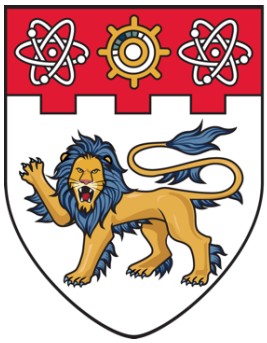


NANYANG TECHNOLOGICAL UNIVERSITY
SINGAPORE

ON NUMERICAL MODELLING OF ATMOSPHERIC GAS DISPERSION USING CFD APPROACH

TRAN LE VU

SCHOOL OF MECHANICAL AND AEROSPACE ENGINEERING

2019

ON NUMERICAL MODELLING OF ATMOSPHERIC GAS DISPERSION USING CFD APPROACH

TRAN LE VU

School of Mechanical and Aerospace Engineering

A thesis submitted to the Nanyang Technological University

in partial fulfilment of the requirement for the degree of

Doctor of Philosophy

2019

Statement of Originality

I hereby certify that the work embodied in this thesis is the result of original research, is free of plagiarised materials, and has not been submitted for a higher degree to any other University or Institution.

19/09/2019

Date

Tran Le Vu

Supervisor Declaration Statement

I have reviewed the content and presentation style of this thesis and declare it is free of plagiarism and of sufficient grammatical clarity to be examined. To the best of my knowledge, the research and writing are those of the candidate except as acknowledged in the Author Attribution Statement. I confirm that the investigations were conducted in accord with the ethics policies and integrity standards of Nanyang Technological University and that the research data are presented honestly and without prejudice.

19/9/2019

Date

Assoc Prof Ng Yin Kwee

Authorship Attribution Statement

This thesis contains material from paper published in peer-reviewed journal in which I am listed as the first author. Sections 4.1, 4.2 of Chapter 4 and Sections 5.2, 5.4 of Chapter 5 are published as:

Tran, V., Ng, E. Y. K., & Skote, M. (2019). CFD simulation of dense gas dispersion in neutral atmospheric boundary layer with OpenFOAM. Meteorology and Atmospheric Physics. https://doi.org/10.1007/s00703-019-00689-2

The contributions of the co-authors are as followed:

1. A/Prof Ng provided advise on the presentation and contents of the paper. He also revised it critically for important intellectual content.

2. A/Prof Skote provided editorial advise on the contents of the paper and highlighted its contributions to existing literature.

19/09/2019

Date

Tran Le Vu

Acknowledgements

Firstly, I would like to acknowledge my supervisor Assoc Prof Ng Yin Kwee, Eddie for his guidance, encouragements and supports. I also want to express my sincere gratitude to my co-supervisor Assoc Prof Dimitris Konovessis and Assoc Prof Martin Skotes for their insightful advices and supports during my PhD study.

I also would like to express my gratitude to Nanyang Technological University for providing me the NTU research scholarship. Without this opportunity and financial support, I can hardly pursue the journey of PhD. Thank you all my friends here in NTU for their joyfulness and supports you bring to me.

Last but not least, I acknowledge the people who mean a lot to me, my family and My Thuan. Your unconditional love and encouragement really touch my heart and motivate me in all my endeavours.

Abstract

Liquefied Natural Gas (LNG) is the fastest-growing gas supply source due to its economic and environmental benefit. However, LNG storage, handling and transportation are exposed to serious risks to the human, equipment and the environment, due to thermal hazards associated with combustion events such as pool fire, vapour cloud fire, explosion or rapid phase transition. Safety assessment and hazards mitigation method should be applied to lower the possibilities of catastrophic disaster relating to the LNG industry. This study is aimed at developing a CFD model of atmospheric gas dispersion and validating its usage for LNG vapour dispersion.

Ensuring accurate description of the Atmospheric Boundary Layer (ABL) is an important task for atmospheric gas dispersion simulation. Either the Reynolds Averaged Navier–Stokes (RANS) equations or Large Eddy Simulations (LES) are used for atmospheric turbulence modelling. The RANS turbulence models are adopted in this study since they are still widely used in practical approach to overcome boundary conditions sensitivity and computationally intensive of the LES. Modelling ABL surface layer as horizontally homogeneous turbulent surface layer (HHTSL) is used to develop a solver for ABL simulation using OpenFOAM CFD library. Monin-Obukhov similarity theory, which is well validated for flows in ABL surface layer over homogeneous surface, is used to model the profiles of velocity, turbulent kinetic energy and turbulence dissipation rate at the inlet boundary. Consistency of flow profiles in HHTSL across the computational domain is ensured by deriving a relation between turbulence model constants and implementing of appropriate wall functions.

A dispersion model is developed and validated using experimental data from wind tunnel tests of dense gas dispersion and field experiments of LNG vapour dispersion. The developed model takes into account buoyancy, the heat transfer from ground to the vapour cloud, the effect of variable temperature on gas properties and variable turbulent Schmidt number. The model is also implemented using OpenFOAM CFD library. Statistical Performance Measures (SPM) proposed for LNG dispersion model are used to compare results from developed model and commercial specialized gas dispersion code FLACS (FLame ACceleration Simulator). Results have shown the developed model fulfils requirements of all SPMs and outperforms FLACS in all factors.

The developed CFD model enables integration of ABL simulation with gas dispersion simulation. Validation results using benchmark data of dense gas dispersion and LNG vapour dispersion promoted the usage of general CFD in solving industrial safety problem, specifically in risk analysis of atmospheric dispersion of dense gas and LNG spill.

Table of contents

List of figures

List of tables

Nomenclature

Acronyms

Symbol	Description
ABL	Atmospheric Boundary Layer
FAC2	Factor of 2
FDS	Fire Dynamics Simulator
FLACS	FLame ACceleration Simulator
HHTSL	Horizontally Homogeneous Turbulent Surface Layer
LES	Large Eddy Simulations
LFL	Lower Flammability Limit
LLNL	Lawrence Livermore National Laboratory
LNG	Liquefied Natural Gas
MBSE	Mean Relative Square Error
MEP	Model Evaluation Protocol
MG	Geometric Mean Bias

MRB	Mean Relative Bias
NFPA	National Fire Protection Agency
RANS	Reynolds-Averaged Navier-Stokes
SPM	Statistical Performance Measures
SST $k-\omega$	Menter's Shear Stress Transport $k-\omega$
VG	Geometric Variance

Dimensionless Numbers/Quantities

Symbol	Description	Definition
Pr	Prandlt number	ν/α
Sc	Schmidt number	ν/D

Greek Symbols

Symbol	Description	Units
α	Thermal diffusivity	m^2/s
α_P	Numerical under-relaxation factor	
ε	Turbulence dissipation rate	$\mathrm{m}^2/\mathrm{s}^3$
ε^+	Near wall scale of turbulence dissipation rate	
κ	von Karman constant	
μ	Dynamic viscosity	kg/ms
μ_t	Turbulent dynamic viscosity	kg/ms

ν	Kinematic viscosity	m^2/s
ν^+	Near wall scale of kinematic viscosity	
ν_t	Turbulent kinematic viscosity	m^2/s
ω	Turbulence specific dissipation rate	$1/\mathrm{s}$
ω^+	Near wall scale of turbulence specific dissipation rate	
ϕ_h	Monin-Obukhov universal temperature similarity function	
ϕ_m	Monin-Obukhov universal momentum similarity function	
ρ	Fluid density	kg/m^3
τ, τ_{ij}	Viscous stress tensor	N/m^2
τ_s	Surface shear stress	N/m^2
θ	Potential temperature	K
θ_*	Friction temperature	K
$\boldsymbol{\delta}$, δ_{ij}	Kronecker delta	

Roman Symbols

Symbol	Description	Units
$\boldsymbol{g}$	Gravitational acceleration vector	m/s
$\boldsymbol{u}$	Velocity vector	m/s
A	matrix of coefficients	
C_m	Experimental measured concentration	

C_p	Predicted concentration from simulation	
c_p	Specific heat	$\mathrm{J/kgK}$
D	Mass diffusivity	$\mathrm{m^2/s}$
E	Smooth wall constant	
h	Enthalpy per unit mass	$\mathrm{J/kg}$
h_{ABL}	Height of ABL	m
k	Turbulence kinetic energy	$\mathrm{m^2/s^2}$
k^+	Near wall scale of turbulence kinetic energy	
L_{MO}	Monin-Obukhov length	m
M	Specie molecular weight	$\mathrm{kmol/kg}$
p	Fluid pressure	$\mathrm{N/m^2}$
p'	Numerical pressure correction	$\mathrm{N/m^2}$
p_{rgh}	Pressure defined without hydrostatic pressure	$\mathrm{N/m^2}$
q_s	Surface heat flux	$\mathrm{W/m^2}$
S_ε	Source term in turbulent dissipation rate equation	
S_k	Source term in turbulent kinetic energy equation	
T_s	Surface temperature	K
u'	Numerical velocity correction	$\mathrm{m/s}$
u^+	Near wall region velocity scale	

u_*	Friction velocity	m/s
w_*	Convective velocity	m/s
Y	Specie mass fraction	
y^+	Near wall region length scale	
y_P	Height of wall adjacent cell	m
z_0	ABL roughness length	m

Subscripts

Symbol	**Description**
α	Species index
eff	Sum of turbulence and laminar part of properties
t	Turbulence part of properties
P	Properties at cell point adjacent to wall
w, s	Properties value at wall/surface

Chapter 1

Introduction

1.1 Motivation

Mostly all human activities are affected by Atmospheric Boundary Layer (ABL). This is also where most air pollution phenomena occur. Understanding of the processes taking place in the ABL has attracted various research study. Some typical applications of ABL related research topics are wind engineering, urban flows, weather forecast, air pollution and risk assessment of hazardous material spills in industrial sites.

One of hazardous materials is Liquefied Natural Gas (LNG). LNG is an effective solution for long-distance natural gas transfer. LNG has become a prefer option for international trading of natural gas. Singapore's first LNG terminal with throughput capacity of 6 million tons a year (MTPA) was opened on 2014. It shows the move of Singapore government to this new emerging LNG market. However, LNG storage, handling, transportation are exposed to serious risks for human, equipments and the environment, due to thermal hazards associated with combustion events such as pool fire, vapour cloud fire, explosion or rapid phase transition. Safety assessment and hazards mitigation method should be applied to lower the possibilities of catastrophic disaster relating to the LNG industry.

Computational Fluid Dynamics (CFD) is increasingly being used in simulation of ABL flows. Open source CFD tool OpenFOAM is a more powerful research tool in comparison to proprietary software because of its flexibility to incorporate new implementation of fields calculation and also for post-processing. Using general CFD code like OpenFOAM in simulating ABL flows also encourage research sharing and reusing code in this specific field where in-house code is usually adopted.

Applying OpenFOAM for ABL gas dispersion is the motivation of this thesis. Successfulness of this will promote the use of general CFD in solving industrial safety problem.

1.2 Atmospheric boundary layer (ABL)

Atmospheric boundary layer (ABL) or planetary boundary layer (PBL) is the lowest part of atmosphere where the surface effects are dominant factors to characterise its properties. Most air pollution phenomena occur within ABL. ABL can be divided into three layers characterised by different scaling factors: roughness layer (from the ground to surface roughness length z_0), surface layer and mixed layer [1]. ABL is usually divided into different types based on the main mechanism of turbulence generation and atmospheric stability [2]. Atmospheric stability characterises the vertical acceleration of the air parcel. Pasquill-Gifford is the most common classification of atmospheric stability. According to this scheme, the atmospheric stability is classified into six classes, from A corresponding to the most unstable to D which is the neutral condition and to F which is the most stable conditions [3], depending on temperature, sensitive heat flux, surface roughness, wind velocity, and wind direction.

Figure 1.1 illustrates physical mechanisms involved in dispersion process of LNG vapour including wind convection, heat transfer from surrounding air, ground and solar radiation, buoyancy, as well as turbulence. Wind convection affects the dispersion process by the effect of wind speed and surface roughness. High wind speed advects the gas cloud more rapidly and produces atmospheric turbulence to increase mixing of the cloud. Surface roughness

determines the relation of advection and dilution process. Heat transfer to the cloud controls the total amount of heat added to the gas cloud; therefore, increases the cloud temperature and lowers the vapour cloud density in the course of dispersion. Together with buoyancy effect, the gas cloud changes from dense gas to tracer gas and eventually become buoyant gas. Turbulence state of ABL or generated by obstacles and terrain are also importation mechanisms affecting the dispersion process. Under stable atmospheric condition, mixing is suppressed due to the damping process of stratified density on vertical movement of air flow. Conversely, unstable atmospheric condition enhances the vertical mixing process. When the dispersion occurs at sloping terrain or in presence of obstructions, these also enhances gravity-driven flow and turbulent mixing.

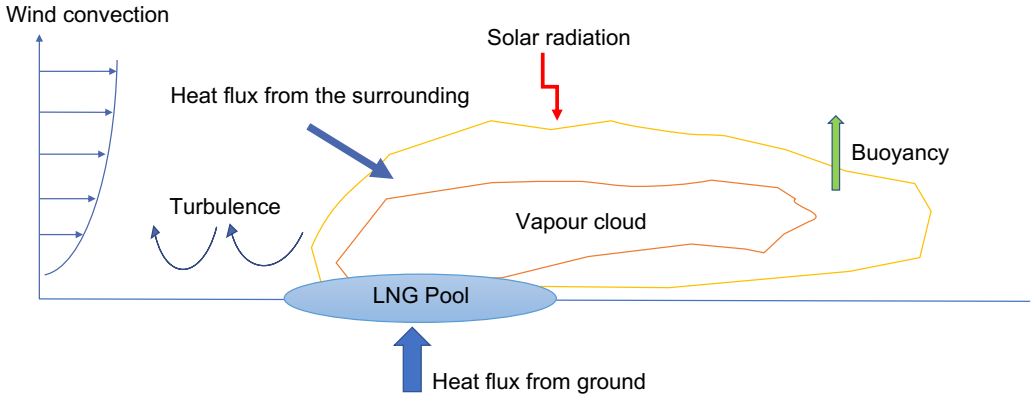


Fig. 1.1 : Illustration of physical mechanisms involving in LNG vapour dispersion

1.2.1 Monin–Obukhov similarity theory

The Monin–Obukhov similarity theory [4] has been widely applied to the surface layer of ABL. It assumes horizontally homogeneous and quasi-stationary flow field. ABL profiles of flow fields are only varied in vertical direction and vertical fluxes are constant.

Some most important scaling parameters in the surface layer are derived from surface shear stress τ_s, surface heat flux q_s and buoyancy variable g/T_s (T_s is the surface temperature). The resulting scaling parameters (Equation (1.1)) are friction velocity u_*, friction temperature

θ_* and the Monin-Obukhov length L_{MO} which is the height where shear effect is still significant in turbulence production:

$$\begin{aligned} u_* &= \sqrt{\frac{\tau_s}{\rho}} \\ \theta_* &= -\frac{q_s}{\rho c_p u_*} \\ L_{MO} &= \frac{T_s u_*^2}{\kappa g \theta_*} \end{aligned} \tag{1.1}$$

where κ is the von Karman constant, ρ, c_p and g are density, specific heat at constant pressure and acceleration of gravity accordingly.

From these definition, it is clear that depend on the heat flux from or to the ground or zero, Monin-Obukhov length L_{MO} varied from $-\infty$ to ∞. Magnitudes of L_{MO} characterised the height where mechanical and buoyant production of turbulence are in balance.

According to Monin-Obukhov theory, velocity and potential temperature mean gradient can be expressed as:

$$\frac{\partial u}{\partial z} = \frac{u_*}{\kappa z}\phi_m(\zeta) \tag{1.2a}$$

$$\frac{\partial \theta}{\partial z} = \frac{\theta_*}{\kappa z}\phi_h(\zeta) \tag{1.2b}$$

where $\zeta = z/L_{MO}$ is stability parameter. Values of ζ are always negative under unstable condition and positive under stable condition. $\phi_m(\zeta)$, $\phi_h(\zeta)$ are universal similarity functions of momentum and heat accordingly derived from empirical data.

Similarity functions have many empirical forms derived from various flat and homogeneous site experiments. Businger et al. [5] used Dyer-Businger equation to derive the relationship between universal function of heat and momentum. They proposed that:

$$\begin{aligned} \phi_m^2 = \phi_h &= (1-16\zeta)^{-1/2} \quad (-5 < \zeta < 0) \\ \phi_m = \phi_h &= 1+5\zeta \qquad\qquad (0 \leq \zeta < 1) \end{aligned} \tag{1.3}$$

However, they also suggested an alteration of von Karman constant $\kappa = 0.35$ and under neutral atmospheric condition $Pr_t^{-1} = 1.35$. The criticism of unrealistic κ, Högström [6] provided a correction to the universal functions of Businger et al. [5] with $\kappa = 0.4$ and $Pr_t^{-1} = 1.05$:

$$\begin{aligned} \phi_m &= (1-19.3\zeta)^{-1/4} \qquad (-2 < \zeta < 0) \\ \phi_m &= 1+6\zeta \qquad\qquad\quad (0 \leq \zeta < 1) \\ \phi_h &= 0.95(1-11.6\zeta)^{-1/2} \quad (-2 < \zeta < 0) \\ \phi_m &= 0.95+7.8\zeta \qquad\quad (0 \leq \zeta < 1) \end{aligned} \tag{1.4}$$

The momentum diffusivity ν_t and heat diffusivity α_t are expressed in relation to the universal similarity functions as:

$$\begin{aligned} \nu_t &= \frac{\kappa z u_*}{\phi_m(\zeta)} \\ \alpha_t &= \frac{\kappa z u_*}{\phi_h(\zeta)} \end{aligned} \tag{1.5}$$

Mean wind and temperature profiles The velocity and temperature profiles can be specified from the integration of Equation (1.2a). These profiles can be written as:

$$u(z) = \frac{u_*}{\kappa}\left[\ln\left(\frac{z}{z_0}\right) - \psi_m\left(\frac{z}{L_{MO}}\right)\right] \tag{1.6}$$

$$\theta(z) = \theta_w \frac{\theta_*}{\kappa}\left[\ln\left(\frac{z}{z_0}\right) - \psi_h\left(\frac{z}{L_{MO}}\right)\right] \tag{1.7}$$

where z_0 is ABL surface roughness practically found from the wind profile. z_0 ranges from 10×10^{-4} m for calm open oceans and up to 3 m in case of urban site with tall buildings [7]. ψ_m, ψ_h are integrated forms of the similarity functions Equation (1.3) [8]:

$$\begin{aligned} \psi_m = \phi_m^2 &= \ln\left[\left(\frac{1+x^2}{2}\right)\left(\frac{1+x}{2}\right)^2\right] - 2\tan^{-1}x + \frac{\pi}{2} \quad (L_{MO} < 0) \\ \psi_h &= 2\ln\left(\frac{1+x^2}{2}\right) \quad (L_{MO} < 0) \\ \phi_m = \phi_h &= -5\frac{z}{L_{MO}} \quad (L_{MO} > 0) \end{aligned} \tag{1.8}$$

where $x = (1 - 16z/L_{MO})^{1/4}$

1.2.2 Mixed-layer similarity

In convective boundary layer, the height of ABL h_{ABL} is used as the length scale. Scaling parameters derived from mixed-layer similarity are convective velocity w_* and convective temperature scale T_* (Equation (1.9)):

$$\begin{aligned} w_* &= \left(\frac{g}{T_s} q_s h_{ABL}\right)^{1/3} \\ T_* &= \frac{q_s}{w_*} \end{aligned} \tag{1.9}$$

Turbulence root-mean-square in horizontal directions σ_u, σ_v are independent of heights as Equation (1.10). The vertical component σ_w increases with height, reaches maximum in the middle then sharply decreases in the upper part of mixed layer. However, a constant value $\sigma_w = 0.6$ can be used as simplified parametrisation in convective layer.

$$\frac{\sigma_u}{w_*} \approx \frac{\sigma_v}{w_*} \approx 0.6; \tag{1.10}$$

The height of ABL h_{ABL} depends on the ABL stability. Under unstable condition, h_{ABL} is typically in the order of 1000 m to 1500 m. For neutral boundary and stable condition, h_{ABL} can be estimated as [7]:

$$\begin{aligned} h_{ABL,neutral} &= 0.3\frac{u_*}{f_c} \\ h_{ABL,stable} &= 0.4\sqrt{\frac{u_* L_{MO}}{f_c}} \end{aligned} \tag{1.11}$$

where u_* is friction velocity, L_{MO} is Monin-Obukhov length, Coriolis parameter f_c defined from the Earth rotational speed $\omega_E = 7.292 \times 10^{-5}\,\mathrm{s}^{-1}$ and the latitude Φ_E as:

$$f_c = 2\omega_E \sin\Phi_E \tag{1.12}$$

1.3 Dense Gas Dispersion

Dense gas dispersion results from heavier-than-air gas release such as CO_2, Chlorine or release at cryogenic temperature such as Liquefied Natural Gas (LNG). Koopman and Ermak [9] discussed two specified denser-than-air cloud behaviours: stable density stratification which results a reduction of vertical turbulent mixing and horizontal gravity-driven flow due to the density gradient. These two effects result a lower and wider cloud observed from LNG vapour experiments.

Releasing at cryogenic temperature, LNG vapour dispersion is one of the most complicated problem in dense gas dispersion. Some key physics involved in the dispersion process of LNG vapour are wind speed, surface roughness, atmospheric stability, terrain effect, and transition to passive dispersion [10]. Releasing at boiling point, LNG vapour cloud has density higher than ambient. Therefore, it exhibits dense gas dispersion behaviours. Reduced turbulent mixing between the dense gas and the surrounding makes ambient air has less significant role in dilution process [11]. This effect may result the lingering of dense gas

cloud, where the cloud travels downwind at a slower rate than the ambient. Experiment observation from `Burro8` test (Figure 1.2a) shows the reduction of wind velocity in the vapour cloud. The highest reduction of wind velocity is at 1 m, while it has insignificant change at 8 m height. This implies that turbulence within cloud is dramatically reduced, and the dispersion process was dominated by the gravity flow. At large spill rate, low wind speed, and stable atmospheric condition, the decoupling between denser-than-air cloud and surrounding will make it more difficult for ambient turbulent air to penetrate the cloud and result a bifurcation structure, where the cloud split into two plume at the centre line (as observed in Figure 1.2b). These are also the worst conditions for dispersion of LNG vapour which result the furthest downwind distance to Lower Flammability Limit (LFL).

Heat transfer from the surrounding and ground surface to the cold LNG vapour cloud is another important factors affecting the LNG vapour dispersion. Other relating heat transfer phenomenon is heat addition or heat removal due to the condensation or evaporation of water vapour and long wave heat radiation. However, the most dominant heat budget to the cold LNG vapour cloud is from the surrounding air and ground surface. The major effect of heat transfer to the LNG dispersion is changing its properties (due to temperature change) and increasing turbulent mixing process which then decreasing the distance to LFL of the vapour cloud. Heat introduced to the cloud will increase its temperature, reduce cloud density. Therefore, shifting the cloud behaviour from dense gas to buoyant gas. Figure 1.2c shows the horizontal concentration of the cloud. It can be seen that the contour of 5 % is elevated, suggesting the evidence of buoyancy which cannot be shown in other tests. Then, it can be concluded that a small part of the cloud can become lighter-than-air if wind speeds are low enough and LNG vapour clouds linger sufficiently long. Therefore, the LNG dispersion model must also take into account the passive dispersion phase. Variable material properties, heat transfer from air to the cloud model and a ground-level heat transfer model are also needed for a sound prediction of LNG dispersion.

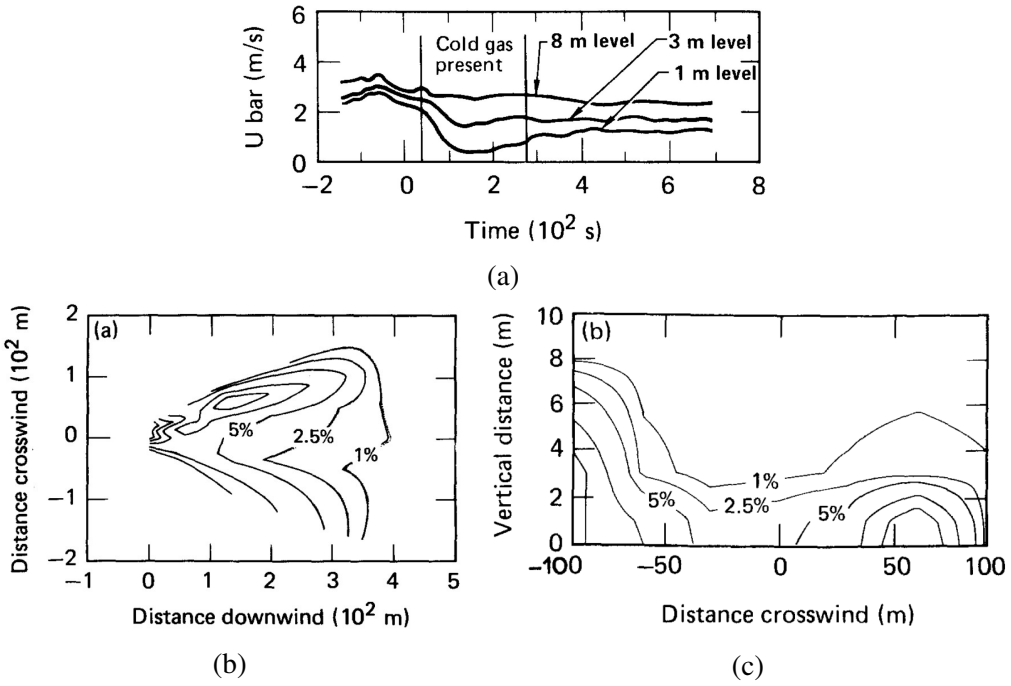


Fig. 1.2 (a) Mean wind speed during `Burro8` test at station T2 (57, 0, 1). (b) Horizontal concentration contour at 1 m above ground level of `Burro8` test at 200 s. (c) Vertical concentration contours at 400 m downwind at the time of 400 s of `Burro8` test. These figures are extracted from [11]

.

1.4 Model evaluation

In the context of evaluating the LNG dispersion model, a tool developed for National Fire Protection Agency (NFPA), so called the Model Evaluation Protocol (MEP) is used. It provides criteria and structure to fully evaluate a dispersion model. It is a three-stages procedure including: scientific assessment, model verification and model validation [10]. Validation is a process that comparing model outputs to measurements over applicable range of the model. This procedure involves a number of aspects including key physics and variables involving the LNG vapour dispersion, selection of scenarios covering the key physical process, identification of validation data sets and physical comparison parameters and selection of statistical performance measures (SPM) and quantitative assessment criteria defining the acceptable range of SPM [10]. The latter two aspects will be discussed in this section.

1.4.1 Validation data sets

In context of LNG vapour dispersion, Health and Safety Laboratory (HSL) created a set of full scale experimental data and wind tunnel test for model validation. The data set has 26 test configurations comprising field tests and wind tunnel tests as summarised in Table 1.2. Most configurations from field tests were under neutral or unstable atmospheric, excluding two high quality data sets from Thorney Island tests which were under a stable atmospheric condition. All field tests were in unobstructed terrain excepts the Falcon series tests which involve a large fence surrounding the LNG source. Most configurations from wind tunnel tests involved obstacles and terrains. Therefore, these tests are mainly used to investigate the effect of obstruction. The data is available in the REDIPHEM database [12] including physical comparison parameters of each test. These are 'maximum arc-wise concentration' which is the maximum concentration across an arc at the specified distance from the source; 'point-wise concentration' data which is the concentration at specific sensor

locations; 'point-wise temperature' data for field tests which is not available for wind-tunnel tests as these were conducted under isothermal condition.

LNG spill tests were conducted in field scale and also wind-tunnel scale. These data are sources to support model development, i.e. being used as the benchmark data to validate dispersion models.

Field scale experiments In U.S., field scale experiments of LNG spills were conducted by Lawrence Livermore National Laboratory (LLNL) from 1977 to 1988. These included Avocet series (1978) conducted in the old spill test facility in China Lake, then upgrading for Burro series (1980), followed by Coyote series in 1981 [13]. A larger spill test facility was constructed for Falcon series in 1987 which was aimed at evaluating the effectiveness of a containment fence and water curtain [14]. During that time, series of similar field tests were carried out independently in U.K. A series of LNG and LPG trials at Maplin Sands were conducted by Shell Research in 1980. HSE examined the dispersion of fixed-volume heavy gas releases in 1984 at Thorney Island. Advantica, acquired by the Germanischer Lloyd (GL) Group in 2007, also carried out experiments on the hazard relating to LNG operations which data was reviewed in [15]. In 2000s, Some experimental tests are carried out but limited data are publicly available such as MUST series [16], MID05 [17], MKOPSC [18]. More recently, Hanna et al. [19] conducted Jack Rabbit field experiments which are releases of one or two tons of pressurized liquefied chlorine and ammonia into a depression; Schleder et al. [20] carried out propane cloud dispersion field tests with and without fence obstructing.

The Burro series test was conducted by LLNL in 1980 aiming at examining the dispersion of LNG vapour under a variety of meteorological conditions. The test consisted of 8 continuous, finite duration releases of LNG onto an approximate 58 m diameter water pond. The Burro test site can be shown in Figure 1.3. `Burro3` was conducted under the most unstable atmospheric conditions. Under unstable atmospheric conditions and low spill rate, the test had the least maximum distance to the LFL. `Burro7` had the largest spill volume,

39.3 m^3, with the longest spill duration of 174 s. As seen in Figure 1.4, the test had the typical steady state characteristics of LNG dispersion defining as the state when vaporization rate equals the spill rate and the cloud reaches its furthest distance to LFL downwind [9]. The test reached its steady state for about 150 s at 140 m down wind, and concentrations varying from 3 % to 7 %. The largest distance to LFL was observed in the `Burro8` test which was in the most stable atmospheric condition and lowest wind speed. Table 1.1 listed meteorological parameters of experiments in Burro series tests.

