

On numerical modelling of atmospheric gas dispersion

Using computational fluid dynamics approach



Tran Le Vu

School of Mechanical and Aerospace Engineering

Nanyang Technological University

This dissertation is submitted for the degree of

Doctor of Philosophy

College of engineering

August 2018

Abstract

Mostly all human activities are affected by Atmospheric boundary layer (ABL). This is also where most air pollution phenomena are occurred. Understanding of the processes taking place in the ABL has attracted various research study.

Computational Fluid Dynamics (CFD) is increasingly being used in simulation of 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. The RANS turbulence models are still widely used in practical approach to overcome boundary conditions sensitivity and computational intensive of the LES. Two equation turbulence models: standard $k - \epsilon$ and SST $k - \omega$ turbulence models are modified substantially using open source CFD tool OpenFOAM to validate its usage in simulating ABL flow.

Monin-Obukhov similarity theory, well validated for flow in ABL surface layer over homogeneous surface, is used to model the profiles of velocity, turbulent kinetic energy and turbulence dissipation rate at inlet of computational domain. Consistency of these profiles across the domain are ensured by deriving the relation between turbulence model constants for horizontally homogeneous constant shear stress flow. This verified turbulence model is then validated using simulated atmospheric dispersion flow of dense gas and field experiments of Liquefied Natural Gas (LNG) vapour dispersion. For simulation of LNG vapour dispersion, the proposed model also takes into accounts the heat transfer from the ground to the vapour

cloud, the effect of variable temperature on gas properties. Statistical Performance Measures (SPM) are compared with LES code FDS (Fire Dynamics Simulator) and specified dispersion code FLACS (FLame ACceleration Simulator) under Model Evaluation Protocol (MEP) proposed for LNG vapour dispersion.

Table of contents

List of figures	ix
List of tables	xiii
Nomenclature	xv
1 Introduction	1
1.1 Motivation	1
1.2 Atmospheric boundary layer (ABL)	2
1.2.1 Monin–Obukhov similarity theory	3
1.2.2 Mixed-layer similarity	5
1.3 Dense Gas Dispersion	6
1.4 Model evaluation	8
1.4.1 Validation data sets	9
1.4.2 Statistical Performance Measure (SPM)	14
1.5 Research objectives and scopes	14
2 Literature Review	17
2.1 Neutral Atmospheric Boundary Layer modelling	17
2.1.1 Horizontally homogeneous boundary layer	17
2.1.2 Boundary conditions	21

2.2	Stratified Atmospheric Boundary Layer modelling	21
2.3	CFD simulation of dense gas dispersion	25
2.4	Concluding remarks	25
3	OpenFOAM Methodology	27
3.1	Governing equations	27
3.1.1	Momentum equation	27
3.1.2	Pressure equation	29
3.1.3	Momentum-pressure coupling algorithms	33
3.1.4	Species transport equation	35
3.1.5	Energy equation	36
3.2	Numerical schemes	39
3.2.1	Time discretisation term	39
3.2.2	Convective term	39
3.2.3	Diffusion term	40
3.3	Linear solvers	41
3.3.1	Solution tolerances	42
3.3.2	Preconditioner	42
3.3.3	Generalised Geometric-Algebraic Multi-Grid solver (GAMG) . . .	43
3.4	Boundary conditions	44
3.5	Thermophysical models	45
3.6	Turbulence models	46
3.6.1	Reynolds-Averaged Navier-Stokes (RANS)	46
3.6.2	Reynold stress transport models	58
3.7	The solvers	60
4	Modelling of Atmospheric Boundary Layer	65

4.1	Full scale simulation of the neutral ABL	65
4.1.1	Domain and mesh	65
4.1.2	Boundary conditions	65
4.1.3	Numerical setting	66
4.1.4	Results and discussion of neutral ABL simulations	67
4.2	Simulations of neutral ABL in presence of obstacles	69
4.2.1	CEDVAL wind tunnel data	69
4.2.2	Domain and mesh generation	70
4.2.3	Boundary conditions	71
4.2.4	Numerical setting	72
4.2.5	Results and discussion of neutral ABL in presence of obstacle simulations	72
4.3	Full scale simulation of the stable stratified ABL	73
4.3.1	Boundary conditions	73
4.3.2	Numerical setting	73
4.3.3	Results and discussion of stable stratified ABL simulation	75
5	Atmospheric boundary layer gas dispersion	79
5.1	Dense gas dispersion in wind tunnel tests	80
5.1.1	Hamburg wind tunnel test	80
5.1.2	Numerical setting	81
5.1.3	Results and discussion of gas dispersion in wind tunnel tests	82
5.2	Dense gas dispersion over flat terrain	86
5.2.1	Domain and grid generation	86
5.2.2	Boundary conditions	88
5.2.3	Thermophysical models	90
5.2.4	Numerical setting	91

5.2.5	Results and discussion of gas dispersion over flat terrain	92
5.3	Dense gas dispersion over obstructed terrain	103
5.3.1	Boundary conditions	104
5.3.2	Results and discussion of gas dispersion over obstructed terrain . .	105
6	Conclusion and future works	107
6.1	Conclusion	107
6.2	Contributions	108
6.3	Limitations and Future works	109

List of figures

1.1	(a) Mean wind speed during Burro8 at station T2 (57, 0, 1). (b) Horizontal concentration contour at 1 m above ground level of Burro8 at 200 s. (c) Vertical concentration contours at 400 m downwind at the time of 400 s of Burro8 test	8
1.2	Burro Test Site [21]	11
1.3	Distance to LFL at 1 m height of Burro tests [21]	12
4.1	Velocity, turbulent kinetic energy and turbulent dissipation rate profiles from simulation of neutral ABL using standard $k - \varepsilon$ and modified $k - \varepsilon$ turbulence model	68
4.2	Velocity, turbulent kinetic energy and turbulent dissipation rate profiles from simulation of neutral ABL using standard $k - \omega$ and SST $k - \omega$ turbulence model	68
4.3	Velocity, turbulent kinetic energy and turbulent dissipation rate profiles from simulations of different kinetic energy levels by varying $C_\mu = 0.09$ and $C_\mu = 0.017$	69
4.4	Obstacle geometry of CEDVAL A1-1	70
4.5	The mesh using for CEDVAL A1 - 1 simulation	71
4.6	Profiles of U , k and ε of CEDVAL A1 - 1 test	73

4.7	Contour plot of velocity on planes $y = 0$, from top to bottom: $k - \varepsilon$ model, Experiment data, $k - \omega$ model	76
4.8	Velocity, turbulent kinetic energy and turbulent dissipation rate profiles in Stable ABL	77
5.1	DA0120 contour of gas concentration	83
5.2	DAT223 contour of gas concentration	83
5.3	Peak concentration DA0120	84
5.4	Peak concentration DAT223	85
5.5	Concentration at $X = 1.84$ of DA0120	85
5.6	Position of gas instruments (EXP) and their transformed points in simulation domain (FOAM) plotted for Burro9 test	87
5.7	The mesh using for Burro9 test	88
5.8	ABL profiles of Burro9	92
5.9	Burro9 mesh sensitivity study	93
5.10	Burro9 ground heat transfer study	94
5.11	Burro9 turbulent Schmidt number Sc_t study	95
5.12	Maximum arc-wise concentration Burro9	95
5.13	Maximum arc-wise concentration Burro3	96
5.14	Maximum arc-wise concentration Burro7	97
5.15	Maximum arc-wise concentration Burro8	97
5.16	Vertical isosurface at $X = 140$, Left: Simulation, Right: Experimental data, From top to bottom: Burro3, Burro7, Burro8 and Burro9	99
5.17	Horizontal isosurface at $Z = 1$; Left: Simulation, Right: Experimental data; From top to bottom:(a) Burro3 (b) Burro7 (c) Burro8 (d) Burro9	100
5.18	Point concentration at 140 m of Burro9	101
5.19	Point concentration at 140 m of Burro3	102

5.20	Point concentration at 140 m of Burro7	102
5.21	Point concentration at 57 m of Burro8	103
5.22	Maximum arc-wise concentration (a) Falcon1 (b) Falcon3 (c) Falcon4 .	105

List of tables

1.1	Burro tests summary extracted from [21]	11
1.2	Validation data set [10]	13
1.3	Statistical Performance Measures [10]	14
2.1	The modified $k - \varepsilon$ model constants by Yang et al. [24]	19
2.2	The standard $k - \varepsilon$ model constants proposed by Alinot and Masson [38] . .	23
3.1	The standard $k - \varepsilon$ model constants	48
3.2	The $k - \omega$ model constants	53
3.3	The LRR mean rate of strain term constants	59
3.4	The LRR model constants	59
4.1	Boundary conditions for neutral ABL simulation	66
4.2	ABL parameters using for neutral ABL simulation	66
4.3	Turbulence models setting for neutral ABL simulation	67
4.4	CEDVAL A1 - 1 ABL parameters	72
4.5	Meteorology parameters used for stable ABL simulation	73
4.6	Boundary conditions for stable stratified ABL simulation	74
4.7	Stable ABL simulations cases	75
5.1	Hamburg flat, unobstructed test case parameters	81
5.2	Turbulent Schmidt number Sc_t in Hamburg tests	82

5.3	Statistical performance measures of Hamburg unobstructed tests	86
5.4	Burro test computational domain and mesh parameters	88
5.5	Burro tests meteorological parameters	89
5.6	Burro test spill conditions	90
5.7	Wall thermal boundary conditions in Burro tests	90
5.8	Coefficients (Eq. (3.34)) of gas thermophysical properties used in Burro tests simulation	91
5.9	Statistical performance measures of Burro tests	103
5.10	Falcon tests meteorological parameters	104
5.11	Falcon test spill conditions	104
5.12	Statistical performance measures of Falcon tests	106

Nomenclature

Acronyms

Symbol	Description
ABL	Atmospheric Boundary Layer
FAC2	Factor of 2
FDS	Fire Dynamics Simulator
FLACS	FLame ACceleration Simulator
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	Prandtl 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	m^2/s^3
ε^+	Near wall scale of turbulence dissipation rate	
κ	von Karman constant	
μ	Dynamic viscosity	kg/ms
ν	Kinematic viscosity	m^2/s
ν^+	Near wall scale of kinematic viscosity	

ω	Turbulence specific dissipation rate	1/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
δ	Kronecker symbol	

Roman Symbols

Symbol	Description	Units
\mathbf{g}	Gravitational acceleration vector	m/s
\mathbf{u}	Velocity vector	m/s
A	matrix of coefficients	
C_m	Experimental measured concentration	
C_p	Predicted concentration from simulation	
c_p	Specific heat	J/kgK
D	Mass diffusivity	m ² /s

E	Smooth wall constant	
h	Enthalpy per unit mass	J/kg
h_{ABL}	Height of ABL	m
k	Turbulence kinetic energy	m ² /s ²
k^+	Near wall scale of turbulence kinetic energy	
L_{MO}	Monin-Obukhov length	m
M	Specie molecular weight	kmol/kg
p	Fluid pressure	N/m ²
p'	Numerical pressure correction	N/m ²
p_{rgh}	Pressure defined without hydrostatic pressure	N/m ²
q_s	Surface heat flux	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	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	
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 are occurred. Understanding of the processes taking place in the ABL has attracted various research study.

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 are occurred 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.

ABL is an important factor affecting the dispersion process by the effect of wind speed, surface roughness and atmospheric stability. Higher wind speed advects the cloud more rapidly, produces atmospheric turbulence to increase mixing of the cloud. Surface roughness determines the relation of advection and dilution 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.

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 the height z , 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. κ is the von Karman constant.

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

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:

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

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

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 JA et al. [5] used Dyer-Businger equation to derive the relationship between universal function of heat and momentum. They proposed that:

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

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 JA et al. [5] with $\kappa = 0.4$ and $Pr_t^{-1} = 1.05$:

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

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 is ranged 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 to 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 parametrization 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} (Equation (1.11)) can be estimated as [7]:

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

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

$$f_c = 2\omega_E \sin \Phi_E \quad (1.12)$$

1.3 Dense Gas Dispersion

Dense gas has relative density larger than 1.15 with respect to air at ambient temperature. Dense gas may 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 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 1.1a) 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 1.1b). 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 dominate 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 1.1c 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.

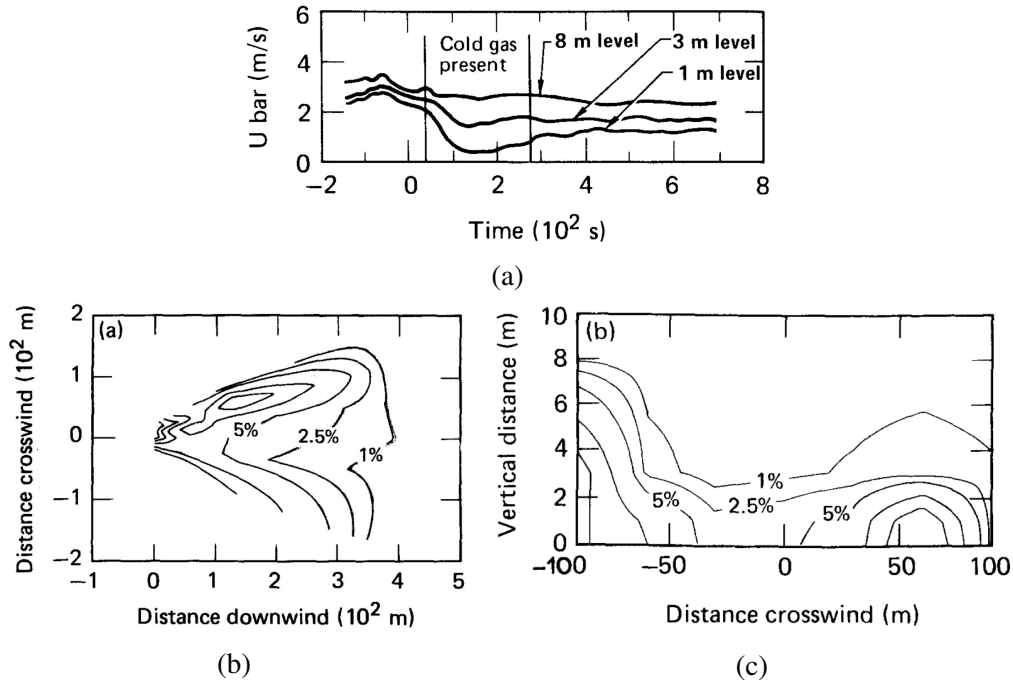


Fig. 1.1 (a) Mean wind speed during Burro8 at station T2 (57, 0, 1). (b) Horizontal concentration contour at 1 m above ground level of Burro8 at 200 s. (c) Vertical concentration contours at 400 m downwind at the time of 400 s of Burro8 test

1.4 Model evaluation

In 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 configuration 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, 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 1.2. 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 1.3, 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.

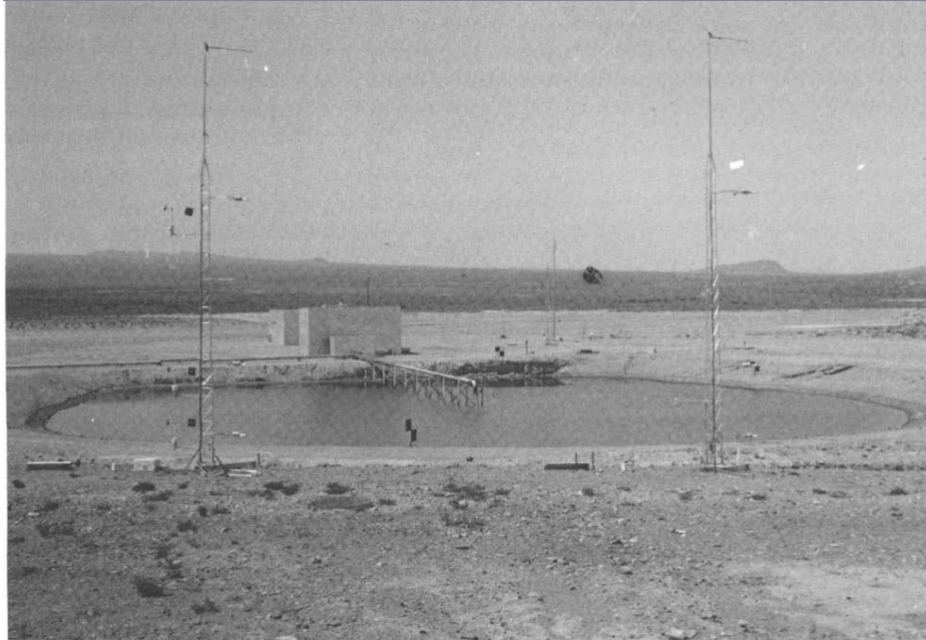


Fig. 1.2 Burro Test Site [21]

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 (ms ⁻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 (ms ⁻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}

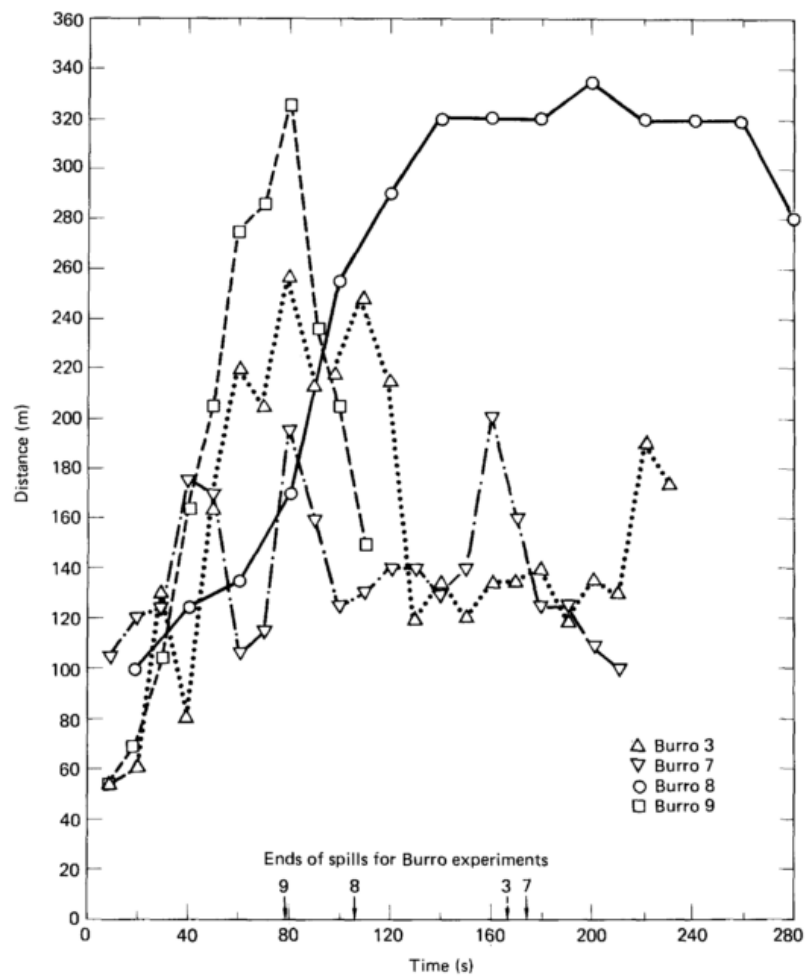


Fig. 1.3 Distance to LFL at 1 m height of Burro tests [21]

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 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 (1980)	Sand 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 (MRSE), 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, \bar{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	$\exp \left(\ln \frac{C_m}{C_p} \right)$	$0.67 < MG < 1.5$
VG	$\exp \left(\left(\ln \frac{C_m}{C_p} \right)^2 \right)$	$VG < 3.3$

1.5 Research objectives and scopes

The thesis is on developing a tool using framework of OpenFOAM to simulate gas dispersion in ABL. The developed model takes into account different atmospheric stability to solve ABL turbulence. RANS turbulence models are used. The dispersion model takes into account the effect 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 flows in ABL is obtaining equilibrium ABL, i.e. zero stream-wise gradients of all variables. For neutral atmospheric boundary layer, Richards and Hoxey [23] proposed the appropriate boundary conditions of mean wind speed and turbulence quantities for the standard $k - \varepsilon$ model. These profiles were derived assuming constant shear stress with height and applied for surface layer of ABL. These 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 [24], 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 also numerical schemes was shown to help to achieve

HHTSL [25, 26, 27]. Under HHTSL, the governing equations can be simplified as:

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

The inlet boundary conditions proposed by Richards and Hoxey [23], widely used in CFD study of atmospheric flow are:

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

These profiles are assured a solution of Equation (2.1), if the model constant, turbulent Prandtl number of the dissipation rate σ_ε , is modified as:

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

Instead of altering model constants, Pontiggia et al. [28] derived the ε equation z -dependent source term from solution of Equation (2.2):

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

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

Constant inlet turbulence kinetic energy proposed by Richards and Hoxey [23] is subjected to many arguments. Since velocity field is limited affected by turbulence kinetic energy

but the concentration field because of enhancing dispersion effect of turbulence. As noted by Parente et al. [26], decreasing k with height was shown in many wind tunnel test. Yang et al. [24] proposed new profile of k and ε for standard $k - \varepsilon$ model. k , ε are the non-linear function of height as:

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

$C_1 = -0.17$ 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. [24]

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

Parente et al. [26] presented an elaborate procedure to ensure the consistency for arbitrary inlet profile of turbulent kinetic energy k . Instead of altering model constants as Yang et al. [24], the effect of non-constant k on momentum and ε equation can be characterised by deriving equation for C_μ :

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

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

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

Richards and Norris [29] revisited the problem of modelling the HHTSL by deriving the inlet profiles directly from the conservation and equilibrium equations. This allows various inlet profiles can be specified by varying the turbulence models constants. 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 model constants as:

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

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

Hargreaves and Wright [30] had shown that zero gradient velocity at the top boundary resulted in a decay of velocity downstream, due to the extraction energy at wall due to wall shear stress. A driving shear stress, zero flux of turbulent kinetic energy and a flux of dissipation rate ε are to be imposed at the upper boundary:

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

For $k - \omega$ models, the specific dissipation ω is solved instead of dissipation rate ε . Profiles of U and k are the same, except the new effective von Karman constant is $\kappa_{k-\omega} = 0.408$. Profiles for ω has expression of:

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

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

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

In present of obstacles, Richards and Norris [29] had shown that eddy viscosity models like $k - \varepsilon$ or $k - \omega$ resulted the over-prediction of stagnation pressures, while Reynolds stress model (RSM) [31] was 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. The placement of this boundary, therefore, is very important. As 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 so far, the computational time will increase 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 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 [32] and high Re number (HRN) with wall function [33]. 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 HRN, which is the first point above the wall to lie in the logarithmic layer [34]. Bäckar and Davidson [35] proposed a hybrid approach, so called numerical wall function, where wall adjacent cells are divided into sub-grid and governing equations are solved with appropriate boundary conditions in this sub-grid.

