-2N_{bo}r \triangle X_3$
52/72	le_N	$4p+3s$	$3r$	$-$	$2p$	$3r$	$-3b-2N_{le}p \triangle X_1$
53/73	ri_N	$4p+3s$	$3r$	$2p$	$-$	$3r$	$-3b+2N_{ri}p \triangle X_1$
54/74	to_N	$4r+3s$	$2r$	$3p$	$3p$	$-$	$-3b+2N_{to}r \triangle X_3$
55	bo_N, le_N	$4p+4r+3s$	$-$	$-$	$2p$	$2r$	$-3b-2N_{le}p \triangle X_1-2N_{bo}r \triangle X_3$
56	bo_N, ri_N	$4p+4r+3s$	$-$	$2p$	$-$	$2r$	$-3b+2N_{ri}p \triangle X_1-2N_{bo}r \triangle X_3$
57	to_N, le_N	$4p+4r+3s$	$2r$	$-$	$2p$	$-$	$-3b-2N_{le}p \triangle X_1+2N_{to}r \triangle X_3$
58	to_N, ri_N	$4p+4r+3s$	$2r$	$2p$	$-$	$-$	$-3b+2N_{ri}p \triangle X_1+2N_{to}r \triangle X_3$
59	le_N, ri_N	$4p+2s$	$2r$	$-$	$-$	$2r$	$-2b-N_{le}p \triangle X_1+N_{ri}p \triangle X_1$
60	bo_N, to_N	$4r+2s$	$-$	$2p$	$2p$	$-$	$-2b-N_{bo}r \triangle X_3+N_{to}r \triangle X_3$
61	bo_N, le_N, ri_N	$12p+8r+6s$	$-$	$-$	$-$	$4r$	$-6b-3N_{le}p \triangle X_1+3N_{ri}p \triangle X_1-4N_{bo}r \triangle X_3$
62	bo_N, ri_N, to_N	$8p+12r+6s$	$-$	$4p$	$-$	$-$	$-6b+4N_{ri}p \triangle X_1-3N_{bo}r \triangle X_3+3N_{to}r \triangle X_3$
63	le_N, ri_N, to_N	$12p+8r+6s$	$4r$	$-$	$-$	$-$	$-6b-3N_{le}p \triangle X_1+3N_{ri}p \triangle X_1+4N_{to}r \triangle X_3$
64	bo_N, le_N, to_N	$8p+12r+6s$	$-$	$-$	$4p$	$-$	$-6b-4N_{le}p \triangle X_1-3N_{bo}r \triangle X_3+3N_{to}r \triangle X_3$

This corresponds to Case-51 in Table A.1. It is to be noted that cases 71-74 are the penultimate nodes located just next to the internal corner nodes i.e. where the internal angle at the corner node is $3\pi/2^c$. As is evident, this approach includes strictly only the internal nodes in the system of equations to be solved and the boundary nodes with NBC are excluded when the equations are getting solved.

It is interesting to note that using this approach does have a drawback. The values of the boundary nodes with NBC can be computed quite conveniently from the values of ϕ of the internal nodes by using the one-sided differencing equation for NBC. However, in a case like Figure A.1 where the corner node is at the intersection of two bounding surfaces defined by NBC, the value of ϕ_0 can be computed in two ways, either by using $N_{bo}, \phi_{11}, \phi_{22}$ or by using N_{le}, ϕ_1, ϕ_2. Since, the values of ϕ at boundary nodes is not solved as part of the equation system, there is no guarantee that the values of ϕ_0 obtained by using either the left boundary or the bottom boundary would match.

A.1.2 Central differencing

An alternate approach to formulate the linear system of equations is by using the equations of central differencing for first order derivatives i.e. NBC. But, as information on the nodes just outside the domain are not available, this approach cannot be applied straightaway. This hurdle is overcome by assuming a set of nodes just outside these boundaries which are usually referred to as 'ghost nodes'. The

Table A.2: FDEs (Boundary nodes with NBC incl. in eqn. system)

Tag	**Desc.** Node on	**Coefficients of** ϕ_{cent}	ϕ_{bo}	ϕ_{le}	ϕ_{ri}	ϕ_{to}	**Known value**
1	bottom bdr.	s	$-$	p	p	$2r$	$-b - 2N_{bo}r \triangle X_3$
2	left bdr.	s	r	$-$	$2p$	r	$-b - 2N_{le}p \triangle X_1$
3	right bdr.	s	r	$2p$	$-$	r	$-b + 2N_{ri}p \triangle X_1$
4	top bdr.	s	$2r$	p	p	$-$	$-b + 2N_{to}r \triangle X_3$

Table A.3: FDEs (Corner nodes with NBC incl. in eqn. system)

Tag	**Desc.** Corner on	**Coefficients of** ϕ_{cent}	ϕ_{bo}	ϕ_{le}	ϕ_{ri}	ϕ_{to}	**Known value**
1	bottom left	s	$-$	$-$	$2p$	$2r$	$-b - 2N_{bo}p \triangle X_1 - 2N_{le}r \triangle X_3$
2	bottom right	s	$-$	$2p$	$-$	$2r$	$-b + 2N_{bo}p \triangle X_1 - 2N_{re}r \triangle X_3$
3	top left	s	$2r$	$-$	$2p$	$-$	$-b - 2N_{to}p \triangle X_1 + 2N_{le}r \triangle X_3$
4	top right	s	$2r$	$2p$	$-$	$-$	$-b + 2N_{to}p \triangle X_1 + 2N_{ri}r \triangle X_3$

nodes $44 - 58$ in Figure A.1 are the ghost nodes. With respect to a node on bottom boundary, the governing equations are

$$b - s\phi_{cent} - p(\phi_{ri} + \phi_{le}) - r(\phi_{to} + \phi_{bo,gh}) = 0$$

where the gh in the suffix indicates ghost node. Using second order central differencing to resolve for first order derivative, the following equation for NBC can be written.

$$N_{bo} = \frac{\phi_{to} - \phi_{bo}}{2 \triangle X_1} \Rightarrow 2N_{bo} \triangle X_1 - \phi_{to} + \phi_{bo,gh} = 0$$

Eliminating $\phi_{bo,gh}$ from these equations gives

$$-b - 2N_{bo,gh}r \triangle X_3 + s\phi_{cent} - p(\phi_{ri} + \phi_{le}) - 2r\phi_{to} = 0$$

Table A.2 summarizes these coefficients for boundary nodes with NBC. It is to be noted that for corner nodes with both connecting boundaries having NBC (e.g. node 0 in the present case) will be influenced by the NBC from both these boundaries. Using a similar approach the equations for such corners nodes can be derived. Table A.3 summarizes the equations for these corner nodes. Unlike the one sided differencing approach, the inconsistency in the value of the corner node is eliminated by solving it as a part of the equation system.