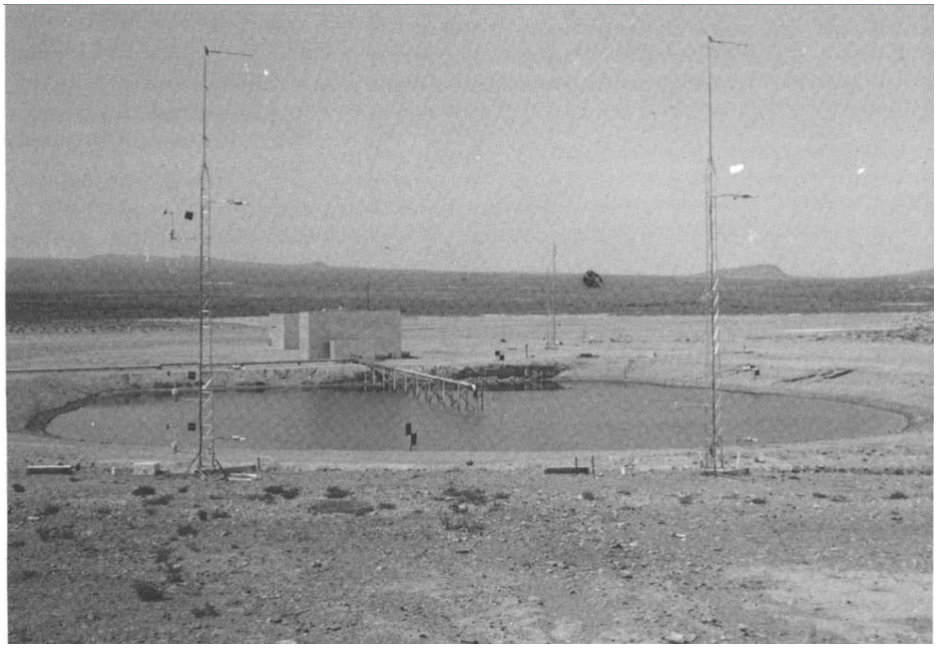


Fig. 1.3 Burro Test Site [21]

The Falcon series were conducted by LLNL in 1987. These comprises 5 large-scale LNG spill tests aiming at evaluating the effectiveness of impoundments walls as a mitigation technique for accidental releases of LNG. LNG was spilled onto a rectangular water pond (60m x 40m). The evaporation rate could be roughly equivalent to the spill flow rate as the designed recirculation system was involved to maximize the evaporation process [22]. LNG was supplied to the pond through 4 pipes, fitted with 0.11m diameter orifices and spaced at 90 degree intervals. The vapour fence, about 8.7m high, surrounded the water pond of a total area of 44m x 88m. The billboard of 13.3m tall, 17.1m wide was used to simulate the effect

Table 1.1 Burro tests summary extracted from [21]

	Burro3	Burro7	Burro8	Burro9
Spill volume (m^3)	34	39.4	28.4	24.2
Spill time (s)	166	174	107	78
Average wind velocity ($m\,s^{-1}$)	5.4	8.4	1.8	5.7
Wind direction (o)	224	208	235	232
Relative humidity (%)	5.2	7.1	4.6	13.1
Temperature at 2 m (°C)	33.8	33.7	33.1	35.4
Sensible heat flux ($W\,m^{-2}$)	-154	-41	2.2	-10
Atmospheric stability	B	D	E	D
Friction velocity ($m\,s^{-1}$)	0.249	0.372	0.074	0.252
Monin-Obukhov length (m)	-9.06	-114	+16.5	-140
Surface roughness length (m)	2×10^{-4}	2×10^{-4}	22×10^{-4}	22×10^{-4}

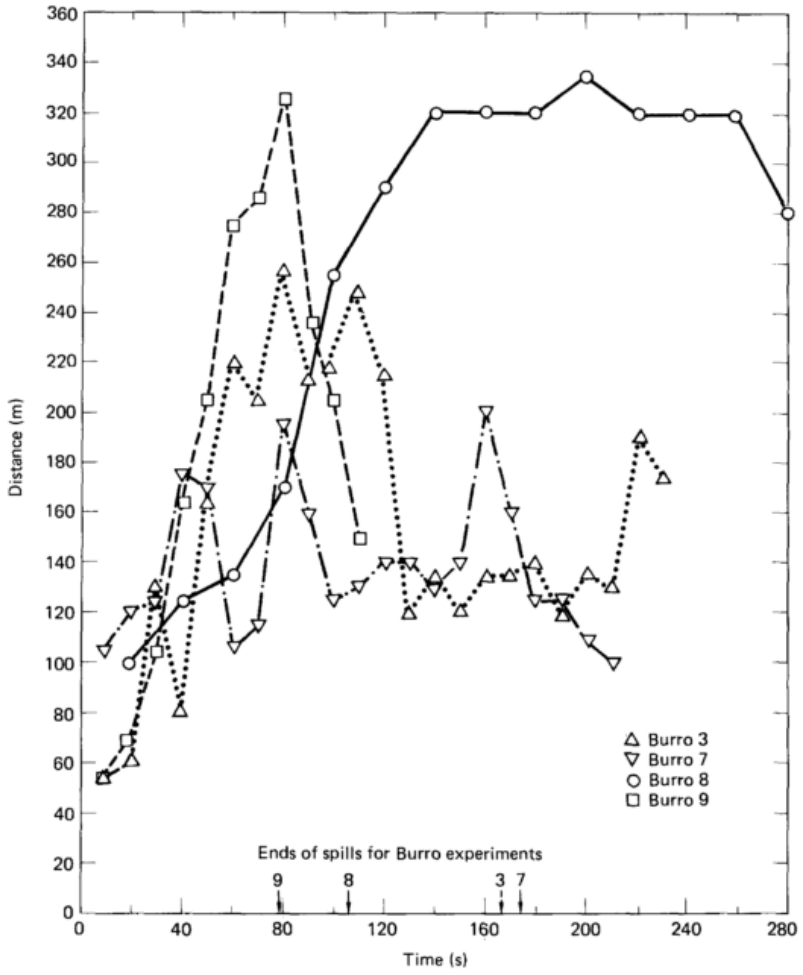


Fig. 1.4 Distance to LFL at 1 m height of Burro tests [21]

of a storage tank or other obstruction. The terrain was flat and the atmospheric condition was stable or neutrally stable.

Wind-tunnel test Wind-tunnel scale tests in The Meteorological Institute at the University of Hamburg (UH), TNO Division for Technology for Society (TNO), Warren Spring Laboratory (WSL) were recorded in REDIPHEM database.

Table 1.2 Validation data set [10]

Experiments	Type	Trials/cases	Description
Maplin Sand (1980)	Field	27, 34, 35	LNG/LPG dispersion over sea
Burro (1980)	Field	3, 7, 8, 9	LNG
Coyote (1980)	Field	3, 5, 6	LNG
Thorney Island (1982 - 1984)	Field	45, 47	Freon 12/N_2 mixture Continuous release
CHRC (2006)	Wind tunnel	A (without obstacles) B (with storage tank and dike) C (with dike)	CO_2
BA-Hamburg	Wind tunnel	DA0120/DAT223 (Unobstructed) 039051/039072 (Upwind fence) DA0501/DA0532 (Downwind fence) 039094/095/097 (Circular fence) DAT647/631/632/637 (Slope)	SF_6
BA-TNO	Wind tunnel	TUV01 (unobstructed), TUV02 (downwind fence), FLS (3-D mapping)	SF_6

1.4.2 Statistical Performance Measure (SPM)

SPMs are means to compare prediction parameters and the measured one. SPM chosen should reflect the bias of these predictions. There are five SPMs using for MEP including mean relative bias (MRB), mean relative square error (MBSE), the fraction of predictions within the factor of two of measurements (FAC2), geometric mean bias (MG) and geometric variance (VG). Definition and acceptability criteria for each SPM are presented in tabular

form as Table 1.3; C_m, C_p are the measured and simulated concentration accordingly, $\overline{A}$ denote the mean operation of variable A.

Table 1.3 Statistical Performance Measures [10]

	Definition	Acceptable criteria
MBR	$\overline{\left(\frac{C_m - C_p}{0.5(C_m - C_p)}\right)}$	$-0.4 < MBR < 0.4$
$MRSE$	$\overline{\left(\frac{(C_m - C_p)^2}{0.25(C_m + C_p)^2}\right)}$	$MRSE < 2.3$
$FAC2$	$\frac{C_m}{C_p}$	$0.5 < FAC2 < 2$
MG	$\exp\left(\overline{\ln \frac{C_m}{C_p}}\right)$	$0.67 < MG < 1.5$
VG	$\exp\left(\overline{\left(\ln \frac{C_m}{C_p}\right)^2}\right)$	$VG < 3.3$

1.5 Research objectives and scopes

The thesis is on developing a tool using framework of OpenFOAM version 5.0 [23] to simulate gas dispersion in ABL. The developed model takes into account different turbulence models to solve ABL turbulence. RANS turbulence models are used. The dispersion model takes into account the effects of gas buoyancy and heat transfer mechanism from the air surrounding to the gas cloud.

Dispersion of cold dense gas behaviour like LNG vapour with transition from dense gas to buoyant gas behaviour due to heat addition from the air surrounding to the vapour cloud, is examined to find the effectiveness of proposed model to replicate behaviours of such flow.

The thesis comprises of five parts. After literature review, the methodology developed using OpenFOAM tool to solve ABL turbulence and ABL dispersion of gas is presented. The proposed model is used to simulate turbulence of different thermal stratified ABL in the

third chapter. The fourth chapter devotes to apply OpenFOAM in simulation of dense gas in ABL. Final part includes conclusions and future works that may be induced from the thesis.

Chapter 2

Literature Review

2.1 Neutral Atmospheric Boundary Layer modelling

2.1.1 Horizontally homogeneous boundary layer

The important task before modelling gas dispersion flow in the ABL is obtaining initial state of ABL flow. One approach to achieve this is using equilibrium ABL, i.e. zero stream-wise gradients of all variables, as the state of ABL flow. For neutral atmospheric boundary layer, Richards and Hoxey [24] proposed boundary conditions of mean wind speed and turbulence quantities for the standard $k-\varepsilon$ model. These profiles were derived assuming constant shear stresses, which is validated for the surface layer of ABL. Therefore, they were used to model ABL surface layer as horizontally homogeneous turbulent surface layer (HHTSL). However, HHTSL was hard to achieved mostly due to the ground boundary conditions [25], which manifested in the decay of velocity profile due to a spike in the turbulent kinetic energy profile close to the ground. The consistency between wall boundary conditions, turbulence model with associated constants and numerical schemes was shown to achieve HHTSL [26, 27, 28]. Under HHTSL, the governing equations for velocity u, turbulent kinetic energy k and turbulent dissipation rate ε can be simplified as [27]:

$$\begin{aligned} \nu_t \frac{\partial u(z)}{\partial z} &= \frac{\tau_w}{\rho} = u_*^2 \\ \frac{\partial}{\partial z}\left(\nu_t \frac{\partial k}{\partial z}\right) + S_k &= 0 \\ \frac{\partial}{\partial z}\left(\frac{\nu_t}{\sigma_\varepsilon}\frac{\partial \varepsilon}{\partial z}\right)(C_{1\varepsilon} - C_{2\varepsilon})\frac{\varepsilon^2}{k} + S_\varepsilon &= 0 \end{aligned} \tag{2.1}$$

where $C_{1\varepsilon}$ and $C_{2\varepsilon}$ are $k-\varepsilon$ the turbulent model's constants. σ_ε is turbulent Prandtl number of ε. S_k and S_ε are the source terms of k and ε equation respectively.

The inlet boundary conditions proposed by Richards and Hoxey [24], which are widely used in CFD study of atmospheric flow, can be written as:

$$\begin{aligned} u(z) &= \frac{u_*}{\kappa}\ln\frac{z+z_0}{z_0} \\ k &= \frac{u_*^2}{\sqrt{C_\mu}} \\ \varepsilon &= \frac{u_*^3}{\kappa(z+z_0)} \end{aligned} \tag{2.2}$$

where C_μ is the $k-\varepsilon$ turbulent model's constant.

These inlet profiles are assured a solution of Equation (2.1) provided that the $k-\varepsilon$ turbulent model's constant and σ_ε satisfy the following equation:

$$\sigma_\varepsilon = \frac{\kappa^2}{(C_{\varepsilon 2} - C_{\varepsilon 1})\sqrt{C_\mu}} \tag{2.3}$$

Instead of altering the model constants, Pontiggia et al. [29] derived the z-dependent source term for ε equation from solution of Equation (2.2):

$$S_\varepsilon = \frac{\rho u_*^4}{(z+z_0)^2}\left[\frac{(C_{\varepsilon 2} - C_{\varepsilon 1})\sqrt{C_\mu}}{\kappa^2} - \frac{1}{\sigma_\varepsilon}\right] - \mu\frac{\rho u_*^3}{2\kappa(z+z_0)^3} \tag{2.4}$$

Under turbulence case, molecular viscosity is negligible. Therefore, the second term is usually ignored.

The turbulence kinetic energy k has limited effect in the velocity field but not the concentration field because it can enhance the gas dispersion effect by turbulence. Constant inlet turbulence kinetic energy proposed by Richards and Hoxey [24] was subjected to many arguments. As noted by Parente et al. [27], the decreasing k with height was shown in many wind tunnel tests. Yang et al. [25] proposed new the profiles of k and ε for standard $k-\varepsilon$ model. k, ε are the non-linear functions of height as:

$$\begin{aligned} k &= \frac{u_*^2}{C_\mu^{1/2}}\sqrt{C_1 \ln\left(\frac{z+z_0}{z_0}\right)+C_2} \\ \varepsilon &= \frac{u_*^3}{\kappa(z+z_0)}\sqrt{C_1 \ln\left(\frac{z+z_0}{z_0}\right)+C_2} \end{aligned} \tag{2.5}$$

where $C_1 = -0.17$ and $C_2 = 1.62$ are constants fitted from their wind tunnel experiments. They also proposed modified standard model constants in Table 2.1.

Table 2.1 The modified $k-\varepsilon$ model constants by Yang et al. [25]

$C_{1\varepsilon}$	$C_{2\varepsilon}$	C_μ	σ_k	σ_ε
1.5	1.92	0.028	1.67	2.51

Parente et al. [27] presented an elaborate procedure to ensure the consistency for arbitrary inlet profile of k. Instead of altering model constants as Yang et al. [25], the effect of non-constant k on momentum and ε equations can be characterised by deriving an equation for C_μ:

$$C_\mu(z) = \frac{u_*^4}{k(z)^2} \tag{2.6}$$

Source terms are added to the k and ε transport equations to obtain equilibrium condition:

$$\begin{aligned} S_k &= \frac{\rho u_* \kappa}{\sigma_k}\frac{\partial}{\partial z}\left((z+z_0)\frac{\partial k}{\partial z}\right) \\ S_\varepsilon &= \frac{\rho u_*^4}{(z+z_0)^2}\left[\frac{(C_{\varepsilon 2}-C_{\varepsilon 1})\sqrt{C_\mu}}{\kappa^2}-\frac{1}{\sigma_\varepsilon}\right] \end{aligned} \tag{2.7}$$

Richards and Norris [30] revisited the problem of modelling the HHTSL by deriving inlet profiles directly from the conservation and equilibrium equations. Using this approach, various inlet profiles can be specified by varying the constants of turbulence model. For standard $k-\varepsilon$ models, the inlet profiles of velocity and turbulence properties are the same as Equation (2.2). However they suggested to change the von Karman constant according to the turbulent model's constants as:

$$\kappa_{k-\varepsilon} = \sqrt{(C_{\varepsilon 2} - C_{\varepsilon 1})\sigma_\varepsilon \sqrt{C_\mu}} \tag{2.8}$$

Using the standard $k-\varepsilon$ model constants (Table 3.1), we can yield $\kappa_{k-\varepsilon} = 0.433$.

Hargreaves and Wright [31] had shown that the zero gradient velocity at the top boundary resulted in a decay of velocity downstream because of the extraction energy at the wall due to the wall shear stress effect. A driving shear stress, zero flux of turbulent kinetic energy and a flux of dissipation rate ε should be imposed at the upper boundary:

$$\begin{aligned} \frac{du}{dz} &= \frac{u_*}{\kappa z} \\ \frac{\mu_t}{\sigma_\varepsilon}\frac{d\varepsilon}{dz} &= -\frac{\rho u_*^4}{\sigma_\varepsilon z} \end{aligned} \tag{2.9}$$

In $k-\omega$ model, the specific dissipation ω is solved instead of dissipation rate ε. The profiles of U and k are the same, except the new effective von Karman constant is $\kappa_{k-\omega} = 0.408$. The profile of ω is derived from following equation:

$$\omega = \frac{u_*}{C_\mu^{1/2}\kappa_{k-\varepsilon} z} \tag{2.10}$$

Similarly to the $k-\varepsilon$ turbulence model, a flux of ω should be imposed at top boundary:

$$\mu_t \frac{d\omega}{dz} = -\frac{\rho u_*^2}{C_\mu^{1/2} z} \tag{2.11}$$

In the presence of obstacles, Richards and Norris [30] had shown that the eddy viscosity models like $k-\varepsilon$ or $k-\omega$ resulted in the over-prediction of stagnation pressures, while Reynolds stress model (RSM) [32] significantly reduced this issue.

2.1.2 Boundary conditions

At the outlet boundary, the flow is assumed fully developed and unidirectional. All flow variables are supposed to be constant in this boundary. Therefore, the placement of this boundary is very important. When the placement is so close to the source, significant errors can be made due to the propagation of the source to other boundaries. Otherwise, if the placement is much far from the source, the computational resource increases dramatically.

The top and side of the computational domain are external boundaries representing the far fields of flow. If a constant pressure is applied in these boundaries, this may alter the inlet wind profile in case the prescribed pressure is not matched with the boundary velocity [7]. The zero gradient boundary condition, which set normal velocity to zero and all others variables are set equal to the inner values, or symmetry condition can be used at the top and side boundaries to reserve the wind profile and eliminate the effect of changing the inlet profiles.

At the wall boundary, two models usually applied for turbulence properties are Low Reynolds number (LRN) turbulence model [33] and high Re number (HRN) with wall functions [34]. HRN models are usually less accurate, and also sensitive to the mesh resolution close to the wall. Adaptive wall functions were developed to overcome the restriction of HRN, which is the first point above the wall to be placed in the logarithmic layer [35]. Bäckar and Davidson [36] proposed a hybrid approach, so called the numerical wall function, where wall adjacent cells are divided into sub-grid and the governing equations are solved with appropriate boundary conditions in this sub-grid.

When applying wall functions for the ground surface, Blocken et al. [37] addressed the inconsistency between the sand-grain rough wall model adopted in general CFD packages and the aerodynamic roughness length concept in ABL flow. They proposed the relationship between these two rough wall models:

$$C_s = \frac{E z_0}{k_s} \tag{2.12}$$

where the constant $E = 9.8$. The roughness constant C_s is used to model the roughness effect. k_s is the sand-grain equivalent roughness height and z_0 is the aerodynamic roughness length.

The mesh refinement near the ground boundary represented by the height of ground adjacent cell y_p is subjected to the following constraint specified between these two length scales:

$$z_0 < k_s < y_p \tag{2.13}$$

2.2 Stratified Atmospheric Boundary Layer modelling

Either the Reynolds Averaged Navier–Stokes (RANS) equations or Large Eddy Simulations (LES) [38, 39] are used for stratified atmospheric turbulence modelling. RANS turbulence models are still widely used in practical approach to overcome boundary conditions sensitivity and computational intensive of the LES.

Thermal stratification results from heat flux of the ground have significant effects to the buoyancy and ABL turbulence. For standard $k-\varepsilon$ model, the source term that accounts for gravity effects in ε equation can be written as [40]:

$$S_{\varepsilon b} = C_{\varepsilon 1}(1 - C_{\varepsilon 3})\frac{\varepsilon}{k} G_b \tag{2.14}$$

G_b is turbulent kinetic energy production source term due to buoyancy:

$$G_b = \beta g_i \frac{\mu_t}{\sigma_T} \left(\frac{\partial T}{\partial x_i} - \frac{g_i}{c_p} \right) \tag{2.15}$$

For the stable stratified ABL, turbulent kinetic energy k and dissipation rate ε can be derived from Monin-Obukhov similarity theory profiles and solving the $k-\varepsilon$ equation [7]. For the height $z \leq 0.1 h_{ABL}$:

$$\begin{aligned} k &= 6u_*^2 \\ \varepsilon &= \frac{u_*^3}{\kappa z} \left(1.24 + 4.3 \frac{z}{L_{MO}} \right) \end{aligned} \tag{2.16}$$

For the height $z > 0.1 h_{ABL}$:

$$\begin{aligned} k &= 6u_*^2 \left(1 - \frac{z}{h_{ABL}} \right)^{1.75} \\ \varepsilon &= \frac{u_*^3}{\kappa z} \left(1.24 + 4.3 \frac{z}{L_{MO}} \right) \left(1 - 0.85 \frac{z}{h_{ABL}} \right)^{1.5} \end{aligned} \tag{2.17}$$

Under the unstable ABL, the heat flux from the ground and the height of ABL play an important role in increasing the turbulence in the air flow. This vertical flow can be characterised using convective velocity scale w_* (Equation (1.9)). Turbulent kinetic energy k and dissipation rate ε under unstable ABL can be defined in Equation (2.18) for $z \leq 0.1 h_{ABL}$ and Equation (2.19) for $z > 0.1 h_{ABL}$.

$$\begin{aligned} k &= 0.36 w_*^2 + 0.85 u_*^2 \left(1 - 3 \frac{z}{h_{ABL}} \right)^{2/3} \\ \varepsilon &= \frac{u_*^3}{\kappa z} \left(1 + 0.5 \left| \frac{z}{L_{MO}} \right|^{2/3} \right)^{1.5} \end{aligned} \tag{2.18}$$

$$
\begin{aligned}
k &= w_*^2 \left[0.36 + 0.9 \left(\frac{z}{h_{ABL}} \right)^{2/3} \left(1 - 0.8 \frac{z}{h_{ABL}} \right)^2 \right] \\
\varepsilon &= \frac{w_*^3}{h_{ABL}} \left(0.8 - 0.3 \frac{z}{h_{ABL}} \right)
\end{aligned}
\tag{2.19}
$$

In order to simulate atmospheric stratification effects, Alinot and Masson [40] changed the model constants (Table 2.2) to achieve a better agreement with atmospheric profile from Monin-Obukhov theory.

Table 2.2 The standard $k-\varepsilon$ model constants proposed by Alinot and Masson [40]

$C_{1\varepsilon}$	$C_{2\varepsilon}$	C_μ	σ_k	σ_ε
1.176	1.92	0.0333	1	3.4 ($L_{MO} > 0$) -4.4 ($L_{MO} < 0$)

The effect of atmospheric stratification on dense gas dispersion CFD simulations was addressed by Pontiggia et al. [41]. The consistency between Monin-Obhukov profiles with $k-\varepsilon$ model is obtained by the addition of z-dependent source term S_ε to the ε transport equation. Under neutral atmospheric stability:

$$
S_\varepsilon(z) = \frac{\rho u_*^4}{z^2} \left[\frac{(C_{\varepsilon 2} - C_{\varepsilon 1})\sqrt{C_\mu}}{\kappa^2} - \frac{1}{\sigma_\varepsilon} \right] - \mu \frac{u_*^3}{2\kappa z^3} \tag{2.20}
$$

Under stable condition:

$$
S_\varepsilon(z) = \frac{\rho u_*^4}{z^2} \left[\frac{(C_{\varepsilon 2} - C_{\varepsilon 1})\sqrt{C_\mu}}{\kappa^2} \phi_\varepsilon^2 \sqrt{\frac{\phi_\varepsilon}{\phi_m}} - \frac{1}{\sigma_\varepsilon} \left(\frac{2}{\phi_m} - \frac{1}{\phi_m^2} + \frac{T_*}{\kappa T} \right) \right] - \mu \frac{u_*^3}{2\kappa z^3} \tag{2.21}
$$

where ϕ_m is the MOST similarity function of momentum (Equation (1.3)) and ϕ_ε is derived from ϕ_m using:

$$\begin{aligned} \phi_\varepsilon &= 1 - z/L(L < 0) \\ \phi_\varepsilon &= \phi_m - z/L(L > 0) \end{aligned} \tag{2.22}$$

In the case of cold dense gas dispersion, heat exchange of gas cloud and the ground surface is a significant heat transfer process. Forced heat convection model was used to find this heat flux [42]:

$$q_s = h_f(T_s - T_f) \tag{2.23}$$

where h_f is the local heat transfer coefficient, T_f is the fluid temperature. Kovalets and Maderich [43] used mixed coefficients between force heat convection and natural convection.

2.3 CFD simulation of ABL flow

There are several studies on using general CFD software such as FLUENT, STAR-CCM+ and OpenFOAM to simulate ABL flow. Pieterse and Harms [8] used STAR-CCM+ commercial code to simulate thermally stratified ABL with the standard $k-\varepsilon$ and SST $k-\omega$ turbulence model. Flores et al. [44] used Detached Eddy Simulation (DES) technique, which incorporates RANS models in near wall region and LES model in the rest of the domain, to simulate atmospheric wind circulation in open pit. The simulation was implemented in OpenFOAM and took in to account effects of buoyancy, stratification and complex geometry. A quasi-compressible approximation (treating density as explicit variable) was applied to model stratification effect. Riddle et al. [45] used FLUENT to simulate neutrally stable atmospheric boundary layer with RSM turbulence model.

Hargreaves and Wright [31] had shown that implementing the fixed flux boundary condition at the top boundary of the computational domain was difficult in general CFD software and usually replaced by the zero gradient condition (no shear stress). However,

this resulted in a decaying boundary layer. They also highlighted that using the default approach of modelling the near wall region in the general CFD software led to significantly unexpected results. Wall functions applied for atmospheric flow are necessarily modified since the standard wall functions are based on empirical data of sand grain roughed pipe [37] and aerodynamic roughness length z_0 is far from similar to this kind of roughness.

2.4 CFD simulation of dense gas dispersion

The values of turbulent Schmidt number Sc_t have significant effect on the dense gas dispersion concentration. Using wind tunnel data, Mokhtarzadeh-Dehghan et al. [46] reported that the value of Sc_t was dependent on the flow thermal stability. Sc_t can range from 0.5 to 2.3 for the flows with Richardson number from 0.1 to 16.

Both proprietary and open-source general CFD software packages are adopted in the simulations of atmospheric dense gas dispersion. Some studies used proprietary software such as ANSYS fluent [22, 47, 48, 49, 50], ANSYS CFX [51] and FLACS [52, 53, 20]. Other used open-source software such as OpenFOAM [54, 55, 56] and FDS [57, 58]. Hansen et al. [53] validated FLACS in all tests in model evaluation database of LNG vapour dispersion including both wind tunnel and field tests.

Several turbulence models were tested for dense gas dispersion problem. Mack and Spruijt [54] used standard $k-\varepsilon$ model with OpenFOAM solver `reactingFoam` to simulate wind tunnel test `DAT632`, which is the gravity driven flow of heavy gas in slope terrain. Different treatments of the buoyancy term in ε equation were investigated. It was shown that standard $k-\varepsilon$ was able to predict turbulence damping due to vertical negative density gradient. Gavelli et al. [22] applied the RSM turbulent model to take into account the directional effect of Reynold stress field (using standard $k-\varepsilon$ model as initial guess of turbulence). The `Falcon1` field test was used as the validation data. Gant et al. [59] simulated the CO_2 field test with realizable $k-\varepsilon$ model. Different RANS models were also tested for dense gas flow

in the presence of obstacles [51]. Tauseef et al. [60] used realisable $k-\varepsilon$ turbulence model for the simulations of two Thorney Island field experiments.

Table 2.3 presents all studies in dense gas dispersion in the literature using OpenFOAM. All classifications included are the solver, ground roughness approach, turbulent models, application and Sc_t. The solvers used including one takes into account gravitational effect like `rhoReactingBuoyantFoam`, treating the releasing gas as a passive scalar transport `turbulentScalarTransportFoam` or a mixture of multi-species like `reactingFoam` and `rhoReactingBuoyantFoam`. The equivalent sand grain roughness approach is used in all studies. RANS turbulent models were adopted in all studies. There is no paper using LES for gas dispersion simulation. All variants of RANS used are standard $k-\varepsilon$, realisable $k-\varepsilon$, RNG $k-\varepsilon$ and buoyant $k-\varepsilon$. Most studies were related to CO_2 dispersion applications. There is no study on LNG dispersion using OpenFOAM. For Sc_t parameters, most papers adopted a value of 0.6 or 0.7 without testing its effect on the gas concentration prediction. A recent paper [61] did a parameter sensitivity test on this with 3 values of Sc_t: 0.3, 0.7 and 0.9.

From the above literature review, all simulations using proprietary general CFD packages implemented the modification of sand grain roughness model for the modelling of atmospheric ground surface roughness. For consistency with inlet condition, the wall functions based on the aerodynamic roughness length z_0 should be used. This helps to ensure the consistency and overcome the constrain of near ground mesh refinement defined by Equation (2.13). The flexibility of open-source CFD package such as OpenFOAM facilitates implementing of such new boundary conditions.

2.5 Concluding remarks

From the investigation of literature, modifications of general CFD are required to successfully simulate the ABL turbulence. These was done intensively in commercial proprietary software.

Table 2.3 Results of studies on using OpenFOAM for atmospheric dense gas dispersion

Sources	Solver	Ground roughness treatment	turbulent models	Application	Sc_t
[61]	Not mentioned	sand-grain roughness	Standard, Realisable and RNG $k-\varepsilon$	CODASC, MUST	0.3, 0.7 and 0.9
[62, 63]	turbulentScalarTransportFoam	sand-grain roughness	Durbin $k-\varepsilon$	CO_2 dispersion over the urban area	0.6
[64]	CO2FOAM	sand-grain roughness	SST $k-\omega$	a puncture in a CO2 buried pipe	0.7
[55, 56]	rhoReactingBuoyantFoam	sand-grain roughness	SST $k-\omega$	An engineering case study of dense gas dispersion	
[54]	reactingFoam	sand-grain roughness	buoyant $k-\varepsilon$	continuous SF_6 gas release over hilly, sloping terrain	0.7

But little works were done in open-source code like OpenFOAM. The modification of general code should be done to successfully apply OpenFOAM in simulating ABL flows.

Ensuring accurate description of the ABL is an important task in any ABL flow study. This can be done by simulating the horizontally homogeneous ABL flow prior of dispersion study. Either the Reynolds Averaged Navier–Stokes (RANS) equations or Large Eddy Simulations (LES) are used for atmospheric turbulence modelling. RANS turbulence models are still widely used in practical approach to overcome boundary conditions sensitivity and computational intensive of the LES. Neutral and thermal stratified ABL should be taken into account to simulate ABL turbulence.

Turbulent Schmidt number Sc_t had a significant effect on ABL dense gas dispersion concentration [46]. Therefore, the solver should be able to take Sc_t as an input parameter. In case of LNG vapour dispersion, the buoyancy effect and ground heat transfer are two important factors. LNG vapour density changes with its cloud temperature. Therefore, it

behaves as dense gas in low temperature but as buoyant gas at higher temperature. The solver should take into account buoyancy effect in this situation.

Chapter 3

OpenFOAM Methodology

Physical properties of a fluid flow such as velocity, pressure, temperature are dependent variables of a mathematical model. In classical physics, governing equations are written from conservation principles, i.e. conservation of mass, momentum and energy. These equations define the change of governing variables temporally and spatially. In these equations, similar physical processes can be described by the same mathematical terms. For example, the conduction of heat and diffusivity of gas concentration are both modelled as diffusive terms. The general scalar transport equation represents modelled terms to characterise different physical processes affecting the change of scalar ϕ:

$$\frac{\partial \phi}{\partial t} + \nabla \cdot (\phi \boldsymbol{u}) + \nabla \cdot (\nabla \phi) = S_\phi \tag{3.1}$$

where ϕ is general scalar, $\boldsymbol{u}$ is the velocity vector, D is diffusivity coefficient. Terms in Eq. 3.1, from left to right, are temporal term, convective term, diffusive term and source term.

The aim of any numerical method is to obtain an approximate solution of mathematical model. In other word, the governing equations in form of partial differential equations are discretised at points in computational space to get algebraic equations system, which then is solved to get a solution for all discretised points. Therefore, the generation of system

of algebraic equations is implemented using two steps: domain discretisation and equation discretisation. Three discretisation method adopted in CFD are Finite Volume Method (FVM), Finite Difference Method (FDM) and Finite Element Method (FEM). Among them, FVM is used in mostly all well established CFD codes [65].

OpenFOAM was firstly introduced in literature by Weller and Tabor [66] based on FVM method. In this Thesis, two basic OpenFOAM solvers `buoyantSimpleFoam` and `rhoReactingBuoyantFoam` are modified substantially and adopted for atmospheric flow and atmospheric gas dispersion simulations. The structure of this Chapter is following. Firstly, the governing equations such as Navier-Stokes equations, energy and species transport equations are written in analytical form. Next, the FVM method using to solve these equations is described. Last but not least, two last Sections present algorithms of two OpenFOAM solvers used for the simulations of atmospheric flow and atmospheric gas dispersion.

In this Chapter, the governing equations are introduced in parallel with OpenFOAM implementation. OpenFOAM keywords for variables, solvers names and codes are formatted in the mono-font style, for example texttttkeywords. The detail of codes using in this Chapter is presented in Appendix A. The list of all variables and abbreviations can be shown in the *Nonmenclature*.

3.1 Governing equations

3.1.1 Momentum equation

The general momentum equation taking into account buoyancy effect can be written as [67]:

$$\frac{\partial}{\partial t}(\rho \boldsymbol{u}) + \boldsymbol{\nabla} \cdot (\rho \boldsymbol{u}\boldsymbol{u}) = -\boldsymbol{\nabla} p + \boldsymbol{\nabla} \cdot \tau + \rho \boldsymbol{g} \tag{3.2}$$

where ρ is fluid density, p is pressure, $\boldsymbol{g}$ is gravitational vector and τ is viscous stress tensor which is written for Newtonian fluids as:

$$\tau = \mu \left(\nabla \boldsymbol{u} + (\nabla \boldsymbol{u})^T \right) - \frac{2}{3} \mu (\nabla \cdot \boldsymbol{u}) \boldsymbol{\delta} \tag{3.3}$$

where μ is fluid dynamic viscosity, $\boldsymbol{\delta}$ is the Kronecker delta.

Pressure gradient and gravity force are combined to form p_{rgh} field:

$$-\nabla p + \rho \boldsymbol{g} = -\nabla p_{rgh} - (\boldsymbol{g} \cdot \boldsymbol{r}) \nabla \rho \tag{3.4}$$

where $\boldsymbol{r}$ is the position vector. p_{rgh} field can be seen as the pressure defined without hydrostatic pressure. p_{rgh} field is solved instead of pressure p.