2.2 Stratified Atmospheric Boundary Layer modelling

Either the Reynolds Averaged Navier–Stokes (RANS) equations or Large Eddy Simulations (LES) [36, 37] 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 [38]:

$$S_{\varepsilon b} = C_{\varepsilon 1}(1 - C_{\varepsilon 3})\frac{\varepsilon}{k}G_b \quad (2.12)$$

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

$$G_b = \beta g_i \frac{\mu_t}{\sigma_T} \left(\frac{\partial T}{\partial x_i} - \frac{g_i}{c_p} \right) \quad (2.13)$$

For 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 $z \leq 0.1h_{ABL}$:

$$\begin{aligned} k &= 6u_*^2 \\ \varepsilon &= \frac{u_*^3}{\kappa z} \left(1.24 + 4.3 \frac{z}{L_{MO}} \right) \end{aligned} \quad (2.14)$$

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

$$\begin{aligned} k &= 6u_*^2 \left(1 - \frac{z}{h_{ABL}} \right)^{1.75} \\ \varepsilon &= \frac{u_*^3}{\kappa z} \left(1.24 + 4.3 \frac{z}{L_{MO}} \right) \left(1 - 0.85 \frac{z}{h_{ABL}} \right)^{1.5} \end{aligned} \quad (2.15)$$

Under unstable ABL, the heat flux from the ground and 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 Equation (1.9). Turbulent kinetic energy k and dissipation rate ε under unstable ABL can be defined in Equation (2.16) for $z \leq 0.1h_{ABL}$ and Equation (2.17)

for $z > 0.1h_{ABL}$.

$$\begin{aligned} k &= 0.36w_*^2 + 0.85u_*^2 \left(1 - 3\frac{z}{h_{ABL}}\right)^{2/3} \\ \varepsilon &= \frac{u_*^3}{\kappa z} \left(1 + 0.5\left|\frac{z}{L_{MO}}\right|^{2/3}\right)^{1.5} \end{aligned} \quad (2.16)$$

$$\begin{aligned} k &= w_*^2 \left[0.36 + 0.9 \left(\frac{z}{h_{ABL}}\right)^{2/3} \left(1 - 0.8\frac{z}{h_{ABL}}\right)^2 \right] \\ \varepsilon &= \frac{w_*^3}{h_{ABL}} \left(0.8 - 0.3\frac{z}{h_{ABL}}\right) \end{aligned} \quad (2.17)$$

In order to simulate atmospheric stratification effects, Alinot and Masson [38] changed 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 [38]

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

Effects of atmospheric stratification on dense gas dispersion CFD simulations was addressed by Pontiggia et al. [39]. The consistency between Monin-Obukhov profiles with $k - \varepsilon$ model is obtained by addition of z-dependent source term S_ε to the ε transport equation. Under neutral atmospheric stability:

$$S_\varepsilon(z) = \frac{\rho u_*^4}{z^2} \left[\frac{(C_{\varepsilon 2} - C_{\varepsilon 1})\sqrt{C_\mu}}{\kappa^2} - \frac{1}{\sigma_\varepsilon} \right] - \mu \frac{u_*^3}{2\kappa z^3} \quad (2.18)$$

Under stable condition:

$$S_\varepsilon(z) = \frac{\rho u_*^4}{z^2} \left[\frac{(C_{\varepsilon 2} - C_{\varepsilon 1})\sqrt{C_\mu}}{\kappa^2} \Phi_\varepsilon^2 \sqrt{\frac{\Phi_\varepsilon}{\Phi_m}} - \frac{1}{\sigma_\varepsilon} \left(\frac{2}{\Phi_m} - \frac{1}{\Phi_m^2} + \frac{T_*}{\kappa T} \right) \right] - \mu \frac{u_*^3}{2\kappa z^3} \quad (2.19)$$

In 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 [40]:

$$q_s = h_f(T_s - T_f) \quad (2.20)$$

h_f is the local heat transfer coefficient, T_f is the fluid temperature. Kovalets and Maderich [41] used mixed coefficients between forced heat convection and natural convection.

Several studies used CFD commercial software such as FLUENT; as well as open-source software such as OpenFOAM to simulate ABL layers. Hargreaves and Wright [30] have shown that applying at fixed variable fluxes at top boundary is difficult in general CFD software and usually replaced by zero gradient (no shear stress), however this will result a decaying boundary layer. They also highlighted different approaches of modelling near wall region can lead to significantly unexpected results especially when using general CFD software. Wall function applied for atmospheric flow is necessarily modified as the standard wall function is based on experiments of sand grain roughed pipe [42] and wind engineering roughness length z_0 is far from similar to this kind of roughness. 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, Garreaud, and Muñoz [43] used Detached Eddy Simulation (DES) technique, which incorporates RANS models in near wall region and LES model in the rest, to simulate atmospheric wind circulation in open pit. The simulation takes in to account effects of buoyancy, stratification and complex geometry. A quasi-compressible approximation (treating density as explicit variable) was applied to incorporate stratification effect. Riddle et al. [44] simulated neutrally stable atmospheric boundary layer with RSM turbulence model.

2.3 CFD simulation of dense gas dispersion

Value of turbulent Schmidt number Sc_t are shown to have significant effect on dense gas dispersion concentration. Using wind tunnel data, Mokhtarzadeh-Dehghan, Akcayoglu, and Robins [45] reported that Sc_t was related with flow thermal stability, which is increased from 0.5 to 2.3 for the flows with Richardson number from 0.1 to 16.

Several studies in literature of atmospheric dense gas dispersion using CFD approach adopted commercial software such as ANSYS fluent [22, 46, 47, 48, 49], ANSYS CFX [50], FLACS [51, 52, 20] as well as open-source software such as OpenFOAM [53, 54, 55], FDS [56, 57]. Hansen et al. [52] validated FLACS in all tests in model evaluation database of LNG vapour dispersion. These include both wind tunnel test as well as field tests.

Several turbulence models were tested for dense gas dispersion problem. Mack and Spruijt [53] used standard $k - \varepsilon$ model with OpenFOAM solver `reactingFoam`, to simulate wind tunnel test case DAT632, which is the gravity driven flow of heavy gas in slope terrain. Different treatments of buoyancy term in ε equation are investigated. It was shown that standard $k - \varepsilon$ is able to predict turbulence damping due to vertical negative density gradient. Gavelli, Bullister, and Kytomaa [22] applied RSM model to account for directional effect of Reynold stress field (using standard $k - \varepsilon$ model as initial guess of turbulence) to simulate Falcon1 field test. Gant et al. [58] simulated CO_2 field test with realizable $k - \varepsilon$ model. Different RANS models were also tested for dense gas flow over obstacle [50]. Tauseef, Rashtchian, and Abbasi [59] used realizable $k - \varepsilon$ for modelling two Thorney Island test cases.

2.4 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.

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 significant effect on ABL dense gas dispersion concentration [45]. Therefore, the solver should be able to take Sc_t as input parameter. In case of LNG vapour dispersion, 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

OpenFOAM was firstly introduced in literature by Weller and Tabor [60]. In this chapter, fluid governing equations are introduced in parallel with OpenFOAM implementation. OpenFOAM keywords for variables, solvers names and codes are styled in `keywords` format. All other variables are defined in *Nomenclature* section. Two basic OpenFOAM solver `buoyantSimpleFoam` and `rhoReactingBuoyantFoam` are modified substantially and adopted for ABL simulation and atmospheric gas dispersion application.

3.1 Governing equations

3.1.1 Momentum equation

The general momentum equation can be written as [61]:

$$\frac{\partial}{\partial t}(\rho \mathbf{u}) + \nabla \cdot (\rho \mathbf{u} \mathbf{u}) = -\nabla p - \nabla \cdot \boldsymbol{\tau} + \rho \mathbf{g} \quad (3.1)$$

where the viscous stress tensor $\boldsymbol{\tau}$ is written for Newtonian fluid as:

$$\boldsymbol{\tau} = \mu (\nabla \mathbf{u} + (\nabla \mathbf{u})^T) - \frac{2}{3} \mu (\nabla \cdot \mathbf{u}) \boldsymbol{\delta} \quad (3.2)$$

Momentum equation solved in OpenFOAM is:

$$\frac{\partial(\rho \mathbf{u})}{\partial t} + \nabla \cdot (\rho \mathbf{u} \mathbf{u}) = -\nabla p + \rho \mathbf{g} + \nabla \cdot (2\mu_{eff} D(\mathbf{u})) - \nabla \left(\frac{2}{3} \mu_{eff} (\nabla \cdot \mathbf{u}) \right) \quad (3.3)$$

The rate of strain tensor: $D(\mathbf{u}) = \frac{1}{2} (\nabla \mathbf{u} + (\nabla \mathbf{u})^T)$. OpenFOAM implementation takes into account the effect of turbulence, therefore, $\mu_{eff} = \mu + \mu_t$ is the sum of molecular and turbulent viscosity. To account for buoyancy effect, pressure gradient and gravity force are combined to form p_{rgh} field:

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

where \mathbf{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 in OpenFOAM.

Code implementation of the above momentum equation in OpenFOAM can be presented in Listing 3.1.

Listing 3.1 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)
6      + turbulence->divDevRhoReff(U)
7      ==
8      fvOptions(rho, U)
9  );
10 UEqn.relax();
11 fvOptions.constrain(UEqn);
12 if (pimple.momentumPredictor())
13 {
14     solve
15     (
16         UEqn

```



```

17      ==
18      fvc::reconstruct
19      (
20          (
21              - ghf*fvc::snGrad(rho)
22              - fvc::snGrad(p_rgh)
23          )*mesh.magSf()
24      )
25  );
26  fvOptions.correct(U);
27  K = 0.5*magSqr(U);
28  }

```

3.1.2 Pressure equation

Pressure equation is used to enforce continuity constrain. Solving pressure equation can assure velocity field to satisfy continuity equation. Poisson equation of pressure in Cartesian coordinate [61] has the form of:

$$\frac{\partial}{\partial x_i} \left(\frac{\partial p}{\partial x_i} \right) = - \frac{\partial}{\partial x_i} \left[\frac{\partial}{\partial x_j} (\rho u_i u_j - \tau_{ij}) \right] + \frac{\partial^2 \rho}{\partial t^2} \quad (3.5)$$

Implicit method for solving momentum Equation (3.3) can be written in discretised form as:

$$A_P^{u_i} u_{i,P}^{n+1} + \sum_l A_l^{u_i} u_{i,l}^{n+1} = Q_{u_i}^{n+1} - \left(\frac{\delta p^{n+1}}{\delta x_i} \right)_P \quad (3.6)$$

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 (3.6) 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:

$$A_P^{u_i} u_{i,P}^{m*} + \sum_l A_l^{u_i} u_{i,l}^{m*} = Q_{u_i}^{m-1} - \left(\frac{\delta p^{m-1}}{\delta x_i} \right)_P \quad (3.7)$$

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:

$$u_{i,P}^{m*} = \tilde{u}_{i,P}^{m*} - \frac{1}{A_P^{u_i}} \left(\frac{\delta p^{m-1}}{\delta x_i} \right)_P \quad (3.8)$$

$$\tilde{u}_{i,P}^{m*} = \frac{Q_{u_i}^{m-1} - \sum_l A_l^{u_i} u_{i,l}^{m*}}{A_P^{u_i}} \quad (3.9)$$

where $\tilde{u}_{i,P}^{m*}$ contains all terms excluding pressure term as presents in Equation (3.9). Since velocity field is calculated from previous pressure, it should be corrected to satisfy continuity equation:

$$\frac{(\delta u_i^m)}{\delta x_i} = 0 \quad (3.10)$$

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

$$\frac{\delta}{\delta x_i} \left[\frac{\rho}{A_P^{u_i}} \left(\frac{\delta p^m}{\delta x_i} \right) \right]_P = \left[\frac{\delta(\rho \tilde{u}_i^{m*})}{\delta x_i} \right]_P \quad (3.11)$$

Solving discretised Poisson 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 3.6. 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:

$$\begin{aligned} u_i^m &= u_i^{m*} + u' \\ p^m &= p^{m-1} + p' \end{aligned} \quad (3.12)$$

Equation (3.8), (3.9) and (3.11) are rewritten in term of velocity correction and pressure correction as:

$$u'_{i,P} = \tilde{u}'_{i,P} - \frac{1}{A_P^{u_i}} \left(\frac{\delta p'}{\delta x_i} \right)_P \quad (3.13)$$

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

$$\tilde{u}'_{i,P} = - \frac{\sum_l A_l^{u_i} u'_{i,l}}{A_P^{u_i}} \quad (3.14)$$

The pressure correction equation:

$$\frac{\delta}{\delta x_i} \left[\frac{\rho}{A_P^{u_i}} \left(\frac{\delta p'}{\delta x_i} \right) \right]_P = \left[\frac{\delta(\rho u_i^{m*})}{\delta x_i} \right]_P + \left[\frac{\delta(\rho \tilde{u}'_i)}{\delta x_i} \right]_P \quad (3.15)$$

The last term in pressure correction Equation (3.15) 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.

The pressure Poisson equation pEqn implemented in OpenFOAM is shown in Listing 3.2.

Listing 3.2 Pressure poisson equation implementation in OpenFOAM

```

1 rho = thermo.rho();
2 // Thermodynamic density needs to be updated by psi*d(p) after the
3 // pressure solution
4 const volScalarField psip0(psi*p);
5 volScalarField rAU(1.0/UEqn.A());
6 surfaceScalarField rhorAUF("rhorAUF", fvc::interpolate(rho*rAU));
7 volVectorField HbyA(constrainHbyA(rAU*UEqn.H(), U, p));
8 surfaceScalarField phig(-rhorAUF*ghf*fvc::snGrad(rho)*mesh.magSf());
9 surfaceScalarField phiHbyA
10 (
11     "phiHbyA",
12     (
13         fvc::flux(rho*HbyA)
14         + rhorAUF*fvc::ddtCorr(rho, U, phi)

```

```

15     )
16     + phig
17 );
18 MRF.makeRelative(fvc::interpolate(rho), phiHbyA);
19 // Update the pressure BCs to ensure flux consistency
20 constrainPressure(p_rgh, rho, U, phiHbyA, rhorAUf, MRF);
21 fvScalarMatrix p_rghDDtEqn
22 (
23     fvc::ddt(rho) + psi*correction(fvm::ddt(p_rgh))
24     + fvc::div(phiHbyA)
25     ==
26     fvOptions(psi, p_rgh, rho.name())
27 );
28 while (pimple.correctNonOrthogonal())
29 {
30     fvScalarMatrix p_rghEqn
31     (
32         p_rghDDtEqn
33         - fvm::laplacian(rhorAUf, p_rgh)
34     );
35
36     p_rghEqn.solve(mesh.solver(p_rgh.select(pimple.finalInnerIter())));
37
38     if (pimple.finalNonOrthogonalIter())
39     {
40         // Calculate the conservative fluxes
41         phi = phiHbyA + p_rghEqn.flux();
42
43         // Explicitly relax pressure for momentum corrector
44         p_rgh.relax();
45
46         // Correct the momentum source with the pressure gradient flux
47         // calculated from the relaxed pressure
48         U = HbyA + rAU*fvc::reconstruct((phig + p_rghEqn.flux())/rhorAUf);
49         U.correctBoundaryConditions();
50         fvOptions.correct(U);
51         K = 0.5*magSqr(U);
52     }
53 }
54 p = p_rgh + rho*gh;

```

```

55 // Thermodynamic density update
56 thermo.correctRho(psi*p - psip0);
57 if (thermo.dpdt())
58 {
59     dpdt = fvc::ddt(p);
60 }
61 #include "rhoEqn.H"
62 #include "compressibleContinuityErrs.H"

```

3.1.3 Momentum-pressure coupling algorithms

SIMPLE Semi-Implicit Method for Pressure Linked Equations (SIMPLE) algorithm overcomes the slow convergence issue resulting from neglecting the term in Equation (3.15) by updating velocity field using Equation (3.12) and (3.13). To improve stability of a computation, under-relaxation is combined with SIMPLE. The variable change is limited from one iteration to the next:

$$p^m = p^{m-1} + \alpha_p p' \quad (3.16)$$

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 to obtain u_i^{m*}
 - 4: Solving pressure correction equation 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 (3.13):

$$u''_{i,P} = \tilde{u}'_{i,P} - \frac{1}{A_P^{u_i}} \left(\frac{\delta p''}{\delta x_i} \right)_P \quad (3.17)$$

In this second corrector step, the $\tilde{u}'_{i,P}$ term is not neglected but determined from Equation (3.14) 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 to obtain u_i^{m*}
 - 3: **for** PISO loop **do**
 - 4: Solving pressure correction equation 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 to obtain u_i^{m*}
 - 4: **for** PIMPLE inner correctors **do**
 - 5: Solving pressure correction equation 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
-

3.1.4 Species transport equation

Transport equation for mass fraction Y for each species α of a mixture, $\alpha = 1, 2, \dots, N$ [62]:

$$\frac{\partial \rho Y_\alpha}{\partial t} + \nabla \cdot (\rho Y_\alpha \mathbf{u}) = -\nabla \cdot (-\rho D_{eff} \nabla Y_\alpha) + r_\alpha \quad (3.18)$$

r_α is the reaction rate of specie α . In the scope of this thesis, non-reacting flow is in concerned, therefore this term is neglected. Specie mass flux \mathbf{j}_α is written in form of Fick's law:

$$\mathbf{j}_\alpha = -\nabla \cdot (-\rho D_{eff} \nabla Y_\alpha) \quad (3.19)$$

The effective mass diffusivity D_{eff} is obtained from dimensionless Schmidt number Sc and effective momentum viscosity:

$$Sc = \frac{\mu_{eff}}{\rho D_{eff}} \quad (3.20)$$

In OpenFOAM, $Sc = 1$, so mass diffusivity is assumed to be the same as viscosity. The code fragments implementation of Equation (3.18) is shown in Listing 3.3:

Listing 3.3 Species transport equation