A.2 Validation of numerical analysis

This appendix provides details of validation of the python3 code which uses the concepts of parallel numerical algorithms for the elliptic PDE by comparing the analytical results with the numerical results. For the purpose of validation, it is necessary to have a function which is known to satisfy the elliptic PDE, eq.(4.13). The family of functions which satisfy homogeneous elliptic PDEs (Grewal, 2012) or Laplace equation could have any of the forms as given in eq.(A.1)

$$\phi = (c_1 e^{px} + c_2 e^{-px})(c_3 \cos py + c_4 \sin py) \tag{A.1a}$$

$$\phi = (c_5 \cos px + c_6 \sin px)(c_7 e^{py} + c_8 e^{-py}) \tag{A.1b}$$

$$\phi = (c_9 x + c_{10})(c_{11} y + c_{12}) \tag{A.1c}$$

For any problem governed by elliptic PDE, the constants in the above equations are determined by the boundary conditions. Based on the first equation i.e. eq.(A.1a), a test function with some arbitrary constants and axes notation as followed throughout this work is chosen and the same is given in eq.(A.2)

$$\phi = (0.5e^{0.009X_1} + 1.5e^{-0.009X_1})(2\cos(0.009X_3) + 7\sin(0.009X_3)) \tag{A.2}$$

The directional derivatives of the above function are given by

$$\phi_{X_1} = -(0.0135e^{-0.009X_1} - 0.0045e^{0.009X_1})(2\cos(0.009X_3) + 7\sin(0.009X_3)) \tag{A.3a}$$

$$\phi_{X_3} = (0.063\cos(0.009X_3) - 0.018\sin(0.009X_3))(1.5e^{-0.009X_1} + 0.5e^{0.009X_1}) \tag{A.3b}$$

A large domain of $2500m \times 1000m$, similar to the node spacing for the simulations undertaken in this work with two openings of arbitrary size $150m \times 20m$ placed at an arbitrary location is considered. The origin is fixed at the centre of the bottom boundary of the domain. The node to node distance considered is $1m$ in either direction. The domain with plot of its boundary conditions is shown in Figure A.2. It is to be noted that the coefficient of the power of the exponential function is chosen to be very small i.e. 0.009, so that the values of ϕ do not become exceedingly large. Also, the values would have ranged from extremely high orders of magnitude at the left and right boundaries to extremely low values at the centre. However, in the

Table A.4: Comparison of analytical and numerical results

		Analytical results			Numerical results			Relative error (%)		
X_1	X_3	ϕ	ϕ_{X_1}	ϕ_{X_3}	ϕ	ϕ_{X_1}	ϕ_{X_3}	ϕ	ϕ_{X_1}	ϕ_{X_3}
−1200	100	494608.2389	−4451.4741	1842.8054	494620.5638	−4451.3031	1842.8314	0.0025	0.0038	0.0014
−1198	100	485784.9386	−4372.0644	1809.9317	485797.717	−4371.8986	1809.9597	0.0026	0.0038	0.0015
−1196	100	477119.0368	−4294.0713	1777.6444	477132.2758	−4293.9067	1777.6728	0.0028	0.0038	0.0016
−1194	100	468607.7257	−4217.4695	1745.933	468621.4171	−4217.2952	1745.9615	0.0029	0.0041	0.0016
−1192	100	460248.2477	−4142.2342	1714.7874	460262.3818	−4142.0655	1714.8167	0.0031	0.0041	0.0017
−1127	132	275995.9093	−2483.9632	260.5148	276023.0047	−2483.8987	260.4483	0.0098	0.0026	0.0255
−1125	132	271072.4272	−2439.6518	255.8675	271099.7109	−2439.6031	255.7912	0.0101	0.002	0.0298
−1123	132	266236.775	−2396.131	251.3031	266264.2029	−2396.1104	251.2145	0.0103	0.0009	0.0353
−1121	132	261487.3857	−2353.3865	246.8201	261514.8757	−2353.3994	246.7172	0.0105	0.0005	0.0417

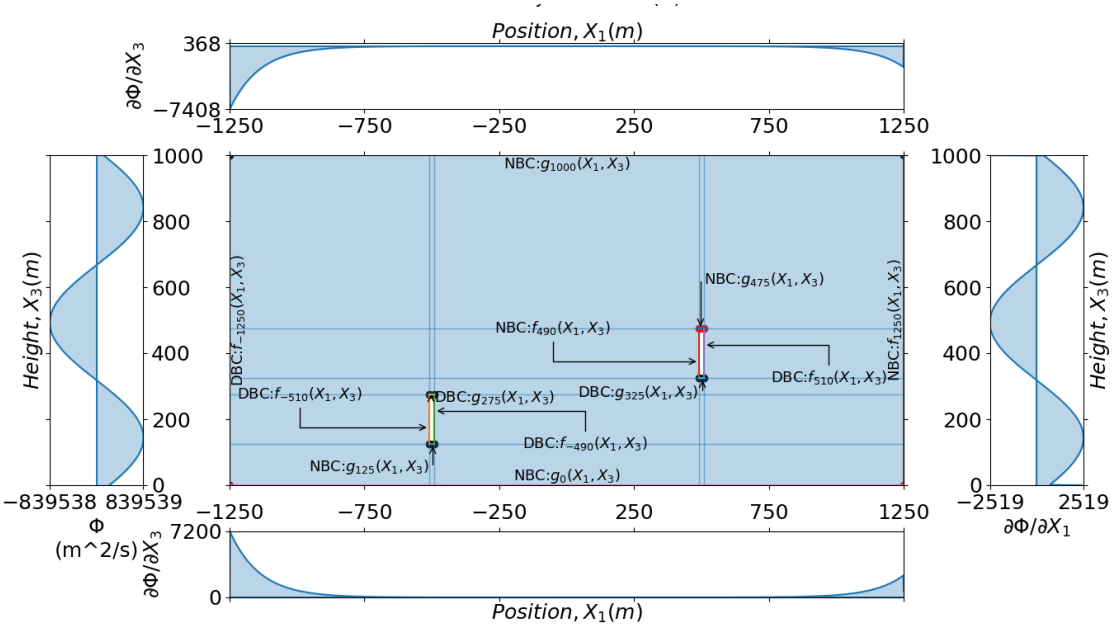


Figure A.2: Domain considered for validation of Python3 code

kind of simulations undertaken, the difference in the values ϕ within the domain are not so extreme and hence evaluating the performance of the solver in such a scenario is not relevant to the present context. The code is run parallelly on all 6 cores of an Intel core i9 processor.