3.1.2 Species transport equation

Transport equation for mass fraction Y for each specie α of a mixture, $\alpha = 1, 2, ..., N$, can be written as [68]:

$$\frac{\partial \rho Y_\alpha}{\partial t} + \nabla \cdot (\rho Y_\alpha \boldsymbol{u}) = \boldsymbol{j}_\alpha + \dot{r_\alpha} \tag{3.5}$$

where $\dot{r_\alpha}$ is the reaction rate of specie α. In the scope of this thesis, non-reacting flow is studied. Therefore, this term is neglected. $\boldsymbol{j}_\alpha$ is the specie mass flux and written using Fick's law as:

$$\boldsymbol{j}_\alpha = -\nabla \cdot (-\rho D \nabla Y_\alpha) \tag{3.6}$$

The mass diffusivity D can be derived from Schmidt number Sc using the analogy between momentum and mass transport. The Schmidt number Sc is defined as the ratio between kinematic viscosity ν and mass diffusivity:

$$Sc = \frac{\nu}{D} \tag{3.7}$$

3.1.3 Energy equation

Energy conservation equation can be written for specific fluid enthalpy h as [23]:

$$\frac{\partial \rho h}{\partial t} + \boldsymbol{\nabla} \cdot (\rho \boldsymbol{u} h) + \frac{\partial \rho K}{\partial t} + \boldsymbol{\nabla} \cdot (\rho \boldsymbol{u} K) - \frac{\partial p}{\partial t} = -\boldsymbol{\nabla} \cdot \boldsymbol{q} + \boldsymbol{\nabla} \cdot (\tau \cdot \boldsymbol{u}) + \rho \boldsymbol{g} \cdot \boldsymbol{u} + \rho r \tag{3.8}$$

K is specific kinetic energy defined as $K \equiv |\boldsymbol{u}|^2/2$, q is the heat flux and r is any specific heat source.

In OpenFOAM, the calculation of fluid temperature is done iteratively using Newton-Raphson method and the enthalpy h. The temperature T_j of j cell is calculated from its enthalpy h_j using:

$$\int_{T_{std}}^{T_j} c_p(T) dT = h_j \tag{3.9}$$

c_p is the specific heat capacity at constant pressure and can be expressed in the form of temperature dependent polynomial function (c_i is a coefficient of the function):

$$c_p(T) = \sum_{i=0}^{7} c_i T^i \tag{3.10}$$

3.2 Turbulence models

3.2.1 Reynolds-Averaged Navier-Stokes (RANS)

Reynolds-Averaged Navier-Stokes (RANS) equations derived by averaging Navier-Stokes equations are listed below [67] (the over-bar above a term symbol means the averaged value of that term):

$$\frac{\partial}{\partial x_i}(\rho \bar{u}_i) = 0 \tag{3.11}$$

$$\frac{\partial}{\partial t}(\rho \bar{u}_i) + \frac{\partial}{\partial x_j}(\rho \bar{u}_i \bar{u}_j) = -\frac{\partial \bar{p}}{\partial x_i} + \frac{\partial \tau_{ij}}{\partial x_j} + \frac{\partial}{\partial x_j}(-\rho \overline{u_i' u_j'}) + \rho g_i \tag{3.12}$$

where the velocity component u_i is decomposed into average part $\bar{u}_i$ and its fluctuation u_i'.

For a scalar ϕ, Reynolds-Averaged governing equation can be written as:

$$\frac{\partial}{\partial t}\left(\rho \bar{\phi}\right) + \frac{\partial}{\partial x_j}(\rho \bar{u}_j \bar{\phi}) = \frac{\partial}{\partial x_j}(\Gamma \frac{\partial \bar{\phi}}{\partial x_j}) + \frac{\partial}{\partial x_j}(-\rho \overline{u_j' \phi'}) \tag{3.13}$$

Turbulent fluxes are modelled using eddy-viscosity hypothesis:

$$-\rho \overline{u_i' u_j'} = \mu_t \left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right) - \frac{2}{3}\rho k \delta_{ij} \tag{3.14}$$

$$-\rho \overline{u_i' \phi'} = \Gamma_t \frac{\partial \phi}{\partial x_j} \tag{3.15}$$

where μ_t is turbulent dynamic viscosity, k is turbulent kinetic energy defined as $k \equiv \overline{u_i' u_i'}/2$, δ_{ij} is the Kronecker delta. Turbulent diffusivity Γ_t can be derived from turbulence Prandtl number Pr_t:

$$\Gamma_t = \frac{\mu_t}{Pr_t} \tag{3.16}$$

3.2.2 The $k-\varepsilon$ model

The $k-\varepsilon$ model relies on Prandtl-Kolmogorov expression of μ_t [69]:

$$\mu_t = \rho C_\mu \frac{k^2}{\varepsilon} \tag{3.17}$$

Two additional transport equations for turbulence kinetic energy k and turbulence dissipation rate ε are required. In OpenFOAM, the standard version from [34] is implemented. The transport equations for k and ε including buoyancy effect are:

$$\frac{D}{Dt}(\rho k) = \frac{\partial}{\partial x_i}\left[\left(\mu + \frac{\mu_t}{\sigma_k}\right)\frac{\partial k}{\partial x_j}\right] + G_k + G_b - \rho\varepsilon \tag{3.18}$$

$$\frac{D}{Dt}(\rho \varepsilon) = \frac{\partial}{\partial x_i}\left[\left(\mu + \frac{\mu_t}{\sigma_\varepsilon}\right)\frac{\partial \varepsilon}{\partial x_j}\right] + C_{1\varepsilon}\frac{\varepsilon}{k}G_k + C_{1\varepsilon}C_{3\varepsilon}\frac{\varepsilon}{k}G_b - C_{2\varepsilon}\rho\frac{\varepsilon^2}{k} \tag{3.19}$$

G_k is the production of turbulence kinetic energy due to mean velocity gradient:

$$G_k = -\rho\overline{u_i' u_j'}\frac{\partial \bar{u}_i}{\partial x_j} \tag{3.20}$$

G_b is the buoyancy source term:

$$G_b = -\frac{\mu_t}{\rho Pr_t}(\boldsymbol{g}\cdot\boldsymbol{\nabla}\rho) \tag{3.21}$$

Other parameters such as $C_{1\varepsilon}$, $C_{2\varepsilon}$, C_μ, σ_k and σ_ε are model constants. The modification of these constants from standard values according to Equation (2.3) is implemented to simulate the horizontal homogeneous ABL flow. These values are presented in Table 3.1 .

Table 3.1 The Modified $k-\varepsilon$ model constants

$C_{1\varepsilon}$	$C_{2\varepsilon}$	C_μ	σ_k	σ_ε
1.44	1.92	0.09	1	1.167

In case of undefined value of $C_{3\varepsilon}$, OpenFOAM [23] implements the default calculation of $C_{3\varepsilon}$ using:

$$C_{3\varepsilon} = tanh\left|\frac{v}{u}\right| \tag{3.22}$$

3.2.3 The SST $k-\omega$ model

The SST $k-\omega$ model [70] is shown to effectively solve turbulence in strong adverse pressure gradient and separation, which is usually failed when using standard $k-\varepsilon$ model.

$$\frac{D}{Dt}\rho k = \nabla\cdot(\rho(\mu+\alpha_k\mu_t)\nabla k)+\rho G_k-\frac{2}{3}\rho k(\nabla\cdot\boldsymbol{u})-\rho\beta^*\omega k+S_k. \tag{3.23}$$

$$\frac{D}{Dt}(\rho\omega) = \nabla\cdot(\rho(\mu+\alpha_\omega\mu_t)\nabla\omega)+\rho\gamma\frac{G_k}{\nu}-\frac{2}{3}\rho\gamma\omega(\nabla\cdot\boldsymbol{u})-\rho\beta\omega^2-\rho(F_1-1)CD_{k\omega}+S_\omega \tag{3.24}$$

G_k is the turbulent production term calculated similar to $k-\varepsilon$ model (Equation (3.20)). The blending function F_1 is a modifier for selecting $k-\omega$ or $k-\varepsilon$ turbulent model without user interaction. This function is zero away from the surface and switch to one inside the boundary layer:

$$F_1 = \tanh\left(\left(\min\left[\max\left(\frac{k^{1/2}}{\beta^*\omega y},\frac{500\nu}{y^2\omega}\right)\frac{4\rho\alpha_{\omega 2}k}{CD_{k\omega}y^2}\right]\right)^4\right) \tag{3.25}$$

$$CD_{k\omega} = \max\left(2\rho\alpha_{\omega 2}\frac{1}{\omega}\frac{\partial k}{\partial x_i}\frac{\partial\omega}{\partial x_i},10^{-10}\right) \tag{3.26}$$

The turbulent kinematic viscosity ν_t is calculated as:

$$\nu_t = a_1\frac{k}{\max(a_1\omega,b_1F_2\sqrt{2S_{ij}S_{ij}})} \tag{3.27}$$

F_2 is the second blending function:

$$F_2 = \tanh\left(\left[\max\left(\frac{2k^{1/2}}{\beta^*\omega y},\frac{500\nu}{y^2\omega}\right)\right]^2\right) \tag{3.28}$$

All other parameters are calculated from model constants given in Table 3.2 and blended coefficient F_1. For example, the α_k parameter is calculated as: $\alpha_k = \alpha_{k1}F_1 + \alpha_{k2}(1-F_1)$.

Table 3.2 The SST $k-\omega$ model constants

α_{k1}	α_{k2}	$\alpha_{\omega1}$	$\alpha_{\omega2}$	β_1	β_2	γ_1	γ_2	β^*	a_1	b_1	c_1
0.85	1.0	0.5	0.856	0.075	0.0828	5/9	0.44	0.09	0.31	1.0	10.0

3.2.4 Wall functions

The near-wall region can be divided into three parts: viscous sub-layer, buffer layer and logarithmic region. The extend of logarithmic region is increased with increasing Reynold number. Resolving the flow near boundary layer requires generally an excessive number of computational cells, because adjacent cells of the boundary are required to be in viscous layer, i.e. the wall length scale $y^+ = 1$ (defined in Equation (3.30)). Wall functions are used to overcome this restriction, by proposing a boundary condition at logarithmic region ($30 < y^+ < 200$). This is not only help to increase cell size of wall adjacent cell but also improve grid maximum aspect ratio, which in turn improving computational stiffness [35].

The derivations of wall functions are rooted from the generality of a region between wall and the outer edge of the logarithmic layer in the quasi-equilibrium boundary layer, e.g. flow over a flat plate at zero-pressure gradient [35]. The RANS equations are simplified in this case as:

$$\frac{d}{dy}\left((\mu+\mu_t)\frac{dU}{dy}\right) = 0 \tag{3.29}$$

Appropriate scales using for near wall region are:

$$u^+ = \frac{U}{u_*}, y^+ = \frac{yu_*}{\nu}, \nu_t^+ = \frac{\nu_t}{\nu}, k^+ = \frac{k}{u_*^2}, \varepsilon^+ = \frac{\varepsilon\nu}{u_*^4}, \omega^+ = \frac{\omega\nu}{u_*^2} \tag{3.30}$$

For RANS models, in viscous sub-layer ($y^+ < 5$), the flow is dominated by viscous effect. Therefore, the fluid shear stress is balanced with the wall shear stress. The velocity is linearly evolved from the wall.

$$u^+ = y^+ \tag{3.31}$$

In logarithmic region, the flow is dominated by turbulence stress. Assuming the sum of viscous and turbulent shear stress is constant and equal to the wall shear stress, as well as Prandtl's assumption for the turbulent viscosity $\nu_t^+ = \kappa y^+$, the velocity scale can be derived as:

$$u^+ = \frac{1}{\kappa}\ln\left(Ey^+\right) \tag{3.32}$$

$\kappa = 0.41$ is the von-Karman constant, E is a constant adjusted by empirical data. Calculation of u_* can be done from solving wall function (given velocity of the first cell) or from solving momentum equation [35]. Eddy-viscosity ν_t is implicitly related to the velocity profile (through κ). Therefore, it may be used to derive a boundary condition for turbulence variables to ensure consistency.

By applying turbulence kinetic energy equation for near wall region using non-dimensional variables defined in Equation (3.30), we have relation for logarithmic layer:

$$k^+ = \frac{C_k}{\kappa}\ln\left(y^+\right) + B_k \tag{3.33}$$

The constant $C_k = -0.416$ and $B_k = 8.366$ are used in OpenFOAM.

Similarly, for ε variable, we have a relationship in logarithmic layer:

$$\varepsilon^+ = \frac{1}{\kappa y^+} \tag{3.34}$$

For ω variable, the relationship is:

$$\omega^+ = \frac{1}{\kappa C_\mu^{1/2}(y^+)^2} \tag{3.35}$$

In OpenFOAM, there are two standard implementations of wall function for turbulence kinetic energy k: `kqRWallFunction` which is the Neumann boundary (zero gradient), and `kLowReWallFunction` which is the fixed value condition. The value of k at boundary patches is calculated from Equation (3.33) using y^+ defined from the known value of friction velocity u_*. u_* can be calculated from simple relation derived by Launder and Spalding [34] with an assumption that the generation and dissipation of turbulent energy are in balance:

$$u_* = C_\mu^{1/4} k_P^{1/2} \tag{3.36}$$

where k_P is turbulence kinetic energy in the first cell adjacent to the wall. Value of k at wall boundary k_w is calculated from k^+ as:

$$k_w = k^+ u_*^2 \tag{3.37}$$

Unlike k wall function, in which the variable value is defined at boundary surface, ε and ω wall functions set their values in cell centre. In `epsilonWallFunction`, ε value is averaged from all surface values defined as wall in the cell:

$$\varepsilon_P = \frac{C_\mu^{0.75} k^{1.5}}{\kappa y_P} \tag{3.38}$$

where y_P is the height of wall-adjacent cell.

ν_t wall function is also necessarily in OpenFOAM simulation to define wall shear stress τ_w as a remedy to the below approximation where wall velocity gradient is significantly larger than the velocity difference between values at the wall adjacent cell u_P and at the wall u_w:

$$\tau_w = \nu \frac{\partial u}{\partial n}|_w \approx \nu \frac{(u_P - u_w)}{y_P} \tag{3.39}$$

Equation for the wall shear stress can be derived using Equations (3.36) and (3.32):

$$\tau_w = \rho u_*^2 = \rho u_* \frac{(u_P - u_w)}{\frac{1}{\kappa}\ln(Ey^+)} \tag{3.40}$$

ν_t wall functions is derived from the above two equations as:

$$\nu_t = \nu \left(\frac{\kappa y^+}{\ln(Ey^+)} - 1 \right) \tag{3.41}$$

ν_t wall functions can also be used to implicitly define velocity at wall adjacent cell. There are several options to use ν_t wall functions in OpenFOAM. `nutLowReWallFunction` set $\nu_t = 0$ which means that the flow near wall must be sufficiently solved. `nutkWallFunction` and `nutUWallFunction` both use Equation (3.41) but with different calculation of y^+. The former uses assumption of Equation (3.36) for the calculation of y^+ as:

$$y^+ = \frac{y C_\mu^{1/4} k^{1/2}}{\nu} \tag{3.42}$$

whereas, `nutUWallFunction` uses Equation (3.32) to derive relationship between y^+ and u_P:

$$y^+ \ln\left(Ey^+\right) - \frac{\kappa y u_P}{\nu} = 0 \tag{3.43}$$

The above equation is solved using Newton-Raphson iterative method to find y^+.

Rough wall modelling In case of sand-grain type rough wall model, implemented in `nutkRoughWallFunction` class, the wall velocity scale can be written as [71]:

$$u^+ = \frac{1}{\kappa}\ln\left(Ey^+\right) - \Delta B \tag{3.44}$$

where ΔB is an adding term comparing to Equation (3.32) of the smooth wall:

$$\Delta B = \frac{1}{\kappa}\ln(f_r) \tag{3.45}$$

f_r quantifies a shift of the intercept due to roughness effects. For fully rough region:

$$f_r = 1 + C_s K_s^+ \tag{3.46}$$

The roughness constant C_s is used to model the roughness effect. A proper roughness constant is dictated mainly by the type of given roughness. The default roughness constant was retained as $C_s = 0.5$, which indicating tightly-packed, uniform sand-grain. The non-dimensional sand-grain roughness height K_s^+ is [35]:

$$K_s^+ = \frac{\rho K_s u_*}{\mu} \tag{3.47}$$

ν_t is calculated using [35]:

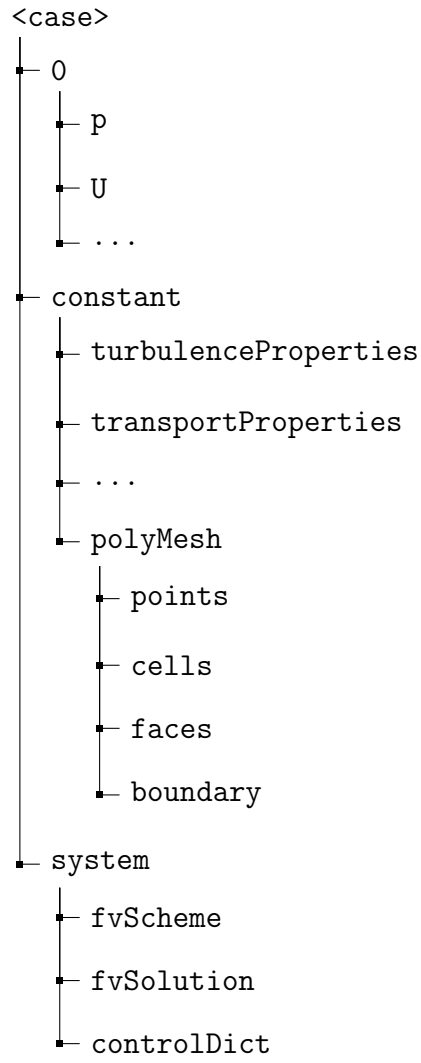
$$\nu_t = \frac{u_* \kappa y_P}{\ln\left(\frac{E y_P}{C_s k_s}\right)} \tag{3.48}$$

For modelling roughness effect in ABL flow, there should be a special attention on deriving wall functions from aerodynamic roughness length z_0. Using ABL velocity profile Equation (2.2) and Equation (3.36), we can have the wall function for ν_t as:

$$\nu_t = \frac{\kappa u_* y_P}{\ln\left(\frac{y_P + z_0}{z_0}\right)} \tag{3.49}$$

3.3 OpenFOAM simulation

A file structure of an OpenFOAM simulation consists of three main folders. Folder `0` contains all initial and boundary conditions of fields solved in the simulation. Folder `constants` comprises configuration files for all models used in the simulation such as: turbulence model, transport properties, thermal properties, etc. and `polyMesh` folder contains mesh information. Folder `system` uses to configure discretisation schemes, iterative solver and solution control. A general file structure of an OpenFOAM simulation is presented below:



3.3.1 Geometry and Mesh data structure

The mesh data is stored in `constant/polyMesh` directory. The typical mesh data files include `points`, `faces`, `owner`, `neighbour` and `boundary`. These files are interconnected to define the mesh.

Unlike a structure mesh constructed in the per-point basis, a mesh in OpenFOAM is constructed on a per-face basis. It characterises in the owner-neighbour addressing used to determine the topological mesh structure. In owner-neighbour addressing, the ordered list of indexes in the mesh faces file is used to define the owner cells and neighbour cells where owner cells are ones have lower index. The face area normal is directed from the owner to the neighbour cell. This can help to reduce the redundancy of computing discretised operations.

3.3.2 Discretisation schemes

Time discretisation term Discretisation schemes are defined using `ddtSchemes` sub-dictionary. Some selected options are:

- `steadyState`: used for steady state simulation.
- `Euler`: used for transient simulation. It is the implicit first order scheme for the time derivatives. The scheme is unconditionally bounded.
- `CrankNicolson`: also used for transient simulation. It is the implicit, second order scheme. An off-centering coefficient is requires for the scheme to be bounded.

Convective term In OpenFOAM, `divSchemes` sub-dictionary contains parameters for discretisation of the term $\nabla \cdot \Phi Q$, where Q is the scalar field and Φ is a surface mass flux defined as:

$$\Phi = \boldsymbol{u}_f \cdot \boldsymbol{S}_f \tag{3.50}$$

An interpolation scheme is required for the calculation of Φ. Φ is calculated using:

$$\Phi_f = \Phi_N + w(\Phi_P - \Phi_N) \tag{3.51}$$

where w is the overall weight derived from the selected interpolation scheme. The subscript f represents value at a surface. The subscript N and P denote two adjacent cell centre points of that surface.

Diffusion term `Gauss` scheme is used for discretisation of Laplacian term $\nabla \cdot (\nu \nabla \boldsymbol{u})$, which is diffusion term in momentum equation. Interpolation scheme for the diffusion coefficient and surface normal gradient scheme for evaluating $(\nabla \boldsymbol{u})$ are also required. All parameters are defined in `laplacianSchemes` sub-dictionary.

Surface normal gradient schemes required for Laplacian term is defined at face surface. `orthogonal` scheme is second order accurate and applied when the vector connecting the cell centres is orthogonal to the face. In case of non-orthogonality, `corrected` scheme is used to maintain second-order accuracy, i.e. an explicit non-orthogonal correction is added. `limited corrected` scheme is used for severe non-orthogonality mesh which may lead to unstable solution.

3.3.3 Momentum-pressure coupling algorithms

Pressure equation Pressure equation is used to enforce continuity constrain. Solving pressure equation can assure velocity field to satisfy continuity equation. Poison equation of pressure in Cartesian coordinate [67] has the form of:

$$\frac{\partial}{\partial x_i}\left(\frac{\partial p}{\partial x_i}\right) = -\frac{\partial}{\partial x_i}\left[\frac{\partial}{\partial x_j}\left(\rho u_i u_j - \tau_{ij}\right)\right] + \frac{\partial^2 \rho}{\partial t^2} \tag{3.52}$$

Implicit method for solving momentum Equation (A.1) can be written in discretised form as:

$$A_P^{u_i} u_{i,P}^{n+1} + \sum_l A_l^{u_i} u_{i,l}^{n+1} = Q_{u_i}^{n+1} - \left(\frac{\delta p^{n+1}}{\delta x_i} \right)_P \tag{3.53}$$

where P is index of velocity node u_i, l denotes neighbour cells, source term Q contains all explicit terms defined using velocity at the previous time step u_i^n and other linearised terms depend on the new time step variables $n+1$. A is sparse square coefficients matrix. The pressure term is written in symbolic difference form.

Solving Equation (3.53) is done by iterative method. Outer iteration counter m is used to denote the current prediction u_i^m of the actual value of u_i^{n+1} at the current time step. Equation solved in each outer iteration is:

$$A_P^{u_i} u_{i,P}^{m*} + \sum_l A_l^{u_i} u_{i,l}^{m*} = Q_{u_i}^{m-1} - \left(\frac{\delta p^{m-1}}{\delta x_i} \right)_P \tag{3.54}$$

where $m*$ is introduced to denote the predicted value of u_i^m. This value is usually not satisfied the continuity equation. Velocity field can be written from the above equation as:

$$u_{i,P}^{m*} = \tilde{u}_{i,P}^{m*} - \frac{1}{A_P^{u_i}} \left(\frac{\delta p^{m-1}}{\delta x_i} \right)_P \tag{3.55}$$

$$\tilde{u}_{i,P}^{m*} = \frac{Q_{u_i}^{m-1} - \sum_l A_l^{u_i} u_{i,l}^{m*}}{A_P^{u_i}} \tag{3.56}$$

where $\tilde{u}_{i,P}^{m*}$ contains all terms excluding pressure term as presents in Equation (3.56). Since velocity field is calculated from previous pressure, it should be corrected to satisfy continuity equation:

$$\frac{(\delta u_i^m)}{\delta x_i} = 0 \tag{3.57}$$

The discretised Poison pressure equation derived from continuity equation is used to correct the velocity field:

$$\frac{\delta}{\delta x_i}\left[\frac{\rho}{A_P^{u_i}}\left(\frac{\delta p^m}{\delta x_i}\right)\right]_P = \left[\frac{\delta(\rho \tilde{u}_i^{m*})}{\delta x_i}\right]_P \tag{3.58}$$

Solving discretised Poison equation results a velocity field satisfies the continuity equation. However, the corrected velocity field and pressure field no longer satisfy the momentum equation, i.e. Equation (3.53). Therefore, other outer iterations are performed until both momentum and continuity equation are satisfied.

In practice, pressure correction p' and velocity correction u' are solved instead of actual values:

$$\begin{aligned} u_i^m &= u_i^{m*} + u' \\ p^m &= p^{m-1} + p' \end{aligned} \tag{3.59}$$

Equations (3.55), (3.56) and (3.58) are rewritten in term of velocity correction and pressure correction as:

$$u'_{i,P} = \tilde{u}'_{i,P} - \frac{1}{A_P^{u_i}}\left(\frac{\delta p'}{\delta x_i}\right)_P \tag{3.60}$$

where $\tilde{u}'_{i,P}$ is written as:

$$\tilde{u}'_{i,P} = -\frac{\sum_l A_l^{u_i} u'_{i,l}}{A_P^{u_i}} \tag{3.61}$$

The pressure correction equation:

$$\frac{\delta}{\delta x_i}\left[\frac{\rho}{A_P^{u_i}}\left(\frac{\delta p'}{\delta x_i}\right)\right]_P = \left[\frac{\delta(\rho u_i^{m*})}{\delta x_i}\right]_P + \left[\frac{\delta(\rho \tilde{u}'_i)}{\delta x_i}\right]_P \tag{3.62}$$

The last term in pressure correction Equation (3.62) is unknown which is neglected when solving for pressure field. When convergence solution is reached, the velocity correction

approaches zero and so does this term. However, this results in a slow convergence rate of pressure field. Momentum-pressure coupling algorithm presented in the next section is employed to achieve a better rate of convergence.

SIMPLE Semi-Implicit Method for Pressure Linked Equations (SIMPLE) algorithm overcomes the slow convergence issue resulting from neglecting the term in Equation (3.62) by updating velocity field using Equation (3.59) and (3.60). To improve stability of a computation, under-relaxation is combined with SIMPLE. The variable change is limited from one iteration to the next:

$$p^m = p^{m-1} + \alpha_p p' \tag{3.63}$$

The SIMPLE algorithm is listed in Algorithm 1.

Algorithm 1 SIMPLE algorithm

1: Calculation of fields at new time t_{n+1} using previous solution of u^n and p^n
2: **for** SIMPLE loop **do**
3: Solving momentum equation (Equation (3.55)) to obtain u_i^{m*}
4: Solving pressure correction equation (Equation (3.58)) for p'
5: Correct velocity to obtain u_i^m and pressure p^m
6: **end for**
7: Advance to the next time step

PISO Pressure-Implicit with Splitting of Operators (PISO) algorithm uses extra correction steps called inner correctors. In this second correction step, the velocity field correction is written similar to Equation (3.60):

$$u''_{i,P} = \tilde{u}'_{i,P} - \frac{1}{A_P^{u_i}} \left(\frac{\delta p''}{\delta x_i} \right)_P \tag{3.64}$$

In this second corrector step, the $\tilde{u}'_{i,P}$ term is not neglected but determined from Equation (3.61) using velocity field u'_i calculated in the first correction step.

The PISO is summarised in Algorithm 2.

Algorithm 2 PISO algorithm

1: Calculation of fields at new time t_{n+1} using previous solution of u^n and p^n
2: Solving momentum equation (Equation (3.55)) to obtain u_i^{m*}
3: **for** PISO loop **do**
4: Solving pressure correction equation (Equation (3.58)) for p'
5: Correct velocity to obtain u_i^m and pressure p^m
6: **end for**
7: Advance to the next time step

PIMPLE PIMPLE is a combined algorithm of PISO and SIMPLE. The number of outer correctors defines the outer iterations where the system of equations are solved until getting convergence solution. Pressure field is corrected within each iteration defined as inner correctors. PIMPLE can be seen as performing SIMPLE for each time step and usually used for transient simulation.

PIMPLE is presented in Algorithm 3.

Algorithm 3 PIMPLE algorithm

1: Calculation of fields at new time t_{n+1} using previous solution of u^n and p^n
2: **for** PIMPLE outer correctors **do**
3: Solving momentum equation (Equation (3.55)) to obtain u_i^{m*}
4: **for** PIMPLE inner correctors **do**
5: Solving pressure correction equation (Equation (3.58)) for p'
6: Correct velocity to obtain u_i^m and pressure p^m
7: **end for**
8: Solving for other fields
9: **end for**
10: Advance to the next time step

3.3.4 Linear solvers

Iterative solvers in OpenFOAM can be divided into for symmetric matrices and asymmetric matrices. The former is results from discretisation of time dependent term and diffusion term, while the latter is from discretisation of convection term. Selection of solver uses `solver` keyword with options are:

- `diagonal`: diagonal solver for both symmetric and asymmetric matrices using for explicit systems.
- `smoothSolver`: solver that uses a smoother for symmetric and asymmetric matrices with a run-time selected smoother.
- `PCG`: preconditioned conjugate gradient for symmetric matrices.
- `PBiCG/PBiCGStab`: (Stabilised) preconditioned bi-conjugate gradient with run-time selectable preconditioner for asymmetric matrices.
- `GAMG`: generalised geometric-algebraic multi-grid solver.

Solution tolerances Iterative solver reduces equation residuals after each iteration. Residuals are normalized to represent error of the solution. In OpenFOAM, three parameters are used to terminate iterations in each time step. `tolerance` represents a solver tolerance value to stop the solver when the residual reaches the value. `relTol` is a relative tolerance defined as the ratio of current over initial residuals. `maxIter` defines the maximum number of iterations. Equations can be solved several times within a time step. In this case, different setting of solutions tolerances can be used, i.e. set solver tolerance just for the last iteration, and relative tolerances for others.

Preconditioner For conjugate gradient solvers, preconditioning options can be divided into those used for symmetric matrix: such as diagonal incomplete-Cholesky (`DIC`), Faster Diagonal-based Incomplete Cholesky (`FDIC`); for asymmetric matrix: such as diagonal incomplete-LU (`DILU`) or for both diagonal preconditioning (`diagonal`); Geometric agglomerated Algebraic MultiGrid (`GAMG`).

Algorithm 4 Conjugate gradient method

1: Initialize iteration index: $k = 0$, initial solution: $\phi^0 = \phi_{in}$, initial residual: $\boldsymbol{\rho}^0 = \boldsymbol{Q} - A\phi_{in}$, initial direction: $\boldsymbol{p}^0 = \mathbf{0}$, $s_0 = 10^{30}$
2: $k = k + 1$
3: Solving: $M\boldsymbol{z}^k = \boldsymbol{\rho}^{k-1}$
4: **for** Construct new solution, residual, search direction: **do**
5: $\quad s^k = \boldsymbol{\rho}^{k-1} \cdot \boldsymbol{z}^{\boldsymbol{k}}$
6: $\quad \beta^k = s^k / s^{k-1}$
7: $\quad \boldsymbol{p}^k = \beta^k \boldsymbol{p}^{k-1}$
8: $\quad \alpha^k = s^k / (\boldsymbol{p}^k \cdot A\boldsymbol{p}^k)$
9: $\quad \phi^k = \phi^{k-1} + \alpha^k \boldsymbol{p}^k$
10: $\quad \rho^k = \rho^{k-1} - \alpha^k A\boldsymbol{p}^k$
11: **end for**
12: Repeat until desired residual is reached

Generalised Geometric-Algebraic Multi-Grid solver (GAMG) Multi-grid solvers use a fast solution from coarse grid to eliminate high frequency errors and use this for finer grid. `faceAreaPair` is a agglomeration algorithm used in OpenFOAM for coarsening the mesh. A simple two-grid iteration method algorithm can be presented in Algorithm 5.

Algorithm 5 Two-grid iteration method

1: Perform iterations on the fine grid
2: Compute residual on fine grid
3: Restrict residual to the coarse grid
4: Perform iterations of correction equation on the coarse grid
5: Interpolate correction to the fine grid
6: Update correction on the fine grid
7: Repeat until desired residual is reached

3.3.5 Boundary conditions

Moukalled et al. [72] highlighted the differences between physical conditions, e.g. "wall", "inlet", "outlet"; geometric constraints, e.g. "symmetry", "periodic"; and boundary conditions. For each physical condition, many derived types of boundary conditions can be imposed. Geometric constrains are applied to reduce the domain size and usually relate to a specific boundary conditions, such as "symmetry" condition implies zero normal flux along the specified boundary. The boundary conditions are used to define boundary field values which can be a prescribed fixed value or a relation with internal field value. Boundary conditions can be classified into three types: "Dirichlet condition", where value of variables is defined; "von Neumann condition", where flux of variables is defined and "Robin condition", where variables and flux are derived from constitutive equations.

In OpenFOAM, boundary of the domain are defined by a set of patches. Each patch consists of cell faces. Therefore, accessing boundary field values are not done directly like in case of internal field but through accessing the patch. Geometrical field concept is used to map the field values to points in the mesh. The concept is implemented in template class `GeometricField`, resulting different models such as `volScalarField` and `volVectorField` which store scalar and vector values in the cell centre; `surfaceScalarField` which stores scalar values in the face centre. `GeometricField` class provide methods to implement the boundary conditions.