```

1 {
2     reaction->correct();
3     Qdot = reaction->Qdot();
4     volScalarField Yt(0.0*Y[0]);
5     forAll(Y, i)
6     {
7         if (i != inertIndex && composition.active(i))
8         {
9             volScalarField& Yi = Y[i];
10
11             fvScalarMatrix YiEqn
12             (
13                 fvm::ddt(rho, Yi)
14                 + mvConvection->fvmDiv(phi, Yi)
15                 - fvm::laplacian(turbulence->muEff(), Yi)
16                 ==
17                 reaction->R(Yi)
18                 + fvOptions(rho, Yi)
19             );
20             YiEqn.relax();
21             fvOptions.constrain(YiEqn);
22             YiEqn.solve(mesh.solver("Yi"));
23             fvOptions.correct(Yi);
24             Yi.max(0.0);
25             Yt += Yi;
26         }
27     }
28     Y[inertIndex] = scalar(1) - Yt;
29     Y[inertIndex].max(0.0);
30 }

```

3.1.5 Energy equation

Energy conservation equation can be written for total fluid enthalpy H as [62]:

$$\frac{\partial}{\partial t}(\rho H) + \nabla \cdot (\rho H \mathbf{u}) = -\nabla \cdot \mathbf{q} - \mathbf{T} : \nabla \mathbf{u} + \frac{Dp}{Dt} + \dot{q}_k \quad (3.21)$$

The energy flux q take into account heat diffusion and mass diffusion as:

$$\mathbf{q} = -\nabla T + \sum_{\alpha=1}^N \frac{H_{\alpha}}{M_{\alpha}} \mathbf{j}_{\alpha} \quad (3.22)$$

where H_{α} and M_{α} are total enthalpy and molecular weight of species α . Specie mass flux \mathbf{j}_{α} is calculated as Equation 3.19.

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:

$$\frac{\partial(\rho h)}{\partial t} + \nabla \cdot (\rho \mathbf{u} h) + \frac{\partial(\rho K)}{\partial t} + \nabla \cdot (\rho \mathbf{u} K) - \frac{\partial p}{\partial t} = \nabla \cdot (\alpha_{eff} \nabla h) + \rho \mathbf{u} \cdot \mathbf{g} \quad (3.23)$$

In the above equation, $K \equiv |\mathbf{u}|^2/2$ 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:

$$\alpha_{eff} = \frac{\rho \nu_t}{Pr_t} + \frac{\mu}{Pr} = \frac{\rho \nu_t}{Pr_t} + \frac{k}{c_p} \quad (3.24)$$

OpenFOAM implementation of energy equation using enthalpy h variable is presented in Listing 3.4.

Listing 3.4 OpenFOAM implementation of Energy equation

```

1 volScalarField& he = thermo.he();
2 fvScalarMatrix EEqn
3 (
4     fvm::ddt(rho, he) + mvConvection->fvmDiv(phi, he)
5     + fvc::ddt(rho, K) + fvc::div(phi, K)
6     + (
7         he.name() == "e"
8         ? fvc::div
9         (

```

```

10         fvc::absolute(phi/fvc::interpolate(rho), U),
11         p,
12         "div(phiv,p)"
13     )
14     : -dpdt
15 )
16 - fvm::laplacian(turbulence->alphaEff(), he)
17 ==
18     Qdot
19 + fvOptions(rho, he)
20 );
21 EEqn.relax();
22 EEqn.solve();
23 thermo.correct();
24 Info<< "min/max(T) = "
25     << min(T).value() << ", " << max(T).value() << endl;

```

The calculation of the temperature is done iteratively using the Newton-Raphson method from solution of energy variable. If the specific heat capacity at constant pressure c_p is expressed in the form of temperature polynomial function:

$$c_p(T) = \sum_{i=0}^7 c_i T^i \quad (3.25)$$

The temperature in the j -th cell T_j is calculated from the following equation:

$$\int_{T_{std}}^{T_j} \left(\sum_{i=0}^7 c_i T^i \right) dT = h_j \quad (3.26)$$

Calculate the temperature from the sensible enthalpy h can be done from `thermo.correct()` after solving the energy equation (Listing 3.4).

3.2 Numerical schemes

3.2.1 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 required for the scheme to be bounded.

3.2.2 Convective term

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

$$\nabla \cdot (\mathbf{u}\phi) = \frac{1}{V} \int_V (\nabla \cdot \mathbf{u}\phi) dV = \frac{1}{V} \oint_S (\mathbf{n} \cdot \mathbf{u}\phi) dS = \frac{1}{V} \left(\sum_{\text{owner}} \mathbf{S} \cdot \mathbf{u}_f \phi - \sum_{\text{neighbor}} \mathbf{S} \cdot \mathbf{u}_f \phi \right) \quad (3.27)$$

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:

$$\Phi = \mathbf{u}_f \cdot \mathbf{S}_f \quad (3.28)$$

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

$$\Phi_f = \Phi_N + w(\Phi_P - \Phi_N) \quad (3.29)$$

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.

An example of selection and implementation for convection term of field Q is seen in Listing 3.5.

Listing 3.5 Selection and divergence scheme implimentation

```

1 divSchemes
2 {
3     default          none;
4     div(phi,Q)       Gauss <interpolation scheme>;
5 }
6
7 // Calculation of matrix coefficient from divergence scheme selection
8 fvm.lower() = -weights.internalField()*faceFlux.internalField();
9 fvm.upper() = fvm.lower() + faceFlux.internalField();
10 fvm.negSumDiag();

```

3.2.3 Diffusion term

Gauss scheme is used for discretisation of Laplacian term $\nabla \cdot (\nu \nabla \mathbf{u})$, which is diffusion term in momentum equation. Interpolation scheme for the diffusion coefficient and surface normal gradient scheme for evaluating $(\nabla \mathbf{u})$ 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. `limitedCorrected` scheme is used for severe non-orthogonality mesh which may lead to unstable solution.

An example of Gradient scheme implementation is listed below:

Listing 3.6 Discretisation of diffusion term

```

1 Foam::fv::gaussGrad<Type>::gradf
2 (
3     const GeometricField<Type, fvsPatchField, surfaceMesh>& ssf,
4     const word& name
5 )
6 {
7     typedef typename outerProduct<vector, Type>::type GradType;
8     const fvMesh& mesh = ssf.mesh();
9     tmp<GeometricField<GradType, fvPatchField, volMesh>> tgGrad;
10    GeometricField<GradType, fvPatchField, volMesh>& gGrad=tgGrad.ref();
11    const labelUList& owner = mesh.owner();
12    const labelUList& neighbour = mesh.neighbour();
13    const vectorField& Sf = mesh.Sf();
14    Field<GradType>& igGrad = gGrad;
15    const Field<Type>& issf = ssf;
16
17    forAll(owner, facei)
18    {
19        GradType Sfssf = Sf[facei]*issf[facei];
20        igGrad[owner[facei]] += Sfssf;
21        igGrad[neighbour[facei]] -= Sfssf;
22    }
23    igGrad /= mesh.V();
24
25    gGrad.correctBoundaryConditions();
26    ...
27 }

```

3.3 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.

3.3.1 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.

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

3.3.3 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
-

3.4 Boundary conditions

Moukalled, Mangani, and Darwish [63] highlighted the differences between physical conditions, e.g. "wall", "inlet", "outlet"; geometric constraints, e.g. "symmetry", "periodic"; and boundary conditions, i.e. set of equations used to define the variables at a domain boundary. For each physical condition, many derived types of boundary conditions can be imposed. Geometric constraints 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. 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.

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:

$$\tau_0 = -\nu_{eff} \frac{dU}{dn} \quad (3.30)$$

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

3.5 Thermophysical models

Energy, heat and transport properties are determined by a set of thermophysical models [64] 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 3.7.

Listing 3.7 ThermoType dictionary

```

1 ThermoType
2 {
3     type          heRhoThermo;
4     mixture       reactingMixture;
5     transport      sutherland;
6     thermo         janaf;
7     energy         sensibleEnthalpy;
8     equationOfState perfectGas;
9     specie         specie;
10 }
```

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:

$$\mu = A_s \frac{\sqrt{T}}{1 + T_s/T} \quad (3.31)$$

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):

$$\mu = \sum_i^{N-1} a_i T^i \quad (3.32)$$

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 N th polynomial function of temperature to define values of c_p as:

$$c_p = \sum_i^{N-1} a_i T^i \quad (3.33)$$

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 N th polynomial function of temperature:

$$\rho = \sum_i^{N-1} a_i T^i \quad (3.34)$$

where a_i is a coefficient of the polynomial.

3.6 Turbulence models

3.6.1 Reynolds-Averaged Navier-Stokes (RANS)

Reynolds-Averaged Navier-Stokes (RANS) equations derived by averaging Navier-Stokes equations are listed below [61]:

$$\frac{\partial}{\partial x_i} (\rho \bar{u}_i) = 0 \quad (3.35)$$

$$\frac{\partial}{\partial t}(\rho \bar{u}_i) + \frac{\partial}{\partial x_j}(\rho \bar{u}_i \bar{u}_j) = -\frac{\partial \bar{p}}{\partial x_i} + \frac{\partial \tau_{ij}}{\partial x_j} + \frac{\partial}{\partial x_j}(-\rho \overline{u'_i u'_j}) + \rho g_i \quad (3.36)$$

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

$$\frac{\partial}{\partial t}(\rho \bar{\phi}) + \frac{\partial}{\partial x_j}(\rho \bar{u}_j \bar{\phi}) = \frac{\partial}{\partial x_j}(\Gamma \frac{\partial \bar{\phi}}{\partial x_j}) + \frac{\partial}{\partial x_j}(-\rho \overline{u'_j \phi'}) \quad (3.37)$$

Turbulent fluxes are modelled using eddy-viscosity hypothesis which treat turbulent fluxes similar to laminar ones:

$$-\rho \overline{u'_i u'_j} = \mu_t \left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right) - \frac{2}{3} \rho k \delta_{ij} \quad (3.38)$$

$$-\rho \overline{u'_i \phi'} = \Gamma_t \frac{\partial \bar{\phi}}{\partial x_j} \quad (3.39)$$

$$-\rho \overline{u'_i \omega'_\alpha} = \Gamma_{t\alpha} \frac{\partial \bar{\omega}_\alpha}{\partial x_j} \quad (3.40)$$

Turbulent diffusivity Γ_t can be defined from turbulence Prandtl number Pr_t :

$$\Gamma_t = \frac{\mu_t}{Pr_t} \quad (3.41)$$

The $k - \varepsilon$ model

The $k - \varepsilon$ model relies on Prandtl-Kolmogorov expression [65] as:

$$\mu_t = \rho C_\mu \frac{k^2}{\varepsilon} \quad (3.42)$$

Two additional transport equations for turbulence kinetic energy and turbulence dissipation rate are required. In OpenFOAM, standard version from [33] is implemented. The

transport equations for k and ε are:

$$\frac{D}{Dt}(\rho k) = \nabla \cdot (\rho D_k \nabla k) + G_k - \frac{2}{3} \rho (\nabla \cdot \mathbf{u}) k - \rho \varepsilon + S_k \quad (3.43)$$

$$\frac{D}{Dt}(\rho \varepsilon) = \nabla \cdot (\rho D_\varepsilon \nabla \varepsilon) + \frac{C_{1\varepsilon} G_k \varepsilon}{k} - \left(\frac{2}{3} C_{1\varepsilon} - C_{3,RDT} \right) \rho (\nabla \cdot \mathbf{u}) \varepsilon - C_{2\varepsilon} \rho \frac{\varepsilon^2}{k} + S_\varepsilon \quad (3.44)$$

The model constants are tabulated in Table 3.1. $C_{3,RDT}$ is the rapid distortion theory (RDT) based compression term with can be excluded by setting the constant to zero.

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

$C_{1\varepsilon}$	$C_{2\varepsilon}$	C_μ	σ_k	σ_ε
1.44	1.92	0.09	1	1.3

To including the effect of buoyancy, the transport equations for k and ε are:

$$\frac{D}{Dt}(\rho k) = \frac{\partial}{\partial x_i} \left[\left(\mu + \frac{\mu_t}{\sigma_k} \right) \frac{\partial k}{\partial x_j} \right] + G_k + G_b - \rho \varepsilon \quad (3.45)$$

$$\frac{D}{Dt}(\rho \varepsilon) = \frac{\partial}{\partial x_i} \left[\left(\mu + \frac{\mu_t}{\sigma_\varepsilon} \right) \frac{\partial \varepsilon}{\partial x_j} \right] + C_{1\varepsilon} \frac{\varepsilon}{k} G_k + C_{1\varepsilon} C_{3\varepsilon} \frac{\varepsilon}{k} G_b - C_{2\varepsilon} \rho \frac{\varepsilon^2}{k} \quad (3.46)$$

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

$$G_k = \mu_t S^2 \quad (3.47)$$

G_b is the buoyancy source term:

$$G_b = -\frac{\mu_t}{\rho Pr_t}(\mathbf{g} \cdot \nabla \rho) = -C_g \nu_t(\mathbf{g} \cdot \nabla \rho) \quad (3.48)$$

Chan [66] proposed model constants as: $C_{1\varepsilon} = 1.44$, $C_{2\varepsilon} = 1.92$, $C_{3\varepsilon} = -0.8$ and $C_{3\varepsilon} = 2.15$ for the unstable and stable regimes respectively.

In case of undefined value of $C_{3\varepsilon}$, OpenFOAM use the default value of C_3 calculated using:

$$C_{3\varepsilon} = \tanh \left| \frac{\nu}{u} \right| \quad (3.49)$$

An example of model coefficients setting, and code for solving turbulence using standard $k - \varepsilon$ model are listed in Listing 3.8.

Listing 3.8 k, ε transport equation implementation in OpenFOAM

```

1 // Local references
2 const alphaField& alpha = this->alpha_;
3 const rhoField& rho = this->rho_;
4 const surfaceScalarField& alphaRhoPhi = this->alphaRhoPhi_;
5 const volVectorField& U = this->U_;
6 volScalarField& nut = this->nut_;
7
8 volScalarField::Internal divU
9 (
10     fvc::div(fvc::absolute(this->phi(), U))().v()
11 );
12 tmp<volTensorField> tgradU = fvc::grad(U);
13 volScalarField::Internal G
14 (
15     this->GName(),
16     nut.v()*(dev(twoSymm(tgradU().v())) && tgradU().v())
17 );
18 tgradU.clear();
19 // Update epsilon and G at the wall
20 epsilon_.boundaryFieldRef().updateCoeffs();
21 // Dissipation equation

```

```

22 tmp<fvScalarMatrix> epsEqn
23 (
24     fvm::ddt(alpha, rho, epsilon_)
25 + fvm::div(alphaRhoPhi, epsilon_)
26 - fvm::laplacian(alpha*rho*DepsilonEff(), epsilon_)
27 ==
28     C1_*alpha()*rho()*G*epsilon_()/k_()
29 - fvm::SuSp(((2.0/3.0)*C1_ - C3_)*alpha()*rho()*divU, epsilon_)
30 - fvm::Sp(C2_*alpha()*rho()*epsilon_()/k_(), epsilon_)
31 + epsilonSource()
32 + fvOptions(alpha, rho, epsilon_)
33 );
34 epsEqn.ref().relax();
35 epsEqn.ref().boundaryManipulate(epsilon_.boundaryFieldRef());
36 solve(epsEqn);
37 bound(epsilon_, this->epsilonMin_);
38 // Turbulent kinetic energy equation
39 tmp<fvScalarMatrix> kEqn
40 (
41     fvm::ddt(alpha, rho, k_)
42 + fvm::div(alphaRhoPhi, k_)
43 - fvm::laplacian(alpha*rho*DkEff(), k_)
44 ==
45     alpha()*rho()*G
46 - fvm::SuSp((2.0/3.0)*alpha()*rho()*divU, k_)
47 - fvm::Sp(alpha()*rho()*epsilon_()/k_(), k_)
48 + kSource()
49 + fvOptions(alpha, rho, k_)
50 );
51 kEqn.ref().relax();
52 solve(kEqn);
53 bound(k_, this->kMin_);
54 correctNut();

```

The inclusion of the buoyancy term in standard $k - \epsilon$ is implemented in OpenFOAM as in Listing 3.9.

Listing 3.9 Implementation of including buoyancy source term in OpenFOAM

```

1 template<class BasicTurbulenceModel>

```

```

2 tmp<volScalarField>
3 buoyantKEpsilon<BasicTurbulenceModel>::Gcoef() const
4 {
5     const uniformDimensionedVectorField& g =
6         this->mesh_.objectRegistry::template
7         lookupObject<uniformDimensionedVectorField>("g");
8     return
9         (Cg_*this->Cmu_)*this->alpha_*this->k_*(g & fvc::grad(this->rho_))
10        /(this->epsilon_ + this->epsilonMin_);
11 }
12
13 template<class BasicTurbulenceModel>
14 tmp<fvScalarMatrix>
15 buoyantKEpsilon<BasicTurbulenceModel>::kSource() const
16 {
17     const uniformDimensionedVectorField& g =
18         this->mesh_.objectRegistry::template
19         lookupObject<uniformDimensionedVectorField>("g");
20     if (mag(g.value()) > SMALL)
21     {
22         return -fvm::SuSp(Gcoef(), this->k_);
23     }
24     else
25     {
26         return kEpsilon<BasicTurbulenceModel>::kSource();
27     }
28 }
29
30 template<class BasicTurbulenceModel>
31 tmp<fvScalarMatrix>
32 buoyantKEpsilon<BasicTurbulenceModel>::epsilonSource() const
33 {
34     const uniformDimensionedVectorField& g =
35         this->mesh_.objectRegistry::template
36         lookupObject<uniformDimensionedVectorField>("g");
37     if (mag(g.value()) > SMALL)
38     {
39         vector gHat(g.value()/mag(g.value()));
40         volScalarField v(gHat & this->U_);
41         volScalarField u

```