Table A.4 shows a comparison of the numerical results with the analytical results. In this case considered for validation, where the boundary values are clearly known by a properly defined function, the results of the numerical results almost nearly match the analytical results. Thus, the code for numerical simulation using the concepts of parallel numerical algorithms can be seen to perform nicely.

The results have also been tested with MATLAB's internal PDE tool for a

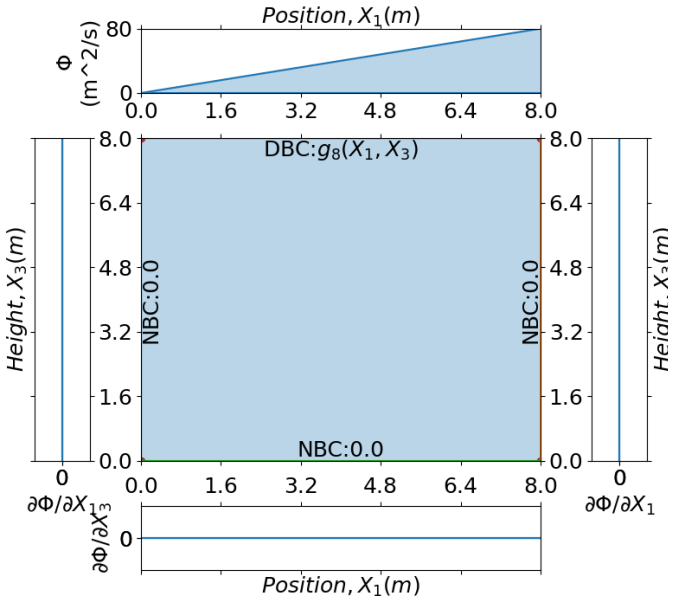
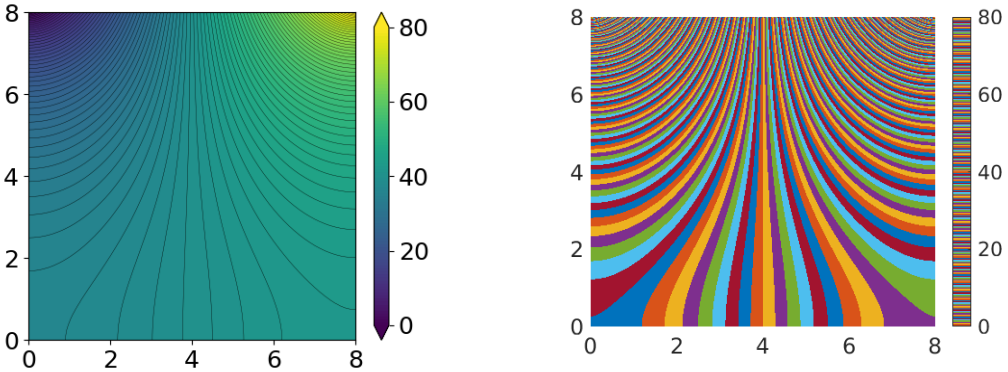


Figure A.3: Domain considered for comparison with MATLAB

small domain with boundary conditions as given in Figure A.3. It can be clearly seen from Figure A.4 that the results match nicely, thereby confirming the validity of the solver.



(a) using Python (b) Using MATLAB

Figure A.4: Comparison with MATLAB's PDE solver

Appendix B

Validation for the turbulence model

B.1 Validation of BiCGStab

In this appendix, few special cases pertaining to the application of BiCGStab in the context of the proposed turbulence model has been presented and the numerical results obtained by executing the python code have been validated with respect to the theoretical values. It is to be noted that though all the values are shown upto four places of decimal, calculations have been done using double precision or 64bits floating point numbers.

B.1.1 Pressure equation

From the equations of pressure presented earlier, it can be observed that any interior node depends on 9 neighbouring nodes in case of a 3D domain and 6 neighbouring nodes in case of 2D domain. Consider for example a linear system with coefficient matrix, A as shown in Table B.1 and RHS of the system, b as shown in Table B.2.

Table B.1: Coefficient matrix, A for pressure

0	0.9795	0	0	0	0	0	0	0	0.2449	0	0	0	0	0	0	0	0
-0.9795	-0.9795	0.9795	0	0	0	0	0	0	0	0.2449	0	0	0	0	0	0	0
0.9795	-0.9795	-0.9795	0.9795	0	0	0	0	0	0	0	0.2449	0	0	0	0	0	0
0	0.9795	-0.9795	-0.9795	0.9795	0	0	0	0	0	0	0	0.2449	0	0	0	0	0
0	0	0.9795	-0.9795	-0.9795	0.9795	0	0	0	0	0	0	0	0.2449	0	0	0	0
0	0	0	0.9795	-0.9795	-0.9795	0.9795	0	0	0	0	0	0	0	0.2449	0	0	0
0	0	0	0	0.9795	-0.9795	-0.9795	0.9795	0	0	0	0	0	0	0	0.2449	0	0
0	0	0	0	0	0.9795	-0.9795	-0.9795	0.9795	0	0	0	0	0	0	0	0.2449	0
0	0	0	0	0	0	0.9795	-0.9795	-0.9795	0	0	0	0	0	0	0	0	0.2449
-0.2343	0	0	0	0	0	0	0	0	-0.2449	0.9372	0	0	0	0	0	0	0
0	-0.2343	0	0	0	0	0	0	0	-0.9372	-1.1821	0.9372	0	0	0	0	0	0
0	0	-0.2343	0	0	0	0	0	0	0.9372	-0.9372	-1.1821	0.9372	0	0	0	0	0
0	0	0	-0.2343	0	0	0	0	0	0	0.9372	-0.9372	-1.1821	0.9372	0	0	0	0
0	0	0	0	-0.2343	0	0	0	0	0	0	0.9372	-0.9372	-1.1821	0.9372	0	0	0
0	0	0	0	0	-0.2343	0	0	0	0	0	0	0.9372	-0.9372	-1.1821	0.9372	0	0
0	0	0	0	0	0	-0.2343	0	0	0	0	0	0	0.9372	-0.9372	-1.1821	0.9372	0
0	0	0	0	0	0	0	-0.2343	0	0	0	0	0	0	0.9372	-0.9372	-1.1821	0.9372
0	0	0	0	0	0	0	0	-0.2343	0	0	0	0	0	0	0.9372	-0.9372	-1.1821

It can be observed the element a_{11} in the leading diagonal is 0. This case corresponds to that of the pressure equation presented earlier in Chapter 6. If the values of x obtained by using BiCGStab are compared with that obtained manually using matrix inversion function in a spreadsheet, the values are approximately the same. This proves the fact that the presence of zero in the leading diagonal does not have any major effect on the performance of BiCGStab.