Dirichlet and von Neumann are the standard boundary condition types for all fields in OpenFOAM. Besides, derived boundary conditions can be used to set boundary conditions for fields from a derived field, e.g. `fixedShearStress` is used to set a constant shear stress τ_0 for velocity field U as:

$$\tau_0 = -\nu_{eff}\frac{dU}{dn} \tag{3.65}$$

where ν_{eff} is kinematic viscosity and n denotes the surface normal.

3.3.6 Thermophysical models

Energy, heat and transport properties are determined by a set of thermophysical models [23] in OpenFOAM. This set is defined in a dictionary, call `thermoType`. An example of `thermoType` dictionary which defines mixture type, transport and thermodynamic properties models, choice of energy equation variable and equation of states is presented in Listing 3.1.

Listing 3.1 ThermoType dictionary

```
ThermoType
{
    type            heRhoThermo;
    mixture         reactingMixture;
    transport       sutherland;
    thermo          janaf;
    energy          sensibleEnthalpy;
    equationOfState perfectGas;
    specie          specie;
}
```

The fluid in a simulation can be classified as single composition and mixture of fixed or variable compositions. Two choices of energy equation variables are either internal energy or enthalpy. Transport and thermodynamic properties are determined using models based on the compressibility $\psi = (RT)^{-1}$ (R is universal gas constant) or the density ρ, which are calculated from pressure and temperature fields.

Transport models are used to calculate transport variables such as dynamic viscosity μ, thermal conductivity κ and thermal diffusivity α. `const` model assumes constant transport properties. `sutherland` model uses Sutherland's formula to define transport properties as a function of temperature:

$$\mu = A_s \frac{\sqrt{T}}{1 + T_s/T} \tag{3.66}$$

where A_s is Sutherland coefficient and T_s is Sutherland temperature. A polynomial function of order N can be used to relate transport properties with temperature field (`polynomial` model):

$$\mu = \sum_{i}^{N-1} a_i T^i \tag{3.67}$$

where a_i is a coefficient of the polynomial.

The thermodynamic models are used to calculate the specific heat c_p (at constant pressure) of the fluid. c_p can be assumed to take a constant value using `hConst` model. `hPolynomial` model uses a Nth polynomial function of temperature to define values of c_p as:

$$c_p = \sum_{i}^{N-1} a_i T^i \tag{3.68}$$

where a_i is a coefficient of the polynomial.

Equation of state is used to derive density field in OpenFOAM. `perfectGas` model uses ideal gas law to relate fluid density ρ with its pressure and temperature: $\rho = p/(RT)$. Other option is `icoPolynomial`, which define density as a Nth polynomial function of temperature:

$$\rho = \sum_{i}^{N-1} a_i T^i \tag{3.69}$$

where a_i is a coefficient of the polynomial.

3.4 OpenFOAM solvers

3.4.1 `ablBuoyantSimpleFoam`

A solver, `ablBuoyantSimpleFoam` is developed from `buoyantSimpleFoam` and `rhoReactingBuoyantFoam` for steady state simulation of ABL. While the solution of steady state simulation of ABL is used as initial condition for simulation gas dispersion. `ablBuoyantSimpleFoam` is compatible with `gasDispersionBuoyantFoam` which is used

to simulate transient gas dispersion in ABL. Thermalphysical model is a density based thermodynamics package with non-reacting mixture of air and gas. Enthalpy h is chosen as energy conservation variable. Buoyancy is taken into account by solving `p_rgh` field defined in Equation (3.4) instead of `p`. Velocity-pressure coupling is solved using *SIMPLE* algorithm. The code is listed in Appendix A

3.4.2 `gasDispersionBuoyantFoam`

`gasDispersionBuoyantFoam` is developed from `rhoReactingBuoyantFoam`. The specie transport equation is modified to apply user-defined turbulent Schmidt number. Taking into account buoyancy effect, `p_rgh` field defined in Equation (3.4) is solved instead of `p`. *PIMPLE* algorithm is used to solve velocity-pressure coupling.

The code is listed in Appendix A.

Chapter 4

Modelling of Atmospheric Boundary Layer

Solving Atmospheric Boundary Layer (ABL) flow prior to a release of gas source is the initial step of a gas dispersion simulation. In this Chapter, the `ablBuoyantSimpleFoam` solver (see Section 3.4) is used to validate its capacity in simulating the horizontal homogeneous ABL flow under neutral stability condition.

4.1 Simulation of ABL over flat terrain

4.1.1 Domain and mesh

In case of a dense gas dispersion, the gas cloud is accumulated near ground. Even for a dense cold LNG vapour that the vapour cloud becomes buoyant when heated by the surrounding, reported results showed that the cloud of interest concentration only reached the height of $8m$ [7] and the gas cloud hardly reached the furthest gas sensors array at $800m$. Therefore, a computational domain of the length D = 1000 m and height H = 50 m is selected for simulations of neutral ABL over flat terrain. The computational domain definition is

sketched in Figure 4.1. The domain boundaries are named as `inlet`, `outlet`, `top` and `ground`. Sampling locations where flow fields are collected for post-processing analysis are also shown in Figure 4.1.

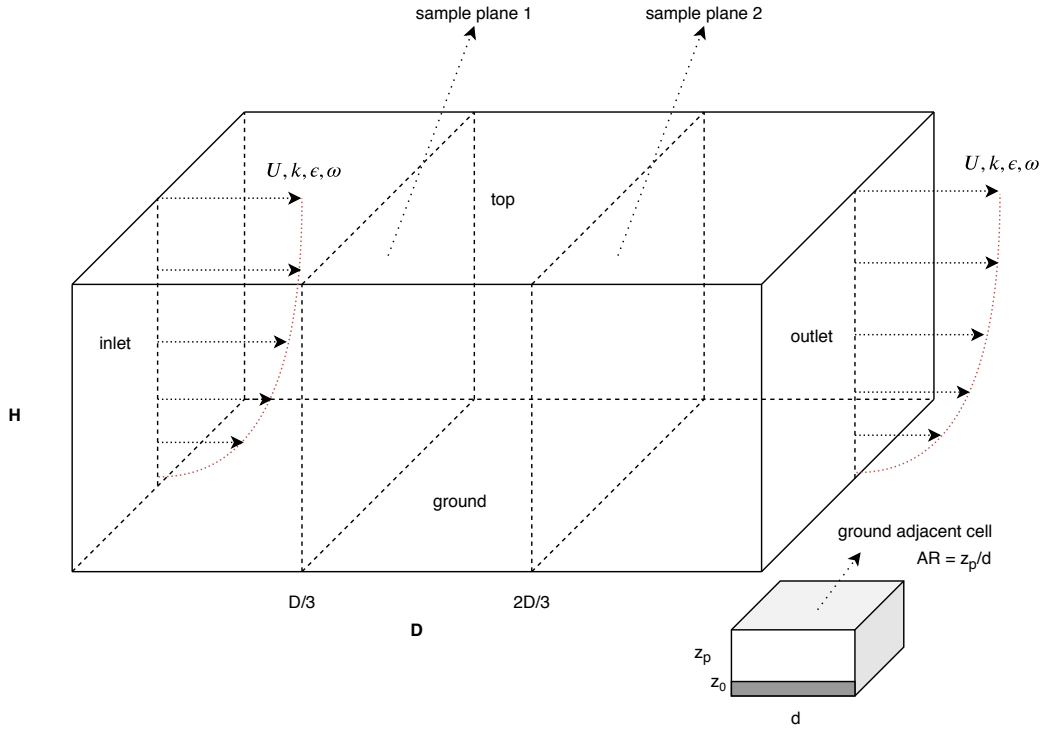


Fig. 4.1 Domain and mesh definition for simulation of ABL over flat terrain

The mesh is stretched in vertical direction in order to capture the flow near ground. Expansion ratio, which is the ratio between succeed cell and current cell in expansion direction, is chosen the value of 1.1. The mesh is uniform in horizontal direction for flat terrain cases. Constant mesh cell size in this direction is defined as the cell size of the mesh d. Ground adjacent cell aspect ratio Ar is defined as the ratio between the height of ground adjacent cell h and cell size of the mesh d (illustrated also in Figure 4.1). These mesh parameters are subjected to a mesh sensitivity test in Section 4.2.

4.1.2 Boundary conditions

Boundary conditions play an important role in the accuracy of simulations. The boundary conditions of all cases are represented in Table 4.1. The names of computational boundaries are defined in Figure 4.1.

The profiles of variables according to MOST (Monin-Obukhov Similarity Theory) are used at `inlet` patch. While flow at `outlet` patch is fully developed, `zeroGradient` condition is used for all variables there.

A constant value of pressure is set at `top` boundary to establish a homogeneous hydrostatic pressure field throughout the domain. There are two approaches to set boundary conditions for u, ε and ω: as a fixed value or fixed gradient according to inlet profiles. The fixed values approach is to present the MOST profile in this patch, while fixed gradients approach can simulate a constant flux assumption for all variables of MOST. A study on the effectiveness of these two approaches is presented in Section 4.3. According to MOST, profile of k is constant; therefore, `zeroGradient` condition is used.

Lastly, wall functions are adopted for all variables in `ground` boundary but the OpenFOAM `noSlip` condition is used for velocity.

Table 4.1 Boundary conditions for neutral ABL simulation

`inlet`	profiles of k, u, ε, ω	Eq. (2.2), (2.10)
`outlet`	`zeroGradient` for all variables	
`top`	`fixedValue` for static pressure	
	`zeroGradient` for k	
	`fixedGradient/fixedValue` for u, ε and ω	Eq. (2.9)
`ground`	`noSlip` for u	
	`nutkWallFunction` for ν_t	
	`epsilonWallFunction` for ε	
	`kqRWallFunction` for k	
	`omegaWallFunction` for ω	

The ABL parameters used to define inlet variable profiles are listed in Table 4.2 according to the reference case by Hargreaves and Wright [31].

Table 4.2 ABL parameters using for neutral ABL simulation

u_* (m/s)	z_0 (m)	u_{ref} (m/s)	z_{ref} (m)
0.625	0.01	10	6

4.1.3 Numerical setting

A steady state simulation is employed using the `ablBuoyantSimpleFoam` solver described in previous chapter. The OpenFOAM discretisation schemes, velocity-pressure coupling algorithm as well as linear solvers are listed below:

- Time schemes: `steadyState`
- Gradient schemes: `Gauss linear`
- Divergence schemes: `Gauss limitedLinear 1`
- Surface normal gradient schemes `corrected`
- Laplacian schemes: `Gauss linear corrected`
- Interpolation schemes: `linear`
- Solving algorithm: `SIMPLE`
- Linear solver for p: `GAMG` with `DICGaussSeidel` preconditioner
- Linear solver for U, h, k, epsilon: `PBiCGStab` with `DILU` preconditioner

Residual controls are set at three order of magnitude for pressure and four order of magnitude for other variables such as U, k, ε, ω and enthalpy h.

4.2 Mesh sensitivity study

4.2.1 Design of experiment

A numerical experiment is used to verify the dependency of mesh parameters in the performance of the solver. Since homogeneous ABL is simulated, the exact solutions is the inlet profiles of numerical variables such as velocity, turbulence kinetic energy, turbulence dissipation rate (in case $k-\varepsilon$ is used as turbulent model). Therefore, the difference between a vertical profile and the inlet profile can be used as a parameter to present the numerical error. In this study, the integration of this difference over vertical direction is used as an index to quantify the solver performance. This integration of a variable can be defined as:

$$\int (f - f_{inlet}) dz \tag{4.1}$$

where f and f_{inlet} are values of a variable at sampling location and at inlet respectively. The integral is in vertical direction z. Three sampling locations used in this study are `outlet` and two sampling planes in the middle of the domain (can be shown in Figure 4.1 and parameterised in Table 4.3).

Different mesh parameters such as mesh size and aspect ratio are examined. To show the effectiveness of modified $k-\varepsilon$ over the standard $k-\varepsilon$ model in simulation of ABL flow, standard $k-\varepsilon$ model is only used in one set of mesh parameters. The list of parameter variations is listed in Table 4.3. Combining between these parameters results in totally 9 cases. Each of parameters has abbreviation which can be combined to form the case. For example, the name `Mesh1Ar1` composed of `Mesh1` parameter and `Ar1` parameter meaning the these parameters are used in the case. This name convention is used in analysing the simulation results.

Table 4.3 Parameters and sampling positions for Mesh sensitivity study

Parameters	Number of variations	Values	Abbreviations
Mesh size	3	2	Mesh1
		1	Mesh2
		0.5	Mesh3
Aspect ratio	2	20	Ar1
		40	Ar2
Turbulence model	1	Modified $k-\varepsilon$	Mke
Sampling position	3	D/3	Mid1
		2D/3	Mid2
		D	Back

4.2.2 Results and discussion

Important variables in the simulation are velocity and turbulent properties such as turbulent kinetic energy k and turbulent dissipation rate ε. The difference of profiles between a sampling location and `inlet` patch is used as the performance quantitative measure. The integration of this difference along vertical axis for each variable is reported in Table 4.4 for velocity, Table 4.5 for k and Table 4.6 for ε. Overall, it can be seen that sampling plane further from `inlet` boundary results in higher value of integral, i.e. value at `Mid1` plane is smaller than `Mid2` plane and `Back` plane has largest value. This means that `inlet` profiles are degraded further downstream in the domain. However, the level of change is small, e.g. the largest difference for velocity occurred in the ground adjacent cell (see Figure 4.2) and the magnitude of change is less than 0.15 (m/s) (only 2.5% the inlet value).

For the result of velocity (Table 4.4), the performance is worse when increasing number of mesh cells in `Back` sampling plane. Further analysis is required to understand the cause of this abnormal. Figure 4.2 shows velocity profile difference between `Back` and `inlet` boundary. It can be seen that most errors are at the near ground cells. `Mesh1` has largest error at ground adjacent but overall error is less than other two meshes. The contribution of these errors is from numerical approximation and as mentioned before that these error are

negligible comparing with the velocity magnitude. In all cases, the performance of `Mesh1Ar1` is the best with least computational time.

Table 4.4 Mesh sensitivity result for velocity field U

Cases	Mesh1Ar1	Mesh2Ar1	Mesh3Ar1	Mesh1Ar2	Mesh2Ar2	Mesh3Ar2
Mid1	0.293	0.251	0.289	0.245	0.28	0.28
Mid2	0.369	0.415	0.55	0.369	0.513	0.465
Back	0.374	0.486	0.672	0.409	0.625	0.516

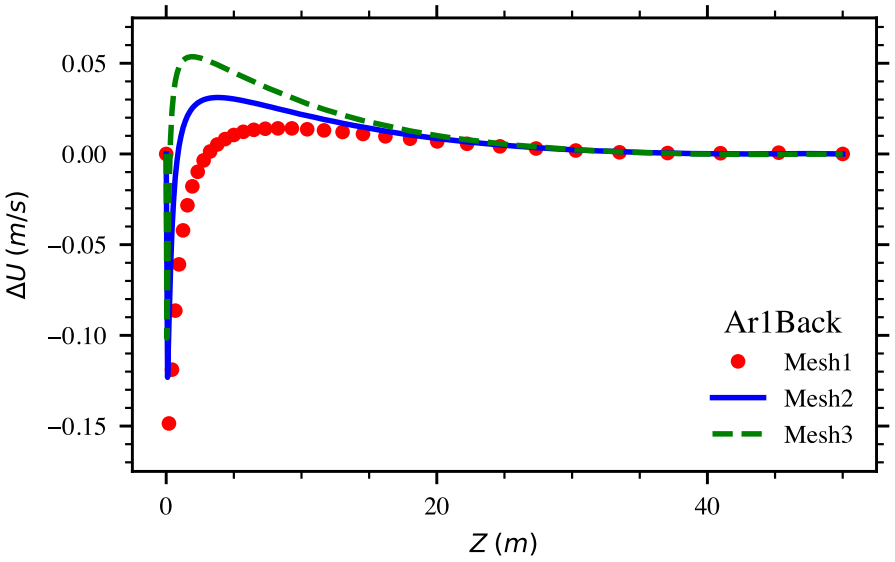


Fig. 4.2 Velocity difference profiles at `Back` location of different meshes

In contrast to velocity, the result for k shows that the mesh refinement does improve the performance metric. Increasing the number of mesh cells helps to enhance the performance metric, i.e. decreasing the integral. Additionally, the performance for aspect ratio `Ar2` is better than `Ar1`. Therefore, it can be concluded that the refinement in ground region help to improve the k results. This can be seen in Figure 4.3 where profiles difference of k with three meshes `Mesh1`, `Mesh2` and `Mesh3` are plotted.

The result for ε in Table 4.6 shows that the mesh parameters do not affect much the performance metric. For example, `Mesh1` `Mesh2` and `Mesh3` have number of cells four times difference but the performance measure values are not different much.

Table 4.5 Mesh sensitivity result for turbulent kinetic energy field k

Cases	Mesh1Ar1	Mesh2Ar1	Mesh3Ar1	Mesh1Ar2	Mesh2Ar2	Mesh3Ar2
Mid1	0.218	0.175	0.161	0.197	0.164	0.132
Mid2	0.422	0.31	0.296	0.389	0.302	0.201
Back	0.564	0.403	0.402	0.521	0.397	0.238

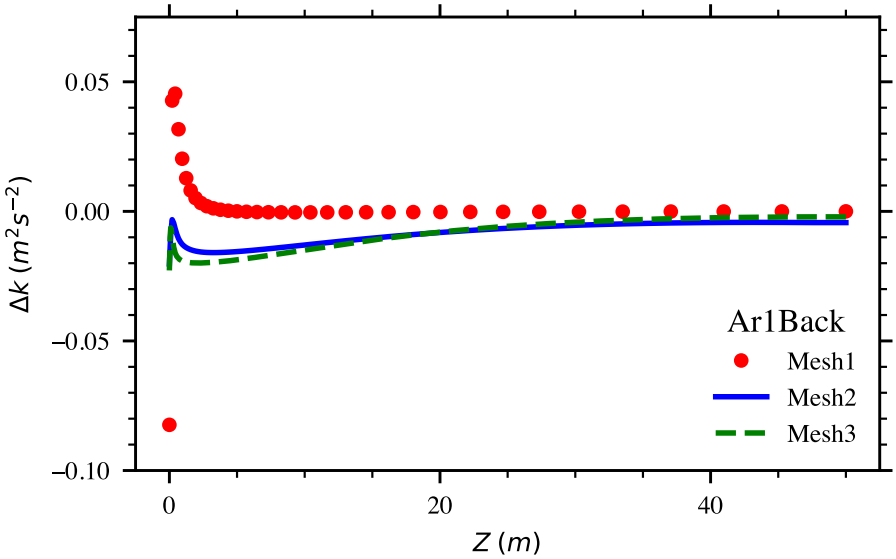


Fig. 4.3 k difference profiles at `Back` location of different meshes

Table 4.6 Mesh sensitivity result for turbulent dissipation rate field ε

Cases	Mesh1Ar1	Mesh2Ar1	Mesh3Ar1	Mesh1Ar2	Mesh2Ar2	Mesh3Ar2
Mid1	0.05	0.049	0.055	0.044	0.047	0.056
Mid2	0.052	0.053	0.061	0.048	0.053	0.058
Back	0.053	0.054	0.063	0.05	0.055	0.059

From results of the mesh dependent experiment, it can be seen that the mesh parameters do not affect much the solver performance in prediction of ε profile, and the mesh refinement helps to improve the k prediction but not the U prediction. However, the level of difference is negligible compare to the variable magnitude in all cases. To balance the solver performance in all variables. The parameters of `Mesh2` are used for simulations in the following sections.

4.3 Turbulence models study

4.3.1 Design of experiment

For further understanding of the effect turbulence models in modelling HHTSL, three turbulent models are subjected to a numerical experiment including the modified $k-\varepsilon$, the modified $k-\varepsilon$ with variable C_μ and the SST $k-\omega$ turbulence models. The $k-\varepsilon$ with variable C_μ model is used to simulate different level of inlet kinetic energy k by altering C_μ according to Equation (2.2). The value of $C_\mu = 0.017$ is altered from default value of $C_\mu = 0.09$ and compared with MOST. The inclusion of source term by Pontiggia et al. [29] is implemented using Equation (2.4).

Two approaches for `top` boundary conditions are also examined including `fixedValue` and `fixedGradient` for u, ε and ω. These parameters are subjected to a numerical experiment and are summarised in Table 4.7.

Table 4.7 Turbulence models setting for neutral ABL simulation

Top boundary condition	`fixedValue` for u, ε and ω	FV
	`fixedGradient` for u, ε and ω	FG
Turbulence model	`modified` $k-\varepsilon$(Eq. (2.3))	Mke
	`modified` $k-\varepsilon$ `with variable` C_μ	MkeVar
	$k-\omega$ `SST`	kOme

4.3.2 Results and discussion

Tables 4.8, 4.9 and 4.10 are results of U, k and ε profiles difference integration at sampling positions. The same trend of increasing values when moving further from `inlet` boundary as in previous sections can be shown. Overall, the values of fixed value cases (denoted by `Fv`) are smaller than fixed gradient cases (denoted by `Fg`). The values of $k-\omega$ and modified $k-\varepsilon$ cases are comparable, while variable c_μ cases have the worst performance.

The velocity result (Table 4.8) shares the general trend discussed above where the performance of Fv cases is better than Fg cases, i.e. FvMke is better than FgMke and FvKome is better than FgKome. We can see this in Figure 4.4, which illustrated the velocity difference profiles between Back and inlet patches. Both FvKome and FgKome have similar performance near ground region, but FvKome is better than FgKome in higher locations. This observation is the same for FvMke and FgMke. Comparing between turbulent models, $k-\varepsilon$ (FgMke) has better performance than SST $k-\omega$ (Kome). From this Figure, we can also see that the velocity at Back is over-estimated comparing with inlet patch ($\Delta U = U_{back} - U_{inlet} > 0$) except in near ground region where it is under-estimated. However the over-estimation or under-estimation is negligible comparing to the magnitude of inlet velocity.

Table 4.8 Turbulence models result for velocity

	FgKome	FgMke	FgMkeVar	FvKome	FvMke	FvMkeVar
Mid1	1.2	1.16	4.35	1.14	1.14	4.08
Mid2	2.32	2.24	8.25	1.93	2.07	7.24
Back	2.93	2.93	10.27	2.16	2.41	8.49

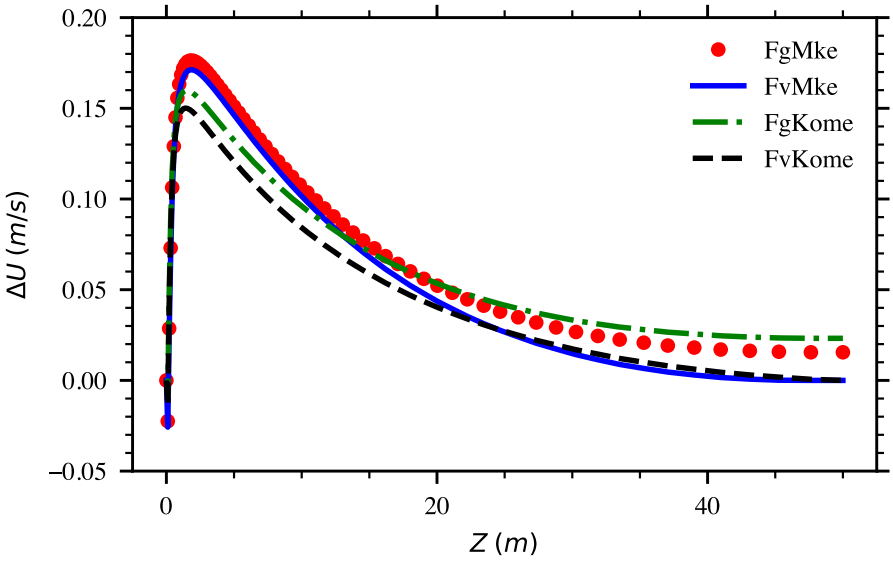


Fig. 4.4 U difference profiles at Back location of $k-\varepsilon$ (Mke) and SST $k-\omega$ (Kome) turbulence models with different top boundary conditions fixed value (Fv) and fixed gradient (Fg).

On the other hand, from the k result in Table 4.9, we can see that performance of `Fg` cases is better than `Fv` cases. Comparing between turbulent models, `FgMke` is better than `FgKome`. Figure 4.5 presents these points. It can also be shown in this Figure that all values of k at `Back` is under-estimated the `inlet` values.

Table 4.9 Turbulence models result for k

	FgKome	FgMke	FgMkeVar	FvKome	FvMke	FvMkeVar
Mid1	1.07	0.79	4.96	1.07	0.77	5
Mid2	1.87	1.42	9.25	2	1.5	9.77
Back	2.2	1.63	11.59	2.55	2.11	12.9

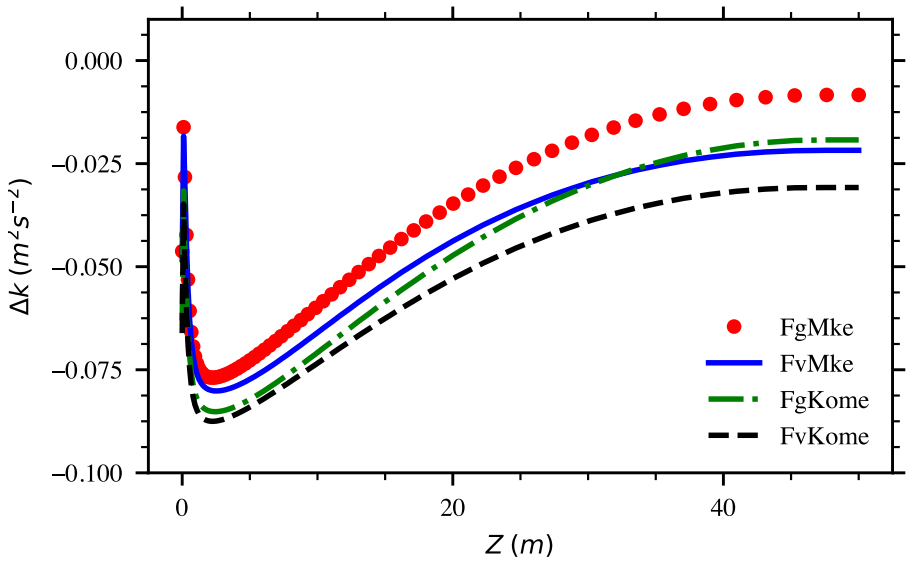


Fig. 4.5 k difference profiles at `Back` location of $k-\varepsilon$ (`Mke`) and SST $k-\omega$ (`Kome`) turbulence models with different `top` boundary conditions fixed value (`Fv`) and fixed gradient (`Fg`).

The result for ε in Table 4.10 shows that modified $k-\varepsilon$ gives smaller value to $k-\omega$. One possible reason for this is due to the magnitude of ω is larger than ε. Therefore, the integral of ω profile is larger than ε. One more observation is that the ε difference profile does not changed much in different sampling locations. Figure 4.6 illustrates this point. It can also be shown that ε values are matched with `inlet` value in most of points excepts in near ground points.

Table 4.10 Turbulence models result for ε/ω

	FgKome	FgMke	FgMkeVar	FvKome	FvMke	FvMkeVar
Mid1	0.93	0.18	0.79	0.92	0.18	0.79
Mid2	1.01	0.21	0.86	1	0.2	0.87
Back	1.04	0.21	0.89	1.01	0.21	0.9

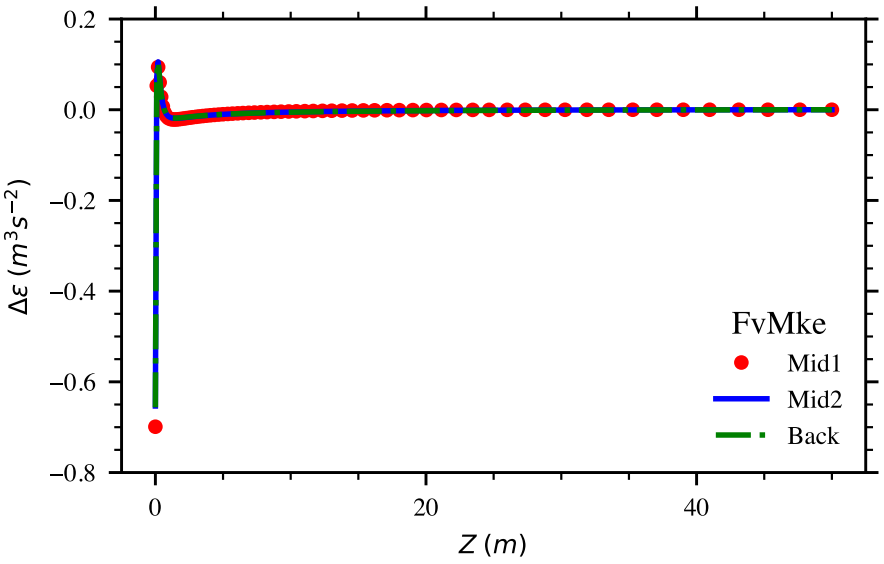


Fig. 4.6 ε difference profiles at different sampling locations of Modified $k-\varepsilon$ and fixed value `top` boundary conditions fixed value (`Fv`) and fixed gradient (`Fg`).

As mentioned in the beginning, variable c_μ cases have the worst performance in Tables 4.8, 4.9 and 4.10. This was due to wall function, where wall treatment used for default $C_\mu = 0.09$ is implemented. The consistency of C_μ value between wall functions helps to improve the performance of developed solver. The result from modelling different turbulence kinetic energy by varying C_μ is presented in Figure 4.7, $C_\mu = 0.09$ and $C_\mu = 0.017$ are implemented in `FvMke` and `FvMkeVar` cases respectively. The profiles of velocity and dissipation rate ε are perfectly matched with Monin-Obukhov profiles. In $C_\mu = 0.017$ simulation, the kinetic energy k near ground is smaller than theory value. However, the values of k are matched with theory values at higher height. Overall, results are accepted for verifying the proposed model in simulating different levels of turbulent kinetic energy.

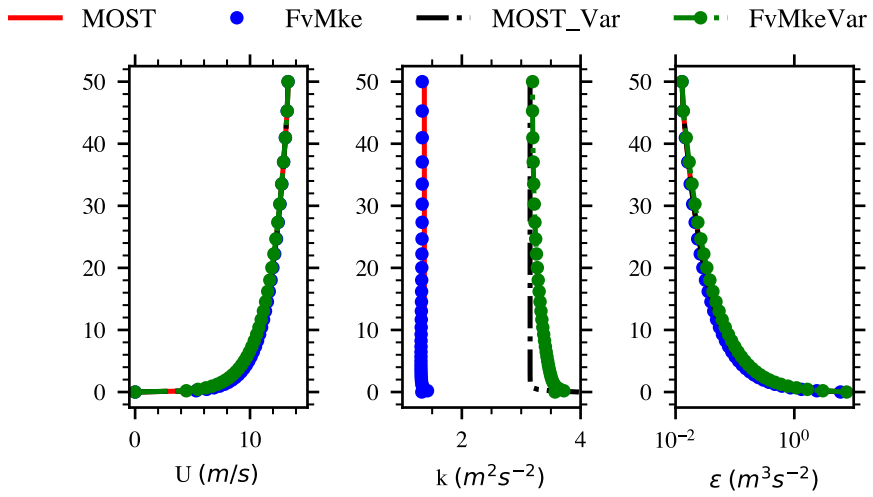


Fig. 4.7 Comparison of velocity, k and ε profiles from MOST (MOST, MOST_Var) and simulations of different kinetic energy levels by varying $C_\mu = 0.09$ (FvMke) and $C_\mu = 0.017$ (FvMkeVar).

In conclusion, the modifications of constants of $k-\varepsilon$ and SST $k-\omega$ models according to Equation (2.3) achieve matched results with MOST. Using Equation (2.3), we can also model different levels of turbulent kinetic energy by varying C_μ constant. The performance of $k-\varepsilon$ and SST $k-\omega$ models using fixed value or fixed gradient top boundary conditions is different in predicting velocity and k but ε and ω predictions are not affected. However, the difference is negligible and both approaches are verified achieving HHTSL.

A variant of $k-\varepsilon$ turbulent model, buoyant $k-\varepsilon$, is used for an extended simulation. A fixed value `top` boundary condition approach is used. Result of velocity profiles from this simulation, as well as $k-\varepsilon$ and SST $k-\omega$ simulations are presented in Figure 4.8. We can see that the developed model is also able to simulate HHTSL using buoyant $k-\varepsilon$ with acceptable degree of difference as $k-\varepsilon$ and SST $k-\omega$ turbulent models.