```

42     (
43         mag(this->U_ - gHat*v)
44         + dimensionedScalar("SMALL", dimVelocity, SMALL)
45     );
46     return -fvM::SuSp(this->C1_*tanh(mag(v)/u)*Gcoef(), this->epsilon_);
47 }
48 else
49 {
50     return kEpsilon<BasicTurbulenceModel>::epsilonSource();
51 }
52 }

```

The SST $k - \omega$ model

The SST $k - \omega$ model [67] is shown to effectively solve turbulence in strong adverse pressure gradients and separation, which is failed when using standard $k - \varepsilon$ model.

$$\frac{D}{Dt}\rho k = \nabla \cdot (\rho(\mu + \alpha_k \mu_t) \nabla k) + \rho G - \frac{2}{3} \rho k (\nabla \cdot \mathbf{u}) - \rho \beta^* \omega k + S_k. \quad (3.50)$$

$$\frac{D}{Dt}(\rho \omega) = \nabla \cdot (\rho(\mu + \alpha_\omega \mu_t) \nabla \omega) + \rho \gamma \frac{G}{v} - \frac{2}{3} \rho \gamma \omega (\nabla \cdot \mathbf{u}) - \rho \beta \omega^2 - \rho (F_1 - 1) CD_{k\omega} + S_\omega \quad (3.51)$$

G is turbulent production term calculated similar to standard $k - \varepsilon$ model. The blending function F_1 is the trick to proper selection of $k - \omega$ and $k - \varepsilon$ without the user interaction, which is 0 away from the surface and switch to 1 inside the boundary layer:

$$F_1 = \tanh \left(\left(\min \left[\max \left(\frac{k^{1/2}}{\beta^* \omega y}, \frac{500\nu}{y^2 \omega} \right) \frac{4\rho \alpha_\omega 2k}{CD_{k\omega} y^2} \right] \right)^4 \right) \quad (3.52)$$

$$CD_{k\omega} = \max \left(2\rho \alpha_\omega 2 \frac{1}{\omega} \frac{\partial k}{\partial x_i} \frac{\partial \omega}{\partial x_i}, 10^{-10} \right) \quad (3.53)$$

The turbulent eddy viscosity:

$$\nu_t = a_1 \frac{k}{\max(a_1 \omega, b_1 F_2 \sqrt{2S_{ij}S_{ij}})} \quad (3.54)$$

F_2 is the second blending function:

$$F_2 = \tanh \left(\left[\max \left(\frac{2k^{1/2}}{\beta^* \omega y}, \frac{500\nu}{y^2 \omega} \right) \right]^2 \right) \quad (3.55)$$

All constants are calculated from blended coefficient F_1 , e.g. $\alpha_k = \alpha_{k1}F_1 + \alpha_{k2}(1 - F_1)$.

The constants table is given in Table 3.2.

Table 3.2 The $k - \omega$ 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

Wall function

The near-wall region can be divided into three parts: the viscous sub-layer, the buffer layer and the logarithmic region. The extend of the logarithmic region is increased with increasing Reynold number. Resolving the flow near boundary layer requires generally an excessive number of computational cells, while the adjacent cells of the boundary are required to be in viscous layer, i.e. the wall length scale $y^+ = 1$. The wall function is 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 [34].

The derivations of wall functions are rooted from the generality of the region between the 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 [34]. The RANS equations are simplified

in this case as:

$$\frac{d}{dy} \left((\mu + \mu_t) \frac{dU}{dy} \right) = 0 \quad (3.56)$$

Appropriate scales using for near wall region are:

$$u^+ = \frac{U}{u_*}, y^+ = \frac{yu_*}{\nu}, \nu_t^+ = \frac{\nu_t}{\nu}, k^+ = \frac{k}{u_*^2}, \varepsilon^+ = \frac{\varepsilon \nu}{u_*^4}, \omega^+ = \frac{\omega \nu}{u_*^2} \quad (3.57)$$

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.

$$u^+ = y^+ \quad (3.58)$$

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

$$u^+ = \frac{1}{\kappa} \ln(Ey^+) \quad (3.59)$$

$\kappa = 0.41$ 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 first cell) or from solving momentum equation [34]. 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.

For $\nu^2 - f$ turbulence model, by applying turbulence kinetic energy equation for near wall region using non-dimensional variables defined in Equation (3.57), we have relation for logarithmic layer:

$$k^+ = \frac{C_k}{\kappa} \ln(y^+) + B_k \quad (3.60)$$

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

For viscous sub-layer:

$$k^+ = \frac{2400}{C_{2\varepsilon}} \left[\frac{1}{(y^+ + C)^4} + \frac{2y^+}{C^3} - \frac{1}{C^2} \right] \quad (3.61)$$

These two regimes are separated using the value of $y_{laminar}^+$

$$y_{laminar}^+ = \frac{\ln(\max(Ey_{laminar}^+, 1))}{\kappa} \quad (3.62)$$

Similarly, for ε variable, we have relation for logarithmic layer:

$$\varepsilon^+ = \frac{1}{\kappa y^+} \quad (3.63)$$

For viscous sub-layer:

$$\varepsilon^+ = 2 \frac{k^+}{(y^+)^2} \quad (3.64)$$

For ω variable, we have relation for logarithmic layer:

$$\omega^+ = \frac{1}{\kappa C_\mu^{1/2} (y^+)^2} \quad (3.65)$$

For viscous sub-layer (β_1 is the $k - \omega$ model constant):

$$\omega^+ = \frac{6}{\beta_1 (y^+)^2} \quad (3.66)$$

Two standard implementations of wall function in OpenFOAM for turbulence kinetic energy k are `kqRWallFunction`, which is Neumann boundary (zero gradient), and `kLowReWallFunction`, which is fixed value condition. The value of boundary patches is calculated from Equation (3.61) or Equation (3.60), where important parameter y^+ is defined using known value of the friction velocity u_* . u_* can be calculated from simple relation derived by Launder and Spalding [33], assuming that generation and dissipation of energy are in

balance:

$$u_* = C_\mu^{1/4} k^{1/2} \quad (3.67)$$

k is turbulence kinetic energy in the first cell adjacent to the wall. Value of k in wall patches is calculated from k^+ as:

$$k = k^+ u_*^2 \quad (3.68)$$

Unlike k wall function, which value is defined at boundary surface, ε and ω wall function define its values in cell centre. Using `epsilonWallFunction`, the value is averaged from all surfaces defined as wall in the cell:

$$\varepsilon_p = \frac{C_\mu^{0.75} k^{1.5}}{\kappa y_P} \quad (3.69)$$

v_t wall functions are also necessarily defined in OpenFOAM. It is used to define wall shear stress as a remedy to below approximation where wall velocity gradient is significantly larger than velocity difference between the adjacent cell and the wall:

$$\tau_w = \nu \frac{\partial u}{\partial n}|_w \approx \nu \frac{(u_P - u_w)}{y_P} \quad (3.70)$$

Equation for the wall shear stress can be derived using Equation (3.67) and (3.59):

$$\tau_w = \rho u_*^2 = \rho u_* \frac{(u_P - u_w)}{\frac{1}{\kappa} \ln(Ey^+)} \quad (3.71)$$

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

$$v_t = \nu \left(\frac{\kappa y^+}{\ln(Ey^+)} - 1 \right) \quad (3.72)$$

v_t wall functions is then used to implicitly define velocity at wall adjacent cell. There are several options to use v_t wall functions in OpenFOAM. `nutLowReWallFunction` set

$v_t = 0$ which means that the flow near wall is sufficiently solved. `nutkWallFunction` and `nutUWallFunction` both use Equation (3.72) but with different calculation of y^+ . The former uses assumption of Eqn 3.67 for the calculation of y^+ as:

$$y^+ = \frac{y C_\mu^{1/4} k^{1/2}}{\nu} \quad (3.73)$$

`nutUWallFunction` using Equation (3.59) to derive relationship between y^+ and u_P :

$$y^+ \ln(Ey^+) - \frac{\kappa y u_P}{\nu} = 0 \quad (3.74)$$

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

`nutkAtmRoughWallFunction` calculate turbulent viscosity at wall adjacent cell as:

$$\nu_t = \frac{\kappa u_* y_P}{\left(\ln \frac{y_P + z_0}{z_0} \right)} \quad (3.75)$$

In case of sand-grain type rough wall model, implemented in `nutkRoughWallFunction` class, wall roughness value affects drag (resistance), heat and mass transfer on the wall. The wall velocity scale can be written as [68]:

$$u^+ = \frac{1}{\kappa} \ln(Ey^+) - \Delta B \quad (3.76)$$

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

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

$$f_r = 1 + C_s K_s^+ \quad (3.77)$$

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:

$$K_s^+ = \frac{\rho K_s u_*}{\mu} \quad (3.78)$$

$$u_* = C_\mu^{1/4} k_p^{1/2}$$

v_t is calculated using:

$$v_t = \frac{u_* \kappa y_P}{\ln \left(\frac{E_{yp}}{C_s k_s} \right)} \quad (3.79)$$

3.6.2 Reynold stress transport models

Launder, Reece and Rodi (LRR) turbulence model [31] is the most commonly used Reynolds stress transport model. Individual Reynolds stresses are solved instead of using turbulent viscosity models, e.g. Equation 3.42. Solving conservation equations for each individual Reynolds stresses is costly, but overcome deficiency due to modelling of turbulent viscosity. The conservation equations are given as:

$$\frac{D}{Dt}(\tau_{ij}) = -\tau_{im} \frac{\partial u_j}{\partial x_m} + \tau_{jm} \frac{\partial u_i}{\partial x_m} + \rho \varepsilon_{ij} - \Pi_{ij} + \frac{\partial}{\partial x_m} \left(v_t \frac{\partial \tau_{ij}}{\partial x_m} + C_s \frac{2k^2}{3\varepsilon} \frac{\partial \tau_{ij}}{\partial x_k} \right) \quad (3.80)$$

Dissipation term: $\varepsilon_{ij} = 2/3 \varepsilon \delta_{ij}$. Pressure strain term comprises two components, $\Pi_{ij} = A_{ij} + M_{ijmn} \frac{\partial u_m}{\partial x_n}$. The term involving fluctuation quantities:

$$A_{ij} = C_{S1} \frac{\varepsilon}{k} \left(\tau_{ij} + \frac{2}{3} \rho k \delta_{ij} \right) \quad (3.81)$$

And the term involving mean rate of strain:

$$M_{ijmn} = -\hat{\alpha} \left(P_{ij} - \frac{1}{3} P_{kk} \delta_{ij} \right) - \hat{\beta} \left(D_{ij} - \frac{1}{3} D_{kk} \delta_{ij} \right) - \hat{\gamma} \rho k S_{ij} \quad (3.82)$$

$$\begin{aligned} P_{ij} &= \tau_{ik} \frac{\partial u_j}{\partial x_k} + \tau_{jk} \frac{\partial u_i}{\partial x_k} \\ D_{ij} &= \tau_{ik} \frac{\partial u_k}{\partial x_j} + \tau_{jk} \frac{\partial u_k}{\partial x_i} \\ S_{ij} &= \frac{1}{2} \left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right) \end{aligned} \quad (3.83)$$

$\hat{\alpha}$, $\hat{\beta}$, $\hat{\gamma}$ are calculated in Table 3.3:

Table 3.3 The LRR mean rate of strain term constants

$\hat{\alpha}$	$\hat{\beta}$	$\hat{\gamma}$
$(8 + C_2)/11$	$(8C_2 - 2)/11$	$(60C_2 - 4)/55$

Equation of turbulence energy dissipation rate is solved for the dissipation term [69]:

$$\frac{D}{Dt}(\varepsilon) = C_\varepsilon \frac{\partial}{\partial x_i} \left[\frac{k}{\varepsilon} \overline{u_i u_j} \frac{\partial \varepsilon}{\partial x_j} \right] - C_{\varepsilon 1} \frac{\varepsilon}{k} G_k - C_{\varepsilon 2} \rho \frac{\varepsilon^2}{k} \quad (3.84)$$

Model constants of the quasi-isotropic variant of this model are:

Table 3.4 The LRR model constants

C_s	C_{S1}	$C_{2\varepsilon}$	$C_{\varepsilon 1}$	$C_{\varepsilon 2}$	C_ε
0.22	1.8	0.4	1.44	1.92	0.15

The default model constants in OpenFOAM are revealed in Listing 3.10.

Listing 3.10 LRR model constants selection

```

1 LRRCoeffs
2 {
3     Cmu      0.09;
4     C1       1.8;
```

```

5      C2          0.6;
6      Ceps1       1.44;
7      Ceps2       1.92;
8      Cs          0.25;
9      Ceps        0.15;
10
11     wallReflection yes;
12     kappa        0.41
13     Cref1        0.5;
14     Cref2        0.3;
15
16     couplingFactor 0.0;
17 }

```

3.7 The solvers

buoyantNonReactingSimpleFoam A solver, `buoyantNonReactingSimpleFoam` 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. `buoyantNonReactingSimpleFoam` is compatible with `buoyantNonReactingPimpleFoam` 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 below:

Listing 3.11 `buoyantNonReactingSimpleFoam` main code

```

1 #include "fvCFD.H"
2 #include "rhoThermo.H"
3 #include "turbulentFluidThermoModel.H"
4 #include "radiationModel.H"
5 #include "simpleControl.H"
6 #include "fvOptions.H"

```



```

7 // * * * * *
8 int main(int argc, char *argv[])
9 {
10     #include "postProcess.H"
11     #include "setRootCase.H"
12     #include "createTime.H"
13     #include "createMesh.H"
14     #include "createControl.H"
15     #include "createFields.H"
16     #include "createFieldRefs.H"
17     #include "createFvOptions.H"
18     #include "initContinuityErrs.H"
19     turbulence->validate();
20     // * * * * *
21     Info<< "\nStarting time loop\n" << endl;
22     while (simple.loop())
23     {
24         Info<< "Time = " << runTime.timeName() << nl << endl;
25         // Pressure-velocity SIMPLE corrector
26         {
27             #include "UEqn.H"
28             #include "EEqn.H"
29             #include "pEqn.H"
30         }
31         turbulence->correct();
32         runTime.write();
33         Info<< "ExecutionTime = " << runTime.elapsedCpuTime() << " s"
34             << "   ClockTime = " << runTime.elapsedClockTime() << " s"
35             << nl << endl;
36     }
37     Info<< "End\n" << endl;
38     return 0;
39 }

```

buoyantNonReactingPimpleFoam buoyantNonReactingPimpleFoam 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 below:

Listing 3.12 buoyantNonReactingPimpleFoam main code

```

1  #include "postProcess.H"
2  #include "setRootCase.H"
3  #include "createTime.H"
4  #include "createMesh.H"
5  #include "createControl.H"
6  #include "createTimeControls.H"
7  #include "initContinuityErrs.H"
8  #include "createFields.H"
9  #include "createFieldRefs.H"
10 #include "createFvOptions.H"
11 turbulence->validate();
12 while (runTime.run())
13 {
14     #include "readTimeControls.H"
15     #include "compressibleCourantNo.H"
16     #include "setDeltaT.H"
17     runTime++;
18     Info<< "Time = " << runTime.timeName() << nl << endl;
19     #include "rhoEqn.H"
20     // Pressure-velocity PIMPLE corrector loop
21     while (pimple.loop())
22     {
23         #include "UEqn.H"
24         #include "YEqn.H"
25         #include "EEqn.H"
26         // Pressure corrector loop
27         while (pimple.correct())
28         {
29             #include "pEqn.H"
30         }
31         if (pimple.turbCorr())
32         {
33             turbulence->correct();
34         }

```

```

35     }
36     rho = thermo.rho();
37     runTime.write();
38     Info<< "ExecutionTime = " << runTime.elapsedCpuTime() << " s"
39         << "   ClockTime = " << runTime.elapsedClockTime() << " s"
40         << nl << endl;
41 }
42 Info<< "End\n" << endl;
43 return 0;

```

rhoEqn is used to solve the continuity for density:

Listing 3.13 rhoEqn.H

```

1 fvScalarMatrix rhoEqn
2 (
3     fvm::ddt(rho)
4     + fvc::div(phi)
5     ==
6     fvOptions(rho)
7 );
8 fvOptions.constrain(rhoEqn);
9 rhoEqn.solve();
10 fvOptions.correct(rho);

```

YEqn is modified with user-defined Sc_i :

Listing 3.14 YEqn.H

```

1 {
2     // Reading user defined Sc number
3     scalar Sct = runTime.controlDict().lookupOrDefault<scalar>("Sc", 1.0);
4     reaction->correct();
5     Qdot = reaction->Qdot();
6     volScalarField Yt(0.0*Y[0]);
7     forAll(Y, i)
8     {
9         if (i != inertIndex && composition.active(i))
10        {

```