Another scenario has been presented in Table B.2 where under-relaxation factor has been introduced in BiCGStab. Compared to the previous case, it can be observed that the values of x show considerable deviation from the expected values. As explained earlier, the reason for this is that introducing the under-relaxation factor modifies the coefficient matrix itself.

Table B.2: b and x computed using BiCGStab and spreadsheet for pressure

b	**x (BiCGStab in python)**	**x (using matrix inversion in spreadsheets)**	**x (BiCGStab with under-relaxation in python)**
0.00041	−0.11986	−0.11463	−0.07192
0.02745	−0.01398	−0.01113	−0.00839
0.01232	−0.10025	−0.09271	−0.06015
0.00058	0.00214	0.00866	0.00128
0.00086	−0.07672	−0.06846	−0.04603
−0.00277	0.01091	0.01849	0.00655
0.00960	−0.06333	−0.05795	−0.03800
−0.00904	0.01839	0.02293	0.01103
−0.00278	−0.05981	−0.0603	−0.03589
−0.00330	0.04860	0.04618	0.02916
0.03828	−0.01840	−0.02011	−0.01104
−0.00025	0.06310	0.05888	0.03786
−0.00038	−0.01070	−0.01548	−0.00642
0.00023	0.06730	0.06123	0.04038
−0.0002	−0.00764	−0.01400	−0.00458
−0.00008	0.06833	0.06345	0.04100
0.00013	−0.00563	−0.00977	−0.00338
−0.00117	0.07458	0.07100	0.04475

B.1.2 Velocity equations

In this section, a sample of the coefficient matrix of velocity equations and the results obtained using BiCGStab has been shown to validate BiCGStab. Table B.3 shows a typical coefficient matrix for the velocity equations.

Table B.3: Coefficient matrix, A for velocity

1	-3E-6	0	0	0	0	0	0	0	-8E-7	0	0	0	0	0	0	0	0
-1	1	-3E-6	0	0	0	0	0	0	0	-8E-7	0	0	0	0	0	0	0
0	-1	1	-3E-6	0	0	0	0	0	0	0	-8E-7	0	0	0	0	0	0
0	0	-1	1	-3E-6	0	0	0	0	0	0	0	-8E-7	0	0	0	0	0
0	0	0	-1	1	-3E-6	0	0	0	0	0	0	0	-8E-7	0	0	0	0
0	0	0	0	-1	1	-3E-6	0	0	0	0	0	0	0	-8E-7	0	0	0
0	0	0	0	0	-1	1	-3E-6	0	0	0	0	0	0	0	-8E-7	0	0
0	0	0	0	0	0	-1	1	-3E-6	0	0	0	0	0	0	0	-8E-7	0
0	0	0	0	0	0	0	-1	1	0	0	0	0	0	0	0	0	-8E-7
-8E-7	0	0	0	0	0	0	0	0	1	-3E-6	0	0	0	0	0	0	0
0	-8E-7	0	0	0	0	0	0	0	-1	1	-3E-6	0	0	0	0	0	0
0	0	-8E-7	0	0	0	0	0	0	0	-1	1	-3E-6	0	0	0	0	0
0	0	0	-8E-7	0	0	0	0	0	0	0	-1	1	-3E-6	0	0	0	0
0	0	0	0	-8E-7	0	0	0	0	0	0	0	-1	1	-3E-6	0	0	0
0	0	0	0	0	-8E-7	0	0	0	0	0	0	0	-1	1	-3E-6	0	0
0	0	0	0	0	0	-8E-7	0	0	0	0	0	0	0	-1	1	-3E-6	0
0	0	0	0	0	0	0	-8E-7	0	0	0	0	0	0	0	-1	1	-3E-6
0	0	0	0	0	0	0	0	-8E-7	0	0	0	0	0	0	0	-1	1

The extremely low values in the this matrix are the terms corresponding to the diffusion or viscous term in the equations.

As can be seen from Table B.4, the values of x computed using the iterative algorithm is nearly the same when compared to the results obtained using matrix inversion in spreadsheets. This case proves that BiCGStab performs properly for the velocity equations in SIMPLER. Though the effect of under-relaxation factor has not been presented, it is worth mentioning that the under-relaxation factor has the same effect in this case just like that with the case of the pressure equations.

Table B.4: b and x computed using BiCGStab and spreadsheet for velocity

b	**x (BiCGStab in python)**	**x (using matrix inversion in spreadsheets)**
−0.01326	−0.01325	−0.01326
−0.00561	−0.01892	−0.01887
−0.00433	−0.02331	−0.02320
−0.00656	−0.02911	−0.02976
−0.00194	−0.02893	−0.03170
−0.00333	−0.03042	−0.03503
−0.00132	−0.03162	−0.03635
−0.00017	−0.03178	−0.03652
0.00627	−0.02346	−0.03025
−0.01357	−0.01356	−0.01357
−0.00366	−0.01728	−0.01723
−0.00149	−0.01883	−0.01872
−0.00051	−0.01855	−0.01924
0.00000	−0.01647	−0.01924
0.00055	−0.01447	−0.01869
0.00043	−0.01481	−0.01826
−0.00081	−0.01655	−0.01907
−0.00089	−0.01586	−0.01996

Publications

Basu, S., Soodhalter, K., Fitzgerald, B., & Basu, B. (2022). Flow in a large wind field with multiple actuators in the presence of constant vorticity. *Phys. of Fluids*, *34*(10). https://doi.org/10.1063/5.0104902

Bibliography

Ainslie, J. (1988). Calculating the flowfield in the wake of wind turbines. *J. of Wind Eng. and Ind. Aerodyn.*, *27*(1), 213–224. https://doi.org/10.1016/0167-6105(88)90037-2

Ainslie, J., & Scott, J. (1990). Theoretical modelling of noise generated by wind turbines. *Wind Eng.*, *14(1)*, 9–14.