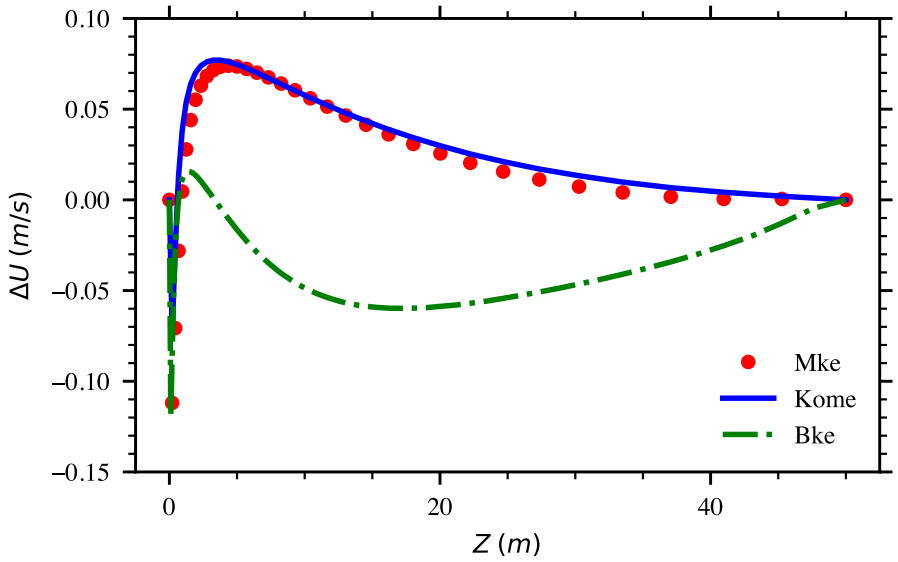


Fig. 4.8 Comparison of velocity different profiles from $k-\varepsilon$, SST $k-\omega$ and buoyant $k-\varepsilon$ turbulent models.

4.4 Roughness length study

4.4.1 Design of experiment

The ground parameters are tested by varying wall adjacent cell size and the roughness length z_0. The roughness length z_0 appeared in logarithmic velocity profile (Equation 2.2) is defined as the height where wind velocity is zero. Typically, z_0 is loosely related to the height of roughness elements such as water, grass, tree, building. Higher roughness element size usually results in higher value of z_0. Three different values of z_0 are selected where $z_0 = 0.001$ represents the flat surface such as dessert, $z_0 = 0.01$ represents the surface with grass and $z_0 = 0.1$ is typically for surface with few trees or many hedges [73]. All cases are summarised in Table 4.11. The ground adjacent cell size is set by varying the aspect ratio `Ar`, the mesh cell size of all meshes is fixed at $2m$.

Table 4.11 Parameters for boundary conditions and roughness length study

Parameters	Number of variations	Values	Abbreviations
Ground adjacent cell size	3	0.1	CS1
		0.05	CS2
		0.025	CS3
Roughness length	3	0.001	Z01
		0.01	Z02
		0.1	Z03

4.4.2 Results and discussion

Similar to previous studies, the performance measure is an integration of profile different between a location and `inlet` boundary. Tables 4.12, 4.13 and 4.14 summarise results of this performance measure for velocity, k and ε respectively at three positions `Mid1`, `Mid2` and `back`. In all cases, `CS1` value results better performance than `CS2` and `CS3`. This means that refinement near ground region does not help to improve the solver performance and even worsen it in high roughness length cases such as `Z03` .

Table 4.12 Turbulence models result for velocity

U	Mid1	Mid2	Back
CS1Z01	0.24	0.37	0.41
CS1Z02	1.09	1.93	2.22
CS1Z03	11.02	17.64	18.94
CS2Z01	0.26	0.46	0.54
CS2Z02	1.74	3.14	3.7
CS2Z03	14.47	23.54	25.63
CS3Z01	0.35	0.63	0.76
CS3Z02	2.64	4.84	5.79
CS3Z03	17.61	29	31.92

Results for k in Table 4.13 and ε in Table 4.14 show that with the small value of roughness length `Z01` ($z_0 = 0.001$), varying adjacent ground cell size from $CS1 = 0.1$ to $CS3 = 0.025$ does not affect the solver performance, e.g. `CS1Z01`, `CS2Z01` and `CS3Z01` have the same

Table 4.13 Turbulence models result for k

k	Mid1	Mid2	Back
CS1Z01	0.2	0.39	0.52
CS1Z02	0.83	1.7	2.37
CS1Z03	8.49	19.27	28.93
CS2Z01	0.19	0.38	0.51
CS2Z02	1.01	2.11	2.98
CS2Z03	10.43	23.25	35.03
CS3Z01	0.19	0.4	0.55
CS3Z02	1.33	2.8	4.01
CS3Z03	12.08	26.91	40.6

Table 4.14 Turbulence models result for ε

epsilon	Mid1	Mid2	Back
CS1Z01	0.04	0.05	0.05
CS1Z02	0.15	0.17	0.19
CS1Z03	2.81	3.2	3.36
CS2Z01	0.04	0.05	0.05
CS2Z02	0.27	0.32	0.34
CS2Z03	3.95	4.48	4.68
CS3Z01	0.04	0.05	0.05
CS3Z02	0.51	0.59	0.62
CS3Z03	5.42	6.13	6.35

values of performance measure. Figure 4.9 illustrates this point by plotting $\Delta\varepsilon$ profiles of different ground cell sizes `CS1`, `CS2` and `CS3`.

However, for larger value of roughness length `Z02` ($z_0 = 0.01$) and `Z03` ($z_0 = 0.1$), the adjacent ground cell size has significant effect on the prediction. It is also shown in these Tables that the performance metric varies significantly according to values of roughness length. Higher values of roughness length result higher values of performance metric. It shows that this set of adjacent ground cell sizes is not appropriate in modelling high roughness length cases. Figure 4.10 illustrates this point where ΔU profiles at `Back` patch with different

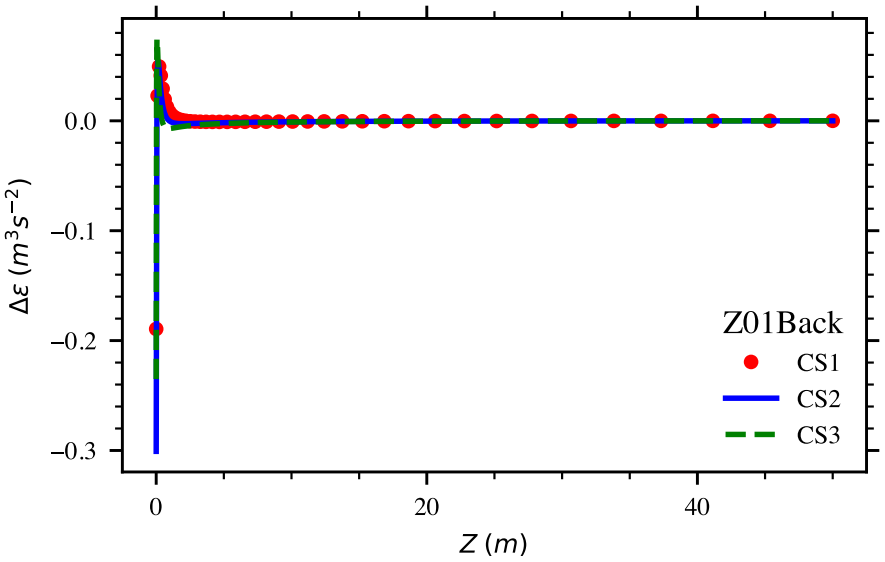


Fig. 4.9 ε difference profiles at `Back` location of different values of ground cell size $CS1 = 0.1$ to $CS3 = 0.025$. The roughness length `Z01` ($z_0 = 0.001$).

values of roughness length and the largest value of adjacent cell size `CS1` are plotted. The cell size `CS1` in these simulations is equal the roughness length of `Z03`.

In Figure 4.10, when the cell size `CS1` equal the roughness length `Z03`, it results in the worst performance case `CS1Z03`. An extended study is conducted for high roughness length value case `Z03` where $z_0 = 0.1$ to find the optimum value of adjacent ground cell size. The cell size values are chosen according to the multiplication of roughness length value. Four values are $S5 = 5z_0$, $S10 = 10z_0$, $S20 = 20z_0$ and $S50 = 50z_0$. Results from these simulations and previous simulation using $S1 = z_0$ are presented in Figure 4.11. This Figure shows that `S10` is the optimum value of ground adjacent cell size. It can be concluded that in a large surface roughness case, the ground adjacent cell size should be carefully chosen. This value needs to be larger than the roughness length z_0 but large value of this can cost the accuracy of predictions.

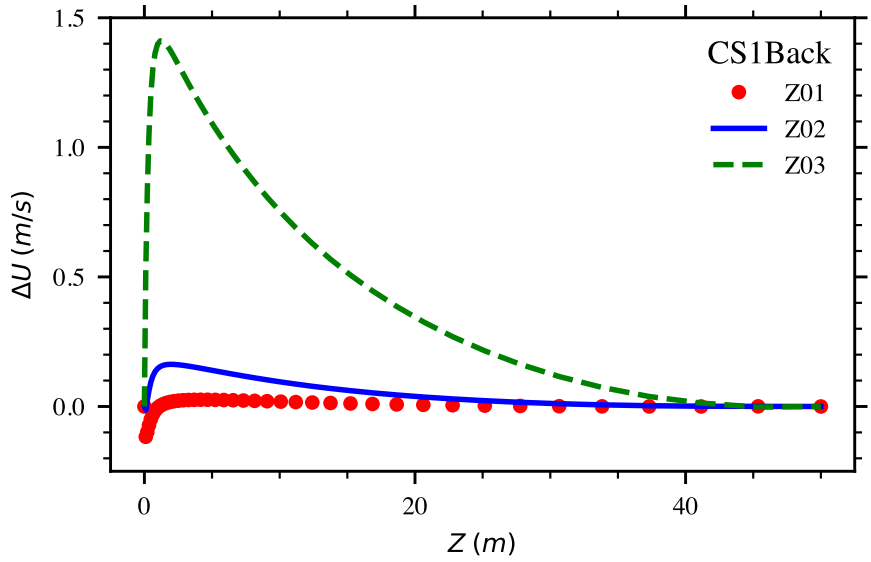


Fig. 4.10 U difference profiles at `Back` location of different values of roughness length $Z01 = 0.001$, $Z02 = 0.01$ and $Z03 = 0.1$. The adjacent cell size is $CS1 = 0.1$.

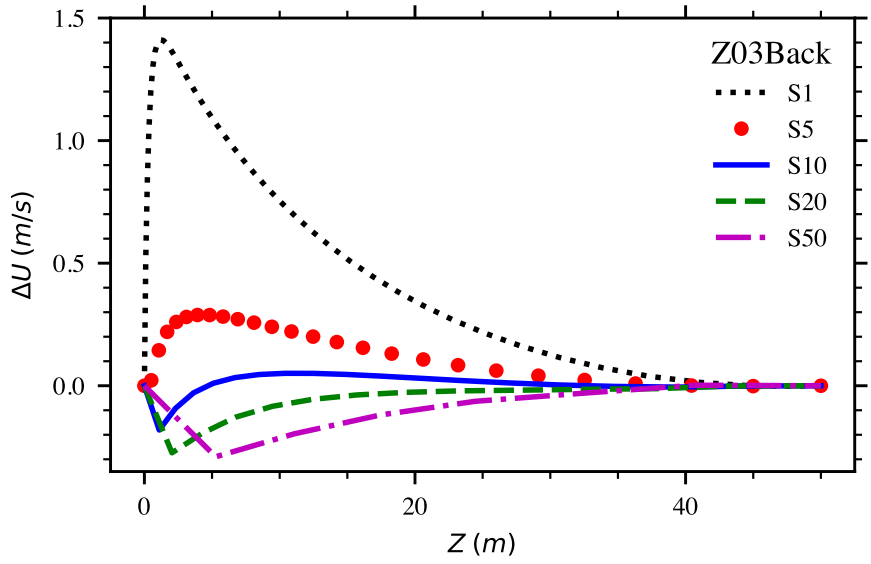


Fig. 4.11 U difference profiles at `Back` location of different values of ground adjacent cell size $S1 = z_0$, $S5 = 5z_0$, $S10 = 10z_0$, $S20 = 20z_0$ and $S50 = 50z_0$. The roughness length is $Z03 = 0.1$.

4.5 Conclusion of simulation of ABL over flat terrain

In this Chapter, different mesh parameters, turbulent models and values of roughness length are tested for simulations of homogeneous atmospheric boundary layer with MOST inlet profiles. An integral of profile difference is proposed to quantify the performance of model. It has been shown that the modification of turbulent model constants are needed to achieve homogeneous profiles from `inlet` to `back` boundary. The performance of developed solver is verified using various turbulent models such as $k-\varepsilon$, SST $k-\omega$ and buoyant $k-\varepsilon$. The modification of C_μ constant to simulate different levels of turbulence kinetic energy k is useful in some scenarios such as when the field measurements of turbulence kinetic energy are different from MOST. The selection of ground adjacent cell size is important in high roughness length simulations. This value needs to be larger than the roughness length z_0 but small enough to assure the accuracy of predictions.

Chapter 5

Atmospheric boundary layer gas dispersion

In this Chapter, gas dispersion simulations using developed model are conducted to validate this model in predicting atmospheric dense gas dispersion over unobstructed terrain. The model validation database identified in MEP (Model evaluation protocol, Section 1.4.2) is used. The unobstructed dispersion tests in the database include wind tunnel tests such as `DA0120` and `DAT223` and field tests such as `Burro3`, `Burro7`, `Burro8` and `Burro9` (test name is deliberately formatted in monospace font). Data from all these tests is considered as the benchmark data in this study.

SPMs (Statistical performance metrics, Section 1.4.1) from OpenFOAM simulations are compared with FLACS (FLame ACceleration Simulator) [52, 53], a commercial CFD software used for explosion modelling and atmospheric dispersion modelling in the field of industrial safety and risk assessment.

5.1 Dense gas dispersion in wind tunnel tests

5.1.1 Hamburg wind tunnel tests

The atmospheric boundary layer was modelled in a open-circuit wind tunnel to investigate the instantaneous and continuous dispersion of heavy gas releases. The test section of the wind tunnel has the dimensions of $1.5\,\text{m} \times 1.0\,\text{m} \times 4.0\,\text{m}$. The flow was in flat floor or disturbed by the presence of obstacles. An adjustable ceiling was utilised to establish a zero pressure gradient boundary layer. Of all tests, the `DA0120` and `DAT223` are unobstructed dispersion tests and included in MEP (Section 1.4.1). Therefore, data from these two tests is used to validate the developed model.

In `DA0120` and `DAT223` tests, continuous source of SF_6 gas was released in flat terrain without obstructions. The gas was injected from the perforated disk with diameter approximately 7 cm. Aspirated hot-wire probes were used to measure gas concentration at the ground level of various locations. Peak concentrations at these locations were reported in the database. This data is used to validate the simulation results.

Dimensional analysis was used to derive similarity laws to match small-scale wind tunnel data and full-scale data. For continuous release, the resulted length L_{cc}, time T_{cc} and velocity U_{cc} scales are calculated in Equation 5.1 [12]. These scales are used to derive the full-scale size parameters from wind-tunnel parameters. All parameters are summarised in Table 5.1. All simulations in this Section are at full-scale.

$$\begin{aligned} L_{cc} &= \left(\frac{\dot{V}_0^2}{g'}\right)^{1/5} \\ T_{cc} &= \left(\frac{\dot{V}_0}{g'^3}\right)^{1/5} \\ U_{cc} &= \left(\dot{V}_0 g'^2\right)^{1/5} \end{aligned} \tag{5.1}$$

where V_0 is the total release volume, g' is the modified gravity:

$$g' = \frac{\rho_{gas} - \rho_{air}}{\rho_{air}} g \tag{5.2}$$

ρ_{gas} and ρ_{air} are the density of the released gas and air respectively. g is the acceleration of gravity.

Table 5.1 Hamburg flat, unobstructed test case parameters

	Unit	DA0120		DAT223	
		Wind tunnel	Full scale	Wind tunnel	Full scale
L_{cc}	m	0.00718		0.01367	
T_{cc}	s	0.01333		0.01839	
Substance		SF_6		SF_6	
Density	kg/m^3	6.27		6.27	
Roughness length	m	0.0001	0.0164	0.0001	0.0164
Wind speed	m/s	0.54	6.92	0.74	9.47
Reference height	m	0.00718	1.077	0.01367	2.24
Ambient temperature	K	293		293	
Source diameter	m	0.07	11.48	0.07	11.48
Spill rate	kg/s	0.0001743	60	0.000872	300

5.1.2 Domain and computational mesh

The computational domain is of the length L = 600 m, width W = 180 m and height H = 24 m. Length of the domain is chosen according to the furthest probing point of the test. In this case, the furthest data point is located at 389.6 m. Therefore, the value of 600 m is used. The domain width and height value are scaled with the gas source diameter d. The width is 15 times and the height is twice the diameter. The boundary patches are named as `inlet`, `outlet`, `top`, `ground`, `symmetry plane`, `side` and `gas inlet` and are illustrated in Figure 5.1.

The suitable type of mesh mainly depends on the type of physics solved. For the dispersion of dense gas cloud simulation, structured mesh with hexahedral cells is proven to be more computationally effective than unstructured mesh using tetrahedral cells [74]. In

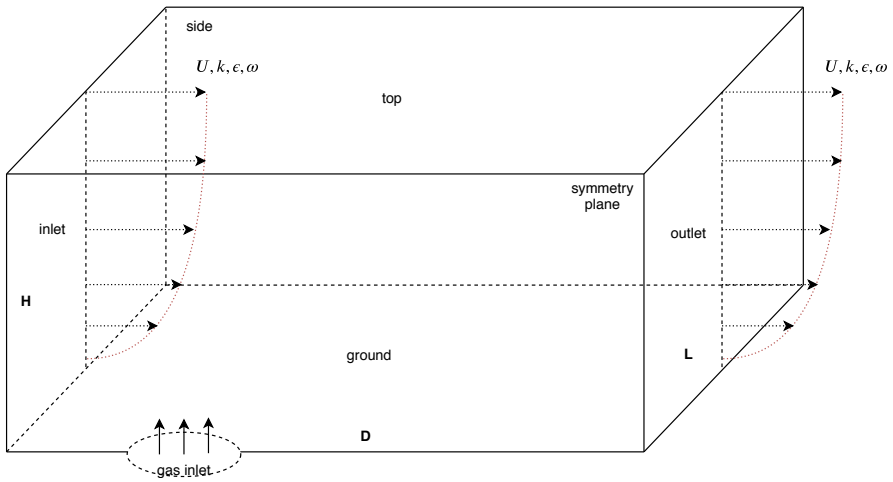


Fig. 5.1 Domain and mesh definition for simulation of gas dispersion over flat terrain

this study, hexahedral meshes generated from OpenFOAM native application `blockMesh` are used for all simulations. The mesh is refined in gas dispersion region to accurately solving the flow there. The adequate number of nodal points used for this study can be determined using the mesh independence study where the effects of mesh parameters on the solution of peak gas concentration are investigated. Detail of this mesh dependent test is presented in Section 5.2.1.

5.1.3 Boundary conditions

Atmospheric and gas inlet boundaries The atmospheric inlet profiles are specified by MOST (Monin–Obukhov similarity theory) using parameters recorded in Table 5.1.

The `flowRateInletVelocity` boundary condition is set at the `gasInlet` boundary, where the volume flow rate of the gas source SF_6 is specified. For other variables, the `zeroGradient` condition is used.

The `top`, `side` and `outlet` boundaries In OpenFOAM, the boundary condition `prghPressure` provides static pressure condition for `p_rgh` field as:

$$p\_rgh = p - \rho gh \tag{5.3}$$

Assuming static pressure is constant throughout the domain, the `prghPressure` condition is used at the `top` boundary with a constant value. At other boundaries, the `zeroGradient` condition is applied.

Velocity, k and ε at `top` boundary are set as a fixed value according to MOST. The `zeroGradient` condition is specified for all variables at `side` and `outlet` boundaries.

The `ground` boundary At the `ground` patch, the `noSlip` condition (zero fixed value) is used for velocity. Wall functions are adopted for other variables such as kinematic turbulent viscosity *nut*, k and ε. The `zeroGradient` condition is set for pressure p and temperature T.

5.1.4 Numerical setting

Firstly, the steady simulation using in previous Chapter, `ablBuoyantSimpleFoam`, is performed to establish a steady-state ABL flow prior to the release of gas source.The discretisation schemes and linear solver setting are the same as in the simulation of neutral ABL (Section 4.1.3).

The transient simulation is then performed using steady simulation solutions as initial fields. The solver `gasDispersionBuoyantFoam` is studied to model multi-species flow where the mixture considered is between air and dense gas SF_6 and take into account buoyancy effect to model the density stratification in dense gas flow. The wind tunnel tests were conducted in isothermal condition. Therefore, constant thermal and transport properties are used for both gases. Residual controls are set as 10^{-3} for pressure velocity and 10^{-4} for other variables such as k, ε, species mass fraction Y_i and enthalpy h. The same discretisation schemes for variables and terms listed in Section 4.1.3 are re-used in this simulation.

5.2 Simulations of dense gas dispersion

5.2.1 Mesh sensitivity study

Two mesh parameters used for the mesh sensitivity study are mesh cell size and wall adjacent cell aspect ratio. The variations of these two parameters are summarised in Table 5.2. The mesh is refined in two horizontal directions, while the mesh size in vertical direction is remained the same. The coarsest mesh is refined in stream-wise and cross stream-wise directions twice. This results in three meshes with number of cells varied in four times to each other. The mesh refinement in vertical direction can be controlled by the aspect ratio. Two wall adjacent cell aspect ratios are used with two times difference to each other. These results in totally six meshes. For the first aspect ratio, the number of mesh cells are ranging from 100000 for the coarsest mesh, 400000 for intermediate mesh and 1.6 million for the finest mesh. For the second aspect ratio, the number of mesh cells of the finest mesh reaches 2 million cells.

Table 5.2 Parameters and Sampling positions for Mesh sensitivity study

Parameters	Number of variations	Values	Abbreviations
Mesh size	3	6	Mesh1
		3	Mesh2
		1.5	Mesh3
Aspect ratio	2	20	Ar1
		40	Ar2

The peak gas concentrations at predefined locations, which are also where gas arrays sensor located, are used as a comparison parameter between meshes.

Table 5.3 summarises the result of all cases in the mesh sensitivity study. We can see that the mesh refinement has effect on the prediction of gas concentrations. However, the `Ar` parameter does not have much effect on this prediction, e.g. both `Mesh1Ar1` and `Mesh1Ar2` give similar result. On the other hand, all values are over-predicted as compared

to benchmark data. This is due to the fact that the solver uses the default value of turbulent Schmidt number $Sc_t = 1$, which is shown not suitable for dense gas dispersion where the concentration diffusivity is much larger than momentum diffusivity due to gravity effect. A numerical experiment on the effect of Sc_t in gas dispersion is conducted in the following section.

Table 5.3 Result for mesh sensitivity study

Positions	50.27	100.53	150.81	201.08	251.35	301.61	351.9	389.6
Mesh1Ar1	18.37	8.96	5.57	3.96	3.05	2.39	2	1.77
Mesh2Ar1	17.57	8.63	5.49	3.86	2.98	2.37	1.96	1.74
Mesh3Ar1	17.02	8.13	5.09	3.33	2.17	2.09	1.68	1.24
Mesh1Ar2	18.39	8.88	5.51	3.91	3	2.36	1.97	1.74
Mesh2Ar2	17.62	8.56	5.42	3.8	2.93	2.33	1.92	1.71
Mesh3Ar2	17.17	8.03	5.19	3.36	2.51	2.12	1.35	1.4
EXP	7.8	4.1	2.4	1.8	1.2	0.9	0.6	0.5

Figure 5.2 presents the ratio of experimental and predicted peak concentrations C_m/C_p at sampling locations, using different meshes and the `Ar1` parameter. It can be shown that even the `Mesh3` requires intensive computation, it does not improve the prediction of gas concentrations much comparing to the `Mesh2`. Therefore, the `Mesh2` is used for the following simulations.

5.2.2 Turbulent Schmidt number study

The effect of turbulent Schmidt number Sc_t is investigated in dense gas dispersion. Three test cases are summarised in Table 5.4. The modified $k-\varepsilon$ and SST $k-\omega$ models which are already validated in simulating ABL over flat terrain in Section 4.1 are continually used to study its combinational effect together with the factor of turbulent Schmidt number Sc_t in predicting dense gas dispersion.

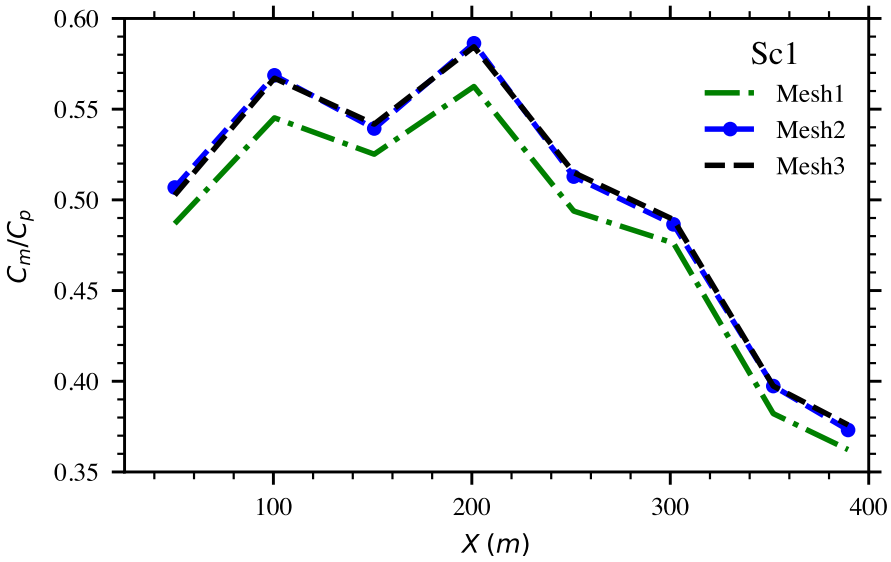


Fig. 5.2 Ratio of measured and predicted peak gas concentrations C_m/C_p in mesh sensitivity study of `DA0120` test

Table 5.4 Turbulent Schmidt number Sc_t in Hamburg tests

Parameters	Number of variations	Values	Abbreviations
Sc_t	1	1	Sc1
		0.7	Sc07
		0.3	Sc03
Turbulence model	2	Modified $k-\varepsilon$	Ske
		SST $k-\omega$	Mke

Table 5.5 shows results of numerical investigation on the effect of turbulence model and turbulent Schmidt number. Overall, both SST $k-\omega$ and modified $k-\varepsilon$ over-predict the gas concentration (the `EXP` row in Table 5.5 is the experimental data).

The predicted and measured peak gas concentrations are compared at several distances from the spill in Figure 5.3 using different values of Sc_t. Turbulent Schmidt number Sc_t has significant effect in predicting dense gas dispersion. The original value of $Sc_t = 1$, with assume species diffusivity equals to viscosity, is shown to over-predicted concentration with factor of three. The developed model has a feature to implement variable Sc_t by reading that value as the user input. The value of $Sc_t = 0.3$ is shown to be the optimum value of all values

Table 5.5 Result for turbulent Schmidt number study

Positions	50.27	100.53	150.81	201.08	251.35	301.61	351.9	389.6
KomeSc1	24.55	11.86	6.96	4.49	3.31	2.56	2.09	1.84
KomeSc07	23.85	11.81	6.91	4.36	3.13	2.36	1.89	1.65
KomeSc03	21.6	10.69	5.83	3.42	2.34	1.73	1.4	1.24
MkeSc1	17.67	8.56	5.42	3.8	2.93	2.33	1.92	1.71
MkeSc07	15.73	7.6	4.76	3.29	2.51	1.99	1.64	1.46
MkeSc03	11.64	4.88	2.82	1.9	1.48	1.23	1.08	1
EXP	7.8	4.1	2.4	1.8	1.2	0.9	0.6	0.5

considered. However, there was a slightly acceptable over-predicted species concentration at a point near the gas source.

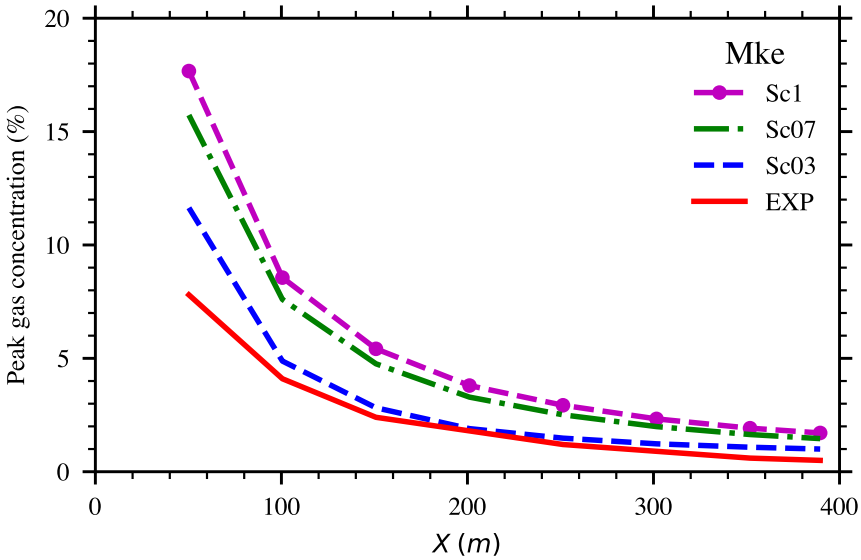


Fig. 5.3 Peak concentration of `DA0120` experiment (EXP) and simulations with various turbulent Schmidt numbers Sc1 = 1, Sc07 = 0.7 and Sc03 = 0.3

Figure 5.4 presents the result of predicted and measured peak gas concentrations using the best value of $Sc_t = 0.3$ and two turbulent models: SST $k-\omega$ and modified $k-\varepsilon$. The modified $k-\varepsilon$ is shown to outperform the SST $k-\omega$ model.

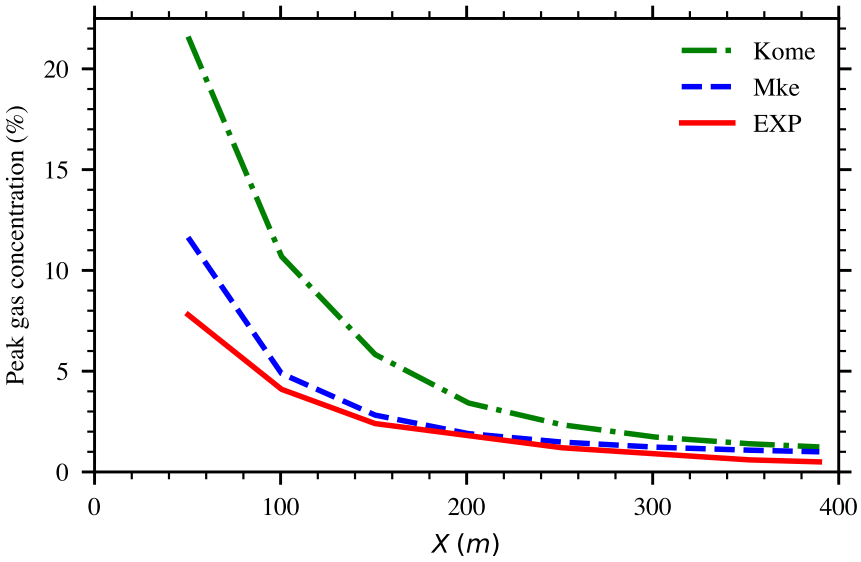


Fig. 5.4 Comparison of peak concentrations of DA0120 experiment (EXP) and simulations using SST $k-\omega$ (Kome) and modified $k-\varepsilon$ (Mke) turbulence models

5.2.3 Roughness length study

Similar to the study of roughness length z_0 in ABL simulations, three different of z_0 are used to test the effect of this parameter in dense gas dispersion. These test cases are summarised in Table 5.4. Higher values of z_0 represents higher surface roughness elements. Therefore, less gas concentration is expected in Z03 case.

Table 5.6 Roughness length z_0 in Hamburg tests

	Case 1	Case 2	Case 3
label	Z01	Z02	Z03
z_0	0.001	0.01	0.1

Effect of the roughness length z_0 in dense gas dispersion is presented in Table 5.7 and Figure 5.5. Increasing value of roughness length lowers the peak gas concentrations at all locations. There is no experimental data for validating this result. However, this trend is reasonable because the higher roughness elements represented by high value of z_0 which trap part of the gas release and result in lower gas concentration.