```
11     volScalarField& Yi = Y[i];
12     fvScalarMatrix YiEqn
13     (
14         fvm::ddt(rho, Yi)
15         + mvConvection->fvmDiv(phi, Yi)
16         - fvm::laplacian(turbulence->muEff()/Sct, Yi)
17         ==
18         reaction->R(Yi)
19         + fvOptions(rho, Yi)
20     );
21     YiEqn.relax();
22     fvOptions.constrain(YiEqn);
23     YiEqn.solve(mesh.solver("Yi"));
24     fvOptions.correct(Yi);
25     Yi.max(0.0);
26     Yt += Yi;
27 }
28 }
29 Y[inertIndex] = scalar(1) - Yt;
30 Y[inertIndex].max(0.0);
31 }
```

Chapter 4

Modelling of Atmospheric Boundary Layer

Successful solving ABL turbulence is a first step to calculate gas dispersion. In this Chapter, different turbulence models are modified to solve horizontal homogeneous ABL surface layer. Three cases are investigated including neutral ABL, neutral ABL in presence of obstacles and thermally stratified ABL.

4.1 Full scale simulation of the neutral ABL

4.1.1 Domain and mesh

The domain of $5000\text{ m} \times 500\text{ m}$ with the resolution of 500×50 cells is used for the simulation of neutral ABL over flat terrain. The mesh is uniform in stream-wise direction and stretched in vertical direction with the expansion ratio of 1.075.

4.1.2 Boundary conditions

The boundary conditions of the cases are represent in Table 4.1.

Table 4.1 Boundary conditions for neutral ABL simulation

ABL inlet	profiles of k , u , ε , ω	Eq. (2.2), (2.10)
ABL outlet	zeroGradient for all variables fixedValue for static pressure	
ABL side	zeroGradient for all variables	
ABL top	zeroGradient for all variables fixedFlux/zeroGradient for u and ε	Eq. (2.9)
ABL bottom	noSlip for u	

ABL parameters used to define inlet variable profiles are listed in Table 4.2 according to the reference case of Hargreaves and Wright [30]:

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

Steady state simulation is employed using `buoyantNonReactingSimpleFoam` described in previous chapter. 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, ϵ : PBiCGStab with DILU preconditioner

Residual control is set at three order of magnitude for pressure and four order of magnitude for other variables such as U, k, ϵ and h . Modification of $k - \epsilon$ and $k - \omega$ models (Equation (2.3)) are used to simulate neutral ABL and comparing with standard models. These three cases are summarised in Table 4.3

Table 4.3 Turbulence models setting for neutral ABL simulation

Turbulence model	standard $k - \epsilon$	Eq. (2.3)
	modified $k - \epsilon$	
	modified $k - \omega$ SST	
Wall functions	nutkWallFunction for ν_t	
	epsilonWallFunction for ϵ	
	kqRWallFunction for k	

Different level of inlet kinetic energy by altering C_μ according to Equation (2.2). The inclusion of source term by Pontiggia et al. [28] is implemented using Equation (2.4). Two values of default value $C_\mu = 0.09$ and $C_\mu = 0.017$ are simulated and compared with Monin-Obukhov theory.

4.1.4 Results and discussion of neutral ABL simulations

Modification of $k - \epsilon$ models achieve matched results as shown in Figure 4.1. Modified model constants as in Equation (2.3) is sufficient to compensate terms deflection from calculation of von-Karman constants $\kappa_{k-\epsilon} = 4.3$ from model constants and universal used $\kappa = 4.1$ used in Monin-Obukhov theory.

On the other hand, modification of $k - \omega$ has insignificant effect comparing with original model as shown in Figure 4.2. This is due to the fact that the calculation of von-Karman constants κ from model constants are close to value used to derive the Monin-Obukhov profiles, which is 4.08.

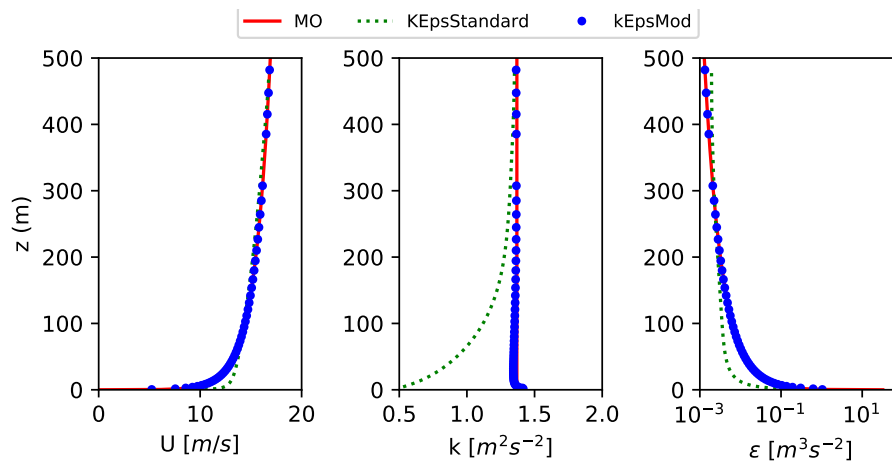


Fig. 4.1 Velocity, turbulent kinetic energy and turbulent dissipation rate profiles from simulation of neutral ABL using standard $k - \epsilon$ and modified $k - \epsilon$ turbulence model

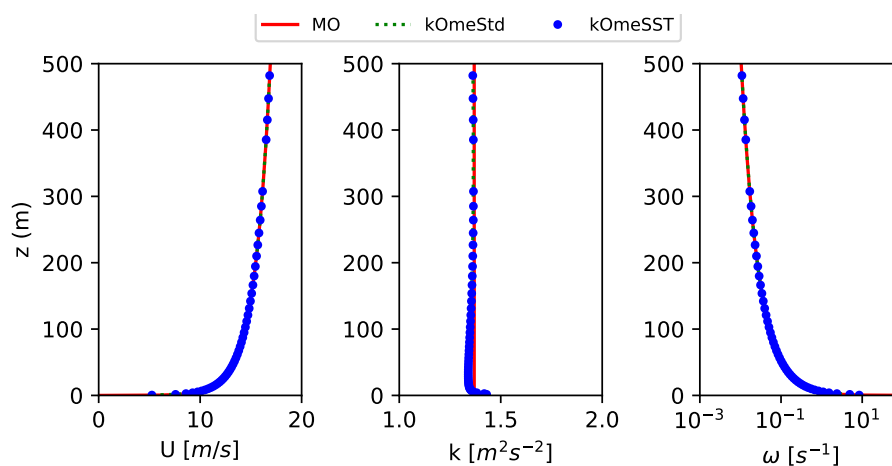


Fig. 4.2 Velocity, turbulent kinetic energy and turbulent dissipation rate profiles from simulation of neutral ABL using standard $k - \omega$ and SST $k - \omega$ turbulence model

Results from modelling different turbulence kinetic energy by varying C_μ are presented in Figure 4.3. The profiles of velocity and dissipation rate are perfectly matched with Monin-Obukhov profiles. In $C_\mu = 0.017$ simulation, the value of k near ground was smaller than theory value, however the kinetic energy level is matched with theory value at higher height. The smaller value of k at wall adjacent cell was due to wall function, where wall treatment used for default $C_\mu = 0.09$ is implemented. However, overall results are accepted for verifying the proposed model in simulating different level of kinetic energy.

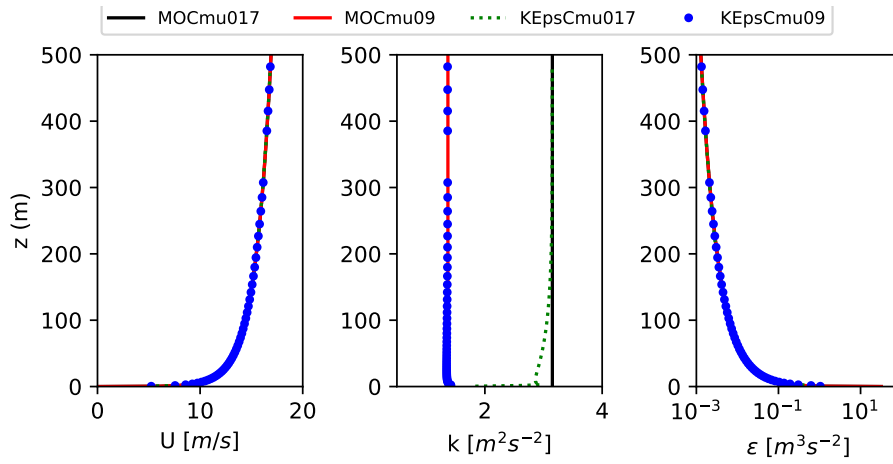


Fig. 4.3 Velocity, turbulent kinetic energy and turbulent dissipation rate profiles from simulations of different kinetic energy levels by varying $C_\mu = 0.09$ and $C_\mu = 0.017$

4.2 Simulations of neutral ABL in presence of obstacles

4.2.1 CEDVAL wind tunnel data

CEDVAL (Compilation of Experimental Data for Validation of Microscale Dispersion Models) wind tunnel data was carried out in the wind tunnel at the Meteorological Institute of the University of Hamburg. In A1-1 test, flow around a rectangular modelled building was tested in simulated atmospheric boundary layer modelled at a scale of 1:200. CEDVAL A1-1 test obstacle geometry is presented in Figure 4.4. The flow was measured in two

measurement planes (horizontal plane at a height of $z = 0.28H$ and vertical plane at $y = 0H$). Profiles of mean velocity and turbulence were obtained from two-component laser Doppler velocimetry. They are reported in terms of three mean velocity components longitudinal (\bar{U}), lateral (\bar{V}) and vertical (\bar{W}), corresponding to three turbulent intensity I_u , I_v and I_w , two Reynold stress $\overline{u'v'}$, $\overline{u'w'}$ [70]. Turbulent kinetic energy can be derived from above data as:

$$k = \frac{1}{2} (I_u \bar{U} + I_v \bar{V} + I_w \bar{W}) \quad (4.1)$$

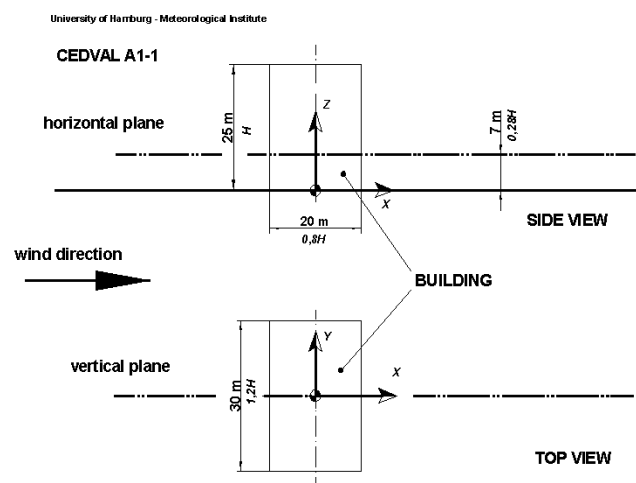


Fig. 4.4 Obstacle geometry of CEDVAL A1-1

Test results comprise of two parts: detailed flow measurements of the boundary layer flow before the model building was mounted in the test section and flow measurements when obstacle was included.

4.2.2 Domain and mesh generation

The domain size to the gas concentration is observed to choose the sufficient domain for further study. The flow can be closely assumed symmetric, only half of the flow will be simulated. The domain of $4\text{ m} \times 1\text{ m}$ is used for the simulation of neutral ABL over flat

terrain. The computational domain surfaces are named as: `ground`, `top`, `frontField`, `backField`, `sideField`. The mesh is shown in Figure 4.5:

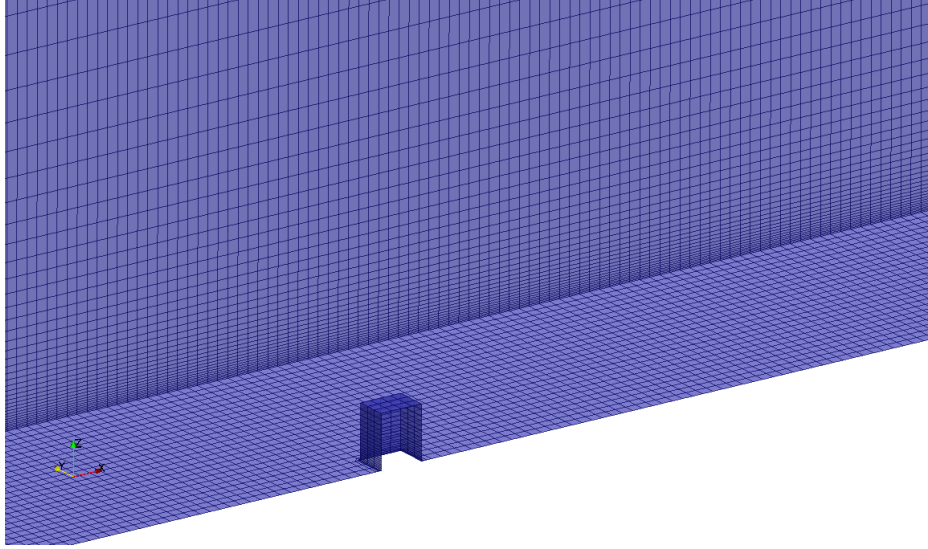


Fig. 4.5 The mesh using for CEDVAL A1 - 1 simulation

The adequate number of nodal points using for the study can be determined with the mesh independence study. Three different meshes which the number of nodes vary with the factor of two are compared. Structured mesh with hexahedral cells is made from OpenFOAM native mesh generation `blockMesh`. The mesh is uniform in the stream and cross-stream direction, while stretched in vertical direction. The maximum aspect ratio of the mesh is in the cell adjacent to the wall direction. By alternating this value, the wall length scale y^+ can be fit to the desired value.

4.2.3 Boundary conditions

ABL parameters are extracted from experimental data, which is summarised in Table 4.4. Profiles of inlet variables are using the same Equation as previous section.

Table 4.4 CEDVAL A1 - 1 ABL parameters

u_* (m/s)	z_0 (m)	u_{ref} (m/s)	z_{ref} (m)
0.374	0.0007	6	0.5

4.2.4 Numerical setting

A same numerical setting as previous section are also used for this simulation. Modified $k - \varepsilon$ and SST $k - \omega$ turbulence models which are effectively reproduced horizontally homogeneous ABL surface layer in flat terrain, are used to predict flow in the presence of obstacles.

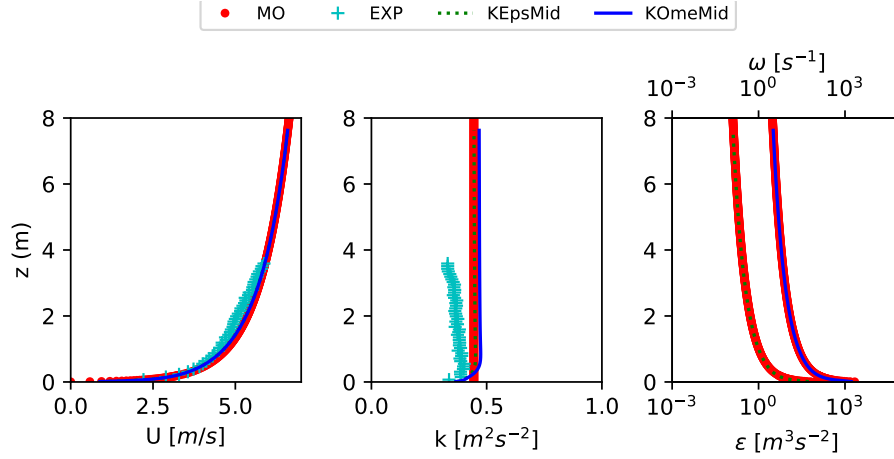
4.2.5 Results and discussion of neutral ABL in presence of obstacle simulations

Approaching flow

Data from approaching flow are compared with the simulation results. Profiles of U , k and ε are sample from the mid plane between the simulation inlet and the obstacle. Results are shown in Figure 4.6. Turbulence kinetic energy are shown to slightly decreased with height from experiments, however constant turbulence are assumed in the inlet profile. Velocity profiles are matched with observations. Turbulence dissipation is not evaluated in the experiments.

Influencing of obstacle

The same performance of $k - \varepsilon$ and $k - \omega$ models in predicting flow around an obstacle. Both are shown to over-predicted reattachment length in Figure 4.7. This is contributed from the fact that models constants are modified to fit the horizontal homogeneous ABL requirements and the deficiency of the models in resolving turbulence of flow in the wake of the obstacle.

Fig. 4.6 Profiles of U , k and ε of CEDVAL A1 - 1 test

4.3 Full scale simulation of the stable stratified ABL

The domain of $500 \text{ m} \times 30 \text{ m} \times 30 \text{ m}$ with the resolution of $500 \times 30 \times 43$ cells is used for the simulation of stable ABL over flat terrain. The mesh is uniform in stream-wise direction and stretched in vertical direction with the expansion ratio of 1.075. ABL parameters used in the simulation are tabulated in Table 4.5.

Table 4.5 Meteorology parameters used for stable ABL simulation

u_*	z_0	θ_*	q_s	L_{MO}