Arthurs, K., Moore, L., Peskin, C., Pitman, E., & Layton, H. (1998). Modeling arteriolar flow and mass transport using the immersed boundary method. *J. of Comput. Phys.*, *147*(2), 402–440. https://doi.org/10.1006/jcph.1998.6097

Baez-Vidal, A., Lehmkuhl, O., Valdivieso, D., & Pérez Segarra, C. (2013). Parallel large eddy simulations of wind farms with the actuator line method. *Procedia Eng.*, *61*, 227–232.

Barthelmie, R., Larsen, G., Frandsen, S., Folkerts, L., Rados, K., Pryor, S., Lange, B., & Schepers, G. (2006). Comparison of wake model simulations with offshore wind turbine wake profiles measured by sodar. *J. of Atmos. and Ocean.Technol.*, *23*(7), 888–901. https://doi.org/10.1175/JTECH1886.1

Bastankhah, M., & Porté-Agel, F. (2014). A new analytical model for wind-turbine wakes. *Renew. Energy*, *70*, 116–123. https://doi.org/10.1016/j.renene.2014.01.002

Basu, B. (2016). Irrotational two-dimensional free-surface steady water flows over a flat bed with underlying currents. *Nonlinear Anal.: Theory, Methods & Appl.*, *147*, 110–124.

Basu, S., Soodhalter, K., Fitzgerald, B., & Basu, B. (2022). Flow in a large wind field with multiple actuators in the presence of constant vorticity. *Phys. of Fluids*, *34*(10). https://doi.org/10.1063/5.0104902

Batchelor, G. (1982). *The theory of homogeneous turbulence*. Cambridge University Press.

Batchelor, G., & Proudman, I. (1954). The effect of rapid distortion of a fluid in turbulent motion. *The Q. J. of Mech. and Appl. Math.*, *7(1)*, 83–103.

Berthelsen, P. (2004). A decomposed immersed interface method for variable coefficient elliptic equations with non-smooth and discontinuous solutions. *J. of Comput. Phys.*, *197*(1), 364–386.

Beyer, R. (1992). A computational model of the cochlea using the immersed boundary method. *J. of Comput. Phys.*, *98*(1), 145–162. https://doi.org/10.1016/0021-9991(92)90180-7

Blondel, F., & Cathelain, M. (2020). An alternative form of the super-gaussian wind turbine wake model. *Wind Energy Sci.*, *5*(3), 1225–1236. https://doi.org/10.5194/wes-5-1225-2020

Bottino, D. (1998). Modeling viscoelastic networks and cell deformation in the context of the immersed boundary method. *J. of Comput. Phys.*, *147*(1), 86–113. https://doi.org/10.1006/jcph.1998.6074

Brereton, G., & Mankbadi, R. (1993). *A rapid-distortion-theory turbulence model for developed unsteady wall-bounded flow*. NASA.

Brower, M., & Robinson, N. (2012). *The openwind deep-array model—development and validation*. AWS Truepower Report.

Burton, T., Jenkins, N., Sharpe, D., & Bossanyi, E. (2011). *Wind energy handbook*. John Wiley; Sons.

Calkin, R., Hempel, R., Hoppe, H., & Wypior, P. (1994). Portable programming with the parmacs message-passing library. *Parallel Comput.*, *20*(4), 615–632.

Chand, K., Henshaw, W., Lundquist, K., & Singer, M. (2010). Cgwind: A high-order accurate simulation tool for wind turbines and wind farms.

Chougule, A., Mann, J., Kelly, M., & Larsen, G. (2017). Modeling atmospheric turbulence via rapid distortion theory: Spectral tensor of velocity and buoyancy. *J. of the Atmos. Sci.*, *74*(4), 949–974.

Churchfield, M. (2013). *A review of wind turbine wake models and future directions* [online], National Renewable Energy Laboratory. https://www.nrel.gov/docs/fy14osti/60208.pdf

Cortez, R., Fauci, L., & Medovikov, A. (2005). The method of regularized stokeslets in three dimensions: Analysis, validation, and application to helical swimming. *Phys. of Fluids*, *17*(3), 031504. https://doi.org/10.1063/1.1830486

Cortez, R., & Minion, M. (2000). The blob projection method for immersed boundary problems. *J. of Comput. Phys.*, *161*(2), 428–453. https://doi.org/10.1006/jcph.2000.6502

Crasto, G., Gravdahl, A., Castellani, F., & Piccioni, E. (2012). Wake modeling with the actuator disc concept. *Energy Procedia*, *24*, 385–392. https://doi.org/10.1016/j.egypro.2012.06.122

Deaves, D., & Harris, R. (1978). *A mathematical model of the structure of strong winds*. CIRIA.

Deissler, R. (1968). Effects of combined two-dimensional shear and normal strain on weak locally homogeneous turbulence and heat transfer. *J. of Math. and Phys.*, *47*(1-4), 310–326. https://doi.org/10.1002/sapm1968471310

Ehrich, S., Schwarz, C., Rahimi, H., Stoevesandt, B., & Peinke, J. (2018). Comparison of the blade element momentum theory with computational fluid dynamics for wind turbine simulations in turbulent inflow. *Appl. Sci.*, *8*(12). https://doi.org/10.3390/app8122513

Emeis, S., & Frandsen, S. (1993). Reduction of horizontal wind speed in a boundary layer with obstacles. *Bound.-Layer Meteorol.*, *64*(3), 297–305. https://doi.org/10.1007/BF00708968

Energy in ireland. (2018). Sustainable Energy Authority of Ireland.

Energy in ireland. (2020). Sustainable Energy Authority of Ireland.

Farr, T., & Hancock, P. (2014). Torque fluctuations caused by upstream mean flow and turbulence. *J. of Phys.: Conf. Ser.*, *555*, 012048. https://doi.org/10.1088/1742-6596/555/1/012048

Frandsen, S. (1992). On the wind speed reduction in the center of large clusters of wind turbines. *J. of Wind Eng. and Ind. Aerodyn.*, *39*(1), 251–265. https://doi.org/10.1016/0167-6105(92)90551-K

Frandsen, S., Barthelmie, R., Pryor, S., Rathmann, O., Larsen, S., Højstrup, J., & Thøgersen, M. (2006). Analytical modelling of wind speed deficit in large offshore wind farms. *Wind Energy*, *9*(1-2), 39–53. https://doi.org/10.1002/we.189

Froude, R. (1889). On the part played in propulsion by difference in pressure. *Trans. of the Inst. of Nav. Archit.*, *30*, 390–405.