Table 5.7 Result for mesh sensitivity study

Positions	50.27	100.53	150.81	201.08	251.35	301.61	351.9	389.6
z01	13.93	6.24	3.65	2.38	1.75	1.36	1.12	1
z02	11.13	4.5	2.53	1.65	1.25	1.02	0.87	0.8
z03	6.89	2.5	1.45	1.04	0.86	0.75	0.67	0.63

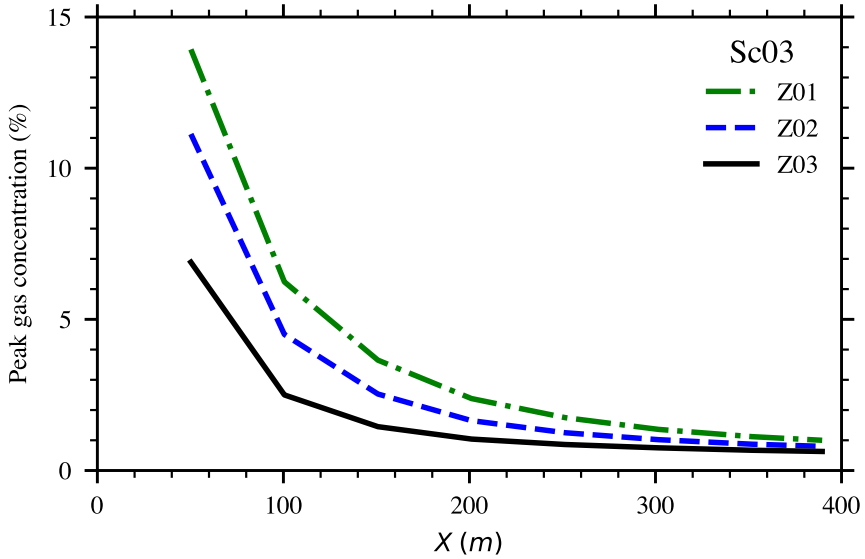


Fig. 5.5 Peak concentration of DA0120 with different roughness length values Z01 = 0.001, Z02 = 0.01 and Z03 = 0.1

5.2.4 Statistical model evaluation

The best combination of parameters in the DA0120 simulation is used for the simulation of DAT223 test, i.e. $Sc_t = 0.3$ and the modified $k-\varepsilon$ turbulent model. The result of this simulation is presented in Figure 5.6. Satisfactory over predicted peak concentration is similar to DA0120 case.

The steady state plumes of DAT0120 and DAT223 tests are plotted in Figures 5.7 and 5.8 respectively. Under higher release volume flow rate and higher wind speed, DAT223 plume is shown for wider and higher concentration in downwind zone.

All predicted and measured values from the two tests DAT0120 and DAT223 are gathered in Figure 5.9.

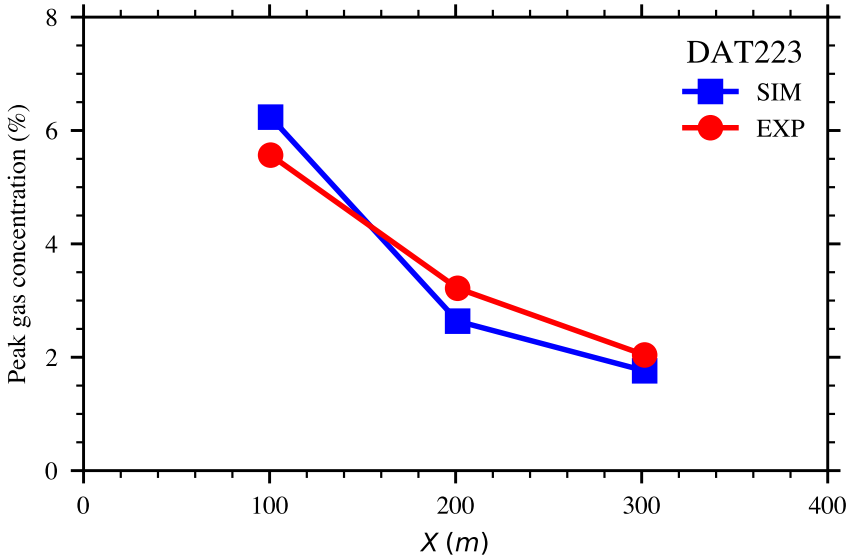


Fig. 5.6 Peak concentration of DAT223 experiment (EXP) and simulation (SIM)

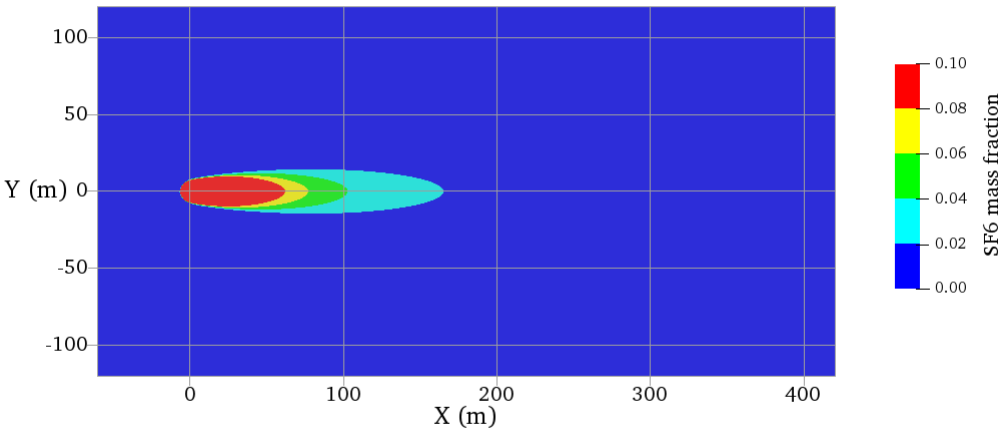


Fig. 5.7 DA0120 contour of gas concentration

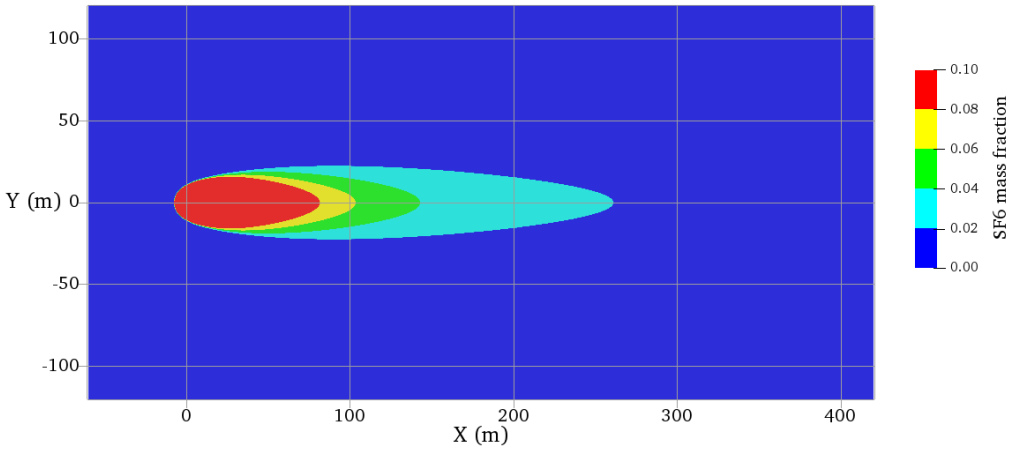


Fig. 5.8 `DAT223` contour of gas concentration

Statistical performance of OpenFOAM results are compared with FLACS, a specified commercial software for gas dispersion, in Table 5.8. FLACS results are extracted from [53]. The performance of current OpenFOAM code is considerably better than FLACS. Even though larger tests were validated in FLACS, proposed model in OpenFOAM is a promising tool for further investigation of atmospheric gas dispersion.

Table 5.8 Statistical performance measures of Hamburg unobstructed tests

	MRB	RMSE	FAC2	MG	VG
Acceptable range	[-0.4,0.4]	< 2.3	[0.5, 2]	[0.67, 1.5]	<3.3
Perfect value	0	0	1	1	1
FLACS [53]	0.25	0.29	0.89	1.34	1.61
FOAM (current study)	0.21	0.11	0.91	0.81	1.12

5.3 LNG vapour dispersion over flat terrain

Four tests in Burro series included in MEP are `Burro3`, `Burro7`, `Burro8` and `Burro9`. These four tests are simulated in this Section to validate the developed model on LNG vapour

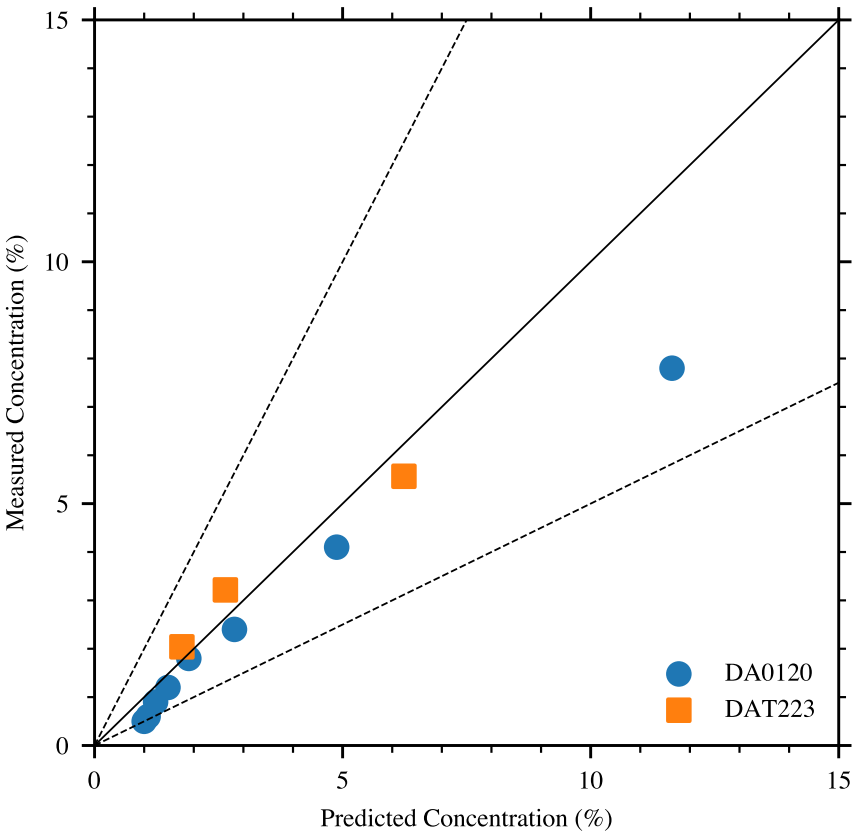


Fig. 5.9 Comparison between predicted and measured concentrations of Hamburg unobstructed tests

dispersion over flat terrain. Of all tests, `Burro7` had the largest spill volume and the longest spill duration. The stability of ABL was varied in these test. The `Burro7` and `Burro9` tests were under neutral stratified ABL. The `Burro3` test was conducted under the most unstable atmospheric conditions. And the `Burro8` test has the most stable atmospheric condition and the lowest wind speed. The detail description of Burro series is summarised in Section 1.4.1.

5.3.1 Domain and computational mesh

Without wind direction fluctuation, the flow is assumed symmetric. Therefore, only half of the flow is simulated. However, symmetric plane of the flow does not coincide with the

centreline of instrument arrays. Therefore, a conversion of data positions should be made when comparing the point-wise data of the simulation and experimental measurements. In Figure 5.10, the instrument arrays at four down-wind positions 57 m, 140 m, 400 m and 800 m are plotted. The transformation of these points to the simulation domain is also presented in that Figure. The location of instruments in simulated domain (x_1, y_1) can be calculated from experimental location (x_2, y_2) using the deflection angle between the wind direction and the centreline of instrument arrays θ_{wind}. This relationship can be written as:

$$\begin{aligned} x_2 &= x_1 \cos\theta_{wind} + y_1 \sin\theta_{wind} \\ y_2 &= y_1 \cos\theta_{wind} - x_1 \sin\theta_{wind} \end{aligned} \tag{5.4}$$

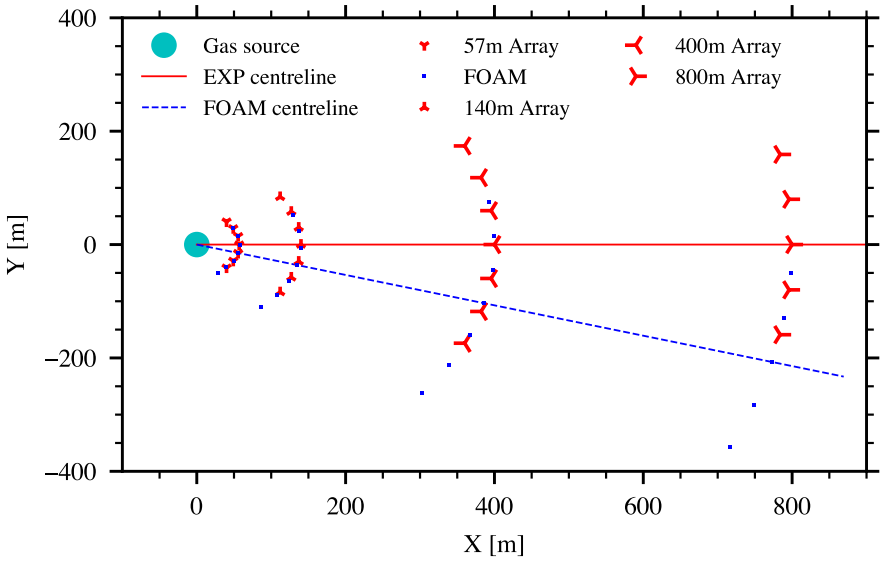


Fig. 5.10 Position of instrument arrays (EXP) and their transformed points in simulation domain (FOAM) in Burro tests

The computational domain used in Burro simulations is similar to the simulation of `DA0120` test (Figure 5.1). The meshing approach is also similar to the `DA0120` simulation that the `blockMesh` application is used and the mesh is refined in gas source region. The mesh independence study is conducted with three different meshes with the number of nodes vary

with a factor of two. The detail of this test is discussed in Section 5.4.1. All computational domain and mesh parameters are summarised in Table 5.12.

5.3.2 Boundary conditions

Atmospheric air inlet boundary Similar to other previous simulations, MOST (Monin-Obukhov similarity theory) is used to specify the atmospheric profiles at the inlet. All required meteorological parameters are tabulated in Table 5.9. The reference temperature T_0 is the reported temperature at $2m$ in the field test.

Table 5.9 Burro tests meteorological parameters

	Burro3	Burro7	Burro8	Burro9
u_{ref}	5.4	8.4	1.8	5.7
z_{ref}	2	2	2	2
z_0	2E-4	2E-4	2E-4	2E-4
u_*	0.249	0.372	0.074	0.252
T_0	33.8	33.7	33.1	35.4

Vapour gas inlet Vapour gas inlet condition is usually obtained from separate source term modelling. There is not much information about the vaporisation of LNG from the experimental data. Therefore, uncertainty arises in the setting of this condition.

Mass flux of LNG or the LNG vaporisation rate is used to derive source term of LNG spill. Luketa-Hanlin et al. [7] reviewed a number of experiments conducting to estimate the LNG vaporisation rate of the spill on water, the range of this value varied between approximately 0.029 to $0.195\,\mathrm{kg\,m^{-2}\,s^{-1}}$. In the case of Burro test, the simulated vaporisation rate is assumed to be $m'' = 0.167\,\mathrm{kg\,m^{-2}\,s^{-1}}$. Density of LNG vapour is approximate as of CH_4 at boiling point $\rho_{LNG} = 1.76\,\mathrm{kg\,m^{-3}}$ [7]. The spill diameter is derived from this vaporisation rate, reported spill mass m and duration Δt:

$$D = \sqrt{\frac{4m}{\pi m'' \Delta t}} \tag{5.5}$$

The volume spill rate is calculated according to the below Equation and used as gas inlet condition for velocity:

$$\dot{V} = \frac{m}{\rho \Delta t} \tag{5.6}$$

All LNG spill variables calculated and used in simulations are summarised in Table 5.10.

Table 5.10 Burro test spill conditions

	Burro3	Burro7	Burro8	Burro9
Vaporization rate ($\mathrm{kg\,m^{-2}\,s^{-1}}$)	0.167	0.167	0.167	0.167
Spill velocity ($\mathrm{m\,s^{-1}}$)	0.024	0.024	0.024	0.024
Spill mass (kg)	14712	17289	12453	10730
Spill duration (s)	167	174	107	79
Volume spill rate ($\mathrm{m^3\,s^{-1}}$)	50.05	56.46	66.13	77.17
Spill pool diameter (m)	25.9	27.5	29.9	32.2

Ground boundary conditions The ground heating is one of most important physical processes in dispersion of LNG vapour. In the report of Burro experiments, Koopman et al. [21] showed that ground heating was a significant heat source transferred to the LNG cloud. The nominal value of heat flux from the ground was $200\,\mathrm{W/m^2}$ when the LNG cloud temperature was 15^0C lower than ambient air temperature.

In this study, three different models of heat transfer from the ground are used to study the effect of ground heating in numerical results, which summarised in Table 5.13. For constant heat flux case, the value of $200\,\mathrm{W/m^2}$ is used. For other variables, the same boundary conditions are used as described in Section 5.1.3

The `top`, `side` and `outlet` boundaries The same boundary conditions used in Section 5.1.3 are specified at the `top`, `side` and `outlet` boundaries.

5.3.3 Thermo-physical models

CoolProp, an open-source thermo-physical property library [75] is used to derive incompressible thermal physical properties for air and CH_4 to take into account the variable gas properties due to temperature changes. Coefficients to derive gas properties as the function of temperature according to Equation (3.69) are presented in Table 5.11.

$$\rho = \sum_{i}^{N} a_i T^i \qquad \text{(3.69 revisited)}$$

Table 5.11 Coefficients (Eq. (3.69)) of gas thermophysical properties used in Burro tests simulation

		a_0	a_1	a_2	a_3
Air	ρ	9.205	-0.094	0.0005	-1.328E-6
	c_p	1092.096	-1.004	0.0042	-7.691E-6
	μ	7.056E-7	6.9536E-8	-3.432E-11	
	κ	0.00333	7.38E-5		
CH_4	ρ	5.405	-0.057	0.0003	8.3E-7
	c_p	3798.83	-33.3575	0.273	-0.0012
	μ	-3.73E-7	4.513E-8	-2.12E-11	
	κ	-0.0073	0.000145		

5.3.4 Numerical setting

The standard $k-\varepsilon$ with modifications has shown its outperformance in comparison with SST $k-\omega$ in predicting dense gas dispersion over flat terrain. Therefore, it is used in all simulations in this Section.

Using the same approach to previous Section, the simulation firstly solve the atmospheric profiles prior to the gas release using a steady simulation. Result from this simulation is used as initial fields of the transient simulation.

The transient simulation is divided into two steps. The first step is during the spill duration, i.e. from the time of zero to the time of spill ends. And the second step is from after spill ends to the simulation end time. The gas inlet is treated as a ground boundary in this later step.

The same setting of numerical schemes and solution control as in Section 5.1.4 are adopted for the simulations in this Section.

5.4 Simulations of Burro LNG vapour dispersion tests

5.4.1 Mesh sensitivity study

Maximum concentration at the arcs of 57 m, 140 m, 400 m and 800 m downwind are used as control parameters for mesh sensitivity study. Three meshes as summarised in Table 5.12 are used to simulate LNG gas dispersion at adiabatic thermal wall condition. The number of mesh cells differs in the factor of two between meshes. Each mesh has a label to represent its case for illustrations.

Table 5.12 Burro tests computational domain and mesh parameters

	Mesh 1	Mesh 2	Mesh 3
Domain region	[(-150, 0, 45), (900, 300, 45)]		
Refined region 1	[(-50, 0, 15) , (300, 100, 15)]		
Refined region 2	[(-25, 0, 7.5) , (150, 50, 7.5)]		
Aspect ratio	20		
Vertical cell expansion ratio	1.1		
Mesh size (m)	6	4	3
Mesh refined size (m)	2.5	1.5	1
Mesh cell numbers	348,406	897,600	1769,652
Label	Mesh1	Mesh2	Mesh3

As mentioned in the description of mesh sensitivity study (Section 5.4.1). Peak gas concentrations at specific arrays downwind are in concerned and used as performance metrics. These concentrations are compared with experimental data. The ratio of measured and predicted concentrations C_m/C_p resulted from three meshes `Mesh1`, `Mesh2` and `Mesh3` are plotted in Figure 5.11. We can see that the mesh refinement has effect on the results of peak concentrations. The difference between `Mesh1` and `Mesh2` is more significant than of `Mesh2` and `Mesh3`. Even though `Mesh3` has twice number of cells more than `Mesh2`, there is no significant change in the predictions of these two meshes. Therefore, the `Mesh2` (Table 5.12) is used in the following simulations. It can also be seen that the simulation using the coarsest mesh, `Mesh1`, agree better with experiment than other two meshes. This means that the setting used for dense gas dispersion is not validated for LNG vapour dispersion. In the Section 5.4.3, an investigation on the effect of turbulent Schmidt number Sc_t is conducted. The validated value of $Sc_t = 0.3$ in dense gas dispersion is varied to find a more appropriate value of Sc_t for LNG vapour dispersion.

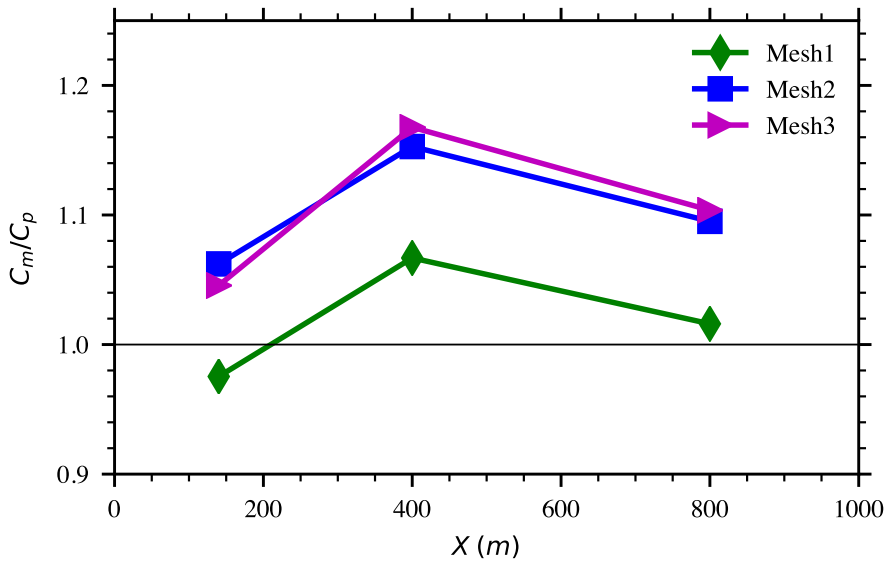


Fig. 5.11 Ratio of measured and predicted peak gas concentrations C_m/C_p in `Burro9` mesh sensitivity study

5.4.2 Ground heat transfer sensitivity study

As highlighted in Section 5.3.2, the heat source from ground can be varied from minimum using zero heat flux boundary condition to maximum using fixed temperature boundary condition. Three different ground heat transfer models as in Table 5.13 are used to examine the effect of ground heat transfer in predicting peak gas concentrations.

Table 5.13 Wall thermal boundary conditions in Burro tests

	Case 1	Case 2	Case 3
Heat transfer model	Adiabatic ground	Constant heat flux	Fixed temperature
Label (Fig. 5.12)	Adiabatic	fixedFlux	fixedTem

Plotted in Figure 5.12 are results from this study. All predictions are under-predicted (all lines are above the line of C_m/C_p = 1 in Figure 5.12). However, the fixed temperature ground case results a better prediction to experiment data than fixed flux and zero gradient cases. Of all cases, fixed temperature ground results in the largest heat flux from the ground to the vapour cloud, so it can compensate other source of heat addition to the cloud which is not considered in the simulation such as the latent heat of vaporisation and the radiation heat.

5.4.3 Turbulent Schmidt number study

The value of $Sc_t = 0.3$ used previously in wind tunnel dense gas dispersion is shown to be not appropriate in accurate prediction of peak gas concentration in LNG vapour dispersion. Therefore, three value of $Sc_t = 0.45$, $Sc_t = 0.3$ and $Sc_t = 0.15$ are used for studying the sensitivity of developed model on predicting peak LNG vapour dispersion concentration.

Table 5.14 Turbulent Schmidt number in `Burro9` test

	Case 1	Case 2	Case 3
Sc_t	0.45	0.3	0.15
Label	Sc1	Sc2	Sc3

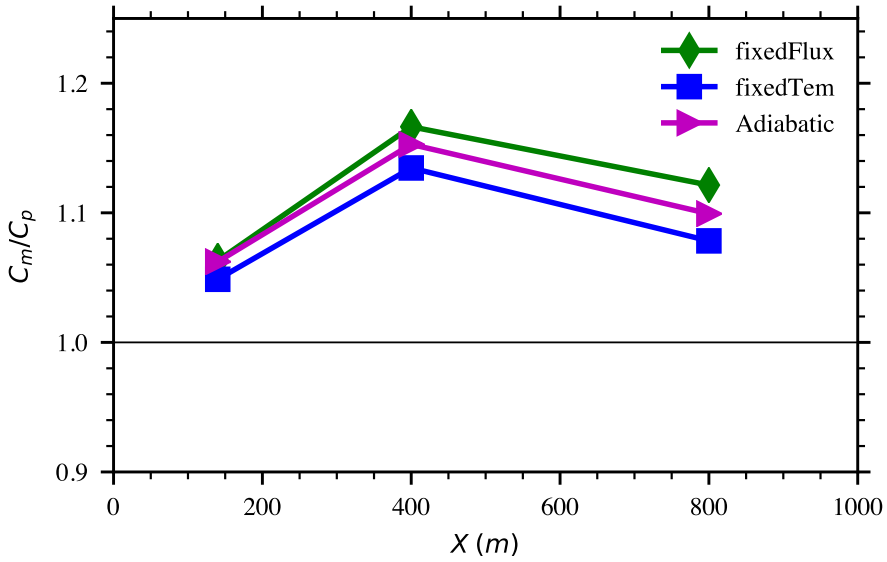


Fig. 5.12 Ratio of measured and predicted peak gas concentrations C_m/C_p in `Burro9` ground heat transfer study. Adiabatic: Adiabatic ground, fixedFlux: Constant heat flux ground and fixedTem: Fixed temperature ground

Results of turbulence Schmidt number study is shown in Figure 5.13. Similar to simulations of dense gas, turbulent Schmidt number Sc_t has significant effect in predicting dense gas dispersion. Decreasing Sc_t number from `Sc1` to `Sc3` increases the predicted peak concentrations and helps to reduce the under-estimations in previous study (Figure 5.12). `Sc3` case results in the perfect prediction of concentration at 800 m arc.

The best value of $Sc_t = 0.15$ shows that this low value of Sc_t can compensate the under-prediction of turbulence in the vapour cloud to get the correct gas concentration predictions. High turbulent level in LNG vapour cloud is very hard to model using simple turbulent model. However, the low value of Sc_t can be used to correct the mass flux in the cloud.

5.4.4 Burro series simulations

Other tests of Burro series such as `Burro3`, `Burro7` and `Burro8` are simulated using the optimum set of parameters resulted from `Burro9` simulations in previous sections. These are `Mesh2`, Modified $k-\varepsilon$ turbulent model, fixed ground temperature and $Sc_t = 0.15$.

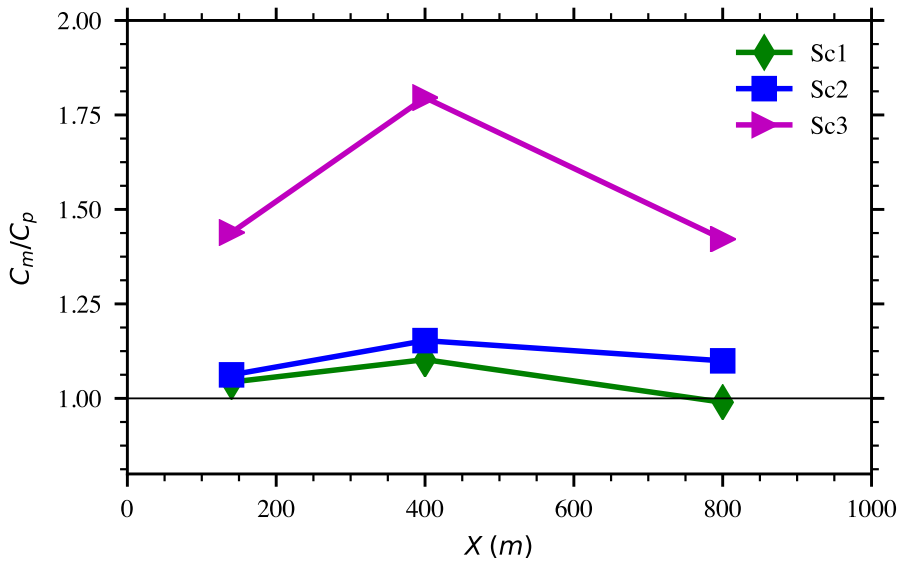


Fig. 5.13 Ratio of measured and predicted peak gas concentrations C_m/C_p in `Burro9` with variable turbulent Schmidt numbers Sc1 = 0.45, Sc2 = 0.3 and Sc3 = 0.15

Peak concentration prediction

The maximum concentrations of instrument arrays are used to compare experimental and simulated data. This comparison for `Burro9` test is shown in Figure 5.14. The OpenFOAM result is under-predicted and accurately predict the peak gas concentration at 800 m arc.

Results from the simulations of other three Burro tests are compared with experimental data to show the overall performance of FOAM in Figures 5.15, 5.16 and 5.17 respectively. Over-predictions are observed in all these simulations.

Figure 5.15 presents the peak concentrations of the `Burro3` test, which conducted in unstable ABL. The peak concentration at 800 m arc is well predicted. However, all other arcs are over-predicted. The over-prediction is higher at 140 m arc and smaller at 57 m and 400 m arcs.

Under unstable to neutral ABL stability in `Burro7`, the over-prediction are shown in all arcs as in Figure 5.16. The over-prediction is higher at 57 m arc and smaller at 140 m and 400 m arcs. The peak concentration at 800 m is however well predicted.