0.074	2E-4	0.145	2.2	16.5

4.3.1 Boundary conditions

The boundary conditions of the cases are represent in Table 4.6.

4.3.2 Numerical setting

Discretisation and linear solver settings are listed below:

- Time schemes: `steadyState`

Table 4.6 Boundary conditions for stable stratified ABL simulation

ABL inlet	profiles of $k, u, \varepsilon, \omega$	Eq. (2.2), 2.10
ABL outlet	zeroGradient for all variables fixedValue for static pressure	
ABL side	zeroGradient for all variables	
ABL top	zeroGradient for all variables fixedFlux/zeroGradient for u and ε	Eq. (2.9)
Turbulence model	buoyant $k - \varepsilon$ buoyant $k - \varepsilon$ with S_ε	Eq. (2.19)
Wall functions	noSlip for u nutkAtmRoughWallFunction for v_t epsilonWallFunction for ε kqRWallFunction for k	

- 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

Two cases (as shown in Table 4.7) are simulated to find appropriate setting for stable ABL. Horizontally homogeneous profiles of velocity and turbulent variables are the target which already achieved for neutral ABL. Results are compared with profiles calculated from Monin-Obukhov theory.

Table 4.7 Stable ABL simulations cases

	Case 1	Case 2	Monin-Obukhov theory
label	FOAM_ORIG	FOAM_MOD	MO
S_ε	0	Eq. (2.19)	

4.3.3 Results and discussion of stable stratified ABL simulation

Results of stable ABL simulations can be shown in Figure 4.8. Profiles of velocity, turbulent kinetic energy k and dissipation rate ε at the outlet boundary are compared with those derived from Monin-Obukhov theory. Standard $k - \varepsilon$ with no modification is shown inadequate to resolve turbulence in this case. Modification of $k - \varepsilon$ models as Equation (2.19) achieved matched results against Monin-Obukhov theory. This verified the appropriate of proposed model in simulation ABL turbulence in stable stratified atmospheric stability.

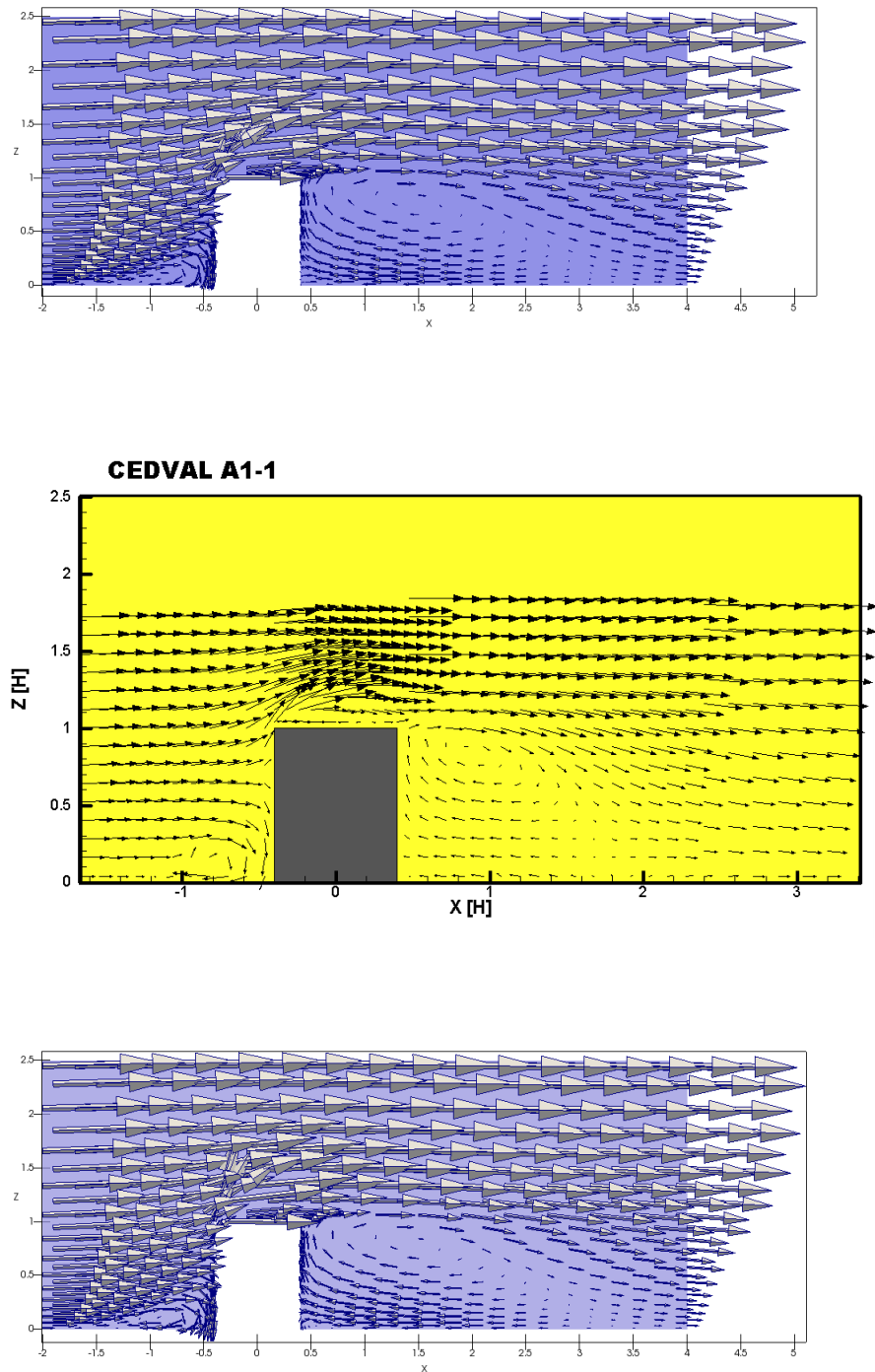


Fig. 4.7 Contour plot of velocity on planes $y=0$, from top to bottom: $k-\varepsilon$ model, Experiment data, $k-\omega$ model

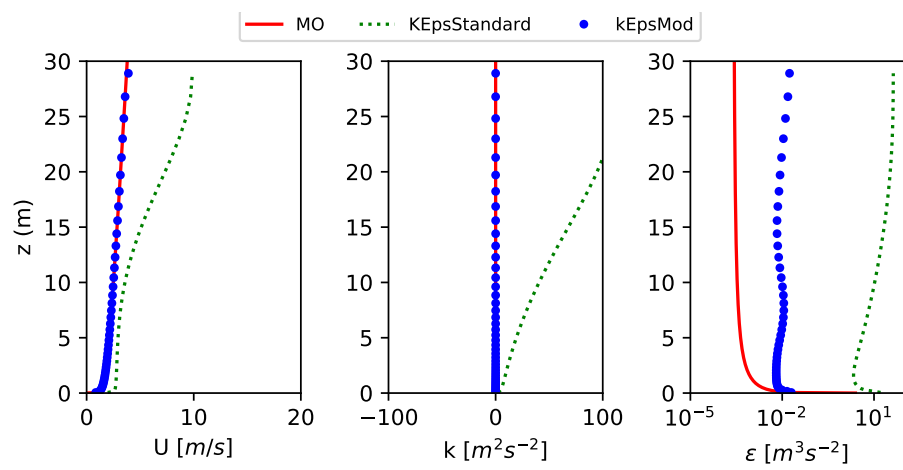


Fig. 4.8 Velocity, turbulent kinetic energy and turbulent dissipation rate profiles in Stable ABL

Chapter 5

Atmospheric boundary layer gas dispersion

In this Chapter, the gas dispersion simulations using OpenFOAM will be validated using wind tunnel tests and field tests. The test case name will be formatted as `testName`.

FDS (Fire Dynamics Simulator) [71] is a low Mach number code using LES turbulence model. FDS uses a second-order accurate finite-difference approximation to the governing equations on a series of connected rectilinear meshes. The flow variables are updated in time using an explicit second-order Runge-Kutta scheme. OpenFOAM concentration results are compared with FDS data extracted from [56].

SPMs from OpenFOAM results will be compared with FLACS (FLame ACceleration Simulator) [51, 52], a commercial CFD software used for explosion modelling and atmospheric dispersion modelling within the field of industrial safety and risk assessment.

5.1 Dense gas dispersion in wind tunnel tests

5.1.1 Hamburg wind tunnel test

Small scaled atmospheric boundary layer was conducted at the University of Hamburg to investigate the dispersion of instantaneous and continuous of heavy gas releases. The test section of the open-circuit 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 is utilized to establish a zero pressure gradient boundary layer. König-Langlo and Schatzmann [72] derived from the wind tunnel data the lower flammability limit (LFL) distances for different obstacle configuration including the worst case, i.e. an along wind street canyon, semi-circular wall as well as unobstructed terrain.

Dimensional analysis was used to derive similarity laws to match small-scale wind tunnel data and full-scale desired data. For instantaneous release, the resulted length L_{ci} , time T_{ci} and velocity U_{ci} scales are [72]:

$$\begin{aligned} L_{ci} &= V_0^{1/3} \\ T_{ci} &= \left(\frac{L_{ci}}{g'} \right)^{1/2} \\ U_{ci} &= (L_{ci} g')^{1/2} \end{aligned} \quad (5.1)$$

Similarly, the resulted length L_{cc} , time T_{cc} and velocity U_{cc} scales for continuous release are:

$$\begin{aligned} L_{cc} &= \left(\frac{\dot{V}_0^2}{g'} \right)^{1/5} \\ T_{cc} &= \left(\frac{\dot{V}_0}{g'^3} \right)^{1/5} \\ U_{cc} &= (\dot{V}_0 g'^2)^{1/5} \end{aligned} \quad (5.2)$$

Flat, unobstructed tests Two tests DA0120 and DAT223, which are included in MEP, are used to validate OpenFOAM solver in prediction of dense gas dispersion over a flat, unobstructed terrain in simulated neutral ABL.

In DA0120 and DAT223 tests, continuous source of SF_6 gas was released in flat terrain without obstructions. The gas is injected from the perforated disk with diameter approximately 7 cm. Aspirated hot-wire probes are used to measure gas concentration at the ground level at various locations. Long averaging times peak concentration data is compared between predicted simulation and reported experiment data. These tests parameters are summarised in Table 5.1.

Table 5.1 Hamburg flat, unobstructed test case parameters

	Unit	DA0120	DAT223
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
Surface roughness	m	0.0001	0.0001
Wind speed	m/s	0.54	0.74
Reference height	m	0.00718	0.01367
Ambient temperature	$^{\circ}C$	20	20
Source diameter	m	0.07	0.07
Spill rate	kg/s	0.0001743	0.000872

5.1.2 Numerical setting

Effect of turbulent Schmidt number Sc_t is investigated in dense gas dispersion. Three test cases are summarised in Table 5.2. Effect of turbulent models is examined by applying modified $k - \epsilon$ and SST $k - \omega$ which are already validated in simulating ABL over flat terrain in Section 4.1.

Firstly, the steady simulation using `bouyantSimpleFoam` is performed to establish the steady ABL flow prior to the dense gas release. The solver accounting to buoyancy effects

Table 5.2 Turbulent Schmidt number Sc_t in Hamburg tests

	Case 1	Case 2	Case 3
label	FOAM_ORIG	FOAM_Sc07	FOAM_Sc03
Sc_t	1	0.7	0.3

bouyantSimpleFoam is used for taking care of density stratification presenting in dense gas flow. The atmospheric inlet profiles are specified by Monin–Obukhov similarity theory with parameters in Table 5.1. Two different turbulent models: standard $k - \varepsilon$ with modifications and SST $k - \omega$ are used to study the ability to simulate the atmospheric boundary of each model. Secondly, the transient simulation is performed using steady simulation solutions as initial fields. A modified version of rhoReactingBouyantFoam is studied to model multi-species flow where mixture considered are air and dense gas SF_6 . The wind tunnel tests were conducted in isothermal condition, therefore constant thermal and transport properties are used for both gases.

In simulations of DA0120 and DAT223 tests, the discretisation schemes and linear solver setting are the same as in the simulation of neutral ABL (Section 4.1).

5.1.3 Results and discussion of gas dispersion in wind tunnel tests

Peak concentration prediction

The steady state plumes of DAT0120 and DAT223 tests are plotted in Figure 5.1 and 5.2 respectively. Under higher release volume flow rate and higher wind speed, DAT223 plume are shown for wider and higher concentration in downwind zone.

The predicted and measured peak gas concentration are compared at several distances from the spill in Figure 5.3. Turbulent Schmidt number Sc_t has significant effect in predicting dense gas dispersion. The original rhoReactingBouyantFoam code, with assume species diffusivity equals to viscosity, is shown to over-predicted concentration with factor of three. New code with implement variable species diffusivity by means of reading Sc_t from file is

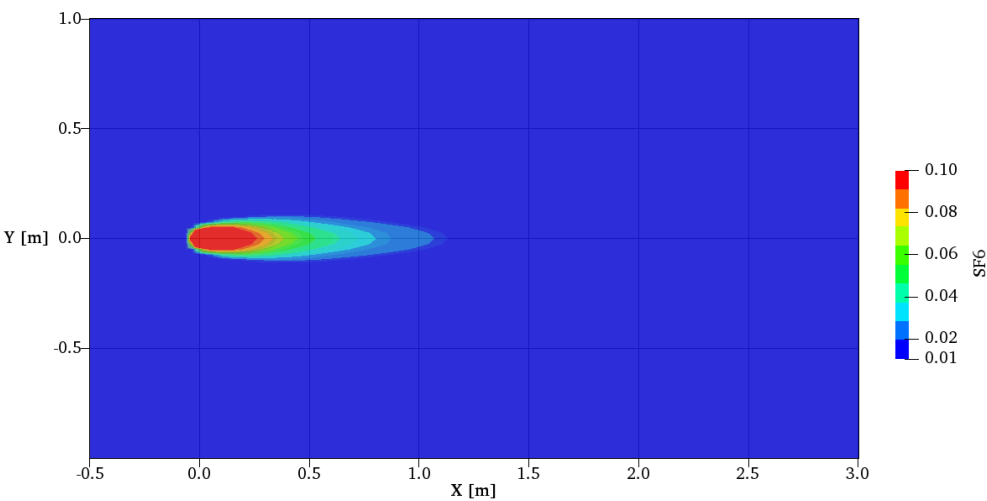


Fig. 5.1 DA0120 contour of gas concentration

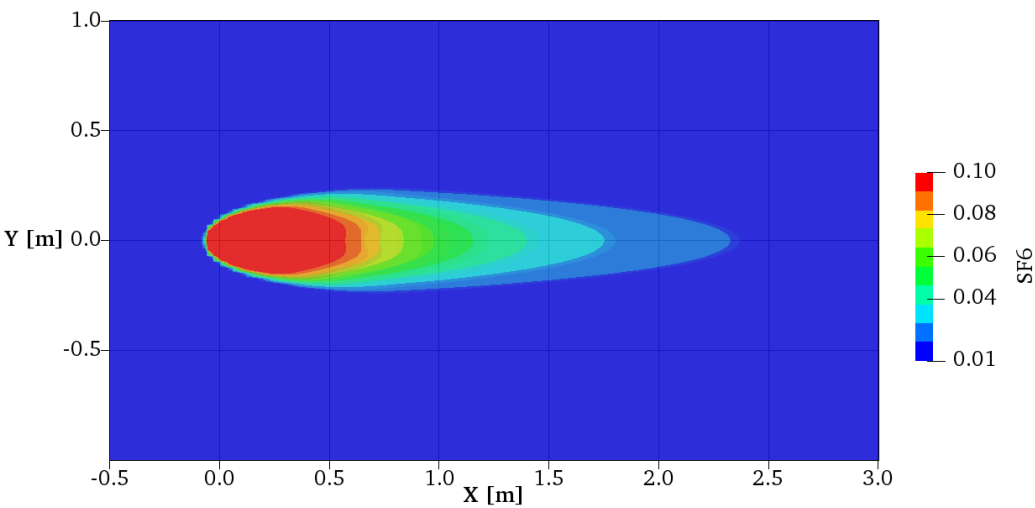


Fig. 5.2 DAT223 contour of gas concentration

implemented. The value of $Sc_t = 0.3$ is shown to perfectly matched with the experimental data, however there was a slightly acceptable over-predicted species concentration at a point near the source release.

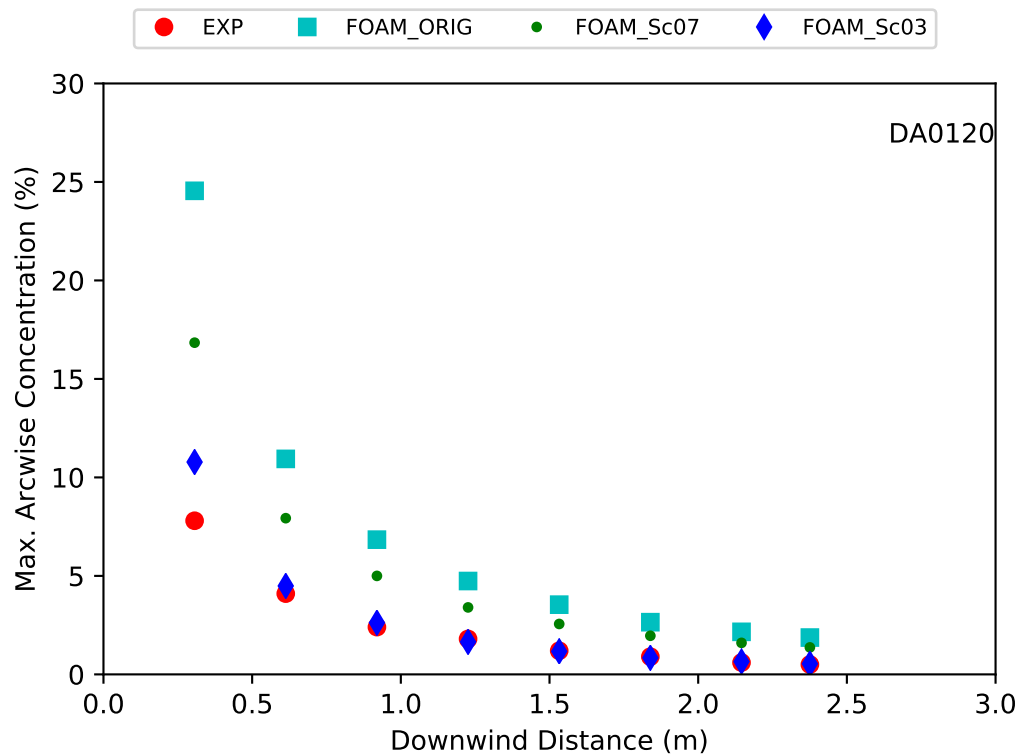


Fig. 5.3 Peak concentration DA0120

Results from DAT223 simulation are presented in Figure 5.4. Satisfactory over predicted peak concentration is similar to DA0120 case.

Point-wise concentration

Figure 5.5 presents gas concentration at downwind distance $X = 1.84$ of DA0120 test. The simulation can reproduce the averaged gas concentration. The first incidence time of gas concentration is earlier than observed in experiments. However, the time reaching averaged maximum concentration is well predicted.

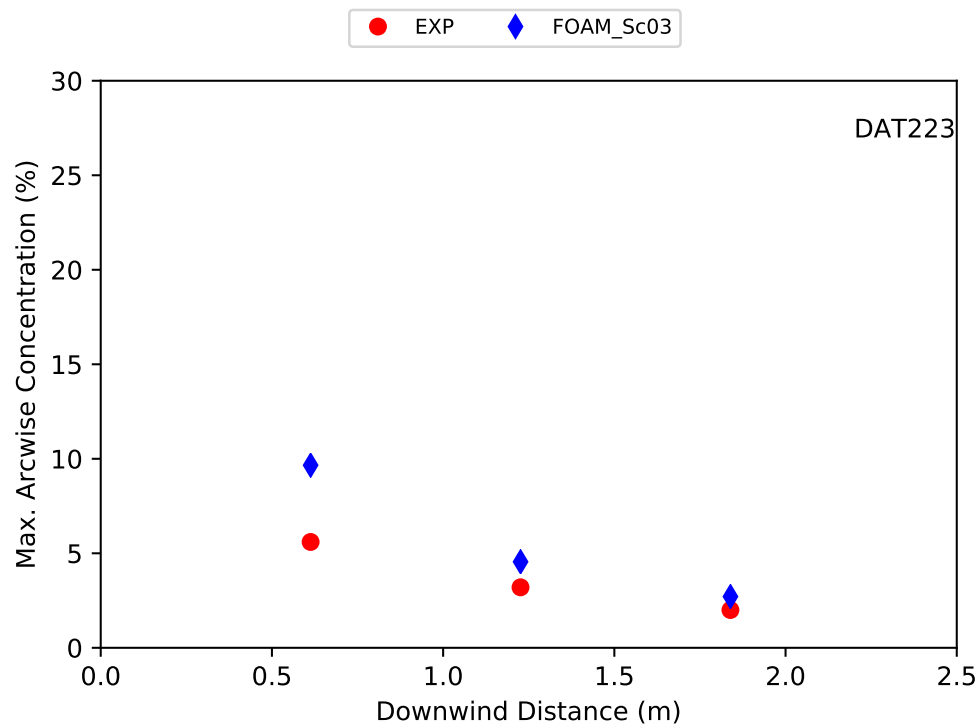
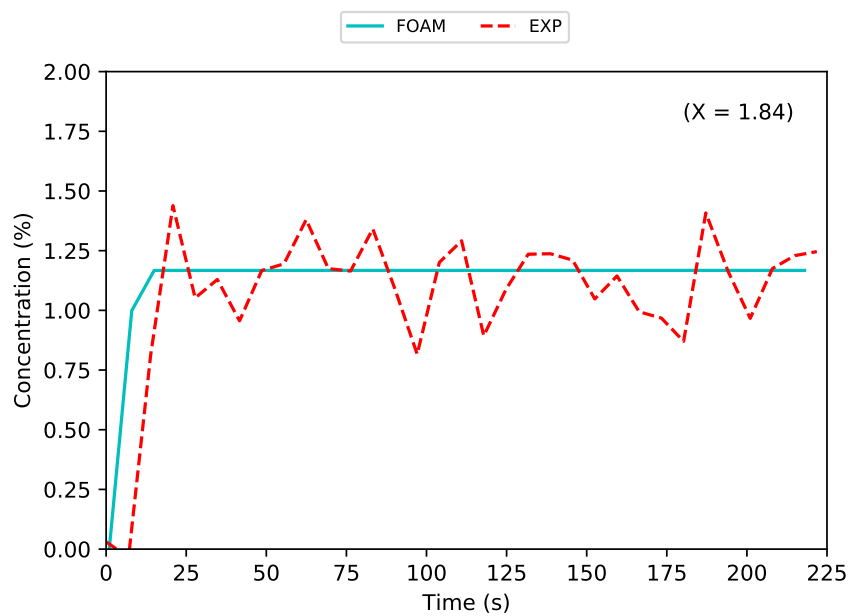


Fig. 5.4 Peak concentration DAT223

Fig. 5.5 Concentration at $X = 1.84$ of DA0120

Statistical model evaluation

Statistical performance of OpenFOAM results are compared with specified commercial code for gas dispersion FLACS in Table 5.3. FLACS results are extracted from [52]. 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 5.3 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 [52]	0.25	0.29	0.89	1.34	1.61
<i>FOAM</i> (current study)	-0.06	0.02	1.07	1.06	1.02

5.2 Dense gas 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 for validation of the proposed model on gas dispersion over flat terrain. Burro7 and Burro9 tests were conducted under neutral ABL. Of all tests, Burro7 had the largest spill volume and the longest spill duration. Burro3 test was conducted under the most unstable atmospheric conditions. Burro8 test was conducted in the most stable atmospheric condition and the lowest wind speed. The detail description of Burro series is summarised in Section 1.4.1.