Gabriel, E., Fagg, G., Bosilca, G., Angskun, T., Dongarra, J., Squyres, J., Sahay, V., Kambadur, P., Barrett, B., Lumsdaine, A., Castain, R., Daniel, D., Graham, R., & Woodall, T. (2004). Open MPI: Goals, concept, and design of a next generation MPI implementation. *Proceedings, 11th European PVM/MPI Users' Group Meeting*, 97–104.

Garratt, J. (1992). *The atmospheric boundary layer*. Cambridge University Press.

Gebraad, P. M. O., Teeuwisse, F. W., van Wingerden, J. W., Fleming, P. A., Ruben, S. D., Marden, J. R., & Pao, L. Y. (2016). Wind plant power optimization through yaw control using a parametric model for wake effects—a cfd simulation study. *Wind Energy*, *19*(1), 95–114. https://doi.org/10.1002/we.1822

Gillespie, A. (2016). *These creative wind turbines will have you rethinking what you know about wind power*. Smithsonian.

Glauert, H. (1926). The analysis of experimental results in the windmill brake and vortex ring states of an airscrew.

Graham, J. (2017). Rapid distortion of turbulence into an open turbine rotor. *J. of Fluid Mech.*, *825*, 764–794.

Grant, A., & Dickens, R. (1993). An implementation of a portable instrumented communication library using cs tools. *Future Generation Comput. Syst.*, *9*(1), 53–61.

Greengard, L., & Rokhlin, V. (1987). A fast algorithm for particle simulations. *J. of Comput. Phys.*, *73*(2), 325–348.

Grewal, B. (2012). *Higher engineering mathematics*. Khanna Publishers.

Gropp, W., Lusk, E., & Skjellum, A. (1999). *Using mpi: Portable parallel programming with the message-passing interface*. Massachussettes Institute of Technology.

Hanifi, A. (2010). *Computational fluid dynamics* [University Lecture], KTH Royal Institute of Technology, Stockholm.

Harlow, F., & Welch, J. (1965). Numerical calculation of time-dependent viscous incompressible flow of fluid with free surface. *The Phys. of Fluids*, *8*, 2182–2189.

Hestenes, M., & Stiefel, E. (1952). Methods of conjugate gradients for solving linear systems. *J. of Res. of the Natl. Bureau of Stand.*, *49*, 409–436 (1953). https://doi.org/10.6028/JRES.049.044

Hoffman, J. (2001). *Numerical methods for engineers and scientists*. Marcel Dekker Inc.

Høistrup, J. (1983). Nibe wake, part one. *Internal report*.

Hunt, J., & Carruthers, D. (1990). Rapid distortion theory and the 'problems' of turbulence. *J. of Fluid Mech.*, *212*, 497–532.

Hussein, A., & El-Shishiny, H. (2012). Modeling and simulation of micro-scale wind farms using high performance computing. *Int. J. of Comput. Methods*, *09*(02), 1240025.

Hwu, W., Mattson, T., & Keutzer, K. (2008). *The concurrency challenge*. IEEE Design and Test of Computers.

Issa, R. (1986). Solution of the implicitly discretised fluid flow equations by operator-splitting. *J. of Comput. Phys.*, *62*(1), 40–65. https://doi.org/10.1016/0021-9991(86)90099-9

Issa, R., Gosman, A., & Watkins, A. (1986). The computation of compressible and incompressible recirculating flows by a non-iterative implicit scheme. *J. of Comput. Phys.*, *62*(1), 66–82. https://doi.org/10.1016/0021-9991(86)90100-2

Isserlis, L. (1918). On a formula for the product-moment coefficient of any order of a normal frequency distribution in any number of variables. *Biometrika*, *12*(1/2), 134–139.

Ivanell, S., Mikkelsen, R., Sørensen, J., & Henningson, D. (2008). Three dimensional actuator disc modelling of wind farm wake interaction, 1–10.

Jang, D., Jetli, R., & Acharya, S. (1986). Comparison of the piso, simpler, and simplec algorithms for the treatment of the pressure-velocity coupling in steady flow poblems. *Numer. Heat Transf.*, *10*(3), 209–228. https://doi.org/10.1080/10407788608913517

Jeandel, D., Brison, J., & Mathieu, J. (1978). Modeling methods in physical and spectral space. *The Phys. of Fluids*, *21*(2), 169–182. https://doi.org/10.1063/1.862211

Jensen, N. (1983). A note on wind generator interaction. *Risø-M No. 2411*, 1–18.

Johnson, C., Graves, A., Tindal, A., Cox, S., Schlez, W., & Neubert, A. (2009). New developments in wake models for large wind farms.

Katic, I., Højstrup, J., & Jensen, N. (1986). A simple model for cluster efficiency. *Proceedings of the European wind energy association conference and exhibition*, 407–409.

King, J., Fleming, P., King, R., Martinez-Tossas, L., Bay, C., Mudafort, R., & Simley, E. (2021). Control-oriented model for secondary effects of wake steering. *Wind Energy Sci.*, *6*(3), 701–714. https://doi.org/10.5194/wes-6-701-2021

Kirby, A., Brazell, M., Yang, Z., Roy, R., Ahrabi, B., Stoellinger, M., Sitaraman, J., & Mavriplis, D. (2019). Wind farm simulations using an overset hp-adaptive approach with blade-resolved turbine models. *The Int. J. of High Performa. Comput. Appl*, *33*(5), 897–923. https://doi.org/10.1177/1094342019832960

Kirk, D., & Hwu, W. (2013). *Programming massively parallel processors: A hands-on approach*. Elsevier.

Korobenko, A., Yan, J., Gohari, S., Sarkar, S., & Bazilevs, Y. (2017). Fsi simulation of two back-to-back wind turbines in atmospheric boundary layer flow. *Comput. & Fluids*, *158*, 167–175.

Kumar, M., & Joshi, P. (2012). Some numerical techniques for solving elliptic interface problems. *Numer. Methods for Partial Differ. Equ.*, *28*(1), 94–114.

Larsen, G., Madsen Aagaard, H., Bingol, F., Mann, J., Ott, S., Sørensen, J., Okulov, V., Troldborg, N., Nielsen, N., & Thomsen, K. (2007). *Dynamic wake meandering modeling*. Risø National Laboratory.