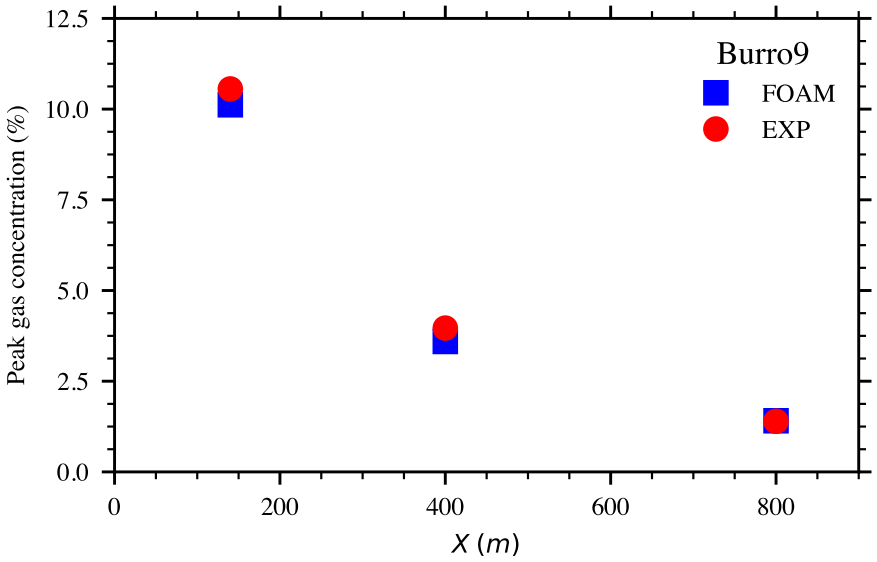


Fig. 5.14 Maximum arc-wise concentrations of `Burro9` experiment (EXP) and simulation using developed solver (FOAM)

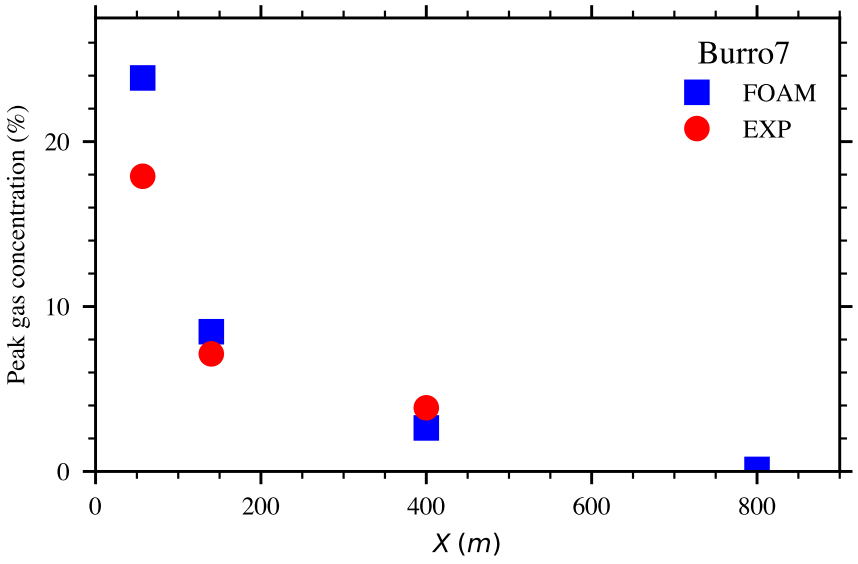


Fig. 5.15 Maximum arc-wise concentrations of `Burro3` experiment (EXP) and simulation using developed solver (FOAM)

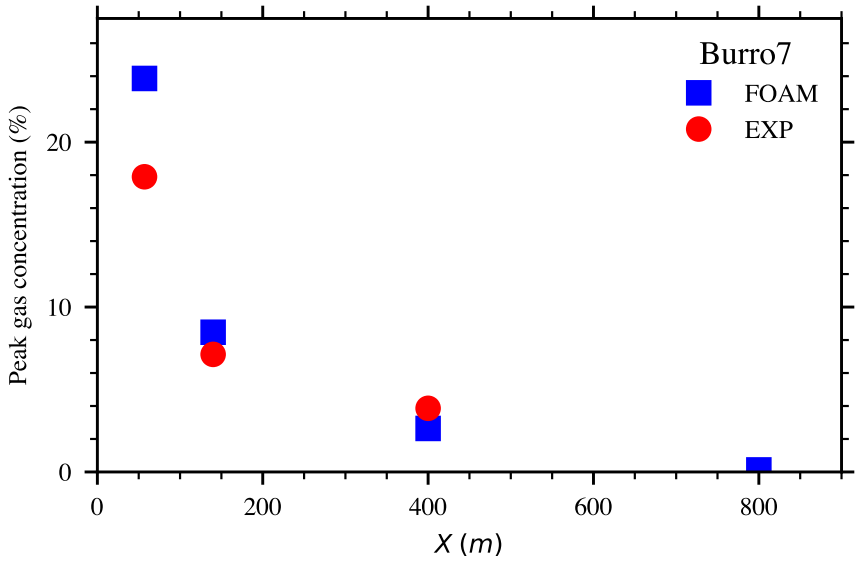


Fig. 5.16 Maximum arc-wise concentrations of `Burro7` experiment (EXP) and simulation using developed solver (FOAM)

Under stable stratified ABL at `Burro8` test, the prediction at near source region 57 m is under-predicted and over-predicted in other arcs as seen in Figure 5.17.

Point-wise profiles

For further understanding the result, the experimental point-wise concentration data is compared to the simulated result. The first point is selected near the source, which is 57 m downwind and the second point is 140 m downwind.

FDS (Fire Dynamics Simulator) [76] is a low Mach number code using LES turbulence model. FDS uses the finite-difference approximation of the governing equations on a series of connected rectilinear mesh. The numerical schemes are second-order accurate. The flow variables are updated in time using an explicit second-order Runge-Kutta scheme. The result of FDS simulation for the `Burro9` test is extracted from [57]. This data is compared with current OpenFOAM simulation to see the difference between LES and RANS turbulent model.

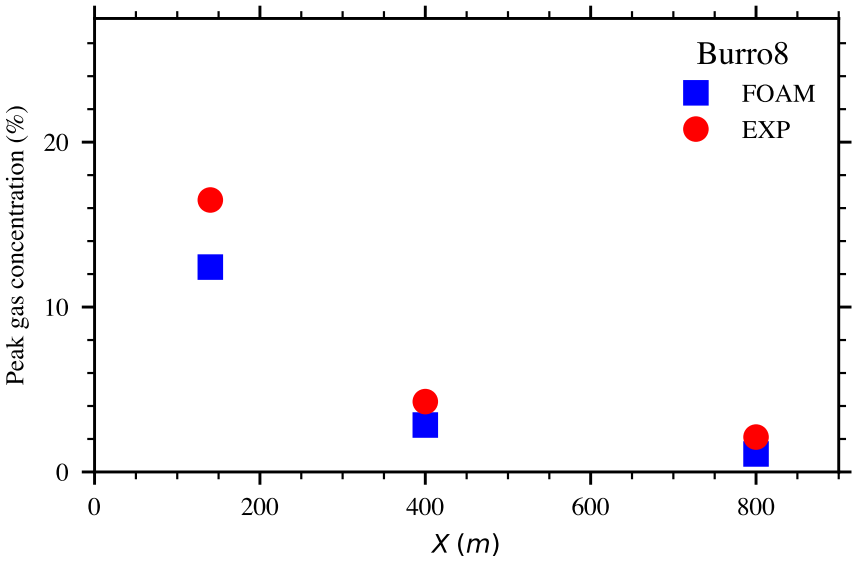


Fig. 5.17 Maximum arc-wise concentrations of `Burro8` experiment (EXP) and simulation using developed solver (FOAM)

Figure 5.18 is a plot of gas concentration at 1 m elevation at 140 m downwind of `Burro9` of experiment and simulations using the developed solver FOAM and FDS. The developed solver shows the good temporal trend of the simulation to the validation data. The peak concentration is underestimated but acceptable. The concentration magnitude is really matching well with the validation data comparing with FDS code where over-prediction of gas concentrations is shown. However, the developed solver cannot predict the fluctuation of gas concentration. This is an advantaged point of LES code FDS, where gas concentration is fluctuating over the time period.

For other tests in Burro series, gas concentration at 1 m elevation are plotted with data from experiments. Temporal variation of gas concentration at 140 m downwind of the `Burro3` and `Burro7` tests are presented in Figures 5.19 and 5.20 respectively. These tests are in unstable ABL, and over-predictions are seen in both tests.

For `Burro8` test, under stable ABL, the gas concentration at 57 m downwind is shown in Figure 5.21. The model is shown to well-predicted the stable concentration at later time but cannot reproduce the peak concentration periods.

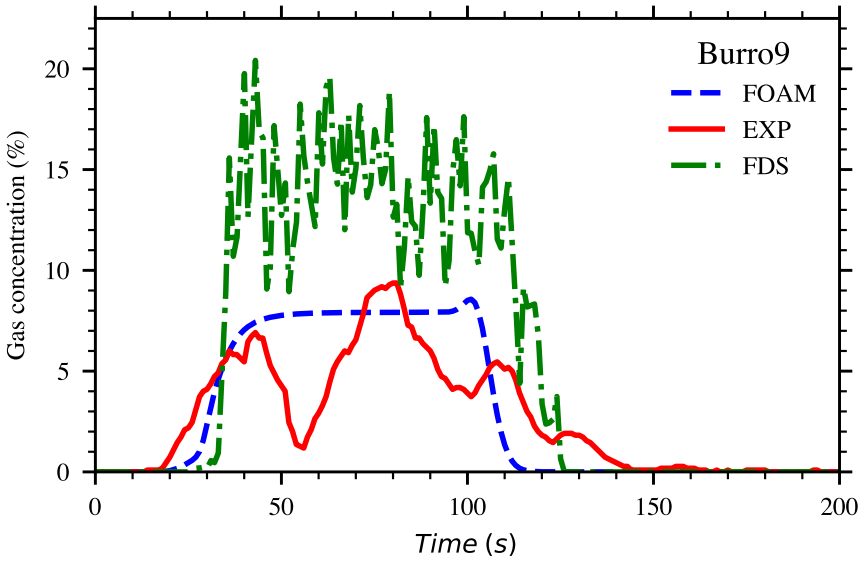


Fig. 5.18 Point concentration at 140 m of `Burro9` experiment (EXP) and simulations using developed solver (FOAM) and FDS code (FDS)

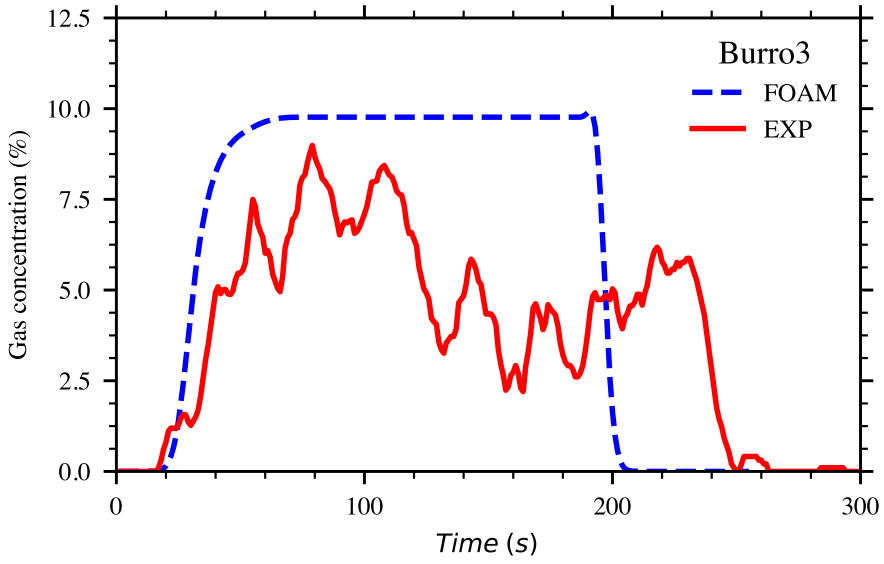


Fig. 5.19 Point concentration at 140 m of `Burro3` experiment (EXP) and simulation using developed solver (FOAM)

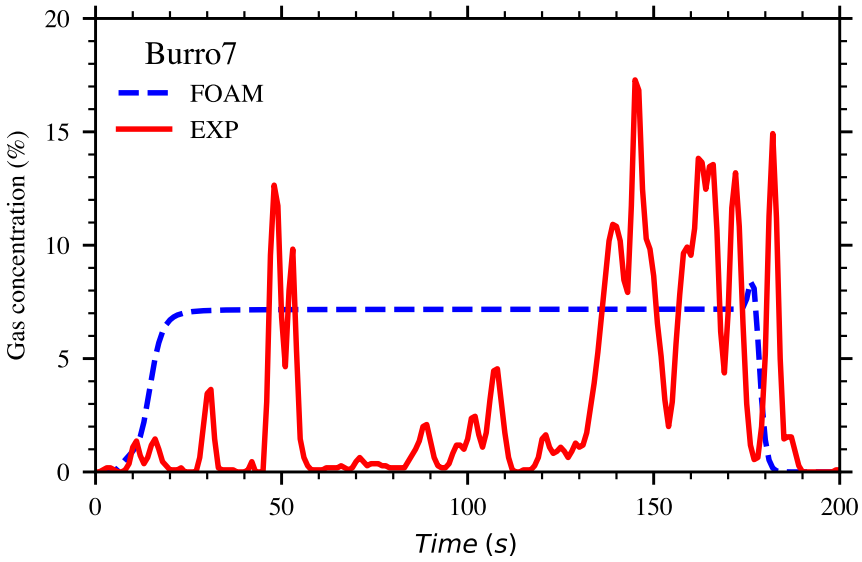


Fig. 5.20 Point concentration at 140 m of `Burro7` experiment (EXP) and simulation using developed solver (FOAM)

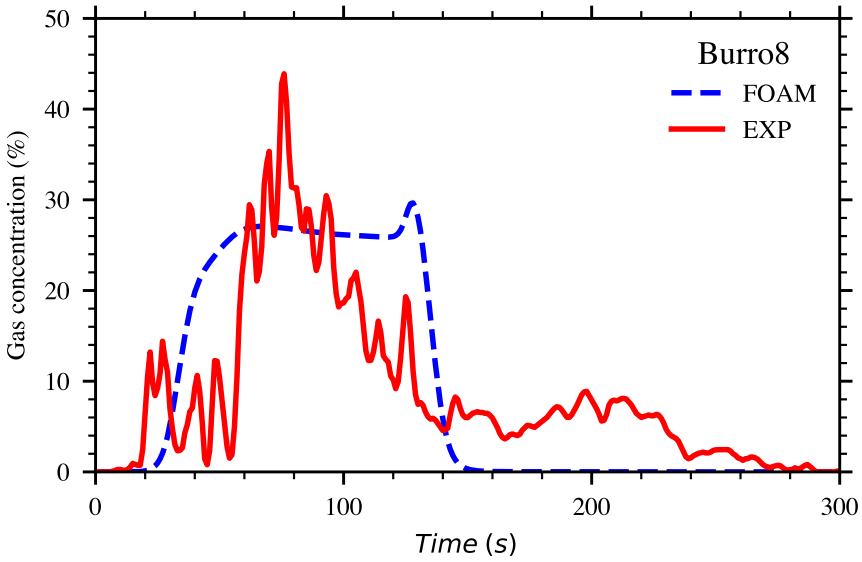


Fig. 5.21 Point concentration at 57 m of `Burro8` experiment (EXP) and simulation using developed solver (FOAM)

Statistical model evaluation

Overall statistical performance of OpenFOAM results are compared with FLACS with data extracted from [53] in Table 5.15. The developed solver fulfils all SMPs, i.e. all SPMs

are in acceptable range. The factor of two measure is perfectly matched with experiment data (FAC2=1). Comparing with FLACS code, the developed solver has better performance. FLACS cannot have perfect measure of FAC2 as this value is 0.94.

Table 5.15 Statistical performance measures of Burro tests

	MRB	RMSE	FAC2	MG	VG
Acceptable range	[-0.4,0.4]	< 2.3	[0.5, 2]	[0.67, 1.5]	<3.3
Perfect value	0	0	1	1	1
FLACS [53]	0.16	0.12	0.94	1.18	1.14
FOAM (current study)	-0.15	0.1	1	1.16	1.11

Figure 5.22 presents the comparison of all predicted and measured concentrations in Burro tests.

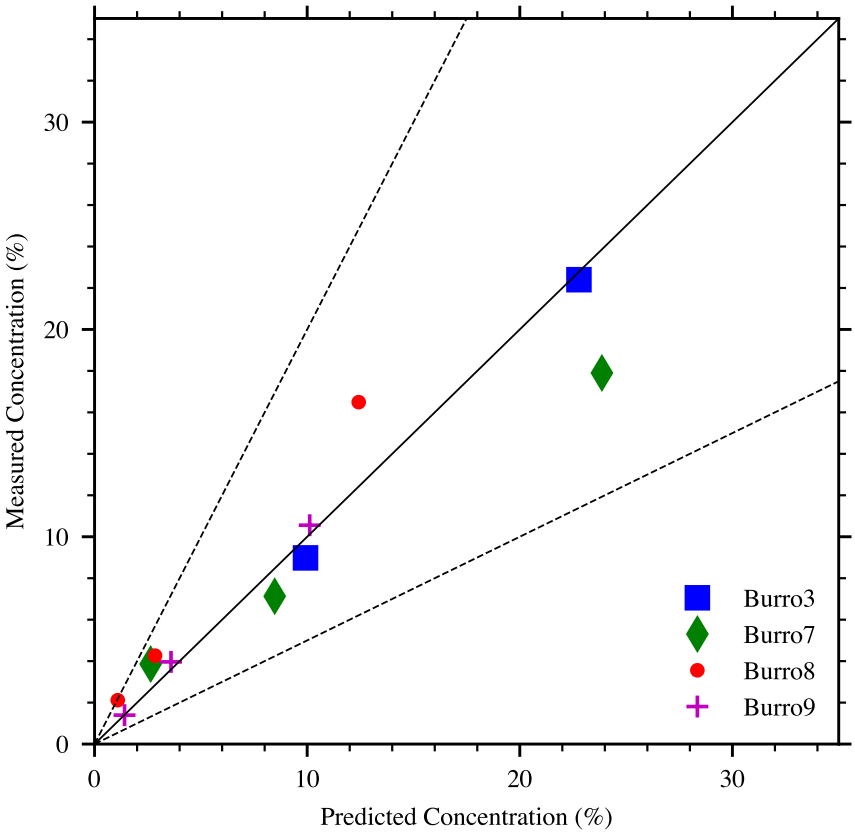


Fig. 5.22 Comparison between predicted and measured concentrations of Burro tests

Chapter 6

Conclusion and future works

6.1 Conclusion

In this Thesis, the atmospheric profiles of velocity, turbulent kinetic energy k and turbulent dissipation rate ε modelled by Monin-Obukhov similarity theory are used as inlet boundary conditions to reproduce horizontal homogeneous atmospheric turbulent surface layer (HHTSL) state in neutrally stratified condition, i.e. these profiles at outlet boundary should be maintained and consistent with inlet profiles. The developed model takes into account the consistency between inlet profiles, turbulent models and wall functions. Various turbulent models are tested including standard $k-\varepsilon$, buoyant $k-\varepsilon$ and SST $k-\omega$. Simulation results have shown the effectiveness of proposed model in simulating HHTSL. Furthermore, the proposed model can simulate different levels of ABL turbulence kinetic energy. Applying the model to simulate ABL flows with different roughness length, it is found that the ground adjacent cell size should be carefully selected. The best value of this cell size is ten times the roughness length in case of high surface roughness case.

For gas dispersion simulations, a solver is developed in OpenFOAM considering buoyancy effect, variable turbulence Schmidt and ground heat transfer. In the study of dense gas dispersion in wind-tunnel tests, the solver is successfully validated in reproducing the

maximum gas concentrations from benchmark database. SPMs resulted from simulations are better than from the specified commercial software for gas dispersion FLACS.

In the study of LNG vapour dispersion, three ground heat transfer assumptions are simulated and compared with validation data defined in Model Evaluation Protocol (MEP). The adiabatic ground assumes zero heat flux from ground to vapour cloud. Whereas, the fixed temperature ground assumes an isothermal ground (ground temperature is constant) resulted in the maximum heat flux. The real heat flux from ground to vapour cloud should be in between these two cases. The last assumption is a fixed flux of heat from ground to vapour cloud. In all three cases, gas peak concentrations are used as validation parameters. The fixed temperature ground gives the closest prediction with experimental gas peak concentrations. This case results in the largest heat flux from ground to vapour cloud, so it can compensate other sources of heat addition to the cloud which are not considered in the simulation such as latent heat of vaporisation and radiation heat. Statistical Performance Measures (SPMs) from simulation results are compared with FLACS, a specified commercial CFD software developed for explosion and atmospheric dispersion. It has been shown that the developed model outperformed FLACS in all SPMs. However, peak concentrations in near-source distances are quite conservative in comparison with validation data. The same conclusion has been made when comparing the simulated and experimental point-wise data of near-source points. The assumption of source term may contribute largely in this error. Therefore, the source modelling should be more well defined for the better prediction.

6.2 Contributions

The consistency between inlet profiles, turbulent models and wall functions is shown to be an essence in predicting horizontal homogeneous ABL. A developed model using OpenFOAM includes modifications of two equation turbulence models such as standard $k-\varepsilon$, buoyant $k-\varepsilon$ and SST $k-\omega$, wall functions of k, ε, ω and ν_t based on surface roughness length,

as well as inlet boundary conditions of U, k, ε, and ω based on Monin-Obukhov Similarity Theory. The model successfully simulates horizontal homogeneous ABL under neutral condition. It can also model different levels of ABL turbulence kinetic energy. Using this model for simulations of different values surface roughness length, it is found that the choice of ground adjacent cell size is a critical factor. This cell size should be larger than the roughness length.

The gas dispersion model taking into account buoyancy effect, variable turbulence Schmidt number Sc_t, ground heat transfer is developed under OpenFOAM platform. The model is validated using data from wind-tunnel unobstructed dense gas dispersion and field tests of LNG dispersion. The model is shown to accurately reproduce peak concentration and meet all SPMs defined in MEP. The fixed temperature ground is an appropriate boundary condition for LNG dispersion simulation. It can help to compensate other heat sources to LNG vapour cloud and accurately predict the gas concentration. For unobstructed gas dispersion, $k-\varepsilon$ model outperforms the SST $k-\omega$ model in predicting the peak gas concentration. Turbulence Schmidt number is an important parameter to adjust the gas mass flux. The best value of this parameter from this study is $Sc_t = 0.3$ for dense gas dispersion and $Sc_t = 0.15$ for LNG vapour dispersion.

6.3 Limitations and Future works

The assumption of ABL surface layer steady profiles constrained the study to only RANS turbulent models. Turbulence is inherently unsteady. Therefore, for more accurately solving atmospheric turbulence, the Large Eddy Simulation (LES) is indeed a promising approach. Nevertheless, LES requires a more intensive computational cost, especially for large scale inherited in atmospheric flows. However, the boundary conditions used in LES should be carefully applied.

The developed solvers are validated for only unobstructed tests. Influence of obstacles and topography on gas dispersion is a possible future work. In this case, not only the proposed turbulent model should ensure the consistency between inlet and outlet ABL boundary, but also predicting validated pattern of turbulence around obstacle. The $k-\varepsilon$ model developed in this Thesis may not have a good performance in this case because of its limitation in predicting recirculation flow near obstacle. But SST $k-\omega$ or other advanced turbulence models are possible consideration to successfully simulate ABL flows with obstruction.

The Thesis has limitation of only validated for neutrally stratified ABL. Under stable and unstable ABL, heat flux from ground play an important role in defining turbulence structure in the ABL. The developed solver has potential to model stratified ABL, since it is based on a compressible solver with thermo-physical library to take into account variable thermal properties such as density. However, for stably and unstably stratified ABL, modification of turbulent model is needed to ensure the consistency with inlet profile. Other potential approach for modelling the dynamic of ABL under stratified stability conditions is using LES turbulent model.

The only gas source type studied in this Thesis is continuous release. Other popular gas sources in industrial application are elevated and instantaneous releases. Furthermore, the gas source considered is in single phase with multi-species components. In case of LNG spill tests, the better scenario is two-phase release where liquid and gas phase of LNG are interacted in the spill pool. Therefore, considering two-phase source is also a direction for future works. However, integrating two-phase flow in a dispersion simulation may be complex and time-consuming. A promising approach is doing a separate simulation of two-phase gas source and using result from this simulation as the source term input for the dispersion simulation. Besides, in simulation of LNG vapour dispersion, a rigorous surface heat transfer model, which is a major heat source to the flow, is required. The transient

behaviour of ground temperature due to contacting with cold flow should be also taken into account in determining the heat transfer from ground to the vapour cloud.

References

[1] Paolo Zannetti. *Air pollution modeling: theories, computational methods and available software*. Springer Science & Business Media, 2013. ISBN: 147574465X.

[2] Paul S Arya. *Introduction to micrometeorology*. Vol. 79. Elsevier, 2001. ISBN: 0080489265.

[3] M Mohan. "Analysis of various schemes for the estimation of atmospheric stability classification". In: *Atmospheric Environment* 32.21 (1998), pp. 3775–3781. ISSN: 13522310. DOI: 10.1016/S1352-2310(98)00109-5. URL: http://www.sciencedirect.com/science/article/pii/S1352231098001095.

[4] Thomas Foken. "50 Years of the Monin–Obukhov Similarity Theory". In: *Boundary-Layer Meteorology* 119.3 (2006), pp. 431–447. ISSN: 1573-1472. DOI: 10.1007/s10546-006-9048-6. URL: https://doi.org/10.1007/s10546-006-9048-6.

[5] J. A. Businger et al. "Flux profile relationships in the atmospheric surface layer". In: *Journal of the Atmospheric Sciences* 28.2 (1971), pp. 181–189.

[6] Ulf Högström. "Review of some basic characteristics of the atmospheric surface layer". In: *Boundary-Layer Meteorology* 78.3 (1996), pp. 215–246. ISSN: 1573-1472. DOI: 10.1007/BF00120937. URL: https://doi.org/10.1007/BF00120937.

[7] Anay Luketa-Hanlin, Ronald P. Koopman, and Donald L. Ermak. "On the application of computational fluid dynamics codes for liquefied natural gas dispersion". In: *Journal of Hazardous Materials* 140.3 (2007), pp. 504–517. ISSN: 03043894. DOI: 10.1016/j.jhazmat.2006.10.023.

[8] J E Pieterse and T M Harms. "CFD investigation of the atmospheric boundary layer under different thermal stability conditions". In: *Journal of Wind Engineering and Industrial Aerodynamics* 121 (2013), pp. 82–97. ISSN: 0167-6105. DOI: https://doi.org/10.1016/j.jweia.2013.07.014. URL: http://www.sciencedirect.com/science/article/pii/S0167610513001591.

[9] Ronald P. Koopman and Donald L. Ermak. "Lessons learned from LNG safety research". In: *Journal of Hazardous Materials* 140.3 (2007), pp. 412–428. ISSN: 03043894. DOI: 10.1016/j.jhazmat.2006.10.042.

[10] M. J. Ivings et al. "A protocol for the evaluation of LNG vapour dispersion models". In: *Journal of Loss Prevention in the Process Industries* 26.1 (2013), pp. 153–163. ISSN: 09504230. DOI: 10.1016/j.jlp.2012.10.005. URL: http://dx.doi.org/10.1016/j.jlp.2012.10.005.

[11] Rex E Britter. "Atmospheric dispersion of dense gases". In: *Annual review of fluid mechanics* 21.1 (1989), pp. 317–344. ISSN: 0066-4189.

[12] Morten Nielsen. *A collection of data from dense gas experiments*. Tech. rep. 1996.

[13] Donald L Ermak et al. *Heavy gas dispersion test summary report*. Tech. rep. 1989.

[14] T C Brown et al. *Falcon series data report: 1987 LNG vapor barrier verification field trials*. Tech. rep. 1990.

[15] Phil Cleaver, Mike Johnson, and Ben Ho. "A summary of some experimental data on LNG safety." In: *Journal of hazardous materials* 140.3 (2007), pp. 429–38. ISSN: 0304-3894. DOI: 10.1016/j.jhazmat.2006.10.047.

[16] C Biltoft. *Customer report for Mock Urban Setting Test. Rep. no*. Tech. rep. 2001.

[17] K Jerry Allwine and Julia E Flaherty. "Urban dispersion program overview and MID05 field study summary". In: (2007).

[18] Benjamin R. Cormier et al. "Application of computational fluid dynamics for LNG vapor dispersion modeling: A study of key parameters". In: *Journal of Loss Prevention in the Process Industries* 22.3 (2009), pp. 332–352. ISSN: 09504230. DOI: 10.1016/j.jlp.2008.12.004. URL: http://www.sciencedirect.com/science/article/pii/S0950423008001629.

[19] Steven Hanna et al. "The Jack Rabbit chlorine release experiments: Implications of dense gas removal from a depression and downwind concentrations". In: *Journal of Hazardous Materials* 213-214.Supplement C (2012), pp. 406–412. ISSN: 0304-3894. DOI: https://doi.org/10.1016/j.jhazmat.2012.02.013. URL: http://www.sciencedirect.com/science/article/pii/S0304389412001598.

[20] A M Schleder et al. "Experimental data and CFD performance for cloud dispersion analysis: The USP-UPC project". In: *Journal of Loss Prevention in the Process Industries* 38 (2015), pp. 125–138. ISSN: 0950-4230. DOI: http://dx.doi.org/10.1016/j.jlp.2015.09.003. URL: http://www.sciencedirect.com/science/article/pii/S0950423015300280.

[21] R.P. Koopman et al. "Analysis of Burro series 40-m3 lng spill experiments". In: *Journal of Hazardous Materials* 6.1-2 (1982), pp. 43–83. ISSN: 03043894. DOI: 10.1016/0304-3894(82)80034-4.

[22] Filippo Gavelli, Edward Bullister, and Harri Kytomaa. "Application of CFD (Fluent) to LNG spills into geometrically complex environments." In: *Journal of hazardous materials* 159.1 (2008), pp. 158–68. ISSN: 0304-3894. DOI: 10.1016/j.jhazmat.2008.02.037.

[23] Christopher J Greenshields. *OpenFOAM user guide version 5*. 2017. URL: https://cfd.direct/openfoam/user-guide/.

[24] P J Richards and R P Hoxey. "Appropriate boundary conditions for computational wind engineering models using the k-ε turbulence model". In: *Journal of Wind Engineering and Industrial Aerodynamics* 46-47.Supplement C (1993), pp. 145–153. ISSN: 0167-6105. DOI: https://doi.org/10.1016/0167-6105(93)90124-7. URL: http://www.sciencedirect.com/science/article/pii/0167610593901247.

[25] Yi Yang et al. "New inflow boundary conditions for modelling the neutral equilibrium atmospheric boundary layer in computational wind engineering". In: *Journal of Wind Engineering and Industrial Aerodynamics* 97.2 (2009), pp. 88–95. ISSN: 0167-6105. DOI: http://dx.doi.org/10.1016/j.jweia.2008.12.001. URL: http://www.sciencedirect.com/science/article/pii/S0167610508001815.

[26] Sumner Jonathon and Masson Christian. "k - ε simulations of the neutral atmospheric boundary layer: analysis and correction of discretization errors on practical grids". In: *International Journal for Numerical Methods in Fluids* 70.6 (2012), pp. 724–741. ISSN: 0271-2091. DOI: doi:10.1002/fld.2709. URL: https://doi.org/10.1002/fld.2709.

[27] A Parente et al. "Improved k–ε model and wall function formulation for the RANS simulation of ABL flows". In: *Journal of Wind Engineering and Industrial Aerodynamics* 99.4 (2011), pp. 267–278. ISSN: 0167-6105. DOI: https://doi.org/10.1016/j.jweia.2010.12.017. URL: http://www.sciencedirect.com/science/article/pii/S016761051100002X.

[28] B. W. Yan et al. "RANS simulation of neutral atmospheric boundary layer flows over complex terrain by proper imposition of boundary conditions and modification on the k-ε model". In: *Environmental Fluid Mechanics* 16.1 (2016), pp. 1–23. ISSN: 15731510. DOI: 10.1007/s10652-015-9408-1.

[29] M Pontiggia et al. "Hazardous gas dispersion: A CFD model accounting for atmospheric stability classes". In: *Journal of hazardous materials* 171.1-3 (2009), pp. 739–747. ISSN: 1873-3336. DOI: 10.1016/j.jhazmat.2009.06.064. URL: http://www.sciencedirect.com/science/article/pii/S0304389409009844.

[30] P J Richards and S E Norris. "Appropriate boundary conditions for computational wind engineering models revisited". In: *Journal of Wind Engineering and Industrial Aerodynamics* 99.4 (2011), pp. 257–266. ISSN: 0167-6105. DOI: https://doi.org/10.1016/j.jweia.2010.12.008. URL: http://www.sciencedirect.com/science/article/pii/S0167610510001418.

[31] D.M. M Hargreaves and N.G. G Wright. "On the use of the k– model in commercial CFD software to model the neutral atmospheric boundary layer". In: *Journal of Wind Engineering and Industrial Aerodynamics* 95.5 (2007), pp. 355–369. ISSN: 01676105. DOI: 10.1016/j.jweia.2006.08.002. URL: http://www.sciencedirect.com/science/article/pii/S016761050600136X.

[32] M M Gibson and B E Launder. "Ground effects on pressure fluctuations in the atmospheric boundary layer". In: *Journal of Fluid Mechanics* 86.3 (1978), pp. 491–511. ISSN: 0022-1120. DOI: DOI:10.1017/S0022112078001251. URL: https://www.cambridge.org/core/article/ground-effects-on-pressure-fluctuations-in-the-atmospheric-boundary-layer/D5FFE2AB889A67F31D2EDDD08D2EB006.

[33] W P Jones and B E Launder. "The prediction of laminarization with a two-equation model of turbulence". In: *International Journal of Heat and Mass Transfer* 15.2 (1972), pp. 301–314. ISSN: 0017-9310. DOI: https://doi.org/10.1016/0017-9310(72)90076-2. URL: http://www.sciencedirect.com/science/article/pii/0017931072900762.

[34] B E Launder and D B Spalding. "The numerical computation of turbulent flows". In: *Computer Methods in Applied Mechanics and Engineering* 3.2 (1974), pp. 269–289. DOI: http://dx.doi.org/10.1016/0045-7825(74)90029-2. URL: http://www.sciencedirect.com/science/article/pii/0045782574900292.