5.2.1 Domain and grid generation

Without wind direction fluctuation, the flow can be closely assumed symmetric, therefore, only half of the flow will be modelled. However, the symmetric plane of the flow does not coincide with the centreline of instrument arrays. Therefore, a conversion of the point position should be made when comparing the point-wise data of the simulation and experimental measurements. Location of instruments in simulated domain (x_1, y_1) can be calculated from instruments location (x_2, y_2) according to the deflection angle between the wind direction

and the centreline of instrument arrays θ_{wind} as:

$$\begin{aligned} x_2 &= x_1 \cos \theta_{wind} + y_1 \sin \theta_{wind} \\ y_2 &= y_1 \cos \theta_{wind} - x_1 \sin \theta_{wind} \end{aligned} \quad (5.3)$$

The arcs of 57 m, 140 m, 400 m and 800 m gas instruments for each height, centreline of instrument arrays, the transformed points used in the Burro9 test simulation and centerline of simulation domain are plotted in Figure 5.6.

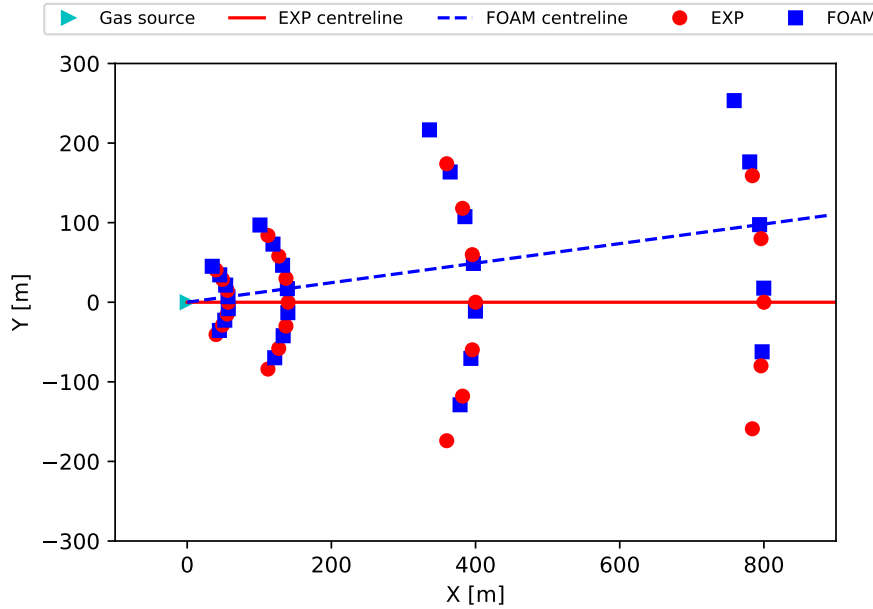


Fig. 5.6 Position of gas instruments (EXP) and their transformed points in simulation domain (FOAM) plotted for Burro9 test

The suitable type of the 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 much more computationally effective than unstructured mesh using tetrahedral cells [73]. The mesh is refined in gas dispersion region to accurately solving the flow there. An example of the mesh used for Burro9 simulation can be shown in Figure 5.7. The adequate number of nodal points used for the study can be determined using the mesh independence study [7] where effects of mesh on solution of peak gas concentration are investigated. Three

different meshes which the number of nodes vary with the factor of two are compared. The best optimum mesh achieved is used for further study. All computational domain and mesh parameters are summarised in Table 5.4.

Table 5.4 Burro test computational domain and mesh parameters

	Mesh 1	Mesh 2	Mesh 3
Domain region	[(-150, 0, 50), (850, 300, 50)]		
Refined region	[(-100, 0, 5), (400, 100, 5)]		
Aspect ratio	20		
Vertical cell expansion ratio	1.075		
Mesh size (m)	10	5	2
Mesh refined size (m)	5	2	1

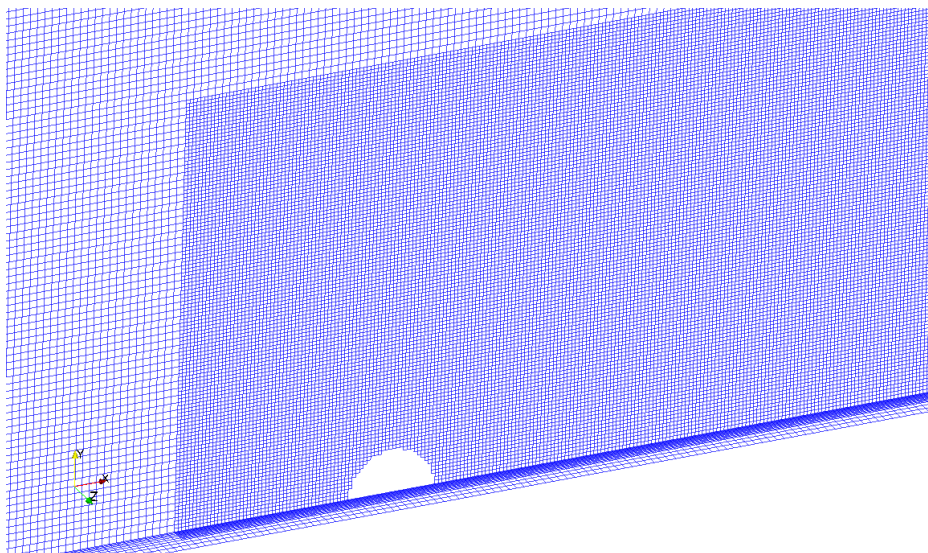


Fig. 5.7 The mesh using for Burro9 test

5.2.2 Boundary conditions

Atmospheric air inlet boundary Monin-Obukhov similarity theory is used to specify the wind velocity and temperature profile at the inlet. All required meteorological parameters are tabulated in Table 5.5.

Table 5.5 Burro tests meteorological parameters

	Burro3	Burro7	Burro8	Burro9
u_*	0.249	0.372	0.074	0.252
z_0	2E-4	2E-4	2E-4	2E-4
u_{ref}	5.4	8.4	1.8	5.7
z_{ref}	2	2	2	2
T_2	33.8	33.7	33.1	35.4
T_*	-0.65	-0.23	0.145	-0.1
q_s	154	41	2.2	-10
L_{MO}	-9.06	-114	16.5	-140

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 at the setting of this condition.

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

$$D = \sqrt{\frac{4m}{\pi m'' \Delta t}} \quad (5.4)$$

Volume spill rate is used as gas inlet condition:

$$\dot{V} = \frac{m}{\rho \Delta t} \quad (5.5)$$

LNG spill variables used in simulation are tabulated in Table 5.6.

Table 5.6 Burro test spill conditions

	Burro3	Burro7	Burro8	Burro9
Vaporization rate ($\text{kg m}^{-2} \text{s}^{-1}$)	0.167	0.167	0.167	0.167
Spill velocity (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 ($\text{m}^3 \text{s}^{-1}$)	50.05	56.46	66.13	77.17
Spill pool diameter (m)	25.9	27.5	29.9	32.2

Ground boundary conditions Three different models of heat transfer from the ground are used to study their effect in numerical results, which summarised in Table 5.7. For constant heat flux case, the value of 200 W/m^2 is used.

Table 5.7 Wall thermal boundary conditions in Burro tests

	Case 1	Case 2	Case 3
Heat transfer model	Adiabatic wall	Constant Heat Flux	Wall temperature
Label (Fig. 5.10)	Adiabatic	fixedFlux	fixedTem

top, side and back boundaries Assuming static pressure are constant throughout the domain, a constant `prghPressure` is set at `top` boundary condition. `prghPressure` provides static pressure condition for `p_rgh` field:

$$p_{rgh} = p - \rho gh \quad (5.6)$$

Velocity and epsilon fluxes are also placed at `top` boundary according to Equation (2.9). `zeroGradient` are specified for all variables at `side` and `back` boundaries.

5.2.3 Thermophysical models

CoolProp, an open-source thermophysical property library [74] is used to derive incompressible thermal physical properties for air and CH_4 to take into account of variable gas

properties due to temperature changes. Coefficients to derive gas properties as the function of temperature according to Equation (3.34) are presented in Table 5.8.

$$\rho = \sum_i^N a_i T^i \quad (3.34 \text{ revisited})$$

Table 5.8 Coefficients (Eq. (3.34)) 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		

5.2.4 Numerical setting

The steady simulation uses the atmospheric inlet specified by Monin–Obukhov similarity theory. Two different turbulent models: standard $k - \varepsilon$ with modifications and SST $k - \omega$ are used to study the ability to simulate the atmospheric boundary of each model.

The transient simulation is divided into two steps. During the spill duration, from the time of zero to the time of spill ends and after spill ends, which the inlet is treated as the solid wall boundary.

5.2.5 Results and discussion of gas dispersion over flat terrain

The steady simulation

Profiles of velocity and turbulence quantities are sampled at the outlet boundary and compared with Monin-Obukhov theory profiles which are used as inlet boundary conditions. The steady state simulation of ABL with $k - \varepsilon$ shows that wind velocity and turbulence profiles are accurately reproduced as presented in Figure 5.8. A small value difference decay of k (less than 10%) are shown for SST $k - \omega$ at the outlet.

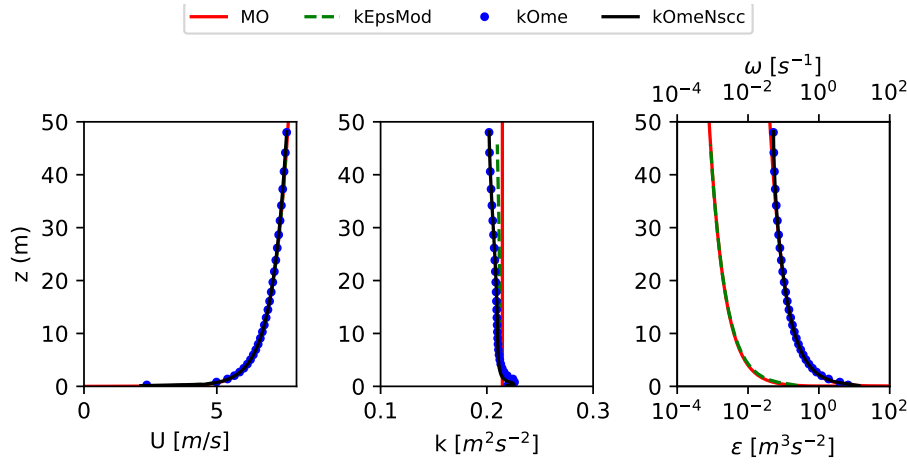


Fig. 5.8 ABL profiles of Burro9

A decay of turbulence kinetic energy of SST $k - \omega$ has not shown in the previous study due to the fact that the domain is extended to a large value downstream in this full scale experiment. A successfulness of modified $k - \varepsilon$ model proved that the proposed modelled can adequately reproduce the Monin-Obukhov ABL profiles in full scale simulation.

Mesh sensitivity study

Since maximum gas concentration at specific arrays downwind is in concerned. Maximum concentration at the arcs of 57 m, 140 m, 400 m and 800 m downwind are used as parameters for mesh sensitivity study. Three meshes with refined factors as summarised in Table 5.4

are used to simulate LNG gas dispersion at adiabatic thermal wall condition. Results of four peak arc-wise concentrations are plotted in Figure 5.9.

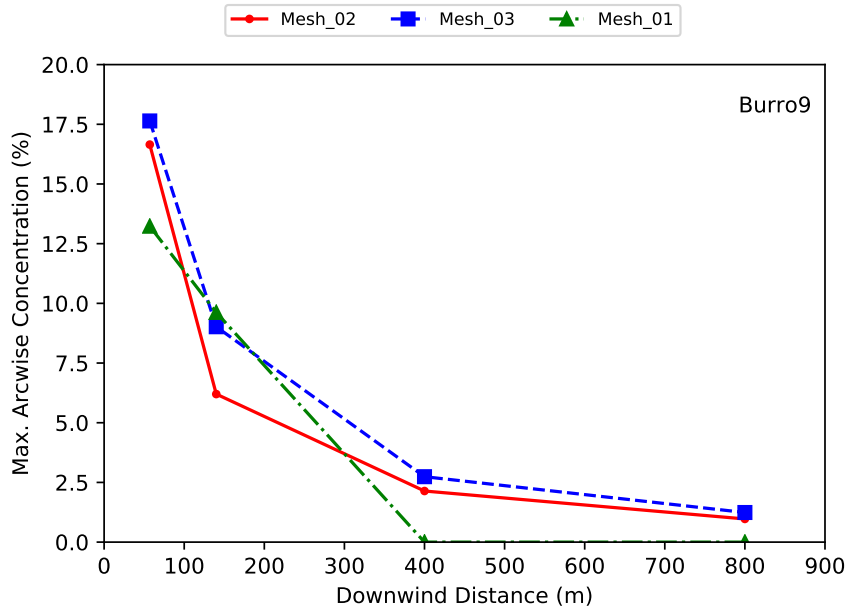


Fig. 5.9 Burro9 mesh sensitivity study

Increasingly mesh refinements help to solve maximum concentration more accurately. The difference values between meshes are significantly reduced. Due to computational restriction, Mesh 3 parameters (Table 5.4) will be used in the following study.

Ground heat transfer sensitivity study

Three different ground heat transfer models as in Table 5.7 are used to examine the effect of ground heat in predicting peak gas concentration. Plotted in Figure 5.10 are results from this study.

The adiabatic case results a better prediction to experiment data than fixed flux and fixed temperature cases. However, all predictions are under-predicted. These may be due to buoyancy effect is over-predicted so the gas concentration at downwind arcs (at 400 and 800 m) are zero in the fixed flux case.

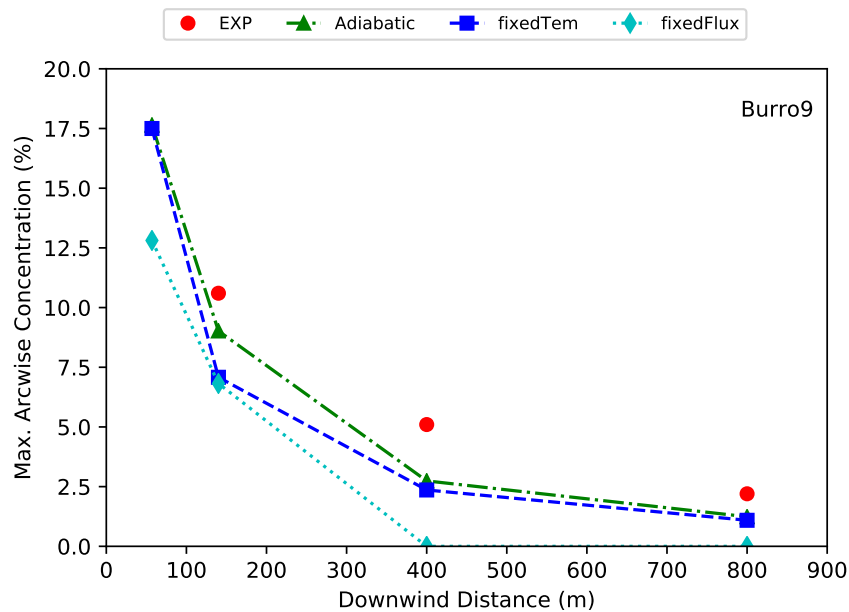


Fig. 5.10 Burro9 ground heat transfer study

Turbulence Schmidt number sensitivity study

Two value of $Sc_t = 1$ and $Sc_t = 0.3$ are used for studying the sensitivity of proposed model on predicting maximum gas concentration. Results are shown in Figure 5.11.

$Sc_t = 0.3$ used previously in wind tunnel dense gas dispersion are shown to be appropriate in accurate prediction of maximum gas concentration in 57 m array and 140 m array. Further downwind, at 400 m array and 800 m array, there is not much significant difference between these two values.

Peak concentration prediction

The comparison of OpenFOAM, FDS, and experimental results for Burro9 test is shown in Figure 5.12. FDS is over-predicted, while OpenFOAM is under-predicted. However, OpenFOAM is accurate in prediction at 800 m arc.

Maximum concentration at four arc-wise sensor arrays at 57 m, 140 m, 400 m and 800 m of other three Burro tests are further compared with experimental data to show the overall

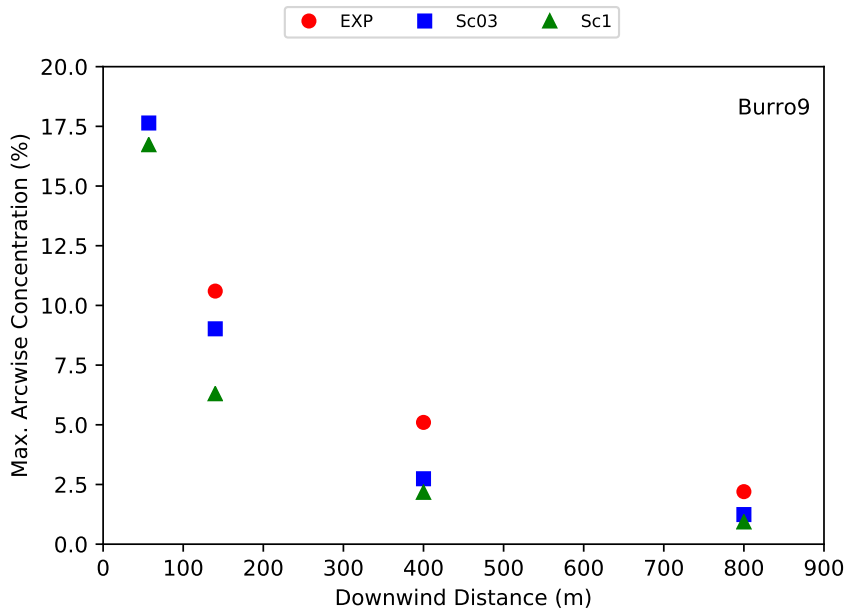


Fig. 5.11 Burro9 turbulent Schmidt number Sc_t study

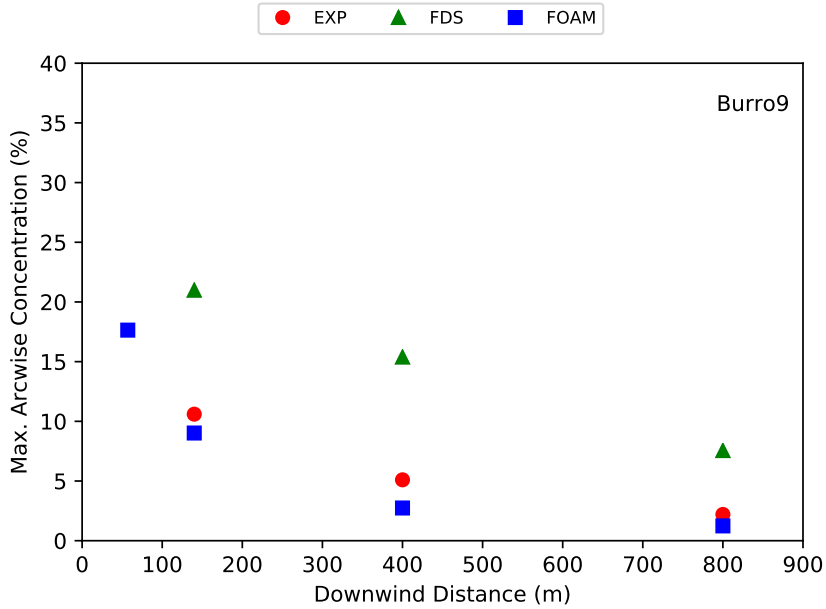


Fig. 5.12 Maximum arc-wise concentration Burro9

performance of FOAM in Figures 5.13, 5.14 and 5.15 respectively. Over-predictions are observed in all these simulations.

Figure 5.13 presents peak concentrations of 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.

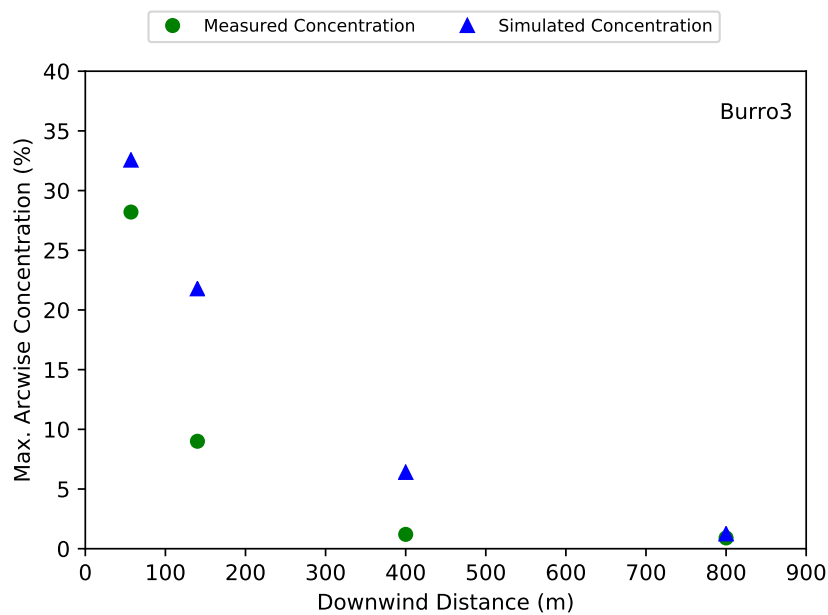


Fig. 5.13 Maximum arc-wise concentration Burro3

Under unstable to neutral ABL stability in Burro7, the over-prediction are shown in all arcs as in Figure 5.14. 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.

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 5.15.

Isosurface contour