Larsen, T., Madsen, H., Larsen, G., & Hansen, K. (2013). Validation of the dynamic wake meander model for loads and power production in the egmond aan zee wind farm. *Wind Energy*, *16*(4), 605–624. https://doi.org/10.1002/we.1563

Lavely, A., Vijayakumar, G., Craven, B., Jayaraman, B., Paterson, E., Nandi, T., & Brasseur, J. (2014). Towards a blade-resolved hybrid urans-les of the nrel 5-mw wind turbine rotor within large eddy simulation of the atmospheric boundary layer. *32nd ASME Wind Energy Symp.*, 1–16. https://doi.org/10.2514/6.2014-0869

Lawton, S., & Crawford, C. (2012). Development and validation of libaero , a potential flow aerodynamics library for horizontal-axis wind turbines. *50th AIAA Aerospace Sci. Meeting including the New Horizons Forum and Aerospace Exposition*.

Lee, M., Kim, J., & Moin, P. (1990). Structure of turbulence at high shear rate. *J. of Fluid Mech.*, *216*, 561–583. https://doi.org/10.1017/S0022112090000532

Li, Z. (1994). *The immersed interface method-a numerical approach for partial differential equations with interfaces* [PhD Thesis], Universiy of Washington. https://faculty.washington.edu/rjl/students/li/liphd.pdf

Li, Z. (2003). An overview of the immersed interface method and its applications. *Taiwan. J. of Math.*, *7*(1), 1–49.

Li, Z., & Ito, K. (2006). *The immersed interface method: Numerical solutions of pdes involving intrefaces and irregular domains*. Society for Industrial; Applied Mathematics.

Li, Z., & Lai, M. (2001). The immersed interface method for the navier–stokes equations with singular forces. *J. of Comput. Phys.*, *171*(2), 822–842.

Lindsey, R., & Dahlman, L. (2021). *Climate change: Global temperature*. NOAA.

Lock, C., Bateman, H., & Townsend, H. (1925). An extension of the vortex theory of airscrews with applications to airscrews of small pitch, including experimental results. *A.R.C. Research Reports and Memorandam*, *1014*.

Madsen, H. (1996). *A cfd analysis of the actuator disc flow compared with momentum theory results*. International Energy Agency.

Mann, J., Peña, A., Troldborg, N., & Andersen, S. J. (2018). How does turbulence change approaching a rotor? *Wind Energy Sci.*, *3*(1), 293–300. https://doi.org/10.5194/wes-3-293-2018

Maronga, B., Gryschka1, M., Heinze, R., Hoffmann, F., Kanani-Sühring, F., Keck, M., Ketelsen, K., Letzel, M. O., Sühring, M., & Raasch, S. (2015). The parallelized large-eddy simulation model (palm) version 4.0 for atmospheric and oceanic flows: Model formulation, recent developments, and future perspectives. *Geosci. Model Dev.*, *8*, 2515–2551. https://doi.org/10.5194/gmd-8-2515-2015,2015

Martinez-Tossas, L., Annoni, J., Fleming, P., & Churchfield, M. (2019). The aerodynamics of the curled wake: A simplified model in view of flow control. *Wind Energy Sci.*, *4*(1), 127–138. https://doi.org/10.5194/wes-4-127-2019

McQueen, D., & Peskin, C. (1997). Shared-memory parallel vector implementation of the immersed boundary method for the computation of blood flow in the beating mammalian heart. *The J. of Supercomput.*, *11*(3), 213–236. https://doi.org/10.1023/A:1007951707260

Mines, C. (2018). *World's tallest offshore wind turbine will tower over iconic buildings*. NBC news.

Mittal, A., Sreenivas, K., Taylor, L., Hereth, L., & Hilbert, C. (2016). Blade-resolved simulations of a model wind turbine: Effect of temporal convergence. *J. of Wind Energy*, *19*(10), 1761–1783. https://doi.org/10.1002/we.1949

Mittal, R., & Iaccarino, G. (2005). Immersed boundary methods. *Annual Rev. of Fluid Mech.*, *37*(1), 239–261. https://doi.org/10.1146/annurev.fluid.37.061903.175743

Monin, A., & Yaglom, A. (1971). *Statistical fluid mechanics: Volume 1*. Dover Publications.

Nielsen, S. (2017). *Aerodynamics of wind turbines* [University Lecture], Tongji University.

Notay, Y. (2000). Flexible conjugate gradients. *SIAM J. Sci. Comput.*, *22*(4), 1444–1460 (electronic). https://doi.org/10.1137/S1064827599362314

NREL. (2021). Floris. version 2.4. https://github.com/NREL/floris

NSEnergy. (2021). *Profiling the top five countries with the highest wind energy capacity*. NSEnergy.

Nygaard, N., Steen, S., Poulsen, L., & Pedersen, J. (2020). Modelling cluster wakes and wind farm blockage. *J. of Phys.: Conf. Ser.*, *1618*(6), 62–72. https://doi.org/10.1088/1742-6596/1618/6/062072

Ott, S., & Nielsen, M. (2014). *Developments of the offshore wind turbine wake model fuga*. DTU Wind Energy.

Palmiter, S., & Katz, J. (2010). Evaluation of a potential flow model for propeller and wind turbine design. *J. of Aircr.*, *47*(5), 1739–1746.

Patankar, S. (1980). *Numerical heat transfer and fluid flow*. Hemisphere Publishing Corporation.

Patankar, S., & Spalding, D. (1972). A calculation procedure for heat, mass and momentum transfer in three-dimensional parabolic flows. *Int. J. of Heat and Mass Transf.*, *15*(10), 1787–1806. https://doi.org/10.1016/0017-9310(72)90054-3

Pearson, J. (1959). The effect of uniform distortion on weak homogeneous turbulence. *J. of Fluid Mech.*, *5*(2), 274–288. https://doi.org/10.1017/S0022112059000192

Peña, A., Gryning, S., & Hasager, C. (2008). Measurements and modelling of the wind speed profile in the marine atmospheric boundary layer. *Bound.-Layer Meteorol.*, *129*, 479–495.

Peskin, C. (1977). Numerical analysis of blood flow in the heart. *J. of Comput. Phys.*, *25*(3), 220–252.

Peskin, C. (1981). Lectures on mathematical aspects of physiology: (i) control of the heart and circulation; (ii) the inner ear; (iii) flow patterns around heart valves. In *Mathematical aspects of physiology* (pp. 1–107). AMS.

Rathman, O., Barthelmie, R., & Frandsen, S. (2006). Turbine wake model for wind resources software, 1–12.

Resor, B. (2013). *Definition of a 5mw/61.5m wind turbine blade reference model*. Sandia National Laboratories.