[35] Georgi Kalitzin et al. "Near-wall behavior of RANS turbulence models and implications for wall functions". In: *Journal of Computational Physics* 204.1 (2005), pp. 265–291. ISSN: 0021-9991. DOI: https://doi.org/10.1016/j.jcp.2004.10.018. URL: http://www.sciencedirect.com/science/article/pii/S0021999104004164.

[36] J.-A. Bäckar and L Davidson. "Evaluation of numerical wall functions on the axisymmetric impinging jet using OpenFOAM". In: *International Journal of Heat and Fluid Flow* 67 (2017), pp. 27–42. ISSN: 0142-727X. DOI: https://doi.org/10.1016/j.ijheatfluidflow.2017.07.004. URL: http://www.sciencedirect.com/science/article/pii/S0142727X16308578.

[37] Bert Blocken, Ted Stathopoulos, and Jan Carmeliet. "CFD simulation of the atmospheric boundary layer: wall function problems". In: *Atmospheric Environment* 41.2 (2007), pp. 238–252. ISSN: 13522310. DOI: 10.1016/j.atmosenv.2006.08.019. URL: http://www.sciencedirect.com/science/article/pii/S135223100600834X.

[38] Chin-Hoh Moeng. "A Large-Eddy-Simulation Model for the Study of Planetary Boundary-Layer Turbulence". EN. In: *Journal of the Atmospheric Sciences* 41.13 (1984), pp. 2052–2062. ISSN: 0022-4928. DOI: 10.1175/1520-0469(1984)041<2052:ALESMF>2.0.CO;2. URL: http://journals.ametsoc.org.ezlibproxy1.ntu.edu.sg/doi/abs/10.1175/1520-0469{\%}281984{\%}29041{\%}3C2052{\%}3AALESMF{\%}3E2.0.CO{\%}3B2.

[39] Eileen M. Saiki, Chin-Hoh Moeng, and Peter P. Sullivan. "Large-Eddy Simulation Of The Stably Stratified Planetary Boundary Layer". en. In: *Boundary-Layer Meteorology* 95.1 (2000), pp. 1–30. ISSN: 1573-1472. DOI: 10.1023/A:1002428223156. URL: http://link.springer.com.ezlibproxy1.ntu.edu.sg/article/10.1023/A{\%}3A1002428223156.

[40] Ce'dric Alinot and Christian Masson. "k-ε Model for the Atmospheric Boundary Layer Under Various Thermal Stratifications". In: *Journal of Solar Energy Engineering* 127.4 (2005), p. 438. ISSN: 01996231. DOI: 10.1115/1.2035704. URL: https://solarenergyengineering.asmedigitalcollection.asme.org/article.aspx?articleID=1457484.

[41] M Pontiggia et al. "Hazardous gas dispersion: A CFD model accounting for atmospheric stability classes". In: *Journal of Hazardous Materials* 171.1-3 (2009), pp. 739–747. ISSN: 1873-3336. DOI: http://dx.doi.org/10.1016/j.jhazmat.2009.06.064. URL: http://www.sciencedirect.com/science/article/pii/S0304389409009844.

[42] Morten Nielsen and Søren Ott. "Heat transfer in large-scale heavy-gas dispersion". In: *Journal of Hazardous Materials* 67.1 (1999), pp. 41–58. ISSN: 03043894. DOI: 10.1016/S0304-3894(99)00016-3. URL: http://www.sciencedirect.com/science/article/pii/S0304389499000163.

[43] I V Kovalets and V S Maderich. "Numerical Simulation of Interaction of the Heavy Gas Cloud with the Atmospheric Surface Layer". In: *Environmental Fluid Mechanics* 6.4 (2006), p. 313. ISSN: 1573-1510. DOI: 10.1007/s10652-005-4288-4. URL: http://dx.doi.org/10.1007/s10652-005-4288-4.

[44] Federico Flores, René Garreaud, and Ricardo C Muñoz. "CFD simulations of turbulent buoyant atmospheric flows over complex geometry: Solver development in OpenFOAM". In: *Computers & Fluids* 82 (2013), pp. 1–13. ISSN: 0045-7930. DOI: http://dx.doi.org/10.1016/j.compfluid.2013.04.029. URL: http://www.sciencedirect.com/science/article/pii/S0045793013001795.

[45] Andrew Riddle et al. "Comparisons between FLUENT and ADMS for atmospheric dispersion modelling". In: *Atmospheric Environment* 38.7 (2004), pp. 1029–1038. ISSN: 13522310. DOI: 10.1016/j.atmosenv.2003.10.052. URL: http://www.sciencedirect.com/science/article/pii/S1352231003009713.

[46] M R Mokhtarzadeh-Dehghan, A Akcayoglu, and A G Robins. "Numerical study and comparison with experiment of dispersion of a heavier-than-air gas in a simulated neutral atmospheric boundary layer". In: *Journal of Wind Engineering and Industrial Aerodynamics* 110 (2012), pp. 10–24. ISSN: 0167-6105. DOI: https://doi.org/10.1016/j.jweia.2012.07.004. URL: http://www.sciencedirect.com/science/article/pii/S0167610512002085.

[47] Xiaobin Zhang et al. "Computational fluid dynamics study on liquefied natural gas dispersion with phase change of water". In: *International Journal of Heat and Mass Transfer* 91 (2015), pp. 347–354. ISSN: 00179310. DOI: 10.1016/j.ijheatmasstransfer.2015.07.117.

[48] Walter Chukwunonso Ikealumba and Hongwei Wu. "Modeling of Liquefied Natural Gas Release and Dispersion: Incorporating a Direct Computational Fluid Dynamics Simulation Method for LNG Spill and Pool Formation". EN. In: *Industrial & Engineering Chemistry Research* 55.6 (2016), pp. 1778–1787. ISSN: 0888-5885. DOI: 10.1021/acs.iecr.5b04490.

[49] Robert N. Meroney. "CFD modeling of dense gas cloud dispersion over irregular terrain". In: *Journal of Wind Engineering and Industrial Aerodynamics* 104-106 (2012), pp. 500–508. ISSN: 01676105. DOI: 10.1016/j.jweia.2012.01.001. URL: http://www.sciencedirect.com/science/article/pii/S0167610512000025.

[50] J. Labovský and L'. Jelemenský. "Verification of CFD pollution dispersion modelling based on experimental data". In: *Journal of Loss Prevention in the Process Industries* 24.2 (2011), pp. 166–177. ISSN: 09504230. DOI: 10.1016/j.jlp.2010.12.005. URL: http://www.sciencedirect.com/science/article/pii/S0950423010001579.

[51] Spyros Sklavounos and Fotis Rigas. "Validation of turbulence models in heavy gas dispersion over obstacles". In: *Journal of Hazardous Materials* 108.1–2 (2004), pp. 9–20. DOI: http://dx.doi.org/10.1016/j.jhazmat.2004.01.005.

[52] Steven R. Hanna, Olav R. Hansen, and Seshu Dharmavaram. "FLACS CFD air quality model performance evaluation with Kit Fox, MUST, Prairie Grass, and EMU observations". In: *Atmospheric Environment* 38.28 (2004), pp. 4675–4687. ISSN: 13522310. DOI: 10.1016/j.atmosenv.2004.05.041.

[53] Olav R. Hansen et al. "Validation of FLACS against experimental data sets from the model evaluation database for LNG vapor dispersion". In: *Journal of Loss Prevention in the Process Industries* 23.6 (2010), pp. 857–877. ISSN: 09504230. DOI: 10.1016/j.jlp.2010.08.005. URL: http://dx.doi.org/10.1016/j.jlp.2010.08.005.

[54] A. Mack and M. P N Spruijt. "Validation of OpenFoam for heavy gas dispersion applications". In: *Journal of Hazardous Materials* 262 (2013), pp. 504–516. ISSN: 03043894. DOI: 10.1016/j.jhazmat.2013.08.065.

[55] J Fiates et al. "An alternative CFD tool for gas dispersion modelling of heavy gas". In: *Journal of Loss Prevention in the Process Industries* 44 (2016), pp. 583–593. DOI: 10.1016/j.jlp.2016.08.002. URL: https://www.scopus.com/inward/record.uri?eid=2-s2.0-84994155203{\&}doi=10.1016{\%}2Fj.jlp.2016.08.002{\&}partnerID=40{\&}md5=f15971c9c9332b320e618d86adf80351.

[56] Juliane Fiates and Sávio S V Vianna. "Numerical modelling of gas dispersion using OpenFOAM". In: *Process Safety and Environmental Protection* 104, Part (2016), pp. 277–293. ISSN: 0957-5820. DOI: http://doi.org/10.1016/j.psep.2016.09.011. URL: http://www.sciencedirect.com/science/article/pii/S0957582016302105.

[57] Yvon Mouilleau and Anousone Champassith. "CFD simulations of atmospheric gas dispersion using the Fire Dynamics Simulator (FDS)". In: *Journal of Loss Prevention in the Process Industries* 22.3 (2009), pp. 316–323. ISSN: 0950-4230. DOI: https://doi.org/10.1016/j.jlp.2008.11.009. URL: http://www.sciencedirect.com/science/article/pii/S0950423008001496.

[58] Noah L. Ryder et al. "Consequence modeling using the fire dynamics simulator". In: *Journal of Hazardous Materials* 115.1 (2004), pp. 149–154. ISSN: 03043894. DOI: 10.1016/j.jhazmat.2004.06.018.

[59] S.E. Gant et al. "Evaluation of multi-phase atmospheric dispersion models for application to Carbon Capture and Storage". In: *Journal of Loss Prevention in the Process Industries* 32 (2014), pp. 286–298. ISSN: 09504230. DOI: 10.1016/j.jlp.2014.09.014. URL: http://www.sciencedirect.com/science/article/pii/S0950423014001570.

[60] S M Tauseef, D Rashtchian, and S A Abbasi. "CFD-based simulation of dense gas dispersion in presence of obstacles". In: *Journal of Loss Prevention in the Process Industries* 24 (2011), pp. 371–376. DOI: 10.1016/j.jlp.2011.01.014.

[61] G Xu et al. "CFD Simulation of Chemical Gas Dispersion under Atmospheric Boundary Conditions". In: *International Journal of Computational Methods* (2019). DOI: 10.1142/S0219876219400115. URL: https://www.scopus.com/inward/record.uri?eid=2-s2.0-85065489575{\&}doi=10.1142{\%}2FS0219876219400115{\&}partnerID=40{\&}md5=32f66587cf999f0a192065a02deabbb7.

[62] F Toja-Silva, C Pregel-Hoderlein, and J Chen. "On the urban geometry generalization for CFD simulation of gas dispersion from chimneys: Comparison with Gaussian plume model". In: *Journal of Wind Engineering and Industrial Aerodynamics* 177 (2018), pp. 1–18. DOI: 10.1016/j.jweia.2018.04.003. URL: https://www.scopus.com/inward/record.uri?eid=2-s2.0-85045441923{\&}doi=10.1016{\%}2Fj.jweia.2018.04.003{\&}partnerID=40{\&}md5=55da1cd28049725f47bfc5c4c7be5f6f.

[63] Francisco Toja-Silva et al. "CFD simulation of CO2 dispersion from urban thermal power plant: Analysis of turbulent Schmidt number and comparison with Gaussian plume model and measurements". In: *Journal of Wind Engineering and Industrial Aerodynamics* 169 (2017), pp. 177–193. ISSN: 0167-6105. DOI: https://doi.org/10.1016/j.jweia.2017.07.015. URL: http://www.sciencedirect.com/science/article/pii/S0167610517302258.

[64] J X Wen et al. "Further development and validation of CO 2 FOAM for the atmospheric dispersion of accidental releases from carbon dioxide pipelines". In: *International Journal of Greenhouse Gas Control* 52 (2016), pp. 293–304. DOI: 10.1016/j.ijggc.2016.07.006. URL: https://www.scopus.com/inward/record.uri?eid=2-s2.0-84978700542{\&}doi=10.1016{\%}2Fj.ijggc.2016.07.006{\&}partnerID=40{\&}md5=bdbd0aea1268f83feddf000ef64704f1.

[65] Henk Kaarle Versteeg and Weeratunge Malalasekera. *An introduction to computational fluid dynamics: the finite volume method*. Pearson education, 2007.

[66] H G Weller and G Tabor. "A tensorial approach to computational continuum mechanics using object-oriented techniques". In: *Computers in Physics* 12.6 (1998), pp. 620–631. ISSN: 08941866. DOI: 10.1063/1.168744.

[67] Joel H Ferziger and Milovan Peric. *Computational methods for fluid dynamics*. Springer Science & Business Media, 2012. ISBN: 3642560261.

[68] Edwin N. Lightfoot R. Byron Bird, Warren E. Stewart. *Transport phenomena*. 2002. ISBN: 0-471-41077-2.

[69] V. Busini. "Atmospheric Dispersion". In: *Reference Module in Chemistry, Molecular Sciences and Chemical Engineering*. 2016. ISBN: 9780124095472. DOI: 10.1016/B978-0-12-409547-2.11078-9.

[70] F R Menter, Matthias Kuntz, and R Langtry. "Ten Years of Industrial Experience with the SST Turbulence Model". In: 2003.

[71] Dalibor Cavar et al. "Comparison of OpenFOAM and EllipSys3D for neutral atmospheric flow over complex terrain". In: *Wind Energy Science Discussions* 1 (2016), pp. 55–70. ISSN: 2366-7621. DOI: 10.5194/wes-2016-3.

[72] Fadl Moukalled, Luca Mangani, and Marwan Darwish. "Implementation of boundary conditions in the finite-volume pressure-based method—Part I: Segregated solvers". In: *Numerical Heat Transfer, Part B: Fundamentals* 69.6 (2016), pp. 534–562. ISSN: 1040-7790. DOI: 10.1080/10407790.2016.1138748. URL: https://doi.org/10.1080/10407790.2016.1138748.

[73] Roland B Stull. *An introduction to boundary layer meteorology*. Vol. 13. Springer Science & Business Media, 2012.

[74] S G Giannissi et al. "Numerical simulation of LNG dispersion under two-phase release conditions". In: *Journal of Loss Prevention in the Process Industries* 26.1 (2013), pp. 245–254. DOI: http://dx.doi.org/10.1016/j.jlp.2012.11.010.

[75] Ian H Bell et al. "Pure and Pseudo-pure Fluid Thermophysical Property Evaluation and the Open-Source Thermophysical Property Library CoolProp". In: *Industrial & Engineering Chemistry Research* 53.6 (2014), pp. 2498–2508. ISSN: 0888-5885. DOI: 10.1021/ie4033999. URL: http://dx.doi.org/10.1021/ie4033999.

[76] Kevin McGrattan et al. "Fire dynamics simulator, user's guide". In: *NIST special publication* 1019 (2013), p. 20.

Appendix A

OpenFOAM code implementation

A.1 Momentum equation

Momentum equation solved in OpenFOAM is:

$$\frac{\partial(\rho \boldsymbol{u})}{\partial t} + \boldsymbol{\nabla} \cdot (\rho \boldsymbol{u}\boldsymbol{u}) = -\boldsymbol{\nabla} p + \rho \boldsymbol{g} + \boldsymbol{\nabla} \cdot \left(2\mu_{eff} D(\boldsymbol{u})\right) - \boldsymbol{\nabla} \left(\frac{2}{3}\mu_{eff}(\boldsymbol{\nabla} \cdot \boldsymbol{u})\right) \tag{A.1}$$

The rate of strain tensor: $D(\boldsymbol{u}) = \frac{1}{2}\left(\nabla \boldsymbol{u} + (\nabla \boldsymbol{u})^T\right)$. OpenFOAM implementation takes into account the effect of turbulence. Therefore,$\mu_{eff} = \mu + \mu_t$ is the sum of molecular and turbulent viscosity.

Code implementation of momentum equation in OpenFOAM can be presented in Listing A.1.

Listing A.1 Momemtum equation code

```
1    MRF.correctBoundaryVelocity(U);
2    fvVectorMatrix UEqn
3    (
4        fvm::ddt(rho, U) + fvm::div(phi, U)
5      + MRF.DDt(rho, U)
```

```
      + turbulence->divDevRhoReff(U)
     ==
        fvOptions(rho, U)
    );
    UEqn.relax();
    fvOptions.constrain(UEqn);
    if (pimple.momentumPredictor())
    {
        solve
        (
            UEqn
         ==
            fvc::reconstruct
            (
                (
                  - ghf*fvc::snGrad(rho)
                  - fvc::snGrad(p_rgh)
                )*mesh.magSf()
            )
        );
        fvOptions.correct(U);
        K = 0.5*magSqr(U);
    }
```

`rhoEqn` is used to solve the continuity for density:

Listing A.2 `rhoEqn.H`

```
fvScalarMatrix rhoEqn
(
    fvm::ddt(rho)
  + fvc::div(phi)
  ==
    fvOptions(rho)
);
fvOptions.constrain(rhoEqn);
rhoEqn.solve();
fvOptions.correct(rho);
```

The pressure Poisson equation `pEqn` implemented in OpenFOAM is shown in Listing A.3.

Listing A.3 Pressure poisson equation implementation in OpenFOAM

```
rho = thermo.rho();
// Thermodynamic density needs to be updated by psi*d(p) after the
// pressure solution
const volScalarField psip0(psi*p);
volScalarField rAU(1.0/UEqn.A());
surfaceScalarField rhorAUf("rhorAUf", fvc::interpolate(rho*rAU));
volVectorField HbyA(constrainHbyA(rAU*UEqn.H(), U, p));
surfaceScalarField phig(-rhorAUf*ghf*fvc::snGrad(rho)*mesh.magSf());
surfaceScalarField phiHbyA
(
    "phiHbyA",
    (
        fvc::flux(rho*HbyA)
      + rhorAUf*fvc::ddtCorr(rho, U, phi)
    )
  + phig
);
MRF.makeRelative(fvc::interpolate(rho), phiHbyA);
// Update the pressure BCs to ensure flux consistency
constrainPressure(p_rgh, rho, U, phiHbyA, rhorAUf, MRF);
fvScalarMatrix p_rghDDtEqn
(
    fvc::ddt(rho) + psi*correction(fvm::ddt(p_rgh))
  + fvc::div(phiHbyA)
  ==
    fvOptions(psi, p_rgh, rho.name())
);
while (pimple.correctNonOrthogonal())
{
    fvScalarMatrix p_rghEqn
    (
        p_rghDDtEqn
      - fvm::laplacian(rhorAUf, p_rgh)
    );

    p_rghEqn.solve(mesh.solver(p_rgh.select(pimple.finalInnerIter())));

    if (pimple.finalNonOrthogonalIter())
```

```
    {
        // Calculate the conservative fluxes
        phi = phiHbyA + p_rghEqn.flux();

        // Explicitly relax pressure for momentum corrector
        p_rgh.relax();

        // Correct the momentum source with the pressure gradient flux
        // calculated from the relaxed pressure
        U = HbyA + rAU*fvc::reconstruct((phig + p_rghEqn.flux())/rhorAUf);
        U.correctBoundaryConditions();
        fvOptions.correct(U);
        K = 0.5*magSqr(U);
    }
}
p = p_rgh + rho*gh;
// Thermodynamic density update
thermo.correctRho(psi*p - psip0);
if (thermo.dpdt())
{
    dpdt = fvc::ddt(p);
}
#include "rhoEqn.H"
#include "compressibleContinuityErrs.H"
```

A.2 Species transport equation

The code fragments implementation of Equation (3.5) is shown in Listing A.4:

Listing A.4 Species transport equation

```
{
    reaction->correct();
    Qdot = reaction->Qdot();
    volScalarField Yt(0.0*Y[0]);
    forAll(Y, i)
    {
        if (i != inertIndex && composition.active(i))
        {
```

```
            volScalarField& Yi = Y[i];

            fvScalarMatrix YiEqn
            (
                fvm::ddt(rho, Yi)
              + mvConvection->fvmDiv(phi, Yi)
              - fvm::laplacian(turbulence->muEff(), Yi)
             ==
                reaction->R(Yi)
              + fvOptions(rho, Yi)
            );
            YiEqn.relax();
            fvOptions.constrain(YiEqn);
            YiEqn.solve(mesh.solver("Yi"));
            fvOptions.correct(Yi);
            Yi.max(0.0);
            Yt += Yi;
        }
    }
    Y[inertIndex] = scalar(1) - Yt;
    Y[inertIndex].max(0.0);
}
```

YEqn is modified with user-defined Sc_t:

Listing A.5 YEqn.H

```
{
    // Reading user defined Sc number
    scalar Sct = runTime.controlDict().lookupOrDefault<scalar>("Sc", 1.0);
    reaction->correct();
    Qdot = reaction->Qdot();
    volScalarField Yt(0.0*Y[0]);
    forAll(Y, i)
    {
        if (i != inertIndex && composition.active(i))
        {
            volScalarField& Yi = Y[i];
            fvScalarMatrix YiEqn
            (
```

```
                fvm::ddt(rho, Yi)
              + mvConvection->fvmDiv(phi, Yi)
              - fvm::laplacian(turbulence->muEff()/Sct, Yi)
             ==
                reaction->R(Yi)
              + fvOptions(rho, Yi)
            );
            YiEqn.relax();
            fvOptions.constrain(YiEqn);
            YiEqn.solve(mesh.solver("Yi"));
            fvOptions.correct(Yi);
            Yi.max(0.0);
            Yt += Yi;
        }
    }
    Y[inertIndex] = scalar(1) - Yt;
    Y[inertIndex].max(0.0);
}
```

A.3 Energy equation

In OpenFOAM, either internal energy or enthalpy can be used as the energy variable. The energy conservation equations for enthalpy per unit mass variable h, which is the sum of internal energy and kinematic pressure $h \equiv e + p/\rho$, can be written as:

$$\frac{\partial(\rho h)}{\partial t} + \boldsymbol{\nabla}\cdot(\rho \boldsymbol{u} h) + \frac{\partial(\rho K)}{\partial t} + \boldsymbol{\nabla}\cdot(\rho \boldsymbol{u} K) - \frac{\partial p}{\partial t} = \boldsymbol{\nabla}\cdot\left(\alpha_{eff}\nabla h\right) + \rho \boldsymbol{u}\cdot\boldsymbol{g} \tag{A.2}$$

In the above equation, $K \equiv |\boldsymbol{u}|^2/2$ is kinetic energy per unit mass, the pressure-work term dp/dt can be excluded by user option, the effective thermal diffusivity α_{eff} is the sum of laminar and turbulent thermal diffusivity:

$$\alpha_{eff} = \frac{\rho \nu_t}{Pr_t} + \frac{\mu}{Pr} = \frac{\rho \nu_t}{Pr_t} + \frac{k}{c_p} \tag{A.3}$$

OpenFOAM implementation of energy equation using enthalpy h variable is presented in Listing A.6.

Listing A.6 OpenFOAM implementation of Energy equation

```
volScalarField& he = thermo.he();
fvScalarMatrix EEqn
(
    fvm::ddt(rho, he) + mvConvection->fvmDiv(phi, he)
  + fvc::ddt(rho, K) + fvc::div(phi, K)
  + (
        he.name() == "e"
      ? fvc::div
        (
            fvc::absolute(phi/fvc::interpolate(rho), U),
            p,
            "div(phiv,p)"
        )
      : -dpdt
    )
  - fvm::laplacian(turbulence->alphaEff(), he)
 ==
    Qdot
  + fvOptions(rho, he)
);
EEqn.relax();
EEqn.solve();
thermo.correct();
Info<< "min/max(T) = "
    << min(T).value() << ", " << max(T).value() << endl;
```

Calculate the temperature from the sensible enthalpy h can be done from `thermo.correct()` after solving the energy equation (Listing A.6).

A.4 FVM

The Gauss integration of scalar field ϕ convection term due to the velocity field can be written as:

$$\nabla \cdot (\boldsymbol{u}\phi) = \frac{1}{V}\int_V (\nabla \cdot \boldsymbol{u}\phi)\, dV = \frac{1}{V}\oint_S (\boldsymbol{n} \cdot \boldsymbol{u}\phi)\, dS = \frac{1}{V}\left(\sum_{\text{owner}} \boldsymbol{S} \cdot \boldsymbol{u}_f\phi - \sum_{\text{neighbor}} \boldsymbol{S} \cdot \boldsymbol{u}_f\phi\right) \tag{A.4}$$

An example of Gradient scheme implementation is listed below:

Listing A.7 Discretisation of diffusion term

```
Foam::fv::gaussGrad<Type>::gradf
(
    const GeometricField<Type, fvsPatchField, surfaceMesh>& ssf,
    const word& name
)
{
    typedef typename outerProduct<vector, Type>::type GradType;
    const fvMesh& mesh = ssf.mesh();
    tmp<GeometricField<GradType, fvPatchField, volMesh>> tgGrad;
    GeometricField<GradType, fvPatchField, volMesh>& gGrad=tgGrad.ref();
    const labelUList& owner = mesh.owner();
    const labelUList& neighbour = mesh.neighbour();
    const vectorField& Sf = mesh.Sf();
    Field<GradType>& igGrad = gGrad;
    const Field<Type>& issf = ssf;

    forAll(owner, facei)
    {
        GradType Sfssf = Sf[facei]*issf[facei];
        igGrad[owner[facei]] += Sfssf;
        igGrad[neighbour[facei]] -= Sfssf;
    }
    igGrad /= mesh.V();

    gGrad.correctBoundaryConditions();
    ...
}
```

An example of selection and implementation for convection term of field Q is seen in Listing A.8.

Listing A.8 Selection and divergence scheme implimentation

```
divSchemes
{
    default         none;
    div(phi,Q)      Gauss <interpolation scheme>;
}

// Calculation of matrix coefficient from divergence scheme selection
fvm.lower() = -weights.internalField()*faceFlux.internalField();
fvm.upper() = fvm.lower() + faceFlux.internalField();
fvm.negSumDiag();
```

A.5 Turbulent model

An example of model coefficients setting, and code for solving turbulence using standard $k-\varepsilon$ model are listed in Listing A.9.

Listing A.9 k, ε transport equation implementation in OpenFOAM

```
// Local references
const alphaField& alpha = this->alpha_;
const rhoField& rho = this->rho_;
const surfaceScalarField& alphaRhoPhi = this->alphaRhoPhi_;
const volVectorField& U = this->U_;
volScalarField& nut = this->nut_;

volScalarField::Internal divU
(
    fvc::div(fvc::absolute(this->phi(), U))().v()
);
tmp<volTensorField> tgradU = fvc::grad(U);
volScalarField::Internal G
(
    this->GName(),
```

```
    nut.v()*(dev(twoSymm(tgradU().v())) && tgradU().v())
);
tgradU.clear();
// Update epsilon and G at the wall
epsilon_.boundaryFieldRef().updateCoeffs();
// Dissipation equation
tmp<fvScalarMatrix> epsEqn
(
    fvm::ddt(alpha, rho, epsilon_)
  + fvm::div(alphaRhoPhi, epsilon_)
  - fvm::laplacian(alpha*rho*DepsilonEff(), epsilon_)
 ==
    C1_*alpha()*rho()*G*epsilon_()/k_()
  - fvm::SuSp(((2.0/3.0)*C1_ - C3_)*alpha()*rho()*divU, epsilon_)
  - fvm::Sp(C2_*alpha()*rho()*epsilon_()/k_(), epsilon_)
  + epsilonSource()
  + fvOptions(alpha, rho, epsilon_)
);
epsEqn.ref().relax();
epsEqn.ref().boundaryManipulate(epsilon_.boundaryFieldRef());
solve(epsEqn);
bound(epsilon_, this->epsilonMin_);
// Turbulent kinetic energy equation
tmp<fvScalarMatrix> kEqn
(
    fvm::ddt(alpha, rho, k_)
  + fvm::div(alphaRhoPhi, k_)
  - fvm::laplacian(alpha*rho*DkEff(), k_)
 ==
    alpha()*rho()*G
  - fvm::SuSp((2.0/3.0)*alpha()*rho()*divU, k_)
  - fvm::Sp(alpha()*rho()*epsilon_()/k_(), k_)
  + kSource()
  + fvOptions(alpha, rho, k_)
);
kEqn.ref().relax();
solve(kEqn);
bound(k_, this->kMin_);
correctNut();
```

The inclusion of the buoyancy term in standard $k-\varepsilon$ is implemented in OpenFOAM as in Listing A.10.

Listing A.10 Implementation of including buoyancy source term in OpenFOAM

```
template<class BasicTurbulenceModel>
tmp<volScalarField>
buoyantKEpsilon<BasicTurbulenceModel>::Gcoef() const
{
    const uniformDimensionedVectorField& g =
        this->mesh_.objectRegistry::template
        lookupObject<uniformDimensionedVectorField>("g");
    return
        (Cg_*this->Cmu_)*this->alpha_*this->k_*(g & fvc::grad(this->rho_))
       /(this->epsilon_ + this->epsilonMin_);
}

template<class BasicTurbulenceModel>
tmp<fvScalarMatrix>
buoyantKEpsilon<BasicTurbulenceModel>::kSource() const
{
    const uniformDimensionedVectorField& g =
        this->mesh_.objectRegistry::template
        lookupObject<uniformDimensionedVectorField>("g");
    if (mag(g.value()) > SMALL)
    {
        return -fvm::SuSp(Gcoef(), this->k_);
    }
    else
    {
        return kEpsilon<BasicTurbulenceModel>::kSource();
    }
}

template<class BasicTurbulenceModel>
tmp<fvScalarMatrix>
buoyantKEpsilon<BasicTurbulenceModel>::epsilonSource() const
{
    const uniformDimensionedVectorField& g =
        this->mesh_.objectRegistry::template
```

```
        lookupObject<uniformDimensionedVectorField>("g");
    if (mag(g.value()) > SMALL)
    {
        vector gHat(g.value()/mag(g.value()));
        volScalarField v(gHat & this->U_);
        volScalarField u
        (
            mag(this->U_ - gHat*v)
          + dimensionedScalar("SMALL", dimVelocity, SMALL)
        );
        return -fvm::SuSp(this->C1_*tanh(mag(v)/u)*Gcoef(), this->epsilon_);
    }
    else
    {
        return kEpsilon<BasicTurbulenceModel>::epsilonSource();
    }
}
```

A.6 ablBuoyantSimpleFoam

Listing A.11 ablBuoyantSimpleFoam main code

```
#include "fvCFD.H"
#include "rhoThermo.H"
#include "turbulentFluidThermoModel.H"
#include "radiationModel.H"
#include "simpleControl.H"
#include "fvOptions.H"
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //
int main(int argc, char *argv[])
{
    #include "postProcess.H"
    #include "setRootCase.H"
    #include "createTime.H"
    #include "createMesh.H"
    #include "createControl.H"
    #include "createFields.H"
    #include "createFieldRefs.H"
```

```
    #include "createFvOptions.H"
    #include "initContinuityErrs.H"
    turbulence->validate();
    // * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //
    Info<< "\nStarting time loop\n" << endl;
    while (simple.loop())
    {
        Info<< "Time = " << runTime.timeName() << nl << endl;
        // Pressure-velocity SIMPLE corrector
        {
            #include "UEqn.H"
            #include "EEqn.H"
            #include "pEqn.H"
        }
        turbulence->correct();
        runTime.write();
        Info<< "ExecutionTime = " << runTime.elapsedCpuTime() << " s"
            << "  ClockTime = " << runTime.elapsedClockTime() << " s"
            << nl << endl;
    }
    Info<< "End\n" << endl;
    return 0;
}
```

A.7 gasDispersionBuoyantFoam

Listing A.12 gasDispersionBuoyantFoam main code

```
#include "postProcess.H"
#include "setRootCase.H"
#include "createTime.H"
#include "createMesh.H"
#include "createControl.H"
#include "createTimeControls.H"
#include "initContinuityErrs.H"
#include "createFields.H"
#include "createFieldRefs.H"
#include "createFvOptions.H"
turbulence->validate();
```

```
while (runTime.run())
{
    #include "readTimeControls.H"
    #include "compressibleCourantNo.H"
    #include "setDeltaT.H"
    runTime++;
    Info<< "Time = " << runTime.timeName() << nl << endl;
    #include "rhoEqn.H"
    // Pressure-velocity PIMPLE corrector loop
    while (pimple.loop())
    {
        #include "UEqn.H"
        #include "YEqn.H"
        #include "EEqn.H"
        // Pressure corrector loop
        while (pimple.correct())
        {
            #include "pEqn.H"
        }
        if (pimple.turbCorr())
        {
            turbulence->correct();
        }
    }
    rho = thermo.rho();
    runTime.write();
    Info<< "ExecutionTime = " << runTime.elapsedCpuTime() << " s"
        << "  ClockTime = " << runTime.elapsedClockTime() << " s"
        << nl << endl;
}
Info<< "End\n" << endl;
return 0;
```