The vertical isosurface contours at $X = 140$ are reported in Figure 5.16. Under-predicted cloud height are shown in all tests indicating that cloud buoyancy is not correctly solved.

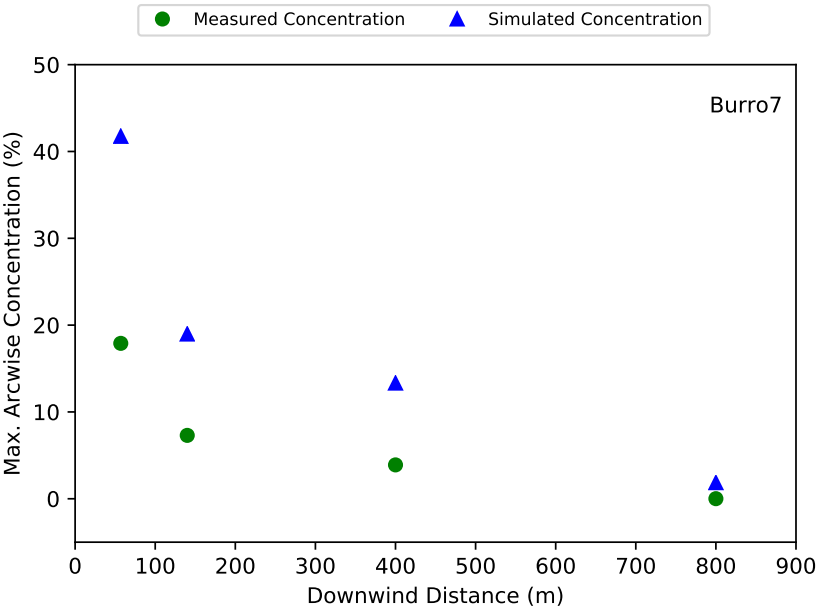


Fig. 5.14 Maximum arc-wise concentration Burro7

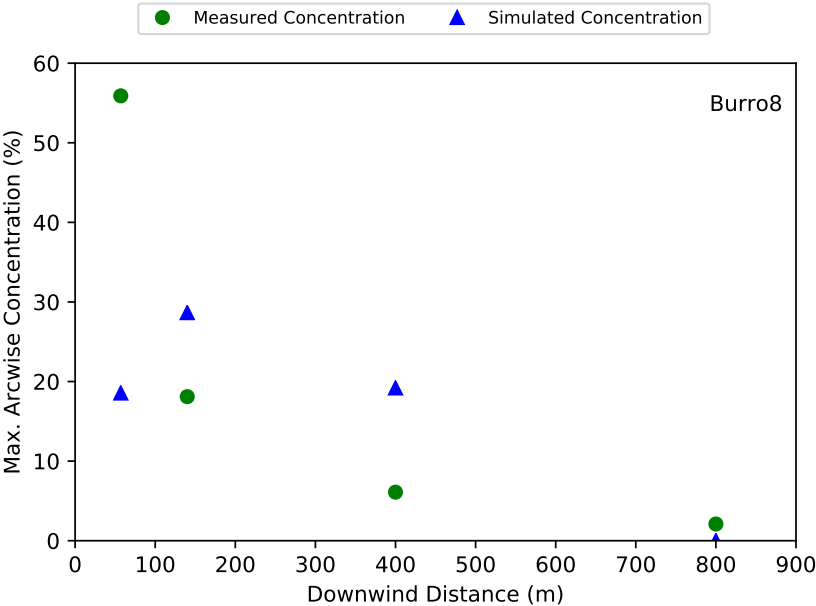


Fig. 5.15 Maximum arc-wise concentration Burro8

The sensitivity of the model on atmospheric stability can be seen in more flatten profile in Burro8 test which under stable condition.

Horizontal isosurface contours at height $Z = 1$ is shown in Figure 5.17. The gas concentration contour is plotted side by side with the contour analysing from experiment data, where the left is resulted from interpolating concentration at some concentration data points (presented in plots by dot black points), the right is from experimental data. Overall, the cloud widths are considerably well predicted. However some special flow structure, for example, bifurcating structure in low wind, stable ABL condition in Burro8 test is not well replicated. The bifurcated structure is shown in the simulation but less significant than that observed from experiment. Over-predicted downwind concentration are observed in all four tests. Besides, it can be seen that the gas move downwind faster. The high concentration can still be found very far from the source than the validation data.

Point-wise profiles

For further understanding the result, the point concentration from experiment will be compared to the simulated results. The first point is selected near the source, which is 57 m downwind and the second point is 140 m downwind.

Figure 5.18 is the plot of gas concentration at 1 m elevation at 140 m downwind of Burro9. It shows the good temporal trend of the simulation to the validation data. The concentration data is fairly matched except the peak duration (from 75 s to 100 s). The peak concentration is underestimated four times of the experimental data. The temporal trend however remains good. The concentration magnitude is really matching well with the validation data.

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 Burro3,

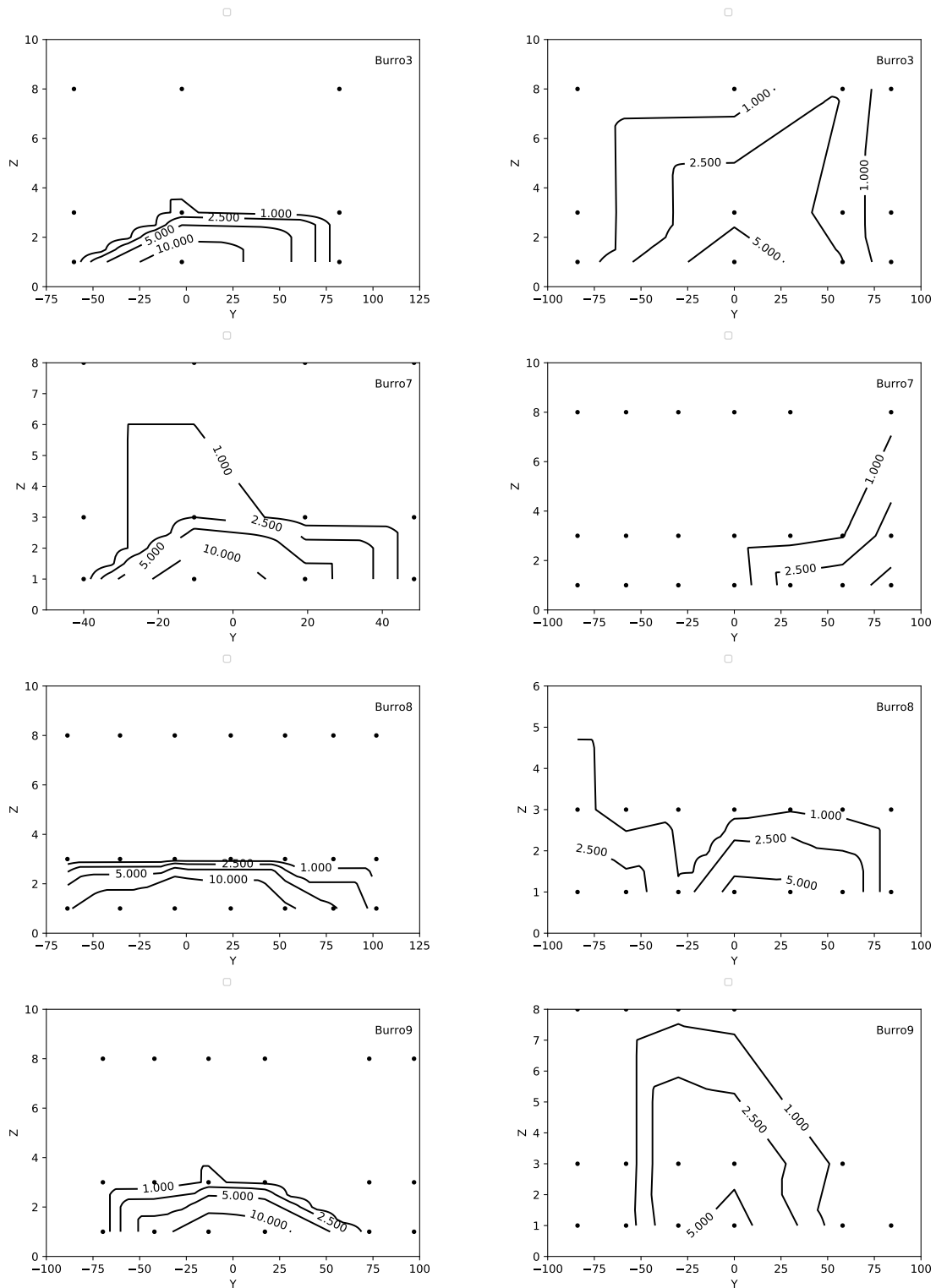


Fig. 5.16 Vertical isosurface at $X = 140$, Left: Simulation, Right: Experimental data, From top to bottom: Burro3, Burro7, Burro8 and Burro9

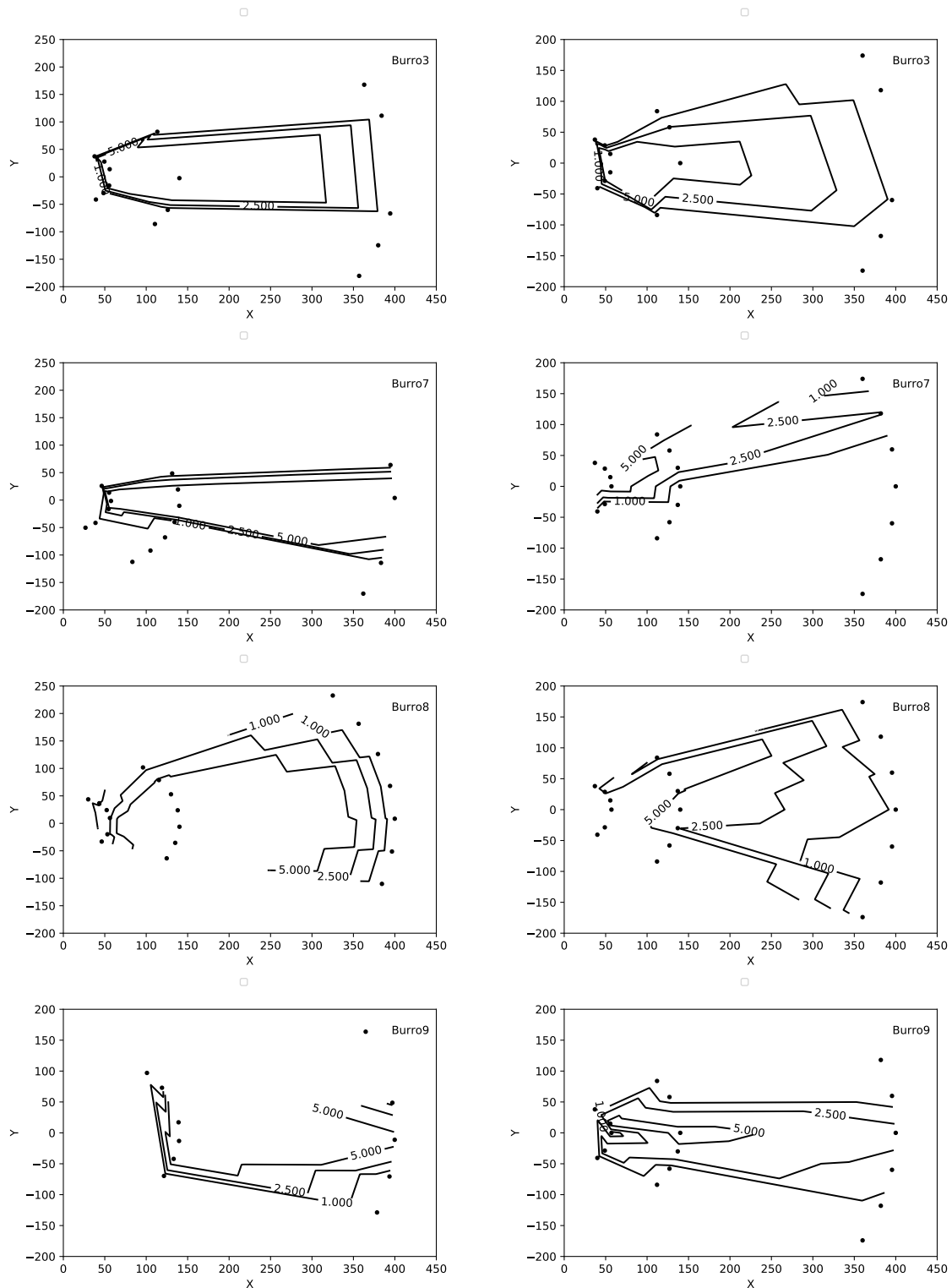


Fig. 5.17 Horizontal isosurface at $Z = 1$; Left: Simulation, Right: Experimental data; From top to bottom: (a) Burro3 (b) Burro7 (c) Burro8 (d) Burro9

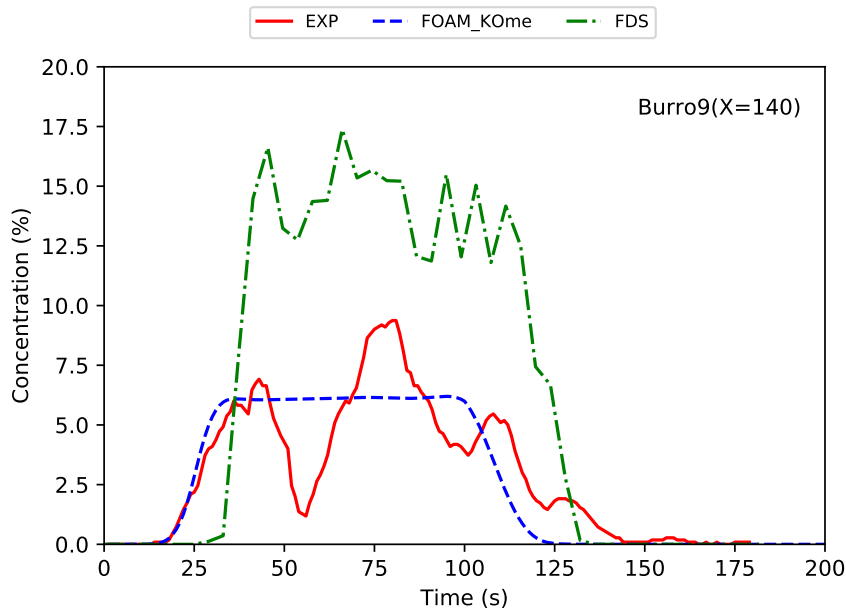


Fig. 5.18 Point concentration at 140 m of Burro9

Burro9 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 can not reproduce the peak concentration periods.

Statistical model evaluation

Overall statistical performance of OpenFOAM results are compared with FLACS with data extracted from [52] in Table 5.9. The predictions do not match all SMPs, however some important SPMs are in acceptable range. All gas concentration are within a factor of two ($FAC2=1$), better than FLACS ($FAC2 = 0.94$).

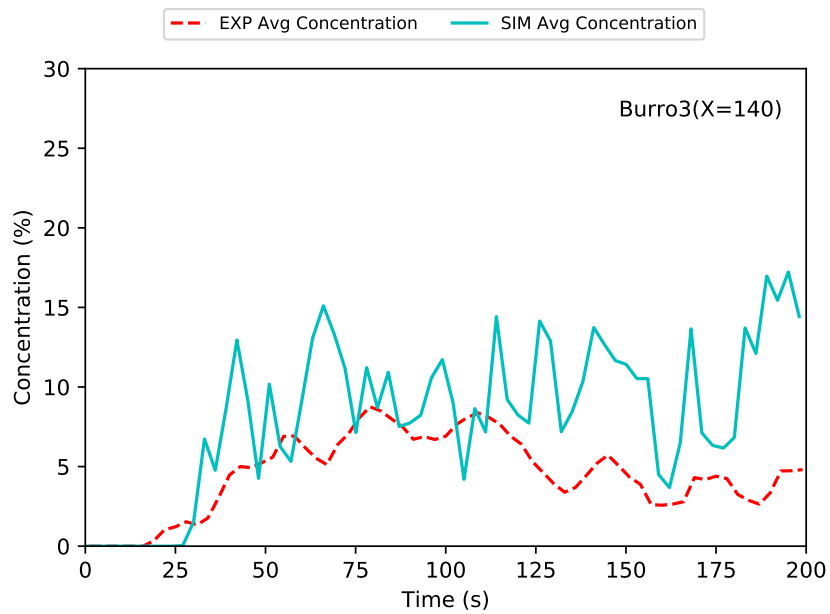


Fig. 5.19 Point concentration at 140 m of Burro3

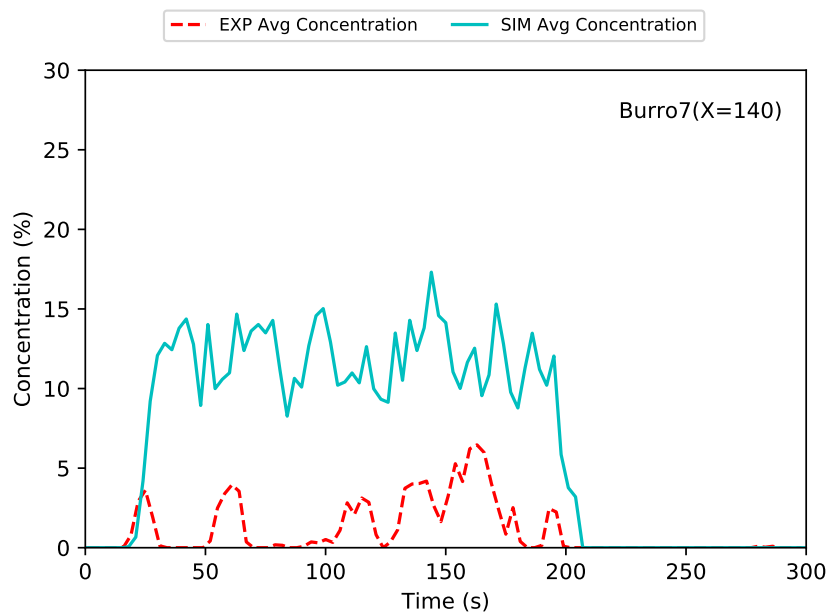


Fig. 5.20 Point concentration at 140 m of Burro7

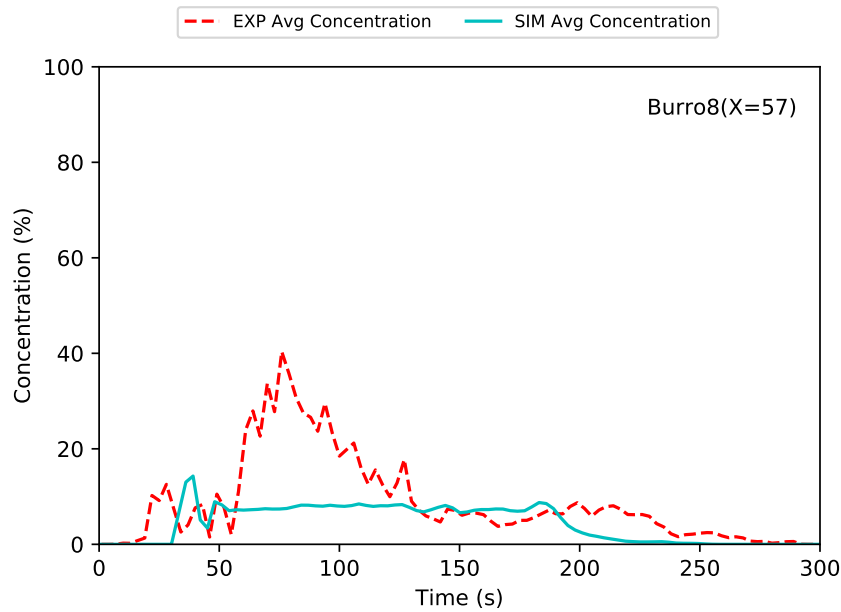


Fig. 5.21 Point concentration at 57 m of Burro8

Table 5.9 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 [52]	0.16	0.12	0.94	1.18	1.14
FOAM Burro9	0.44	0.23	1	0.63	1.28

5.3 Dense gas dispersion over obstructed terrain

Three Falcon tests included in MEP namely Falcon1, Falcon3 and Falcon4 are simulated using OpenFOAM to examine the effectiveness of proposed models in simulating field LNG dispersion in presence of obstacles. In all tests, the terrain was flat and the atmospheric stability condition varied as stable or neutral. The detail description of Falcon tests is introduced in Section 1.4.1.

5.3.1 Boundary conditions

Similarly to Burro tests simulation, Monin-Obukhov similarity theory is used to specify the wind velocity and temperature profile at the inlet. All required meteorological parameters are summarised in Table 5.10.

Table 5.10 Falcon tests meteorological parameters

	Falcon1	Falcon3	Falcon4
Stability	G	D	D-E
L_{MO}	4.96	-422	69.4
u_*	0.061	0.305	0.369
T_*	0.058	-0.018	0.152
z_0	0.008	0.008	0.008

The same approach as Section 5.2.2 is used to derive the vapour gas inlet. The LNG vaporization rate is assumed taking the value of $\dot{m} = 0.167 \text{ kg m}^{-2} \text{ s}^{-1}$. The spill diameter is derived from this vaporization rate, reported spill mass m and duration δt :

$$D = \sqrt{\frac{4m}{\pi \dot{m} \delta t}} \quad (5.7)$$

LNG spill variables used in simulation of Falcon tests are tabulated in Table 5.6.

Table 5.11 Falcon test spill conditions

	Falcon1	Falcon3	Falcon4
Vaporization rate ($\text{kg m}^{-2} \text{ s}^{-1}$)	0.167	0.167	0.167
Spill mass (kg)	28074	21435	18984
Spill duration (s)	131	154	301
Spill pool diameter (m)	19.5	16.0	10.8

5.3.2 Results and discussion of gas dispersion over obstructed terrain

Peak concentration prediction

Maximum concentration at four arc-wise sensor arrays at 50 m, 150 m and 250 m of the three Falcon tests are compared with experimental data. As seen in Figure 5.22, the prediction of maximum concentration are best for Falcon4 test.

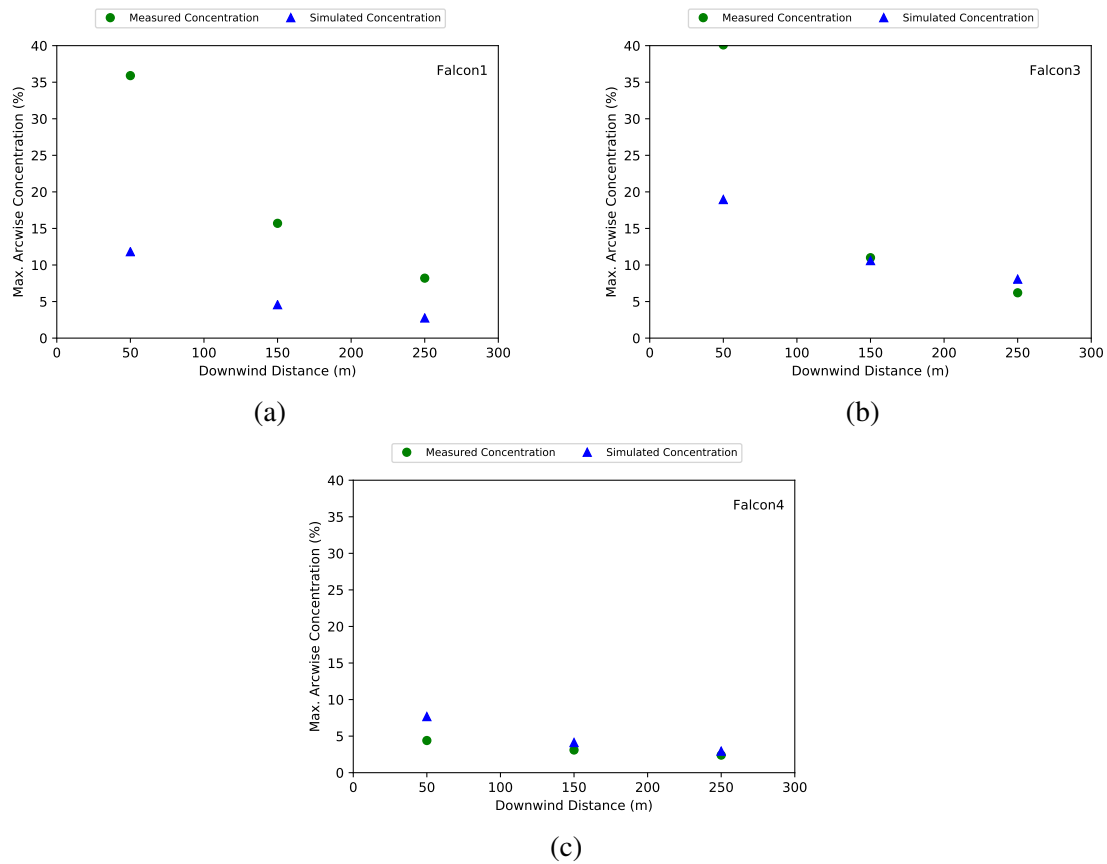


Fig. 5.22 Maximum arc-wise concentration (a) Falcon1 (b) Falcon3 (c) Falcon4

Statistical model evaluation

Statistical performance of OpenFOAM results are compared with FLACS in which data is extracted from [52]. These results are presented in Table 5.12. The proposed model simulation results fit all SPMs acceptable range for Falcon4 test. This shows the superior

performance of proposed code in comparison with FLACS when this software as it was unable to predict the peak gas within the factor of two ($FAC2 = 0$).

Table 5.12 Statistical performance measures of Falcon 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 [52]	1.35	1.88	0	5.56	23.65
FOAM Falcon4	-0.34	0.14	1	1.42	1.15

Chapter 6

Conclusion and future works

6.1 Conclusion

The atmospheric condition models including the velocity profile, turbulent kinetic energy and turbulent dissipation rate profiles by Monin-Obukhov similarity theory are used as boundary conditions to reproduce horizontal homogeneous atmospheric surface layer. In both neutral and thermal stratified ABLs, the profiles at outlet boundary are successfully maintained and consistent with inlet profiles. It shows the effectiveness of the proposed model in simulating horizontal homogeneous atmospheric surface layer. Furthermore, the proposed model can simulate different levels of ABL turbulence kinetic energy.

In the study of dense gas dispersion in neutral simulated ABL, a solver with variable turbulent Schmidt number is successfully validated in reproduced maximum gas concentration. SPM from simulation results are better than from the specified commercial software for gas dispersion FLACS.

In the study of LNG, a dense cold gas vapour dispersion in ABL, three ground heat transfer assumptions are simulated and compared with the validation data. The gas peak concentration is used as validation parameters. Adiabatic wall assuming zero heat flux from the ground to the gas cloud and fixed temperature model which assuming isothermal

ground (ground temperature is constant). The real heat flux to the gas cloud should be in between these two cases. The other model assuming a fixed flux of heat to the gas cloud is also included. Adiabatic wall gave the closest prediction of gas peak concentration. This may be explained as the case had the largest heat flux from ground to vapour cloud so it can compensate other source of heat addition to the cloud which is not considered in the simulation. The detail comparison with point-wise data showed that predictions at near-source point are rather conservative. The assumption of source term contributes largely in this error. The concentration peak in near source distance is quite conservative in comparison with validation data. The source modelling should be more well defined for better predictions. Statistical Performance Measures (SPMs) from the simulation results are compared with LES code FDS and specified dispersion code FLACS under Model Evaluation Protocol (MEP).

6.2 Contributions

Two equation turbulence models: standard $k - \varepsilon$ and SST $k - \omega$ turbulence models are modified substantially using OpenFOAM to simulate neutral and thermal stratified ABL turbulence. The proposed model can simulate different levels of ABL turbulence kinetic energy.

The dispersion model taking into account buoyancy effect, variable turbulence Schmidt number, ground heat transfer is developed under OpenFOAM platform. The model is shown to accurately reproduce peak concentration in simulated neutral ABL. In field experiments of LNG dispersion, the model is shown to be over-predicted the buoyancy effect. The further investigation should be done to accurately predict peak concentration in this case.

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, Large Eddy Simulation (LES) is indeed a promising approach. Nevertheless, LES requires a more intensive computational cost, especially for large scale presenting in atmospheric flow. Another consideration is that boundary conditions used in LES should be carefully applied.

In case of atmospheric variable buoyancy cold gas dispersion, e.g. LNG vapour dispersion in this study, 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 contact with cold flow should be also taken into account in determining the heat transfer from ground to the cloud.