Reynolds, O. (1895). Iv. on the dynamical theory of incompressible viscous fluids and the determination of the criterion. *Philos. Trans. of the R. Soc. of Lond. (A.)*, *186*, 123–164. https://doi.org/10.1098/rsta.1895.0004

Richards, P., & Norris, S. (2019). Appropriate boundary conditions for computational wind engineering: Still an issue after 25 years. *J. of Wind Eng. and Ind. Aerodyn.*, *190*, 245–255.

Saad, Y. (2003). *Iterative methods for sparse linear systems*. Society for Industrial; Applied Mathematics.

Sayed, M., Lutz, T., Krämer, E., Shayegan, S., & Wüchner, R. (2019). Aeroelastic analysis of 10 mw wind turbine using cfd–csd explicit fsi-coupling approach. *J. of Fluids and Structures*, *87*, 354–377. https://doi.org/10.1016/j.jfluidstructs.2019.03.023

Schlez, W., & Neubert, A. (2009). New developments in large wind farm modelling, 1–8.

Shane, C. (2011). *Potential flow modelling for wind turbines*. Masters Thesis, University of Victoria.

Sørensen, J., & Shen, W. (2002). Numerical modeling of wind turbine wakes. *J. of Fluids Eng.*, *124*(2), 393–399. https://doi.org/10.1115/1.1471361

Sørensen, N. (2015). *Ellipsys2d/3d*. http://www.the-numerical-wind-tunnel.dtu.dk/ellipsys

Stromsta, K. (2010). *Denmark's horns rev 1 wind farm rendered inoperative by fault*. Recharge.

Stull, R. (2016). *Practical meteorology: An algebra-based survey of atmospheric science*. AVP International.

Sun, H., & Yang, H. (2018). Study on an innovative three-dimensional wind turbine wake model. *Appl. Energy*, *226*, 483–493. https://doi.org/10.1016/j.apenergy.2018.06.027

Sun, H., & Yang, H. (2020). Numerical investigation of the average wind speed of a single wind turbine and development of a novel three-dimensional multiple wind turbine wake model. *Renewable Energy*, *147*, 192–203. https://doi.org/10.1016/j.renene.2019.08.122

Sunderam, V. (1990). Pvm: A framework for parallel distributed computing. *Concurr.: Pract. and Exp.*, *2*(4), 315–339.

Sutter, H., & Larus, J. (2005). Software and the concurrency revolution: Leveraging the full power of multicore processors demands new tools and new thinking from the software industry. *Queue*, *3*(7), 54–62. https://doi.org/10.1145/1095408.1095421

Tan, Z., Lim, K., & Khoo, B. (2009a). A fast immersed interface method for solving Stokes flows on irregular domains. *Comput. & Fluids*, *38*(10), 1973–1983. https://doi.org/10.1016/j.compfluid.2009.06.004

Tan, Z., Lim, K., & Khoo, B. (2009b). An immersed interface method for Stokes flows with fixed/moving interfaces and rigid boundaries. *J. of Comput. Phys.*, *228*(18), 6855–6881. https://doi.org/10.1016/j.jcp.2009.06.005

Thøgersen, M. (2005). *Wind pro/park: Introduction to wind turbine wake modelling and wake generated turbulence*. EMD International A/S, Niels Jernes Vej 10, DK-9220 Aalborg, Denmark.

Tian, L., Zhu, W., Shen, W., Zhao, N., & Shen, Z. (2015). Development and validation of a new two-dimensional wake model for wind turbine wakes. *J. of Wind Eng. and Ind. Aerodyn.*, *137*, 90–99. https://doi.org/10.1016/j.jweia.2014.12.001

Townsend, A. (1980). *The structure of turbulent shear flow*. Cambridge University Press.

van der Vorst, H. (1992). Bi-CGSTAB: A fast and smoothly converging variant of Bi-CG for the solution of nonsymmetric linear systems. *SIAM J. on Sci. and Stat. Comput.*, *13*(2), 631–644. https://doi.org/10.1137/0913035

Van Doormaal, J., & Raithby, G. (1984). Enhancements of the simple method for predicting incompressible fluid flows. *Numer. Heat Transf.*, *7*(2), 147–163. https://doi.org/10.1080/01495728408961817

van Kuik, G. A. M., & Lignarolo, L. E. M. (2016). Potential flow solutions for energy extracting actuator disc flows. *Wind Energy*, *19*(8), 1391–1406.

van Kuik, G. A. M., Yu, W., Sarmast, S., & Ivanell, S. (2015). Comparison of actuator disc and joukowsky rotor flows, to explore the need for a tip correction. *J. of Phys.: Conf. Series*, *625*, 012013.

Versteeg, H., & Malalasekera, W. (2007). *An introduction to computational fluid dynamics*. Pearson Education Limited.

Vogel, J. (2007). Flexible BiCG and flexible Bi-CGSTAB for nonsymmetric linear systems. *Appl. Math. and Comput.*, *188*(1), 226–233. https://doi.org/10.1016/j.amc.2006.09.116

Voutsinas, M., S.G.and Belessis, & Rados, K. (1995). Investigation of the yawed operation of wind turbines by means of a vortex particle method. http://www.fluid.mech.ntua.gr/wind/belagard/agard.html

Walker, D. (1992). *Standards for message passing in a distributed memory environment*. Oak Ridge National Laboratory.

Walker, D., & Dongarra, J. (1995). *Mpi: A standard message passing interface*. Oak Ridge National Laboratory.

Waters, S. (2014). *Fluids and waves* [University Lecture], University of Oxford.

Weaver, J. (2017). *World's tallest wind turbine built in germany*. electrek.

Wiegmann, A., & Bube, K. (2000). The explicit-jump immersed interface method: Finite difference methods for pdes with piecewise smooth solutions. *SIAM J. on Numer. Anal.*, *37*(3), 827–862.

Wilson, J., & Flesch, T. (2004). An idealized mean wind profile for the atmospheric boundary layer. *Bound.-Layer Meteorol.*, *110*, 281–299.

Xu, G., & Sankar, L. (2000a). Computational study of horizontal axis wind turbines. *37th Aerospace Sci. Meeting and Exhibit*, 35–39.

Xu, G., & Sankar, L. (2000b). Effects of transition, turbulence and yaw on the performance of horizontal axis wind turbines. *2000 ASME Wind Energy Symposium*, 259–265.

Xu, S. (2008). The immersed interface method for simulating prescribed motion of rigid objects in an incompressible viscous flow. *J. of Comput. Phys.*, *227*(10), 5045–5071. https://doi.org/10.1016/j.jcp.2008.01.053